USING EFFICIENT SUPANOVA KERNEL FOR HEART DISEASE DIAGNOSIS

Proc. Proc. ANNIE06, Intelligent Engineering Systems Through Artificial Neural Networks, St. Louis, MO, November, 2006, vol. 16, ASME, New York, NY, pp. 305-310

BOLESLAW SZYMANSKI
Rensselaer Polytechnic Institute
Department of Computer Science

LONG HAN
Rensselaer Polytechnic Institute
Decision Sciences & Eng. Systems

MARK EMBRECHTS
Rensselaer Polytechnic Institute
Decision Sciences & Eng. Systems

ALEXANDER ROSS
Cardiomag Imaging, Inc.
Schenectady, NY

KARSTEN STERNICKEL
Cardiomag Imaging, Inc.
Schenectady, NY

LIJUAN ZHU
Rensselaer Polytechnic Institute
Department of Computer Science

ABSTRACT
Many machine learning methods focus on the quality of prediction results as their final purpose. Spline kernel based methods attempt to provide also transparency to the prediction identifying features that are important in the decision process. In this paper, we present a new heuristic for computing efficiently sparse kernel in SUPANOVA. We applied it to a benchmark Boston housing market dataset and to socially important problem of improving the detection of heart diseases in the population using a novel, non-invasive measurement of the heart activities based on magnetic field produced by the human heart. On this data, 83.7% predictions were correct, exceeding the results obtained using the standard Support Vector Machine and equivalent kernels. Equally good results were achieved by the spline kernel on a benchmark Boston housing market dataset.

INTRODUCTION

Kernel transformations are frequently used in machine learning methods to transform the input domain into a feature domain so that linear methods can be used to find an optimal solution to the learning problem. The most prominent examples of such methods are Support Vector Machines (SVM) [1] and Partial Least Square (PLS) approaches [2]. Despite their predictive power and efficient implementations, those and others machine learning techniques provide answers, but they do not give hints on what basis these answers were reached. To address this weakness, Gunn [3,4] proposed to use spline kernels and combined them with a full combinatorial decomposition of the feature set. These models explicitly identify feature subsets that are used in producing the answers. Those subsets can then be used to discern the reasons for predictions. The key element of this approach is a sparse solution to the fully decomposed spline kernel prediction function that can be used for hypothesis forming. However, SUPANOVA is not without its challenges, which arise when an efficient and scalable implementation is desired.

In this paper, we assume that there are N training data points in the form of vectors $\mathbf{x}^i=[x^i_1,x^i_2,\ldots,x^i_n]$, $i=1,2,\ldots,N$. Each vector represents values of n features and has the corresponding output value, y^i. We will denote the matrix containing these vectors (or training data points) as $\boldsymbol{X}$ and the vector of the corresponding output values as $\boldsymbol{y}$. We assume that the data are pre-processed using Mahalanobis scaling [5].

We want to find a function f that is represented by these data points and their values, as well for any new *testing* data point $y^i \sim f(\mathbf{x}^i)$. We use the following basic kernel model: $f(\mathbf{x}^i)=K(\boldsymbol{X},\mathbf{x}^i).a$, where a kernel function, $K(\mathbf{x}^j,\mathbf{x}^i)$, measures similarity between vectors $\mathbf{x}^j$ and $\mathbf{x}^i$.

SUPANOVA AND ITS IMPLEMENTATION

The SUPANOVA method represents the solution as a sum of kernels that decompose functions of the order n into a sum of terms that are *unitary, 2-ary,…,n-ary* order functions of the original arguments. Each function higher than first order uses a product of spline functions to represent its arguments. Hence, kernel function is replaced with a sum of kernels that measure similarity of argument vectors on a subset of features. Denoting $M=2^{n-1}$, we get:

$$f(x^i)=\sum_{j=0}^{M} c_j K_j(X,x^i).a \qquad c_j \geq 0. \tag{1}$$

The appropriate multivariate ANOVA kernel of two vectors is given by a tensor product of a univariate kernel plus a bias term. As an operator, we are using a spline kernel in the form of a piece-wise cubic polynomial: $k_{spline}(u,v) = uv + (u+v)\min(u,v)/2 - \min(u,v)^3/6$. Clearly, the number of potential terms in this representation is very large, but vector c should be very sparse.

The loss function for this kind of representation consists of three (and not two as is the case in traditional kernels) terms: (i) the error of modeling that measures the distance of the prediction from the real results (equal to traditional kernels) (ii) the smoothness of the representation that is dependent on metrics of vector a (again the same as in traditional kernels), and (iii) the sparseness of the representation that is specific to the spline kernels, and is defined ideally by the number of non-zeros in vector c. For the quadratic loss for the entire training data set, the error function is just the sum of the three defined above terms:

$$\Phi(a,c)=\left| y-\sum_{j=0}^{M} c_j \times K_j.a \right|_2^2 + \lambda_a \sum_{j=0}^{M} c_j a^T \times K_j.a + \lambda_c \sum_{j=0}^{M} \left| c_j \right|_0 \tag{2}$$

The sparseness and smoothness terms are weighted by regularization parameters λ_a and λ_c. They strike balance between the quality of the predictions on the training data and the model's ability to generalize to the testing or new data and need to be found iteratively. A solution consists of an initialization step followed by the single iteration:

Initialize: $c' = 1$ $a' = \mathrm{argmin}_a(\Phi(a,c'))$; **Single step**: $c^* = \mathrm{argmin}_c(\Phi(a',c))$.

In the initialization step, only λ_a needs to be determined, so we use an automatic search procedure to locate a local minimum of the validation error in 8-fold cross-validation runs. Then, vector ***a*** can be found by solving a system of linear equations resulting from taking the derivatives for all elements of this vector in the error function. To compute vector ***c,*** we set $\lambda_a = 0$ and select such λ_c that the loss is equal to the loss of the validation error in the initialization step. As the result, $\lambda_c = \frac{\lambda_a}{M}\sum_{j=0}^{M} a'^T \times K_j .a'$. To find optimal vector ***c***, we use a greedy selection of the non-zero entries, one-by-one, and measures the corresponding error of the solution until this error cannot be further decreased.

The results that we obtained from benchmarks confirmed that a vector selected in such a way is close to the vector that minimizes the loss function. In contrast, Gunn [3] uses the convex quadratic solver to solve an approximation of the cost function, in which zero norm of the third term in Eq. (2) is replaced by the first norm. Hence, we optimize approximately (by using greedy selection of the vector ***c*** non-zero elements) the exact loss function, whereas Gunn optimizes exactly (by using a quadratic problem solver) the approximate expression of the loss function. Another advantage of the heuristics is its memory efficiency. The size of the sparse kernel array is $O(MN^2)$, however, as shown later, our heuristic requires storing only one element for each column of the spline kernel. Hence, the size of the storage needed for our implementation is just $O(MN)$. The computational complexity of the entire computation is $O(MN^2)$, as this is the number of the spline kernel elements, each which needs to be computed.

PERFORMANCE MEASUREMENTS AND DATA BENCHMARKS

We use two metrics evaluate the prediction performance: the Root Mean Square Error index or RMSE, which is defined as the average value of the squared error and r^2, defined as the correlation coefficient squared between target values and predictions for the response. Two related metrics are used on test or validation data: q^2 and Q^2 are defined as $1-r^2$ and $1-R^2$, respectively. Ideally, both values should be similar and small (see [5] for details).

The model quality was measured on a machine learning benchmark from UCI Repository for Machine Learning Data-Bases [6]: the Boston housing market dataset. It has been extracted from a study of the effects of air pollution on housing prices. The data provides the median price for 1970 of owner-occupied houses in 506 census tracts within the Boston metropolitan area.
Thirteen features characterizing each census tract are available for use in predicting the median price, including crime rate, mean no. of rooms, distance to job center, etc. These data were used also by Gunn [3,4]. Our results from processing the Boston housing market dataset are shown in Figure 1. The concentration of points near the diagonal and low values of q^2 and Q^2 indicate high quality of the results. This dataset has a significant number of dimensions,

13, and therefore a relatively high number of feature combinations (2^{13}=8192). Yet, non-linearity of the data is limited, so the model that we ran included only empty feature set, single features and pairs of features, yielding 92 feature combinations in total. Sparseness of the solution is excellent in this case as only 40 elements of vector ***c*** were used in the solution. Our implementation selected only one single feature: mean number of rooms. It also selected the same binary combinations with mean number of rooms and with percentage built pre 1940 as the Gunn's implementation did (see [3]. These results demonstrate that our heuristic achieves a high accuracy of predictions while preserving the transparency of the solution supported by the original SUPANOVA implementation.

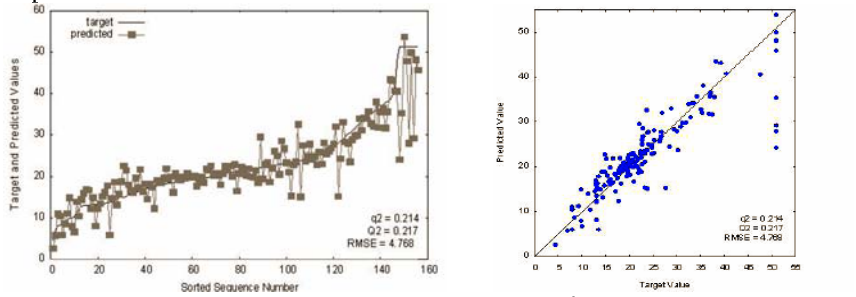

Figure 1. Boston housing market results with binary features.

To simplify the analysis of input variables, we introduced a parameter in the algorithm to control the maximum number of features in feature combinations considered in the solution. In general, it could be expected that the significant elements of vector ***c*** may correspond only to unitary, binary and tertiary combinations. For example, the results for Boston housing market dataset were obtained using at most binary combination of features. Adding tertiary feature combinations (see Fig. 2) did not make any obvious improvements in the predictions.

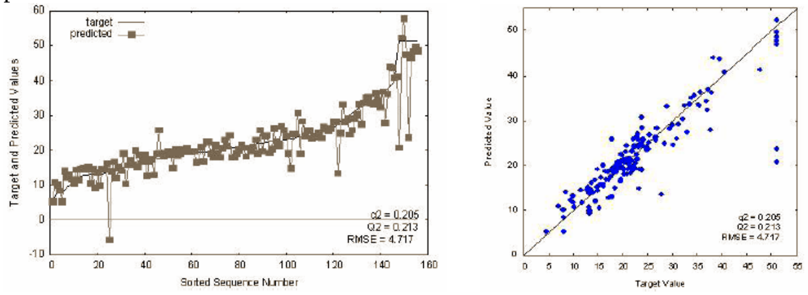

Figure 2. Boston housing market results with tertiary features.

Another interesting observation is that the values of the ***c*** vector elements mainly decrease in the order of their selection. For example, the values of the first 20 elements selected by the heuristic vary from 4.86 to 0.36, whereas values of the next 20 elements range from 0.11 to 0.29. This is not surprising, as our heuristic was designed to do exactly that, meaning that it selects the non-zero elements in the order of their importance for the solution. The fact that the order of the selection is very strongly correlated with the magnitude of the impact confirms that the heuristic works correctly, and in fact the difference between the order of selection and the order of magnitude of the value of non-zero elements is a measure of optimality of the heuristics.

The aim of Magnetocardiogram (MCG) based cardiology is to rapidly identify and localize the onset of heart disease from measuring the magnetic field of the heart. In this application we are interested in detecting ischemia, i.e., a common form of heart disease associated with dead tissues in the heart. A MCG device acquires data at 36 locations above the torso by making four sequential measurements in mutually adjacent positions. In each position the nine sensors measure the cardiac magnetic field for 90 seconds using a sampling rate of 1000 Hz leading to 36 individual time series. For diagnosis of ischemia, a bandwidth of 0.5 Hz to 20 Hz is needed, so a hardware low pass filter at 100 Hz using 6^{th}-order Bessel filter characteristics is applied, followed by an additional digital low pass filter at 20 Hz using the same characteristics, but a high order filter. The original MCG time domain data in the T-wave are further processed to yield 74 variables. Variables 1-36, 37-72, and 73-74 can be grouped together, with the first group representing the peak of the T-wave delays at different sensors, the second group, amplitude of this peak and the fourth group, the position of the peak on the torso. 325 patients sample data were available for the automated detection of ischemic heart disease. There are two response classes: negative and positive. The MCG data are normalized and 241 instances are randomly selected as training data, the remaining 84 samples are used as test data.

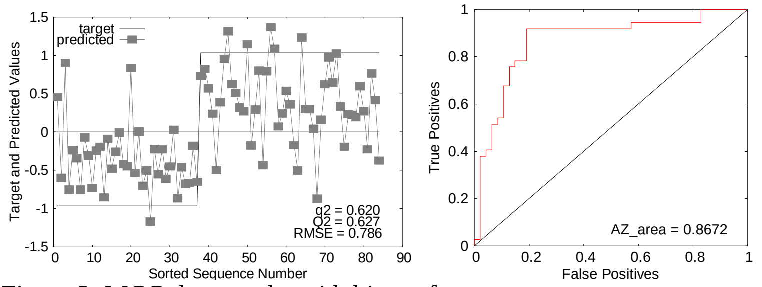

Figure 3. MCG data results with binary features.

The results are lower quality than in the case of Boston market housing dataset, but this is to be expected, as the MCG data are much more complex and noisy. The achieved error measure q^2 and Q^2, shown on the left side of Fig. 3, are close to the measurements obtained using KPLS based Machine Learning [5]. In case of ROC curve shown on the right part of Fig. 3, the area under the curve was larger than in the case of KPLS solution, indicating high quality of the predication for the MCG dataset. Like in the previous dataset, tertiary feature combinations did not improve the results, and for binary feature combination, the number of non-zero elements was very small. The predications were achieved with only 10 elements of vector ***c*** present, out of 2701 possible, so sparseness of 0.4% was achieved. All non-zero elements represented binary combination of features, indicating high non-linearity of data.

CONCLUSIONS

The quality of prediction of our implementation of SUPANOVA methods matched or exceeded other advanced machine learning techniques, such as KPLS kernels [5] in both tested cases. Since the magnitude of the entry in the vector ***c*** is correlated with its impact on the solution, we conjecture that the elements with small magnitude can be dropped without decreasing the quality of the approximation error but with improvement to the sparseness of the solution. Using this insight is important, because sparser the solution is the more transparent the model becomes, thereby increasing the value of spline kernel solutions to the users. Hence, the efficiency of computing sparse vector ***c*** can be increased by stopping heuristic when the non-zero elements become small, even if they still marginally decrease the overall error on training data. We will investigate this improvement in future work.

The heuristic is also capable of eliminating linearly or nearly linearly dependent feature combinations (including single features), only one of such features will be selected with a non-zero coefficient in vector ***c***, whereas the other (or others if there are more than one) would have their coefficients kept at zero, as the inclusion of any of them would not improve the solution.

ACKNOWLEDGEMENT

The presented material is based upon work supported by the National Science Foundation under Award Number 0349589. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or US Government.

REFERENCES

[1] V. Vapnik, *The Nature of Statistical Learning Theory}.*Springer, New York, 1995.
[2] S.Wold, M. Sjostrum, and L.Eriksson, ``PLS-Regression: a Basic Tool of Chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, pp. 109--130, 2001.
[3] S. Gunn and J. Kandola, ``Structural Modelling with Sparse Kernels," *Machine Learning*, vol. 48, no. 1, pp. 137--163, 2002.
[4] J. Kandola, ``Structural Modelling with Sparse Kernels," Department of Electronics and Computer Science, University of Southampton, U.K, 2001.
[5] M.Embrechts, B. Szymanski, and K. Sternickel, *Introduction to Scientific Data Mining.* Computationally Intelligent Hybrid Systems: The Fusion of Soft Computing and Hard Computing}. Wiley, New York, 2005.
[6] C. J. Merz and P. M. Murphy, ``UCI repository for machine learning data-bases," http://www.ics.uci.edu/~mlearn/MLRepository.html Department of Information and Computer Science. University of California, Irvine, 1996.