Handling Redundancy in Ensembles of Learned Models Using Principal Components

Christopher J. Merz and Michael J. Pazzani
Dept. of Information and Computer Science
University of California, Irvine, CA 92717-3425 U.S.A.
{cmerz,pazzani}@ics.uci.edu

Keywords: Combining multiple models, regression, principal components.

Abstract

When combining a set of learned models to form an improved estimator, the issue of redundancy in the set of models must be addressed. Existing methods for addressing this problem have failed to perform robustly, especially as the redundancy in the set of learned models increases. Recently, a variant of principal components regression, PCR*, demonstrated that these limitations could be overcome by mapping the original learned models to a set of principal components and then choosing which components to include in the final regression. Weights for the original learned models are then be derived from the weights of the principal components regression. The focus of this paper is to compare PCR*'s cross-validation-based stopping criteria for choosing the number of principal components to existing methods. Experimental results show that existing stopping criteria are often too conservative and discard useful components leading to poor performance.

Introduction

Combining a set of learned models[1] to improve classification and regression estimates has been an area of much research in machine learning and neural networks (for an extensive list of research, see (Merz and Pazzani 1996)). The challenge of this problem is to decide which models to rely on for prediction and how much weight to give each.

A good description of the problem of combining a set of learned models is given by (Perrone and Cooper 1992). Suppose two sets of data are given: a training set Train $= (x_m, y_m)$ and a test set Test $= (x_l, y_l)$. Now suppose Train is used to build a set of functions, $\mathcal{F} = f_i(x)$, each element of which approximates $f(x)$. The goal is to find the best approximation of $f(x)$ using $\mathcal{F}$.

In particular, we consider linear combinations of the elements of $\mathcal{F}$ to approximate $f(x)$, i.e.,

$$\hat{f}(x) = \alpha_0 + \sum_{i=1}^{N} \alpha_i f_i(x)$$

where α_i is the coefficient or weight of $f_i(x)$, α_0 is the optional constant term, and $N = |\mathcal{F}|$.

Previous research ((Perrone and Cooper 1992) and (Merz and Pazzani 1996)) has shown that existing approaches to finding these weights are highly sensitive to correlation or redundancy in the members of $\mathcal{F}$. To summarize and extract the "relevant" information from the learned models, variants of principal components regression (PCR) ((Draper and Smith 1981)) may be used. The main idea of PCR is to map the original learned models to a set of (independent) principal components where each component is a linear combination of the original learned models, and then to build a regression equation using some subset of the principal components to predict $f(x)$.

The advantage to this representation is that the components are sorted according to how much information (or variance) from the original learned models they contain. Given this representation, the goal is

[1] A learned model may be anything from a decision/regression tree to a neural network.

to choose the number of principal components to include in the final regression by retaining the first k which meet a preselected stopping criteria. The basic approach is summarized as follows:

1. Do a principal components analysis (PCA) on the learned models' performances on the training data (i.e., do a PCA on the matrix, M, where $M_{i,j}$ is the j-th model's reponse for the i-th training example) to produce a set of principal components, $PC = \{PC_1, ..., PC_N\}$.
2. Use a stopping criteria to decide on k, the number of principal components to use[2].
3. Do a least squares regression on the selected components (i.e., include PC_i for $i \leq k$).
4. Derive the weights, α_i, for the original learned models by expanding

   $$f_{PCR*} = \beta_1 PC_1 + ... + \beta_k PC_k$$

   according to

   $$PC_i = \gamma_{i,0} f_0 + ... + \gamma_{i,N} f_N,$$

   and simplifying for the coefficients of f_i. Note that $\gamma_{i,j}$ is the j-th coefficient of the i-th principal component.

The second step is very important because choosing too few or too many principal components may result in underfitting or overfitting, respectively. The stopping criteria used in (Merz and Pazzani 1996) (and summarized in the next section) is based on cross-validation. The goal of this research is to evaluate other methods for choosing the number of components to include in the regression and compare them to the method used in PCR*.

The next section summarizes the various stopping criteria investigated. Following that is a section on the experiment setup and results. A discussion of the results is then given followed by some concluding remarks.

Stopping Criterion Evaluated

Given a set of principal components derived from the learned models, $\mathcal{F}$, we now summarize the various stopping criteria for choosing the number of principal components to use in the final regression. The methods other than PCR* were selected from a list of commonly used methods described in a similar study by (Jackson 1993).

- **PCR***: The basic idea is to include successive principal components (i.e., ordered by the amount of variance they capture in the original learned models) in the regression estimate of $f(x)$ until all are used[3]. As each component is added, the error of regressing on the components included is estimated via 10-fold cross-validation on the training data. For example, a regression is done using all but one partition of the data, the error of that regression is esimated on the partition held out. This is repeated 9 more times to estimate the average error in regressing on that set of components. The subset of k components which results in the lowest estimated error is then used in finding the final weights. A detailed algorithm for PCR* is given in (Merz and Pazzani 1996).
- **Kaiser-Guttman**: This commonly used stopping rule is based on the average value of the eigenvalues. Only those components which exceed the average are retained.
- **Scree Test**: This rule is based on a plot of the eigenvalues. Smaller eigenvalues, representing random variation, tend to lie on a straight line. The significant components are taken to be those which lie above the line. In practice, this point is determined by computing the slopes of adjacent pairs of eigenvalues. The point where the biggest change in slope occurs is taken to be the last component worth retaining.
- **Broken Stick**: Plotting the eigenvalues of a principal components analysis of random data would resemble the broken stick distribution, i.e.,

  $$b_k = \sum_{i=k}^{p} \frac{1}{i},$$

  where p is the number of variables and b_k is the size of the eigenvalue for the k^{th} component. Those eigenvalues which exceed the values generated by the broken stick model are retained.
- **Proportion of total variance**: In this method all components are included up to some arbitrary proportion of the total variance. Experiments in this paper include components comprising 99.5% of the total variance.
- **Test of sphericity**: In this test, each component is evaluated to see whether it is significantly different from the collection of subsequent components. Those components which are significantly different from the latter components are retained. The test statistic is calculated as

  $$(p-k) ln[\sum_{i=k+1}^{p} \frac{\lambda_i}{(p-k)}] - \sum_{i=k+1}^{p} \lambda_i,$$

  where p is the number of variables, k represents a specific component, λ_i is the eigenvalue of component i, and n is the number of observations. If the

[2] It should be noted that in PCR, all of the principal components are used (i.e., $k = N$).

[3] Note that least squares regression using all p principal components (denoted PCR_p) is equivalent to standard linear regression on the original members of $\mathcal{F}$.

Table 0.1: Dataset Summary

Dataset	No. Cts/Discrete Attributes	No. of Examples	Source
auto-mpg	5/3	398	CMU
bodyfat	14/none	252	CMU
cpu	6/1	209	UCI
housing	12/1	506	CMU

resultant statistic is multiplied by $n-k$, the product is χ^2 distributed with $0.5*(p-k-1)(p-k+2)$ degrees of freedom.

Experimentation and Analysis

The set of learned models, $\mathcal{F}$, were generated using Backpropogation ((Rumelhart et al. 1986)). For a given domain, each network topology differs only in the intial random weights selected.

A summary of the domains used in the experiments is given in Table 0.1. In the Source column, UCI denotes datasets from the UCI data repository, and CMU denotes datasets taken from the Statistics Library at Carnegie Mellon University. All the datasets used involve the task of *numeric* prediction.

One experiment was conducted for each of the domains. The goal of each experiment was to illustrate how well each of the methods handles redundancy in the set of learned models.

There were 20 trials run for each of the datasets. On each trial the data was randomly divided into 70% training data and 30% test data. These trials were rerun for varying sizes of $\mathcal{F}$. As more models are included the linear dependence amongst them goes up showing how well the redundancy problem is handled[4].

Tables 2 thru 5 show the average error results for each of the datasets. Each row is a particular method and each column is the size of $\mathcal{F}$ for that run. PCR* is grouped with PCR_1 (i.e., principal component regression using just the first principal component) and PCR_p (i.e., principal components regression using all p principal components).

Abbreviations for the other tests are: Scree (SCREE), Kaiser Guttman (KG), Broken Stick (BS), Proportion of Variance (PVAR), and Sphericity (SPH).

The rows labelled BEM and GEM are taken from (Perrone and Cooper 1992) and correspond to the Basic Ensemble Method and Generalized Ensemble Method, respectively. BEM basically averages the members of $\mathcal{F}$ while GEM performs a constrained least squares regression (i.e., $\sum_{i=1}^{p} \alpha_i = 1$). Both methods are included here as a point of reference to other methods which do not use principal components.

[4] This is verified by observing the eigenvalues of the principal components and values in the covariance matrix of the models in $\mathcal{F}$

Table 0.2: Average error for *bodyfat* data.

Run	10	20	30	40	50	60
BEM	1.026	1.038	1.039	1.040	1.041	1.042
GEM	1.015	0.839	0.827	0.844	0.857	0.921
PCR_1	1.036	1.050	1.051	1.052	1.053	1.054
PCR_p	1.017	0.837	0.814	0.850	0.848	0.892
PCR*	0.986	0.832	0.800	0.830	0.786	0.806
KG	1.036	1.050	1.051	1.052	1.053	1.054
SCREE	1.062	1.048	1.052	1.052	1.059	1.067
BS	1.036	1.050	1.051	1.052	1.053	1.054
PVAR	1.036	1.050	1.051	1.052	1.053	1.054
SPH	1.036	1.050	1.051	1.052	1.053	1.054
DEF	7.239	7.239	7.239	7.239	7.239	7.239
PCR_{or}	0.964	0.786	0.737	0.696	0.688	0.671

The last two rows serve as a point of comparison. The row labeled DEFAULT reports the results for a combining method which always predicts the mean of the training data. One can think of this as linear regression with just the constant term. This is considered a lower bound on acceptable performance for PCR* (or any other method). The row labeled PCR_{Or} represents the oracle combiner which always chooses the best principle component at which to stop. This is considered an upper bound on how well PCR* is expected to do.

Table 0.2 shows that PCR* outperforms all other methods for the *bodyfat* data. All of these differences were statistically significant[5] except with respect to GEM which was only significantly worse for 60 models.

The results for the *housing* dataset are given in Table 0.3. For all sizes of $\mathcal{F}$, PCR* is significantly better than all other stopping rules (*Scree* for only 10 and 20 models) as well as BEM, and PCR_1.

In the *auto-mpg* domain results given in Table 0.4, PCR* is significantly better than BEM and all stopping rules (except *Scree*).

The results for the *cpu* dataset given in Table 0.5 demonstrate PCR*'s ability to stop early enough to avoid the pitfalls of stopping too late (i.e., PCR_p). For all sizes of $\mathcal{F}$, PCR* is significantly better than PCR_p (and GEM). Also, at 30 models it is significantly better than the *scree* test.

On all four datasets, the *Kaiser-Guttman*, *broken stick*, and *sphericity* tests always stopped at the first component. The *proportion of variance* test always stopped at the first or second component. The *Scree* test nearly always stopped at components 2-4 except on the *cpu* domain where its median was 7. However, for PCR* it depended on the domain and the size of

[5] A "statistically significant" difference is determined by a two-tailed paired t-tests with $p \leq .05$.

Table 0.3: Average error for *housing* data.

Run	10	20	30	40	50
BEM	2.788	2.779	2.775	2.773	2.771
GEM	2.721	2.564	2.598	2.572	2.567
PCR_1	2.782	2.771	2.768	2.766	2.764
PCR_p	2.719	2.563	2.597	2.568	2.567
PCR*	2.699	2.603	2.592	2.593	2.563
KG	2.782	2.771	2.768	2.766	2.764
SCREE	2.749	2.674	2.643	2.640	2.642
BS	2.782	2.771	2.768	2.766	2.764
PVAR	2.779	2.743	2.751	2.752	2.721
SPH	2.782	2.771	2.768	2.766	2.764
DEF	7.093	7.093	7.093	7.093	7.093
PCR_{or}	2.655	2.489	2.473	2.454	2.421

Table 0.4: Average error for *auto-mpg* data.

Run	10	20	30	40	50
BEM	2.197	2.191	2.189	2.189	2.189
GEM	2.164	2.140	2.113	2.140	2.208
PCR_1	2.197	2.190	2.188	2.188	2.188
PCR_p	2.162	2.135	2.108	2.133	2.200
PCR*	2.170	2.170	2.168	2.179	2.176
KG	2.197	2.190	2.188	2.188	2.188
SCREE	2.186	2.179	2.181	2.179	2.180
BS	2.197	2.190	2.188	2.188	2.188
PVAR	2.197	2.190	2.189	2.188	2.188
SPH	2.197	2.190	2.188	2.188	2.188
DEF	6.665	6.665	6.665	6.665	6.665
PCR_{or}	2.134	2.104	2.071	2.059	2.059

Table 0.5: Average error for *cpu* data.

Run	10	20	30	40
BEM	38.57	38.61	39.09	38.63
GEM	46.59	55.75	93.10	155.23
PCR_1	39.00	39.05	39.54	39.07
PCR_p	44.83	55.37	90.76	167.09
PCR*	40.26	40.45	39.47	40.05
KG	39.00	39.05	39.54	39.07
SCREE	41.01	40.59	41.83	40.41
BS	39.00	39.05	39.54	39.07
PVAR	39.01	39.10	39.50	39.00
SPH	39.00	39.05	39.54	39.07
DEF	97.85	97.85	98.81	97.85
PCR_{or}	35.63	34.46	34.15	32.63

$\mathcal{F}$ (which corresponds directly with the number of available components); in the *cpu* dataset it tended to stop at the early components, in the *auto-mpg* dataset it varied across the whole spectrum of components available, for the *housing* dataset it tended to stop in the latter half of the available components, and in the *bodyfat* dataset it frequently stopped in the latter one-fourth of the available components.

Discussion

Given that the typical pattern of eigenvalues for a set of learned models is to have more than 99% of the variance explained in the first principal component, it is not surprising that most of the stopping rules stop early. In particular, no other component could be above average for the *Kaiser-Guttman* criteria, it is easy to exceed the *proportion of variance* threshold, and the remaining components have so little variance they do not appear significant to the *broken stick* and *sphericity* tests. This leaves the *Scree* test which still tends to stop early. In short, none of these methods appears to be well suited for situations where a large amount of correlation/redundancy exists in the members of $\mathcal{F}$.

On the other hand, the basic trend for PCR* is that it is always near or ahead of the best method for each dataset. At times, as in the *cpu* and *auto-mpg* datasets, it performs like the more conservative methods (i.e., *Kaiser-Guttman*, *broken stick*, *proportion of variance*, *sphericity*, PCR_1, and BEM). In other domains such as *housing*, it tends to perform like the less constrained methods: PCR_p and GEM. Finally, in the *bodyfat* dataset it performs better than any other method.

Conclusion

In summary, PCR* has proved to be a robust method for choosing the number of principal components to use in principal components regression for the task of combining multiple models. The stopping criteria used outperforms a collection of existing criterion by using cross-validation to estimate the generalization performance expected from adding each principal component to the regression equation. By choosing the set of components with the best estimate of generalization performance, PCR* avoids including to few components and underfitting the data, or including to many components and overfitting the data.

References

Draper, N.R., Smith, H. (1981). *Applied Regression Analysis*. New York, NY: John Wiley and Sons.

Jackson, D.A. (1993). Stopping Rules in Principal Components Analysis: A comparison of heuristical and statistical approaches. *Ecology, 74(8)* 2204–2214. Ecological Society of America.

Leblanc, M., Tibshirani, R. (1993) Combining estimates in regression and classification Dept. of Statistics, University of Toronto, TR.

Merz, C.J., Pazzani, M. (1996) Handling Redundancy and Bias in Ensembles of Neural Networks Submitted to the special issue of *Connection Science* on combining neural networks.

Perrone, M. P., Cooper, L. N., (1993). When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. *Neural Networks for Speech and Image Processing*, edited by Mammone, R. J.. New York: Chapman and Hall.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Interior Representation by Error Propagation. *Parallel Distributed Processing, 1* 318–362. Cambridge, MASS.: MIT Press.

Wolpert, D. H. (1992). Stacked Generalization. *Neural Networks, 5*, 241–259.