

Application of Stacked Generalization to a Protein Localization Prediction Task

Melissa K. Carroll, M.S. and Sung-Hyuk Cha, Ph.D.

Pace University, School of Computer Science and Information Systems

Melissa K. Carroll c/o Sung-Hyuk Cha
Pace University School of CSIS
Goldstein Center
861 Bedford Road
Pleasantville, NY 10570
mcarroll@alumni.pace.edu
data mining, protein localization, stacked generalization

Abstract

A stacked generalization data mining approach was applied to a simplified version of the KDD Cup 2001 protein localization task. Four level-0 models were developed: an Artificial Neural Network, a Decision Tree, a Nearest Neighbor Classifier, and a Hybrid model that trained an Artificial Neural Network using those inputs selected as important by the Decision Tree. These models were developed from both a random sampling of the training dataset and a sampling designed to ensure an equal distribution of the localization variable. The predictions of these level-0 models were used to develop three level-1 generalizers: an Artificial Neural Network, a Decision Tree, and a Naïve Bayesian Classifier. The accuracy rates of the various models and generalizers suggest a modestly improved performance when using stacked generalization. Among the level-0 models, the Nearest Neighbor and the Hybrid performed slightly better than the Artificial Neural Network and Decision Tree. The performance of the three level-1 generalizers was roughly equivalent. Overall, both level-0 models and level-1 generalizers achieved better accuracy when equal distribution sampling was not performed.

Introduction

Data Mining, the application of machine learning algorithms to large databases, is commonly used to help solve problems in bioinformatics, such as prediction of protein localization from information about associated genes. Stacked Generalization (Wolpert, 1992) is a method sometimes used to combine data mining models. In this method, multiple models are developed from a set of training data, as in a typical data mining scenario. This dataset and these models are referred to

as level-0 data and level-0 models, respectively. A separate set of training data is also created, sometimes by leaving aside a portion of an original training dataset. A dataset is then created consisting of the predictions of the level-0 models for each instance in the second training set. This dataset is known as the level-1 data and contains one attribute for each level-0 model. It is important that the predictions in this dataset are made from instances that are not in the level-0 data used to train the level-0 models. A model is then developed from the level-1 data and is referred to as the level-1 generalizer. To make predictions using the generalizer, predictions for each instance in the test dataset are first made by each of the level-0 models. These predictions are then used to generate final predictions for the test instances, using the level-1 generalizer. Stacked Generalization has sometimes been found to perform better than level-0 models alone and than other methods for combining models, such as majority vote (Ting and Witten, 1997).

KDD Cup is an annual data mining competition sponsored by the ACM SIGKDD. The organizers provide training and test datasets. The submission with the highest accuracy rate on the test dataset wins. One task in 2001 was to predict protein localization from information about the anonymized genes of an organism, including class, phenotype, chromosome, essentiality, and interactions (Page, 2001).

Purpose

The purpose of this work was to develop four diverse types of level-0 models and three diverse types of level-1 generalizers from a simplified version of the KDD Cup 2001 dataset and to compare the performance of all the models and generalizers. The first three level-0 models were an Artificial Neural Network (ANN), a Decision Tree, and a Nearest Neighbor Classifier (NN). Since ANNs often have difficulty learning in the presence of a large number of inputs, Decision Trees are sometimes used as feature selectors. Thus, a fourth level-0 model was created by training an ANN on only the input variables selected by the Decision Tree model as important. The three level-1 generalizers consisted

of an ANN, a Decision Tree, and a Naïve Bayesian Classifier.

In addition, due to the nature of the dataset, two sub-approaches were taken. Unequally distributed target variables can sometimes cause difficulty in training models such as Artificial Neural Networks. In the simplified training dataset, the target variable, localization, was unequally distributed. Thus, while one approach created the level-0 data from a random sampling of the simplified training dataset, a second approach sampled from the training dataset to ensure an equal distribution of the localization variable. These approaches were also compared.

Methods

This work focused on a subset of the KDD Cup dataset in which those variables pertaining to interactions between genes were omitted. Correspondingly, it was decided that in order to allow for more robust model development, this task would make predictions for only those instances whose target was among the 3 most common localizations. In both the KDD Cup training and test datasets, these three values were nucleus, cytoplasm, and mitochondria. Together, instances with localizations selected from these three values made up 72.74% of the KDD Cup training dataset (627 instances) and 71.99% of the test dataset (275 instances). Table 1 indicates the relative distribution of the target variables in both resulting datasets.

Table 1

Localization	Training N (Percentage)	Test N (Percentage)
Nucleus	366 (58.4%)	174 (63.3%)
Cytoplasm	192 (30.6%)	66 (24.0%)
Mitochondria	69 (11.0%)	35 (12.7%)

The stacked generalization approach decided upon was to partition the resulting training dataset into a training set of 314 instances and a validation set of 313 instances. The new training set would be used as the level-0 data and the validation set would be used to generate the predictions in the level-1 data. These sets were created by randomly assigning instances to either dataset.

From the 444 input variables not initially excluded from the original training set of 627 instances, variables were subsequently excluded from the training set of 314 instances that were effectively unary, meaning all instances had the same value for that attribute. 274 such variables were excluded, leaving 169 binary input variables and 1 categorical input variable (chromosome). No variables contained missing values.

Due to the skewed distribution of the target variable, another training set was created from the set of 314 in which each localization value occurred with equal frequency. To accomplish this goal, all instances

of the least frequent localization, mitochondria, were included. Since there were 35 such instances, 35 instances for each of nucleus and cytoplasm were then randomly sampled from among those instances having the respective value as the target localization, resulting in an equally distributed set of 105 instances. The same 170 variables were used and a weight variable was also included in the dataset to provide information to the models about the frequency of the class in the unequally distributed dataset.

Level-0 Models

Artificial Neural Network (ANN): A multilayer perceptron (fully-connected feedforward network) was created that used linear combination functions in the hidden and output layers, a sigmoid activation function in the hidden layer, and a Multiple Bernoulli error function. One input node was created for each binary input variable. Chromosome was converted to 15 dummy variables to capture the 16 categories and an input node was created for each dummy variable. The resulting network thus had 186 input nodes. Localization was also converted to dummy variables, representing nucleus and mitochondria, and two output nodes were used to represent them. Only one hidden unit was added to avoid overfitting. A bias node was connected to all non-input nodes. The weights were randomly initialized to values between -0.5 and 0.5. The training of the network used Dual Quasi-Newton Optimization due to the moderate number of weights (191). Training was halted when the relative change of the network's misclassification rate between iterations was < 0.0001 , which occurred after 72 iterations for the unequally distributed dataset and 69 iterations for the equally distributed dataset.

Decision Tree: An algorithm based on CHAID was used with a splitting criterion of chi-square $p < 0.2$. The tree had the following properties: a minimum of 1 observations in a leaf, 3 observations for a split search, a maximum of 2 branches from a node, a maximum depth of 6, 5 splitting rules saved in each node, and 0 surrogate rules saved in each node. Model selection was based on the proportion of instances correctly classified.

Nearest Neighbor Classifier (NN): Each instance to be predicted was compared to each instance in the training set. The predicted localization was that of the instance in the training set that matched the instance to be predicted on the greatest number of attributes. A match was defined as exactly equal values for the attribute between the two instances, recalling that there were no missing values. In the case of ties for the number of matches, the localization from among the candidates that occurred most frequently in the

unequally distributed training set was used, even when training on the equally distributed training set.

Hybrid Tree/ANN: The Decision Tree model was trained as described above. Only those variables selected as important by the model, i.e. placed as nodes in the tree, were used as inputs to an ANN model developed using the same algorithm as described above. For both the equally and unequally distributed datasets, Levenberg-Marquardt Optimization was used due to the relatively small number of weights (16 and 11). The training process took 83 and 54 iterations for the unequally and equally distributed datasets, respectively. Table 2 lists the variables used in both Hybrid models, in order of importance.

Table 2

	Unequally Distributed	Equally Distributed
1	essential	essential
2	phenotype__auxotrophies__carbon	phenotype__auxotrophies__carbon
3	complex_translation__complexes	complex_transcription__complexes
4	complex_transcription__complexes	complex_mitochondrial__translocas
5	class_transcription__factors	complex_translation__complexes
6	complex_mitochondrial__translocas	
7	phenotype_nucleic__acid__metabolis	
8	phenotype_cell_cycle__defects	
9	class_gtpase__activating__proteins	
10	class_protein__phosphatases	

Level-1 Generalizers

ANN: This generalizer was developed in an identical manner to the level-0 ANN model. 8 input nodes represented the 4 3-level input variables. 2 output nodes, 1 hidden node, and a bias were included. Levenberg-Marquardt Optimization was used due to the small number of weights (13 and 12). Training was completed after 31 and 100 iterations for the unequally distributed and equally distributed datasets, respectively.

Decision Tree: This generalizer was designed and trained identically to the level-0 tree model.

Naïve Bayesian Classifier: Bayes Rule was applied to the level-1 data attributes to calculate likelihoods of each localization value for each combination of attributes. For each combination, the value with the highest likelihood was chosen as the predicted value for that attribute set.

Results

Table 3 shows the accuracy rates of the level-0 models on both the validation set of 313 instances and the modified test dataset of 275 instances. The rates were calculated by dividing the total number of correct predictions by the total number of instances. One gene appeared in the test dataset twice, but the two instances were counted separately. Therefore, these figures are very slightly conservative because that instance was impossible to predict correctly both times.

Table 3

Approach	Dataset	ANN	Tree	NN	Hybrid
Unequally Distributed	Validation	65.8%	72.2%	73.8%	71.3%
	Test	64.7%	65.1%	70.6%	71.3%
Equally Distributed	Validation	62.9%	61.7%	64.9%	62.3%
	Test	66.9%	59.6%	65.1%	61.8%

Table 4 shows the accuracy rates of the level-1 generalizers on the test dataset of 275 instances. Once again, these rates are conservative due to the duplicate instance.

Table 4

Approach	Level-1 ANN	Level-1 Tree	Level-1 Bayesian
Unequally Distributed	71.27%	71.64%	72.00%
Equally Distributed	65.82%	67.64%	70.18%

As a reference point for judging the results, the highest accuracy rate achieved in the actual KDD competition was 71.1%. The 5 next most accurate submissions were between 68.5% and 70.6%. While the actual competition required making predictions from among many more possible values for the target variable, the actual competitors had more attributes to use as predictors, since their dataset included the interaction variables. Thus, it is believed that the best of these results are roughly comparable to well-performing KDD Cup results.

Certain statistical comparisons were performed based on both a priori and post-hoc hypotheses. These comparisons were made by performing chi-square tests on the success/failure ratios of the 2 models to be compared. A one-tailed probability of 95% or greater was taken to be significant.

The first tests compared level-0 models using only the unequally distributed approach and the predictions of the models for the test dataset. The performance of the Hybrid level-0 model was found not to be significantly better than the Tree model ($z = -1.56$; $p = 0.06$), but significantly better than the ANN model ($z = -1.65$; $p < 0.05$). The level-0 NN's accuracy was found not to be significantly better than the worst performing level-0 model, the ANN ($z = -1.46$; $p = 0.07$), however it should be noted that the NN did not perform as well on the test dataset as on the validation dataset.

The next tests compared the unequally distributed approach to the equally distributed approach by looking at the performance of certain level-0 models on the test dataset. The Hybrid model was found to be significantly better when using the unequally distributed approach ($z = -2.35$; $p < 0.01$). However, the NN did not perform significantly better when using the unequally distributed approach ($z = -1.37$; $p = 0.09$).

The final tests examined the use of stacked generalization. As Table 5 indicates, when using the unequally distributed approach, all of the level-1 generalizers performed significantly better than the level-0 ANN model. However, no level-1 generalizers performed significantly better than the level-0 NN. With the equally distributed approach, no level-1 generalizers significantly outperformed the level-0 ANN and thus neither the level-0 NN.

Table 5

Approach	Level-0 Model	Level-1 Generalizer	z	p
Unequally Distributed	ANN	Bayesian	-1.83	0.033
		ANN	-1.65	0.050
		Tree	-1.74	0.041
	NN	Bayesian	-0.38	0.353
		ANN	-0.19	0.426
		Tree	-0.28	0.389
Equally Distributed	ANN	Bayesian	-0.83	0.204
		ANN	0.27	0.607
		Tree	-0.18	0.428

Conclusions and Future Work

As hypothesized based on previous findings, Stacked Generalization generally produced more accurate predictors of test data than the level-0 models alone. However, the performance of the best level-0 models was not significantly worse than that of the level-1 generalizers. In fact, level-0 models outperformed the level-1 generalizers in several cases on the validation dataset. Various modifications to the Stacked Generalization implementation may improve its performance. The simple approach used here of partitioning the training dataset into training and validation sets could be replaced by the more common method of using sampling similarly to cross-validation. The partitioning approach may have resulted in too few instances. It may also have led to the elimination of too many variables since variables were more likely to be considered unary given the smaller training dataset. The cross-validation approach could also lead to better level-0 models, which could in turn lead to better level-1 generalizers. Applying cross-validation would also give a better estimate of model performance and would eliminate the need for less meaningful statistical comparisons. Some findings have suggested that using the posterior probabilities of level-0 models as the basis

of level-1 data is more effective than using the actual predicted values of the models (Ting and Witten, 1997). Doing so may produce better results for this task. Different level-1 generalizers could also be used, for instance more linear types. Finally, a level-2 generalizer could be built from these level-1 generalizers, as could level-3 generalizers, etc.

In comparing specific level-0 models, the Nearest Neighbor and Hybrid models were more accurate than the ANN and Tree alone on the test dataset, though not necessarily significantly so. In fact, the Tree outperformed the Hybrid Tree on the validation dataset. The differences may therefore be negligible. Better trained ANNs and Trees may actually have outperformed the other models. For instance, estimation of an appropriate number of hidden units based on the data's noise level led to the ANN containing only one hidden node, which may have been insufficient for finding nonlinear patterns.

The three level-1 models also performed quite comparably. Other research has suggested that linear models may work best as level-1 generalizers (Witten and Frank, 2000), so the Naïve Bayesian Classifier might have been expected to outperform the ANN and Tree, an hypothesis that was not clearly supported. One alteration that may improve the results of the Bayesian model would be to apply an A Priori-type search on the level-1 data to reject attribute conclusions without enough support before calculating the likelihoods.

The results generally indicate that, for this task, the unequally distributed dataset approach resulted in better performance than the equally distributed approach. One reason for this finding may simply be that the sample size in the equally distributed dataset was too small. The weight variable could also be omitted, since, in a protein localization prediction task, the prior probabilities cannot be known unless the localizations of all proteins in the organism from which the instances are compiled are known. It is also possible that this approach is simply less effective for this task.

Finally, a future project might involve applying the methods used here to the actual KDD Cup task.

References

- Page, D. (2001). *KDD Cup 2001*. Website located at <http://www.cs.wisc.edu/~dpage/kddcup2001/>.
- Ting, K.M., Witten, I.H. (1997). *Stacked generalization: when does it work?*. *Proc International Joint Conference on Artificial Intelligence*, Japan, 866-871.
- Witten, I.H., Frank, E. (2000). *Data Mining*. Morgan Kaufmann (San Francisco).
- Wolpert, D.H. (1992). *Stacked Generalization*. *Neural Networks*, 5:241-259.