Published in final edited form as:

Methods. 2016 January 15; 93: 92–102. doi:10.1016/j.ymeth.2015.08.016.

Predicting protein function and other biomedical characteristics with heterogeneous ensembles

Sean Whalen¹, Om Prakash Pandey², and Gaurav Pandey^{2,3}

Sean Whalen: shwhalen@gmail.com; Om Prakash Pandey: omprakash.pandey@mssm.edu; Gaurav Pandey: gaurav.pandey@mssm.edu

¹Gladstone Institutes, University of California, San Francisco, CA, USA

²Icahn Institute for Genomics and Multiscale Biology and Department of Genetics and Genomic Sciences, Icahn School of Medicine at Mount Sinai, New York, NY, USA

³Graduate School of Biomedical Sciences, Icahn School of Medicine at Mount Sinai, New York, NY, USA

Abstract

Prediction problems in biomedical sciences, including protein function prediction (PFP), are generally quite difficult. This is due in part to incomplete knowledge of the cellular phenomenon of interest, the appropriateness and data quality of the variables and measurements used for prediction, as well as a lack of consensus regarding the ideal predictor for specific problems. In such scenarios, a powerful approach to improving prediction performance is to construct heterogeneous ensemble predictors that combine the output of diverse individual predictors that capture complementary aspects of the problems and/or datasets. In this paper, we demonstrate the potential of such heterogeneous ensembles, derived from stacking and ensemble selection methods, for addressing PFP and other similar biomedical prediction problems. Deeper analysis of these results shows that the superior predictive ability of these methods, especially stacking, can be attributed to their attention to the following aspects of the ensemble learning process: (i) better balance of diversity and performance, (ii) more effective calibration of outputs and (iii) more robust incorporation of additional base predictors. Finally, to make the effective application of heterogeneous ensembles to large complex datasets (*big data*) feasible, we present *DataSink*, a distributed ensemble learning framework, and demonstrate its sound scalability using the examined datasets. DataSink is publicly available from <https://github.com/shwhalen/datasink>.

Keywords

Protein function prediction; Heterogeneous ensembles; Nested cross-validation; Diversity-performance tradeoff; Ensemble calibration; Distributed machine learning

Correspondence to: Gaurav Pandey, gaurav.pandey@mssm.edu.

Publisher's Disclaimer: This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

1. Introduction

Prediction problems in biomedical sciences, including protein function prediction (PFP) (Pandey et al. (2006); Sharan et al. (2007)), are generally quite difficult. This is due in part to incomplete knowledge of how the cellular phenomenon of interest is influenced by the variables and measurements used for prediction, as well as a lack of consensus regarding the ideal predictor for specific problems. Even from a data perspective, the frequent presence of extreme class imbalance, missing values, heterogeneous data sources of different scales, overlapping feature distributions, and measurement noise further complicate prediction. Indeed, given these challenges, several community-wide exercises, most notably CAFA (Radivojac et al. (2013)) and Mouse-Func (Pena-Castillo et al. (2008)), have been organized to assess the state of the art in PFP. The approaches used to participate in these exercises use a variety of biological information, such as the amino acid sequence and three-dimensional structure of proteins, as well as systems-level data like gene expression and protein-protein interactions. They also use a diverse array of prediction methodologies from machine learning, statistics, network theory and others. Although these exercises indicated general principles that can help enhance PFP, they were generally unable to determine the *best method* for this problem out of the participating approaches, partly due to the problems listed above.

In scenarios like these, a powerful approach to improving prediction performance is to construct ensemble predictors that combine the output of individual predictors (Rokach (2009); Seni and Elder (2010)). These predictors have been immensely successful in producing accurate predictions for many biomedical prediction tasks (Yang et al. (2010); Altmann et al. (2008); Liu et al. (2012); Khan et al. (2010); Pandey et al. (2010)), including protein function prediction (Valentini (2014); Yu et al. (2013); Guan et al. (2008)). The success of these methods is attributed to their ability to reinforce accurate predictions as well as correct errors across many diverse base predictors (Tumer and Ghosh (1996)). Diversity among the base predictors is key to ensemble performance: If there is complete consensus (no diversity) the ensemble cannot outperform the best base predictor, yet an ensemble lacking any consensus (highest diversity) is unlikely to perform well due to weak base predictors. Successful ensemble methods strike a balance between the diversity and accuracy of the ensemble (Kuncheva and Whitaker (2003); Dietterich (2000)).

A wide variety of methods have been proposed to create ensembles consisting of diverse base predictors that benefit from both their consensus and disagreement (Rokach (2009); Seni and Elder (2010)). Popular methods like bagging (Breiman (1996)), boosting (Schapire and Freund (2012)) and random forest (Breiman (2001)) generate this diversity by sampling from or assigning weights to training examples. However, they generally utilize a single type of base predictor to build the ensemble. Such *homogeneous* ensembles may not be the best choice for biomedical problems like PFP where the ideal base prediction method is unclear, as discussed earlier. A more potent approach in this scenario is to build ensembles from the predictions of a wide variety of *heterogeneous* base prediction methods, such as the predictions from a variety of PFP methods submitted to CAFA (Radivojac et al. (2013)).

Two commonly used heterogeneous ensemble methods include a form of meta-learning called *stacking* (Merz (1999); Wolpert (1992)), and the *ensemble selection* method (Caruana et al. (2004, 2006)). Stacking constructs a higher-level predictive model over the predictions of base predictors, while ensemble selection uses an iterative strategy to select base predictors for the ensemble while balancing diversity and performance. Due to their ability to utilize heterogeneous base predictors, these approaches have produced superior performance across several application domains (Altmann et al. (2008); Niculescu-Mizil et al. (2009)).

In this paper, we present a comparative analysis of several heterogeneous ensemble methods applied to large and diverse sets of base predictors. This comparison is carried out in the context of the prediction of protein function, as well as that of genetic interactions (Boucher and Jenna (2013); Pandey et al. (2010)), a problem intimately related to PFP. In addition to assessing the overall performance of these methods on these problems, we will investigate the following critical aspects of heterogeneous ensembles that have not been investigated before:

1. How these methods try to achieve the diversity-performance tradeoff inherent in ensemble learning.
2. The importance of base predictor calibration for effective ensemble performance.
3. Dependence of heterogeneous ensemble performance on the number and type of base predictors constituting them.

We explored the first two aspects in our previous work on this problem (Whalen and Pandey (2013)). Other than this work, there have been few, if any, such analyses of ensembles constructed from such a large set of diverse base predictors, and we expect our results will shed light on the inner dynamics of ensemble predictors, especially heterogeneous ones. These insights are expected to have wide applicability across diverse applications of ensemble learning beyond PFP.

Finally, we present DataSink, a scalable distributed implementation of the heterogeneous ensemble methods analyzed in this study (available at <https://github.com/shwhalen/datasink>). This implementation is built on the insight that efficient (nested) cross-validation is critical for robust ensemble learning, and utilizes several parallelization opportunities in this process to enhance scalability. DataSink is implemented in Python and uses Weka (Hall et al. (2009)) for base predictor learning (unless they are provided, such as in CAFA), and the pandas/scikit-learn analytics stack (Pedregosa et al. (2011); McKinney (2012)) for ensemble construction. We illustrate how DataSink can be used for PFP and other similar prediction problems, and through this illustration, we demonstrate its scalability capabilities and other salient features.

2. Materials and Methods

2.1. Problem Definitions and Datasets

To assess the potential of heterogeneous ensembles for enhancing PFP, we assess their performance on three PFP instances. We also evaluate their performance on the closely related problem of prediction of genetic interactions.

2.1.1. Protein Function Prediction—Gene expression data are a commonly used data source for predicting protein function, as the simultaneous measurement of gene expression across the entire genome enables effective inference of functional relationships and annotations (Pandey et al. (2006); Sharan et al. (2007)). Thus, for the PFP assessment, we use the gene expression compendium of Hughes et al. (2000) to predict the functions of roughly 4,000 baker's yeast (*S. cerevisiae*) genes. The three most abundant functional labels (GO terms) from the list of most biologically interesting and actionable Gene Ontology Biological Process terms compiled by Myers et al. (2006) are used in our evaluation. These labels are GO:0051252 (regulation of RNA metabolic process), GO:0006366 (transcription from RNA polymerase II promoter) and GO:0016192 (vesicle-mediated transport). We refer to these prediction problems as PF1, PF2 and PF3 respectively (details in Table 1). Note that we demonstrate on only these three large labels due to the substantial amount of computation needed even for a single label (detailed in subsequent sections), and hope to report much more extensive results in future work. We expect the results presented here to be representative and the methodology to be applicable to other (functional) labels as well.

2.1.2. Genetic Interaction Prediction—Genetic interactions (GIs) are a category of cellular interactions that are inferred by comparing the effect of the simultaneous knockout of two genes with that of knocking them out individually (Hartman et al. (2001)). Since these interactions represent cases of functional buffering and interconnections, their knowledge is very useful for understanding cellular pathways and their interactions, and predicting gene function (Horn et al. (2011); Boucher and Jenna (2013); Kelley and Ideker (2005); Mohammadi et al. (2012); Michaut and Bader (2012)). However, despite their utility, and substantial progress in experimental mapping of GIs in model organisms (Horn et al. (2011); Costanzo et al. (2010)), there is a general paucity of GI data for several organisms important for biomedical research. To address this problem, some of us (Pandey et al. (2010)) and several others (Boucher and Jenna (2013)) have developed various computational approaches to predict GIs. In the current study, we wish to assess how heterogeneous ensembles can help advance this GI prediction effort. For this, we use the dataset developed in our previous work (Pandey et al. (2010)), which focuses on predicting GIs between genes from *S. cerevisiae* (baker's yeast) using features that denote functional relationships between gene pairs. Some such features include correlation between expression profiles, extent of co-evolution, and the presence or absence of physical interactions between their corresponding proteins (see Table 2 for an illustration). This particular experiment enables us to assess the predictive ability of heterogeneous ensembles on a problem that is related but complementary to PFP, and the scalability of our implementation DataSink, given the much larger size of this dataset (Table 1).

Finally, note that both the types of datasets we considered involve binary labels, so the results presented are in the classification context. However, the concepts and methods discussed apply to the general *prediction* scenario, such as multi-class and regression problems. Thus, we generally refer to methods/models used for addressing this class of problems as *predictors*.

2.2. Experimental Setup

Our rigorous experimental assessment of heterogeneous ensembles on the above PF and GI datasets was set up as shown in Figure 1. For these experiments, a total of 27 diverse base predictor types are trained using the statistical language R (R Core Team (2012)), primarily through the RWeka interface (Hornik et al. (2009)) to the data mining software Weka (Hall et al. (2009)). We expanded this set of base predictors using some R machine learning packages as well. Among these are homogeneous ensemble predictors based on boosting and random forest also. Base predictors are trained using 10-fold cross-validation (CV), where each training split is resampled with replacement 10 times, and then balanced using undersampling of the majority class. The latter is a standard and essential step to prevent learning decision boundaries biased to the majority class in the presence of extreme class imbalance such as ours (Table 1). See Table 3 for the details as well as the individual performance of these base predictors.

One of the most important components of our setup is the use of *nested cross-validation* (NestedCV) for the training (if needed) and testing of the heterogeneous ensemble methods being assessed (Figure 1). Under a standard k-fold CV procedure involving only one training and test split in each CV round, training of the ensembles involves the use of labels of the same training examples twice: first for the training of the base predictors, and then for the training of the heterogeneous ensemble over their predictions. This repeated use of the training labels is likely to lead to significant overfitting, and thus needs to be addressed. For this, we utilize a two-tier NestedCV procedure, in which the training split in each round of the outer CV is further broken into inner training and test splits for an additional round of cross-validation (Steps 1 and 2 of Figure 1). The base predictors are trained on the inner training split (Step 3) and these trained predictors are applied to the inner test split (Step 4) to generate predictions for the constituent examples (Step 5). The predictions so generated over all the inner CV rounds are collected for all the examples in the outer training CV split (Step 6) and then used to train the heterogeneous ensembles using the stacking and ensemble selection methods described below (Step 7). These ensembles are then applied to the outer CV test split (Step 8). The resultant predictions generated for the constituent examples are again collected (Step 9) and evaluated (step 10) to assess the overall predictive ability of the various ensemble methods considered. Note that in Step 7, the ensembles are trained on predictions generated from examples in the inner CV test splits, which were themselves not involved in the training of the base predictors (Step 3). Due to this separation of examples that the base predictors and ensembles are trained on, this NestedCV approach reduces the likelihood of overfitting while maintaining the fairness of the training and testing procedure.

The overall performance is measured by combining the predictions made on each outer CV test split from both the base predictors and the heterogeneous ensembles, and calculating the

corresponding area under the ROC curve (AUC score) (Fawcett (2006)). This assessment is also carried out using the F_{max} measure, which was shown to be a reliable metric in a recent large-scale protein function prediction assessment (Radivojac et al. (2013)). This measure is the maximum value of the F-measure across all the values of precision and recall at many thresholds applied to the prediction scores. Furthermore, to draw meaningful conclusions about predictive abilities of methods, it is critical to determine if the performance differences among methods are statistically significant. For this, we employ the standard methodology given by Demšar (2006) to test for statistically significant performance differences between multiple methods across multiple datasets. The Friedman test (Friedman (1937)) first determines if there are statistically significant differences between any pair of methods over all datasets, followed by a post-hoc Nemenyi test (Nemenyi (1963)) to calculate a p-value for each pair of methods. This is the non-parametric equivalent of ANOVA combined with a Tukey HSD post-hoc test, where the assumption of normally distributed values is removed by using rank transformations. As many of the assumptions of parametric tests are violated by machine learning algorithms, the Friedman/Nemenyi test is preferred despite reduced statistical power (Demšar (2006)).

2.3. Heterogeneous Ensemble Methods

The following heterogeneous ensemble methods are considered in this study.

2.3.1. Mean Aggregation—The predictions of each base predictor become columns in a matrix where rows are instances and the entry at row i , column j is the probability of instance i belonging to the positive class as predicted by predictor j . The unsupervised aggregation method operates by applying the mean across rows to produce an aggregate prediction probability for each example.

2.3.2. Stacking—Meta-learning is a general technique for improving the performance of multiple predictors by using the meta information they provide (Vilalta and Drissi (2002)). A common approach to meta-learning is stacked generalization (*stacking*) (Wolpert (1992)), that trains a higher-level (*level 1*) predictor on the outputs of base (*level 0*) predictors. Using the standard formulation of Ting and Witten (1999), we perform stacking with a level 1 logistic regression predictor trained on the probabilistic outputs of multiple heterogeneous level 0 predictors. Although more sophisticated predictors may be used, a simple logistic regression meta-predictor helps avoid overfitting which typically results in superior performance (Ting and Witten (1999)). In addition, its coefficients have an intuitive interpretation as the weighted importance of each level 0 predictor (Altmann et al. (2008)). The layer 1 predictor is trained and evaluated using the NestedCV-based setup described above.

Furthermore, in addition to stacking across *all* predictor outputs, we also evaluate an *aggregate* form of stacking. Here, the outputs of all 10 resampled (bagged) versions of each base predictor type, say SVM, are averaged and used as a single level 0 input to the meta learner. Intuitively, this combines outputs from predictors that have a similar nature, and thus similar expected performance and calibration. This allows stacking to focus on weights between (instead of within) predictor types.

2.3.3. Ensemble Selection—Ensemble selection is the process of choosing a subset of all available base predictors that perform well together, since including every base predictor in the ensemble may decrease performance. Testing all possible predictor combinations quickly becomes infeasible for ensembles of any practical size, and thus, heuristics are used to approximate the optimal subset. Remember that the performance of the ensemble can only improve upon that of the best base predictor if the ensemble has a sufficient pool of accurate and diverse predictors, and thus, successful selection methods must consider both these requirements.

We establish a baseline for this approach by performing simple *greedy* ensemble selection: sorting base predictors by their individual performance and iteratively adding the best currently unselected predictor to the ensemble. This approach disregards how well the latest predictor actually complements the current constituents of the ensemble.

Improving on this approach, Caruana et al.'s ensemble selection (CES) method (Caruana et al. (2004, 2006)) begins with an empty ensemble and iteratively adds new predictors that maximize its performance according to a chosen metric. At each iteration, a number of candidate predictors are randomly selected and the performance resulting from the addition of each candidate to the current ensemble is evaluated. The candidate resulting in the best ensemble performance is selected, the ensemble is updated with it, and the process repeats until a maximum ensemble size is reached. The evaluation of candidates according to their performance with the ensemble, instead of in isolation, improves the performance of CES over simple greedy selection.

Additional improvements employed by the native CES algorithm include 1) initializing the ensemble with the top n base predictors, and 2) allowing predictors to be added multiple times. The latter is particularly important as without replacement, the best predictors are added early and ensemble performance then decreases as worse performing predictors are forced into the ensemble. Replacement gives more weight to the best performing predictors while still allowing for diversity. We use an initial ensemble size of $n = 2$ to reduce the effect of multiple bagged versions of a single high performance predictor dominating the selection process, and (for completeness) evaluate all candidate predictors instead of sampling.

Ensemble selection, both greedy and CES, is performed (this approach's form of training) and evaluated using the NestedCV-based setup described above. Furthermore, to implement CES more efficiently, we sped up the evaluation of each candidate being considered for addition to the ensemble by aggregate the intermediate (inner CV) predictions using a cumulative moving average.

3. Results: Ensemble Performance

The performance (AUC score) of the heterogeneous ensemble methods described above on the PF and GI datasets is summarized in Table 4, and the corresponding statistical significance (p-value) of the comparative performance of each pair of methods is given in Table 5. Before discussing these results, we would like to point out that such biomedical prediction problems are generally quite difficult, and even small improvements in predictive

accuracy have the potential for large contributions to biomedical knowledge. For instance, consistent with interaction rates (0.5–1%) observed in other organisms, assume that 1 million GIs exist among the approximately 2×10^8 gene pairs in the human genome. Thus, even a 1% improvement in the accuracy of a prediction algorithm can lead to the discovery of 10,000 novel GIs, many of which may prove critical for biomedical applications like protein function prediction (Michaut and Bader (2012)) and therapeutic discovery for cancers (Boucher and Jenna (2013)). For instance, the computational discovery of genetic interactions between the human genes EGFR-IFIH1 and FKBP9L-MOSC2 potentially enables novel therapies for glioblastoma (Szczurek et al. (2013)), the most common and aggressive type of brain tumor in humans.

Several trends can be observed from the overall performance and statistical significance results in Tables 4 and 5 respectively. First, we find that *mean*, the highest performing among all the simple aggregation ensemble methods we tested, generally performed poorly even when compared to the best base predictor for all the datasets except PF2. This is because simple (unsupervised) mean may not be the best way to aggregate heterogeneous predictors that can have uncalibrated outputs with potentially different scales or notions of probability. We discuss the issue of calibration for ensembles in Section 4.2.

Supervised heterogeneous ensemble methods, namely stacking and ensemble selection, usually perform better than their counterparts, but with differences between the performance of the variants of these methods. First, the aggregated version of stacking, where bagged versions of base predictor types are aggregated before stacking, performs the best overall among all the prediction methods. Furthermore, this aggregated version significantly outperforms the stacking variant in which no such aggregation is done (Stacking (All); $p = 0.0025$), thus indicating that incorporating information about the inter-relationships among the base predictors leads to a more predictive ensemble. Similarly, among the ensemble selection methods, CES consistently outperforms the greedy method on all datasets, although not statistically significantly (Table 5). Greedy selection achieves best performance for ensemble sizes of 10, 14, 45, and 38 for GI, PF1, PF2, and PF3, respectively, while CES is optimal for sizes 70, 43, 34, and 56 respectively for the same datasets. Although the best performing ensembles for both selection methods are close in performance, simple greedy selection is much worse for non-optimal ensemble sizes than CES and its performance typically degrades after the best few base predictors are selected. Thus, on average CES is the superior selection method. Again, this is because CES selects a more diverse and mutually beneficial set of base predictors in the ensemble. See Section 4.1 for details of the above observations about ensemble selection.

Also importantly, the aggregated stacking and CES methods, which perform statistically comparably, significantly outperform the best base predictor for all the data sets ($p = 0.000136$ and 0.0019) respectively. Notably, the best base predictor for all the datasets are homogeneous ensembles based on boosting or random forest, thus validating our original hypothesis that building heterogeneous ensembles on top of homogeneous ones and other non-ensemble base predictors can be significantly advantageous. We also note that nested cross-validation, relative to a single validation set, improves the performance of both stacking and CES by increasing the amount of meta data available as well as the bagging

that occurs as a result. For instance, standard cross-validation that uses the same labels twice, once each for base classifier and ensemble learning, is only able to yield an AUC of 0.756, substantially lower than 0.812 yielded by NestedCV. More nested folds are expected to increase the quality of the meta data and thus affect the performance of these methods as well, although computation time increases substantially and motivates our selection of $k = 5$ nested folds. Due to these advantages of heterogeneous ensembles and nested cross-validation, each method we evaluated out-performed the previous state of the art AUC of 0.741 for GI prediction established in our previous work on this problem (Pandey et al. (2010)).

Finally, we also assessed the performance of all the methods in terms of the F_{max} measure (Radiwojac et al. (2013)). Note that the greedy selection and CES methods had to be re-run with F_{max} as the optimization criterion for this assessment, but the predictions from the other methods could be re-evaluated directly. The results in Table 6 show that most of the trends observed with AUC are valid here as well, especially that almost all the ensemble methods perform better than the base predictors for all the datasets. In particular, aggregated stacking again produces the best performance among all the methods for all the datasets. The only notable deviation in these results is that greedy selection performs slightly better than CES for the PF datasets. This is likely because F_{max} is focused only on the predictions made for examples of the positive class, and greedy selection might favor this mode of evaluation. Since AUC covers both the classes in its evaluation, we conducted the further analyses reported in this subsequent sections using this measure.

4. Results: Ensemble Characteristics

Since heterogeneous ensembles are generally learnt from a large number and variety of base predictors, the ensemble learning process can be quite dynamic and complicated. In the following subsections, we analyze several critical aspects of this process, namely the roles of ensemble diversity and calibration and the starting set of base predictors. There have been few, if any, such analyses of ensembles constructed from a large set of diverse base predictors. We expect the results will shed light on the inner dynamics of ensemble predictors, especially heterogeneous ones.

4.1. The Role of Diversity

As mentioned in the Introduction, the relationship between ensemble diversity and performance has immediate impact on the performance of ensemble methods, yet has not been formally proven despite extensive study (Kuncheva and Whitaker (2003); Tang et al. (2006)). We undertake the analysis of the relationship for heterogeneous ensembles here, for which we need a measure for the diversity of an ensemble. We measure diversity using Yule's Q -statistic (Yule (1900)) by first creating predicted labels from thresholded predictor probabilities, yielding a 1 for values greater than 0.5 (appropriate due to the balancing of the classes before training) and 0 otherwise. Given the predicted labels produced by each pair of predictors (base) D_i and D_k , we generate a contingency table counting how often each predictor produces the correct label in relation to the other:

	D_k correct (1)	D_k incorrect (0)
D_i correct (1)	N^{11}	N^{10}
D_i wrong (0)	N^{01}	N^{00}

The pairwise Q statistic is then defined as:

$$Q_{i,k} = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}. \quad (1)$$

This produces values tending towards 1 when D_i and D_k correctly classify the same instances, 0 when they do not, and -1 when they are negatively correlated. We evaluated additional diversity measures, such as Cohen (1960)'s κ -statistic, but found little practical difference between the measures (in agreement with Kuncheva and Whitaker (2003)) and focus on Q for its simplicity. Also, the raw Q values were transformed into their corresponding $1 - |Q|$ values, so that 0 represents no diversity and 1 represents maximum diversity for graphical clarity. Furthermore, for brevity, we analyze this relationship using GI and PF3 as representative datasets, although the trends observed generalize to PF1 and PF2 as well. Finally, multicore performance and diversity measures are implemented in C++ using the Rcpp package (Eddelbuettel and François (2011)). This proves essential for their practical use with large ensembles and nested cross validation.

Figure 2 plots ensemble diversity and performance for the GI and PF3 datasets as a function of the iteration number of the simple greedy selection and CES methods detailed in Section 2.3.3. Performance is measured in terms of AUC, while diversity is measured as the average pairwise diversity between the predictors included in the ensemble based on the Q statistic defined above. These plots reveal how CES (top curve) successfully exploits the tradeoff between diversity and performance while a purely greedy approach (bottom curve) actually decreases in performance over iterations after the best individual base predictors are added. This is shown via coloring, where CES shifts from red to yellow (better performance) as its diversity increases while greedy selection grows darker red (worse performance) as its diversity only slightly increases. Note that while greedy selection increases ensemble diversity around iteration 30 for PF3, overall performance continues to decrease. This demonstrates that diversity must be balanced with accuracy to create well-performing ensembles.

To illustrate in more detail using Figure 2(b), the first predictors chosen by CES (in order) are rf.1, rf.7, gbm.2, RBFClassifier.0, MultilayerPerceptron.9, and gbm.3, where numbers indicate bagged versions of a base predictor. RBFClassifier.0 is a low performance, high diversity predictor, while the others are the opposite (see Table 3 for a summary of base predictors). This ensemble shows how CES tends to repeatedly select base predictors that improve performance, then selects a more diverse and typically worse performing predictor. This manifests in the left part of the upper curve where diversity is low and then jumps to its first peak. After this, a random forest is added again to balance performance and diversity

drops until rising again at the next peak. The process repeats while the algorithm approaches a weighted equilibrium of better performing, lower diversity and worse performing, higher diversity predictors. This agrees with recent observations that diversity enforces a kind of regularization for ensembles (Tang et al. (2006); Li et al. (2012)): Performance stops increasing when there is no more diversity to extract from the pool of possible predictors.

Since ensemble selection and stacking are top performers and can both be interpreted as learning to weight/select different base predictors, we next compare the most heavily weighted predictors selected by CES (Weight^c) with the coefficients of a level 1 logistic regression meta-learner (Weight^m). We compute Weight^c as the normalized counts of predictors included in the ensemble, resulting in greater weight for predictors selected multiple times. These weights for PF3 are shown in Table 7.

Nearly the same predictors receive the most weight under both approaches (though logistic regression coefficients were not restricted to positive values, so we cannot directly compare weights between methods). The general trend of the relative weights explains the oscillations seen in Figure 2: Higher performance, lower diversity predictors are repeatedly paired with higher diversity, lower performing predictors to build an effective ensemble. A more complete picture of this selection emerges by examining the full list of candidate base predictors (Table 8), with the most weighted ensemble (CES and stacking) predictors shown in bold. The highest performing, lowest diversity GBM and RF predictors appear at the top of the list while VFI and IBk are near the bottom. Although there are more diverse predictors than VFI and IBk, they were not selected due to their lower performance and lower utility for the ensembles.

This analysis illustrates how diversity and performance are balanced during ensemble construction, and also gives new insight into the nature of ensemble selection and stacking due to the convergent weights of these seemingly different approaches. This insight opens avenues for new algorithms for heterogeneous ensemble learning that can balance diversity and performance in a structured and mathematically rigorous way. Lin et al. (2012) made a commendable beginning in this direction by proposing such an algorithm for enhancing the prediction of protein subcellular localization. However they applied these algorithms to a set of only about ten base predictors for this problem, and the true potential of these algorithms will be realized when they can be applied to the scale (number and type) of base predictors tested in our study.

4.2. The Role of Calibration

A key factor in the performance difference between stacking and CES is illustrated by stacking's selection of MultilayerPerceptron instead of RBF-Classifer for PF3 (Table 7). This difference in the relative weighting of predictors, or the exchange of one predictor for another in the final ensemble, persists across our datasets. We suggest this is due to the ability of the layer 1 predictor in stacking to learn a function on the probabilistic outputs of base predictors and compensate for potential differences in *calibration*, contributing to the superior performance of this method.

A binary predictor (classifier) is said to be *well-calibrated* if it is correct p percent of the time for predictions of confidence p (Bella et al. (2012)). However, accuracy and calibration are related but not the same: A binary predictor that flips a fair coin for a balanced dataset will be calibrated but not accurate. Relatedly, many well-performing predictors do not produce calibrated probabilities. For instance, an uncalibrated SVM outputs the distance of an instance from a hyperplane, which is not a true posterior probability of an instance belonging to a class, even if converted to one using established methods (Platt (1999)). Measures such as AUC are not sensitive to calibration for base predictors, and the effects of calibration on heterogeneous ensemble learning have only recently been studied (Bella et al. (2012)). This section further investigates this relationship.

Several methods exist for evaluating the calibration of probabilistic predictors. One such method, the *Brier score*, assesses how close (on average) a predictor's probabilistic output is to the correct binary label (Brier (1950)):

$$BS = \frac{1}{N} \sum_{i=1}^N (f_i - o_i)^2 \quad (2)$$

over all instances i . This is simply the mean squared error evaluated in the context of probabilistic binary classification. Lower scores indicate better calibration.

Figure 3 plots the Brier scores for each base predictor we used against its performance for the GI dataset, as well as the ensemble Brier scores for each iteration of CES and greedy selection. As expected, these plots show that predictors and ensembles with calibrated outputs generally perform better. Note in particular the calibration and performance of simple greedy selection, with initial iterations in the upper left of the panel showing high performing well-calibrated base predictors chosen for the ensemble, but moving to the lower right as sub-optimal predictors are forced into the ensemble. In contrast, CES starts with points in the lower right and moves to the upper left as both ensemble calibration and performance improve over iterations. The upper left of the CES plot suggests the benefit of additional predictors outweighs a slight loss in calibration during its final iterations.

Aggregated stacking produces a layer 1 predictor with approximately half the Brier score (0.083) of CES or the best base predictors. Since this approach learns a function over probabilities it is able to adjust to the different scales used by potentially ill-calibrated predictors in a heterogeneous ensemble. This explains the difference in the final weights assigned by stacking and CES to the base predictors in Table 7: Although the relative weights are mostly the same, logistic regression is able to correct for the lack of calibration across predictors and better incorporate the predictions of MultilayerPerceptron, whereas CES cannot. In this case, a calibrated MultilayerPerceptron serves to improve performance of the ensemble and thus stacking outperforms CES. Overall, these results show that better calibration leads to improved ensemble performance.

4.3. Dependence on the starting set of base predictors

The previous section reported the performance of heterogeneous ensembles learnt from a set of 27 base predictors that were chosen on the basis of availability of implementation and to

maximize coverage of the predictor space. A natural question to ask then is “how does the performance of the ensemble depend on the size of the initial set of base predictors”? To answer this question, we simulated the sequence in which a user may add base predictors to the set in an effort to enhance prediction performance. This simulation naturally starts with the best base predictor for the corresponding dataset, and then adds two randomly selected predictors to the initial set from Table 3 in each iteration, terminating when all of the 27 base predictors have been considered. In each iteration, heterogeneous ensembles are learnt using the mean, CES and aggregated stacking methods, and evaluated using the methodology shown in Figure 1. This whole process is repeated 20 times and the average and standard error of performance (AUC scores) recorded for each of these methods.

Figure 4 shows the average AUC scores (with standard error) obtained from the described simulation for the PF3 and GI datasets. In the case of PF3, the performance of both mean and CES is essentially statistically constant and indistinguishable across the range of base predictor types. In contrast, aggregated stacking is the only method that is able to consistently make use of the additional information from new initial base predictors to improve its performance. These trends become even clearer in the GI results, which offers many more examples than PF3 for reliable training and evaluation of ensembles. Here, simple mean essentially deteriorates with more base predictors due to its lack of consideration of synergies between the base predictors and potential adverse effects of calibration issues mentioned above. CES, which is better able to tackle these challenges, shows limited improvements due to its relatively ad hoc and non-deterministic nature. Finally, once again, aggregated stacking stands out as the method that is able to extract the most benefit from the growing set of base predictors. This is due to its ability to learn a deterministic predictive model on top of the base predictor outputs. This observation strongly supports the superior performance of this method in the overall evaluation (Table 4).

In summary, this section demonstrates the tradeoff between performance and diversity made by CES and examines its connection with stacking. There is significant overlap in the relative weights of the most important base predictors selected by both methods. From this set of predictors, stacking often assigns more weight to a particular predictor as compared to CES and this result holds across our datasets. We attribute the superior performance of stacking to this difference, originating from its ability to accommodate differences in predictor calibration that are likely to occur in large heterogeneous ensembles. This claim is substantiated by its significantly lower Brier score compared to CES, as well as the correlation between ensemble calibration and performance. Finally, yet another reason for stacking’s success in our evaluation is its superior ability to assimilate information from a growing set of base predictors. Overall, these results shed light on important issues related to the dynamics of ensemble predictors, especially heterogeneous ones, and we hope that this knowledge will lead to more effective heterogeneous ensemble methods.

5. DataSink: A distributed implementation of heterogeneous ensemble methods

The above results demonstrate the potential of heterogeneous ensembles to enhance our ability to address challenging biomedical prediction problems, such as protein function prediction. However, these results were generated using a prototype implementation that may not be easily usable by general users. Thus, to ease other researchers' and practitioners' use of these methods in their work, we have prepared a high-performance framework, named *DataSink*. This framework implements the pipeline shown in Figure 1 for learning from and applying such ensembles to potentially large datasets. *DataSink* is publicly available from <https://github.com/shwhalen/datasink> and is implemented in Python, which offers flexibility, add-on packages, robustness and ease of use and interpretation. This framework utilizes Weka (Hall et al. (2009)) for base predictor learning and application, and the pandas/scikit-learn analytics stack (McKinney (2012); Pedregosa et al. (2011)) for ensemble learning. Detailed step-by-step instructions for how to install *DataSink*'s requisites, as well as use the pipeline to learn and evaluate heterogeneous ensembles on (Weka-formatted) datasets are available at the above URL. PFP datasets, once Weka-formatted, can be processed using the same commands. Finally, *DataSink* also includes implementations of ensemble diversity measures in Cython (Behnel et al. (2011)), which are even faster than the prototype Rcpp implementations mentioned earlier.

DataSink is based on the insight that nested cross-validation (NestedCV) is critical for robust ensemble learning, as explained in Section 2.2, and thus making the NestedCV process more efficient can substantially improve the overall time requirements of ensemble learning. More specifically, it can be seen that the training and test prediction operations of both base predictors within each inner CV split and ensemble predictors within outer CV rounds are independent of other such splits, and thus can be easily parallelized. This parallelization can be easily accomplished across multiple CPUs of cluster systems and multiple cores of multi-core individual machines. Furthermore, the stacking and CES ensemble learning algorithms, which are difficult to parallelize as above, can be made more efficient by exploiting the multi-core architecture of individual machines. *DataSink* provides users options to have both the above optimizations with minimal changes to its configuration.

We evaluated the computational benefits of the multi-CPU and multi-core design of *DataSink* by running it (under default parameter settings) on the datasets used in this study. We chose to perform this experiment only with CES as it is the most time consuming of all the methods tested and is thus expected to show the maximum variation under different system configurations. Specifically, we ran *DataSink*'s CES implementation on the concerned datasets on Minerva, Mount Sinai's cluster system, with the number of CPUs available for this computation being incrementally doubled from 1 to 32. Figure 5 shows the time consumed in the two phases of the overall computation, namely NestedCV-based base predictor output *generation* and ensemble learning using *CES*, on the representative PF3 and GI datasets (observations on PF1 and PF2 were very similar). As expected the generation phase takes significantly more time than CES for both datasets. However, due to the much

higher level of parallelization in the generation phase, DataSink was able to achieve almost a double speedup for the smaller PF3 dataset with every doubling of the number of CPUs, especially when the number of CPUs is low. The speedups are about 1.6-fold for GI, due to the much heavier communication overhead for this substantially larger dataset. Similar speedups were obtained for the ensemble learning (CES) phase, although this tends to saturate beyond 16 processors due to the lower level of parallelization achievable for this phase. Overall, these results show that DataSink's distributed/parallel design is indeed able to make the (heterogeneous) ensemble learning process significantly more efficient, thus making the application of these methods to large complex datasets (*big data*) substantially more feasible.

6. Discussion

This paper proposed heterogeneous ensemble methods as viable approaches for addressing difficult biomedical prediction problems, including protein function prediction. These ensembles assimilate a large number and variety of base predictors and attempt to balance their mutual performance and diversity to produce more accurate predictions than the individual base predictors. Two established heterogeneous ensemble methods, namely stacking and ensemble selection, were rigorously evaluated on several large genomics datasets, with the goal being protein function or genetic interaction prediction. This evaluation showed that both these methods produced comparable and better performance than the other methods tested. In particular, the versions of these ensemble methods that took into account the mutual interrelationships among the base predictors, like similarity (aggregated stacking) or diversity (Caruana et al's ensemble selection), performed the best. Deeper analysis of this performance showed that the superior predictive ability of these methods, especially aggregated stacking, can be attributed to their attention to the following aspects of the collection of base predictors: (i) better balancing of diversity and performance, (ii) more effective calibration of outputs and (iii) more robust incorporation of additional base classifiers. Finally, to make the effective application of these methods to large complex datasets (*big data*) feasible, we presented *DataSink*, a distributed ensemble learning framework, and demonstrated its sound scalability using the examined datasets.

Our extensive study indicated several avenues for developing more effective heterogeneous ensemble learning methods and finding novel applications for them. The most revealing finding from this study was the dynamic nature of how these methods try to balance ensemble diversity and performance, although only indirectly. Given the criticality of this balance to the success of heterogeneous ensembles, developing novel algorithms that try to find the optimal balance directly and automatically can enhance the performance of such ensembles. Diversity measures, such as the Q statistic discussed earlier, might play an important role here. Approaches such as those proposed by Lin et al. (2012) can be an important step forward in this direction. On the implementation front, although our DataSink framework exploits several parallelization opportunities to make the heterogeneous ensemble learning process more efficient, there are potentially many more avenues to improve this efficiency further. Finally, our results demonstrate the potential of these ensembles for addressing difficult biomedical prediction problems. We believe the niche application areas for these ensembles are ones where a wide array of base predictors, whose

training/generation we can't control, are the most accessible form of knowledge for a given problem or dataset. One such area is that of crowdsourcing-based prediction competitions/assessments like those hosted by CAFA (Radivojac et al. (2013)), MouseFunc (Pena-Castillo et al. (2008)) and DREAM (Boutros et al. (2014)), as well as commercial/government platforms like Kaggle, InnoCentive and Challenge.gov. In these challenges, there is a natural diversity of expertise among the participants, as well as among the methods and auxiliary datasets used to generate predictions, besides several other sources of variation. This setting offers a great opportunity for learning heterogeneous ensembles to assimilate the knowledge embedded in these predictions and develop even better solutions to the target problems. In summary, there exist major opportunities for the development of novel heterogeneous ensemble learning methods and applying them in contexts that may not have been satisfactorily addressed so far.

Acknowledgments

We thank the Icahn Institute for Genomics and Multiscale Biology and the Minerva supercomputing team for their financial and technical support of this work. GP was partially supported by NIH grant# R01-GM114434.

References

- Altmann A, Rosen-Zvi M, Prosperi M, Aharoni E, Neuvirth H, Schülter E, Büch J, Struck D, Peres Y, Incardona F, Sönnernborg A, Kaiser R, Zazzi M, Lengauer T. Comparison of Classifier Fusion Methods for Predicting Response to Anti HIV-1 Therapy. *PLoS ONE*. 2008; 3(10):e3470. [PubMed: 18941628]
- Behnel S, Bradshaw R, Citro C, Dalcin L, Seljebotn DS, Smith K. Cython: The best of both worlds. *Computing in Science & Engineering*. 2011; 13(2):31–39.
- Bella A, Ferri C, Hernández-Orallo J, Ramírez-Quintana MJ. On the Effect of Calibration in Classifier Combination. *Applied Intelligence*. 2012:1–20.
- Boucher B, Jenna S. Genetic interaction networks: better understand to better predict. *Frontiers in Genetics*. 2013; 4(290)
- Boutros P, Margolin A, Stuart J, Califano A, Stolovitzky G. Toward better benchmarking: challenge-based methods assessment in cancer genomics. *Genome Biology*. 2014; 15(9):462. [PubMed: 25314947]
- Breiman L. Bagging Predictors. *Machine Learning*. 1996; 24(2):123–140.
- Breiman L. Random forests. *Machine learning*. 2001; 45(1):5–32.
- Brier GW. Verification of Forecasts Expressed in Terms of Probability. *Monthly Weather Review*. 1950; 78(1):1–3.
- Caruana, R.; Munson, A.; Niculescu-Mizil, A. Getting the Most Out of Ensemble Selection. *Proceedings of the 6th International Conference on Data Mining*; 2006. p. 828-833.
- Caruana, R.; Niculescu-Mizil, A.; Crew, G.; Ksikes, A. Ensemble Selection from Libraries of Models. *Proceedings of the 21st International Conference on Machine Learning*; 2004. p. 18-26.
- Cohen J. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*. 1960; 20(1):37–46.
- Costanzo M, Baryshnikova A, Bellay J, Kim Y, Spear ED, Sevier CS, Ding H, Koh JLY, Toufighi K, Mostafavi S, Prinz J, St Onge RP, VanderSluis B, Makhnevych T, Vizeacoumar FJ, Alizadeh S, Bahr S, Brost RL, Chen Y, Cokol M, Deshpande R, Li Z, Lin ZY, Liang W, Marback M, Paw J, San Luis BJ, Shuteriqi E, Tong AHY, van Dyk N, Wallace IM, Whitney JA, Weirauch MT, Zhong G, Zhu H, Houry WA, Brudno M, Ragibizadeh S, Papp B, Pál C, Roth FP, Giaever G, Nislow C, Troyanskaya OG, Bussey H, Bader GD, Gingras AC, Morris QD, Kim PM, Kaiser CA, Myers CL, Andrews BJ, Boone C. The Genetic Landscape of a Cell. *Science*. 2010; 327(5964):425–431. [PubMed: 20093466]

- Demšar J. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*. 2006; 7(Jan):1–30.
- Dietterich TG. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning*. 2000; 40(2):139–157.
- Eddelbuettel D, François R. Rcpp: Seamless R and C++ Integration. *Journal of Statistical Software*. 2011; 40(8):1–18.
- Fawcett T. An introduction to ROC analysis. *Pattern Recognition Letters*. 2006; 27(8):861–874.
- Friedman JH, Hastie T, Tibshirani RJ. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*. 2010; 33(1):1–22. [PubMed: 20808728]
- Friedman M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*. 1937; 32(200):675–701.
- Guan Y, Myers C, Hess D, Barutcuoglu Z, Caudy A, Troyanskaya O. Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biology*. 2008; 9(Suppl 1):S3. [PubMed: 18613947]
- Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*. 2009; 11(1):10–18.
- Hartman JL, Garvik B, Hartwell L. Principles for the Buffering of Genetic Variation. *Science*. 2001; 291(5506):1001–1004. [PubMed: 11232561]
- Hastie T, Tibshirani RJ, Narasimhan B, Chu G. Diagnosis of Multiple Cancer Types by Shrunk Centroids of Gene Expression. *Proceedings of the National Academy of Sciences of the United States of America*. 2002; 99(10):6567–6572. [PubMed: 12011421]
- Horn T, Sandmann T, Fischer B, Axelsson E, Huber W, Boutros M. Mapping of Signaling Networks Through Synthetic Genetic Interaction Analysis by RNAi. *Nature Methods*. 2011; 8(4):341–346. [PubMed: 21378980]
- Hornik K, Buchta C, Zeileis A. Open-Source Machine Learning: R Meets Weka. *Computational Statistics*. 2009; 24(2):225–232.
- Hothorn T, Bühlmann P, Kneib T, Schmid M, Hofner B. Model-Based Boosting 2.0. *Journal of Machine Learning Research*. 2010; 11(Aug):2109–2113.
- Hughes TR, Marton MJ, Jones AR, Roberts CJ, Stoughton R, Armour CD, Bennett HA, Coffey E, Dai H, He YD, Kidd MJ, King AM, Meyer MR, Slade D, Lum PY, Stepaniants SB, Shoemaker DD, Gachotte D, Chakraburty K, Simon J, Bard M, Friend SH. Functional Discovery via a Compendium of Expression Profiles. *Cell*. 2000; 102(1):109–126. [PubMed: 10929718]
- Kelley R, Ideker T. Systematic Interpretation of Genetic Interactions Using Protein Networks. *Nature Biotechnology*. 2005; 23(5):561–566.
- Khan A, Majid A, Choi TS. Predicting Protein Subcellular Location: Exploiting Amino Acid Based Sequence of Feature Spaces and Fusion of Diverse Classifiers. *Amino Acids*. 2010; 38(1):347–350. [PubMed: 19198979]
- Kuncheva LI, Whitaker CJ. Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. *Machine Learning*. 2003; 51(2):181–207.
- Li, N.; Yu, Y.; Zhou, Z-H. Diversity Regularized Ensemble Pruning. *Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases*; 2012. p. 330–345.
- Liaw A, Wiener M. Classification and Regression by randomForest. *R News*. 2002; 2(3):18–22.
- Lin JR, Mondal AM, Liu R, Hu J. Minimalist ensemble algorithms for genome-wide protein localization prediction. *BMC bioinformatics*. 2012; 13(1):157. [PubMed: 22759391]
- Liu M, Zhang D, Shen D. Ensemble Sparse Classification of Alzheimer’s Disease. *Neuroimage*. 2012; 60(2):1106–1116. [PubMed: 22270352]
- McKinney, W. Python for data analysis: Data wrangling with Pandas, NumPy, and IPython. O’Reilly Media, Inc; 2012.
- Merz CJ. Using Correspondence Analysis to Combine Classifiers. *Machine Learning*. 1999; 36(1–2): 33–58.

- Michaut M, Bader GD. Multiple genetic interaction experiments provide complementary information useful for gene function prediction. *PLoS Comput Biol*. 2012; 8(6):e1002559. [PubMed: 22737063]
- Mohammadi, S.; Kollias, G.; Grama, A. Role of synthetic genetic interactions in understanding functional interactions among pathways. *Pacific Symposium on Biocomputing*; 2012. p. 43-54.
- Myers CL, Barrett DR, Hibbs MA, Huttenhower C, Troyanskaya OG. Finding Function: Evaluation Methods for Functional Genomic Data. *BMC Genomics*. 2006; 7(187)
- Nemenyi, PB. PhD thesis. Princeton University; 1963. Distribution-Free Multiple Comparisons.
- Niculescu-Mizil A, Perlich C, Swirszcz G, Sindhvani V, Liu Y. Winning the KDD Cup Orange Challenge with Ensemble Selection. *Journal of Machine Learning Research Proceedings Track*. 2009; 7:23–24.
- Pandey, G.; Kumar, V.; Steinbach, M. Tech Rep 06–028. University of Minnesota; 2006. Computational Approaches for Protein Function Prediction: A Survey.
- Pandey G, Zhang B, Chang AN, Myers CL, Zhu J, Kumar V, Schadt EE. An Integrative Multi-Network and Multi-Classifer Approach to Predict Genetic Interactions. *PLoS Computational Biology*. 2010; 6(9):e1000928. [PubMed: 20838583]
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*. 2011; 12:2825–2830.
- Pena-Castillo L, Tasan M, Myers C, Lee H, Joshi T, Zhang C, Guan Y, Leone M, Pagnani A, Kim W, et al. A critical assessment of mus musculus gene function prediction using integrated genomic evidence. *Genome Biology*. 2008; 9(Suppl 1):S2. [PubMed: 18613946]
- Platt JC. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Advances in Large Margin Classifiers*. 1999; 10(3):61–74.
- R Core Team. R: A Language and Environment for Statistical Computing. 2012
- Radivojac P, Clark WT, Oron TR, Schnoes AM, Wittkop T, Sokolov A, Graim K, Funk C, Verspoor K, Ben-Hur A, et al. A large-scale evaluation of computational protein function prediction. *Nature methods*. 2013; 10:221–227. [PubMed: 23353650]
- Ridgeway G. gbm: Generalized Boosted Regression Models. 2013
- Rokach L. Ensemble-Based Classifiers. *Artificial Intelligence Review*. 2009; 33(1–2):1–39.
- Schapire, RE.; Freund, Y. *Boosting: Foundations and Algorithms*. MIT Press; 2012.
- Seni, G.; Elder, J. *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*. Morgan & Claypool; 2010.
- Sharan R, Ulitsky I, Shamir R. Network-based prediction of protein function. *Molecular systems biology*. 2007; 3(1)
- Szczurek E, Misra N, Vingron M. Synthetic Sickness or Lethality Points at Candidate Combination Therapy Targets in Glioblastoma. *International Journal of Cancer*. 2013; 133(9):2123–2132.
- Tang EK, Suganthan PN, Yao X. An Analysis of Diversity Measures. *Machine Learning*. 2006; 65(1): 247–271.
- Ting KM, Witten IH. Issues in Stacked Generalization. *Journal of Artificial Intelligence Research*. 1999; 10(1):271–289.
- Tumer K, Ghosh J. Error Correlation and Error Reduction in Ensemble Classifiers. *Connection Science*. 1996; 8(3–4):385–404.
- Valentini G. Hierarchical ensemble methods for protein function prediction. *International Scholarly Research Notices*. 2014; 2014:901419.
- Venables, WN.; Ripley, BD. *Modern Applied Statistics with S*. 4. Springer; 2002.
- Vilalta R, Drissi Y. A Perspective View and Survey of Meta-Learning. *Artificial Intelligence Review*. 2002; 18(2):77–95.
- Whalen, S.; Pandey, G. A Comparative Analysis of Ensemble Classifiers: Case Studies in Genomics. *Proceedings of the 13th IEEE International Conference on Data Mining*; 2013. p. 807-816.
- Wolpert DH. Stacked Generalization. *Neural Networks*. 1992; 5(2):241–259.
- Yang P, Yang YH, Zhou BB, Zomaya AY. A Review of Ensemble Methods in Bioinformatics. *Current Bioinformatics*. 2010; 5(4):296–308.

- Yu G, Rangwala H, Domeniconi C, Zhang G, Yu Z. Protein function prediction using multilabel ensemble classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. 2013; 10(4):1045–1057. [PubMed: 24334396]
- Yule GU. On the Association of Attributes in Statistics: With Illustrations from the Material of the Childhood Society. *Philosophical Transactions of the Royal Society of London, Series A*. 1900; 194:257–319.

- The potential of heterogeneous ensembles as prediction methods is studied.
- These methods are assessed for protein function prediction and other problems.
- Heterogeneous ensemble performance is explained in terms of their internal dynamics.
- DataSink, a scalable implementation of these methods, is presented for others' use.

Figure 1.

Nested cross-validation (NestedCV)-based pipeline used for training and evaluation of heterogeneous ensembles. The sequence followed is marked by step numbers. The pipeline is based on two-tier cross-validation (CV), outer and inner. The inner CV round is used for base predictor training and prediction generation on the corresponding training and test splits respectively. These predictions from all the base predictors from the outer CV training split are collected and used for training the heterogeneous ensemble methods tested. These trained ensemble models are then applied to the outer CV test split, which are eventually collected after all rounds of outer CV and evaluated.

Figure 2.

Ensemble diversity and performance as a function of iteration number for both greedy selection (bottom curve) and CES (top curve) on the GI and PF3 datasets. For GI, greedy selection fails to improve diversity and performance decreases with ensemble size, while CES successfully balances diversity and performance (shown by a shift in color from red to yellow as height increases) and reaches an equilibrium over time (iterations). For PF3, greedy selection manages to improve diversity around iteration 30 but accuracy decreases, demonstrating that just diverse or just accurate predictions alone are not enough for accurate ensembles.

Figure 3.

Performance as a function of calibration as measured by the Brier score Brier (1950) for base predictors (left), ensembles for each iteration of CES (middle), and ensembles for each strictly greedy iteration (right). Iterations for greedy selection move from the upper left to the lower right, while those for CES start in the lower right and move to the upper left. This shows better-calibrated (lower Brier score) predictors and ensembles have higher overall performance.

Figure 4. Variation of performance of heterogeneous ensemble methods with the number of base predictors the process starts with. Aggregated stacking most effectively and stably incorporates the information in additional base predictors to improve overall ensemble performance.

Figure 5.

Scalability of DataSink in terms of the time taken in the base predictor training and prediction *generation* (green bars) and heterogeneous ensemble (*CES*) learning (yellow bars) phases. *CES* is used as the representative heterogeneous ensemble method as it is the most time consuming of all the methods tested. Overall, DataSink enables close to double speedups, especially in the base prediction generation phase, every time the number of CPUs are doubled, thus validating its distributed efficiency.

Table 1

Details of genetic interaction (GI) and protein function (PF) datasets, including the number of features, number of examples in the minority (positive) and majority (negative) classes, and total number of examples.

Problem	#Features	#Positives	#Negatives	#Total
GI	152	9,994	125,509	135,503
PF1	300	382	3,597	3,979
PF2	300	344	3,635	3,979
PF3	300	327	3,652	3,979

Table 2

Feature matrix of genetic interactions, where n rows represent pairs of genes measured by features $F_1, \dots, F_m$ having label 1 if they are known to interact, 0 if they do not, and ? if their interaction has not yet been established.

Gene Pair	F_1	F_2	...	F_m	Interaction?
Pair ₁	0.5	0.1	...	0.7	1
Pair ₂	0.2	0.7	...	0.8	0
⋮	⋮	⋮	⋮	⋮	⋮
Pair _{n}	0.3	0.9	...	0.1	?

Table 3

Individual performance of 27 base predictors on the genetic interaction and protein function datasets. R packages include a citation for the method. For all others, refer to the Weka documentation. For boosting methods with selectable base learners, the default (normally a Decision Stump) is used.

Classifier	Performance			
	GI	PF1	PF2	PF3
Functions				
glmboost (Hothorn et al. (2010))	0.72	0.65	0.71	0.72
glmnet (Friedman et al. (2010))	0.73	0.63	0.71	0.73
Logistic	0.73	0.61	0.66	0.71
MultilayerPerceptron	0.74	0.64	0.71	0.74
multinom (Venables and Ripley (2002))	0.73	0.61	0.67	0.71
RBFClassifier	0.73	0.62	0.69	0.74
RBFNetwork	0.56	0.51	0.52	0.58
SGD	0.73	0.63	0.70	0.73
SimpleLogistic	0.73	0.65	0.72	0.73
SMO	0.73	0.64	0.70	0.73
SPegasos	0.66	0.52	0.56	0.56
VotedPerceptron	0.65	0.62	0.70	0.71
Trees				
AdaBoostM1	0.71	0.65	0.67	0.73
ADTree	0.73	0.64	0.67	0.75
gbm (Ridgeway (2013))	0.77	0.68	0.72	0.78
J48	0.75	0.60	0.65	0.71
LADTree	0.74	0.64	0.69	0.75
LMT	0.76	0.62	0.71	0.75
LogitBoost	0.73	0.65	0.69	0.75
MultiBoostAB	0.70	0.63	0.66	0.70
RandomTree	0.71	0.57	0.60	0.63
randomForest (Liaw and Wiener (2002))	0.79	0.67	0.72	0.76
Rule-Based				
JRip	0.76	0.63	0.67	0.70
PART	0.76	0.59	0.65	0.73
Other				
IBk	0.70	0.61	0.66	0.70
pam (Hastie et al. (2002))	0.71	0.62	0.66	0.64
	0.64	0.55	0.56	0.63

Table 4

Performance (AUC) of heterogeneous ensemble methods for protein function and genetic interaction datasets. The best performing base predictor (random forest for the GI dataset and boosting (GBM) for PFs) is given for reference.

Method	Performance (AUC)			
	GI	PF1	PF2	PF3
Best Base Classifier	0.79	0.68	0.72	0.78
Mean Aggregation	0.763	0.669	0.732	0.773
Greedy Selection	0.792	0.684	0.734	0.779
CES	0.802	0.686	0.741	0.785
Stacking (Aggregated)	0.812	0.687	0.742	0.788
Stacking (All)	0.809	0.684	0.726	0.773

Table 5

Pairwise performance comparison of multiple non-ensemble and ensemble methods across datasets. Only pairs with statistically significant differences, determined by Friedman/Nemenyi tests at $\alpha = 0.05$, are shown.

Method A	Method B	p-value
Best Base Classifier	CES	0.001902
Best Base Classifier	Stacking (Aggregated)	0.000136
CES	Mean Aggregation	0.000300
CES	Stacking (All)	0.029952
Greedy Selection	Mean Aggregation	0.029952
Greedy Selection	Stacking (Aggregated)	0.005364
Mean Aggregation	Stacking (Aggregated)	0.000022
Stacking (Aggregated)	Stacking (All)	0.002472

Table 6

Performance (F_{max}) of heterogeneous ensemble methods for protein function and genetic interaction datasets. The best performing base predictor (random forest for the GI dataset and boosting (GBM) for PFs) is given for reference.

Method	Performance (F_{max})			
	GI	PF1	PF2	PF3
Best Base Classifier	0.341	0.267	0.299	0.341
Mean Aggregation	0.340	0.276	0.334	0.37
Greedy Selection	0.362	0.282	0.326	0.371
CES	0.378	0.258	0.324	0.354
Stacking (Aggregated)	0.397	0.285	0.331	0.373
Stacking (All)	0.376	0.236	0.288	0.365

Table 7

The most-weighted predictors produced by stacking with logistic regression (Weight^m) and CES (Weight^c) for the PF3 dataset, along with their average pairwise diversity and performance.

Classifier	Weight ^m	Weight ^c	Diversity	AUC
rf	0.25	0.21	0.39	0.71
gbm	0.20	0.27	0.42	0.72
RBFClassifier	-	0.05	0.45	0.71
MultilayerPerceptron	0.09	-	0.46	0.70
SGD	0.09	0.04	0.47	0.69
VFI	0.11	0.11	0.71	0.66
IBk	0.09	0.13	0.72	0.68

Table 8

Candidate predictors sorted by mean pairwise diversity and performance. The most heavily weighted predictors for both CES and stacking are shown in bold. This trend, which holds across datasets, shows the pairing of higher-performance lower-diversity predictors with their complements, demonstrating how seemingly disparate approaches create a balance of diversity and performance.

Classifier	Diversity	AUC
rf	0.386	0.712
gbm	0.419	0.720
glmnet	0.450	0.694
glmboost	0.452	0.680
RBFClassifier	0.453	0.713
SimpleLogistic	0.459	0.689
MultilayerPerceptron	0.459	0.706
SMO	0.462	0.695
SGD	0.470	0.693
LMT	0.472	0.692
pam	0.525	0.673
LogitBoost	0.528	0.691
ADTree	0.539	0.682
VotedPerceptron	0.540	0.694
multinom	0.553	0.677
LADTree	0.568	0.678
Logistic	0.574	0.669
AdaBoostM1	0.584	0.684
PART	0.615	0.652
MultiBoostAB	0.615	0.679
J48	0.632	0.645
JRip	0.687	0.627
VFI	0.713	0.662
IBk	0.720	0.682
RBFNetwork	0.778	0.653
RandomTree	0.863	0.602
SPegasos	0.980	0.634