Using Structural Information in Machine Learning Applications

Zafer Barutçuoğlu

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
By the Department of
Computer Science

Advisor: Robert E. Schapire

September 2008

© Copyright by Zafer Barutçuoğlu, 2008. All rights reserved.

Abstract

Classification problems encountered in real-life applications often have domain-specific structural information available on the measured data, which cannot be readily accommodated by conventional machine learning algorithms. Ignoring the structure and blindly running a conventional algorithm on the numerical data can compromise the quality of solutions.

This thesis provides answers to two such complementary settings: one where there is a hierarchy among multiple class labels (output structure), and one where the input features are known to be sequentially correlated (input structure). Probabilistic graphical models are used to encode the dependencies, and model parameters are estimated using efficient inference algorithms. While both scenarios are motivated by real bioinformatics problems, namely gene function prediction and aneuploidy-based cancer classification, they have applications in other domains as well, such as computer graphics, music, and text classification.

The first part focuses on structure among a group of output classes. Large numbers of overlapping classes are found to be organized in hierarchies in many domains. In multi-label classification over such a hierarchy, members of a class must also belong to all of its parents. Training an independent classifier for each class is a common approach, but this may yield labels for a given example that collectively violate this constraint. We propose a principled method of resolving such inconsistencies to increase accuracy over all classes. Our approach is to view the hierarchy as a graphical model, and then to employ Bayesian inference to infer the most likely set of hierarchically consistent class labels from independent base classifier predictions. This method is applicable over any type of base classification algorithm. Experiments on synthetic data, as well as real data sets from bioinformatics and computer graphics domains, illustrate its behavior under a range of conditions, and demonstrate that it is able to improve accuracy at all levels of a hierarchy.

The second part focuses on structure among input features, in the form of a sequential relationship. Generic non-sequential machine learning models assume no importance in the order of inputs. Conversely, sequence models (e.g. Hidden Markov Models) need to assume stationarity to keep the number of parameters manageable, modeling only sequence-wide stability and losing the significance of

particular positions. We propose a fixed-length sequence classification method that combines sequential correlations with positional features in a sparsely regularized solution, with training and inference algorithms in linear-time of sequence length. Motivated by the problem of tumor classification by genetic copy number changes, our method can identify copy number alteration regions in noisy array-CGH data, and locate the genes of clinical relevance driving these alterations and affecting the cancer label. Experiments on synthetic array-CGH data modeled from real human breast tumors, as well as real tumor datasets from breast cancer, bladder cancer, and uveal melanoma, demonstrate that the our method matches or exceeds state-of-the-art methods in accuracy, and is able to produce biologically significant predictions for clinically relevant genes.

Acknowledgments

This research was supported by NSF grant IIS-0513552.

I would like to express my deepest gratitude to my advisor Rob Schapire and co-advisor Olga Troyanskaya for their sound guidance, brilliant feedback, and hearty encouragement. I feel truly privileged and honored to have worked with them.

I would also like to thank my committee members David Blei, Mona Singh, and Fei-Fei Li, and my collaborators who have contributed to the quality of this work. Camelia Chiriac and Rajiv Ayyangar performed the laboratory work in Chapter 2, and Kara Dolinski provided her expert advice on the Gene Ontology. Chris DeCoro has been a true champion of Bayesian Aggregation; he found the problem settings of Chapters 3 and 4, co-authored the resulting papers, and presented them at the conferences, kindly funded by Szymon Rusinkiewicz. For the results in Chapter 3, Philip Shilane and Szymon Rusinkiewicz provided invaluable feedback. Rebecca Fiebrink provided the excellent domain expertise behind the work in Chapter 4 and co-authored the resulting paper. Cory McKay shared the Bodhidharma MIDI dataset used for evaluation, and provided helpful comments. Edoardo Airoldi and Vanessa Dumeaux contributed to the interpretation and presentation of the results in Chapter 6 with their experienced input.

I dedicate this thesis to my parents, Asuman and Sacit Barutçuoğlu.

Contents

List of Figures

Introduction

Machine learning is a crossroads of statistics and computer science, focusing on the automated extraction of information from data. A common problem type is *classification*, where available input-output examples are processed by a learning algorithm to produce a classifier which maps input instances to output classes. The inductive bias of the statistical model used determines how the classification is "generalized" from the given examples to previously unseen inputs.

Although many general-purpose classification methods are available (e.g. Neural Networks, Support Vector Machines, k-Nearest Neighbors, Gaussian Mixture Models, Naïve Bayes, Decision Trees), there is no one "silver bullet" machine learning model. By definition, inductive generalization from data is made possible by making assumptions that constrain or prioritize the hypothesis space. If the assumptions hold, the model can make useful predictions from fewer-than-exhaustive data. If they don't, the desired solution may be overlooked in favor of a poorer hypothesis. General-purpose machine learning can only afford to make very conservative assumptions, treating all possible datasets similarly, although domain experts often have more knowledge about the problem than represented by the data. Ignoring such knowledge and blindly treating the problem as yet another number-crunching task sacrifices valid assumptions that could be encoded for better generalization.

Not all domain knowledge is readily representable in statistical terms, but graphical models have proven to be a conveniently applicable machinery for capturing the *structure* in many problems, directly constraining the hypothesis space by encoding conditional independence assumptions among model variables. The inclusion of structure information in a machine learning model also preserves some of the reusability of machine learning methods; problems from different domains

can exhibit similar structure, sharing solution models.

This thesis is organized in two parts, presenting novel solutions to two real-world problem types, one with structure among multiple output labels, and one with structure among input features. Both methods are motivated by real applications in need of efficient, accurate, and interpretable solutions.

Part I addresses hierarchical classification, a case of *output structure* among a multitude of classes in the form of a hierarchy. The problem is defined, and our solution method, Hierarchical Bayesian Aggregation, is described in Chapter 1, including illustrative analytical experiments on synthetic datasets. Chapter 2 presents its application to gene function prediction for yeast (*Saccharomyces cerevisiae*), the original real-world problem for which our method was developed (Barutcuoglu *et al.*, 2006). The novel function predictions for yeast genes by our method have also been verified by laboratory experimentation. More recently, Bayesian Aggregation was also successfully applied to mouse (*Mus musculus*) gene function prediction (Guan *et al.*, 2008). Chapters 3 and 4 present additional real applications that our method has already found, in the hierarchical classification of three-dimensional shapes in computer graphics (Barutcuoglu and DeCoro, 2006), and the classification of songs by a hierarchy of genres (DeCoro *et al.*, 2007), respectively. A summary of these results are also available in Barutcuoglu *et al.* (2008). Experimental results in all of these applications demonstrate the significant accuracy gains by our method through exploiting the hierarchical structure in the problems.

Part II addresses a case of *input structure* where the input features are known to be sequentially correlated. Motivated by the problem of cancer DNA analysis by genetic copy numbers, Chapter 5 presents our new method, Heterogeneous Hidden Conditional Random Field (HHCRF), a fixed-length sequence classifier that efficiently combines non-sequential and sequential features in a sparse solution, with analytical results on synthetic data. Chapter 6 presents our results on real tumor datasets from breast cancer, bladder cancer, and melanoma. HHCRF matches or exceeds the performance of state-of-the-art methods in these results, and is able to produce biologically useful hypotheses for clinically relevant genes.

Although the original problems for both parts were from bioinformatics, our structured models are novel machine learning methods beyond solely bioinformatical tools. Hierarchical Bayesian Aggregation has already found successful applica-

tions in other domains as presented, and HHCRF may also find other applications in the future, such as in text classification and image processing.

There has been a recent development of theory and algorithms for structured prediction in machine learning, focusing on kernelized maximum-margin solutions over arbitrary structures (see Bakir *et al.*, 2007, for a recent survey). In comparison, our methods in this thesis are geared one step closer to the data, focusing on linear-time efficient solutions for particular types of structure, not addressed by existing structured prediction methods.

Part I

Output Structure: Hierarchical Classification

Large collections of classes are often organized into hierarchies in many domains, such as the Gene Ontology in bioinformatics, 3D object taxonomies in computer graphics, genres in music classification, and web directory categorizations. In the general multi-label classification setup, an example is allowed to belong to multiple classes, but a hierarchical organization implies that members of a class must also be members of all of its super-classes.

To illustrate, Figure 1 shows a subhierarchy from the Gene Ontology, where genes (examples) are to be annotated to biological functions (classes) that they are responsible for. The hierarchical organization of the classes implies that if a gene is annotated to the "Cell Differentiation" class for instance, it also implicitly belongs to its parent, the "Cellular Process" class.

The conventional approach in multi-label classification scenarios is to decompose the problem into independent binary tasks, one for each class. This allows the flexibility of using any available well-understood binary classification algorithm as best suited for the data at hand, at the expense of ignoring correlations among classes. However, in the presence of a hierarchy, these correlations become hard constraints that are likely to be violated by independent predictors. Namely, an example may be predicted as positive for one class and negative for its parent class, so obviously at least one of them must be making a mistake. Not only does this pair of predictions require some modification to be semantically consistent with the hierarchy, but also by making the right modification, overall accuracy may be improved.

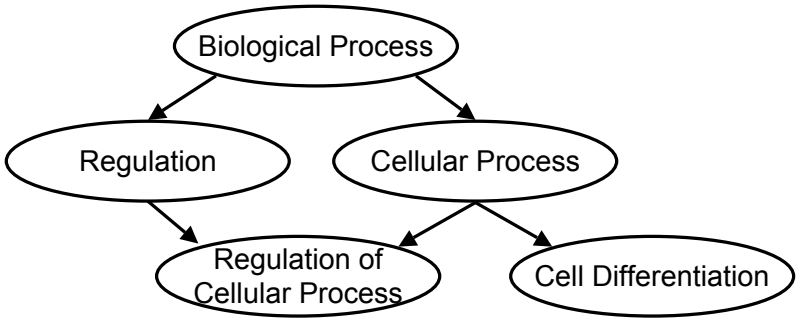


Figure 1: Example of a class hierarchy (a subset of the "biological process" hierarchy from the Gene Ontology). If a gene is annotated to a function class, it implicitly belongs to all of its parent (and ancestor) classes as well.

We have developed the method of Bayesian Aggregation, constructing a Bayesian network from the hierarchy to infer the most likely set of hierarchically consistent class labels from the possibly inconsistent predictions of a set of independent base classifiers for a given example. This framework can accommodate any choice of base classifier, as long as estimates of their prediction accuracies are available on held-out validation data. The hierarchy itself can be any directed acyclic graph, not just a tree.

Other approaches to hierarchical classification include training the classifiers in a top-down manner, where the children are trained only on their parents' members, and are not consulted at all during evaluation if their parent predicts negative (Cesa-Bianchi *et al.*, 2004). This results in simpler classification tasks in the more specific classes, but the higher-level classes derive no benefit from the hierarchy despite the disproportionately large responsibility of their errors affecting all descendants. In contrast, our method allows classes at every level to be influenced by parents and children alike.

Another category of algorithms train all base classifiers in a correlated manner using the hierarchy, making the hierarchical information available during training as well as evaluation (McCallum *et al.*, 1998; Dumais and Chen, 2000; Cai and Hofmann, 2004; Dekel *et al.*, 2004). However, these algorithms do not address multi-label classification. Also, they are currently available as extensions to specific classification models such as perceptrons or SVMs, while our method transparently allows any choice of base classifier, and even different models in the same hierarchy.

In Chapter 1 we describe Bayesian Aggregation, how to construct the Bayesian network, estimate its parameters, and perform inference. We provide extensions of the method to handle more constrained hierarchical scenarios, such as disallowing positive non-leaf predictions without a positive child prediction, or only allowing positive predictions in one class and its ancestors, as well as an adaptation for ordinal classification. We also present experimental results on a range of synthetic datasets, illustrating the behavior of the method under different conditions. Accuracy improvements are observed at all levels of a hierarchy.

In Chapters 2–4, Bayesian Aggregation is applied to three real datasets. Chapter 2 is on predicting biological function of yeast genes over the Gene Ontology hierarchy. Chapter 3 is on categorizing 3-dimensional objects in the Princeton

Shape Benchmark. Chapter 4 is on predicting musical genres for songs in the MIREX Symbolic Music Dataset. Because of different application objectives, the scenarios differ in many ways, including the choice of base classifier, binary-versus-real valued outputs, parameter estimation strategy, and evaluation criteria. In all cases, Bayesian Aggregation provides significant improvement over independent base classifiers.

Chapter 1

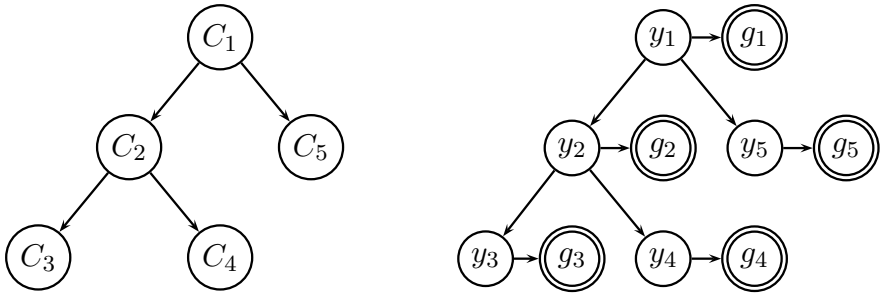
Hierarchical Bayesian Aggregation

In this chapter we first formalize hierarchical consistency, and then describe how to construct a graphical model to infer the most likely hierarchically consistent labels for an example, given possibly inconsistent prior predictions.

For a set of classes $C_1, \ldots, C_n$ organized in a directed acyclic graph (DAG) representing a general-to-specific hierarchy, let $pa(i)$ denote the indices of the parents of class C_i. Given an example x, for each class C_i let the true label $y_i \in \{0,1\}$ denote its membership in that class. The hierarchical consistency constraint dictates all members of a child class also belong to all of its parent classes; i.e., if $y_i = 1$, then $y_j = 1$ for all $j \in pa(i)$. We are also given an independent base classifier for each class C_i which outputs a prediction $g_i \in \{0,1\}$ for the example x, with no regard to being hierarchically consistent among themselves. For a given test example x, we want to determine the most likely set of (consistent) true labels $y_1, \ldots, y_n$ given the (possibly inconsistent) base classifier predictions $g_1, \ldots, g_n$.

1.1 The Bayesian Network

Our goal is to model the joint probability distribution $P(y_1, \ldots, y_n, g_1, \ldots, g_n)$ in a compactly parameterizable way that allows efficient maximization of the posterior $P(y_1, \ldots, y_n | g_1, \ldots, g_n)$. For this purpose, we use the "Bayesian network" paradigm, which represents random variables in a graphical model to encode conditional independence relationships and provides efficient inference methods. See Heckerman (1998) for a tutorial.



(a) Hierarchy of classes (b) Bayesian network

Figure 1.1: The class hierarchy (a) is transformed into a Bayesian network (b). The y nodes are the binary-valued hidden nodes representing actual membership to the class, and the corresponding g nodes are the observed classifier outputs.

Let us construct a Bayesian network from the hierarchy, with edges to each y_i from all $y_j : j \in pa(i)$, and edges to each g_i from the corresponding y_i (Figure 1.1). The network will have a particular configuration of value assignments for an example x (so technically all variables are also conditioned on x, which is omitted for clarity since it is always observed).

This structure assumes conditional independence of the prediction g_i from any other prediction g_j or true label y_j ($j \neq i$) given its true label y_i. The hierarchical edges among the y nodes correspond to conditional class priors, and effectively encode hierarchical consistency, while the edges from y_i to g_i represent the predictive accuracy of the base classifier. A set of predictions for an example x corresponds to the g node values being observed, and any standard method of Bayesian network inference can be used to obtain the maximum-likelihood assignment to all unobserved y nodes, or the marginal probability of an individual class C_i, as detailed in Section 1.3.

1.2 Parameter Estimation

Before inference can be performed, the parameters of this model need to be estimated from data, which are the conditional probability distributions for each variable given its parents in the network.

Specifically, these parameters are the hierarchical class priors $P(y_i|\mathbf{y}_{pa(i)})$ where $\mathbf{y}_{pa(i)}$ denotes all parent y nodes of y_i, and the base classifier output distributions

$P(g_i|y_i)$. Although possible, it is not necessary to use the costly EM algorithm to estimate these. $P(y_i|\mathbf{y}_{pa(i)})$ can be straightforwardly estimated by frequency from training set labels. Indeed, only $P(y_i|\mathbf{y}_{pa(i)} = \mathbf{1})$ needs to be estimated (simply the ratio of parent positives that are also positive in the child y_i). Since training set labels must be hierarchically consistent by definition, the probability that y_i is positive when any of the parents is negative, $P(y_i = 1|\mathbf{y}_{pa(i)} \neq \mathbf{1})$, will be zero, which is also what ensures hierarchical consistency of labels during inference. If Laplace smoothing is to be added, care must be taken to keep these probabilities zero.

The parameters $P(g_i|y_i)$ represent the base classifier output distributions to be expected on previously unseen examples, so estimating them over examples that have been used in training is likely to be severely biased. Assuming that the training data distribution reflects the test data distribution, part of the available data should be held out from training, so the base classifiers can be evaluated on this held-out validation set to provide a better estimate of their performance on new examples. If training data is too scarce to hold out completely, resampling methods such as k-fold cross-validation or bootstrapping can be used instead.

In the case of discrete base classifier outputs, $P(g_i|y_i)$ can be estimated using the confusion matrices over held-out data, where $P(g_i = 0|y_i = 0)$ will be the ratio of negative examples classified correctly (TN/(TN+FP)), $P(g_i = 1|y_i = 1)$ the ratio of positive examples classified correctly (TP/(TP+FN)), $P(g_i = 1|y_i = 0)$ the ratio of negative examples classified as positive, and $P(g_i = 0|y_i = 1)$ the ratio of positive examples classified as negative.

If the base classifiers are able to output real-valued predictions such as probabilities or confidence ratings ($g_i \in \mathbb{R}$), the now-continuous distributions $P(g_i|y_i = 0)$ and $P(g_i|y_i = 1)$ can be modeled parametrically, for example as Gaussians. If $P(g_i|y_i = 0)$ is taken to be the Gaussian $\mathcal{N}(\mu_0, \sigma_0^2)$, its parameters μ_0 and σ_0^2 can be estimated as the mean and variance of the base classifier outputs for the held-out negative examples of that class. Similarly, $P(g_i|y_i = 1) = (\mu_1, \sigma_1^2)$ can be estimated over the held-out positive examples of the class. Instead of explicitly converting them into binary outputs by manual thresholding, providing real-valued base classifier outputs like this allows the system greater freedom during inference, as it will essentially be implicitly performing thresholding for all classes to maximize overall accuracy.

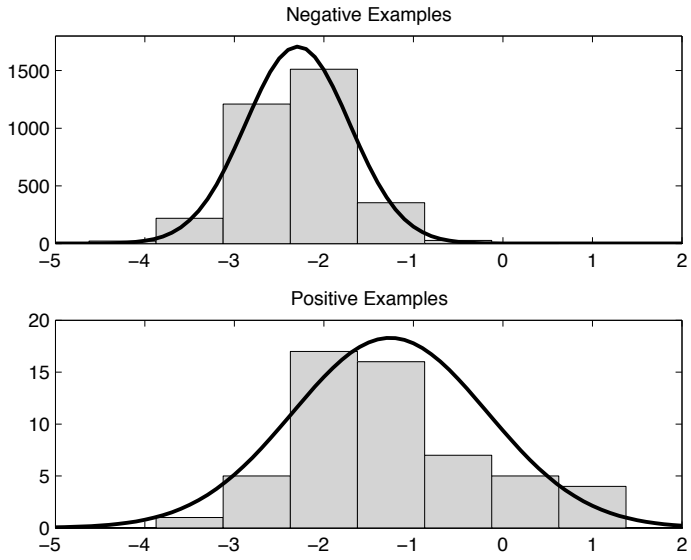


Figure 1.2: Modeling continuous classifier outputs using Gaussians ("chromosome segregation" class of the Gene Ontology). The histograms indicate the distribution of the median outputs of 10 unthresholded SVMs over positive and negative validation examples. (The y-axes have different scales.)

See Figure 1.2 for an example using Gaussians to model the output distributions $P(g_i|y_i = 0)$ and $P(g_i|y_i = 1)$ where the base classifier output is the median of 10 unthresholded support vector machines for the "chromosome segregation" class in the Gene Ontology, described in Chapter 2. The histograms depict the output distributions on positive and negative held-out examples, from which the mean and variance parameters are estimated for inference.

1.3 Inference

Once the Bayesian network is constructed, any Bayesian inference algorithm can be used to compute the most likely label configuration

$$\arg\max_{y_1,\ldots,y_n} P(y_1,\ldots,y_n|g_1,\ldots,g_n). \tag{1.1}$$

Depending on the hierarchy, an exact (e.g. junction trees), or approximate (e.g.

Monte Carlo methods) inference algorithm may be preferable. In the case of tree hierarchies (no multiple parents), an extension of the Viterbi algorithm to trees is available for efficient linear-time inference (Durand *et al.*, 2004).

Instead of the hierarchy-wide most likely binary label configuration, it may also be desirable to compute an example's real-valued marginal membership probability

$$P(y_i|g_1,\ldots,g_n) \tag{1.2}$$

for each class C_i. Even though the assignment of labels maximizing (1.2) for each class does not necessarily maximize (1.1) as a whole, the real values from (1.2) can be thresholded at different levels depending on the application's risk model, or can be used to rank all examples for a class by membership probability.

1.4 Variations and Extensions

We next discuss some of the many possible variations on the basic technique outlined above.

1.4.1 Upward Edges

The hierarchical edges from each y node downward to its children imply the assumption of conditional independence of the children given the (positive) parent label. Reversing these edges to point upwards in the hierarchy (leaving the edges from y_i to g_i unaffected) still results in an acyclic graph, but also allows sibling nodes to be correlated given the parent, which may make more causal sense in some practical applications. In that case, the Bayesian network is parameterized over children, so $P(y_i|\mathbf{y}_{ch(i)})$ will be estimated (where $ch(i)$ denotes the child class indices of C_i), and $P(y_i = 0|\mathbf{y}_{ch(i)} \neq \mathbf{0})$ will be kept zero to ensure hierarchical consistency. However, hierarchies tend to have many more children than parents, so the increase in the number of multinomial conditional probability entries (exponential in the number of children) will also result in fewer data available to estimate each value, and the degradation in parameter accuracy may offset the benefits of better causal modeling, yielding worse inference results. In practice, both configurations should be compared on the data at hand.

1.4.2 Forcing Leaf Predictions

Some applications may dictate that positive labels for each example must propagate down to leaf nodes; that is, a non-leaf node cannot be positive when none of its children are positive. Here the children of a positive node obviously must be correlated, and using upward edges as described above with the additional setting of $P(y_i = 1|\mathbf{y}_{ch(i)} = \mathbf{0}) = 0$ extends our method to this case. Inference under these settings cannot result in label configurations with terminal positive predictions at non-leaf nodes.

1.4.3 One-Class Predictions

Another possible variation on hierarchical predictions is to require each example to belong to exactly one most-specific class, together with all of its ancestors (in the case of a tree, this is a single branch from the root, but not necessarily reaching a leaf). To enforce this constraint, the binary y nodes in our Bayesian network can be converted to be multinomial, with y_i having $|ch(i)| + 2$ possible values, one for "negative", one for "terminal positive" (none of the children are positive), and one for each child being the positive descendant. The conditional probabilities will then be set to ensure that when y_i has the value "negative" or "terminal positive" all of its children are "negative", and when y_i has the value "child y_j is positive" its child y_j is not "negative" and all other children are "negative".

This constraint may also be combined with forcing leaf predictions, which will be akin to making a multi-class single-label prediction over the leaf classes. While it is possible to enforce the multinomial constraints using the upward arrows in the above case instead of downward, this is not necessary, as removing the "terminal positive" value from the multinomial variables is sufficient.

1.4.4 Ordinal Classification

An interesting special case to consider is a single-branch hierarchy, that is, a chain of classes, which may represent an array of predictors for binary "greater-than" decisions on an ordinal value range. For example, predicting user ratings for movies over the ordered label set of one to five, we may have the first classifier predicting

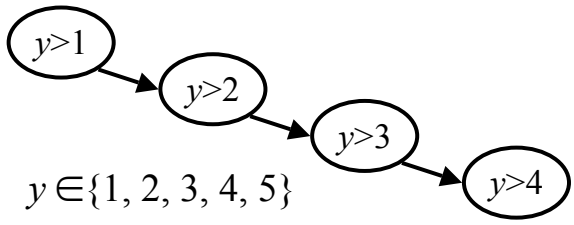


Figure 1.3: Constructing a single-branch hierarchy for performing ordinal classification using Bayesian Aggregation.

whether the label is greater than one, the second classifier predicting whether it is greater than two, and so on (Figure 1.3). Ordinal consistency of these predictions is equivalent to hierarchical consistency down a single branch. Constructing our Bayesian network for this chain hierarchy yields a Hidden Markov Model, which makes the standard Viterbi algorithm directly applicable for inference.

1.5 Analysis on Synthetic Data

Being aware of the performance characteristics of a method and its optimal conditions of usage depending on the data at hand is crucial for informed application of machine learning. To illustrate the behavior of Bayesian Aggregation over different hierarchies, data distributions, and base classifier accuracies, we performed evaluations on a variety of synthetic datasets. The objective of these experiments was to observe the method's performance under different combinations of controlled conditions: wide versus narrow hierarchies, shallow versus deep hierarchies, sparse leaf classes (few positives, which might make leaf classes less useful) versus well-populated leaf classes (many positives, which may lead to too many positives and too few negatives in high-level classes, making those less useful), in combinations over a range of base classifier accuracies. These results were originally made available in Barutcuoglu *et al.* (2008).

To provide different data distributions, we used two alternative strategies for generating synthetic data. In *bottom-up* data generation, every class initially contains a fraction p of all examples as positives, sampled independently. Then, given the hierarchy, each class also inherits all positives of its descendants. This results in leaf classes having exactly pN positives for N examples, but the number

of positives grows very quickly when going up in the hierarchy, especially for large numbers of children. In *top-down* generation, top-level classes contain a fraction q of all examples as positives, and every child class has the fraction q of its parents' positives, yielding q^{j+1} positives at a depth-j class in a tree hierarchy. This yields a relatively more gradual change of positive/negative ratio from one level to another, independent of the number of children.

A base classifier for each class is assumed to be available, yielding binary predictions on these examples, known to have a fixed accuracy a, the probability of predicting the correct label. We then construct our Bayesian network, and infer the corrected predictions for each example as the most likely true labels given the classifier predictions. The hierarchical edges in the Bayesian network are downward for the top-down data (children uncorrelated given the parent), and upward for the bottom-up data (children correlated given the parent).

Note that since our Bayesian network does not need to see the input features, and the labels and base classifier outputs are already available, it is not necessary to actually generate the input features and train real base classifiers on them. All we are concerned with is that these base classifiers have made their predictions with a known accuracy.

We generated bottom-up data with $p = 0.01$ and top-down data with $q = 0.75$, over $a \in \{0.70, 0.80, 0.90\}$ on four tree hierarchies: `T2D4` and `T2D6` are binary trees of depth four and six, and `T5D2` and `T5D3` are 5-ary trees of depth two and three, respectively. Accuracies before and after Bayesian correction, averaged per node depth, are reported in Table 1.1.

Note that in terms of the fraction of positive examples, `T2D4` is identical to the top depth-4 subtree of `T2D6` for top-down data, to the bottom depth-4 subtrees of `T2D6` for bottom-up data, and a similar relationship holds for `T5D2` and `T5D3`; hence the depth columns are aligned accordingly in the table.

1.5.1 Observations

Bayesian Aggregation clearly improves average accuracy in all cases except one. Large accuracy improvements can be found at every level.

Comparing `T2D4` to `T2D6` and `T5D2` to `T5D3` on top-down and bottom-up data

shows that the benefits from the addition of new nodes at the top or the bottom of the hierarchy gradually propagate to all other levels.

Closer inspection of the predictions T5D3 with $a = 70\%$ on top-down data, as well as the local deteriorations in some levels of the other cases, reveals that starting out with relatively inaccurate classifiers hinders the improvement from the aggregation. Note that the term "inaccurate" is relative to the node. For example, the 70% accurate base classifier at the root of T5D3 is worse than a constant positive prediction, as the root data is 75% positive.

To illustrate, in the absence of all hierarchical edges, inference on the isolated two-node subnetwork $y_i \to g_i$ will have one of four effects; y_i will always be equal to g_i, or always its reverse, or constant-positive, or constant-negative, whichever maximizes accuracy. If g_i has accuracy a and the percentage of positives in the data is p, y_i will effectively disregard the information from the base classifier unless $a > p$ and $a > 1 - p$. This condition is violated at the italicized levels in Table 1.1, and although less unforgiving in the presence of an actual hierarchy, improvement generally suffers with such inaccurate classifiers. With base classifiers that are generally better than constant predictions, accuracy improvements are substantial.

The difference between having a broad and shallow hierarchy versus a narrow and deep one can be observed by comparing T2D4 to T5D2 (both have 31 nodes in total) and T2D6 (127 nodes) to T5D3 (156 nodes). Except for the degenerate 70% accurate case, T5D2 and T5D3 have larger improvement at the root than their narrower, deeper binary tree counterparts. However, the binary trees have larger improvements towards the lower levels, and therefore better accuracy on average, as the (unweighted) average is dominated by the large number of leaves. In other words, to reap the maximum benefit from Bayesian Aggregation, it is better for a node to have immediate children than to have them arranged in deeper levels, but deep cascading benefits more nodes, and yields larger average improvement in accuracy.

The distribution of improvement across the hierarchy varies with the label distribution and base accuracy. Although deeper nodes seem to get the largest improvement in general, the 90% accurate top-down data (the only case with no worse-than-constant classifiers) seems to favor mid-level classes in T2D4 and T2D6, and higher-level classes in T5D2 and T5D3.

1.5.2 ROC Analysis

Although we start out with base classifiers of equal accuracy on positives and negatives, their corresponding post-inference most-likely labels are affected by the skew of the data, specifically because Bayesian inference is improving accuracy. To observe this effect, it is useful to visualize classifiers on the ROC (receiver operating characteristics) axes, namely the true-positive ratio (accuracy on positive examples) against the false-positive ratio (error on negative examples). Figure 1.4 shows one case, where each base classifier is a dot in the center cluster of 90% TP and FP ratios (variation is due to sampling), and each post-inference maximum-likelihood label, viewed as a predictor, is an asterisk.

The clustered spread of the post-inference predictors along an arc around the base classifiers can be explained by the property that points of equal accuracy (TP+TN) lie on parallel lines (*isoaccuracy* lines) on the ROC axes. While ROC coordinates have the property of being insensitive to the positive/negative skew of the data, skew changes the slope of the isoaccuracy lines, flatter than diagonal for positive majority, and steeper for negative majority. The effect of the Bayesian Aggregation is to make each classifier more accurate, moving it on the ROC axes perpendicularly to isoaccuracy lines, but since each level of the hierarchy has a different ratio of positives, these movements are in different directions, fanning out from the original common location. The post-inference asterisks in Figure 1.4 are visibly clustered by level, with the root at the top (flattest isoaccuracy line, most upward deflection) and the leaves in the bottom cluster (steepest isoaccuracy, most leftward deflection).

1.6 Discussion

We outlined a general and principled method for resolving hierarchical inconsistencies among a set of independent classifiers, thus increasing accuracy over all classes. Its usefulness was demonstrated under a range of conditions on synthetic data. We also described various extensions of the method to handle other constraints, as well as a special case for ordinal classification.

Future work on Bayesian Aggregation may provide extensions where the in-

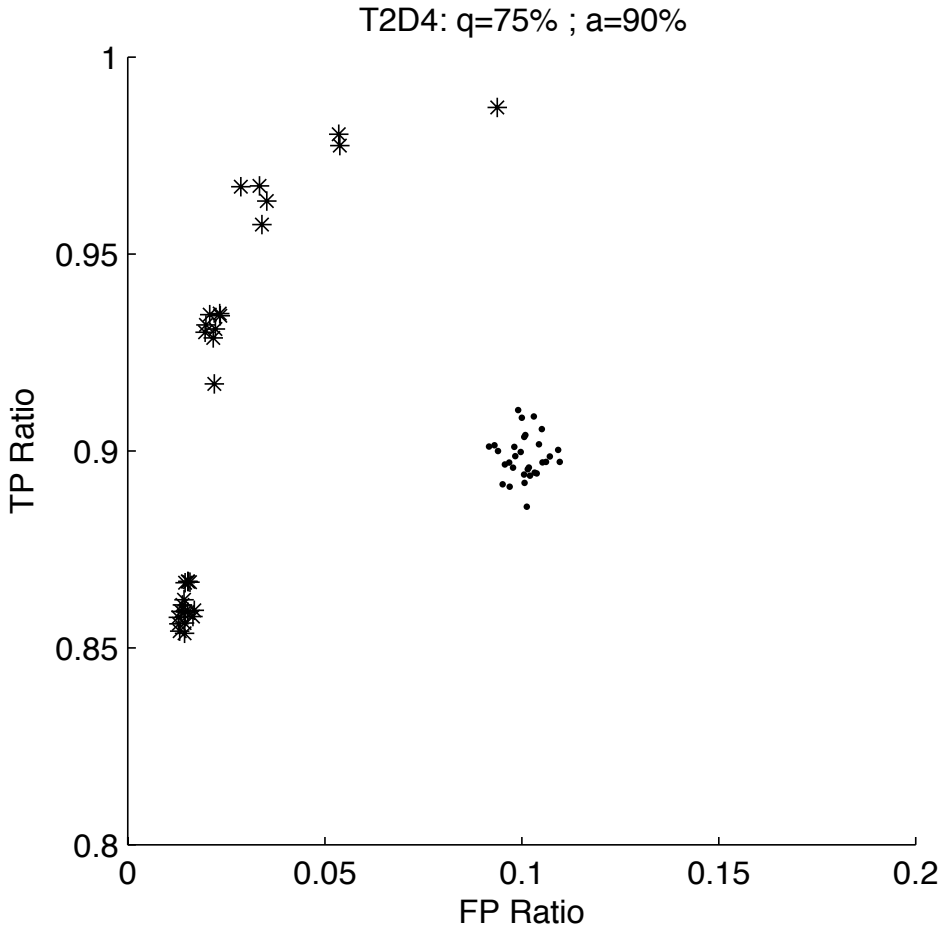


Figure 1.4: ROC plot detail of classes before and after Bayesian inference. Dots indicate base classifiers (variation due to sampling), and asterisks indicate post-inference predictions for each class (T2D4 top-down data, $\alpha = 90\%$). Post-inference coordinates are clustered along an arc because the increase in accuracy after inference moves each class in a direction determined by its positive/negative ratio, which depends on its depth in the hierarchy in our data.

ference algorithm optimizes other performance criteria than overall accuracy, such as explicitly maximizing AUC scores. The sensitivity of the method to different parameter estimation errors may also be explored.

In the remaining chapters of Part I, we present applications of Bayesian Aggregation to real problems.

Table 1.1: **Synthetic Data Results.** a is the base classifier accuracy, a' is the accuracy of the most likely labels after inference, averaged over all classes, and $\Delta a = a' - a$, broken down by nodes at different depths in the tree, shown as percentages. Italicized levels indicate base classifiers that are less accurate than a constant negative or positive prediction.

NAME	a	a'	Δa	Δa **BY DEPTH (ROOT → LEAF)**						
Top-down, 75% pos. at root				ROOT						
T2D4	70	78.6	8.5	*−8.8*	−3.6	3.2	9.2	*12.1*		
T2D6	70	84.8	14.8	*−18.9*	−10.6	−1.7	5.3	*11.6*	*15.8*	*18.7*
T5D2	70	66.9	−3.2	*−14.2*	−8.6	−1.7				
T5D3	70	70.3	0.3	*−22.6*	−16.1	−5.6	2.3			
T2D4	80	89.4	9.4	5.8	7.2	8.6	9.4	10.1		
T2D6	80	93.7	13.7	7.4	9.7	10.9	12.2	13.3	*13.8*	*14.3*
T5D2	80	86.1	6.2	12.1	9.7	5.3				
T5D3	80	89.4	9.4	18.7	13.5	12.0	8.7			
T2D4	90	95.9	6.1	6.5	6.8	6.6	6.4	5.6		
T2D6	90	97.8	7.8	6.7	6.9	7.9	8.2	8.2	8.1	7.6
T5D2	90	94.4	4.3	9.4	7.1	3.6				
T5D3	90	95.9	5.8	10.0	9.6	8.0	5.2			
Bottom-up, 1% pos. at leaf										LEAF
T2D4	70	96.1	26.1			3.9	*16.0*	*23.3*	*26.9*	*29.1*
T2D6	70	95.3	25.2	*−24.3*	*−7.6*	7.0	*17.1*	*23.4*	*27.1*	*28.7*
T5D2	70	97.4	27.4					3.7	*24.3*	*29.0*
T5D3	70	96.9	26.9				*−48.5*	3.1	*24.0*	*29.0*
T2D4	80	97.1	17.1			3.9	10.6	*15.3*	*17.6*	*19.0*
T2D6	80	97.1	17.1	*3.1*	*4.9*	7.6	12.6	*15.5*	*17.6*	*19.0*
T5D2	80	97.3	17.3					−6.2	*14.0*	*18.9*
T5D3	80	97.3	17.3				*−6.0*	-0.6	*14.3*	*18.8*
T2D4	90	98.9	9.0			5.1	6.5	*8.4*	*9.0*	*9.7*
T2D6	90	99.0	9.0	*7.1*	*6.4*	7.0	7.6	*8.5*	*9.0*	*9.5*
T5D2	90	98.6	8.6					1.4	*6.7*	*9.2*
T5D3	90	98.5	8.5				0.2	1.7	*6.8*	*9.2*

Chapter 2

Gene Function Prediction

Discovering biological functions of an organism's genes is a central goal of functional genomics. Assigning functions for every gene with traditional experimental techniques could take decades, but the currently accumulated data from different biological sources make it possible to generate automated predictions that guide laboratory experiments and speed up the annotation process. Several studies have applied machine learning methods to data from biological experiments to infer functional similarities among genes, or directly predict function for unknown genes (e.g. Karaoz *et al.*, 2004; Clare and King, 2003; Lanckriet *et al.*, 2004b). Recently, integration of different types of biological data in a single model has shown promising improvements, using such learning methods as support vector machines and Bayes nets (e.g. Lanckriet *et al.*, 2004a; Troyanskaya *et al.*, 2003; Chen and Yu, 2004; Pavlidis *et al.*, 2002).

Collections of functional class definitions, such as the Gene Ontology (GO) (Ashburner *et al.*, 2000) and CYGD of MIPS (Guldener *et al.*, 2005), are organized hierarchically, where general functions include other more specific functions. However, existing prediction approaches typically formulate the problem on a per-function basis, ignoring this structure. A separate independent classifier is constructed to predict membership for each functional class. This turns the problem into a convenient form for common machine learning algorithms, but abstracts away any information in the functional hierarchy, essentially allowing the classifiers as a whole to violate the principle that a gene cannot belong to a child class without also belonging to its parents in the hierarchy. This structure has not been exploited in any

previous work to improve classification and enforce consistency.

We apply Bayesian Aggregation on biological data from a variety of sources on the well-studied model organism *Saccharomyces cerevisiae*, known as baker's or brewer's yeast, to predict gene function over the Gene Ontology hierarchy. Our method preserves the hierarchical information to increase predictive accuracy over all classes. We demonstrate the superior performance of our model on held-out data as well as new annotations added to the Gene Ontology after our model was trained. We also make new biologically relevant predictions on genes with previously unknown function, which are then verified by laboratory experiments. The results in this chapter have been published in Barutcuoglu *et al.* (2006). Bayesian Aggregation was later successfully applied also to mouse (*Mus musculus*) gene function prediction by Guan *et al.* (2008).

2.1 Training Individual Classifiers

Our training process starts by training the individual classifiers. In this phase, independent classifiers are conventionally trained for each class, with no regard to the hierarchy. Each classifier is responsible for predicting membership to a particular class by means of a binary or real-valued output. For the classifiers in our experiments, we used support vector machines without thresholding their outputs, as will be described in more detail.

Since validation results will be needed for the Bayes net, some training examples have to be held out. However, for most classes in our data positive examples are too rare to hold out completely, so we use bootstrapping (random sampling with replacement) to create 10 bootstrap samples from the full training data for a class, each of which excludes a ratio of approximately 0.368 of all examples, and may contain multiple copies of the rest (Efron and Tibshirani, 1994). Each bootstrap sample is used to train a separate classifier, yielding 10 classifiers for a class. For a previously unseen example, each classifier of the class will be evaluated, and the median of their outputs will be taken as the combined output. This corresponds to taking a majority vote with a given threshold, and is known as *bagging* (bootstrap aggregating) (Breiman, 1996). For evaluating training examples for validation, only those classifiers which were not trained on that example will be included in

Algorithm 1 Parameter estimation for a class using bagging

Given: Training set D of size N;

for $i = 1$ to 10 **do**
  Create bootstrap sample S_i of size N from D (randomly with replacement).
  Train base classifier C_i on S_i.
end for

Parameter estimation:
for $j = 1$ to N **do**
  Get prediction g_j for input $x_j \in D$ as the median of outputs from $\{C_i : x_j \notin S_i\}$
end for
Estimate $P(g|y)$ using predictions g_j and true labels y_j for $j \in \{1, \ldots, N\}$.

the aggregation, as shown in Algorithm 1.

A widely used alternative to bootstrapping is k-fold cross-validation. In this setting we prefer bootstrapping because to achieve a held-out example ratio comparable to a bootstrap sample (a ratio of approximately 0.368 of N unique examples are expected to not appear in a bootstrap sample of size N), cross-validation would be limited to $k = 3$, reducing the benefit of aggregation (Barutcuoglu and Alpaydin, 2003).

We use the aggregate predictor as the final classifier instead of training a new one on all training data, because in the latter case the final classifier could have an unexpectedly high error that would be impossible to detect. On the other hand, an aggregate classifier is expected to improve the accuracy of individual classifiers, which have known accuracies from validation.

2.2 Constructing the Bayes Net

The next step is to construct the Bayes net to combine outputs. First, we assume the aggregate classifier outputs to have Gaussian distributions for positive and negative examples. Using the validation-phase predictions from held-out examples (or rather, held-out bootstrap-classifiers for each example), we estimate means and variances for each class. Figure 1.2 is a typical plot of outputs for a GO node, showing that our assumption of Gaussian distributions is not unreasonable. These estimated distributions define the conditional probabilities $P(g_i|y_i = 0)$ and $P(g_i|y_i = 1)$

in our Bayes net.

Since the hierarchy's branching factor was not prohibitively high to use upward edges (as discussed in Section 1.4.1), we performed our experiments with upward as well as downward edges. Upward edges proved to yield slightly better results, as we expected because of a better modeling of conditional dependencies: given that an example is a member of a parent class, knowing its membership to one child should alter its expectation of membership to another child. All results reported below are using upward hierarchical edges.

2.3 Support Vector Machines

The *support vector machine* (SVM) classifier is a state-of-the-art machine learning method that separates positive and negative examples using a linear decision boundary (a hyperplane) in a feature space, and aims to achieve better generalization through a principle termed *maximizing the margin* (Burges, 1998).

For our individual classifiers, we trained the SVMs using the SVM^{light} software (Joachims, 1999).

We experimented with linear SVMs as well as radial-basis-function (RBF) kernel SVMs using various radius values, over a wide range of capacity (C) values. The functional genomics data we used turned out to be linearly separable in the input space, and cross-validated results showed that generally hard-margin ($C = \infty$) linear SVMs performed slightly better than soft-margin linear or RBF-kernel SVMs, RBF kernels severely overfitting at times. This is not surprising considering the large number of input features compared to the relatively small number of training examples.

As a sanity check, we also trained Logistic Regression classifiers which lack the maximum-margin properties of SVMs. Logistic Regression classifiers showed worse generalization than SVMs as expected, and we chose hard-margin linear SVMs as the base classifiers in this study.

However, instead of regularly thresholding an example's distance to the separating hyperplane at zero to obtain a binary prediction, we use unthresholded SVM outputs for the purposes of Bayesian combination. Therefore, all further mentions of SVM predictions are unbounded real values. The calibration of these outputs to

actual probabilities is done implicitly by Bayesian Aggregation's output distribution modeling.

2.4 Data Sources and Processing

2.4.1 Training Labels

Our training labels come from the "biological process" hierarchy of the Gene Ontology (GO). A positive example for a class denotes a gene that is annotated to that node or one of its descendant nodes in GO. This upward propagation of annotations is natural because of the hierarchical nature of GO, and it also has the consequence that all training label configurations are hierarchically consistent by definition.

Of the 1000 nodes in the Gene Ontology which have direct annotations for yeast, in consultation with a yeast geneticist, we chose 105 nodes that have both a reasonable number of annotations (at least 1 direct and 15 total annotations) and were specific enough to have some biological significance if a new gene were predicted to belong there.

Unfortunately, GO annotations are almost exclusively positive[1]. This renders the dataset unsuitable for typical classifiers as is, as there are few negative examples to separate from the positives. Similarly to previous work, we use an annotated gene as a negative example for the nodes it is not annotated to. There is some justification to this *ad hoc* introduction of negatives; if a gene is already known to be annotated to a node, it is less likely to be annotated to additional nodes. As a slight improvement to this, we also do not include a node's direct positive annotations as negatives in any of its descendants, reasoning with the same biological intent of making the assumed set of negatives more specific. Our experiments show that this additional filtering of introduced negatives provides a small improvement in accuracy, but more importantly it yields more unused examples to be evaluated later at the child nodes for new discoveries.

[1]The GO does include occasional negative annotations, but these are the very few cases found to be particularly surprising for biologists.

2.4.2 Interaction Data

Pairwise interaction datasets denote the existence of an interaction between pairs of proteins (gene products) under certain experimental conditions. The biological premise is that if two proteins have a certain interaction, their genes might be more likely to belong to the same functional class.

In our experiments we use the GRID collection of pairwise interaction datasets (Breitkreutz *et al.*, 2003). Our snapshot contains 8 types of interactions (Affinity Precipitation, Two Hybrid, Synthetic Lethality, Affinity Chromatography, Synthetic Rescue, Dosage Lethality, Purified Complex, Biochemical Assay). Each of these datasets can be viewed as a square binary matrix whose non-zero elements denote an interaction between the row gene and the column gene. To transform this second-order data type into the form expected by typical machine learning algorithms, we treat each column of a matrix as a feature, treating rows as the examples' feature vectors. For the 9 matrices (and their transposes since they are not necessarily symmetric), we concatenate the feature vectors obtained as such, and obtain 88200 features for 4900 genes.

2.4.3 Microarrays

Microarray datasets are real-valued matrices measuring gene expression levels under different experimental conditions. We use gene expression microarray data from the Stanford Microarray Database (SMD) containing results from several publications (Spellman *et al.*, 1998; Gasch *et al.*, 2000, 2001; Sudarsanam *et al.*, 2000; Yoshimoto *et al.*, 2002; Chu *et al.*, 1998; Shakoury-Elizeh *et al.*, 2004; Ogawa *et al.*, 2000), providing a total of 342 real-valued features for 5737 common genes. 4524 of these genes are among the 4900 genes in GRID, and we fix this set of 4524 genes to be used in our experiments.

Microarray entries typically include missing values due to experimental imperfections. We estimate such entries using the widely accepted `KNNimpute` algorithm (Troyanskaya *et al.*, 2001) with $k = 15$.

2.4.4 Colocalization Data

Colocalization datasets provide information about where the gene products are found in the cell. If two proteins are found in the same locales in the cell, their genes may be more likely to belong to the same class.

We combine two colocalization datasets. One is the curated localization data from the "molecular complex" hierarchy of the Gene Ontology, of which we use 148 terms, providing data for a total of 1043 genes. The other is an adaptation of the O'Shea data (Huh *et al.*, 2003) used in a previous function study, and contains 2902 genes (Jansen *et al.*, 2003).

We treat each locale of each dataset as a binary feature where 1 denotes membership. For genes not included in these datasets, we use zero values.

2.4.5 Transcription Factor Binding Sites

Transcription factor binding sites data are another grouping of genes based on a common attribute. If two genes are known to have a common transcription factor binding site, they are more likely to belong to the same biological process, since they are being transcribed together.

We use the PROSPECT dataset (Fujibuchi *et al.*, 2001), which has 27 experimentally identified sites for 146 genes. As before, we treat the presence of absence of each site as a binary feature and use zeros for genes not included.

2.4.6 Processing of Input Data

We integrate all data into a single feature set by concatenating all feature vectors for a gene. Namely, the input features for an example consist of that gene's interaction flags with all genes for all interaction experiments, flags for membership to each colocalization locale and possession of each transcription factor binding site, and the log-ratio values for each slide in each microarray experiment. Real-valued features (currently only microarray data) are z-score normalized to zero-mean and unit-variance.

Such direct combination of all data into a single dataset was termed *early integration* in Pavlidis *et al.* (2002), and used with feature scaling, was found to be the

best-performing data integration policy for most of their selected classes. Although we do not apply explicit feature scaling in our experiments, our linear SVMs are essentially feature scaling operations themselves, since an inner product with the separating hyperplane normal vector is a particular linear weighting of features.

Most interaction datasets being rather sparse, most of the columns in the resulting dataset of $88,721$ dimensions contain less than two non-zero entries. Removing all such uninformative columns leaves 5930 features for the 3465 annotated genes.

Upon training of our linear SVMs, we can assess how much each data type contributes to classification decisions based on the linear coefficients for decision boundaries. We examined the ratio of each data type's absolute-valued weights to the total (including bias). Although the ratios vary widely for different nodes, on average the interaction data contribute 60%, microarrays 36%, colocalization 4% and transcription factor binding sites less than 1% of the weights. The number of interaction features is disproportionately larger ($88,200$) than microarrays (342), but the interaction datasets are very sparse and binary while microarrays are dense and real-valued. Thus, the substantial contribution of microarrays is not surprising, and for 27 of the 105 classes microarray data indeed contribute the most to the weights.

2.5 Experimental Results

2.5.1 Prediction Accuracy on Held-out Data

We use the *AUC score* (area under the ROC curve) as a measure of the ranking accuracy of the classifiers' outputs. This measure is invariant to changes in the actual calibration of outputs as long as their ordering stays the same, so it eliminates the issue of unbounded SVM outputs and Bayes net probabilities having different range and calibration. For each node, we compare the AUC scores of the aggregate SVM output and the marginal probability from the Bayes net.

Our approach of hierarchical Bayesian correction increased AUC for 93 of the 105 nodes. Our GO subhierarchy is shown in Figure 2.1, colored by changes in AUC score where darker shades of blue indicate larger increases, and darker shades of red indicate larger decreases. The largest improvements were observed at deeper

nodes. The Bayesian combination has substantially increased accuracy for the vast majority of functional categories, as evident from the comparison of AUC scores before and after hierarchical improvement (Figure 2.2).

Figure 2.3 and Figure 2.4 compare a set of raw SVM (aggregate) predictions to the corresponding Bayesian marginal predictions for a particular held-out gene. Using the default zero threshold for SVMs and 0.5 for the Bayes net probabilities, the Bayes net corrects the inconsistency in the SVM predictions, and also correctly changes predictions for lower-level nodes as well.

The average change in AUC over all nodes is +0.033 (a ratio of 4% improvement over the old AUC), with a minimum of −0.031 and maximum of +0.346 (63% improvement over the old AUC). Note that this comparison of absolute changes in AUC does not take into account that for fairly accurate classifiers the room for improvement is small. One way to correct this is to observe the proportion of *decrease* in the area *over* the ROC curve (1−AUC), which is literally the room for improvement. On average, the new area *over* the ROC curve is 21% less than the old one.

The AUC deterioration in some nodes after Bayesian combination might be due to poor output modeling by Gaussians, or correlated errors (mutual misleading) among nodes. In any case, such nodes are identified in these cross-validation results before actual use, so for those nodes SVM predictions can simply be used unmodified. In that case, we get an average AUC change of +0.035.

2.5.2 Predictions for New Annotations

To maintain consistency of our experiments over time, we used a snapshot of the Gene Ontology annotations taken at the beginning of this work in April 2004. Since then, as of July 2005, 88 previously unannotated yeast genes for which we have input data have been annotated in our selected hierarchy of 105 nodes (or their descendants), yielding 346 new gene-to-node pairs. For these 88 genes, we examined predictions using our old snapshot data before and after Bayesian correction.

Independent SVM classifiers for these genes achieve 32% precision (TP/(TP+FP)) and 7% sensitivity (TP/(TP+FN)). Using Bayesian correction improves the sensitivity three times (21%) at comparable precision (31%). Furthermore, at the same sensitivity (7%), the Bayes net produces 51% precision. As the Bayes net gives a

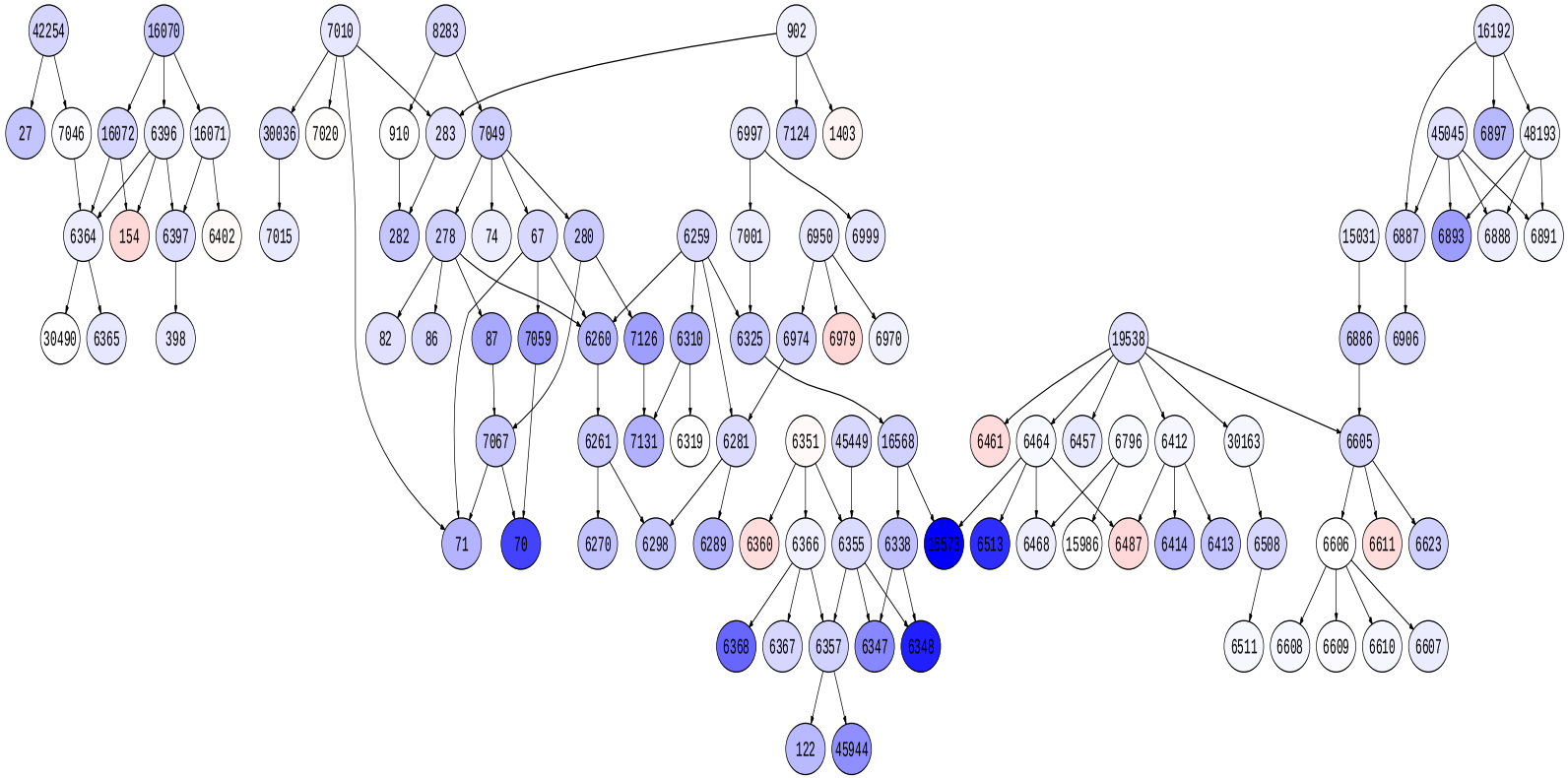


Figure 2.1: The GO subhierarchy used in this study colored by AUC changes. Darker shades of blue indicate largest improvement and darker shades of red indicate largest deterioration. White nodes indicate no change in AUC. Nodes are labeled with abbreviated GO IDs; e.g. "7087" stands for `GO:0007087`. Most nodes improve in AUC, with the largest improvement observed at leaf nodes. Among the few deteriorations, most are also at leaf nodes.

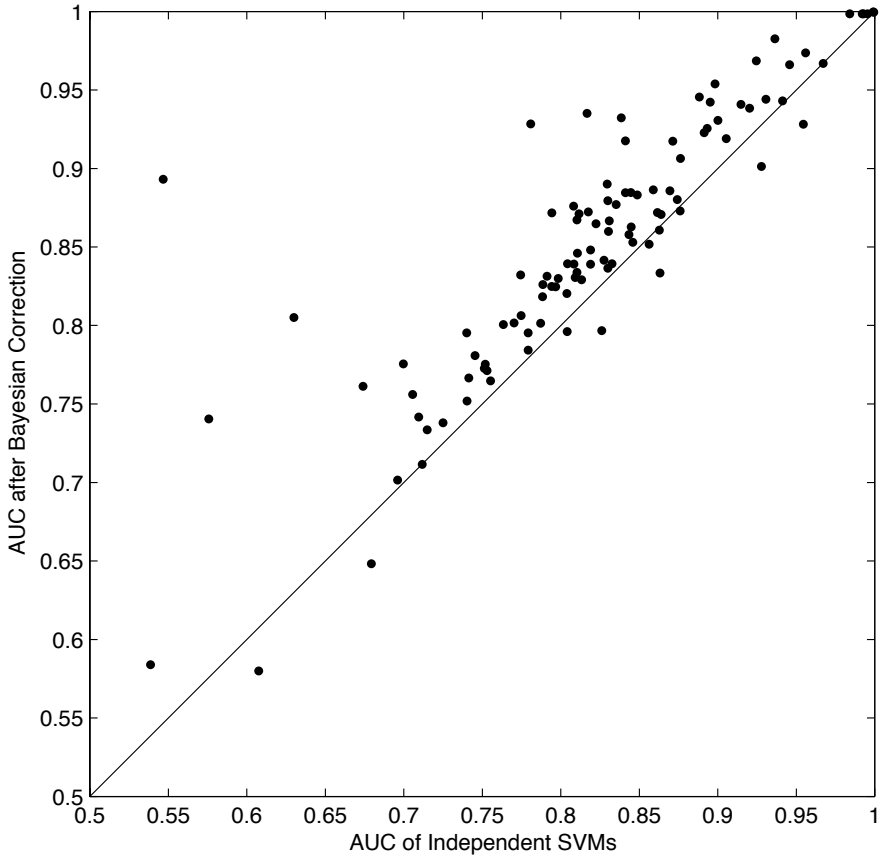


Figure 2.2: Scatterplot of AUCs for the 105 Gene Ontology classes, after versus before Bayesian Aggregation. Points above the diagonal correspond to improvements by our method.

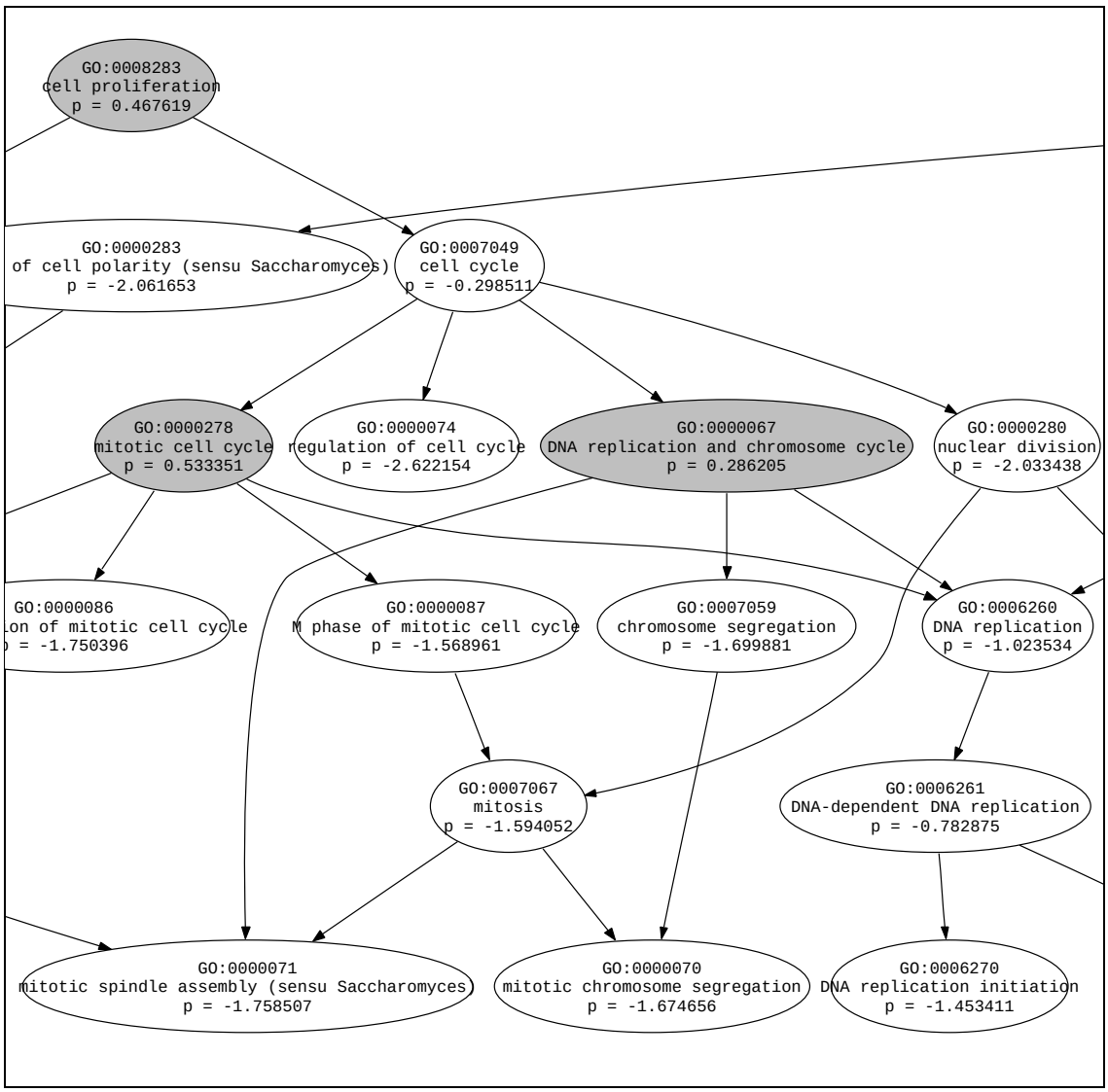


Figure 2.3: Independent SVM outputs (gray if above 0) for held-out gene YNL261W. SVM outputs have an inconsistency at the GO:0007049 "cell cycle" node.

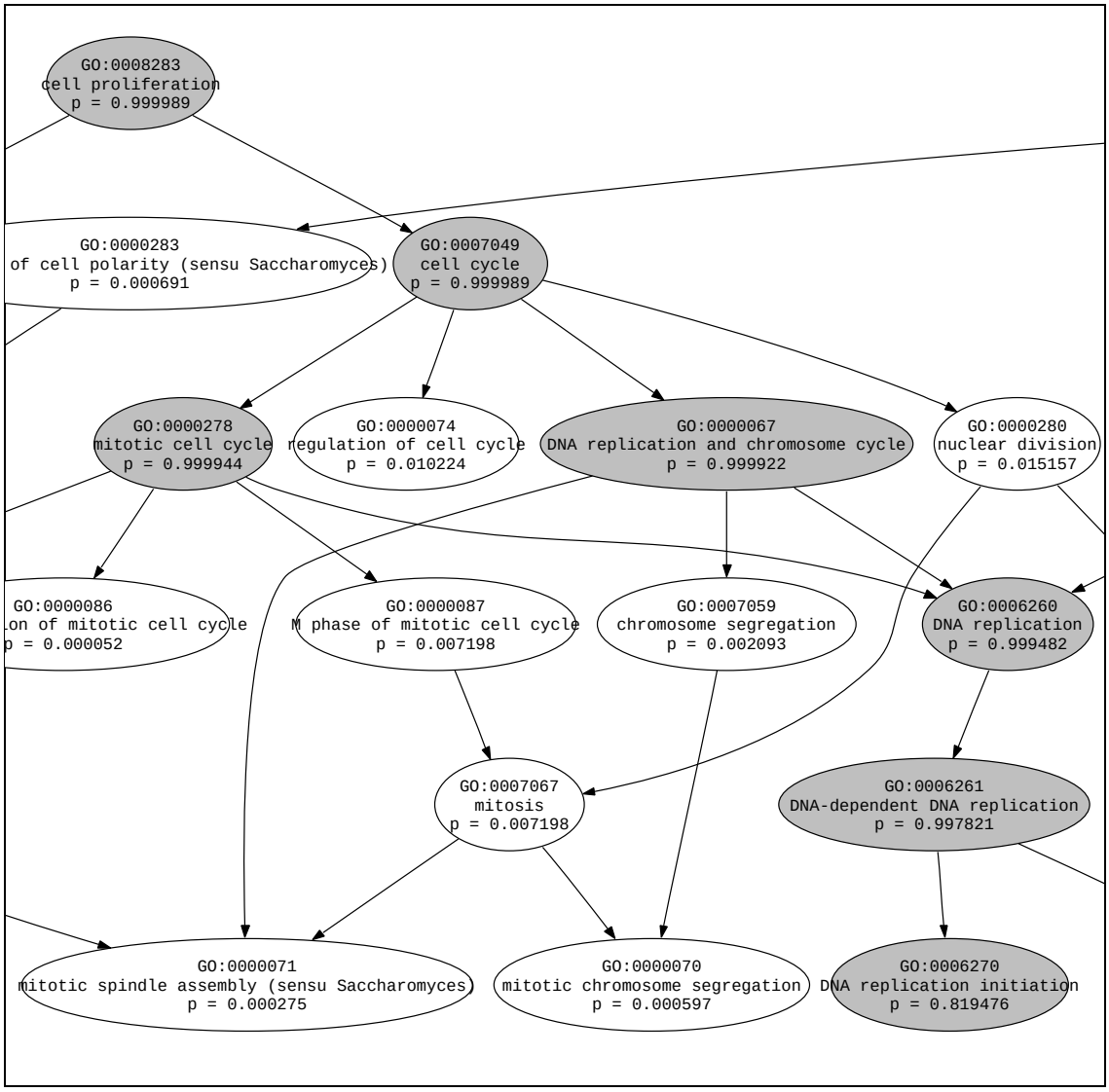


Figure 2.4: Marginal probabilities from the Bayes net (gray if above 0.5) for held-out gene YNL261W. The SVM output inconsistency at the GO:0007049 "cell cycle" node is corrected by Bayesian Aggregation. Also, the Bayes net can change predictions for seemingly consistent nodes as well, including leaf nodes. For this gene, all Bayesian predictions thresholded at $p = 0.5$ are correct. YNL261W is a subunit of the origin recognition complex that binds to replication origin and directs DNA replication (Bell, 2002).

confidence-based output, by thresholding at the desired level our method can be used to leverage sensitivity or precision or both.

Although some SVM predictions are incorrectly changed from positive to negative due to noisy data, many more predictions are correctly modified, and even with a high-precision threshold of 0.99 our Bayesian scheme yields more true positives than independent SVMs. Incidentally, independent SVMs cannot find any of the direct annotations (which are of particular interest for being the most specific) for the 88 genes, while the Bayes-corrected system is able to correctly predict some, even for the higher thresholds.

2.5.3 Predictions of Novel Proteins Involved in Mitosis

Our system makes predictions of function for many unannotated genes. Such predictions can be used to guide experimental verification of these functions. To assess the system's ability to make biologically relevant predictions that can be readily tested experimentally, we have examined in detail a small group of unknown genes with function predictions related to mitosis. All of these predictions were introduced by our hierarchical system; i.e., they were not predicted as positive by the individual classifiers prior to hierarchical combination.

YMR144W is an unknown genes that our system predicted to be involved in mitotic chromosome segregation. Indeed, when we examine a *Saccharomyces cerevisiae* strain lacking this gene, the cells show significant increase in chromosome segregation defects as compared to wild-type cells (F-score $= 6.6 \times 10^{-12}$), with multiple large budded cells with nuclei in the bud neck. This phenotype is consistent with that of ctf4Δ mutant, a strain lacking a known chromosome segregation gene (Miles and Formosa, 1992).

For the YOR315W gene, we make a novel prediction of mitotic spindle assembly. Yeast cells lacking this gene cannot properly separate their DNA during cell division; DNA is localized in elongated clumps along the spindle, mostly in the mother cells. These nuclear defects in large budded cells are significant with F-score of 2.8×10^{-12}. Furthermore, using anti-α-tubulin antibody, we demonstrate that YOR315WΔ cells have misaligned spindles during cell division, supporting our specific predictions of YOR315W function (Figures 2.5, 2.6).

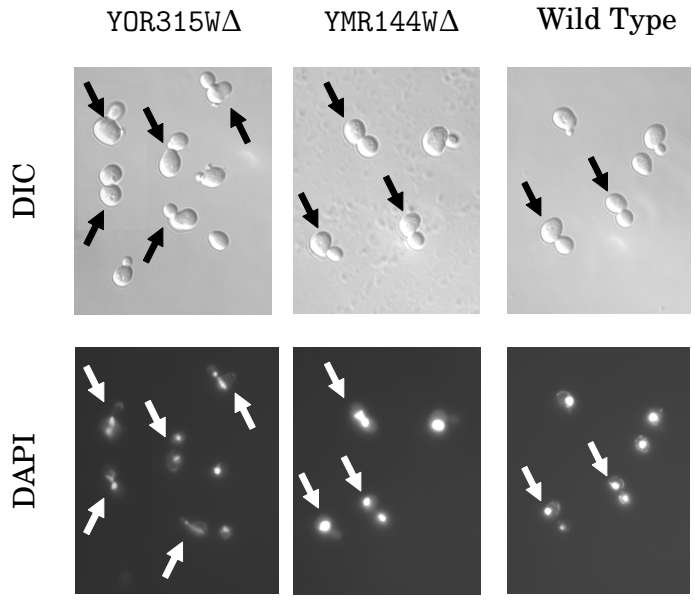


Figure 2.5: Laboratory validation of predictions. Yeast cells were stained, and photographed using Differential Interference Contrast (DIC) imaging or DAPI staining. Populations of cells lacking either YMR144W or YOR315W have a significantly higher number of large-budded cells with nuclear defects.

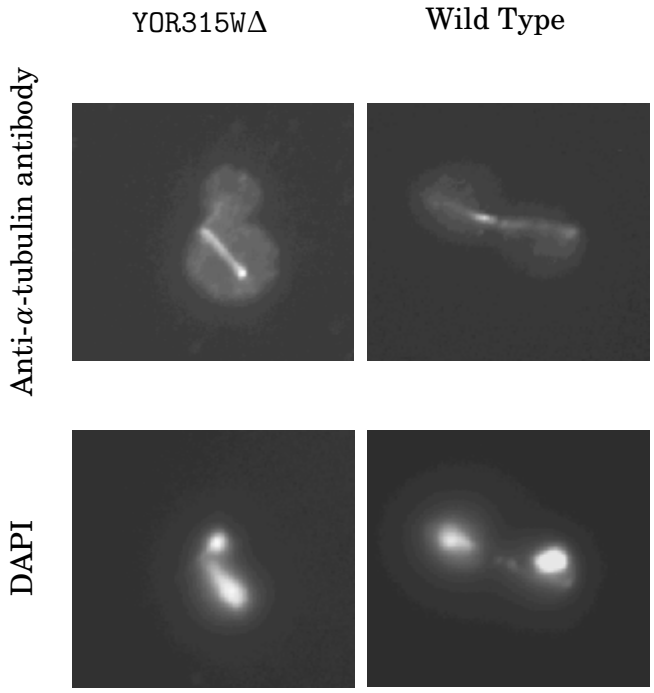


Figure 2.6: Cells were fixed and their spindles were visualized with an anti-α-tubulin antibody. Large-budded cells lacking YOR315W exhibit frequent misaligned spindles and nuclear defects.

YMR299C was predicted by our system to be involved in mitotic cell cycle. A recent study by Lee *et al.* (2005) has shown that this gene encodes a protein that is part of a dynein pathway that plays a critical role in mitosis, supporting our prediction. Another gene YOR315W was assigned by our system to a more general term cell cycle, and this prediction has some support (though not proof) from a study by Zhang *et al.* (2005) who also hypothesized its involvement in cell cycle by an independent method.

While these verifications do not prove that all novel predictions made by our system are accurate, they demonstrate that these predictions can be readily tested by directed experiments. Until experimental validation, these are predictions, not true functional assignments. However, the availability of such predictions can greatly accelerate the assignment of annotations by driving experimentation.

2.6 Discussion

Our proposed method of correcting inconsistent predictions in a multi-label class hierarchy was shown to improve performance for the majority of functions. In addition to eliminating inconsistencies, our Bayesian scheme also implicitly transforms unbounded real-valued classifier outputs into marginal probabilities and provides better calibration. Although our base classifiers of choice were SVMs, our system is a generic hierarchical ensemble method for leveraging accuracy that can work with any type of classifier without modification. Thus, given existing trained classifiers (e.g. from previous research on GO) our method can be directly applied to improve performance, needing only cross-validation results for output distribution modeling which are most often already available.

Prediction over the Gene Ontology is characterized by the virtual lack of negative examples. One path to explore as future work is applying density estimation algorithms, such as *Maximum Entropy* (Berger *et al.*, 1996), which work with only positive examples to estimate a probability density function over the input space. Unfortunately the lack of a real gold standard that includes negatives will still prevent comparing such an algorithm to our current approach on an equal basis.

Chapter 3

Shape Classification

Shape analysis in computer graphics focuses on analyzing and processing 2D and 3D geometric shapes. Basic physical representations of geometry, such as a collection of vertices and surfaces in coordinate space, are not directly suitable for semantic purposes such as object similarity or functional categorization, however. For instance, the 3D models for a sofa and a stool may not look similar, but an application may need to be aware that functionally both are seats.

Thus, a common problem in shape analysis involves assigning semantic meaning to geometry. Classifying a given shape into a pre-existing set of categories provides such meaning and relates it to similar objects. Applications vary widely, ranging from protein binding sites in molecular biology, to object recognition in computer vision. In addition to specifying the classes themselves, an application may define relationships among classes, commonly in the form of a general-to-specific hierarchy.

We applied the method of Bayesian Aggregation for the classification of shapes into a hierarchical set of classes to take advantage of the relationships represented by the hierarchy to improve classification accuracy and ranking performance. This work was in collaboration with Christopher R. DeCoro, and the results were published in Barutcuoglu and DeCoro (2006).

3.1 Data Sources and Processing

3D objects are commonly represented in digital form by specifying the polygons of the object's surface, where each polygon consists of a list of vertices. This geometric model is useful for the visual rendering of an object, but it does not lend itself directly to higher-level analyses, such as the similarity of one object to another. Even a simple rotation can produce a geometric model that is difficult to match to the original. Extracting higher-level feature vectors (called *shape descriptors*) from geometric models that capture qualitative similarities between objects is an area of active research in computer graphics.

Although these shape descriptors have generally been used for matching tasks, here we use them for classification. Matching only involves retrieving from a set of shapes the ones most similar to a given shape. Classification starts with a set of shapes that are pre-tagged with class labels, and aims to predict the most probable class for a newly presented shape. In this sense, classification extends matching by assigning semantic meaning to shapes and their similarities. For our training and testing examples, we use the models, descriptors and class assignments as provided in the Princeton Shape Benchmark (Shilane *et al.*, 2004), containing 1814 models in 198 classes, arranged in a hierarchy that is up to four levels deep.

For our experiments, we chose the Spherical Harmonic descriptor (SHD) (Kazhdan *et al.*, 2003; Kazhdan, 2004) as the shape descriptor. This method is designed to produce a feature vector for each object where the L_2 metric represents rotationally invariant similarity, and has been shown empirically to perform well for the task of shape matching; see for example the applications of SHD in Funkhouser *et al.* (2003); Min (2003); Shilane *et al.* (2004). Bayesian Aggregation is independent of the particular shape descriptor used, though final classification accuracy will naturally be higher for more discriminating descriptors. To form our dataset, each 3D geometric model in the Princeton Shape Benchmark is first converted into its SHD vector of 544 real-valued features, such that for any pair of such vectors, the L_2 metric provides a meaningful measure of similarity. We then use these shape descriptors as feature vectors in our classification algorithm.

As our shape descriptor is designed for matching by the L_2 metric, a natural classification method to use is k-Nearest Neighbors (kNN) which, given a feature

vector for an example to be classified, finds the k nearest feature vectors from the labeled training set, and chooses the most common value among the k labels (e.g. Alpaydin, 2004). The value k acts as a smoothing parameter, and larger values of k will tolerate more noise in the training set. In the case of $k = 1$ (1-NN) this is equivalent to choosing the label of the nearest training example. The assumption of kNN is that similarity between examples is represented accurately by the Euclidean distance of their feature vectors, for which our shape descriptors are specifically designed, making kNN a reasonable choice as a base classifier.

Since predictions from each kNN classifier are binary (as opposed to the continuous unthresholded SVM outputs in Chapter 2), the Bayesian network's base classifier output distributions $P(g_i|y_i)$ are multinomial, i.e., confusion matrices over held-out data, as described in Section 1.2.

3.2 Experimental Results

3.2.1 Classification Accuracy

We evaluated our method by first performing two-fold cross-validation on each class using kNN (the scarcity of positive examples in many classes prevented us from using a larger number of folds.) The value of k for each class was chosen as the best $k \in \{1,3,5,7,9\}$ using leave-one-out cross-validation accuracy within each training fold. 134 of 170 classes chose $k = 1$, and none chose $k = 9$. Then we combined the held-out confusion matrices from both folds to build our Bayesian network for the hierarchy. We used upwards hierarchical edges as in Chapter 2, and it yielded slightly better results than downward.

We inferred each example's most likely set of binary labels, as well as its marginal probabilities $P(y_i|g_1,\ldots,g_n)$ for each class. Comparing the binary most-likely labels to the base classifiers, 132 of 170 classes improved in accuracy. Figure 3.1 shows a scatterplot of accuracies for each class before and after Bayesian Aggregation. 38 remained the same, the base classifiers being already over 99% accurate.

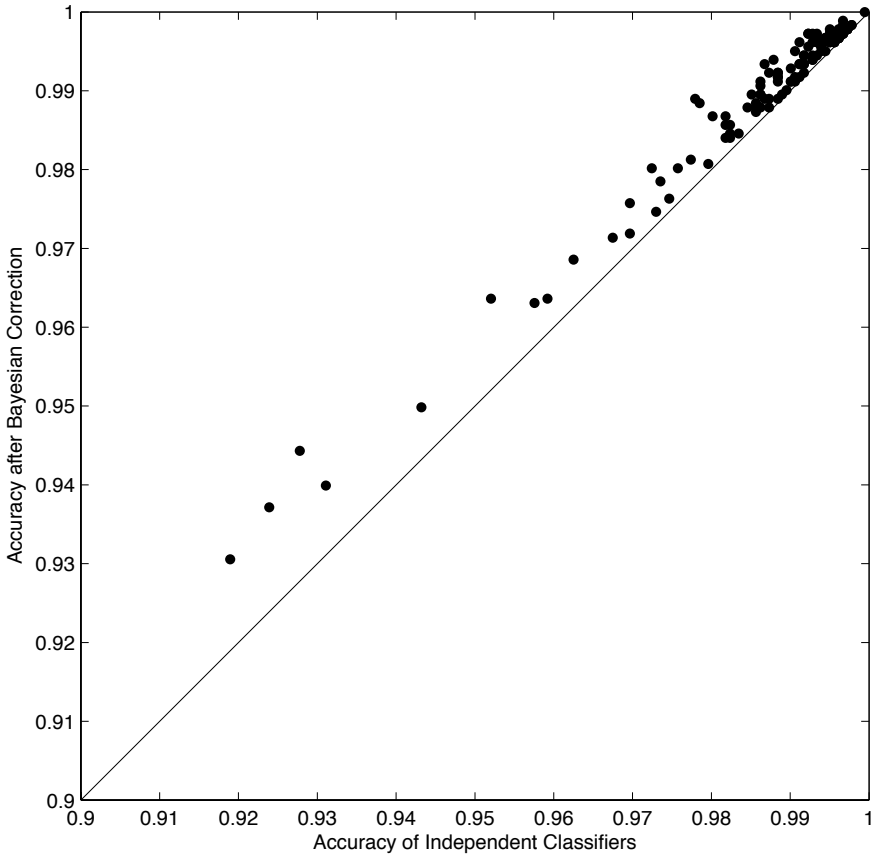


Figure 3.1: Scatterplot of accuracies on the Princeton Shape Benchmark classes, after versus before Bayesian Aggregation.

3.2.2 Ranking Performance

The predictions from the initial kNN classifiers are binary (for each class, we consider each example in the training set strictly as either a positive or negative example) whereas the marginal probabilities from the Bayesian network are real-valued. This allows us to threshold the Bayesian probabilities at different thresholds to obtain different binary predictions according to the utility model of different applications. For example, a particular application might have a very high cost for false-positives while false-negatives might be less of a concern, so in that case we would choose a high probability threshold and get fewer but more confident positive predictions. Instead of evaluating with a particular threshold, we use the AUC score (area under the ROC curve) which provides an evaluation over all possible thresholds.

Since the original kNN classifiers have binary outputs, they correspond to a single point on the ROC axes instead of a curve. However, for any two points (classifiers) on an ROC graph, one can obtain any other point on the connecting line segment by randomly selecting between the two classifiers' predictions with a Bernoulli distribution, so we compare the convex hull of the Bayes-net ROC curve to the "convex hull" of the naked kNN, which is its single point connected to $(0,0)$ and $(1,1)$.

The mean AUC score over the 170 independent kNN classifiers was 0.7004. 27 of these had an AUC of 0.5, meaning they were no better than random guessing. The mean AUC score of marginal Bayesian probabilities was 0.8369, reflecting a mean AUC improvement of 0.1365. All classes improved in AUC score, with the exception of one class (`torso`) which stayed the same at 0.9994. The scatter-plot of AUC scores before and after Bayesian Aggregation is shown in Figure 3.2.

Figure 3.4 shows an illustrative set of predictions for the example **m42**, which is a model of a flying eagle. Independent kNN classifier predictions exhibit an inconsistency at node `flying_creature` and false positives for `fantasy_animal` and `dragon`. Bayesian Aggregation yields $p = 0$ for the false positives, correctly fixes the inconsistency at `flying_creature`, and further yields $p = 0.454809$ for the leaf `flying_bird` of which **m42** actually is a member. Its sibling leaf nodes `standing_bird` ($p = 0.174441$) and `duck` ($p = 0.139243$) have significantly lower probability. After those, the next highest probability in the entire hierarchy is very

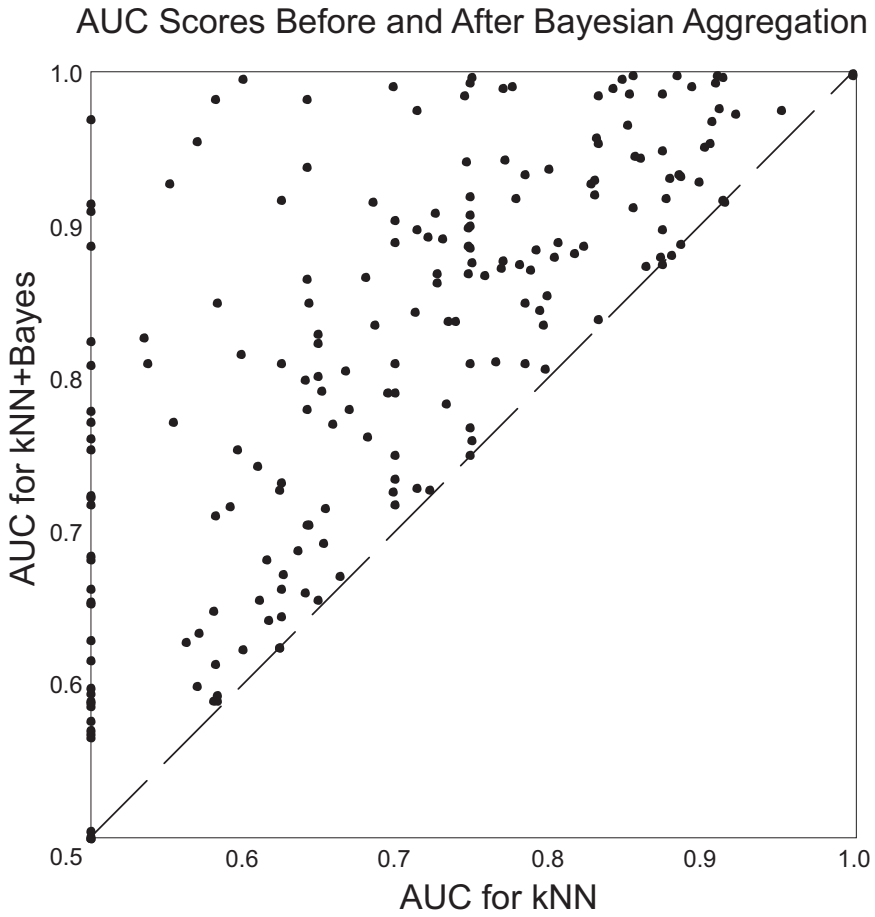


Figure 3.2: Per-class improvement in AUC score. For each class, we compare the AUC score of simple kNN versus the AUC score for the class after the application of Bayesian Aggregation. The dashed line represents no change in AUC, and the values range from 0.5 (random guessing) to 1.0 (perfect accuracy). As the figure shows, the use of Bayesian Aggregation results in a significant improvement in classification accuracy.

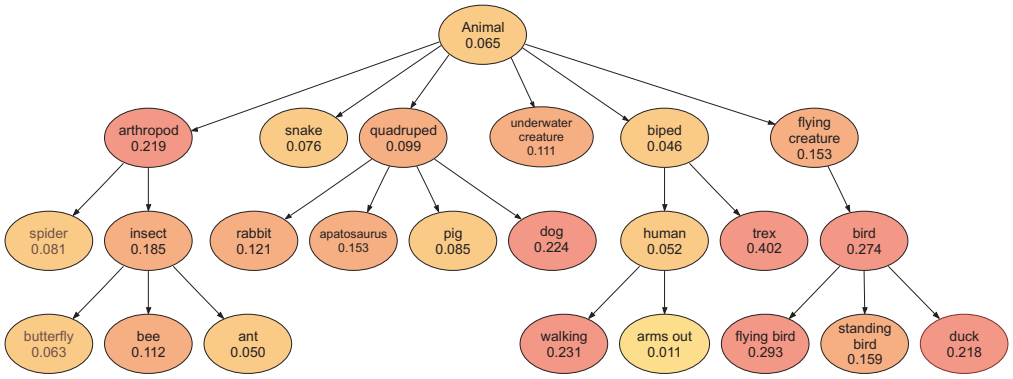


Figure 3.3: Hierarchical improvements in AUC score. For the "animal" branch of the hierarchy, we show the improvement in AUC score for each node, after applying our method. The darker nodes are those with larger increases in AUC score. As the figure shows, all of the nodes improved, many by significant amounts. This is only one slice of the entire Princeton Shape Benchmark, but other branches display similar results.

low ($p = 0.038353$). This example demonstrates that our method extends beyond only fixing inconsistencies and is able to improve classifiers anywhere in the hierarchy. For further examples of this, we show a portion of the hierarchy in Figure 3.3, and the amount of improvement in AUC score for each node. All of the nodes shown improved, many by significant amounts. Note for example the improvement in the `bird` branch, which corresponds to the classifiers used for Figure 3.4.

3.2.3 Comparison to Heuristic Correction

To evaluate the improvement that is due to our method, we also compared our results to two heuristic methods for hierarchical inconsistency resolution. One such method propagates negative predictions downward, so that if a node predicts that an example is not a member, its children are also presumed to predict that the example is not a member, disregarding their actual classifier predictions (Cesa-Bianchi *et al.*, 2004). While this does prevent inconsistencies, it showed very poor performance: 88 of the classes decreased in performance, and the average change in AUC score is -0.05. Alternatively, the heuristic of propagating positive predictions upward, favoring child node predictions, increased the average AUC score by only 0.01, significantly less than the improvement by our method.

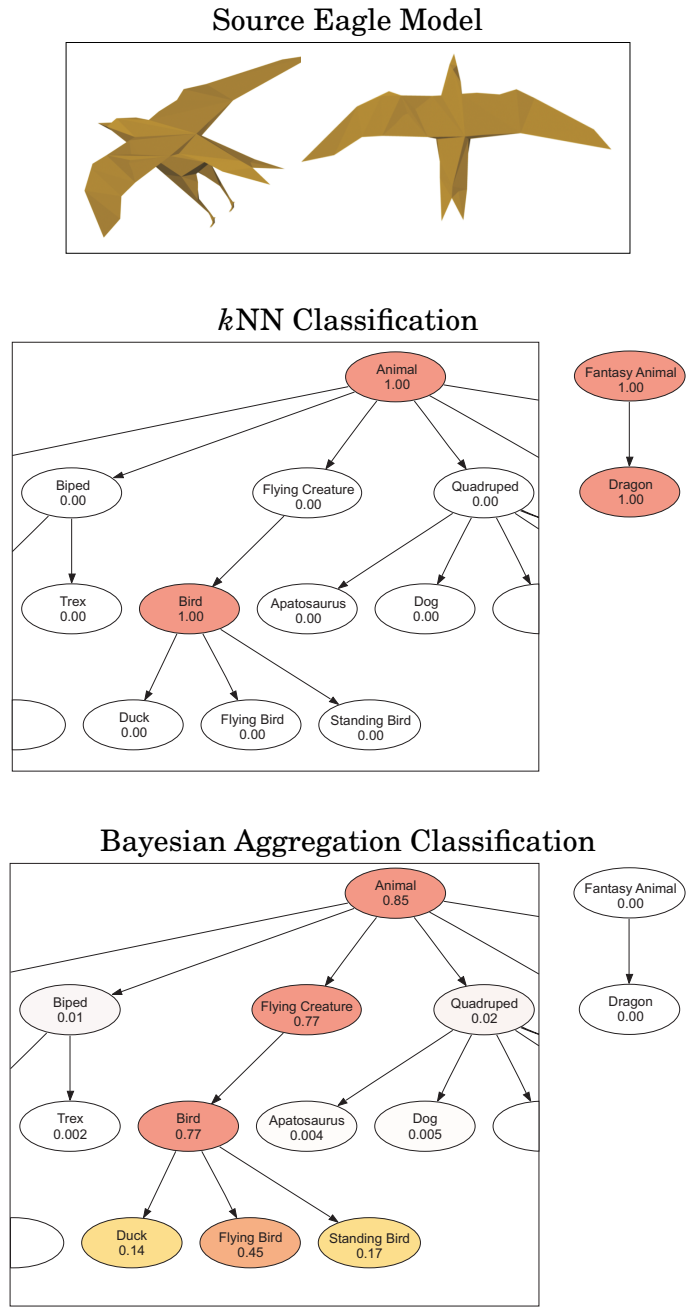


Figure 3.4: This model of a flying eagle should be correctly classified as a `flying_bird` and, for consistency, all of `flying_bird`'s ancestor classes. The standard k-Nearest-Neighbors algorithm classifies the model as a bird and an animal, but not as a flying bird or a flying creature; it also incorrectly identifies the eagle as a fantasy animal and a dragon. Our Bayesian Aggregation algorithm, using the results of the kNN classifiers, produces a more accurate prediction for the `flying_bird` class, resolves the inconsistency with the `flying_creature` class, and rules out the possibility that the model is a dragon.

3.3 Discussion

Bayesian Aggregation improved classification accuracy and ranking performance at all levels of the Princeton Shape Benchmark hierarchy over kNN base classifiers.

The incarnation of the Bayesian Aggregation model we used on this dataset did not enforce the single-branch nature of the Princeton Shape Benchmark; i.e., it allowed multiple-branch positive predictions. We chose to leave it as such in this evaluation to keep comparisons to kNN fair. Implementing the previously mentioned single-branch variant of Bayesian Aggregation may further improve performance.

Chapter 4

Musical Genre Classification

Many musical concepts are inherently hierarchical. Some notion of hierarchy is implicitly or explicitly at play in concepts such as genre and mood (which have overlapping coarse and fine categories), instrument timbre (which is grouped by instrument families), and meter. When the hierarchy has been accommodated in automatic classification of these concepts, it has generally been in quite simple ways, such as classifying with a single root classifier and then using the output of each parent to choose a child classifier in a top-down approach, such as McKay (2004), or in ways that are acutely tailored to the learning task or classification method at hand (e.g. Klapuri *et al.*, 2006; Rauber *et al.*, 2002).

Genre classification has been a popular task in music information retrieval since originally posed by Tzanetakis *et al.* (2001). The basic objective is to predict the musical genre of a song, based on features extracted from the audio or a symbolic representation of the song. Genre is a culturally relevant and practically useful concept, and genre classification systems have the potential to be quite useful in organizing and allowing efficient access to music databases, as shown by McKay and Fujinaga (2006). Further, the same work showed that the use of multiple class assignments and user-specified ontological structure is beneficial in principle, both of which are inherently supported by Bayesian Aggregation.

We demonstrate the performance of Bayesian Aggregation on classification into a hierarchy of genres, and we show that it allows for a significant improvement in classification accuracy compared to the predictions of independent classifiers without aggregation, and other existing methods, while guaranteeing hierarchi-

cally consistent predictions. We also demonstrate that the outputs of such a classification system may offer improvements to similarity search systems built on genre classifiers. This work was in collaboration with Christopher R. DeCoro and Rebecca Fiebrink, and the results were published in DeCoro *et al.* (2007).

4.1 Data Sources and Processing

The annual Music Information Retrieval Evaluation Exchange (MIREX) competition is an event sponsored by the International Symposium on Music Information Retrieval (ISMIR) to evaluate the state-of-the-art in methods for extracting high-level information from musical examples. In 2005, the contest featured both audio and symbolic (MIDI[1]) genre classification tasks, in which 950 individual songs were to be classified into a given 55-class hierarchy of musical genres, with 38 leaf classes (McKay and Fujinaga, 2005).

The winning participants of the symbolic genre classification task employed the Bodhidharma MIDI classification system, originally presented in McKay and Fujinaga (2004) and expanded in McKay (2004). Bodhidharma extracts 111 high-level features such as instrumentation, rhythm, dynamics, and chords. It also considers hierarchical relationships via a top-down classification approach, where classifier outputs at each branch of a hierarchy determine which child classifiers will be used to further refine the classification.

In our evaluation, we used the MIREX 2005 symbolic genre dataset and the same features as extracted by Bodhidharma.

4.2 Experimental Results

4.2.1 Genre Classification

In the MIREX 2005 symbolic genre classification dataset, each training example belongs to very few nodes, in comparison to all the other nodes for which it counts

[1]MIDI (Musical Instrument Digital Interface) is a protocol that enables electronic musical equipment to communicate. As opposed to an audio signal, it is a stream of digital event messages such as the pitch and duration of musical notes.

as a negative example. Consequently, the number of negative examples is disproportionately larger than the number of positives for each node. As a performance measure on such data, the default definition of accuracy, where a false positive and a false negative are deemed to carry equal weight, can be misleading. For instance, a classifier that unconditionally predicts everything as negative will have very high accuracy and appear desirable, although it provides no practical benefit. Instead, a skew-insensitive performance measure is necessary, both for optimizing during training and for analyzing results, which will penalize errors on the few positive examples proportionally more highly than errors on the ample negatives. AUC score is one such measure. Another one, used by previous work on this dataset, is *skew-insensitive accuracy*, defined as the average of sensitivity (accuracy on positive examples) and specificity (accuracy on negative examples):

$$0.5\frac{\textit{true positives}}{\textit{true pos.} + \textit{false neg.}} + 0.5\frac{\textit{true negatives}}{\textit{true neg.} + \textit{false pos.}}.$$

On the MIREX 2005 symbolic genre classification dataset, we trained linear SVMs using the *SVMlight* software Joachims (1999) with the appropriate cost factor setting to compensate for class skew (as the features are not Euclidean, kNN classification was less applicable). We used three-fold cross-validation, obtaining three SVMs for each class, and a held-out prediction in that class for each example from the SVM that it was held out from. The Bayesian network was then constructed using these distributions as described in Section 1.2, using downward hierarchical edges due to the high branching factor of the hierarchy.

We computed marginal probabilities for the hierarchically consistent hidden labels using Bayesian inference. The average skew-insensitive accuracy over all 55 classes was 76.8% for independent SVMs, and 85.1% after Bayesian Aggregation thresholded at $p = 0.5$. Figure 4.1 shows a scatterplot of the individual classes before and after aggregation.

To compare our results to previous work on this dataset, we also computed the "raw accuracy" statistic reported in the MIREX 2005 contest results, which relies on the one-leaf-only nature of song labels in this dataset and picks as genre prediction the leaf node with the highest output. Under this multi-class single-label criterion, the winning Bodhidharma system scored 46.1; the median raw accuracy was 41.0.

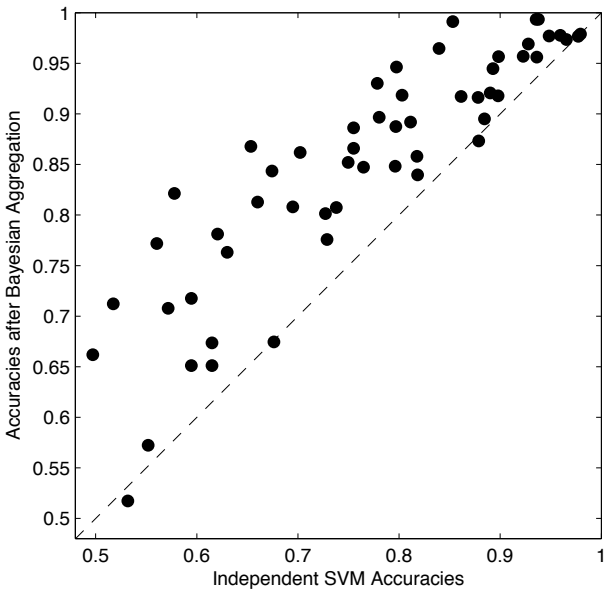


Figure 4.1: Scatterplot of MIREX skew-insensitive accuracies after versus before Bayesian Aggregation.

Using the same feature set, our SVM base classifiers achieved 56.0 raw accuracy. While already improving significantly relative to previous state-of-the-art, after applying Bayesian Aggregation accuracy improved to 60.1, compared to the top MIREX 2005 contest entries in Table 4.1. Considering that this is a multi-class single-label (one-of-m) evaluation of choosing the correct one out of 38 leaf nodes, where a random guess would have less than 3% accuracy, both the improvement over previous results and the improvement over independent SVMs by Bayesian Aggregation are significant.

4.2.2 Similarity Search

A related application in music information retrieval is to search for songs "similar" in genre to a query song, or equivalently, rank all songs in a database by similarity to the query song using genre classification as a measure. For our experiments, we defined the similarity of two songs as the number of their equal binary labels in the hierarchy, which decreases as the path distance of their leaf classes in the hierarchy increases. We computed the "true similarity" of every pair of songs using

Algorithm	Raw Accuracy
Bayesian Aggregation	60.1%
Independent SVMs	56.0%
Bodhidharma	46.1%
Basili et al. (NB)	45.0%
Basili et al. (J48)	41.0%
Li	39.8%
Ponce de Leon & Inesta	15.3%

Table 4.1: Independent SVMs and Bayesian Aggregation compared to MIREX 2005 contest entries by single-label multi-class "raw accuracy".

the actual labels, and the predicted similarities from both independent SVMs and subsequently the Bayes-aggregated predictions. To avoid selecting an arbitrary threshold, classes along the branch of the maximum-confidence leaf were selected as the positive predictions for each example. Using each song as the query, all other songs were sorted by similarity, and the top predicted results were compared to the top results as given by true similarity. Across all examples, of the 100 most-similar songs by "true similarity", an average of 52% were retrieved by independent SVMs, improved to 62% with Bayesian Aggregation. Similarly, of the top 50, an initial 46% retrieval rate improved to 52% after Bayesian Aggregation.

West and Lamere (2007) have used Euclidean distance between genre classifier soft outputs to perform similarity computation for playlist generation and collection visualization, but without regard to hierarchical class organization. Our results suggest that applying Bayesian Aggregation to such a system could improve on their approach as well.

4.3 Discussion

Definitions of musical genres are inherently ambiguous and overlapping, making continuous-valued interpretations of membership meaningful. As our experiments show, considering genre relationships during classification allows exploitation of this fact while avoiding the limitations of traditional, discrete classifiers. Post-aggregation predictions are significantly improved, both compared to their base

classifiers, as well as to previous work. Especially in datasets such as this one, with a limited number of examples, our algorithm is able to effectively expand the training set for each class, by allowing for soft assignments to related classes.

This has the important consequence that the performance of our algorithm has a degree of independence from hierarchy topology; leaf-nodes can be finely specified, even when this leads to few training examples per class, as soft assignment to related classes can provide additional contextual information for classification. Conversely, arbitrarily broad high-level classes can exist in the genre tree without loss of classification performance, even when such nodes are difficult to classify directly; in both cases, Bayesian Aggregation implicitly adapts to the hierarchy.

This results from the principled manner in which we aggregate results; naïve heuristics tend to fail in such cases. For example, binary classifiers trained on arbitrarily broad root classes whose members share little similarity in feature space would lead to near-random classifications of novel examples, while using highly specified leaf-nodes with few examples leads to overfitting, and its resulting loss of classification generality. However, Bayesian Aggregation is aware when base classifiers for a given node are generalizing poorly, and will make final predictions using related higher-performance classes.

Part II

Input Structure: Sequence Classification

For the problem of classifying fixed-length noisy observation sequences, we build an integrated model that combines sequence-wide and position-specific features to cooperatively denoise observations by sequential correlation and learn a sparse set of sequence positions that are significant in predicting the sequence class. We provide an efficient L1-regularized discriminative training algorithm for the model. Our experiments on synthetic data and real genomic cancer prediction data show that our method is superior, both in prediction accuracy and relevant feature discovery, to the common practice of preprocessing with a purely homogeneous sequence model and then learning with a purely non-sequential algorithm.

For sequence classification problems, where each example has a large number of sequentially-correlated noisy inputs and one classification label for the whole sequence, models developed for speech or text tasks, such as Hidden Markov Models, generally learn a small number of position-invariant parameters to reduce model complexity and to accommodate the variable-length aspect of such data. In scenarios where the sequence length is fixed and position in the sequence is known to be significant for classification, these assumptions lead to a loss of information. Conversely, applying a conventional non-sequential machine learning algorithm directly on the noisy input features ignores the sequential information and is likely to suffer from overfitting.

One such scenario arises in predicting clinical properties of cancer by microarray-based comparative genomic hybridization (array-CGH) measurements. The heterogeneity of cancer cannot always be recognized by tumor morphology, but may be reflected by the underlying genetic aberrations. In particular, genes are observed to undergo changes in copy number from their normal value (two copies for genes on non-sex human chromosomes) and these changes are known to be associated with cancer. Array-CGH methods provide high-throughput data to determine genetic copy numbers, but determining the clinically relevant copy number changes remains a challenge. Non-sequential classification methods for linking recurrent alterations to clinical outcome ignore sequential correlations in selecting relevant features. Conversely, arbitrary-length sequence classification methods can only model overall copy number instability, without regard to any particular position in the genome.

In Chapter 5, we present the Heterogeneous Hidden Conditional Random Field,

a new classifier for fixed-length noisy observation sequences, which combines sequence-wide and position-specific features to cooperatively denoise observations by sequential correlation and learn a sparse set of sequence positions that are significant in predicting the sequence class. Applied to the motivating problem of array-CGH analysis, it jointly classifies tumors, infers copy numbers, and identifies clinically relevant positions in recurrent alteration regions, as demonstrated on synthetic datasets modeled from real human breast cancer data. We provide an efficient L_1-regularized discriminative training algorithm, which notably selects a small set of candidate genes most likely to be clinically relevant and driving the recurrent amplicons of importance. Our method thus provides unbiased starting points in deciding which genomic regions and which genes in particular to pursue for further examination.

In Chapter 6, HHCRF is applied to real genomic cancer prediction datasets on breast (two datasets), bladder, and uveal melanoma tumors. The results show that our method is superior, both in prediction accuracy and relevant feature discovery, to existing methods. We also demonstrate that it can be used to generate novel biological hypotheses for breast cancer.

By capturing the sequentiality as well as the locality of changes, our integrated model provides better noise reduction, and achieves more relevant gene retrieval and more accurate classification than existing methods.

Chapter 5

Heterogeneous Hidden Conditional Random Fields

One of the major challenges in the management of cancer is its heterogeneity: cancer patients with the same stage of disease can have markedly different treatment responses and survival outcomes. This heterogeneity cannot always be recognized by tumor morphology, but may reflect the complexity of underlying genetic aberrations.

Depending on the instability present in the tumor and the selection environment, tumor cells may acquire alterations, called *aneuploidies*, ranging from large segments with single copy number alterations to narrow homozygous deletions or high level amplifications (Heim and Mitelman, 1989). Array comparative genomic hybridization (*array-CGH*) is a technique by which it is possible to detect and map genetic changes that involve gain or loss of segments of genomic DNA. Downstream analyses involve classifying the samples and finding copy number alterations that are associated with known biological markers. Finding regions of recurrent aneuploidy, called *amplicons*, for a tumor type can reveal candidate cancer genes that have undergone selection for altered expression associated with tumor growth (Albertson *et al.*, 2000; Snijders *et al.*, 2005; Brown *et al.*, 2006).

Although recent developments have enabled experiments to measure copy number on a genome scale with high genomic resolution, individual point measurements are still noisy, making the crucial separation of signal from noise difficult. A point deviation in array-CGH measurements can be due to a true difference in

copy number, or a measurement artifact. A key factor for filtering out noise is to note the strong sequential correlation in copy numbers throughout the genome, and numerous methods have been successfully applied to sequentially detect regions of constant aneuploidy (see Lai *et al.*, 2005, for a survey).

Performing sequential aneuploidy detection on an individual genome, however, with no regard to recurrent patterns across different genomes, ignores correlations among similar tumor samples. In particular, if genomes in a sample set have been differentially labeled with a clinical target attribute (e.g. grade, subtype, recurrence, survival), then a *supervised* (label-aware) analysis can focus directly on the potentially clinically relevant patterns of aneuploidy, rather than relying solely on unsupervised sequential correlation. In addition to providing a direct predictive model for clinical diagnostic or prognostic applications, a supervised model can distinguish biomarker genes possibly relevant to tumor development from clinically irrelevant copy number changes.

Several studies have demonstrated the importance of supervised methods on CGH data for tumor classification, prognosis, and candidate gene search (see van Beers and Nederlof, 2006, for a recent survey). However, the all-purpose predictive models that have been used for analysis, such as naïve-Bayes (Wessels *et al.*, 2002), Support Vector Machines (Jonsson *et al.*, 2005) and various conventional statistics, all ignore the sequential information captured by unsupervised aneuploidy detection methods. This simplistic order-insensitive interpretation of array-CGH data is likely to cause the statistical bias of known correlations to be accounted for as variance, discarding clinically relevant signals as noise.

Only the recent H-HMM (Shah *et al.*, 2007) and Fused SVM (Rapaport *et al.*, 2008) models demonstrate the benefits of supervised sequential array-CGH analysis over many tumor samples for identifying clinically important regions of aneuploidy, but identifying the causal genes within these amplicons remains an open challenge. Thus, no existing method can perform a supervised identification of the clinically relevant genes in the process of extracting copy number profiles for tumor classification.

In this work, we present a method that combines a sequential representation of copy numbers with outcome-related gene selection to build a supervised predictor for a clinical variable by selecting clinically relevant genes that "drive" recurrent

amplicons. Our method combines the sequential de-noising and the classification aspects in one integrated supervised architecture, so that they can cooperatively learn a better overall predictive model, without loss of relevant signal to either. We provide an efficient, regularized training algorithm that finds a sparse interpretable solution that directly identifies cancer-related genes. We extensively evaluate this method on both synthetic data and four biological datasets of breast, uveal melanoma, and bladder tumors. We demonstrate that our method is substantially better than state-of-the-art methods and can be used to make new biological and clinically relevant hypotheses.

5.1 Probabilistic Model

Our model explicitly represents the discrete copy number at a probe location as a latent random variable. Each array-CGH measurement is an observed variable sampled exclusively from its underlying copy number's measurement level mean with a random noise distribution. The sequentiality of copy numbers is represented by pairwise correlations between adjacent latent variables.

The entire sequence has a clinical label to be predicted, which in our model is affected directly by the discrete copy number profile. The real-valued observations relate to the sequence label only through the latent copy numbers variables, making the sequence label conditionally independent of the observed measurements given the copy numbers. This decoupling reflects our explicit modeling of the observations as noisy representations of copy number levels: if we already knew the true copy numbers, the noisy observations would no longer be relevant to the prediction label. The model is illustrated in Figure 5.1.

Furthermore, we assume the sequence label to be directly affected by only a small subset of positions in the copy number profile. Part of the learning process is the selection of these positions, the cancer-related loci, by applying a sparse regularization on the $c_i - s$ edges in Figure 5.1.

The method first learns the model's parameters on a training dataset of array-CGH sequences with known sequence labels. A regularization parameter determines how many cancer-related positions are selected. Once the model is built, it can be used to predict the most likely sequence labels for new sequences. Discrete

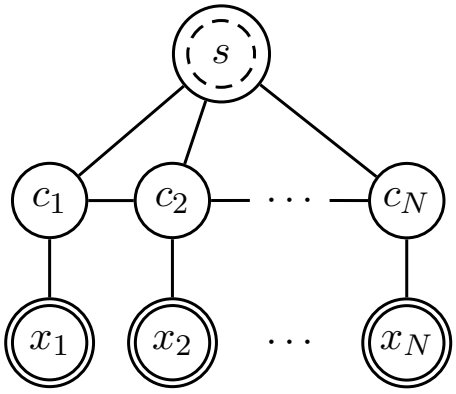


Figure 5.1: The Heterogeneous Hidden Conditional Random Field model. The variables x_i are observed, c_i are hidden, and the sequence label s is only observed during training. An exponential model for $p(s,\mathbf{c}|\mathbf{x})$ is tuned to maximize the class-conditional likelihood $p(s|\mathbf{x})$ of training data.

copy number profiles can also be queried as the most likely assignments of the latent copy number variables given observed data. For evaluations, a cross-validation or held-out samples protocol is used.

For a particular training example, let s be the clinical label of the whole sequence, let x_i denote the observation, and c_i the latent variable at position $i \in \{1,\ldots,N\}$ whose value can be one of C different copy number states.

Given the observations $\mathbf{x}$ for an example, we use an exponential model for the conditional probability of the other variables:

$$p_{\boldsymbol{\theta}}(s,\mathbf{c}|\mathbf{x}) = \frac{1}{Z_{\boldsymbol{\theta}}(\mathbf{x})} \exp\left(\boldsymbol{\theta} \cdot \mathbf{f}(s,\mathbf{c},\mathbf{x})\right) \tag{5.1}$$

where $Z_{\boldsymbol{\theta}}(\mathbf{x})$ is a normalization factor, $\boldsymbol{\theta} = (\boldsymbol{\rho},\boldsymbol{\lambda},\boldsymbol{\omega})$ are the model parameters, and $\mathbf{f}$ is a vector of features. In principle, the features could be any relevant real-valued functions of s, $\mathbf{c}$, and $\mathbf{x}$, but in our model, we consider features of only three types

corresponding to the three edge types in Figure 5.1. Thus,

$$\begin{aligned}\boldsymbol{\theta}\cdot\mathbf{f}(s,\mathbf{c},\mathbf{x}) =& \boldsymbol{\rho}\cdot\sum_{i=2}^{N}\mathbf{f}_{pair}(c_{i-1},c_i,s)\\ &+\sum_{i=1}^{N}\boldsymbol{\lambda}_i\cdot\mathbf{f}_{local}(c_i,s)\\ &+\boldsymbol{\omega}\cdot\sum_{i=1}^{N}\mathbf{f}_{obs}(c_i,x_i).\end{aligned} \tag{5.2}$$

The pairwise features $\mathbf{f}_{pair}$ and the corresponding parameters $\boldsymbol{\rho}$ model the sequence-wide correlation of adjacent nodes for each class. The local features $\mathbf{f}_{local}$ and their parameters $\boldsymbol{\lambda}_i$ model the correlation of latent variable c_i and the label s. And the observation features $\mathbf{f}_{obs}$ and their parameters $\boldsymbol{\omega}$ model the correlation of latent variable c_i and its noisy observation x_i.

For discrete latent variables and class label, the feature functions $\mathbf{f}_{pair}$ and $\mathbf{f}_{local}$ are typically defined to be 1 for a particular combination of arguments and 0 otherwise. The pairwise parameters $\boldsymbol{\rho}$ then correspond to (unnormalized) log-probabilities of a homogeneous HMM's hidden state transitions. For real valued observations, $\mathbf{f}_{obs}(c,x)$ can be defined as $(1,x,x^2)$ if $c = c'$ (and zero otherwise) for each latent variable value c', the sufficient statistics for Gaussian distributions.

The position-dependent local parameters, which make the model heterogeneous, allow the model to interpolate between a homogeneous sequence-wide hypothesis and one that ignores correlations. If all local parameters are made zero, the model is a fully homogeneous random field, and classification only depends on sequence-wide stability of latent state. Conversely, if they are unconstrained and allowed to overpower the pairwise component, classification will depend almost fully on them, and the model will be akin to logistic regression. In our model, we constrain the L_1 norm of the local parameters $\boldsymbol{\lambda}$ to adjust this tradeoff, which also encourages sparsity and results in an interpretable solution.

5.2 Training

The model is trained discriminatively, minimizing the conditional negative log-likelihood of labels over the empirical distribution $\tilde{p}(s,\mathbf{x})$ of the training data:

$$\mathscr{L}_{\boldsymbol{\theta}} = -\sum_{s,\mathbf{x}} \tilde{p}(s,\mathbf{x}) \log p_{\boldsymbol{\theta}}(s|\mathbf{x}) \tag{5.3}$$

subject to the regularization constraint $||\boldsymbol{\lambda}||_1 \leq \beta$.

The L_1 constraint encourages a sparse solution in feature weights, as in LASSO (Tibshirani, 1996), and serves to select only a few local features (the clinically important loci) among the many positions. Note that this regularization does not include the transition and observation parameters, as their numbers are already small compared to the (often very large) sequence length.

We use a gradient-based procedure to solve the optimization problem. The partial derivative of the objective loss with respect to any parameter θ_k is:

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial \theta_k} &= \sum_x \tilde{p}(x) \sum_{s,\mathbf{c}} p_{\boldsymbol{\theta}}(s,\mathbf{c}|\mathbf{x}) f_k - \sum_{s,\mathbf{x}} \tilde{p}(s,\mathbf{x}) \sum_{\mathbf{c}} p_{\boldsymbol{\theta}}(\mathbf{c}|s,\mathbf{x}) f_k \\ &= \mathbb{E}_{\tilde{p}(\mathbf{x}) p_{\boldsymbol{\theta}}(s,\mathbf{c}|\mathbf{x})}[f_k] - \mathbb{E}_{\tilde{p}(\mathbf{x},s) p_{\boldsymbol{\theta}}(\mathbf{c}|s,\mathbf{x})}[f_k]. \end{aligned} \tag{5.4}$$

Although $p_{\boldsymbol{\theta}}(s|\mathbf{x})$ in (5.3) and the expectations in (5.4) call for marginalizing $p_{\boldsymbol{\theta}}(s,\mathbf{c}|\mathbf{x})$ as defined in (5.1) over the exponentially many value combinations of the latent variables $\mathbf{c}$, a dynamic programming solution exists, similar to the forward-backward procedure for HMMs (e.g. Alpaydin, 2004), scaling linearly with sequence length:

$$\begin{aligned} M_i(c_i, c_{i+1}, s) &\equiv \exp\left[\boldsymbol{\rho} \cdot \mathbf{f}_{pair}(c_i, c_{i+1}, s)\right] \\ N_i(c_i, s, \mathbf{x}) &\equiv \exp[\boldsymbol{\lambda}_i \cdot \mathbf{f}_{local}(c_i, s) + \boldsymbol{\omega} \cdot \mathbf{f}_{obs}(c_i, x_i)] \\ A_1(c_1, s, \mathbf{x}) &\equiv 1 \\ A_i(c_i, s, \mathbf{x}) &\equiv \sum_{c_{i-1}} M_{i-1}(c_{i-1}, c_i, s) N_{i-1}(c_{i-1}, s, \mathbf{x}) A_{i-1}(c_{i-1}, s, \mathbf{x}) \\ B_n(c_n, s, \mathbf{x}) &\equiv 1 \\ B_i(c_i, s, \mathbf{x}) &\equiv \sum_{c_{i+1}} M_i(c_i, c_{i+1}, s) N_{i+1}(c_{i+1}, s, \mathbf{x}) B_{i+1}(c_{i+1}, s, \mathbf{x}) \end{aligned}$$

$$
\begin{aligned}
p_{\boldsymbol{\theta}}(s, c_{i-1}, c_i|\mathbf{x}) &= \frac{1}{Z(\mathbf{x})} \sum_{c_1 \cdots c_{i-2}} \sum_{c_{i+1} \cdots c_n} \prod_{i=1}^{n-1} M_i(c_i, c_{i+1}, s) \prod_{i=1}^{n} N_i(c_i, s, \mathbf{x}) \\
&= \frac{1}{Z(\mathbf{x})} M_{i-1} N_{i-1} N_i A_{i-1} B_i \\
p_{\boldsymbol{\theta}}(s, c_i|\mathbf{x}) &= \frac{1}{Z(\mathbf{x})} \sum_{c_1 \cdots c_{i-1}} \sum_{c_{i+1} \cdots c_n} \prod_{i=1}^{n-1} M_i(c_i, c_{i+1}, s) \prod_{i=1}^{n} N_i(c_i, s, \mathbf{x}) \\
&= \frac{1}{Z(\mathbf{x})} N_i A_i B_i
\end{aligned}
$$

where $Z(\mathbf{x})$ is a normalization factor. Dynamic programming is achieved in the computations of A_i and B_i, taking only a linear number of steps in sequence length n, and thus all necessary feature expectations can be efficiently computed using these marginalized conditional probabilities.

5.3 Gradient LASSO

To solve the problem while satisfying the regularization constraint $||\boldsymbol{\lambda}||_1 \leq \beta$, we incorporate the Gradient LASSO algorithm (Kim and Kim, 2004), with a minor modification.

Gradient LASSO is an interior point method for optimizing a differentiable function subject to L_1 constraints. It maintains an explicitly sparse current solution, alternating between a coordinatewise gradient step, which may add a new non-zero parameter, and a multivariate gradient step over the non-zero parameters, which may make one of them zero. The constraints are always kept satisfied, by starting inside the constraint simplex and bounding step sizes. When the current parameters satisfy the constraint by equality and local gradient descent is about to violate it, the gradient is projected onto the boundary, and linearity of L_1 constraint boundaries make line search along the boundary possible.

Our version of Gradient LASSO (summarized in Algorithm 2) differs slightly from the original presented by Kim and Kim (2004): in the deletion step, if the current solution is not on the constraint boundary, we use a less conservative maximum step size Δ to accelerate learning.

Algorithm 2 Gradient LASSO (modified)

Objective: min $L(\boldsymbol{\lambda})$ s.t. $||\boldsymbol{\lambda}|| \leq \beta$

repeat

Addition step:

Compute gradient $\nabla = (\partial L/\partial\lambda_1, \ldots, \partial L/\partial\lambda_d)$

Choose coordinate $k = \arg\max_i |\nabla_i|$

$h_k = -\beta \text{sign}(\nabla_k)$; $h_i = 0$ for all $i \neq k$

$\hat{\alpha} = \arg\min_{\alpha \in [0,1]} L((1-\alpha)\boldsymbol{\lambda} + \alpha \mathbf{h})$

$\boldsymbol{\lambda} \leftarrow (1-\hat{\alpha})\boldsymbol{\lambda} + \hat{\alpha}\mathbf{h}$

Deletion step:

Compute gradient $\nabla = (\partial L/\partial\lambda_1, \ldots, \partial L/\partial\lambda_d)$

Let $\boldsymbol{\sigma} = \{i : \lambda_i \neq 0\}$

Let $p = \nabla \cdot \mathbf{z}$ where $z_i = \text{sign}(\lambda_i)$

$$h_j = \begin{cases} 0 & \text{if } j \notin \boldsymbol{\sigma} \\ -\nabla_j + p z_j / |\lambda_j| & \text{if } j \in \boldsymbol{\sigma}, p < 0 \text{ and } ||\boldsymbol{\lambda}||_1 = \beta \\ -\nabla_j & \text{if } j \in \boldsymbol{\sigma}, \text{otherwise} \end{cases}$$

$$\Delta = \begin{cases} \min_{j \in \boldsymbol{\sigma}} \{-\lambda_j / h_j : \lambda_j h_j < 0\} & \text{if } ||\boldsymbol{\lambda}||_1 = \beta \\ (\beta - ||\boldsymbol{\lambda}||_1)/||\mathbf{h}||_1 & \text{if } ||\boldsymbol{\lambda}||_1 < \beta \end{cases}$$

$\hat{\alpha} = \arg\min_{\alpha \in [0,\Delta]} L(\boldsymbol{\lambda} + \alpha \mathbf{h})$

$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \hat{\alpha}\mathbf{h}$

until converged

5.4 Unconstrained Parameters

The unregularized parameters of our model $(\boldsymbol{\rho}, \boldsymbol{\omega})$ are optimized after each two-step Gradient LASSO iteration, using the gradient-based L-BFGS algorithm (Nocedal, 1980), a limited-memory quasi-Newton method for unconstrained optimization, while the regularized parameters $\boldsymbol{\lambda}$ are kept unchanged.

Note that the unconstrained optimization step causes the constrained problem objective $L(\boldsymbol{\lambda})$ to change between iterations, and therefore the optimality of its current solution. The two-step Gradient LASSO algorithm, by adding newly relevant features and deleting obsolete features as necessary, is able to robustly cope with this concept drift without compromising sparsity, which would not have been possible with strictly growing or shrinking algorithms.

In our implementation, we constrained $\boldsymbol{\rho}$ to be diagonal, and used k-means clustering to initialize $\boldsymbol{\omega}$. If the output distributions $P(x|c)$ are assumed to be Gaussian, unsupervised clustering of all observations in all sequences provides mean

and variance estimates for them and the multinomial $P(c)$. Then multiplying out the Gaussians' exponents and applying Bayes' theorem yields an unsupervised initialization estimate for the weights $\boldsymbol{\omega}$ of exponential features $(1, x, x^2)$. Explicitly modeling $P(x|c)$ as Gaussians is discussed in the generative model in Section 5.7.1.

5.5 Evaluation with Synthetic Data

To assess the performance of our method under different controlled conditions, we created synthetic datasets reflecting key properties of array-CGH microarrays using the following process.

In accordance with laboratory evidence suggesting that amplicons are selected based on certain underlying driver genes (Albertson, 2006), 5 "oncogene" positions were randomly chosen for each dataset of fixed sequence length N. Then, amplicons of width $\sim \mathcal{N}(15, 5)$ and uniform random offset were created to contain each oncogene position with probability $1 - \varepsilon$ for positive examples and ε for negative examples (i.e., the inversion noise ε decreases the correlation of amplicon existence and positive label). Copy number levels were limited to normal (ratio = 1) and amplified (ratio = 1.5, reflecting tumor sample heterogeneity).

Realistic microarray measurement noise was then added, according to the exponential model proposed by Rocke and Durbin (2001) and using parameters estimated by Myers *et al.* (2004) from real human breast cancer array-CGH data. The "clean" versions of all datasets, prior to microarray measurement noise addition, were also stored for comparison.

We generated 10 instances of 1000-sequence datasets for each combination of $N \in \{100, 1000\}$ and $\varepsilon \in \{0, 0.25\}$, with even positive/negative ratio. For each 1000-sequence instance, 50 examples were used for training and 950 for test.

Over the 10 instances for each setting, we ran our Heterogeneous Hidden Conditional Random Field (HHCRF) model with $C = 2$ states ("normal" and "amplified") and $\beta \in \{5, 10, 20\}$ for 100 iterations, and compared it to a purely non-sequential Logistic Regression (LR) model tuned by gradient descent with learning rate 0.1 and momentum 0.5 over 100 iterations.

As a sparsely regularized model for comparison, we used L_p-regularized Logistic Regression (L_pLR) (Liu *et al.*, 2007) whose effectiveness has been demonstrated

on expression microarray data. We used the parameters $p = 0.1$ and $\gamma = 10^{-4}$ as suggested (though we did try other combinations with less success), and regularization weight $\beta \in \{0.1, 0.3, 0.5, 1, 3, 5\}$, gradient-optimized with learning rate 0.1 and momentum 0.5 over 500 iterations.

In addition, as a means of taking sequential correlations into account for noise reduction, we also ran L_pLR after preprocessing the data with a moving average of window size 50 ($L_p\text{LR}_{w50}$).

5.6 Experimental Results

We evaluated our method on a range of synthetic datasets modeled after real cancer microarrays, and then on four biological datasets of breast, uveal melanoma, and bladder tumors. The following results demonstrate that our method performs substantially better than state-of-the-art classification methods, and is able to make new clinically relevant predictions for key amplicons and candidate marker genes.

5.6.1 Synthetic Data

Classification

In synthetic experiments with data generated to resemble real microarray data (Rocke and Durbin, 2001, Methods), HHCRF consistently achieved significantly (by Student's paired t-test with $p < 10^{-4}$; i.e. confidence $> 99.99\%$) higher classification accuracy as compared to logistic regression (LR), L_p-regularized logistic regression (L_pLR), and L_p-regularized logistic regression preprocessed with a moving average of window size 50 ($L_p\text{LR}_{w50}$) (Figure 5.2).

We also ran LR and HHCRF on the "clean" versions (without microarray measurement noise) of the datasets (the other models were omitted since sparsity and smoothing became irrelevant in the absence of noise variance). Still, HHCRF performed better than LR, especially for the datasets with inversion noise ($\varepsilon = 0.25$), which suggests that HHCRF's pairwise parameters $\boldsymbol{\rho}$ capture sequence-wide stability properties and contribute to the classification task beyond simply filtering out observation noise.

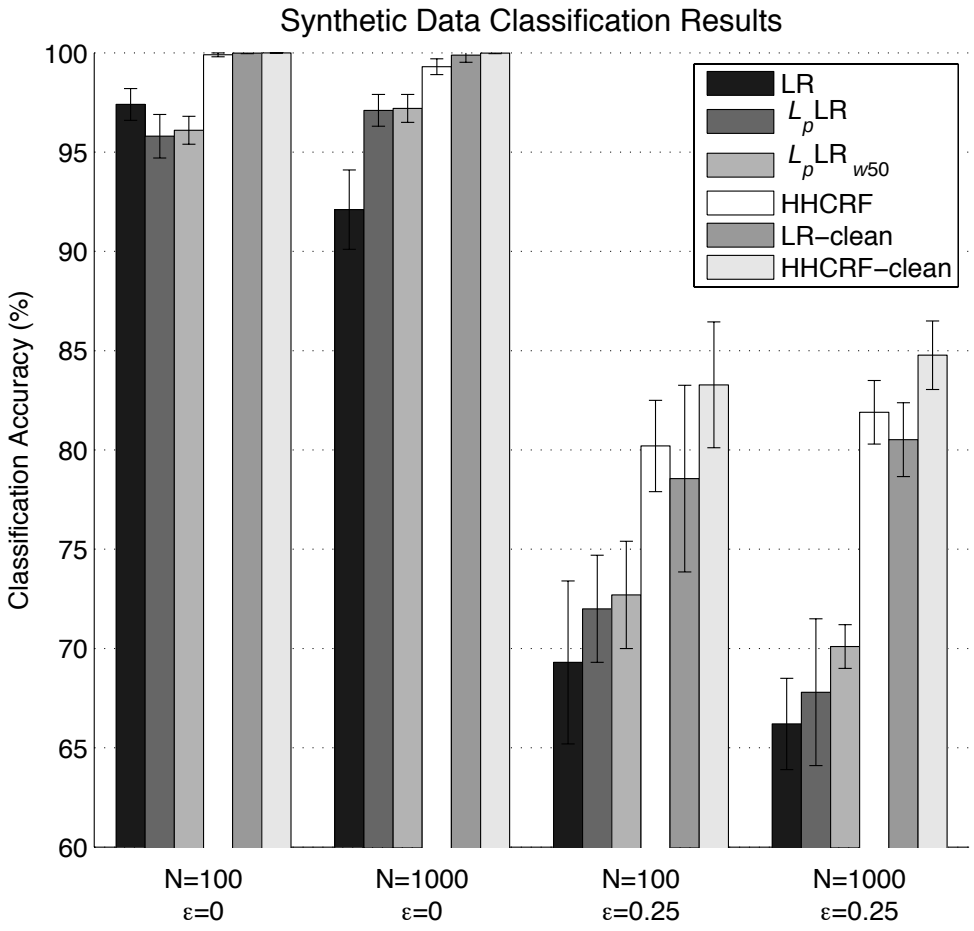


Figure 5.2: Synthetic data classification accuracies (with std. dev. error bars) for sequence length N and inversion noise ε over 10 instances of 50-training/950-test-example runs for the best cross-validated parameter settings of each model, for logistic regression (LR), with L_p regularization (L_pLR), preprocessed with a moving average of sequence window size 50 ($L_p\text{LR}_{w50}$), HHCRF, and results on data without simulated microarray measurement noise (LR-clean, HHCRF-clean).

Indeed, HHCRF accuracy on *noisy* data is comparable to the "clean" data accuracy of LR, and indeed significantly better on the more difficult $\varepsilon = 0.25$ datasets (with 96% confidence for $N = 1000, \varepsilon = 0.25$), demonstrating the extent to which HHCRF is able to cope with experimental microarray noise.

N	ϵ_{inv}	Accuracy	Precision	Recall
100	0	76.1	52.9	98.2
1000	0	90.3	25.9	96.2
100	0.25	84.3	67.9	87.3
1000	0.25	88.6	21.7	86.2

Table 5.1: **Synthetic Data Amplification Results** Synthetic data amplification discovery statistics for the HHCRF models in Figure 5.2 over all genes in all test examples. True positives are amplified genes that were correctly inferred as amplified, and false positives are unamplified genes inferred by the model as amplified.

Copy Number Inference

The integral copy numbers for the classified sequences are the by-product of our model's classification task, obtainable by an efficient Viterbi-like max-product algorithm. Having the true underlying copy number states (normal versus amplified) for the synthetic data, we compared the states inferred by HHCRF to the true values. Note that the other models in the comparison cannot infer actual copy numbers at all. Table 5.1 summarizes the recovery of the true amplification states over all genes of all test sequences, where true positives are amplified genes inferred as amplified, and false positives are unamplified genes inferred as amplified. The high recall (TP/[TP+FN]) and comparatively lower precision (TP/[TP+FP]) reveal a tendency to avoid false negatives, which is not surprising considering that the discriminative loss is incurred only through selected oncogenes (non-zero local parameters) which are much more likely to be amplified than other genes, making false negatives more costly than false positives. In this situation, suggesting the biologist a more extensive candidate list is important, as additional information such as known oncogene status can be used to filter candidates. Thus our algorithm is effective in suggesting potential causative gene hypotheses that the user can examine for biologically interesting possibilities to follow up on.

Oncogene Discovery

Comparing the sparse set of "predicted oncogenes" selected by the model to the underlying true oncogenes requires a soft measure of overlap, both in set membership and also in terms of gene similarity, because Gradient LASSO reports only

one in a group of genes that are always amplified together. For this purpose, we define a *co-amplification matrix* between the predicted oncogenes (rows) and the true oncogenes (columns), with entries denoting the correlation coefficients of the two genes' true copy numbers over test data. In practice, this copy number correlation provides a useful post-processing step to retrieve other candidate genes highly co-amplified with those selected by the model.

We then define *co-precision* as the mean of row maximums (average co-amplification of a predicted oncogene with the closest true oncogene) and *co-recall* as the mean of column maximums (average co-amplification of a true oncogene with the closest predicted oncogene). Thus, a model that returns only some of the true oncogenes, but no false predictions, will have high co-precision and low co-recall. Conversely, if all true oncogenes are found, but with many other spurious predictions, then co-recall will be high, and co-precision low. As desired, these measures are not affected much if several highly co-amplified genes are returned for one true oncogene.

These statistics, along with their harmonic mean (*co-F-measure*), are shown for the HHCRF models on the synthetic datasets in Figure 5.3.

The high co-recall values demonstrate successful recovery of most true oncogenes, decreasing with sequence length and ϵ difficulty, while the co-precision values indicate that the numbers of spurious predicted oncogenes were limited.

Also observable in Figure 5.3 is the effect of the regularization weight β on model complexity, directly increasing the number of predicted oncogenes.

5.7 Discussion

5.7.1 Generative Model

An observable consequence of the discriminative design is that the label-driven gradient feedback makes relatively small updates on the observation parameters. While the resulting dependence on their initial values is not a major practical concern as they can be easily initialized to sensible estimates by clustering or unsupervised sequence models, we also implemented a generative version of our model to see whether it would alleviate this sensitivity.

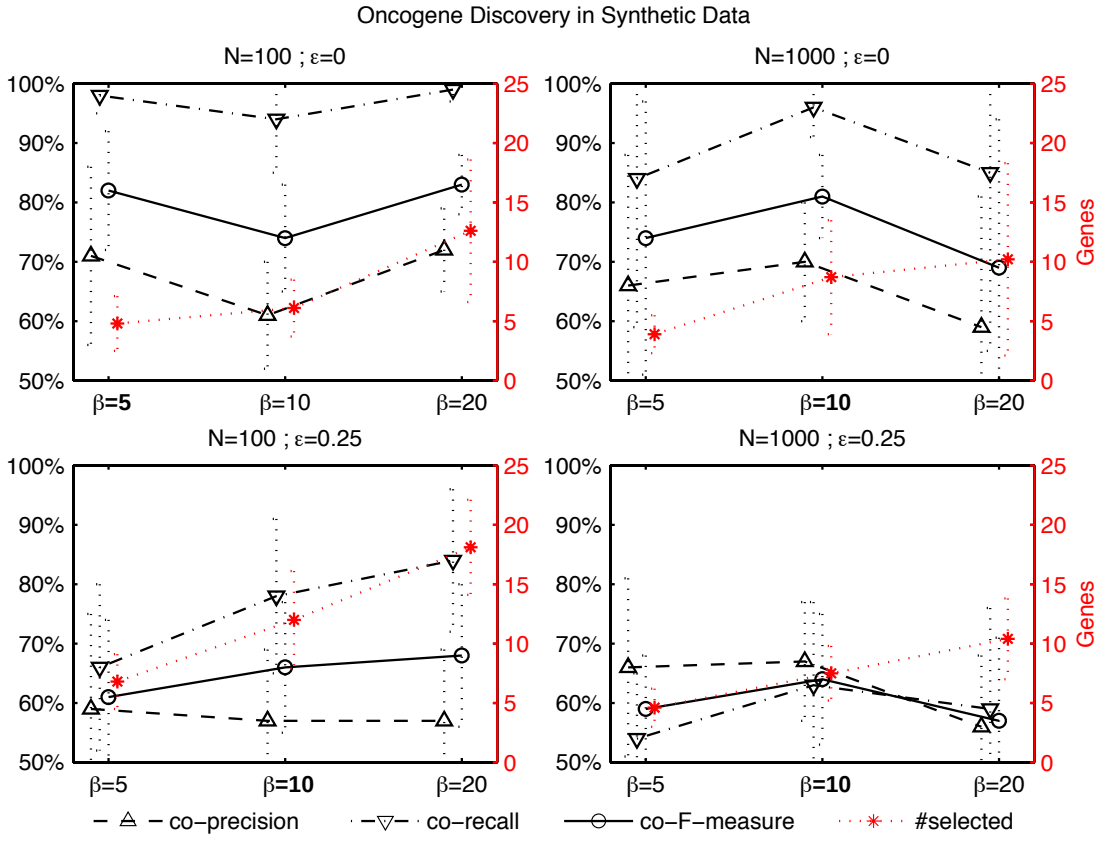


Figure 5.3: Synthetic data oncogene discovery statistics (with std. dev. error bars) for HHCRF with $\beta \in \{5, 10, 20\}$ over the 10 instances of each dataset. **Bold** labels indicate the β values with the highest classification accuracies in Figure 5.2. #selected (on the right x-axis) is the number of predicted oncogenes, compared to the 5 true oncogenes. Co-precision, co-recall, and co-F-measure are percentages defined on the co-amplification matrix between the predicted and true oncogenes.

The generative version models the joint distribution:

$$\begin{aligned} p_{\boldsymbol{\theta}}(s, \mathbf{c}, \mathbf{x}) &= p_{\boldsymbol{\theta}}(\mathbf{x}|s, \mathbf{c}) p_{\boldsymbol{\theta}}(s, \mathbf{c}) = p_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{x}|\mathbf{c}) p_{\boldsymbol{\rho},\boldsymbol{\lambda}}(s, \mathbf{c}) \\ &= \prod_{i=1}^{n} g\left(x_i; \mu_{c_i}, \sigma_{c_i}\right) \times \frac{1}{Z_{\boldsymbol{\rho},\boldsymbol{\lambda}}} \exp\left(\boldsymbol{\rho} \cdot \mathbf{f}_{pair}(\mathbf{c}, s) + \boldsymbol{\lambda} \cdot \mathbf{f}_{local}(\mathbf{c}, s)\right) \end{aligned}$$

where $g(x; \mu, \sigma)$ is the normalized Gaussian density of mean μ and variance σ^2 at x.

In contrast to the discriminative architecture, this version explicitly models $p(x_i|c_i)$ as Gaussian (as in Shah *et al.*, 2007), and the joint probability model al-

lows training by maximizing the joint log-likelihood $\sum_{\mathbf{x},s} \tilde{p}(\mathbf{x},s)\log p_{\boldsymbol{\theta}}(\mathbf{x},s)$. This model can also include unlabeled examples, if available, for training by leaving s latent.

Although optimizing a joint loss does update the observation parameters (now $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$) more directly, this model did not perform well in our classification experiments. All copy number levels quickly converged to overlapping Gaussians of similar means and large variance. Considering that the effect of the supervision label on loss is only through the selected non-zero local parameters, this suggests that the faster optimization at the observation level did not allow enough time for useful local parameters to be selected before relevant information was lost to maximize unsupervised observation likelihood, and the model was essentially reduced to a purely homogeneous one.

Despite these initial results, the generative model deserves further investigation because the Gaussian observations assumption is supported by data, and it may thus be encoding more prior knowledge into the design than the discriminative one, in addition to being more directly interpretable than generic exponential features. A discriminative training scheme, optimizing the conditional likelihood on the generative model, remains to be explored.

5.7.2 Other Extensions

Our model can be extended in several directions.

One piece of information that is not considered in most of the methods is the physical distance of the probes along the genome; uniform spacing is assumed. If two probes indicating the same direction of change are very far apart, the probability that they refer to the same alteration should be lower than if they had been closer. When such individual probe-to-probe distance information is available, it can be directly encoded into the pairwise features of HHCRF, similarly to the non-homogeneous HMM model by Rueda and Diaz-Uriarte (2007).

The correlation-based post-processing step, retrieving similar genes from the selected oncogenes, can be necessary because of the L_1 loss minimized by Gradient LASSO: if two or more genes are equally important, picking only one of them is L_1-optimal. The desired grouping effect can be provided by a hybrid L_1+L_2 ex-

tension of Gradient LASSO, analogous to the Elastic Net (Zou and Hastie, 2005) extension of LASSO (Tibshirani, 1996), which will select all similarly important genes simultaneously due to the L_2 component.

Although using a finite set of possible copy number levels may be sufficient in practice, incorporating *hierarchical Dirichlet processes* can allow copy numbers to grow arbitrarily (Teh *et al.*, 2006). Then array-CGH measurements can also be modeled to have an explicitly linear dependency on copy number, further reducing the number of parameters.

Replacing maximum likelihood training with a Bayesian treatment, working with posterior distributions of model parameters (similar to Qi *et al.*, 2005) can reduce overfitting during training. Maximizing the classification margin (similar to Taskar *et al.*, 2004) may also provide better generalization performance.

A semi-Markov extension (Sarawagi and Cohen, 2004), using subsequence features, may prescribe alternative forms (e.g. Gaussian) for the decay of altered-state probability around predicted cancer genes, instead of the current exponential spikes caused by multiplicative propagation through pairwise transitions. This can allow wide amplification regions to be represented by fewer selected genes, leading to sparser solutions.

Finally, the model can be fit to expression microarrays as well, although the statistical assumptions may hold less strongly. Gene expression (mRNA) microarrays include information about the dynamic regulatory relationships among genes, but expression levels are still affected by copy numbers. If both array-CGH and expression microarray data are available for a dataset, HHCRF can use them together, by simply adding a new set of observed variables stemming from the same latent copy numbers.

In the next chapter, we present results on applying HHCRF to real array-CGH datasets from breast, uveal melanoma, and bladder tumors.

Chapter 6

Analysis of Breast, Melanoma, and Bladder Tumors

In this chapter, we present results on applying the Heterogeneous Hidden Conditional Random Field method to real cancer datasets of breast, uveal melanoma, and bladder tumors. Classification accuracies are compared to state-of-the-art methods. We also make novel predictions on clinically relevant loci, and provide literature evidence supporting our predictions.

6.1 Breast Cancer Data

We applied HHCRF to two breast cancer datasets for the task of identifying amplicons and potential causative genes predictive of high tumor grade. In both experiments, HHCRF successfully classified held-out examples significantly more accurately than a non-sequential SVM model, and made candidate gene predictions for relevance to tumor grade.

6.1.1 Pollack *et al.* (2002) Breast Tumor Data

On the 6691-gene human breast tumor array-CGH data from Pollack *et al.* (2002), we applied HHCRF with $C = 4$ copy number levels to classify tumors with histological grade 3-versus-all (17 positives out of 42). Over 5-fold cross-validation, held-out classification accuracies (mean$\pm$std.dev.) for $\beta \in \{5, 10, 20\}$ were $76 \pm 07\%$, $67 \pm 10\%$,

Index	Name	Weight	Evidence
98	ARID1A	$+0.61\uparrow$	Huang *et al.* (2007)
353	VDUP1	$+1.30\downarrow$	Han *et al.* (2003)
4505	co-amplified with CUL4A	$-0.69\downarrow$	Nag *et al.* (2004)
5289	*H. sapiens* clone 23596	$+1.14\uparrow$	Yi *et al.* (2007)
5634	FLJ23403	$-1.26\downarrow$	Beitzinger *et al.* (2008)

Table 6.1: **Selected Genes for Pollack *et al.* (2002) Data** Positive weights make a positive (high-grade) label more likely when amplified (↑) or deleted (↓), and negative weights make a negative label more likely. Microarray feature 4505 does not have a gene name, but it is highly co-amplified (corr.coeff.=0.69) with nearby feature 4515 (CUL4A).

and $64 \pm 07\%$ respectively, compared to $60 \pm 20\%$ for a linear SVM. In addition to lower variance, HHCRF with $\beta = 5$ was statistically significantly more accurate (with 96% paired t-test confidence) than the SVM.

We then trained HHCRF with $\beta = 5$ on all 42 sequences, and examined the chosen genes. Table 6.1 shows the selected genes and their non-zero local weights. Among the selected genes, several have known connections to tumor formation. ARID1A has been identified as a presumptive tumor suppressor (Huang *et al.*, 2007), and VDUP1 is a known tumor suppressor (Han *et al.*, 2003). "Homo sapiens clone 23596 mRNA sequence" has been observed to be highly expressed in breast cancer cell lines (Yi *et al.*, 2007), and downregulation of FLJ23403 (alias FAM38B) has been linked to human cancers (Beitzinger *et al.*, 2008).

Due to the non-grouping character of L_1 regularization, finding a relevant gene can suppress the subsequent detection of similar genes. In particular, Gradient LASSO picks only one gene out of a region that is always amplified together. To circumvent this effect, a correlation-based post-processing step can be applied after learning, to retrieve other relevant genes whose inferred copy numbers are highly correlated with the representative ones that were found by Gradient LASSO. For example, in Table 6.1, microarray feature 4505 does not match to a named gene, but its highest correlation (coefficient 0.69) in copy number is with the nearby feature 4515 (CUL4A), a known breast cancer amplification (Nag *et al.*, 2004).

Index	chr	Clone Name	Weight
262	3	RP11-129P2	$-1.19\uparrow$
566	5	CTD-2004C12	$+1.68\downarrow$
657	6	RP11-47E20	$-1.25\downarrow$
883	8	RP11-116F9	$+1.34\downarrow$
953	8	RP11-44N11	$+1.25\uparrow$
1725	16	RP11-52E21	$-0.90\downarrow$
1738	16	RP11-140K16	$-0.38\downarrow$
1780	17	DMPC-HFF#1-61H8	$+0.09\uparrow$
2078	22	RP1-238C15	$-1.33\downarrow$
2086	22	RP11-35I10	$-0.59\downarrow$

Table 6.2: **Selected Probes for Chin *et al.* (2006) Data** Positive weights make a positive (high-grade) label more likely when amplified (↑) or deleted (↓), and negative weights make a negative label more likely.

6.1.2 Chin *et al.* (2006) Breast Tumor Data

The human breast tumor array-CGH data from Chin *et al.* (2006) has measurements for 2149 probe positions, not mapping directly to individual genes. Again, we ran HHCRF experiments with $C = 4$ for grade 3-versus-all (69 positives out of 141). The 5-fold cross validation classification accuracies for $\beta \in \{5, 10, 20\}$ were $70 \pm 12\%$, $71 \pm 12\%$, and $67 \pm 07\%$ respectively, compared to $68 \pm 10\%$ for a linear SVM. HHCRF with $\beta = 10$ was more accurate than the SVM with 83% paired t-test confidence. As before, we then trained HHCRF with $\beta = 10$ on all 141 sequences for novel prediction, and Table 6.2 shows the selected probes.

Figure 6.1 shows part of a copy number profile extracted for high-grade breast tumor sequence `b0499`. In addition to determining the amplified and deleted regions, our model selected position 953 as a clinically relevant locus in determining tumor grade, predicted to correspond to the "driver" gene for the 942..975 amplicon.

6.2 Institut Curie Melanoma and Bladder Data

We also obtained successful results by applying our model on uveal melanoma and bladder tumor data from Institut Curie, used in the evaluation of the Fused SVM algorithm in Rapaport *et al.* (2008). HHCRF classification performance exceeded

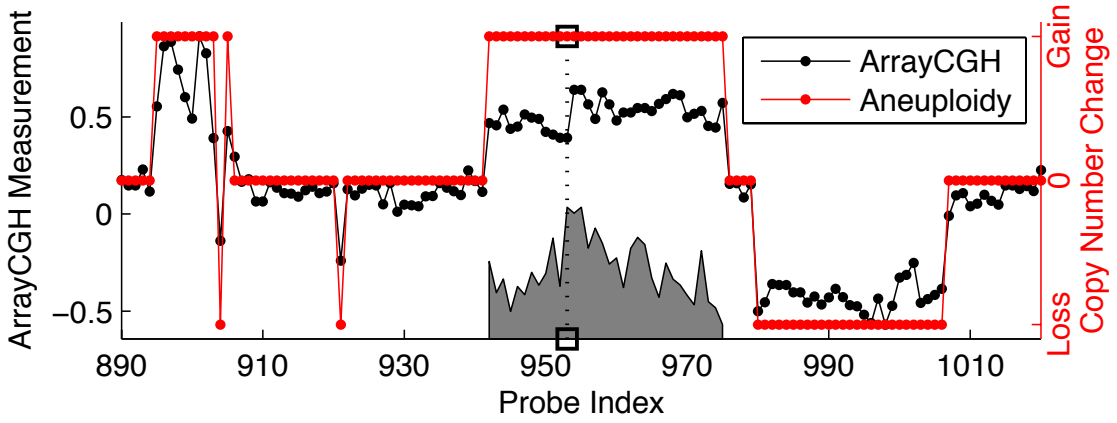


Figure 6.1: Aneuploidies detected in a high-grade breast tumor from Chin *et al.* (2006). Our method detects the amplified and deleted regions, and also pinpoints probe 953 (□) in the 942–975 amplicon as one of the ten clinically-important positions selected for relevance to high tumor grade, by analyzing across all tumor profiles in the dataset. The shaded area shows copy number correlations with the selected probe.

(uveal melanoma tumors) or was comparable to (bladder tumors) that of Fused SVM in these results. HHCRF produces a more interpretable model, outputting a specific set of outcome-related "amplicon-driving" genes.

It should be noted Fused SVM does not limit the amplitudes of altered regions to a shared set of copy number levels; this may provide a better fit in the presence of high variance in tumor heterogeneity across many samples. Alternatively, if the effects of tumor heterogeneity and normal cell contamination have already been normalized out by the increasingly popular flow cytometric sorting techniques, HHCRF assumptions will hold stronger. In practice, model selection should ultimately be guided by application objectives and the particular data at hand.

6.2.1 Uveal Melanoma Tumors

The uveal melanoma tumor data has array-CGH profiles with 3649 probes on non-sex chromosomes. Classifying by whether liver metastasis occurred within 24 months versus not (35 positives out of 78 tumors), HHCRF with $C = 5$ states made a total of 10 test errors (87% accuracy) over 10 cross-validation folds for $\beta = 5$, 11 errors (86% accuracy) for $\beta = 10$, and 8 errors (90% accuracy) for $\beta = 20$, compared

to the best 10-fold cross-validation results from Fused SVM at 17 errors (78% accuracy).

6.2.2 Bladder Tumors

The bladder carcinoma dataset contains array-CGH profiles with 2143 probes on non-sex chromosomes. On classification by tumor stage Ta-versus-T2+ (16 "stage Ta" positives out of 48 tumors with stage labels), HHCRF with $C = 5$ states made a total of 7 test errors (85% accuracy) over 10 cross-validation folds for $\beta = 5$ and $\beta = 10$, and 8 test errors (83% accuracy) for $\beta = 20$. The best HHCRF error is on par with the best leave-one-out estimate of Fused SVM (7 errors) reported in Rapaport *et al.* (2008).

Classifying by tumor grade 1-versus-higher (12 "grade 1" positives out of 57 tumors), HHCRF with $C = 5$ states made a total of 10 test errors (82% accuracy) over 10 cross-validation folds for $\beta = 5$, 9 errors (84% accuracy) for $\beta = 10$, and 11 errors (81% accuracy) for $\beta = 20$, compared to the best leave-one-out estimate reported by Fused SVM (7 errors).

6.3 Discussion

We presented the Heterogeneous Hidden Conditional Random Field, a novel fixed-length sequence classifier, applied to array-CGH data for jointly classifying tumors by clinical label, extracting copy number profiles, identifying clinically relevant genes. We demonstrated its effectiveness on synthetic and real datasets, and described a generative variation and other extensions.

A particularly important feature of our method is to estimate the clinical significance of detected copy number changes. When the genome-wide profile is scanned for potentially new regions of interest, quantitative statistics about the aberrations are critical in order to decide which region to pursue for further examination. Our model highlights the most clinically relevant aneuploidy regions as those containing the predictive genes it has selected. The method also allows prioritization of genes harbored within the chromosomal regions of interest, starting with the explicitly selected genes and extending to others in similarity by co-amplification. In

previous studies, prior biological knowledge was heavily used to infer causal genes in amplified regions, and thus, many known or putative oncogenes were credited as the driver genes, while some potentially novel cancer-driving genes may have been overlooked.

Conclusion

We presented Hierarchical Bayesian Aggregation and Heterogeneous Hidden Conditional Random Fields, methods for exploiting output and input structure, respectively. Both methods owe their proven success to incorporating known structural relationships into the final hypothesis using the graphical models paradigm.

Another common principle to both methods is the avoidance of one-way processing steps. HHCRF avoids de-noising the observations by unsupervised preprocessing, and builds an integrated model where the observation layer is allowed feedback from the supervised label. Bayesian Aggregation avoids taking a top-down or bottom-up sequence of higher-precedence evaluations, and solves the most likely set of class labels collectively, where responsibility and improvement are distributed more evenly over all hierarchy levels. Instead of decoupling parts of the problem, both methods allow better distribution of information through integration. An exception is the full training of the base classifiers before Bayesian Aggregation, which is less desirable than an integrated set of base classifiers that learn in tandem through the Bayesian network, although such a design is likely to be limited to a particular base classifier model, sacrificing some of the flexibility of the current design. The downside of integration is increased model complexity, but both methods manage to deal with it effectively, and have proven to be accurate and practicable solutions to their problem settings.

HHCRF additionally deals with the issue of knowledge extraction, where the application objective is more the interpretability of the final hyptohesis than the best black-box predictor possible. The sparse solution of a handful of "causal" genes has provided that knowledge, in addition to regularizing model complexity against overfitting.

Machine learning is a multi-faceted balancing act, between sheer accuracy ver-

sus interpretability, decoupling versus integration, model expressiveness versus overfitting, and many other criteria. Throwing generic classifiers at data and expecting all judgment to come from cross-validation is wasteful at best. Sound design decisions for the best models require a deep understanding of the problem and data at hand, in addition to being well-informed about the many statistical modeling options available. We have added to those options with our new methods in this thesis, with experimental illustrations of their characteristics, in addition to achieving our main goal of improving the state-of-the-art in the motivating real-world problems. We hope that our methods will be instrumental to others in developing even better methods in the future, as we have built on the excellent works of many others before us.

Bibliography

Albertson, D., Ylstra, B., Segraves, R., Collins, C., Dairkee, S., Kowbel, D., Kuo, W., Gray, J., and Pinkel, D. (2000). Quantitative mapping of amplicon structure by array CGH identifies CYP24 as a candidate oncogene. *Nature Genetics*, **25**(2), 144–146.

Albertson, D. G. (2006). Gene amplification in cancer. *Trends Genet*, **22**(8), 447–455.

Alpaydin, E. (2004). *Introduction To Machine Learning*. MIT Press.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*, **25**(1), 25–29.

Bakir, G., Hofmann, T., Schölkopf, B., Smola, A., Taskar, B., and Vishwanathan, S., editors (2007). *Predicting Structured Data*. Neural Information Processing. The MIT Press.

Barutcuoglu, Z. and Alpaydin, E. (2003). A comparison of model aggregation methods for regression. *International Conference on Artificial Neural Networks and Neural Information Processing (ICANN/ICONIP)*, **2714**, 76–83.

Barutcuoglu, Z. and DeCoro, C. R. (2006). Hierarchical Shape Classification Using Bayesian Aggregation. *Proc. IEEE Intl. Conf. on Shape Modeling and Applications (SMI'06)*.

Barutcuoglu, Z., Schapire, R., and Troyanskaya, O. (2006). Hierarchical multi-label prediction of gene function. *Bioinformatics*, **22**(7), 830–836.

Barutcuoglu, Z., DeCoro, C., Schapire, R., and Troyanskaya, O. (2008). Bayesian Aggregation for Hierarchical Classification. Technical Report TR-785-07, Princeton University, Department of Computer Science.

Beitzinger, M., Hofmann, L., Oswald, C., Beinoraviciute-Kellner, R., Sauer, M., Griesmann, H., Bretz, A. C., Burek, C., Rosenwald, A., and Stiewe, T. (2008). p73 poses a barrier to malignant transformation by limiting anchorage-independent growth. *EMBO J*, **27**(5), 792–803.

Bell, S. P. (2002). The origin recognition complex: from simple origins to complex functions. *Genes Dev*, **16**(6), 659–672.

Berger, A., Della Pietra, V., and Della Pietra, S. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, **22**(1), 39–71.

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, **24**(2), 123–140.

Breitkreutz, B.-J., Stark, C., and Tyers, M. (2003). The GRID: the General Repository for Interaction Datasets. *Genome Biol*, **4**(3), R23.

Brown, L. A., Irving, J., Parker, R., Kim, H., Press, J. Z., Longacre, T. A., Chia, S., Magliocco, A., Makretsov, N., Gilks, B., Pollack, J., and Huntsman, D. (2006). Amplification of EMSY, a novel oncogene on 11q13, in high grade ovarian surface epithelial carcinomas. *Gynecol Oncol*, **100**(2), 264–270.

Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, **2**(2), 121–167.

Cai, L. and Hofmann, T. (2004). Hierarchical Document Categorization with Support Vector Machines. *ACM 13th Conference on Information and Knowledge Management*.

Cesa-Bianchi, N., Conconi, A., and Gentile, C. (2004). Regret Bounds for Hierarchical Classification with Linear-Threshold Functions. *The 17th Annual Conference on Learning Theory*, pages 93–108.

Chen, Y. and Yu, D. (2004). Global protein function annotation through mining genome-scale data in yeast *Saccharomyces cerevisiae*. *Nucleic Acids Res.*, **32**(21), 6414–24.

Chin, K., DeVries, S., Fridlyand, J., Spellman, P. T., Roydasgupta, R., Kuo, W.-L., Lapuk, A., Neve, R. M., Qian, Z., Ryder, T., Chen, F., Feiler, H., Tokuyasu, T., Kingsley, C., Dairkee, S., Meng, Z., Chew, K., Pinkel, D., Jain, A., Ljung, B. M., Esserman, L., Albertson, D. G., Waldman, F. M., and Gray, J. W. (2006). Genomic and transcriptional aberrations linked to breast cancer pathophysiologies. *Cancer Cell*, **10**(6), 529–541.

Chu, S., DeRisi, J., Eisen, M., Mulholland, J., Botstein, D., Brown, P. O., and Herskowitz, I. (1998). The transcriptional program of sporulation in budding yeast. *Science*, **282**(5389), 699–705.

Clare, A. and King, R. (2003). Predicting gene function in *Saccharomyces cerevisiase*. *Bioinformatics*, **19**(2), 1142–49.

DeCoro, C., Barutcuoglu, Z., and Fiebrink, R. (2007). Bayesian aggregation for hierarchical genre classification. *International Symposium on Music Information Retrieval*.

Dekel, O., Keshet, J., and Singer, Y. (2004). Large margin hierarchical classification. *ICML'04*, pages 209–216.

Dumais, S. and Chen, H. (2000). Hierarchical classification of Web content. *Proc. 23rd ACM Intl. Conf. on Research and Development in Information Retrieval (SIGIR'00)*, pages 256–263.

Durand, J.-B., Gonçalvès, P., and Guédon, Y. (2004). Computational Methods for Hidden Markov Tree Models—An Application to Wavelet Trees. *IEEE Transactions on Signal Processing*, **52**(9), 2551–2560.

Efron, B. and Tibshirani, R. J. (1994). *An Introduction to the Bootstrap*. Chapman & Hall/CRC.

Fujibuchi, W., Anderson, J. S., and Landsman, D. (2001). PROSPECT improves cis-acting regulatory element prediction by integrating expression profile data with consensus pattern searches. *Nucleic Acids Res*, **29**(19), 3988–3996.

Funkhouser, T., Min, P., Kazhdan, M., Chen, J., Halderman, A., Dobkin, D., and Jacobs, D. (2003). A Search Engine for 3D Models. *ACM Transactions on Graphics (TOG)*, pages 83–105.

Gasch, A. P., Spellman, P. T., Kao, C. M., Carmel-Harel, O., Eisen, M. B., Storz, G., Botstein, D., and Brown, P. O. (2000). Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell*, **11**(12), 4241–4257.

Gasch, A. P., Huang, M., Metzner, S., Botstein, D., Elledge, S. J., and Brown, P. O. (2001). Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mec1p. *Mol Biol Cell*, **12**(10), 2987–3003.

Guan, Y., Myers, C., Hess, D., Barutcuoglu, Z., Caudy, A., and Troyanskaya, O. (2008). Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biology*, **9**(1), S3.

Guldener, U., Munsterkotter, M., Kastenmuller, G., Strack, N., van Helden, J., Lemer, C., Richelles, J., Wodak, S. J., Garcia-Martinez, J., Perez-Ortin, J. E., Michael, H., Kaps, A., Talla, E., Dujon, B., Andre, B., Souciet, J. L., De Montigny, J., Bon, E., Gaillardin, C., and Mewes, H. W. (2005). CYGD: the Comprehensive Yeast Genome Database. *Nucleic Acids Res*, **33**(Database issue), D364–8.

Han, S. H., Jeon, J. H., Ju, H. R., Jung, U., Kim, K. Y., Yoo, H. S., Lee, Y. H., Song, K. S., Hwang, H. M., Na, Y. S., Yang, Y., Lee, K. N., and Choi, I. (2003). VDUP1 upregulated by TGF-beta1 and 1,25-dihydorxyvitamin D3 inhibits tumor cell growth by blocking cell-cycle progression. *Oncogene*, **22**(26), 4035–4046.

Heckerman, D. (1998). A tutorial on learning with Bayesian networks. *Learning in Graphical Models*, pages 301–354.

Heim, S. and Mitelman, F. (1989). Primary chromosome abnormalities in human neoplasia. *Adv Cancer Res*, **52**, 1–43.

Huang, J., Zhao, Y.-L., Li, Y., Fletcher, J. A., and Xiao, S. (2007). Genomic and functional evidence for an ARID1A tumor suppressor role. *Genes Chromosomes Cancer*, **46**(8), 745–750.

Huh, W.-K., Falvo, J. V., Gerke, L. C., Carroll, A. S., Howson, R. W., Weissman, J. S., and O'Shea, E. K. (2003). Global analysis of protein localization in budding yeast. *Nature*, **425**(6959), 686–691.

Jansen, R., Yu, H., Greenbaum, D., Kluger, Y., Krogan, N. J., Chung, S., Emili, A., Snyder, M., Greenblatt, J. F., and Gerstein, M. (2003). A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, **302**(5644), 449–453.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in kernel methods: support vector learning*, pages 169–184. MIT Press, Cambridge, MA, USA.

Jonsson, G., Naylor, T. L., Vallon-Christersson, J., Staaf, J., Huang, J., Ward, M. R., Greshock, J. D., Luts, L., Olsson, H., Rahman, N., Stratton, M., Ringner, M., Borg, A., and Weber, B. L. (2005). Distinct genomic profiles in hereditary breast tumors identified by array-based comparative genomic hybridization. *Cancer Res*, **65**(17), 7612–7621.

Karaoz, U., Murali, T. M., Letovsky, S., Zheng, Y., Ding, C., Cantor, C. R., and Kasif, S. (2004). Whole-genome annotation by using evidence integration in functional-linkage networks. *Proc Natl Acad Sci U S A*, **101**(9), 2888–2893.

Kazhdan, M. (2004). *Shape Representations and Algorithms for 3D Model Retrieval*. Ph.D. thesis, Princeton University.

Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. (2003). Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Symposium on Geometry Processing*, pages 167–175.

Kim, Y. and Kim, J. (2004). Gradient LASSO for feature selection. In *ICML '04: Proceedings of the 21st International Conference on Machine Learning*, page 60, New York, NY, USA. ACM.

Klapuri, A. P., Eronen, A. J., and Astola, J. T. (2006). Analysis of the meter of acoustic musical signals. *Audio, Speech and Language Processing, IEEE Trans-*

actions on [see also Speech and Audio Processing, IEEE Transactions on], **14**(1), 342–355.

Lai, W. R., Johnson, M. D., Kucherlapati, R., and Park, P. J. (2005). Comparative analysis of algorithms for identifying amplifications and deletions in array CGH data. *Bioinformatics*, **21**(19), 3763–3770.

Lanckriet, G. R. G., Deng, M., Cristianini, N., Jordan, M. I., and Noble, W. S. (2004a). Kernel-based data fusion and its application to protein function prediction in yeast. *Pac Symp Biocomput*, pages 300–311.

Lanckriet, G. R. G., De Bie, T., Cristianini, N., Jordan, M. I., and Noble, W. S. (2004b). A statistical framework for genomic data fusion. *Bioinformatics*, **20**(16), 2626–2635.

Lee, W.-L., Kaiser, M. A., and Cooper, J. A. (2005). The offloading model for dynein function: differential function of motor subunits. *J Cell Biol*, **168**(2), 201–207.

Liu, Z., Jiang, F., Tian, G., Wang, S., Sato, F., Meltzer, S. J., and Tan, M. (2007). Sparse logistic regression with Lp penalty for biomarker identification. *Stat Appl Genet Mol Biol*, **6**, Article6.

McCallum, A., Rosenfeld, R., Mitchell, T., and Ng, A. (1998). Improving text classification by shrinkage in a hierarchy of classes. *Proceedings of the Fifteenth International Conference on Machine Learning*, **367**.

McKay, C. (2004). *Automatic Genre Classification of MIDI Recordings*. M.a. thesis, McGill University, Canada.

McKay, C. and Fujinaga, I. (2004). Automatic Genre Classification Using Large High-Level Musical Feature Sets. *Proc. ISMIR*.

McKay, C. and Fujinaga, I. (2005). The Bodhidharma system and the results of the MIREX 2005 symbolic genre classification contest. *Proc. ISMIR*.

McKay, C. and Fujinaga, I. (2006). Musical genre classification: Is it worth pursuing and how can it be improved? *Proceedings of the Seventh International Conference on Music Information Retrieval (ISMIR'06)*, pages 101–106.

Miles, J. and Formosa, T. (1992). Evidence that POB1, a Saccharomyces cerevisiae protein that binds to DNA polymerase alpha, acts in DNA metabolism in vivo. *Mol Cell Biol*, **12**(12), 5724–5735.

Min, P. (2003). *A 3D Model Search Engine*. Ph.D. thesis, Princeton University.

Myers, C. L., Dunham, M. J., Kung, S. Y., and Troyanskaya, O. G. (2004). Accurate detection of aneuploidies in array CGH and gene expression microarray data. *Bioinformatics*, **20**(18), 3533–3543.

Nag, A., Bagchi, S., and Raychaudhuri, P. (2004). Cul4A physically associates with MDM2 and participates in the proteolysis of p53. *Cancer Res*, **64**(22), 8152–8155.

Nocedal, J. (1980). Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation*, **35**(151), 773–782.

Ogawa, N., DeRisi, J., and Brown, P. O. (2000). New components of a system for phosphate accumulation and polyphosphate metabolism in Saccharomyces cerevisiae revealed by genomic expression analysis. *Mol Biol Cell*, **11**(12), 4309–4321.

Pavlidis, P., Weston, J., Cai, J., and Noble, W. S. (2002). Learning gene functional classifications from multiple data types. *J. Comput. Biol.*, **9**(2), 401–11.

Pollack, J. R., Sorlie, T., Perou, C. M., Rees, C. A., Jeffrey, S. S., Lonning, P. E., Tibshirani, R., Botstein, D., Borresen-Dale, A.-L., and Brown, P. O. (2002). Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors. *Proc Natl Acad Sci U S A*, **99**(20), 12963–12968.

Qi, Y., Szummer, M., and Minka, T. P. (2005). Bayesian Conditional Random Fields. In *AI & Statistics*.

Rapaport, F., Barillot, E., and Vert, J.-P. (2008). Classification of arrayCGH data using a fused SVM. *Bioinformatics*.

Rauber, A., Pampalk, E., and Merkl, D. (2002). Using psychoacoustic models and self-organizing maps to create a hierarchical structuring of music by musical

styles. *Proceedings of the 3rd International Conference on Music Information Retrieval*, pages 71–80.

Rocke, D. M. and Durbin, B. (2001). A model for measurement error for gene expression arrays. *J Comput Biol*, **8**(6), 557–569.

Rueda, O. M. and Diaz-Uriarte, R. (2007). Flexible and accurate detection of genomic copy-number changes from aCGH. *PLoS Comput Biol*, **3**(6), e122.

Sarawagi, S. and Cohen, W. W. (2004). Semi-Markov Conditional Random Fields for Information Extraction. In *ICML '04: Proceedings of the 21st International Conference on Machine Learning*.

Shah, S. P., Lam, W. L., Ng, R. T., and Murphy, K. P. (2007). Modeling recurrent DNA copy number alterations in array CGH data. *Bioinformatics*, **23**(13), i450–8.

Shakoury-Elizeh, M., Tiedeman, J., Rashford, J., Ferea, T., Demeter, J., Garcia, E., Rolfes, R., Brown, P. O., Botstein, D., and Philpott, C. C. (2004). Transcriptional remodeling in response to iron deprivation in Saccharomyces cerevisiae. *Mol Biol Cell*, **15**(3), 1233–1243.

Shilane, P., Kazhdan, M., Min, P., and Funkhouser, T. (2004). The Princeton Shape Benchmark. In *Proc. Shape Modeling International, Genoa, Italy*.

Snijders, A. M., Schmidt, B. L., Fridlyand, J., Dekker, N., Pinkel, D., Jordan, R. C. K., and Albertson, D. G. (2005). Rare amplicons implicate frequent deregulation of cell fate specification pathways in oral squamous cell carcinoma. *Oncogene*, **24**(26), 4232–4242.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., and Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, **9**(12), 3273–3297.

Sudarsanam, P., Iyer, V. R., Brown, P. O., and Winston, F. (2000). Whole-genome expression analysis of snf/swi mutants of Saccharomyces cerevisiae. *Proc Natl Acad Sci U S A*, **97**(7), 3364–3369.

Taskar, B., Guestrin, C., and Koller, D. (2004). Max-Margin Markov Networks. In *Advances in Neural Information Processing Systems 16*.

Teh, Whye, Y., Jordan, Michael, I., Beal, Matthew, J., Blei, and David, M. (2006). Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, **101**(476), 1566–1581.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **58**(1), 267–288.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics*, **17**(6), 520–525.

Troyanskaya, O. G., Dolinski, K., Owen, A. B., Altman, R. B., and Botstein, D. (2003). A Bayesian framework for combining heterogeneous data sources for gene function prediction (in Saccharomyces cerevisiae). *Proc Natl Acad Sci U S A*, **100**(14), 8348–8353.

Tzanetakis, G., Essl, G., and Cook, P. (2001). Automatic musical genre classification of audio signals. *In Proceedings of the Int. Symposium on Music Information Retrieval (ISMIR)*, pages 205–210.

van Beers, E. H. and Nederlof, P. M. (2006). Array-CGH and breast cancer. *Breast Cancer Res*, **8**(3), 210.

Wessels, L. F. A., van Welsem, T., Hart, A. A. M., van't Veer, L. J., Reinders, M. J. T., and Nederlof, P. M. (2002). Molecular classification of breast carcinomas by comparative genomic hybridization: a specific somatic genetic profile for BRCA1 tumors. *Cancer Res*, **62**(23), 7110–7117.

West, K. and Lamere, P. (2007). A Model-Based Approach to Constructing Music Similarity Functions. *EURASIP Journal on Advances in Signal Processing*, **2007**, 1–10.

Yi, C.-H., Smith, D. J., West, W. W., and Hollingsworth, M. A. (2007). Loss of fibulin-2 expression is associated with breast cancer progression. *Am J Pathol*, **170**(5), 1535–1545.

Yoshimoto, H., Saltsman, K., Gasch, A. P., Li, H. X., Ogawa, N., Botstein, D., Brown, P. O., and Cyert, M. S. (2002). Genome-wide analysis of gene expression regulated by the calcineurin/Crz1p signaling pathway in Saccharomyces cerevisiae. *J Biol Chem*, **277**(34), 31079–31088.

Zhang, L. V., King, O. D., Wong, S. L., Goldberg, D. S., Tong, A. H. Y., Lesage, G., Andrews, B., Bussey, H., Boone, C., and Roth, F. P. (2005). Motifs, themes and thematic maps of an integrated Saccharomyces cerevisiae interaction network. *J Biol*, **4**(2), 6.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**(2), 301–320.