CS229 Project Report

Laura Miron
Stanford University
lmiron@stanford.edu

Benoit Pit–Claudel
Stanford University
bpitcla@stanford.edu

1. Introduction

A primary goal in biology is to fully understand the mechanisms by which DNA, specifically the coding sections or genes, control all biological processes within every organism. Two main obstacles to this goal are: (i) laboratory methods for definitively determining the function of a gene can require years and hundreds of thousands of dollars in research for even a single gene, and (ii) historically, independent research groups have not followed the same system for categorizing and labelling gene function, making sharing data across group prohibitively difficult. Recently, however, the scientific community has come to recognize the Gene Ontology (GO) hierarchy [1, 2] maintained by the Gene Ontology Consortium as a unified standard for gene annotations, and great strides have been made in applying machine learning techniques to predicting gene function and GO annotations programmatically, based on a number of possible features such as sequence, pairwise interaction data, presence of histone markers, microarray gene expression data, and more.

This paper replicates results from "Hierarchical multi-label prediction of gene function" by Zafer Barutcuoglu et al., 2006 [3], which proposes a novel way of predicting GO labels for genes from *Saccharomyces cerevisiae* (yeast), by training individual support vector machine (SVM) classifiers for each node of the GO, predicting membership or non-membership for input genes. Other papers publish similar techniques for using SVMs to predict functional annotation, however, the central insight of Barutcuoglu et al. is that the hierarchical nature of the GO can be leveraged to improve the accuracy of the SVM predictions by constructing a Bayes net from the SVM results for each GO node, and making a single prediction for each input about membership/non-membership to all classes.

2. Data Collection and Processing

Training examples were constructed by combining pairwise interaction and microarray expression data for each of 5493 yeast genes (all genes for which relevant data was available), and labeled according to their GO annotations– see table 1 for a summary of the properties of the datasets used.

2.1. Gene Ontology Labels

The GO is a directed, acyclic graph where each node consists of a term name, a unique alphanumeric identifier, a definition with cited sources, and a namespace to which it belongs. The three namespaces are **cellular component**, the parts of a cell or its extracellular environment, **molecular function**, the elemental activities of a gene product at the molecular level, and **biological process**, operations or sets of molecular events with a defined beginning and end, pertinent to the functioning of cells, tissues, and organisms.

GO annotation data from UniProtKB [4] was used, with additional node relationships drawn from the GO graph downloaded from the GO Consortium website[1] (`go-basic.obo`) and parsed with Obonet [5] as a Networkx [6] object. An important note is that GO definitions are constantly updated based on new biological discoveries. Since the publication of Barutcuoglu et al., several ids have been renamed, deemed to be duplicates of existing ids, or retired. UniProtKB was used to update go ids from all the data sources to be consistent with the newest version of the GO; this processing might have been a key contributor to the improved accuracy of the presented predictions versus Barutcuoglu et al.

Each data point has 95 labels, describing whether the gene has been annotated or not with each of the 95 nodes of the GO included in this study. Of the 1000 nodes in the GO which have direct annotations for yeast, this paper considers only the 95 nodes that correspond to the 105 selected in [3] in the last version of the GO hierarchy, in accordance with their reasoning that these nodes have both a reasonable number of annotations (at least 1 direct and 15 total) and are specific enough to have some biological significance if a new gene were predicted to belong there. All chosen nodes are drawn from the biological process hierarchy of the GO, so while some edges in the full ontology represent relationships across domains, within the obtained graph every edge

[1] `http://www.geneontology.org/page/download-ontology`

represents a parent-child relationship where the parent describes a superset of the functions described in each child.

The training labels take the values $\hat{y_j} \in \{-1, 0, 1\}$. A gene is a positive example of both the most specific nodes to which it is annotated, and all parents of that node. This propagation accurately captures the hierarchical nature of the GO, increases the available number of positive training examples for each class, and ensures that all training labels are hierarchically consistent by definition.

Because GO annotations are almost exclusively positive, and typical machine learning algorithms require a balanced amount of positive and negative examples, all examples not annotated to a particular node *or any of its parents* were labeled to be a negative example. Thus, a gene that is annotated to a parent of class i, but not to class i is considered a 0 example, and is left out of the training set for the ensemble of SVMs for node i.

2.2. BioGRID Data

While Barutcuoglu et al used a BioGRID [7, 8] dataset of interactions among yeast genes filtered on specific types of interaction, this paper considers all of the recorded interaction data, regardless of the type of interaction. This choice was primarily due to the numerous changes in interaction type classifications, which rendered their filtering obsolete.

BioGRID datasets each include a certain number of experience records; each interaction type corresponds to a certain type of experiment. Each record includes

– The two genes interacting with each other, using different types of labelling (NCBI classification, also known as entrez gene IDs, or formerly LocusLink[9], BioGRID ID, as well as their aliases in these classifications),

– The publication that showed this interaction,

– The Taxonomy ID[9] uniquely identifying the specie in which the interaction was observed.

– The specific interaction type, using the Molecular Interactions Controlled Vocabulary[10],

Pairwise interaction data is typically represented as a square boolean matrix with genes along both axes, where a nonzero entry denotes an interaction exists between the row gene and the column gene. Each column of this matrix was treated as a feature vector for an example, with the existence or non-existence of an interaction with each other example forming the features.

2.3. Microarrays

DNA microarrays are grids of microscopic spots containing DNA probes subjected to various experimental conditions and used to measure gene expression levels. Microarray datasets are real-valued matrices where the rows are genes or other DNA snips, and the columns are standardized conditions. The microarray data was retrieved from several sources (Chu et al., 1998[11]; Spellman et al., 1998[12]; Sudarsanam et al., 2000[13]; Gasch et al., 2000 [14, 15]), obtained through the NBCI *Gene Expression Omnibus*[9]. From these datasets, 161 features were selected, corresponding to experimental conditions for which data is available for all 5493 training examples. For each example, this real-valued 161 degree feature vector was concatenated with the BioGRID feature vector.

Due to experimental imperfections, microarray results often include missing values. These values were filled in using the `KNNimpute` which had been proven effective by Troyanskaya et al., 2001 [16] with k = 15. A separate KNN model was created for each feature using `scikit.learn` [17], trained on all examples that are not null in that feature. The training examples for each KNN model are additionally preprocessed by replacing other null features with the class average for that feature.

Dataset	BioGRID	Microarray data	Final
Number of different genes represented	7,157	5493	5395
Number of positive interactions	738,333	-	737,000
Datatype	Boolean	Float	Mix

Table 1: Summary of properties for the datasets used in this study. The final number of genes is the intersection of common genes represented in both datasets.

3. Statistical Methods

The method presented in this section is similar to the one presented by Barutcuoglu et al. It consists in using Bayesian inference on separately trained SVM classifiers.

3.1. SVM Classifiers on Individual GO Nodes

First, individual support vector machines are trained for each of the 95 chosen GO nodes. SVM classifiers are machine learning methods that separates positive and negative examples with a linear decision boundary or hyperplane, in a feature space. The `scikit.learn` implementation of SVMs was used, and experiment with both linear and RBF-kernel SVMs using various γ and C parameter values. Results of these tests are presented in section 4.1.

3.2. Dev/Tests Sets Given Few Positive Examples

Since a validation set was needed, but the total number of positive examples for each class is relatively small, bootstrapping (random sampling with replacement) is used to create 10 samples for each classifier, each using 0.632 of the full set of training examples. At test time, each test example is evaluated only on the classifiers for which it is not part of the training set and labeled according to the median of the 10 predictions.

3.3. Bayesian Inference

After training the individual classifiers, a Bayes net representing the 93-node GO hierarchy is constructed, with two Bayes nodes for each GO node, where $\hat{y}_i$ represents the SVM prediction for node i and y_i represents true membership in class i, as illustrated by figure 2.

$P(\hat{y}_i \mid y_i)$ is estimated with maximum likelihood estimation during SVM training. $P(y_i = 1 \mid anychild(y_i) = 1)$ is assumed to be 1.0 before Laplace smoothing, since training labels are enforced to be hierarchically consistent, *i.e.* if a training example is positive for any child class, it is guaranteed to be positive for its parent classes.

Finally, exact inference is performed using `pgmpy`[18]. For each test example, each $\hat{y}_i$ is assigned according to the SVM predictions for that example, and the assignment of all y_i that maximizes the joint distribution $P(y_1, y_2, ..., y_i)$ is found. By enforcing consistency among predictions for different nodes for a single input example the Bayes net predictions improve the accuracy of the SVM classifier predictions alone.

4. Results and Discussion

The results presented in this section show significant improvement compared to Barutcuoglu et al., both for the SVM node classifiers considered separately, and for the Bayesian network made of all these nodes. Some biological functions reached an accuracy of 1 over the more than 1,000 test examples.

4.1. SVM

Grid search was applied on the most represented GO node (GO:0045944, positive regulation of transcription by RNA polymerase II) to find the optimal kernel and the optimal penalty parameter for the error term. Heatmaps of these grid searchs are presented in figure 1.

The scores were computed using a 10 fold cross validation using scikit-learn [17] implementation of SVMs and accuracy and area under the receiver operating characteristic (AUROC) scores.

AUROC represents the probability that a random positive example will be ranked higher by the classifier than a random negative example. Given datasets with a small fraction of positive examples, it is possible to achieve a high accuracy simply by classifying every example as negative, and the AUROC score is therefore a better measure of the performance of a classifier.

Precision and recall were computed counting the number of correctly and incorrectly classified positive and negative examples.

Since the implementation includes a Bayesian network, and since the number of positive examples is small compared to the number of negative examples, it is mostly important to classify as many positive examples as positive. Indeed, the Bayesian network will have an easier time correcting false positives than false negatives. Therefore, the kernel that maximizes the recall was selected, *i.e.* linear kernel with $C \sim 1000$. Using these values, the precision was multiplied by more than 2.5 and the recall by 10 when compared to the results from Barutcuoglu et al. The average obtained accuracy over all the GO nodes is over 97.7%, and the average AUROC is over 0.787.

It is interesting to note that Barutcuoglu et al. used high C value, and an RBF kernel. While the high C value is to be expected, given, as they noted, the high number of features compared with the number of examples, the fact a different kernel was found to be the most efficient could possibly be explained by the updates to the datasets over the years: more experiments were made, giving classifiers used in this project more training examples – Barutcuoglu et al. acknowledge ”noisy data” as a reason for wrongly predicted examples.

In particular, the choice made in this paper to include all types of interactions, initially based on the fact that the interaction types singled out by Barutcuoglu et al. were obsolete, seems to have had a very positive impact on the final results.

Another explanation could come from the updates to the GO classification: as mentioned in section 2.1, the 105 GO nodes selected by Barutcuoglu et al. correspond in the current classification to 95 nodes; equivalent functions have been united in single nodes.

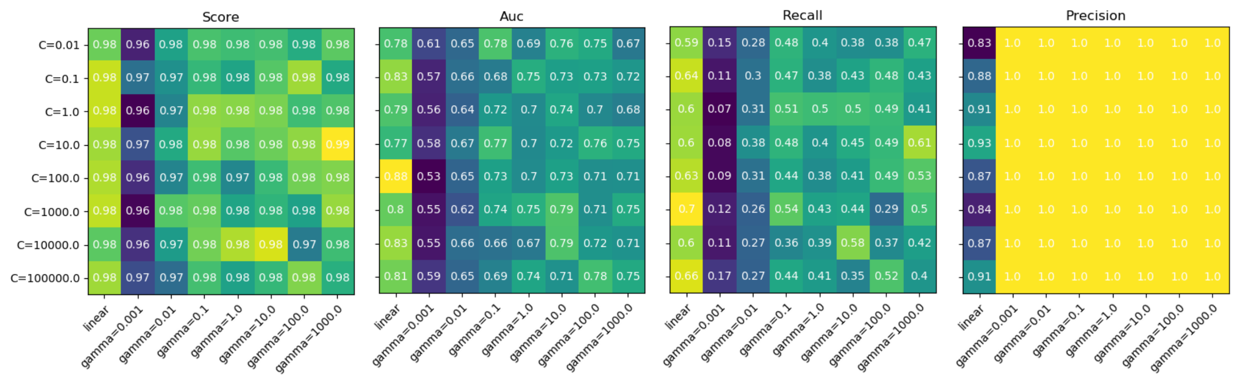
4.2. Bayesian Results

Due to time and computational constraints, as well as having to solve several bugs in the `pgmpy` code base, the Bayes net prediction code was only run on the subset of GO nodes presented in figure 3. The accuracy and AUROC score on this subset of GO nodes, are displayed in table 2.

The Bayes net predictions universally improved upon both the accuracy and AUROC score of the SVM classifiers alone. Although the accuracy was already very high for the SVMs alone, the AUROC was not, and the Bayes net provided significant improvement.

5. Conclusion

This paper successfully reproduced the method developed in Barutcuoglu et al. [3] to classify gene functions using machine learning. The results obtained are very promising in that they show that it is possible to predict almost



(a) Accuracy. (b) AUROC score. (c) Recall. (d) Precision.

Figure 1: Comparison of SVM performance depending on parameters. Scale from violet (low score, minimum at 0), to bright yellow (high score, maximum at 1, for example the precision of all the RBF kernel SVM).

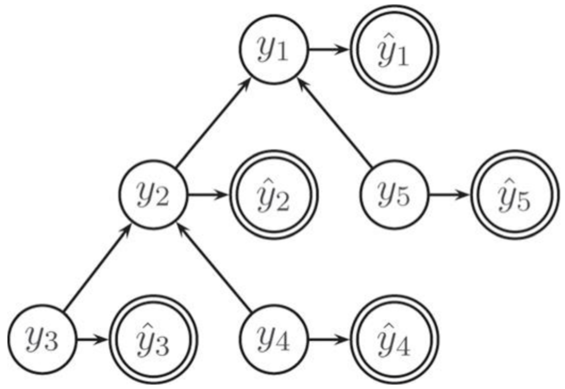


Figure 2: Structure of Bayes net. $\hat{y}_i$ represent SVM predictions, y_i represent the true label of the input example for GO node i [3]

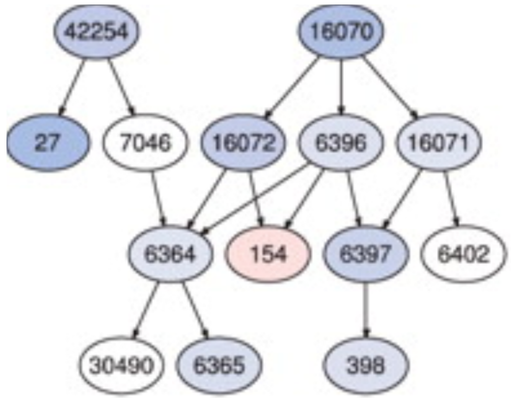


Figure 3: Subtree of GO nodes used in Bayesian predictions (some ids are from out of data GO) [3]

GO ID	SVM Accuracy	SVM AUC	Bayes Accuracy	Bayes AUC
GO:0016071	0.975	0.917	1.0	1.0
GO:0030490	0.987	0.184	1.0	1.0
GO:0042254	0.974	0.841	.999	1.0
GO:0006396	0.944	0.746	0.998	1.0
GO:0000398	0.981	0.376	.995	1.0
GO:0000154	0.987	0.330	1.0	1.0
GO:0000027	0.993	0.570	1.0	0.999
GO:0006367	0.982	0.523	1.0	1.0
GO:0006364	0.971	0.571	.998	1.0
GO:0016072	0.980	0.920	.998	1.0
GO:0016070	0.870	0.490	1.0	1.0

Table 2: Accurary and AUROC scores for SVM classifiers alone versus Bayes net predictions for selected GO nodes

perfectly gene functions based on interaction and microarray data only (thus not using other data such as colocalization data–where the gene is expressed–or transcription factor binding sites data–with which other genes is a specific gene expressed).

The implementation of Barutcuoglu et al. presented in this paper achieved far better score than those presented in the original paper. Reasons for these vast improvements may include, among others, more comprehensive datasets, and slightly different choices in data selection.

An interesting future step would be to try to apply this model to both more complex species–for example humans, and less studied ones–for example newly discovered species. Since the number of features was important compared with the number of training examples, it is possible that the model presented in this paper does some overfitting. Classifying genes from another specie would therefore probably require the model to be retrained, even for the features share by this specie and the yeast.

While this model reduces the amount of experiments needed to determine the functions of all the genes of a specie, it still requires a large amount of experiments. Especially for newly discovered species, it would be interesting to determine the threshold in terms of the number of experiments required to achieve good accuracy, *i.e.* the most significant features, in order to limit the amount of biological experiments needed, and thus reduce the cost and length of the process.

6. Code and Contributions

Code: https://github.com/BenoitPitClaudel/CS229

Laura: selection of paper to replicate, collection and processing of microarray data, implementation of Bayes net predict (and all code in project.py), contributed to poster and writeup.

Benoit: collection and processing of GO annotations, collection and processing of BioGRID pairwise interaction data, SVM experiments and parameter selection, contributed to poster and writeup.

References

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

[2] Expansion of the Gene Ontology knowledgebase and resources. *Nucleic Acids Res.*, 45(D1):D331–D338, 01 2017.

[3] Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006.

[4] The UniProt Consortium. Uniprot: the universal protein knowledgebase. *Nucleic Acids Research*, 45(D1):D158–D169, 2017.

[5] Obonet. `https://pypi.org/project/obonet/`.

[6] Networkx. `https://networkx.github.io/`.

[7] A. Chatr-Aryamontri, R. Oughtred, L. Boucher, J. Rust, C. Chang, N. K. Kolas, L. O'Donnell, S. Oster, C. Theesfeld, A. Sellam, C. Stark, B. J. Breitkreutz, K. Dolinski, and M. Tyers. The BioGRID interaction database: 2017 update. *Nucleic Acids Res.*, 45(D1):D369–D379, 01 2017.

[8] C. Stark, B. J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers. BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.*, 34(Database issue):D535–539, Jan 2006.

[9] Tanya Barrett, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L. Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. Ncbi geo: archive for functional genomics data sets–update. *Nucleic Acids Research*, 41(D1):D991–D995, 2013.

[10] European bioinformatics institute. `https://www.ebi.ac.uk/ols/ontologies/mi`.

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

[12] Paul T. Spellman, Gavin Sherlock, Michael Q. Zhang, Vishwanath R. Iyer, Kirk Anders, Michael B. Eisen, Patrick O. Brown, David Botstein, Bruce Futcher, and Gerald R. Fink. Comprehensive identification of cell cycleregulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297, 1998. PMID: 9843569.

[13] Priya Sudarsanam, Vishwanath R. Iyer, Patrick O. Brown, and Fred Winston. Whole-genome expression analysis of snf/swi mutants of saccharomyces cerevisiae. *Proceedings of the National Academy of Sciences*, 97(7):3364–3369, 2000.

[14] Audrey P. Gasch, Paul T. Spellman, Camilla M. Kao, Orna Carmel-Harel, Michael B. Eisen, Gisela Storz, David Botstein, Patrick O. Brown, and Pamela A. Silver. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11(12):4241–4257, 2018/11/18 2000.

[15] Audrey P. Gasch, Mingxia Huang, Sandra Metzner, David Botstein, Stephen J. Elledge, Patrick O. Brown, and Peter Walter. Genomic expression responses to dna-damaging agents and the regulatory role of the yeast atr homolog mec1p. *Molecular Biology of the Cell*, 12(10):2987–3003, 2001. PMID: 11598186.

[16] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6), 2001.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] Ankur Ankan. pgmpy.