Machine Learning in Computational Biology to Accelerate High-Throughput Protein Expression

Supplementary Information

Anand Sastry[a], Jonathan Monk[a], Hanna Tegel[c], Mathias Uhlen[b,c], Bernhard O. Palsson[a,b], Johan Rockberg*[c], Elizabeth Brunk*[a,b]

* Correspondence should be addressed to: EB (ebrunk@ucsd.edu) and JR (johanr@biotech.kth.se)
[a] Department of Bioengineering, University of California San Diego CA 92093
[b] The Novo Nordisk Foundation Center for Biosustainability, Technical University of Denmark, Horsholm Denmark
[c] KTH Royal Institute of Technology, School of Biotechnology, AlbaNova University Center, SE-106 91 Stockholm, Sweden

Table of Contents:

Supplementary Methods:

Description of Features:

The basic features for the expression dataset includes nucleotide counts, trinucleotide counts, amino acid counts, and the length of the sequences. We also calculated the GC content of both the entire sequence and the first 30 nucleotides. The stability of the mRNA was calculated using UnaFOLD (Markham and Zuker 2008) for both the entire mRNA length and for the first 40 nucleotides, which has been shown to affect protein expression (Kudla et al. 2009). The tRNA adaptation index was also computed for each mRNA strand using the tRNA counts from GtRNAdb for E. coli BL21DE3 (dos Reis, Savva, and Wernisch 2004; Chan and Lowe 2009). Physical properties of the peptides were calculated using the Biopython ProtParam module, such as molecular weight, isoelectric point, aromaticity, instability, grand average value of hydropathy (GRAVY), and overall charge. We also calculated the fraction of various types of amino acids, such as aliphatic, hydrophobic, polar, uncharged polar, positive, negative, sulfur-containing, amide-containing, and alcohol-containing amino acids. Presence of Shine-Dalgarno and Shine-Dalgarno-like sequences were included as features in both the forward and reverse strands, as defined in (Ebrahim et al. 2016). Additional features were generated using structural predictors, since only the primary structure was known for the peptides. We ran all sequences through the secondary structure predictor SCRATCH-1D to generate secondary structure predictions using 3 and 8 letter predictors, and generated solvent accessibility profiles for each sequence (Cheng et al. 2005). From this data, we were able to calculate the fraction of buried and exposed hydrophobic residues. The intrinsically disordered regions of each PrEST were predicted using three different disorder predictors: DisEMBL, RONN, and DISOPRED3 (Jones and Cozzetto 2015; Linding et al. 2003; Yang et al. 2005). All 147 properties were combined to form the feature matrix of the expression dataset. A limited feature set was constructed for the solubility data, including the amino acid counts, the length of the sequences, and the physical properties of the amino acids

and peptides for a total of 38 features (Bazett-Jones et al. 2008). The pipeline to generate the expression feature matrix is included in the GitHub repository at https://github.com/SBRG/Protein_ML as “create_feature_matrix.ipynb”, and the condensed solubility pipeline can be found in the “solubility” directory as “solubility_create_feature_matrix.ipynb”.

Data Preprocessing:

The raw expression dataset consisted of the concentration, source protein, amino acid sequence, and nucleotide sequence for 46,521 PrESTs. Duplicate measurements existed for 1,315 PrESTs, and these measurements were merged by averaging the concentration. The resulting 45,206 PrESTs were split into three expression levels: high expression for PrESTs in the top 25% of concentrations, low expression for the bottom 25%, and mid-expression for all other PrESTs. The mid-expression peptides were removed from the dataset to minimize the effect of noise on the labels.

The initial 16,082 PrESTs from the solubility dataset were divided into 5 solubility classes. The most soluble class consisted of 7,667 PrESTs, and the bottom three solubility classes consisted a total of 3,324 PrESTs. One solubility class was dropped to improve class separation. In order to remove the effects of imbalanced class sizes, the class weightings for the machine learning algorithms were balanced, resulting in similar scores for accuracy, precision, recall, AUC, and F-1 score.

The datasets were then split into a training set consisting of 70% of the data, and a testing set with the remaining data. After the split, the training data was scaled to have zero mean and unit variance. This same scaling factor was applied to the testing set to ensure that no bias was introduced by the testing data in the pre-processing steps.

Machine Learning:

The random forest, support vector machine, and logistic regression machine learning algorithms were implemented using the Sci-kit Learn (Pedregosa et al. 2011) packages for Python, and the neural network was implemented using TensorFlow (Abadi et al. 2015). The free parameters for the Sci-kit Learn models were optimized against a subset of the training data (n = 1000) using 3-fold cross validation. The free parameter for the SVM and Logistic Regressor is the penalty term *C*. The free parameters for the Random Forest classifier were the number of trees in the forest, maximum tree depth, minimum samples per leaf, and minimum samples per split. Increasing the forest size can improve classification accuracy, but reduces speed of computation. The other three parameters can prevent overfitting common to tree-based methods. The deep learning algorithm requires a longer training time, so each hyperparameter (optimization algorithm, architecture, and dropout rate) was individually optimized against the training data. The neural network used 3 layers with 100, 200, and 100 hidden units respectively. A dropout rate of 0.5 was implemented to prevent neuron overfitting (Srivastava et al. 2014), and the network was optimized using the RMSProp optimization algorithm (Tieleman and Hinton 2012). After optimizing the parameters and training the models, an ensemble model was generated by averaging the prediction probabilities of the top two models. Neither dimension reduction using PCA nor feature selection using a Linear Support Vector Classifier provided any improvement on the classification accuracies. This pipeline is included in the github

repository as “classification_workflow.ipynb” and “solubility/solubility_classification_workflow.ipynb” for the expression and solubility datasets, respectively.

Computer-aided Selection of Highly-expressed Proteins:

As with the machine learning workflow, the preprocessing steps included averaging together duplicate measurements and discarding peptides not in the top and bottom 25th percentiles. When generating the HPA dataset, more PrESTs were tested for proteins when their original PrESTs were poorly expressed. This increased the proportion of poorly expressed PrESTs from certain proteins. In order to reduce this bias, proteins with over 5 PrESTs were discarded from the dataset. The training set consisted of all PrESTs from proteins with only one PrEST and PrESTs generated from a randomly selected subset of the remaining proteins. The testing dataset consisted of the remaining PrESTs. The machine learning algorithms were fitted to the data as described above.

In order to minimize the number of experiments, we selected the PrEST from each protein that had the highest probability of success as predicted by the model. The selected PrESTs were validated against their true concentrations from the experimental data, and the PrESTs from all proteins with a correctly selected PrEST were discarded from the testing dataset. This process was repeated until the dataset was exhausted (See Figure 5e and an IPython notebook titled “retrospective_analysis.ipynb”).

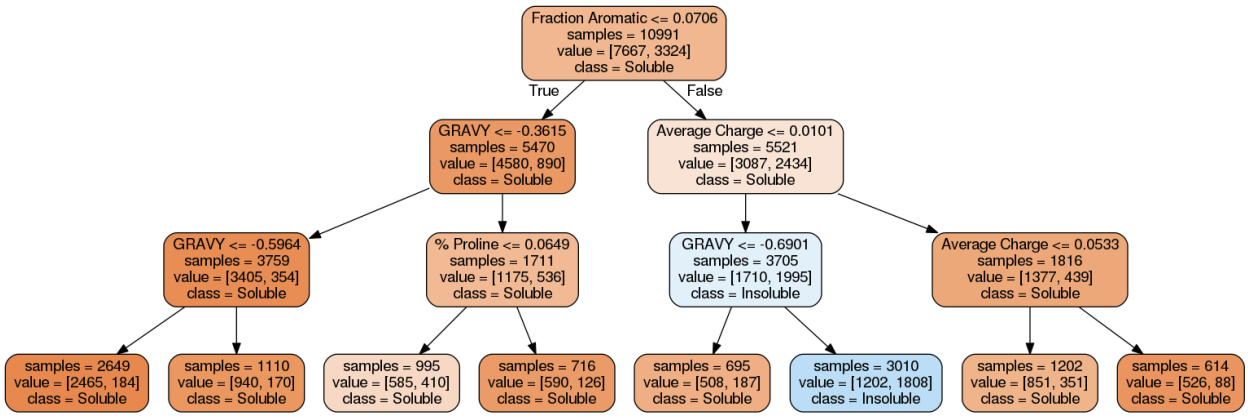
Supplementary Tables

Supplementary Table 1. Relative importance of the top 20 features from the random forest classifier on the expression dataset.

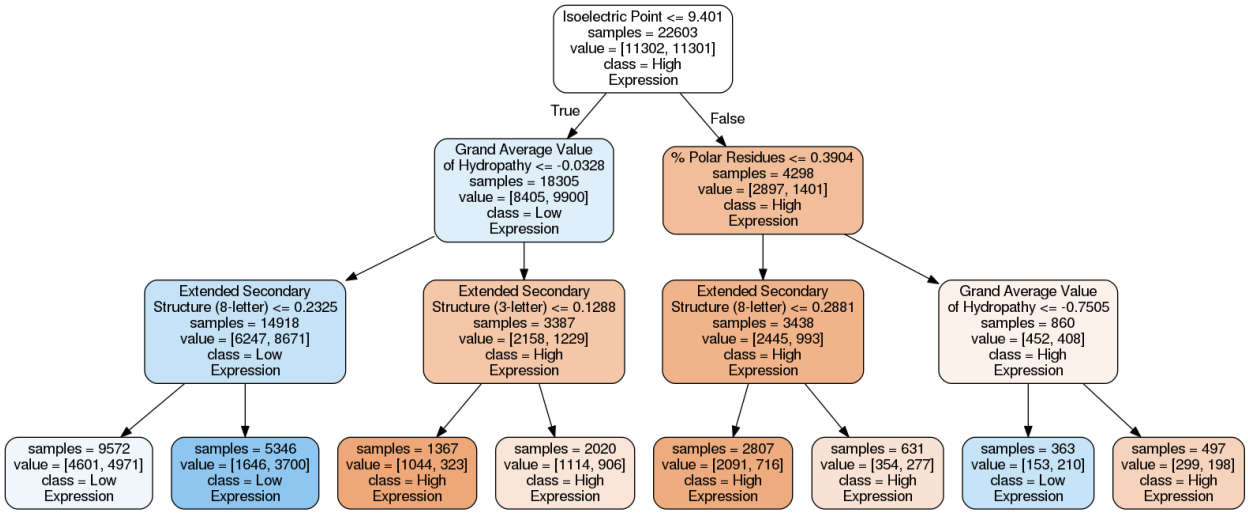
#	Feature	Relative Importance
1	Isoelectric Point	1.000000
2	Grand Average Value of Hydropathy	0.819925
3	% Leucine	0.775152
4	Average Charge	0.732603
5	% Polar Residues	0.728989
6	Extended Secondary Structure (8-letter)	0.693007
7	% Hydrophobic Solvent-Inaccessible Residues	0.688061
8	Extended Secondary Structure (3-letter)	0.672070
9	% Tyrosine	0.661099
10	Average GRAVY of Solvent-Inaccessible Residues	0.614977
11	% Hydrophobic Residues	0.561206
12	Alpha Helix Secondary Structure (8-letter)	0.555996
13	Helical Secondary Structure (3-letter)	0.554443
14	Average Disorder (RONN)	0.505789
15	Molecular Weight	0.500565
16	Fraction of Disordered Residues (DisEMBL COILS)	0.454956
17	% Negatively Charged Amino Acids	0.454504
18	% Uncharged Polar Amino Acids	0.444589
19	% Threonine	0.433684
20	% Proline	0.432263

Supplementary Figures

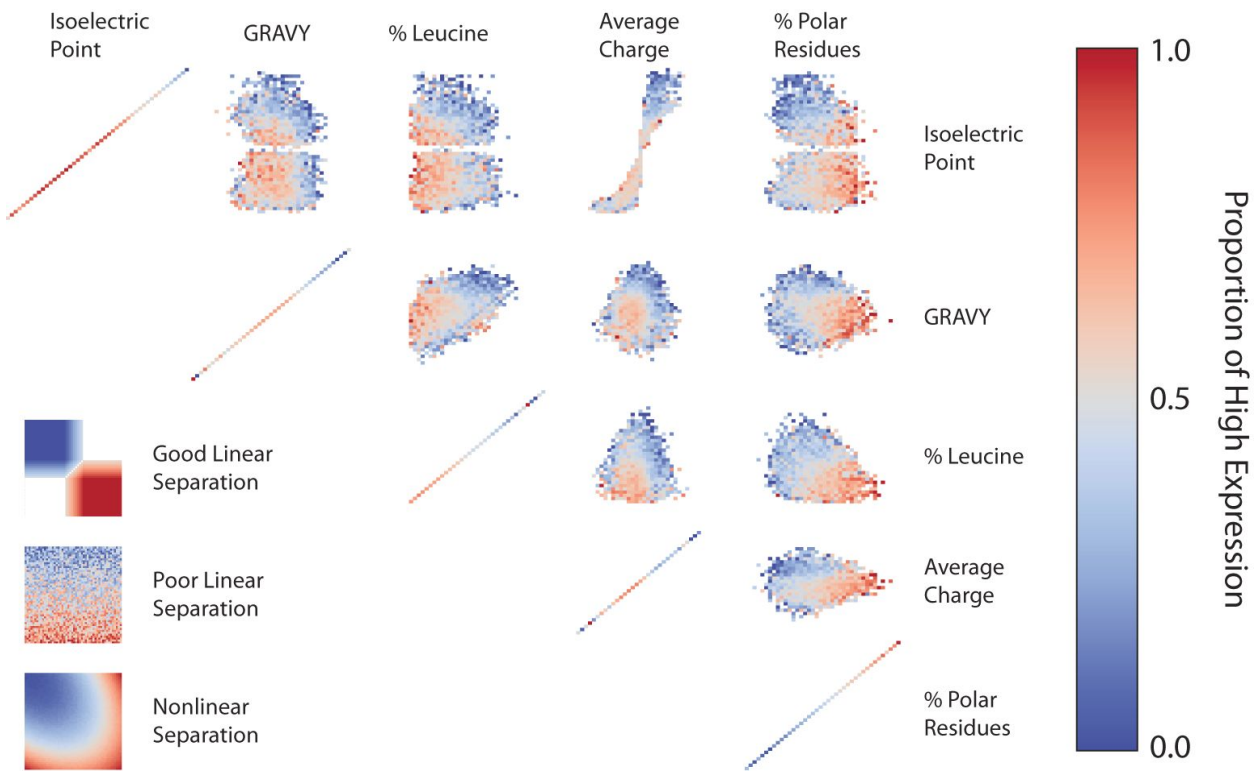
a)



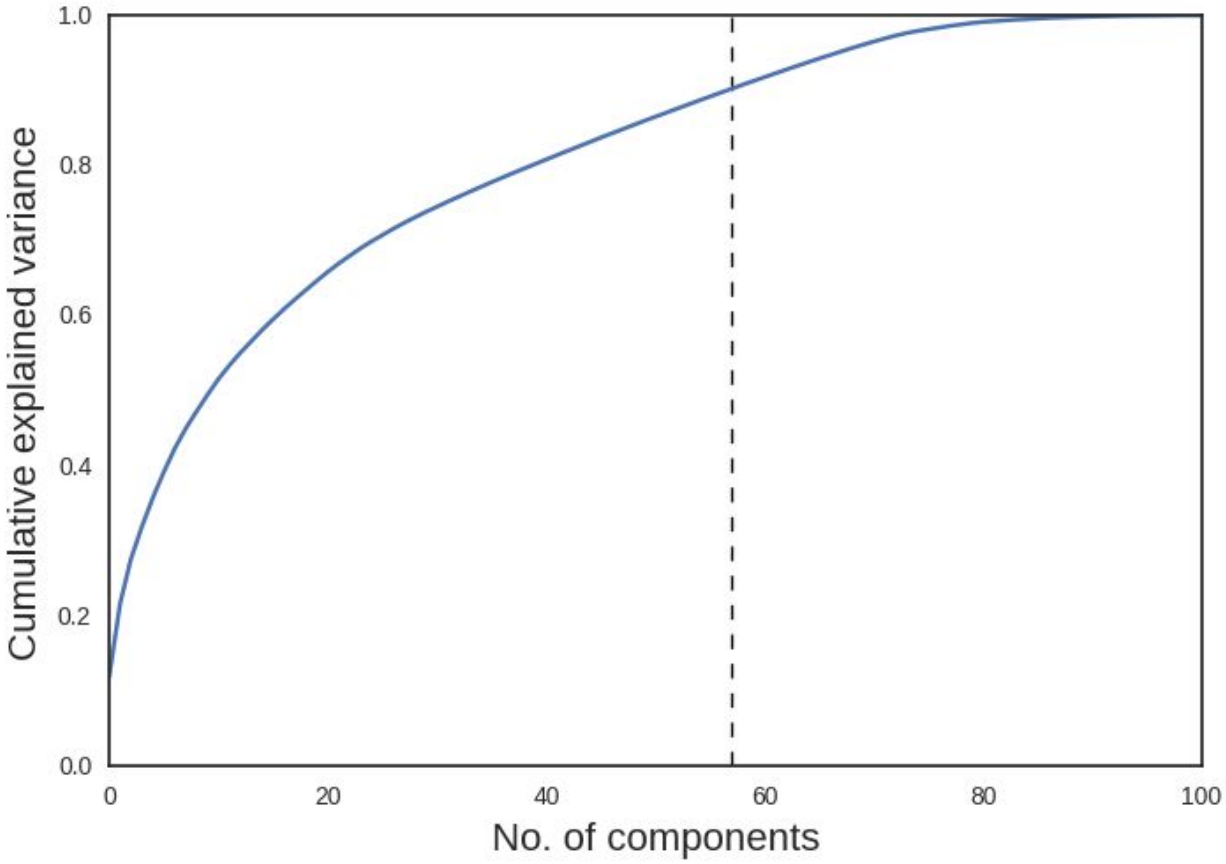
b)



Supplementary Figure 1: Decision tree to separate a) soluble and insoluble proteins and b) high- and low-expression proteins based on the top 5 features from their respective random forest models. The trees have been pruned to a depth of 3 for readability.



Supplementary Figure 2: Heatmap displaying the separation in expression level achieved by the top five features as determined from the random forest algorithm. Features with a strong influence on expression level will have discernable clusters of dark red or blue, indicating regions of mostly high or mostly low expressed proteins respectively. Note the nonlinear effects of the combined features.



Supplementary Figure 3: Number of dimensions captured by the feature set as calculated by PCA. 57 components capture 90% of the variance, displaying both the level of redundancy in some features and the multidimensionality of the feature set.

Supplementary References

Abadi, Martın, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, et al. 2015. “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015.” *Software Available from Tensorflow. Org* 1.

Bazett-Jones, David P., Ren Li, Eden Fussner, Rosa Nisman, and Hesam Dehghani. 2008. “Elucidating Chromatin and Nuclear Domain Architecture with Electron Spectroscopic Imaging.” *Chromosome Research: An International Journal on the Molecular, Supramolecular and Evolutionary Aspects of Chromosome Biology* 16 (3): 397–412.

Chan, Patricia P., and Todd M. Lowe. 2009. “GtRNAdb: A Database of Transfer RNA Genes Detected in Genomic Sequence.” *Nucleic Acids Research* 37 (Database issue): D93–97.

Cheng, J., A. Z. Randall, M. J. Sweredoski, and P. Baldi. 2005. “SCRATCH: A Protein Structure and Structural Feature Prediction Server.” *Nucleic Acids Research* 33 (Web Server issue): W72–76.

Ebrahim, Ali, Elizabeth Brunk, Justin Tan, Edward J. O’Brien, Donghyuk Kim, Richard Szubin, Joshua A. Lerman, et al. 2016. “Multi-Omic Data Integration Enables Discovery of Hidden Biological Regularities.” *Nature Communications* 7 (October): 13091.

Jones, David T., and Domenico Cozzetto. 2015. “DISOPRED3: Precise Disordered Region Predictions with Annotated Protein-Binding Activity.” *Bioinformatics* 31 (6): 857–63.

Kudla, Grzegorz, Andrew W. Murray, David Tollervey, and Joshua B. Plotkin. 2009. “Coding-Sequence Determinants of Gene Expression in Escherichia Coli.” *Science* 324 (5924): 255–58.

Linding, Rune, Lars Juhl Jensen, Francesca Diella, Peer Bork, Toby J. Gibson, and Robert B. Russell. 2003. “Protein Disorder Prediction: Implications for Structural Proteomics.” *Structure* 11 (11): 1453–59.

Markham, Nicholas R., and Michael Zuker. 2008. “UNAFold: Software for Nucleic Acid Folding and Hybridization.” *Methods in Molecular Biology* 453: 3–31.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. 2011. “Scikit-Learn: Machine Learning in Python.” *Journal of Machine Learning Research: JMLR* 12 (Oct): 2825–30.

Reis, Mario dos, Renos Savva, and Lorenz Wernisch. 2004. “Solving the Riddle of Codon Usage Preferences: A Test for Translational Selection.” *Nucleic Acids Research* 32 (17): 5036–44.

Srivastava, Nitish, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting.” *Journal of Machine Learning Research: JMLR* 15 (1): 1929–58.

Tieleman, Tijmen, and Geoffrey Hinton. 2012. “Lecture 6.5-Rmsprop: Divide the Gradient by a Running Average of Its Recent Magnitude.” *COURSERA: Neural Networks for Machine Learning* 4 (2).

Yang, Zheng Rong, Rebecca Thomson, Philip McNeil, and Robert M. Esnouf. 2005. “RONN: The Bio-Basis Function Neural Network Technique Applied to the Detection of Natively Disordered Regions in Proteins.” *Bioinformatics* 21 (16): 3369–76.