BENG 213 - Metabolic Promiscuity in *E.coli* - Checkpoint 3

Jason Hyun, Kevin Glass

Abstract

Enzyme promiscuity, or the ability of an enzyme to catalyze multiple reactions, is a complex protein feature present in over a third of the metabolic enzymes in *E. coli*. We demonstrate here several machine learning approaches to predict enzyme promiscuity from enzyme substrate structures, protein domain annotations, and protein sequences. Best performance was observed using an LSTM to predict promiscuity directly from sequence (74.4% accuracy), followed by k-NN with substrate structures (72.7%), and SVM with domain annotations (68.5%). However, this performance is only somewhat better than random prediction and it is insufficient to say that enzyme promiscuity has been learned by these models; rather it appears likely the instances of enzyme promiscuity are unique phenomena, and that there may be no global patterns to be learned. We conclude with an algorithmic approach to predict reactions in *E. coli* directly by clustering known substrates, which may be a better alternative for discovering new reactions than predicting enzyme promiscuity in general.

Literature Review

E. coli is a commonly studied gram negative bacterium commonly used in lab experiments, and has had its metabolic network reconstructed repeatedly [1]. The model of E. coli we used for this project was iML1515, which features 1,515 open reading frames, 2719 reactions with 1,192 unique metabolites. The model is also linked to 1,515 protein structures [2], which allows us to incorporate and analyze promiscuity as being related to variety of data types provided by the model, such as protein structural information, metabolite structural information, gene activity, network connectivity, and enzyme family. Enzyme promiscuity, as defined by enzymes catalyzing multiple different reactions for which they are not normally recognized, can be quantized using multiple different approaches chemical [3], however for our purposes, we are defining promiscuity to be one gene set catalyzing more than one reaction. Collectively, the metabolism of promiscuous enzymes beyond their normal recognized catalytic function can be referred to as a cell's "Underground metabolism" [5]. Previous models have been generated of E. coli's underground metabolism, incorporating existing enzyme chemical information to an existing genome scale model of E. coli from Brenda [4]. Enzyme promiscuity data for this reaction network was generated to study the side reactions present in the network and to better understand the shadow metabolism, and thus how E. coli might respond to different conditions.

Enzyme promiscuity data has been previously incorporated into genome scale models of E. coli [4], and has been used to try and understand the effects of the underground metabolism on network dynamics. Furthermore, previous studies have also classified enzymes within E. coli as generalists (catalyzing more than one reaction) versus specialists (only catalyzing one reaction) and have analyzed various network properties along this spectrum of specificity [6]. Furthermore, metabolic networks have been analyzed with in vivo experimentation to better understand promiscuity within a network, and determine previously unknown underground activity [7]. These analyses have either incorporated enzyme promiscuity data into known biochemical reaction networks, analyzed promiscuity from a network perspective, or have tried to experimentally determine promiscuity based on in-vivo knockouts. Previous analysis have yielded that promiscuous reactions can be incorporated into existing reaction networks, and can provide advantages for growth under specific conditions, can be predicted using a combination of gene knockouts and the already established metabolic network, and can provide resistance to perturbations in a cells requirement and may require less complex regulation. Thus, enzyme promiscuity in a network has been studied in the context of experimental gene knockouts, metabolic network structure, metabolite similarity, protein structural information, and fluxomic information.

Open areas of research are whether enzyme promiscuity can be predicted using supervised learning methods by utilizing structural, fluxomic, metabolic, gene knockout information, and whether existing reactions can provide a framework for discovering unknown promiscuous reactions. Furthermore, can we use a combination of these factors to determine possible areas within a metabolic network that promiscuity might exist? Also, as generalist and specialist enzymes have been hypothesized to evolve from duplicates of a given enzyme, can enzyme promiscuity be understood in the context of relatedness to a specialized, high flux enzyme, or pathway position? Furthermore, possible applications of group contribution for calculating metabolite similarity, and ergo likely substrate promiscuity is also an area of potential exploration.

Methods

Comparison of enzymes on the basis of shared substrates and impact on promiscuity

Defining enzymes and promiscuity directly from iML1515

Our analysis to date is based on the iML1515 reconstruction of *E. coli* [2]. We first defined an enzyme as a set of gene(s) able to independently catalyze at least one reaction in iML1515. This was accomplished by iterating through each reaction in iML1515, splitting the boolean GPRs at each ‘OR’, treating each gene (or group of genes if ‘AND’s are present) as an enzyme, and assigning the reaction to the enzyme. Multiple reactions were assigned to an enzyme if its gene group was encountered more than once during this process. This yielded 1242 enzymes, of which 974 were not related to transport. Each enzyme was also verified to be consistent with the GPR of every reaction assigned to that enzyme. Finally, Uniprot accession IDs for each gene was extracted from iML1515-GEMPro and assigned to the appropriate enzyme(s); while not analyzed at this checkpoint, these annotations may be used in the future to integrate protein structure data.

We defined promiscuity as whether or not more than one reaction is assigned to an enzyme. **Figure 1** shows a histogram of the number of reaction assigned per enzyme for non-transport enzymes, ranging from 1 to 41 reactions. About two-thirds of the enzymes were specific, and most promiscuous enzymes catalyzed 5 or fewer reactions.

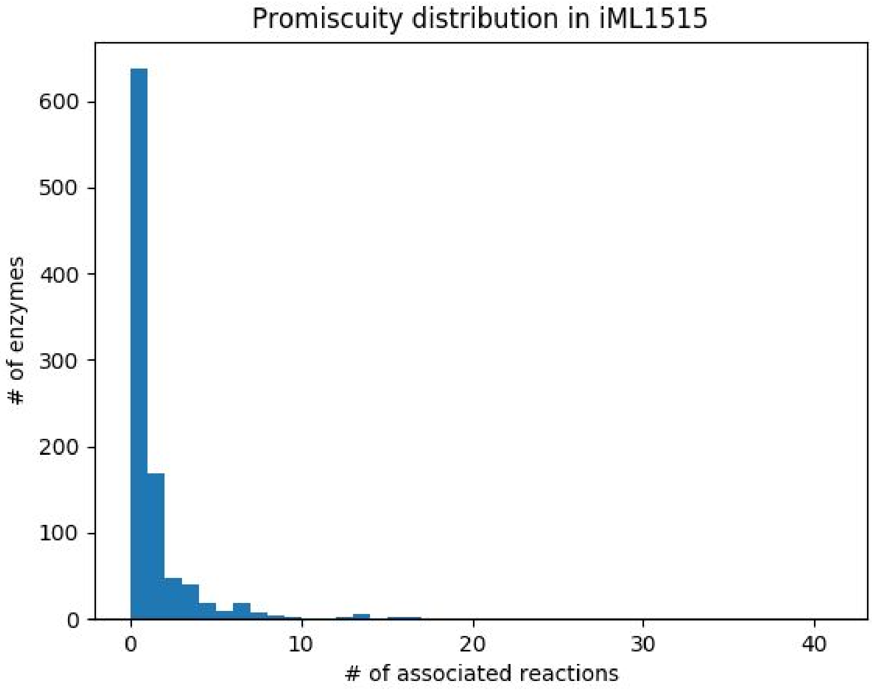

Figure 1: Histogram of # reactions enzymes for the 974 non-transport enzymes extracted from iML1515.

Constructing a distance matrix between enzymes based on shared substrates

In order to quantify the relationships between enzymes encoded in iML1515, we first examined the substrates associated with each enzyme. For a given enzyme, we defined its substrate set as the union of all substrates across all reactions associated with the enzyme. Reaction products were also included for reversible reactions, and H_2O and H^+ were excluded. A similarity matrix was constructed for all pairs of enzymes using the Jaccard index between their substrate sets. Similarly, a distance matrix was constructed as 1 - Jaccard index.

The Jaccard index was used as a simple exploratory metric to compare enzymes, based on the hypothesis that enzymes that share substrates or reactions may have similar structures / active sites and possibly similarly promiscuity properties.

Visualization of shared-substrate Jaccard distance matrix

We applied three approaches to visualize the substrate-based Jaccard distance matrix: PCA, tSNE, and heatmap with hierarchical clustering. For the PCA plot **(Figure 2)**, data appeared to be structured along three directions when plotted along the first two components; however, the structure appeared independent of enzyme promiscuity, and the first two components captured very little total variance (27.6% and 6.9%). Curiously, when generating a similar PCA plot for a z-score normalized distance matrix, even less variance was captured by the first two components (11.9%, and 5.0%) and again any visible structure appeared independent of enzyme promiscuity. This is likely due to distance matrix being not normally distributed and bounded between [0,1] from the definition of Jaccard index, making z-score normalization not appropriate for this data. Similarly for the tSNE plot **(Figure 3)**, the data formed a single large cluster along with several distant, smaller clusters, but most clusters did not segregate on the basis of enzyme promiscuity. This structure was observed across multiple tSNE visualizations.

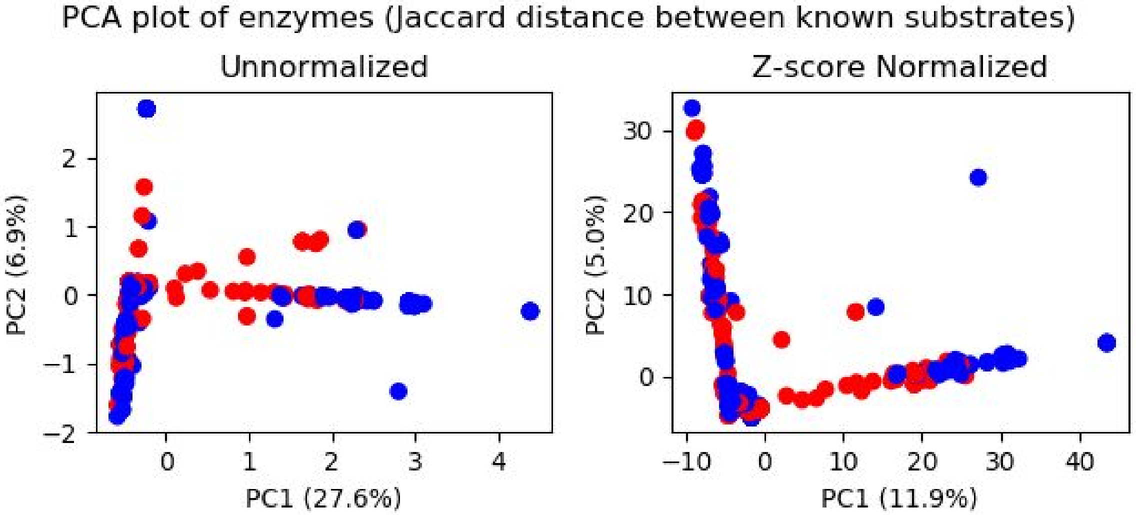

Figure 2: PCA (with and without z-score normalization) plot of substrate-based Jaccard distance matrix for 974 non-transport enzymes extracted from iML1515. Promiscuous enzymes are shown in red, specific enzymes are shown in blue.

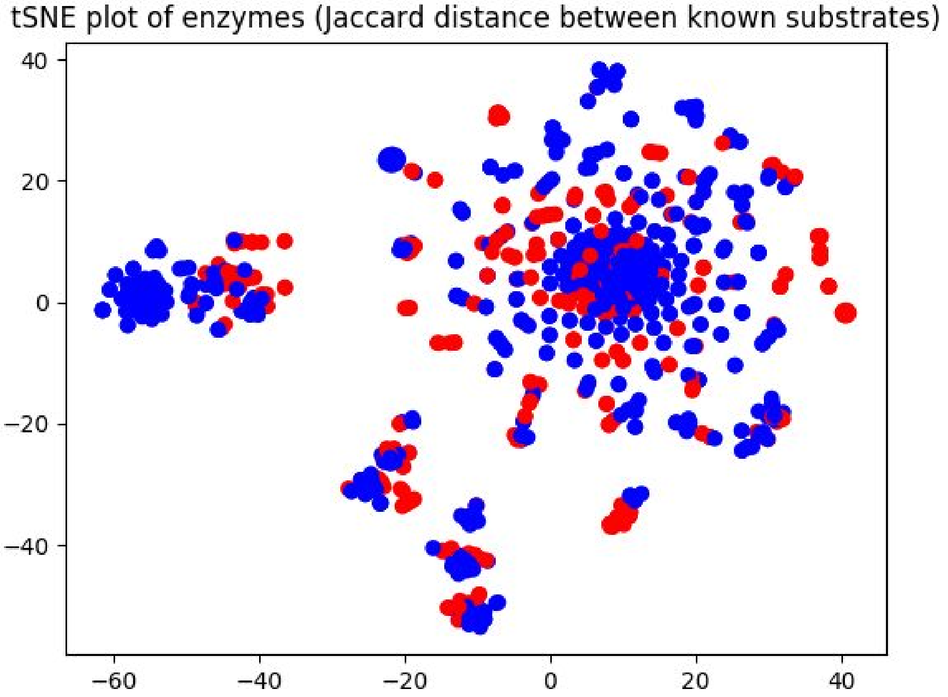

Figure 3: tSNE visualization of substrate-based Jaccard distance matrix for 974 non-transport enzymes extracted from iML1515. Promiscuous enzymes are shown in red, specific enzymes are shown in blue.

For the heatmap visualization, we first clustered the enzymes using three types of hierarchical/agglomerative clustering methods that are compatible with non-Euclidean distances: Nearest Point (single), UPGMA (average), and WPGMA (weighted). **Figure 4** shows dendrograms and heatmaps for these approaches. Clusters or heatmap blocks (i.e. groups of enzymes that share many substrates among themselves) generated by Nearest Point were much different from those derived by UPGMA or WPGMA, which had similar results. UPGMA/WPGMA also appeared to have identified about 5 relatively large clusters, as well as a number of smaller clusters. An interpretation of clusters is in the following section.

Consistency of clusters across different clustering methods using shared-substrate distances

In addition to the hierarchical clustering approaches used for heatmap visualization, we also examined DBSCAN as an alternative approach. These methods were selected as they are able to handle non-Euclidean distances (i.e. Jaccard distances) and uneven cluster sizes, particularly important to preserve smaller clusters in the presence of larger clusters that may form simply due to shared cofactors. DBSCAN was run with many parameters to better understand the appropriate number of clusters to extract from hierarchical clustering. **Table 1** shows the number of clusters (not including unassigned) inferred by DBSCAN with different neighborhood sizes and minimum cluster sizes:

Table 1: Number of enzyme clusters inferred by DBSCAN from Jaccard distance matrix using different neighborhood sizes (ε) and minimum cluster sizes.

ε = 0.5	Min. Cluster Size:	2	3	4	5	6	7	8	9	10	11	12
	# Clusters:	135	63	35	21	13	13	7	5	4	4	3
ε = 0.7	Min. Cluster Size:	2	3	4	5	6	7	8	9	10	11	12
	# Clusters:	71	34	25	13	7	9	8	5	5	6	4
ε = 0.9	Min. Cluster Size:	2	3	4	5	6	7	8	9	10	11	12
	# Clusters	27	13	9	6	3	2	2	2	2	2	2

We found that 13 clusters were observed for multiple settings, (ε = 0.5, min_size = 6 or 7; ε = 0.7, min_size = 5; ε = 0.9, min_size = 3) and that below 9 clusters, the total number of clusters was unstable with respect to DBSCAN parameters (i.e. for epsilon = 0.7, total clusters observed would fluctuate up and down between 9 and 4 as the minimum cluster size increased from 6 to 12). Additionally, it is possible to infer 10-13 clusters by visual inspection of the tSNE plot from **Figure 3**. From this, we adjusted the cut threshold of each hierarchical clustering method to produce 13 clusters. This resulted in six different clustering methods: hierarchical clustering to extract 13 clusters using Nearest Point, UPGMA, or WPGMA, and three DBSCAN methods:

1. DBSCAN, ε = 0.5, minimum cluster size = 6 (13 clusters + 769 unassigned)
2. DBSCAN, ε = 0.7, minimum cluster size = 5 (13 clusters + 386 unassigned)
3. DBSCAN, ε = 0.9, minimum cluster size = 3 (13 clusters + 132 unassigned)

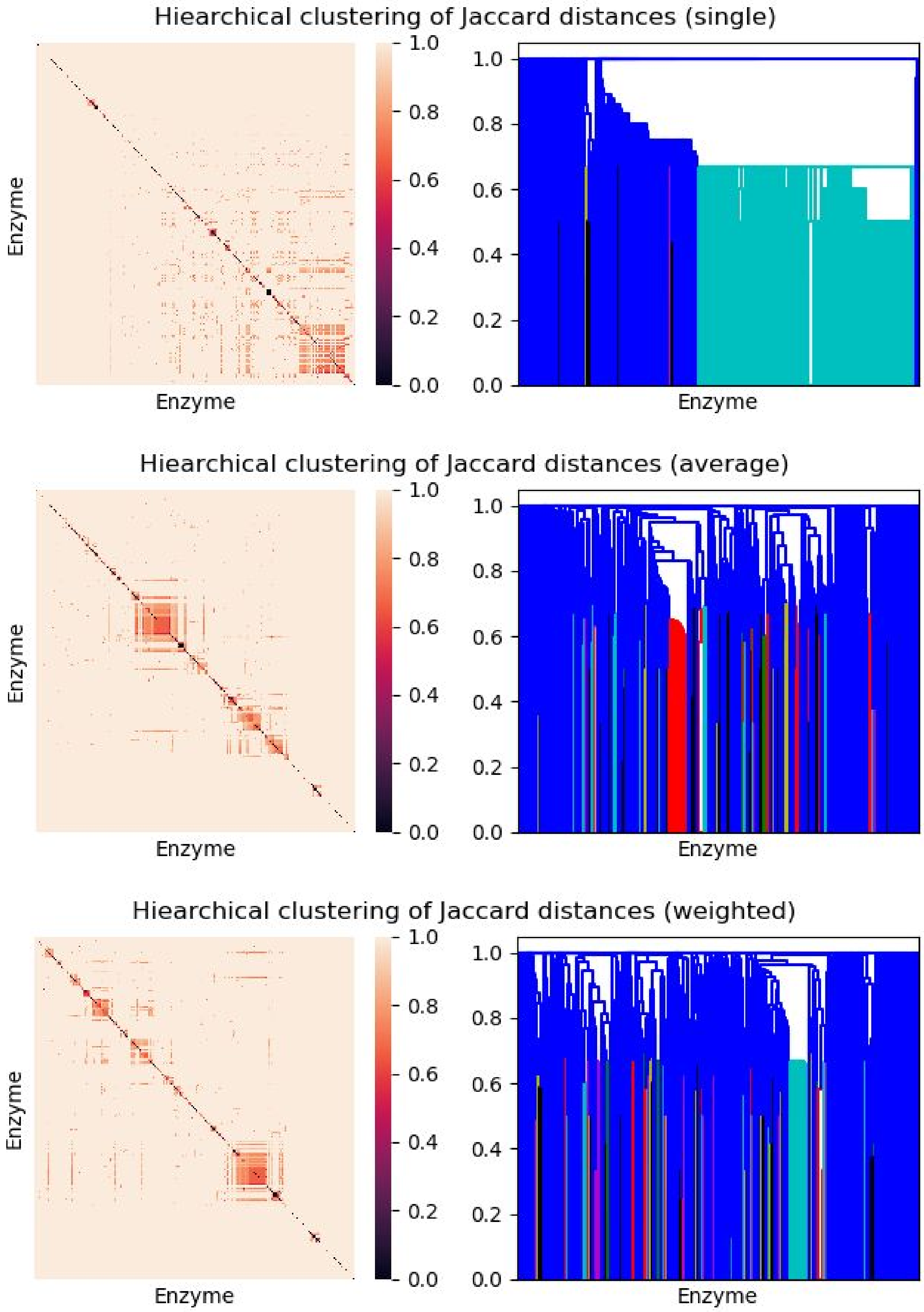

Figure 4: Heatmap and dendrogram of substrate-based Jaccard distances after hierarchical clustering of 974 non-transport enzymes extracted from iML1515. Nearest Point (single), UPGMA (average), and WPGMA (weighted) was used for clustering.

To evaluate how consistent clusters were across the six approaches, we computed the Rand Index between each pair of clusterings (i.e. the fraction of pairs of elements that are either in the same cluster or in differents clusters for both clusterings). **Figure 5** shows a heatmap of the pairwise Rand Indexes. As previously observed for Hierarchical Clustering, we found that UPGMA and WPGMA produced very similar clusterings that were distinct from the Nearest-Point clustering. Additionally, we found that the DBSCAN clustering with large neighborhoods (ε = 0.9) was similar to that of UPGMA/WPGMA hierarchical clustering, while DBSCAN with smaller neighborhoods was quite different; however, this may be due to the large number of unassigned data points when using smaller neighborhoods in which the Rand Index would inflate how different two clusterings are. Taking this into account, there appeared to be two distinct clusterings between the methods, one derived from DBSCAN or Hierarchical Clustering with either UPGMA or WPGMA, and another from Hierarchical Clustering with Nearest-Point.

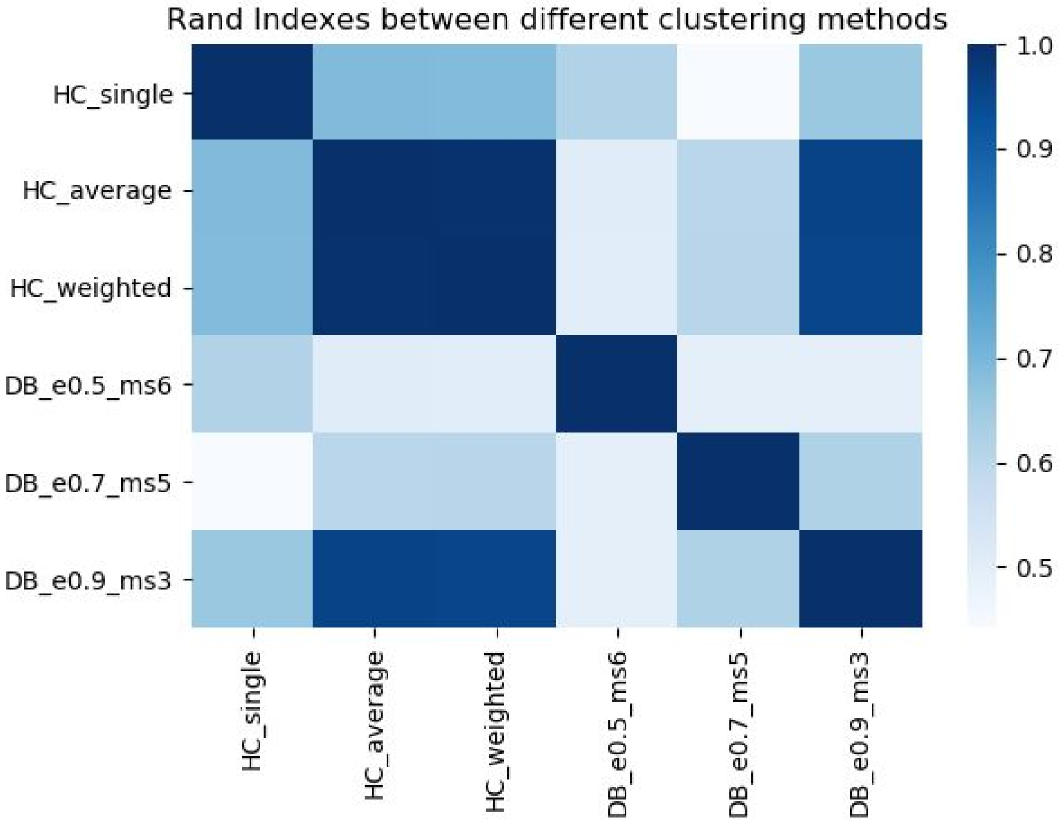

Figure 5: Rand Indexes between different clustering methods applied to Jaccard distances between the substrates of the 974 non-transport enzymes extracted from iML1515.

However, upon further analysis, we found that all three Hierarchical Clustering methods had essentially failed to discern significant small clusters, each producing a single very large cluster with all remaining clusters containing just 1-2 points each; in order for for more than one sizable cluster to be observed, more than 100 clusters would have to be extracted from the hierarchy, in which still most clusters would still contain 1-2 points. In contrast, DBSCAN with large neighborhoods (method 6) produced a similar very large cluster, but also several smaller clusters with at least 5 points each. This is significant, as clusters of size 2 may likely be consecutive reversible enzymes in a pathway (i.e. A ↔ B and B ↔ C), while larger clusters can illustrate more complex relationships with regards to shared substrates. This suggests that the Rand Index may not be suitable for comparing clustering in this case as cluster sizes are very uneven and dominated by a single large cluster.

Interpretation of shared-substrate-based clusters with respect to enzyme promiscuity

Proceeding with the 13 clusters derived by DBSCAN with large neighborhoods (method 6, shown), we first checked if any of the clusters were enriched for either specific or promiscuous enzymes. In general, the clustering was overall not very predictive of enzyme promiscuity, with a homogeneity of 10.2% and a completeness of 7.9%. However, this may be due to the single large, essentially uninformative cluster generated by all tested clustering approaches; several of the smaller clusters DBSCAN produced were highly enriched for either promiscuous or specific enzymes (**Table 2**).

Table 2: Characterization of clusters derived from DBSCAN using ε = 0.9, minimum cluster size = 3. Dominant subsystem is defined as any subsystem that represents at least half of all the unique reactions that are catalyzed by enzymes in a cluster, as annotated in iML1515. The largest two clusters (0 and -1, essentially unassigned enzymes) were not examined.

Label	Cluster Size	# Promiscuous Enzymes	Percent Promiscuous	# Unique Reactions	Dominant Subsystems
0	763	261	34.21	-	-
1	3	3	100	29	Cell Envelope Biosynthesis (25/29)
2	4	4	100	33	Nucleotide Salvage Pathway (20/33)
3	34	34	100	29	Murein Biosynthesis (15/29) Murein Recycling (14/29)
4	4	3	75	3	Alternate Carbon Metabolism (3/3)
5	5	4	80	4	Diverse
6	6	2	33.33	5	Nucleotide Salvage Pathway (3/4)
7	5	3	60	6	Lipopolysaccharide Biosynthesis / Recycling (4/6)
8	3	1	33.33	2	Diverse
9	5	1	20	6	Cofactor and Prosthetic Group Biosynthesis (6/6)
10	3	0	0	3	Diverse
11	3	0	0	3	Cell Envelope Biosynthesis (3/3)
12	4	0	0	4	Lipopolysaccharide Biosynthesis / Recycling (4/4)
-1	132	21	15.91	-	-

Among the smaller clusters, notably clusters 1 and 2 were composed of a small number of highly promiscuous enzymes (cluster size << number of unique reactions). Cluster 3, the “largest” small cluster was composed a set of redundant, promiscuous enzymes involved in Murein Biosynthesis and Recycling. Overall, we found that many of these smaller clusters are enriched for either promiscuous or specific enzymes, but for the most part produced a clustering along subsystem annotations.

Finally, for evaluation of clustering independent of promiscuity labels, we applied the same PCA and heatmap visualization from before to the DBSCAN clusters (**Figure 6**). DBSCAN with large neighborhoods seemed to have failed to reproduce many of the loosely related blocks that was visible from the hierarchically clustered heatmaps (**Figure 4**), though it did manage to isolate smaller, more tightly connected blocks (darker blocks in bottom right of heatmap). Additionally, the small clusters were not well separated based on the PCA plot, with almost all of the remaining variance occuring in the largest, uninformative cluster. This is consistent with the Silhouette coefficient (ratio of mean intra-cluster distance to mean nearest-cluster distance) being just 3.8%. It is possible that the nature of this dataset with non-Euclidean, bounded distances and very uneven cluster size distribution is to blame for the poor clustering performance.

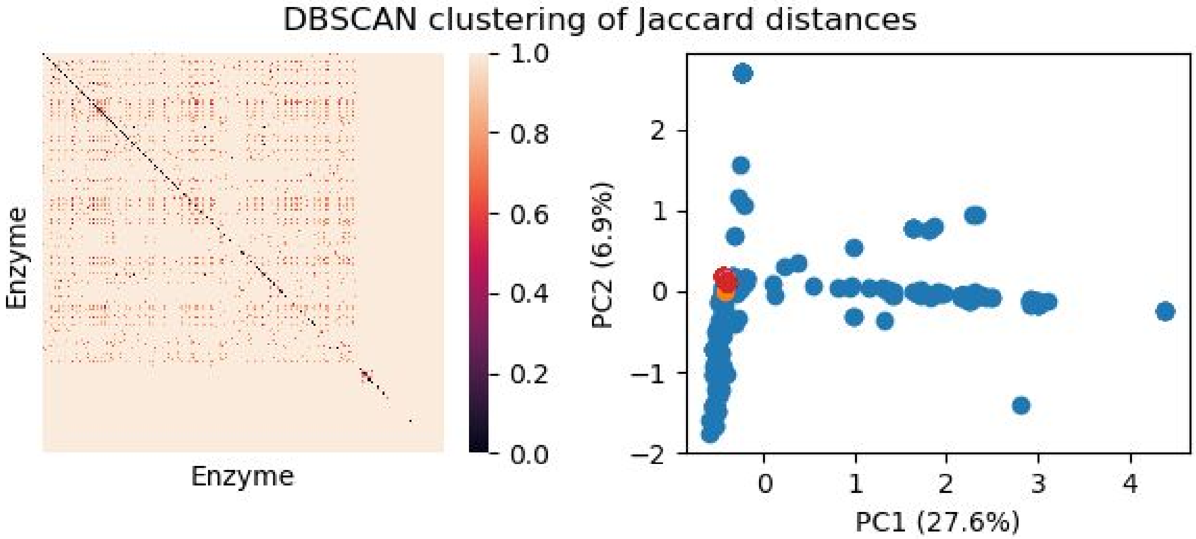

Figure 6: Visualization of DBSCAN clustering applied to Jaccard distances. While 13 clusters are present, all except the largest cluster are concentrated in a single spot in the PCA plot.

Ultimately, examining just the Jaccard distances between enzymes’ known substrates does not appear to be very informative with regards to promiscuity. While revisiting this analysis with a much larger number of clusters (as was visible from hierarchical clustering) may expose additional, smaller patterns between substrate and promiscuity, we do not think it is likely that just this data will help identify novel candidates for promiscuous enzymes. Moving forward, we examine other data types concerning protein structure and metabolite structure to predict enzyme promiscuity.

Predicting enzyme promiscuity from CATH domain annotations

Domain-based characterization of enzymes extracted from iML1515 using CATH

We used the CATH database to characterize enzymes extracted from iML1515 in the previous checkpoint at a protein domain-level. CATH domains were used instead of the existing GEM-Pro domain annotations for two reasons, 1) original annotations were a mixture of SCOPe and PDP annotations without a unified hierarchy, and 2) CATH annotations were available for nearly all PDB structures. For each enzyme (defined as a set of genes), we took the PDB ID of each gene in the enzyme from GEM-Pro, then looked up all domains associated with each PDB using the CATH database to create domain annotations for each enzyme (**Figure 7**).

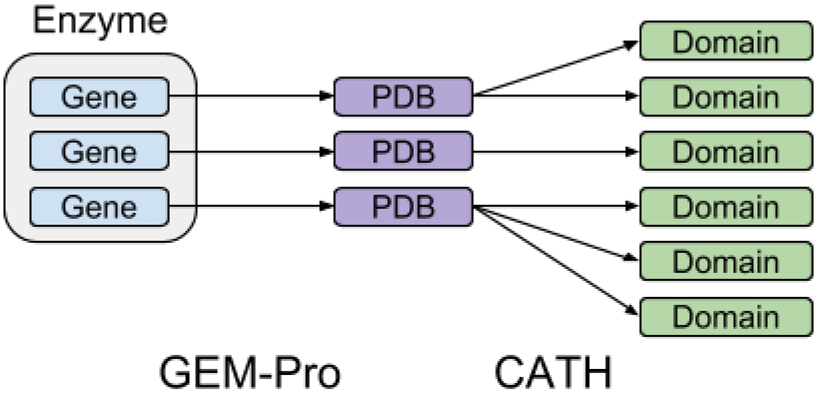

Figure 7: Pipeline for domain annotation of iML1515 enzymes

Of the 974 non-transport enzymes derived from iML1515, 523 enzymes had PDB annotations for some or all its genes. Of those, 504 enzymes had CATH annotations for some or all of their associated PDB structures, encompassing 2687 unique CATH domains. 447 enzymes had complete CATH annotations (i.e. all genes had PDBs, which all had CATH annotations).

Testing associations between CATH domain clusters and enzyme promiscuity

With these annotations, we aimed to test whether there are domains or domain clusters in the CATH hierarchy that are associated with enzyme promiscuity. As a preliminary step, we visualized the domain annotations in the hierarchy to a depth of 4 (CATH hierarchy has a depth of 8), at which most leaf nodes had less than 5 enzymes associated (**Figure 8**). On visual inspection we could not see any obvious domain clusters of significant size that appeared to be strongly associated with either promiscuity or specificity.

To test this more rigorously, we applied a statistical test to all domain clusters of size 10 or greater, of which there were 30. For each domain cluster, a Fisher's exact test was applied to a 2x2 contingency table constructed based on counts of how many enzymes were in/not in the domain cluster and how many were/were not promiscuous.

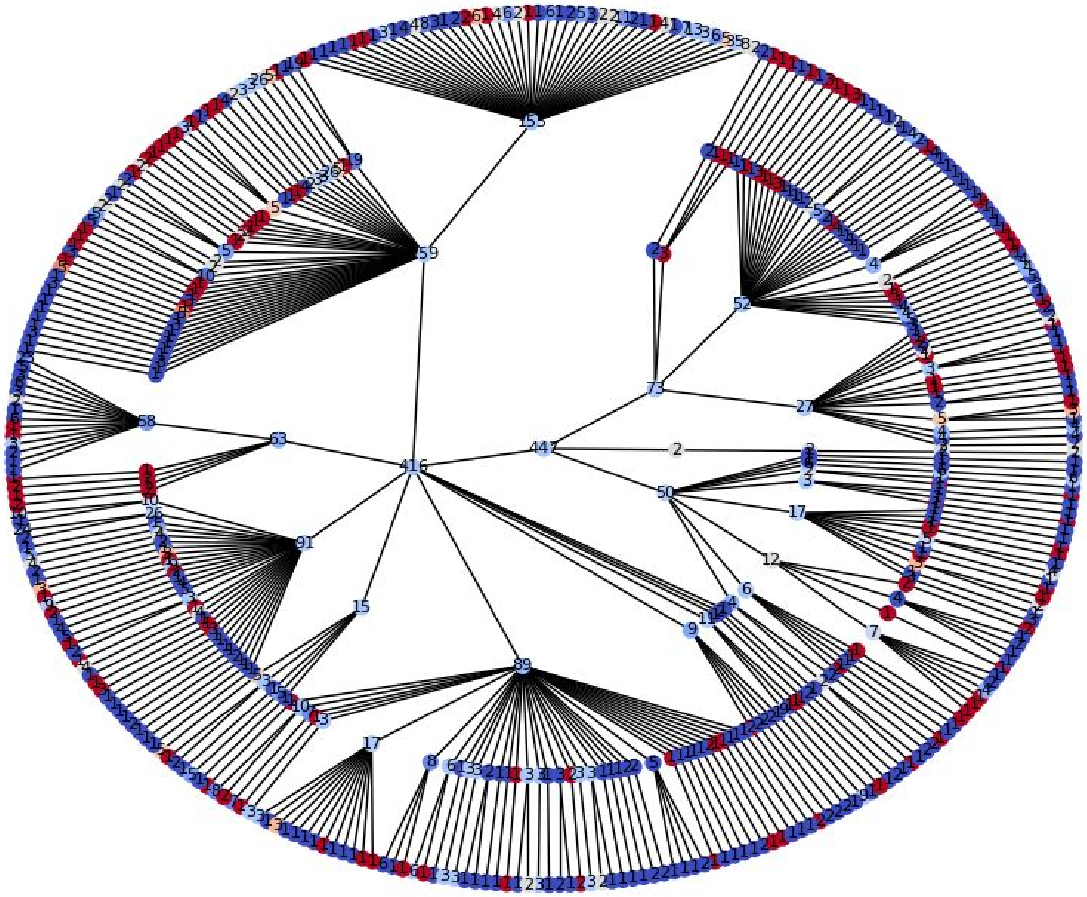

Figure 8: CATH domain hierarchy and enzyme promiscuity in iML1515, depth = 4. Each node corresponds to a CATH domain cluster, labeled by the number of enzymes that contain a domain that falls into that cluster. Nodes are colored based on the fraction of associated enzymes that are promiscuous (red = more promiscuous, blue = less promiscuous).

Without any correction, the Fisher's Exact tests yielded three CATH domain clusters that were significant at a 5% threshold. However, two of those domain clusters (3.40.30 and 3.40.30.10) were identical, as there were no enzymes with domains in the 3.40.30 cluster that were not also in the 3.40.30.10 cluster:

● 3.20.20:	Size = 58,	p-value = 0.01467
● 3.40.30:	Size = 19,	p-value = 0.04547
● 3.40.30.10:	Size = 19,	p-value = 0.04547

Applying a Bonferroni correction, the adjusted significance threshold is 0.05/30 = 0.0017 to assure a FWER below 5%. At this threshold, no domain clusters are significant. Alternatively, with the Benjamini-Hochberg correction to control FDR to be at most 5%, the corrected significant thresholds becomes (0.05/30, 0.05*2/30, 0.05*3/30) = (0.0017, 0.0033, 0.005) for the three domain clusters when ranked by p-value, respectively. However, even with this less conservative correction, still no domain clusters are significant. The controlled FWER or FDR would have to at least 44% (0.01467 * 30) for even a single domain cluster to be considered significant under correction.

This is consistent with our initial observation from the visualization that no large domain cluster appears to be significantly associated with enzyme promiscuity. This suggests that there is no protein domain from the CATH database that is individually informative with regards to enzyme promiscuity, though combinations of domains may be more informative.

Encoding enzyme-level CATH domain annotations as fixed-length vectors

Since each enzyme is associated with a variable number of CATH domains, we considered using one-hot encoding to convert this information into fixed-length vectors, where each domain corresponds to a position in a vector that is either 0 or 1 based on the presence or absence of that domain in a given enzyme. However, as there are thousands of unique CATH domains in our dataset, we leveraged the CATH domain hierarchy for dimension reduction. Similar to our statistical analysis, considered only CATH domain clusters that were associated with some minimum number of enzymes. We then excluded CATH domain clusters in which all child clusters were also included, since a parent cluster is fully dependent on its child clusters. The number of clusters extracted at different cluster size thresholds is below (**Table 3**).

Table 3: Number of independent domain clusters (equal to the dimension of the corresponding domain-based encoding) at varying minimum thresholds for number of associated enzymes.

Minimum domain cluster Size:	5	6	7	8	9
Number of domain clusters:	67	51	40	37	33

The selected domain clusters for a given threshold were used to encode each enzyme, in which each domain cluster corresponds to a position in a vector that is either 0 or 1 based on the presence or absence of a domain within that domain cluster in a given enzyme. We assessed our supervised learning methods for all five encoding lengths.

Model selection for the prediction of enzyme promiscuity from CATH domain annotations

We used the domain encodings the predict enzyme promiscuity (a true/false label, based on whether the enzyme can catalyze multiple reactions in iML1515). To select a model for this binary classification problem, we tested four different models: Support Vector Machine, (Multinomial) Naive Bayes, Decision Tree, and k-Nearest Neighbors. For SVM, linear and radial basis function kernels were tested. For k-Nearest Neighbors, k values from 5 to 20 were tested and results for the highest average accuracy for each encoding is reported. Unspecified parameters were set to defaults as implemented in sklearn. Average and standard deviation of prediction accuracy under 5-fold cross validation for these methods are in **Table 4**.

Across all encoding lengths, the order of performance between the models was the same: SVM (rbf) > SVM (linear) > kNN > Decision Tree > Naive Bayes. Accuracies ranged from 60.2% to 68.5% across all models and encodings; since the maximum average accuracy is low, this suggests that the CATH domain data on its own is not particularly predictive for enzyme promiscuity. Curiously, SVM with a radial basis function kernel performed the best in all cases with by far the least variance in prediction accuracy based on cross-validation, but also performed identically across different encoding lengths. At this time, we do not have an explanation for why SVM with rbf kernel performs identically for different encoding lengths.

Table 4: Performance of different supervised learning models at predicting enzyme promiscuity from CATH domain annotations. Learning tested against multiple possible encodings, based on the number of CATH domain clusters used for encoding. For k-NN, the optimal k between 5 and 20 is reported.

Model	**Accuracy ± Standard Deviation (5-fold CV)**				
SVM (linear)	0.682 ± 0.016	0.667 ± 0.009	0.675 ± 0.023	0.667 ± 0.038	0.673 ± 0.026
SVM (rbf)	0.685 ± 0.002	0.685 ± 0.002	0.685 ± 0.002	0.685 ± 0.002	0.685 ± 0.002
Naive Bayes	0.615 ± 0.018	0.602 ± 0.030	0.617 ± 0.043	0.622 ± 0.040	0.626 ± 0.034
Decision Tree	0.624 ± 0.029	0.637 ± 0.034	0.640 ± 0.042	0.637 ± 0.047	0.633 ± 0.038
k-NN*	0.653 ± 0.019	0.655 ± 0.032	0.644 ± 0.050	0.644 ± 0.057	0.633 ± 0.034
Min. Cluster Size	5	6	7	8	9
Encoding Length	67	51	40	37	33

Performance of SVM at predicting enzyme promiscuity from CATH domains

We proceeded with the SVM model with radial basis function kernel, which consistently demonstrated the highest average accuracy compared to other tested models. To generate precision-recall and ROC curves, we swept through different thresholds for the signed distance from the separating hyperplane at which to label a point as promiscuous or not (in the standard implementation, the threshold is 0; positive signed distances would be assigned promiscuous and negative signed distances would be assigned specific). Positives are enzymes labeled as promiscuous, negatives are enzymes labeled as specific, and PR / ROC curves were generated for different encoding lengths based on the minimum domain cluster size accepted (**Figure 9**).

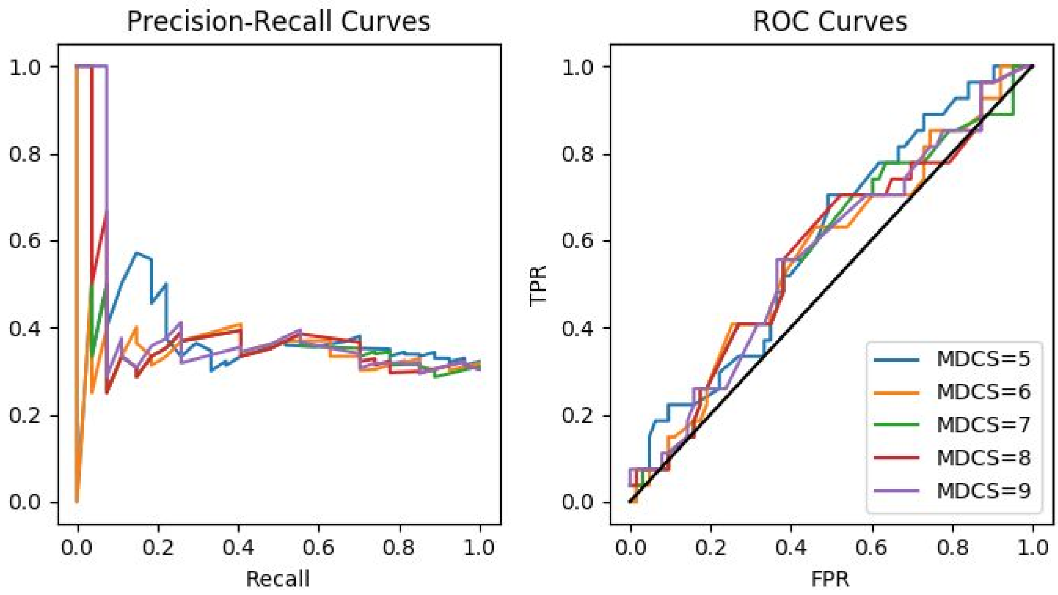

Figure 9: Precision-Recall and ROC curves for SVM model with rbf kernel for predicting enzyme promiscuity from CATH domains. Curves for different encoding lengths based on the minimum domain cluster size (MDCS) used for encoding, as described previously.

Unfortunately, the plots suggest that the SVM model is only marginally better than random classification. For all 5 domain encodings tested, we find then when recall is low (fraction of promiscuous enzymes detected) the precision can be very high (fraction of promiscuous enzyme calls that are correct), but precision rapidly plummets to baseline once recall reaches 25% (about a third of all enzymes are labeled promiscuous). Similarly, all ROC curves lie very close to the random prediction line. Overall, while there may be a better approach towards encoding domain information into vectors for supervised learning, results so far suggests that CATH domains are not very informative with respect to enzyme promiscuity.

Using an LSTM to predict enzyme promiscuity directly from protein sequence

To leverage a more complete dataset containing information regarding enzyme structure, we employed a long-short-term-memory neural network (LSTM) to predict enzyme promiscuity directly from protein sequence. We chose to use an LSTM compared to simpler, sequence- based learning models such as Hidden Markov Models (HMM), since an LSTM is capable of learning longer range dependencies between inputs in a sequence while HMMs are by definition memoryless. Particularly, we used a “many-to-one” implementation of the LSTM, in which a sequence of inputs is used to predict a single label for the whole sequence.

To encode protein sequences as vector sequences, we first tried encoding each amino acid as a 20-length binary vectors, in which all elements are 0 except for the position corresponding to the amino acid. However, this approach encountered technical issues (vanishing gradient problem, very large memory use) due to many protein sequences being very long. To address this issue, we divided sequences into fixed length windows, and encoded each window as a 20-length vector with the counts of each amino acid, normalized to a Euclidean norm of 1. This approach essentially produces a “low-resolution” protein sequence, but offers two benefits: 1) sequence lengths are much shorter, and 2) using all possible reading frames increases the number of data points.

We developed an LSTM in pyTorch to accept a sequence of 20-length vectors, propagate a 10-length state vector, and output a 2-length vector at each step (specific vs promiscuous). We tested window sizes from 1 to 6, in which the data (aggregating all reading frames) was split into a 9:1 training/validation set, and the model was trained for 10 epochs using a cross-entropy loss function. We found that the highest validation accuracy observed during training was 74.4% and was often achieved after a very small number of epochs (**Figure 10**), which is superior to the domain-based methods but not high enough to confidently say that sequence features relevant to enzyme promiscuity have been learned. Interestingly, the highest performance achieved was largely independent of the window size used to generate vectors for LSTM training and validation, which suggests that the “low resolution” protein sequences are just as informative with respect to enzyme promiscuity than complete protein sequences, while allowing for less computationally intensive training.

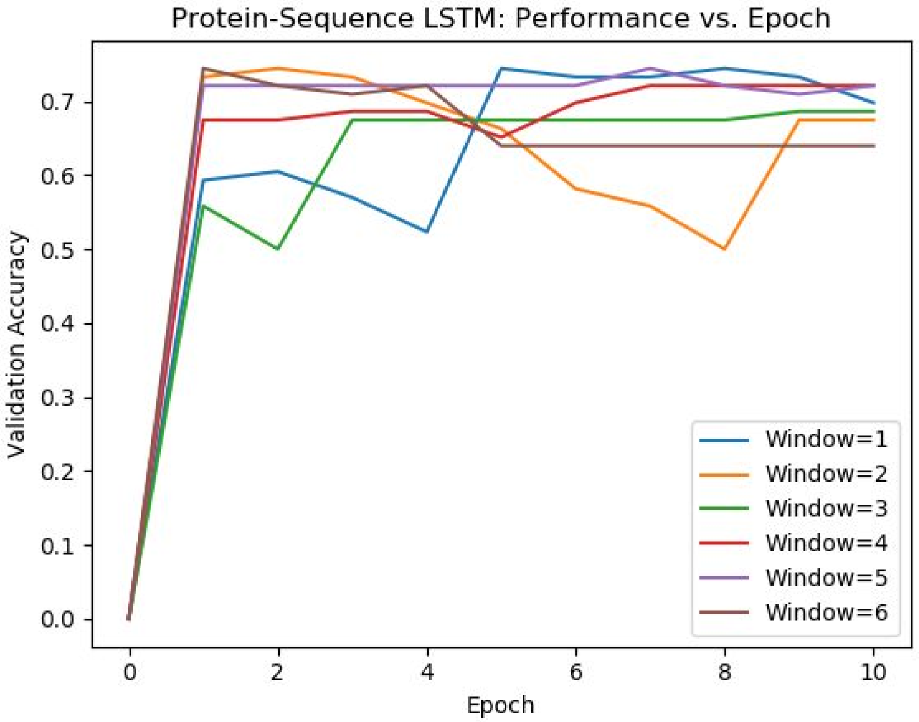

Figure 10: Validation accuracy during training of LSTMs with different window sizes

Predicting enzyme promiscuity from substrate structures

Fixed-length encoding of metabolite structures by molecular fingerprinting

Molecular fingerprinting is the generation of characteristic vectors where each element of the hashed bit vector corresponds to a fragment of a given molecular structure [8]. Among different fingerprinting techniques, we employed a molecule-based fingerprinting method, where all subgraphs are examined up to a certain length, and hashed to a position in a vector. The algorithm used in RDKit by default is similar to the daylight fingerprinting method where, each fingerprint encodes atom sequences up to a specified length (typically from two to seven atoms), with each such sequence being hashed using multiple hashing procedures so that each bit is associated with multiple fragments and each fragment with multiple bit positions.

Using RDKit a chemical analysis tool for Python, we generated molecular fingerprints of length 1024 where each fragment corresponds to a bond length of 7, where each fingerprint is folded down into the desired bit length [9] . Each bit in the vector corresponds to the presence or absence of a fragment in the .mol file. This vector can be converted into a numpy array for analysis with sklearn. Some limitations of the fingerprinting method is that it does not account for stereochemistry, and does not handle metal ions very well. Furthermore, it is possible that the vectors produced by RDKit can be non-unique for a given molecule, depending upon the parameters put into the program. We used a density of 0, so it is unfolded. Molecular fingerprinting has been previously used to search large databases of molecules to determine similar molecules to known molecules of interest [8].

To generate these fingerprints, we first scraped KEGG to obtain .mol files for each metabolite they were available for. In turn these molecule files were used to generate molecular fingerprints with bit length 7 and density 0, which were then put into a spreadsheet and associated with a given metabolite. This spreadsheet was used for subsequent analysis.

Machine learning prediction of promiscuity from metabolite structures

After obtaining chemical fingerprints for different metabolites, those fingerprints were then loaded and added to an object of each affected metabolite with its fingerprint. Each of these objects were then converted into reaction objects that possessed these metabolites with their fingerprints. These reaction objects in turn have reactant and product fingerprints which are the linear combination of the vectorized fingerprints of their metabolites. Then, from these reactant and product vectors for each reaction, averaged and row corrected reactant and product vectors were made for each enzyme, which was an object containing the name and the reactions associated with the enzyme.

We trained our model to predict enzyme promiscuity based on the enzyme's row corrected, averaged reaction vector and the promiscuity value (1 for promiscuous, 0 for non-promiscuous). We used five fold cross validation to assess the accuracy of different methods of supervised learning on the data set and the results are shown in **Table 5**. Furthermore, PR and ROC curves for the kNN predictor with are shown in **Figure 11**.

Table 5: Performance of different supervised learning models at predicting enzyme promiscuity from molecular fingerprints of substrates.

Model	Accuracy ± Standard Deviation (5-fold CV)
SVM (linear)	0.661 ± 0.002
SVM (rbf)	0.661 ± 0.002
Naive Bayes	0.661 ± 0.002
Decision Tree	0.703 ± 0.033
k-NN (k=5 optimal)	0.727 ± 0.01

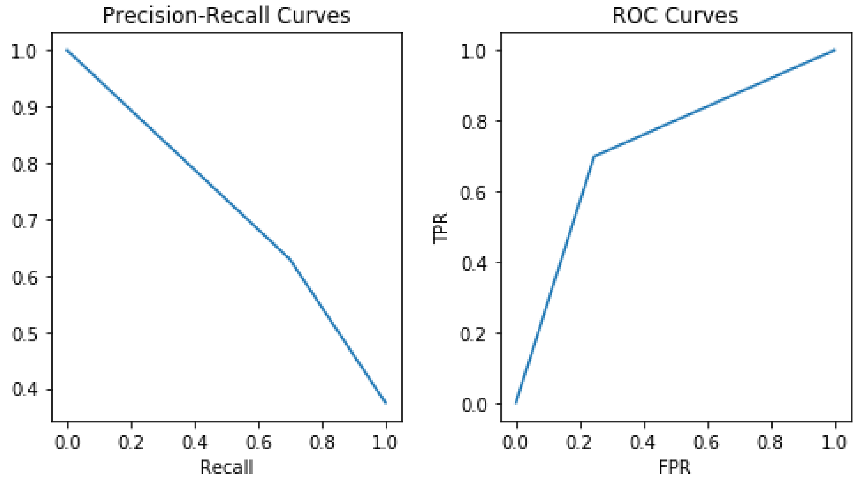

Figure 11: Precision-Recall and ROC curves for the k-NN model (k=5)

Predicting new reactions from similarities in substrate structures

Shifting our focus away from “general” enzyme promiscuity to specific reactions, we developed an algorithm to predict new reactions not yet annotated in iML1515 on the basis of similar substrates structures. We pursued an algorithmic approach or a statistical one, due to the inherent class imbalance issues present in predicting specific reactions (i.e. the number of enzymes accepting a substrate << number that do not). We considered an approach comprised of three steps: 1) for a given enzyme, cluster all known substrates based on structure, 2) mark all existing metabolites in iML1515 that fall within a cluster as potential substrates, and 3) predict corresponding products from the potential substrates.

Using the 1024-length binary vector fingerprints from above, pairwise substrate distances were computed as (1 - Jaccard Index) between each pair of fingerprints. However, hierarchical clustering with these distances revealed that it is difficult to systematically determine the optimal number of clusters for each enzyme’s substrates. To resolve this issue, we developed a heuristic to capture most of the area of each cluster. We first identified bidirectional nearest neighbors, i.e. a pair of substrates (s_1,s_2) such that s_1 is closest to s_2 among all other substrates, and vice versa. We then checked whether or not the pair distance was significantly small, i.e. if the pair distance is less than the 5th-percentile in the distribution of distances from s_1 vs. all other metabolites, and s_2 vs. all other metabolites. Finally, using these pairs, potential substrates were identified as any metabolite whose distance to either element in a pair is less than the pair distance. A visualization of these steps is shown in **Figure 12**:

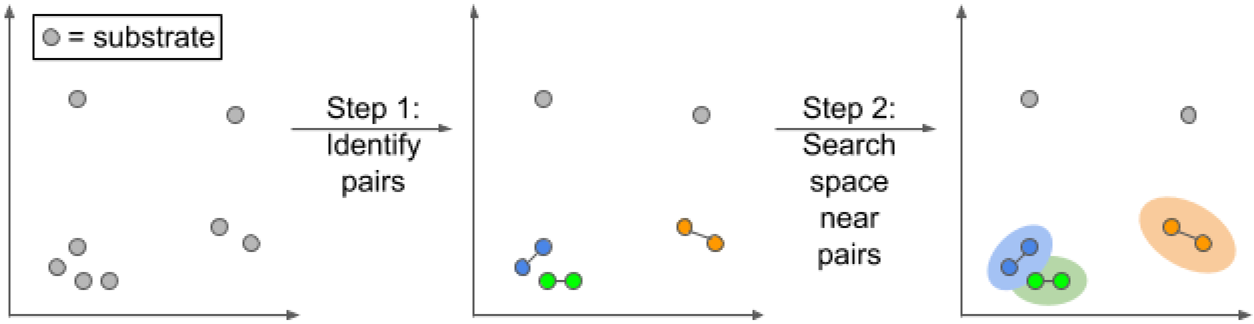

Figure 12: Identification of cluster areas using a bidirectional nearest neighbors heuristic. Substrates are represented in a simplified 2D metabolite structure space.

Finally, to identify products corresponding to potential substrates, we modeled reactions as sets of bitwise operations between fingerprints. For every substrate/product pair in an enzyme, a reaction bitmask is defined as the set of positions in the fingerprint that change from the substrate to the product. If the fingerprints for a candidate substrate and the current substrate are identical at those positions, the reaction is applied to the candidate by flipping the bits at those positions in its fingerprint to generate the fingerprint of the product. Furthermore, candidate substrates in which no reaction bitmask is applicable across all substrate/product pairs are filtered out.

Using this approach, the algorithm was able to propose over 900 new reactions for iML1515. However, as it is impossible to convert a fingerprint into a molecular structure, we examined only the proposed reactions in which the predicted product fingerprint was already known. This identified only one new reaction: an alternative nucleoside triphosphatase reaction was newly assigned the the gene b4161, which is already known to be a promiscuous nucleoside triphosphate. Specifically, b4161 is known to act on ATP, CTP, GDP, and ITP, and the algorithm assigns XTP as an alternative substrate: $XTP + H_2O \rightarrow XDP + H^+ + P_i$. This XTP reaction is already catalyzed by b4394, another promiscuous nucleoside triphosphatase, which suggests that there may be significant enzymatic promiscuity and/or redundancy in nucleotide salvage pathways.

Due to time constraints, additional benchmarking of this algorithm fell outside the scope of this project. Future work may involve validation with older models or models with reactions removed, to see if missing reactions can be predicted. Additionally, starting with a larger database of metabolite structures beyond those in iML1515 may provide more candidates substrates and allow for better resolution of corresponding products. Finally, as this approach is only able to predict new reactions for known promiscuous enzymes, additional development is necessary to generalize the criteria used for predicting new substrates for specific enzymes and discovering new forms of enzyme promiscuity.

Conclusions

We demonstrated here the use of four distinct data types to predict enzyme promiscuity (substrate overlap, protein domain annotations, protein sequences, substrate structures), using a diverse range of both unsupervised and supervised learning techniques (PCA, tSNE, hierarchical clustering, DBSCAN, SVM, Decision Tree, Naive Bayes, kNN, and LSTM). Ultimately, we found that none of these methods were able to predict enzyme promiscuity to an accuracy greater than 75% (best methods being protein-sequence LSTM and substrate -structure kNN), which is only somewhat better than random prediction. It may be reasonable to conclude that instances of enzyme promiscuity are unique phenomena, and that there do not exist any global trends that can be learned from the data types we examined; however the approaches used here may be applied to use these any of these data types to predict different definitions for enzyme promiscuity, or any other binary label that can be assigned to an enzyme.

Finally, we began development of a workflow for proposing new reactions in iML1515 on the basis of substrate structure clusters, and are currently able to identify over 900 possible new reactions (though only 1 such reaction had a readily identifiable product). Additional benchmarking and validation of this algorithm may provide a more promising alternative to predicting new reactions in *E. coli* than attempting to predict general enzyme promiscuity.

References

Code for all analyses is available here: https://github.com/jhyun95/ecoli_promiscuity

All analysis was implemented in Python3.6. Extraction and definitions of enzymes from iML1515 was implemented in part with COBRApy. Hierarchical clustering was implemented through SciPy. PCA, tSNE, and DBSCAN were implemented through scikit-learn (sklearn). LSTM learning was implemented through pyTorch.

[1] McCloskey D, Palsson BØ, Feist AM (2013) Basic and applied uses of genome-scale metabolic network reconstructions of Escherichia coli. *Mol Syst Biol*. 9:661. doi: 10.1038/msb.2013.18.

[2] Monk JM, Lloyd CJ, Brunk E, Mih N, Sastry A, King Z, Takeuchi R, Nomura W, Zhang Z, Mori H, Feist AM, Palsson BØ (2017). iML1515, a knowledgebase that computes Escherichia coli traits. *Nat Biotechnol*. 35(10):904-908. doi: 10.1038/nbt.3956.

[3] Khersonsky O, Tawfik DS (2010). Enzyme promiscuity: a mechanistic and evolutionary perspective. *Annu Rev Biochem.*79:471-505. doi: 10.1146/annurev-biochem-030409-143718.

[4] Notebaart RA, Szappanos B, Kintses B, Pál F, Györkei Á, Bogos B, Lázár V, Spohn R, Csörgő B, Wagner A, Ruppin E, Pál C, Papp B (2014). Network-level architecture and the evolutionary potential of underground metabolism. *Proc Natl Acad Sci USA*. 111(32):11762-7. doi: 10.1073/pnas.1406102111.

[5] D'Ari R, Casadesus J (1998). Underground metabolism. *BioEssays*. 20:181–186

[6] Nam H, Lewis NE, Lerman JA, Lee DH, Chang RL, Kim D, Palsson BØ (2013). Network context and selection in the evolution to enzyme specificity. *Science*. 337(6098): 1101–1104. doi: 10.1126/science.1216861.

[7] Guzmán GI, Utrilla J, Nurk S, Brunk E, Monk JM, Ebrahim A, Palsson BØ, Feist AM (2015). Model-driven discovery of underground metabolic functions in Escherichia coli. *Proc Natl Acad Sci USA*. 112(3):929-34. doi: 10.1073/pnas.1414218112.

[8] Willett, P. Similarity Searching Using 2D Structural Fingerprints. *Methods in Molecular Biology* 133-158 (2010). doi:10.1007/978-1-60761-839-3_5

[9] Getting Started with the RDKit in Python — The RDKit 2018.03.1 documentation. *Rdkit.org*(2018). at <http://www.rdkit.org/docs/GettingStartedInPython.html#fingerprinting-and-molecular-similarity>