Bioinformatics, 35(3), 2019, 442–451
doi: 10.1093/bioinformatics/bty672
Advance Access Publication Date: 1 August 2018
Original Paper

Sequence analysis

Effusion: prediction of protein function from sequence similarity networks

Jeffrey M. Yunes [1,2] **and Patricia C. Babbitt** [2,3,4,*]

[1]UC Berkeley - UCSF Graduate Program in Bioengineering, [2]Department of Bioengineering and Therapeutic Sciences, [3]Department of Pharmaceutical Chemistry, and [4]Quantitative Biosciences Institute, University of California, San Francisco, CA 94158, USA

*To whom correspondence should be addressed.

Associate Editor: John Hancock

Received on January 25, 2018; revised on July 24, 2018; editorial decision on July 25, 2018; accepted on July 30, 2018

Abstract

Motivation: Critical evaluation of methods for protein function prediction shows that data integration improves the performance of methods that predict protein function, but a basic BLAST-based method is still a top contender. We sought to engineer a method that modernizes the classical approach while avoiding pitfalls common to state-of-the-art methods.
Results: We present a method for predicting protein function, *Effusion*, which uses a sequence similarity network to add context for homology transfer, a probabilistic model to account for the uncertainty in labels and function propagation, and the structure of the Gene Ontology (GO) to best utilize sparse input labels and make consistent output predictions. Effusion's model makes it practical to integrate rare experimental data and abundant primary sequence and sequence similarity. We demonstrate Effusion's performance using a critical evaluation method and provide an in-depth analysis. We also dissect the design decisions we used to address challenges for predicting protein function. Finally, we propose directions in which the framework of the method can be modified for additional predictive power.
Availability and implementation: The source code for an implementation of Effusion is freely available at https://github.com/babbittlab/effusion.
Contact: babbitt@cgl.ucsf.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Determining the function of gene products is necessary for understanding life, valuable for its applications in health and industry, and a foundational problem in bioinformatics. The number of protein sequences, now greatly amplified by metagenomic sequencing projects, continues to grow at a much faster rate than the number of proteins with experimentally determined functions. As a result, computational prediction of protein function is needed more than ever (Friedberg, 2006).

A common approach for predicting the molecular function (MF) of a given query protein is to search a sequence database of annotated proteins and derive the predicted function from the annotations of the most similar sequences found (Conesa *et al.*, 2005; Martin *et al.*, 2004). Despite a number of caveats (Gilks *et al.*, 2005; Rost, 2002; Schnoes *et al.*, 2009; Tian and Skolnick, 2003; Todd *et al.*, 2001), this approach remains popular due to the wide availability of sequence data, the speed at which a similarity search can be conducted (Altschul, 1997; Buchfink *et al.*, 2015), and the simplicity and robustness of the method.

As of March 17, 2017, UniProtKB (The UniProt Consortium, 2017) includes 80 758 400 protein sequences. BLAST (Altschul, 1997) and DIAMOND (Buchfink *et al.*, 2015) can be used to quickly search a database of such sequences for proteins that are similar in sequence to a given query.

The Gene Ontology Consortium represents protein function using a controlled vocabulary of terms, related hierarchically, where more general activities are ancestors of more specific activities. (The Gene Ontology Consortium, 2017) The Gene Ontology (GO) models three distinct aspects of protein function: MF, biological process, and cellular component. We focus on *MF*, which is defined as "the biochemical activity (including specific binding to ligands or structures) of a gene product" (Ashburner *et al.*, 2000). There are 10 885 GO terms in the MF ontology. About 8799 of these are leaves, which have no child terms. A protein function can be represented by a combination of these GO terms.

The Gene Ontology Annotation database (GOA) (Huntley *et al.*, 2009) contains a list of associations between UniProtKB identifiers and GO terms. Each association is complemented with metadata, including the date the association was made and an evidence code indicating whether the annotation was assigned by a curator using either experimental or computational analysis, or assigned automatically. A positive annotation to a specific GO term implies a positive annotation to any of its more general ancestor GO terms. Only 562 971 protein sequences in UniProtKB have an experimentally determined annotation. As these annotations come from various labs and genome annotation consortia, neither the proteins nor the GO terms are studied uniformly.

An effective model for protein function prediction must take into account several idiosyncrasies of protein function and the data available to use for its prediction. There are a tremendous number of protein sequences available, but very few are functionally characterized. Experimental annotations, which usually describe a protein's function in part or at a high level, are expensive to obtain, rare, and collected with bias (Schnoes *et al.*, 2013). Negative annotations, which indicate that a given protein does not have a given activity, are nearly non-existent. The label space is large, and there is not a single protein that has a complete label in Gene Ontology Annotation (GOA), with a positive or negative annotation for every functional term in GO. Some GO terms also have few, or no, associated proteins, thwarting typical classification algorithms that require many samples per class.

Although these characteristics complicate function prediction, a method can be constructed to benefit from the constraints they impose. For example, sequence similarity networks (SSNs) use unannotated proteins to provide context for predicting MFs (Atkinson *et al.*, 2009; Davidson *et al.*, 2018; Li *et al.*, 2013; Martin *et al.*, 2013; Sharan *et al.*, 2007). Visually, they show putative clusters of conserved function, space between clusters with few proteins where there may be a change in function, and unexplored regions of the sequence space. A semi-supervised classifier uses networks to guide the drawing of functional boundaries between protein clusters.

As another example, a method that uses a separate classifier for each GO term will likely have too few training samples for each one, and the resulting predictions may be inconsistent with respect to the GO hierarchy. However, viewing the problem as a structured prediction problem will not only result in consistent output labels, but will be able to take advantage of annotations throughout GO (Barutcuoglu *et al.*, 2006; Eisner *et al.*, 2005; Jiang *et al.*, 2008; Obozinski *et al.*, 2008; Sokolov and Ben-Hur, 2011).

Evaluation of protein function is also complex. The sample labels used for training and for evaluation are derived from GOA, so they are also structured, incomplete, and collected unevenly. The use of basic evaluation methods and metrics is not always appropriate, and it was difficult to compare results from more involved evaluation protocols. To make progress on these issues, the protein function prediction community conducts an experiment every few years called the Critical Assessment of Function Annotation (CAFA) (Radivojac *et al.*, 2013; Jiang *et al.*, 2016). This community effort has identified methods, metrics, and baselines that better indicate the extent to which prediction methods can automate function annotation. In this work, we make use of these best practices.

Most protein network–based probabilistic graphical models (PGMs) focus on protein-protein interaction networks (Kourmpetis *et al.*, 2010), but some authors present or suggest a PGM based on a SSN (Deng *et al.*, 2004; Letovsky and Kasif, 2003). Carroll and Pavlovic (2006) and Mitrofanova *et al.* (2011) additionally incorporate the structure of GO into their PGM. We build on the ideas of these methods, but use a model, parameters, and algorithm that are better suited to the problem of predicting protein function.

Here, we propose and evaluate a new method, *Effusion*, that uses a network of partially characterized sequences to suggest accurate function predictions. The use of SSNs, the incorporation of GO, and the application of PGMs has been previously reported. However, our method, its model, and its parameters are the first to integrate these features in a way that is accurate, practical, and extensible. Specifically, our model, inspired by network analysis in computational biology (Atkinson *et al.*, 2009; Barber and Babbitt, 2012; Brown and Babbitt, 2012, 2014), admits a highly interpretable set of parameters, which we can learn for each GO term and from all experimental annotations, augment them with pseudocounts, and submit to general-purpose inference algorithms. Evaluation of the predictions shows that our method can accurately discern the MFs of a protein, even when faced with partial, autocorrelated samples and classes that are imbalanced and related hierarchically.

2 Materials and methods

A graphical summary of the method is shown in Figure 1.

First, we build a protein network with edges of sequence similarity. This network is quickly constructed by broadly BLASTing (Altschul, 1997) sequence queries to collect homologs (Fig. 1a) and using DIAMOND (Buchfink *et al.*, 2015) to fill in the all-by-all sequence similarity edges of the network (Fig. 1b).

Second, we construct a tractable PGM based on this network (Pearl, 1988). The network is first converted to a minimum spanning tree (MST), pruned to query proteins or proteins with non-electronic annotation, and directed outward from the query (Fig. 1c). For each edge in the MST, we link the corresponding MF terms from GO (Fig. 1d). Since the model is directed, the parameters are simply conditional probabilities that can be calculated by counting over all pairs of neighboring proteins that have experimental annotations.

Finally, we perform inference using general-purpose, state-of-the-art inference software and output the predictions. A network view of the output predictions for two GO terms is shown in Figure 1e. Since the method has a GO-structured model of protein function, detailed predictions are given for every protein in the network. We show the detailed predictions for the query in Figure 1f.

The details of the method follow.

2.1 Preprocessing

We downloaded the datasets for our analysis in early April, 2017.

- protein sequences: UniProtKB, compiled from SwissProt and TrEMBL, version 2017_03
- experimental annotations: Gene Ontology Annotation Database, version 2017-03-11
- controlled vocabulary: Gene Ontology, version 2017-03-31

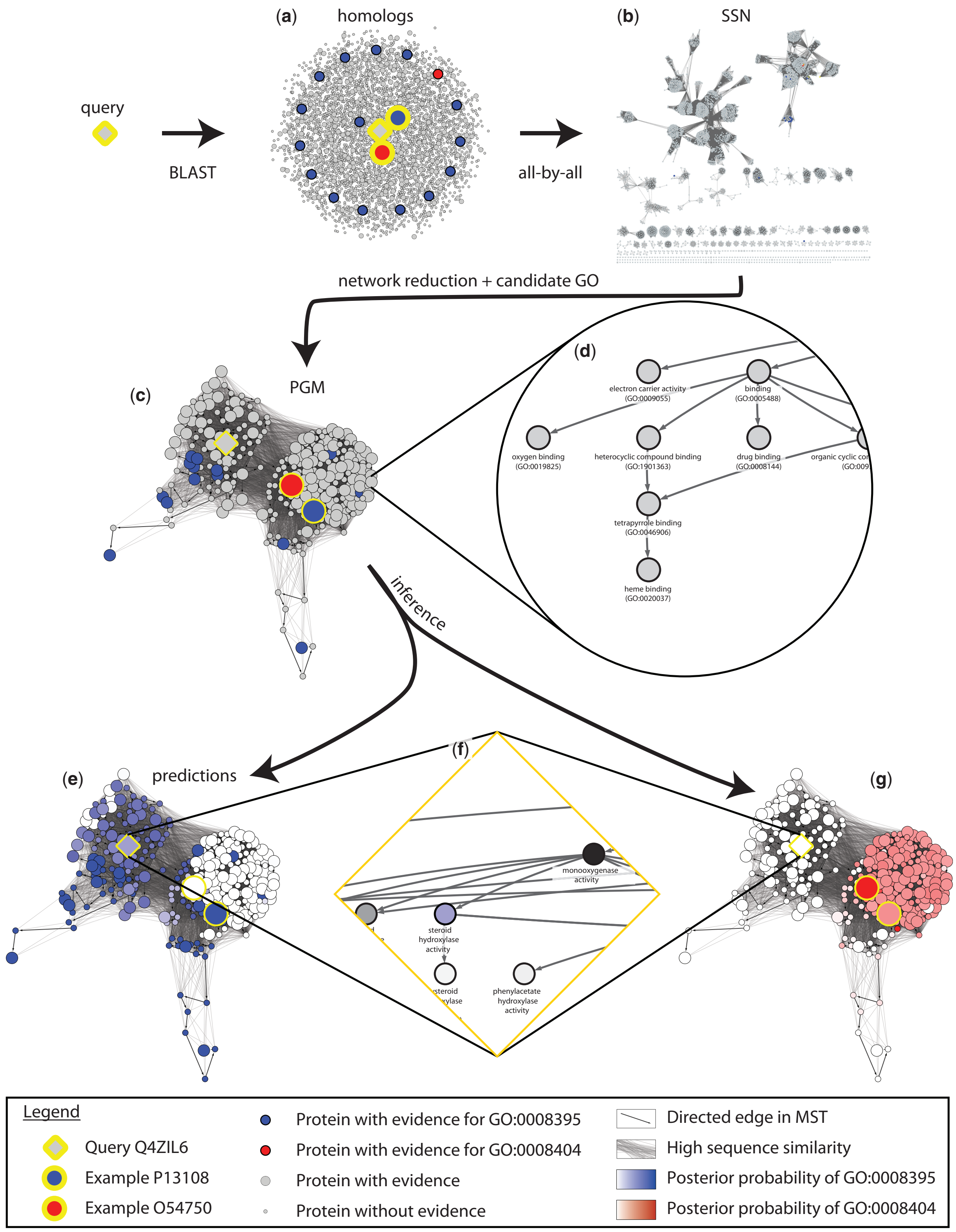
(a) homologs
(b) SSN
query
BLAST
all-by-all
network reduction + candidate GO
(c) PGM
(d)
electron carrier activity (GO:0009055)
binding (GO:0005488)
oxygen binding (GO:0019825)
heterocyclic compound binding (GO:1901363)
drug binding (GO:0008144)
organic cyclic co (GO:009
tetrapyrrole binding (GO:0046906)
heme binding (GO:0020037)
inference
(e) predictions
(f)
monooxygenase activity
steroid hydroxylase activity
phenylacetate hydroxylase activity
(g)
Legend
Query Q4ZIL6
Example P13108
Example O54750
Protein with evidence for GO:0008395
Protein with evidence for GO:0008404
Protein with evidence
Protein without evidence
Directed edge in MST
High sequence similarity
Posterior probability of GO:0008395
Posterior probability of GO:0008404

Fig. 1. Graphical Summary of Effusion, using cytochrome P450 CYP1C1 (UniProt Q4ZIL6) as an example. (**a**) Homologs of UniProt Q4ZIL6 collected by BLAST. UniProt Q4ZIL6 has been experimentally annotated to a descendent of steroid hydroxylase activity (GO:0008395), but this annotation was one of the ones withheld during the test phase, and it has no annotation to any arachidonic acid 14, 15-epoxygenase activity (GO:0008404), an annotation for a homolog of the query reported by BLAST. (**b**) An SSN is built from the all-by-all edges computed via DIAMOND. The network is visualized with Cytoscape (Shannon, 2003) using Organic Layout. (**c**) The reduced network. The layout is applied to all edges, but only the MST edges are used in the model. The resulting network is used as the topology of a PGM. (**d**) The protein function of each node is represented by a subset of GO, with each GO term represented by a Bernoulli random variable. (**e**) Two views of the network following inference. The left figure is shaded according to the probability of GO:0008395. The right figure is shaded according to the probability of GO:0008404. (**f**) Probabilities for a subset of terms for query UniProt Q4ZIL6. Probabilities are calculated for each candidate GO term for each node. Nodes are shaded from white being 0% to black being 100%, except for the node representing GO:0008395, which is colored a shade of blue based on its posterior probability

- sequence similarity: Computed by BLAST version 2.6.0+ for single query searches, or via DIAMOND version 0.9.10 for all-by-all calculations

We only included annotations to UniProtKB identifiers, excluding a smaller number of annotations linked to gene products in other databases. As is standard in the field, we excluded Inferred from Electronic Annotation (IEA) annotations (Gilks *et al.*, 2005; Jiang *et al.*, 2016). For each positive annotation, we explicitly added positive annotations for each ancestor GO term for the same protein and annotation metadata (i.e. date and evidence code). Similarly, a negative annotation for a protein to a GO term was considered a negative annotation for the same protein to every descendent GO term. We refer to the resulting set of annotations as *preprocessed* annotations.

The dataset for the discovery phase included all preprocessed annotations through 2015. Annotations from January 1, 2016 onward were withheld for all purposes until final evaluation and analysis.

2.2 Building the protein network

For a given query protein, we build a SSN, where nodes represent proteins and edges represent pairwise sequence similarity. This network is quickly constructed by collecting homologs with BLAST, and then using DIAMOND to calculate the all-by-all sequence similarity scores to fill in the edges of the network. BLAST is run with a permissive E-value threshold (1×10^{-8}), but limited to sequences that cover 90% of the query, and filtered to sequences with a bit score per residue of at least 0.25. The requirement for 90% query cover is intended to avoid transferring functions for a domain in the subject that does not exist in the query. We use the resulting sequences to format a DIAMOND DB, and search each resulting sequence against this DB using DIAMOND (E-value $= 1 \times 10^{-8}$, query cover $= 90\%$, subject cover $= 90\%$, max target seqs $= 1000$, minimum bit score per residue of 1.4×0.25). The all-by-all calculation uses more restrictive parameters to limit the number of edges for reasons of computational practicability. This heuristic usually provides the MST edges we need for building the model without wasting space and time (see Section 3.2).

2.3 Constructing a tractable probabilistic graphical model

The protein network is first reduced to make it more amenable to learning and inference. To get the reduced network, we convert the protein network to a MST (edge weight $= -\text{bit score per residue}$, negated for maximum spanning tree of highest scoring edges), direct it outward from the query, and prune it to query proteins or proteins with non-electronic annotation (evidence code not IEA). The goal of this network reduction was to imbue local factors with a global probabilistic interpretation, which facilitates parameter learning. Another benefit of this reduction is that it results in inference running more quickly, by removing tight loops in SSNs, and allowing pruning. However, we note that the PGM is still not a tree because variables generally still have multiple parents, coming from GO and from the corresponding term of the parent protein in the reduced network. Therefore, iterative, approximation algorithms are needed for inference.

2.3.1 Protein template

Each protein in the reduced network generates a subgraph in the PGM, by instantiating a copy of the *protein template*. The protein template models the MF of a protein. It has the topology of a subgraph of the Gene Ontology Consortium's MF ontology. Each node in the subgraph is a *candidate GO term*. These are chosen by collecting every GO term for which there is a positive or negative preprocessed annotation in the SSN. Each candidate GO term, for each protein, is modeled as a Bernoulli random variable, and we eventually calculate marginal posterior probabilities for each of them (Section 2.6).

We implemented and evaluated two models for relating the GO terms within a protein template: a top-down model and a bottom-up model. Each model has its own advantages and disadvantages.

In a top-down model, the parent(s) of a variable representing a GO term t for protein i include the parents (more general terms) in GO, so the factors $\psi(x_t^i, \text{pa}(x_t^i))$ are $P(x_t^i|\text{pa}(x_t^i)) = P(x_t^i|x^i_{\text{GO parents}(t),\ \text{other model parents}(x_t^i)})$. It is more common to model a PGM in a top-down fashion, because PGMs that limit factor sizes, in particular those that have only a single parent per node, admit more tractable PGM inference.

However, a top-down model has limitations for modeling protein function. If, for example, a protein is annotated as a DNA polymerase, then that protein has an implied annotation to polymerase in general, and that will give the protein a high probability of any type of polymerase, such as an RNA polymerase. In this scenario, the posterior probability for RNA polymerase could be higher than the posterior probability for any specific DNA polymerase.

To address this, we added supplementary negative evidence as follows. For each protein with evidence, if the weighted contingency table (Section 2.5) shows that a particular unobserved term is unlikely ($< 50\%$) given the observed values of its sibling terms, then we infer a negative annotation for that sibling. For the example above, since an annotation for RNA polymerase is unlikely to co-occur with an annotation for DNA polymerase, then a protein with a positive annotation for DNA polymerase and no annotation for RNA polymerase would have a supplementary negative annotation for RNA polymerase.

As these supplementary negative annotations are considered as evidence, they are deemed certain. Since they are not always correct, the necessity of negative annotations is a limitation of the top-down model. For example, a query which is known to have transferase activity (GO:0016740) in its training data will be given no chance of having hydrolase activity (GO:0016787), because the two terms are siblings, and $P(\text{hydrolase}|\text{transferase}) = 25.3\% < 50\%$. However, our data show that the terms were co-annotated to the same protein 834 times.

In our example above, we would prefer that our model were powerful enough to allow evidence for DNA polymerase to *explain-away* our belief in RNA polymerase. Therefore, we also experimented with a bottom-up model, where the parents of a variable representing GO term t for protein i include the children (more specific terms) in GO: $\psi(x_t^i, \text{pa}(x_t^i)) := P(x_t^i|\text{pa}(x_t^i)) = P(x_t^i|x^i_{\text{GO children}(t),\text{other model parents}(x_t^i)})$.

We use *GO kin* to refer to GO parents in the top-down model, and GO children in the bottom-up model. As we continue the description of our method, *model* refers generally to both the top-down and bottom-up model, except as specified.

2.3.2 Incorporating sequence similarity edges

When two proteins are similar in sequence and have an MST edge between them, we connect the corresponding MF terms. Assuming reasonable parameters, the factor associated with this edge induces two proteins that are similar in sequence to have similar MFs.

2.4 Information content (IC) of GO terms

The IC of a GO term g is calculated by

$$\mathrm{IC}(g) = -\log_2(P(g|\mathrm{GO\,parents}(g)))$$

2.5 Parameter learning

Since the model is directed, the parameters have a global probabilistic interpretation: each variable in the probabilistic model adds a factor representing the conditional probability of that variable given its parents.

For the model just presented, the parameters are $P(x_t^i | x^i_{\mathrm{GO\,kin}(t)}, x_t^{\mathrm{BLAST\,parent}(i)})$, the probability of protein i having term t, given the GO kin of GO term t for protein i, and the corresponding GO term for the BLAST parent.

We can learn these parameters from all available experimental data, not just the data in a network for a specific query. To do so, we compare the label for each protein with a preprocessed annotation to the label of the most similar protein with a preprocessed annotation. The similarity of the most similar protein must be below the similarity thresholds specified above for building the network. We count the number of times there was a gain of function, loss of function, or other such events. We use thiolester hydrolase activity (GO:0016790) as an example to show the contingency tables we use for calculating the parameters. Table 1 shows the contingency table of raw counts.

We note also that we learn parameters for each GO term. So, for example, the probability of a hydrolase losing its ability to hydrolyze ester bonds GO:0016788 over adjacent proteins is low (1.5%); the probability of a hydrolase losing its ability to hydrolyze a thiolester bond GO:0016790 is higher (4.3%).

Our contingency tables are based on preprocessed annotations rather than inferred functions, so incomplete annotations result in a model that make a gain or loss of function between neighboring proteins very likely. We address this by weighting the count contributed by each protein by the IC of the protein's label $\mathrm{IC}(\mathrm{label}) = \sum_{g \in \mathrm{label}} \mathrm{IC}(g)$. An example contingency table of weighted counts is shown in Supplementary Table S1.

In order for our model to have a chance at predicting rare GO terms, we added pseudocounts to our contingency tables. Our aim was to add counts from contingency tables for similar terms, but with more experimental evidence. Therefore, we transformed each raw (or IC weighted) contingency table for GO term g, $C_g^{\mathrm{raw\ /\ weighted}}$, with the recursion

$$\begin{aligned} C_g^{\mathrm{pseudo}} &:= 0.10 \times C_{\mathrm{MRCA(GO\,parents(g))}} \\ C_g &= C_g^{\mathrm{raw\ /\ weighted}} + C_g^{\mathrm{pseudo}} \end{aligned}$$

where MRCA($\cdot$) is the most recent common ancestor. A contingency table with pseudocounts added to the raw counts are shown in Supplementary Table S2.

Table 1. Raw contingency table for GO:0016790

Protein's annotation to GO:0016788 (GO parent)	BLAST neighbor's annotation to GO:0016790	Count protein annotation to GO:0016790 is negative or unknown	Count protein is positively annotated to GO:0016790
–/?	–/?	41446	0
–/?	+	5	0
+	–/?	1453	5
+	+	3	19

2.6 Inference

Our models were complex enough that it was intractable to use exact inference algorithms or standard approximate algorithms. Fortunately, software implementations of algorithms that won the recent Uncertainty in Artificial Intelligence (UAI) inference competition (Dechter,R. personal communication), namely variations on adaptive inference and SampleSearch (Acar *et al.*, 2012; Gogate and Dechter, 2011), gave good results on our model. By default, we used adaptive inference with conditioning (ai_cond) when evaluating our test predictions.

Runtime and required memory depends on the number of proteins in the pruned SSN, the number and topology of the candidate GO terms, and the parameters given to the inference engine. Since these numbers varied greatly per query, we selected an algorithm that uses the maximum amount of time and memory given to it. Specifically, we set a per query limit of 40 minutes of CPU time and 8 GB memory.

2.7 Post-processing

We applied the following post-processing uniformly to the raw predictions of all the methods that we evaluated.

We evaluate against some methods that do not use the structure of GO, and these methods may be disadvantaged as a result of predicting a very general term with the same probability as a specific term. We break ties in favor of more general terms by applying the following transformation:

$$\begin{aligned} P^{\mathrm{new}} &:= P^{\mathrm{raw}} \times P(\mathrm{Depth} = \mathrm{GO\,term\,depth}) \\ P(\mathrm{Depth} = \mathrm{GO\,term\,depth}) &:= 1 - \epsilon \times \mathrm{GO\,term\,depth} \end{aligned}$$

with $\epsilon := 0.0001$.

Some methods used for evaluation predict the same GO term more than once for a single protein. We resolve these by keeping only the prediction with the highest probability.

2.8 Evaluation

We performed evaluation via temporal holdout (Greene and Troyanskaya, 2012). The testing phase used annotations through 2015 for training, and withheld annotations from the start of 2016. This reflects the methodology of the CAFA (Clark and Radivojac, 2013; Jiang *et al.*, 2016), is reflective of the true task of automating the manual process of characterization of protein function, and is widely recommended (Greene and Troyanskaya, 2012).

All 2757 proteins with a new annotation to a GO term in the MF ontology inferred by direct assay (evidence code IDA) were used as the evaluation set. We evaluated all proteins and all terms with this criterion. We did not limit our evaluation to proteins that had no annotations in the training set. We included all GO terms in the MF ontology, and we did not exclude GO terms that are rarely observed, nor did we exclude proteins annotated only to rare GO terms.

We used performance metrics that are revealing and critical, proposed or suggested by (Clark and Radivojac, 2013), with minor modifications. *Weighted true positive (WTP)* and *weighted false positive (WFP)* are similar to the true positive count and the false positive count, respectively, but weight the counts by the IC of the GO terms to account for the imbalanced, hierarchically structured label space. Dividing WTP by the IC of the predicted terms gives *weighted precision (WPr)*. Similarly, dividing WTP by the IC of the terms in the standard gives *weighted recall (WRc)*. In an attempt to upweight high quality samples and down-weight low quality samples, *sample-weighted weighted precision (SW-WPr)* and *sample-weighted weighted recall (SW-WRc)* additionally use a weighted average over the evaluation proteins, where the weight is the IC of the true annotation. Neither recall nor its weighted variants are expected to go to 100%, since, for some evaluation proteins, withheld GO terms may not exist in any of the preprocessed training annotations. N_e represents the number of proteins being evaluated.

$$\mathrm{wtp}(\tau) = \frac{1}{N_e}\sum_{i=1}^{N_e}\sum_{t\in P_i(\tau)\cap T_i}\mathrm{IC}(t)$$

$$\mathrm{wfp}(\tau) = \frac{1}{N_e}\sum_{i=1}^{N_e}\sum_{t\in P_i(\tau)\setminus T_i}\mathrm{IC}(t)$$

$$\mathrm{wpr}(\tau) = \frac{1}{N_e}\sum_{i=1}^{N_e}\frac{\sum_{t\in P_i(\tau)\cap T_i}\mathrm{IC}(t)}{\sum_{t\in P_i(\tau)}\mathrm{IC}(t)}$$

$$\mathrm{wrc}(\tau) = \frac{1}{N_e}\sum_{i=1}^{N_e}\frac{\sum_{t\in P_i(\tau)\cap T_i}\mathrm{IC}(t)}{\sum_{t\in T_i}\mathrm{IC}(t)}$$

$$\text{sw-wpr}(\tau) = \sum_{i=1}^{N_e}\frac{\mathrm{IC}(T_i)}{\sum_{j=1}^{N_e}\mathrm{IC}(T_j)}\frac{\sum_{t\in P_i(\tau)\cap T_i}\mathrm{IC}(t)}{\sum_{t\in P_i(\tau)}\mathrm{IC}(t)}$$

$$\text{sw-wrc}(\tau) = \sum_{i=1}^{N_e}\frac{\mathrm{IC}(T_i)}{\sum_{j=1}^{N_e}\mathrm{IC}(T_j)}\frac{\sum_{t\in P_i(\tau)\cap T_i}\mathrm{IC}(t)}{\sum_{t\in T_i}\mathrm{IC}(t)}$$

We also report $F_{\beta=0.5} = (1+\beta^2)\cdot\frac{\text{precision}\cdot\text{recall}}{\beta^2\cdot\text{precision}+\text{recall}}$, the harmonic mean between precision and recall, where precision is twice as important as recall.

We compare our method to several others, including a standard method, intuitive baselines, and state-of-the-art methods. Details on their implementations and parameters can be found in Supplementary Text S1.

We implemented a sequence-similarity-based method, referred to in this paper simply as BLAST. BLAST is a high-performing method and is used in CAFA evaluations and by many annotation pipelines (Hamp *et al.*, 2013). Since both Effusion and BLAST use the same data, the results are highly interpretable.

We plot a baseline to convey relative scale. The *Random BLAST* method is implemented the same as the BLAST method, using the same parameters and thresholds, except that the preprocessed annotations are transferred with a probability equal to a number chosen uniformly at random. Note that this does not merely assign random probabilities to all candidate GO terms— probabilities will remain consistent with GO by construction.

A naïve method ignores the particular query protein, and predicts the same GO terms for every query, based on the background frequencies of the GO term in the dataset. Whereas in a balanced binary classification setting it would be easy to intuit the value of a performance metric or shape of a performance chart for a naive method, this is more difficult in the case of a multi-classification problem with unbalanced classes. In this work, we show the results for a modified naive baseline, *Naive+*, which is more useful when we are evaluating proteins that were partially annotated in the training data.

We also compared Effusion to two methods based on the most similar method described in the literature (Carroll and Pavlovic, 2006). *Carroll2006* is a close implementation of the published method, but shares Effusion's process for collecting homologs, and determining candidate functions. *Carroll2006+* is a close implementation of Effusion, in that it uses our reduced network and other modifications, but it uses the parameters described of the original method, namely, the normalized BLAST scores.

We also compare Effusion to SIFTER (Engelhardt *et al.*, 2011; Sahraeian *et al.*, 2015), for which there is full pipeline available, was shown to be a top performer in CAFA (Jiang *et al.*, 2016; Radivojac *et al.*, 2013), and although its goal is automated phylogenetic analysis, it is based on sequence data like Effusion. Although SIFTER used the same evaluation datasets as the other methods, it uses a different method for gathering homologs.

3 Results

Effusion is a simple sequence-similarity only method that utilizes a probabilistic model to account for the uncertainty in labels and function propagation, unlabeled protein data to add context for homology transfer, and the structure of GO to best utilize sparse input labels and make consistent output predictions. It uses an MST reduction of the network so that parameters can be calculated from all experimental data, weighted by the quality of the annotations, augmented with annotations and pseudocounts derived from the data, and used as input to general purpose PGM inference algorithms.

We provide an implementation of Effusion. The source code, written in Python, is available online. We rely on software written by others in C++ for the parts of Effusion that are computationally intensive, namely the database software, similarity computations, and inference engines.

We first analyze the predictions made by the method as a whole, and then show the effects of the various components of the method.

3.1 Comparative analysis

The dataset for the test phase contained 2757 proteins that had an Inferred from Direct Assay (IDA) annotation to the molecular function ontology dated in 2016. All of these were included for evaluation, except where indicated otherwise.

The BLAST-based method made non-root MF predictions for 75.2% (2072/2757) of the evaluation proteins. The remaining proteins did not have a protein with positive preprocessed annotations within the thresholds. Effusion (top down, adaptive inference with conditioning) made non-root MF predictions for 72.1% (1989/2757) of the total, and the bottom-up model made non-root MF predictions for 71.4% (1969/2757) of the total. Carroll2006 made non-root MF predictions for 40.6% (1119/2757) of the queries. Carroll2006+, which has Effusion's optimizations, made non-root MF predictions for 68.8% (1898/2757) of the queries. SIFTER, which does not use the same restrictive thresholds as the above methods, made non-root MF predictions for 76.3% (2103/2757) of the queries.

Compared to BLAST, Effusion has additional steps that can fail (e.g. inference) or result in a loss of proteins with evidence (i.e. the MST heuristic). Therefore, we first looked at the performance of Effusion (top-down and bottom-up, ai_cond) and the baselines over all evaluation proteins, including those proteins for which the methods failed to make a prediction (Fig. 2; Supplementary Fig. S1). As a more complex method, Effusion is disadvantaged for this mode of evaluation, and this analysis gives a lower bound on Effusion's real-world performance. According to all metrics, Effusion generally performs better than all the other methods.

The proteins where Effusion failed to make a prediction, but BLAST was able to make a prediction, were usually due to the uniform constraints on time and memory applied to all evaluation proteins (data not shown). In many cases, it would be possible for a user with a special interest in a specific protein to get a prediction by providing more resources. Therefore, we calculated an upper bound on performance by evaluating the methods on the subset of evaluation proteins for which every method being considered was

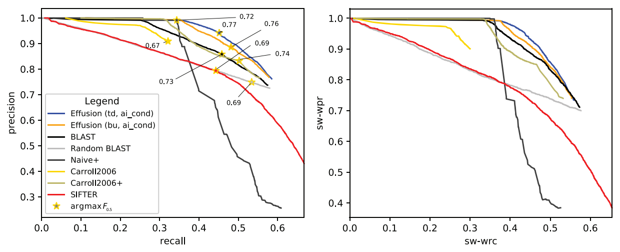

Fig. 2. Performance plots over all proteins in the test set, regardless of whether any of the methods failed to make predictions. (Left) Precision vs. Recall (Right) Sample-Weighted Weighted Precision vs. Sample-Weighted Weighted Recall

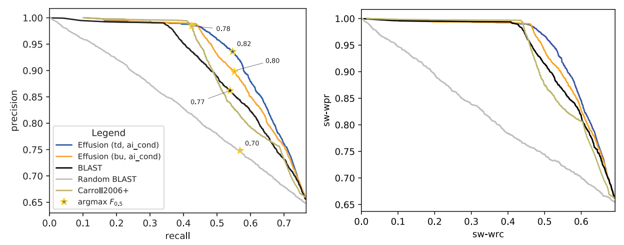

Fig. 3. Performance plots over treated proteins. Carroll2006 was not plotted because of its low coverage. SIFTER was not plotted due to its effect on the scale. (Left) Precision vs. Recall (Right) Sample-Weighted Weighted Precision vs. Sample-Weighted Weighted Recall

successful in making a prediction. We call these proteins *treated* proteins. The plots are similar to those of Figure 2 and Supplementary Figure S1, and are shown in Figure 3 and Supplementary Figure S2. As expected from the evaluation under the full set of evaluation proteins, Effusion generally performed better than all the other methods.

We performed a similar analysis on a per protein basis, essentially comparing 1942 classifiers of Effusion (top-down, ai_cond) and by BLAST. It is expected that Effusion and BLAST will predict the same GO terms in the same order for many proteins; the methods use the same data, and 538/1942 of the queries were associated with a protein template with ≤ 1 candidate leaf terms, or a network with ≤ 1 proteins with evidence. However, we identified 782 proteins where, for the GO terms predicted by both methods, Effusion reordered the predictions made by BLAST and resulted in a change in performance, according to area under the *WFP* versus *WTP* curve (y versus x). In general, Effusion accumulated the same bits of true positives, with fewer bits of false positives, for 53% (418/782) of the queries ($P = 0.03$, binomial test with $H_0 = 0.5$). The percentage increased to 61.9% (313/505) when we limited the analysis to those queries where the query itself did not have evidence.

We were especially interested about the ability of our method to differentiate catalytic activities. This is an important and difficult problem (Almonacid and Babbitt, 2011); in functionally diverse enzyme superfamilies, homologous members have evolved to catalyze many different chemical reactions (Gerlt and Babbitt, 2001), and these proteins are often misannotated in public databases (Schnoes *et al.*, 2009). We performed an additional evaluation constrained to the subset of GO representing GO terms that are descendants of catalytic activity (GO:0003824). 1160 of the evaluation proteins had an annotation to a GO term in this catalytic subset. Supplementary Figure S3 show plots the performance of our methods on the catalytic subset. The performance of Effusion (top-down, ai_cond) and BLAST differed on 142 enzymes, according to area under the *WFP* versus *WTP* curve. Effusion outperformed BLAST on 64.1% (91/142) of the queries ($P = 0.0005$).

Figure 1 illustrates Effusion's utility with a protein from the test dataset, identified by cytochrome P450 CYP1C1 (UniProt Q4ZIL6) in Zebrafish. One of the experimental annotations withheld from the test dataset was for testosterone 6-beta-hydroxylase activity (GO:0050649), which is a type of steroid hydroxylase activity (GO:0008395). This protein is not experimentally annotated to arachidonic acid 14, 15-epoxygenase activity (GO:0008404).

Effusion was able to combine evidence for GO:0008395 from the surrounding network context of UniProt Q4ZIL6 (see Fig. 1c). It correctly predicted GO:0008395 at 36.3% (Fig. 1e), at which point Effusion predicted no false positives. Effusion also predicted GO:0008404 lower at <1% (Fig. 1f). Supplementary Table S3 details the predictions for UniProt Q4ZIL6 made with Effusion.

BLAST, however, predicted this protein to have GO:0008404 with a probability of 20.0%, based on its proximity to O54750 (E-value $= 1.13045 \times 10^{-86}$, total bit score 278.87, bit score per residue $= 0.53$, alignment length $= 488$, query length $= 523$, subject length $= 488$, identities $= 161$, positives $= 259$). Compared to Effusion, BLAST accumulated 19 false positives (82.38 bits of information), before predicting GO:0008395 at 17%, based on hitting P13108 (E-value $= 2.62 \times 10^{-78}$, total bit score 256.914, bit score per residue $= 0.49$, alignment length $= 502$, subject length $= 502$, identities $= 167$, positives $= 256$). Predictions for UniProt Q4ZIL6 made with BLAST are shown in Supplementary Table S4.

3.2 Creation and use of protein networks

Effusion networks were generated quickly using BLAST to collect the homologs and DIAMOND to fill in the all-by-all edges that could be used in the MST (Supplementary Table S5, Supplementary Fig. S4). Effusion succeeded in making networks for 98.3% (2710/2757) of the queries. The median number of nodes in the protein network was 268.5, the maximum was 71 286, and 31 networks had only the query. The distributions for the number of nodes in the protein network are shown in Supplementary Figure S5.

After reducing the network to the MST and pruning the protein network to proteins that either had evidence or were queries, 899 networks had only the query remaining. This was due either to an absence of proteins with evidence in the original network, or to the more stringent threshold used for collecting the MST edges. The mean number of nodes in the reduced network was 57, if we exclude networks that contained only the query, with a maximum network size of 3284 nodes. 1987 networks had at least one protein with non-root positive preprocessed evidence. The distribution of the number of proteins in the reduced network are shown in Supplementary Figure S6, and the distribution of the number of proteins with positive, non-root evidence is shown in Supplementary Figure S7.

We compared Effusion to several baselines that do not use a semi-supervised approach to measure the value of adding the network context, shown in Supplementary Figure S8. The BLAST-like method used Effusion's probabilistic framework, but only included network edges from the query to proteins with evidence. The supervised method also used Effusions's probabilistic framework, but completely unlabeled proteins were deleted from the protein network and excluded from the subsequent model. The full network version of Effusion, which used the most unlabeled proteins, performed the best, and the BLAST-like version of Effusion, which used the fewest unlabeled proteins, performed the worst.

3.3 Candidate GO ontology effectively embedded within each protein of the model

The candidate GO terms overlapped significantly with the terms for the query that were in the standard (Supplementary Text S2), but they did not overlap exactly. Methods that predicted all of the candidate terms would have incurred, on average, 8.25 true positives (13.08 bits of information), but also have 7.80 false positives (20.34 bits of information) and 4.79 false negatives (9.86 bits of information).

The speed at which inference can be run is dominated by the number of parents of a variable in the graphical model, which are GO parents in the top-down model, or modeled GO children in the bottom-up model. The distributions for the maximum of these per query are shown in Supplementary Figure S9.

We compared the standard top-down method with supplementary negative evidence, the standard bottom-up, which does not require negative evidence, and the top-down method without the supplementary evidence. The performance curves are in Supplementary Figure S10. While the improvement seen here is modest, it was crucial when experimenting with larger networks that included non-homologous proteins (e.g. those found by searching protein interaction databases, data not shown).

Q921C5 provides an example that exemplifies the difference between the models (Supplementary Fig. S11). Without supplementary negative evidence, the top-down model incorrectly predicted Q921C5 as having clathrin binding (GO:0030276) with a probability of 26.4% that ranked it above a few correct GO terms (Supplementary Table S6). This is because the query Q921C5 had evidence for Rab GTPase binding (GO:0017137) and therefore implied evidence for ancestor term protein binding (GO:0005515), and $P(\text{GO:0030276}|\text{GO:0005515})$ is relatively high (11.0%). However, we also have implied evidence for enzyme binding (GO:0019899), and since GO:0030276 is unlikely (35.7%) to co-occur with GO:0019899, we assume this protein does not have GO:0030276. The full table of predictions is shown in Supplementary Table S7. Notice that there were some correct GO terms, such as macromolecular complex binding (GO:0044877), that were also predicted at 0% due to incorrect supplementary negative evidence for the query. On the other hand, because the bottom-up model has factors over all the child GO terms, it does not require negative evidence. The bottom-up model predicted GO:0030276 at only 19.6%. A table of predictions for Q921C5 using the bottom-up model is shown in Supplementary Table S8.

3.4 Data-derived alterations of parameters

The added value of weighting counts by information content of the sample is shown in Supplementary Figure S12. An example illustrating the effect of weighting is serine–tRNA ligase (UniProt P34945), shown in Supplementary Figure S13. Without weighting, the probability of a protein being involved in binding (GO:0005488) given its protein network parent has GO:0005488 is 90.7%. In the network, the probability of correct function decreases quickly, and the query is predicted at 32.6%. After weighting, $P(x_i^{\text{GO:0005488}}|x_{\text{pa}(i)}^{\text{GO:0005488}})$ increased to 95.2% and the probability for the query having the term increased to 59.3%.

The value of adding pseudocounts to the contingency table is shown in Supplementary Figure S14. An example from the top-down model is serine protease 57 (UniProt Q6UWY2), which had withheld experimental annotation of sulfur compound binding (GO:1901681), is shown in Supplementary Figure S15. This GO term is rarely observed experimentally, so the calculated statistic for $P(x_i^{\text{GO:1901681}}|x_{\text{pa}(i)}^{\text{GO:1901681}})$ was only 73.8%, the probability for this term decayed quickly from the protein with evidence to the query protein, and the query protein was predicted to have GO:1901681 at only 20.6%. After the addition of pseudocounts by the method described above, the calculated statistic for $P(x_i^{\text{GO:1901681}}|x_{\text{pa}(i)}^{\text{GO:1901681}})$ increased to 92%, and the prediction for the query increased to 56.2%. Pseudocounts were especially beneficial for the bottom-up model, due to its sensitivity to probabilities at the leaves.

3.5 Inference on real-world protein data

Although inference using ai_cond (Acar *et al.*, 2012) succeeded in making predictions for most 97.7% (1942/1987) proteins with evidence in the reduced network, the software still gave poor results on some queries; for example, for eight queries, the software predicted low probabilities for the root MF term.

Overall, the performance plots in Supplementary Figure S16 show two clusters of methods with similar performance. These clusters correspond to the top-down model and the bottom-up model.

4 Discussion and conclusions

By engineering a method and model that accounts for the essential idiosyncrasies of protein function prediction, we were able to make predictions that are practical and more accurate than state-of-the-art methods, using the same primary source of data.

We implemented and evaluated two constructions for the protein template that generated models with different semantics. Our analysis revealed interesting tradeoffs between the two models. Although the bottom-up model has the advantage that it does not require negative evidence, it has other limitations that offset its value. Namely, it is sensitive to leaf probabilities, and since each factor typically depends on more variables than it does in the top-down model, it is also less amenable to inference.

We also evaluated several inference algorithms for use with our method. Although we could not get reasonable predictions for all queries, even with heuristics, we had more success with a top-performing inference software identified by the UAI competition. We encourage further development of inference algorithms and software, and encourage participation in critical assessments on real world problems.

Effusion's main similarities to the method described in Carroll and Pavlovic (2006) and Mitrofanova *et al.* (2011) are the general use of protein networks, modeling of GO, and a probability model. However, there are fundamental differences with these works in each aspect of our methods. The previous methods used a model whose factors lack a global probabilistic interpretation, and since learning maximum likelihood parameters on a per network basis would have been infeasible, they used normalized similarity scores. Effusion, on the other hand, has a model whose factors are simply conditional probability functions, and therefore we can learn maximum likelihood parameters from all available experimental data. Significantly, this allows us to incorporate parameters that are specific to each GO term, thereby giving us reasonable results on problems that have a wide range of GO terms, rather than limiting predictions to only a few (i.e. < 10) GO terms. Additionally, our formulation allows the incorporation of data derived pseudocounts, inference with general purpose, rather than ad-hoc, inference algorithms, and ranked probabilities for all GO terms, rather than setting an arbitrary threshold for prediction. In contrast to Carroll, we have shown that Effusion's innovations enable accurate and practical prediction of protein function.

While it is common for labs to provide the algorithm for predicting protein function, they rarely provide the pipeline necessary for preparing the data and parameters, making it difficult to compare methods on the same training and test data. Fortunately, the high-throughput pipeline for SIFTER was available. This method is highly relevant because it is based on sequence similarity, its aim is to discern functions among relatively close homologs rather than predicting general functions from remote homologs, and it also uses a Bayesian network. There are substantial differences, however. Whereas Effusion and the other methods used BLAST to collect close homologs, SIFTER searches Pfam and builds a phylogenetic tree for each cluster found (Punta *et al.*, 2012). SIFTER uses a continuous time Markov chain in its model for protein function evolution, but does not model GO. Whereas Effusion currently requires edges to reflect alignment across the entire sequence, SIFTER combines results from all the Pfam domains in a query protein. While we have shown that Effusion outperforms SIFTER, our analysis indicates this is largely due to SIFTER's underperformance on query proteins with evidence in the training data (Supplementary Fig. S17). Although many former assessments exclude proteins with any functional annotation in the training data from evaluation, we note that this is an artificial restriction, since it is common practice to study partially annotated proteins. Indeed, this restriction has been relaxed in the most recent CAFA.

In this report, we describe the first version of Effusion, a simple, high performing method that suggests specific protein function with a model that uses protein networks and incorporates GO. Although Effusion only uses sequences that are highly similar across their entire lengths, and only models the MF aspect of GO, it is designed to be extendable to model additional aspects of protein function, and additional, more finely grained relationships between proteins and GO terms. For example, by making use of resources such as domain-centric GO (dcGO) (Fang and Gough, 2013), we could include sequences that align only over a domain, and propagate function at higher granularity.

Acknowledgements

We thank Dr. Ben Polacco, Dr. Eyal Akiva, Dr. Shoshana Brown, and Ms. Carly Weiss for fruitful discussions. We also thank Prof. Hao Li from University of California, San Francisco and Prof. John Huelsenbeck from University of California, Berkeley for helpful input over the course of this work.

Funding

This work was supported by the National Institutes of Health R01 GM-60595 to PCB.

Conflict of Interest: none declared.

References

Acar,U.A. *et al.* (2012) Adaptive inference on general graphical models. arXiv e-prints 1206.3234.

Almonacid,D.E. and Babbitt,P.C. (2011) Toward mechanistic classification of enzyme functions. *Curr. Opin. Chem. Biol.*, **15**, 435–442.

Altschul,S.F. *et al.* (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.

Ashburner,M. *et al.* (2000) Gene ontology: tool for the unification of biology. *Nat. Genet.*, **25**, 25–29.

Atkinson,H.J. *et al.* (2009) Using sequence similarity networks for visualization of relationships across diverse protein superfamilies. *PLoS ONE*, **4**, e4345.

Barber,A.E. and Babbitt,P.C. (2012) Pythoscape: a framework for generation of large protein similarity networks. *Bioinformatics*, **28**, 2845–2846.

Barutcuoglu,Z. *et al.* (2006) Hierarchical multi-label prediction of gene function. *Bioinformatics*, **22**, 830–836.

Brown,S.D. and Babbitt,P.C. (2012) Inference of functional properties from large-scale analysis of enzyme superfamilies. *J. Biol. Chem.*, **287**, 35–42.

Brown,S.D. and Babbitt,P.C. (2014) New insights about enzyme evolution from large scale studies of sequence and structure relationships. *J. Biol. Chem.*, **289**, 30221–30228.

Buchfink,B. *et al.* (2015) Fast and sensitive protein alignment using DIAMOND. *Nat. Methods*, **12**, 59–60.

Carroll,S. and Pavlovic,V. (2006) Protein classification using probabilistic chain graphs and the gene ontology structure. *Bioinformatics*, **22**, 1871–1878.

Clark,W.T. and Radivojac,P. (2013) Information-theoretic evaluation of predicted ontological annotations. *Bioinformatics*, **29**, i53–i61.

Conesa,A. *et al.* (2005) Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. *Bioinformatics*, **21**, 3674–3676.

Davidson,R. *et al.* (2018) A global view of structure–function relationships in the tautomerase superfamily. *J. Biol. Chem.*, **293**, 2342–2357.

Deng,M. *et al.* (2004) An integrated probabilistic model for functional prediction of proteins. *J. Comput. Biol.*, **11**, 463–475.

Eisner,R. *et al.* (2005) Improving protein function prediction using the hierarchical structure of the gene ontology. In: *2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*. pp. 1–10. IEEE, CA, USA.

Engelhardt,B.E. *et al.* (2011) Genome-scale phylogenetic function annotation of large and diverse protein families. *Genome Res.*, **21**, 1969–1980.

Fang,H. and Gough,J. (2013) dcGO: database of domain-centric ontologies on functions, phenotypes, diseases and more. *Nucleic Acids Res.*, **41**, D536–D544.

Friedberg,I. (2006) Automated protein function prediction–the genomic challenge. *Briefings Bioinf.*, **7**, 225–242.

Gerlt,J.A. and Babbitt,P.C. (2001) Divergent evolution of enzymatic function: mechanistically diverse superfamilies and functionally distinct suprafamilies. *Annu. Rev. Biochem.*, **70**, 209–246.

Gilks,W.R. *et al.* (2005) Percolation of annotation errors through hierarchically structured protein sequence databases. *Math. Biosci.*, **193**, 223–234.

Gogate,V. and Dechter,R. (2011) SampleSearch: importance sampling in presence of determinism. *Artif. Intell.*, **175**, 694–729.

Greene,C.S. and Troyanskaya,O.G. (2012) Accurate evaluation and analysis of functional genomics data and methods. *Ann. N.Y. Acad. Sci.*, **1260**, 95–100.

Hamp,T. *et al.* (2013) Homology-based inference sets the bar high for protein function prediction. *BMC Bioinformatics*, **14**, S7.

Huntley,R. *et al.* (2009) The gene ontology annotation (GOA) database. *Nature Precedings*, **32**, D262–D266.

Jiang,X. *et al.* (2008) Integration of relational and hierarchical network information for protein function prediction. *BMC Bioinf.*, **9**, 350.

Jiang,Y. *et al.* (2016) An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biol.*, **17**, 184.

Kourmpetis,Y.A.I. *et al.* (2010) Bayesian Markov random field analysis for protein function prediction based on network data. *PLoS One*, **5**, e9293.

Letovsky,S. and Kasif,S. (2003) Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics*, **19**, i197–i204.

Li,W. *et al.* (2013) Pclust: protein network visualization highlighting experimental data. *Bioinformatics*, **29**, 2647–2648.

Martin,A.J.M. *et al.* (2013) PANADA: protein association network annotation, determination and analysis. *PLoS One*, **8**, e78383.

Martin,D.M. *et al.* (2004) GOtcha: a new method for prediction of protein function assessed by the annotation of seven genomes. *BMC Bioinf.*, **5**, 178.

Mitrofanova,A. *et al.* (2011) Prediction of protein functions with gene ontology and interspecies protein homology data. *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **8**, 775–784.

Obozinski,G. *et al.* (2008) Consistent probabilistic outputs for protein function prediction. *Genome Biol.*, **9**, S6.

Pearl,J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1st edn. ISBN-10: 1558604790.

Punta,M. *et al*. (2012) The pfam protein families database. *Nucleic Acids Res*, **40**, D290–D301.

Radivojac,P. *et al*. (2013) A large-scale evaluation of computational protein function prediction. *Nat. Methods*, **10**, 221.

Rost,B. (2002) Enzyme function less conserved than anticipated. *J. Mol. Biol*., **318**, 595–608.

Sahraeian,S.M. *et al*. (2015) SIFTER search: a web server for accurate phylogeny-based protein function prediction. *Nucleic Acids Res*., **43**, W141–W147.

Schnoes,A.M. *et al*. (2009) Annotation error in public databases: misannotation of molecular function in enzyme superfamilies. *PLoS Comput. Biol*., **5**, e1000605.

Schnoes,A.M. *et al*. (2013) Biases in the experimental annotations of protein function and their effect on our understanding of protein function space. *PLoS Comput. Biol*., **9**, e1003063.

Shannon,P. *et al*. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res*., **13**, 2498–2504.

Sharan,R. *et al*. (2007) Network-based prediction of protein function. *Mol. Syst. Biol*., **3**, 88.

Sokolov,A. and Ben-Hur,A. (2011) Multi-view prediction of protein function. In: *Proceedings of the 2nd ACM Conference on Bioinformatics, Computational Biology and Biomedicine - BCB '11*. pp. 135–142. ACM Press.

The Gene Ontology Consortium. (2017) Expansion of the Gene Ontology knowledgebase and resource. *Nucleic Acids Res*., **45**, D331–D338.

Tian,W. and Skolnick,J. (2003) How well is enzyme function conserved as a function of pairwise sequence identity? *J. Mol. Biol*., **333**, 863–882.

Todd,A.E. *et al*. (2001) Evolution of function in protein superfamilies, from a structural perspective. *J. Mol. Biol*., **307**, 1113–1143.