Liu, Lizhi; Huang, Xiaodi; Mamitsuka, Hiroshi; Zhu, Shanfeng

HPOLabeler

Published in:
Bioinformatics (Oxford, England)

DOI:
10.1093/bioinformatics/btaa284

Published: 15/07/2020

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Liu, L., Huang, X., Mamitsuka, H., & Zhu, S. (2020). HPOLabeler: improving prediction of human protein-phenotype associations by learning to rank. *Bioinformatics (Oxford, England)*, *36*(14), 4180-4188. https://doi.org/10.1093/bioinformatics/btaa284

Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Original Paper

OXFORD

Genome analysis

HPOLabeler: improving prediction of human protein-phenotype associations by learning to rank

Lizhi Liu [1,2], Xiaodi Huang [3], Hiroshi Mamitsuka [4,5], and Shanfeng Zhu [1,2]*

[1]School of Computer Science and Shanghai Key Lab of Intelligent Information Processing and [2]Shanghai Institute of Artificial Intelligence Algorithms and Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, Shanghai 200433, China, [3]School of Computing and Mathematics, Charles Sturt University, Albury, NSW 2640, Australia, [4]Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji 611-0011, Japan, [5]Department of Computer Science, Aalto University, Finland

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Annotating human proteins by abnormal phenotypes has become an important topic. Human Phenotype Ontology (HPO) is a standardized vocabulary of phenotypic abnormalities encountered in human diseases. As of Nov. 2019, only less than 4,000 proteins have been annotated with HPO. Thus a computational approach for accurately predicting protein-HPO associations would be important, while no methods have outperformed a simple Naive approach in the CAFA2 (second Critical Assessment of Functional Annotation, 2013-14).

Results: We present HPOLabeler, which is able to use a wide variety of evidence, such as protein-protein interaction networks (PPI), Gene Ontology (GO), InterPro, trigram frequency and HPO term frequency, in the framework of learning to rank (LTR). LTR has been proved to be powerful for solving large-scale, multi-label ranking problems in bioinformatics. Given an input protein, LTR outputs the ranked list of HPO terms from a series of input scores given to the candidate HPO terms by component learning models (logistic regression, nearest neighbor and a Naive method), which are trained from given multiple evidence. We empirically evaluate HPOLabeler extensively through mainly two experiments of cross-validation and temporal validation, for which HPOLabeler significantly outperformed all component models and competing methods including the current state-of-the-art method. We further found that 1) PPI is most informative for prediction among diverse data sources, and 2) low prediction performance of temporal validation might be caused by incomplete annotation of new proteins.

Availability: http://issubmission.sjtu.edu.cn/hpolabeler/

Contact: zhusf@fudan.edu.cn

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

With the completion of Human Genome Project, modern geneticists focus on understanding how proteins influence phenotypes by leveraging information hidden in proteomics data (Legrain *et al.*, 2011). For promoting studies on phenomics, researchers develop a standardized vocabulary that describes human abnormal phenotypes and semantic relationships, called Human Phenotype Ontology (HPO) (Köhler *et al.*, 2019). HPO contains over 14,000 terms organized hierarchically in a directed acyclic graph (DAG) (Fig. 1). Each HPO term (a node in the graph) denotes a symptom or a phenotypic abnormality that characterizes a disease. The directed edge between two terms represents an "**is-a**" relationship, implying that the term for a node is more specific to those for ancestors. That is, a given term of a protein can be propagated to all ancestors of the node, which is called "true-path-rule". As of 2018, HPO has five sub-ontologies (CC, CM, MI, PA and Freq) with the common root named *All*.

Currently, around 50% of rare diseases (about 7,000) have been elucidated on the genetic cause, while a large number of genetic disorders have yet to be recognized (Boycott *et al.*, 2013; Chong *et al.*, 2015).

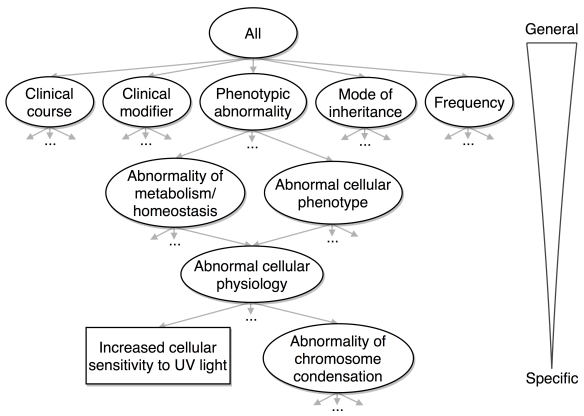

Fig. 1. A fraction of the Human Phenotype Ontology released after Dec. 2017, where each oval denotes an internal node and each rectangle denotes a leaf node.

Also, complex genetic mechanisms of these disorders are unknown (Lowe and Reddy, 2015). The HPO derives its disease↔gene links from NCBI MedGen[1], which in turn derives the links from OMIM [2]. At present, the vast majority of known disease↔gene links for monogenic (Mendelian) diseases are included in the HPO resource. The genetics community widely expects that several thousand additional Mendelian disease genes remain to be discovered. On the other hand, as of Sep. 2019, only less than 4,000 proteins have been associated with HPO terms. That is, currently known disease↔gene links in HPO are just only a tip of true links, while manual curation is very time-consuming. Nonetheless, exploring the relationship between human proteins and abnormal phenotypes is of great importance in the prevention, diagnosis, and treatment of diseases (Groza *et al.*, 2015). Hence, an accurate and efficient computational tool for annotating human proteins with HPO terms is imperative for biomedical research and clinical practice.

Computational HPO annotation of human proteins is a large-scale, multi-label learning problem, in which each human protein (instance) is associated with multiple HPO terms (labels). Solving this problem is very challenging: 1) instance imbalances: For all 14,586 HPO terms by Sep. 2019, more than 1,600 HPO terms are annotated with only one protein, and about 40% of HPO terms are associated with over ten proteins (see Fig. S1). 2) label imbalances: For example, Prelamin-A/C (UniProt ID: P02545) is associated with 955 HPO terms, while Ephrin type-A receptor 2 (P29317) is annotated with only 20 HPO terms. 3) hardness in temporal prediction: CAFA2 had a special track on automatic HPO annotation for *no-knowledge* proteins, which are not associated with any HPO terms before the submission date (Jan. 2014) but received HPO annotations by Sep. 2014 (Jiang *et al.*, 2016). In this track, the best performance in $F_{\max}$ was achieved by a Naive method (Jiang *et al.*, 2016), which just counted the number of appearances of each HPO term in the database. This result implies the hardness of temporal annotation of HPO terms.

For tacking this problem, we propose HPOLabeler, which integrates diverse types of evidence into a Learning to Rank (LTR) framework (see more descriptions on LTR in the Supplementary Materials). LTR was originally developed for ranking documents with respect to an input query, and has been successfully applied to large-scale, multi-label learning problems in bioinformatics, such as biomedical document indexing (Liu *et al.*, 2015), protein function prediction (You *et al.*, 2018), and drug discovery (Yuan *et al.*, 2016). In particular, our previous work on large-scale protein function prediction, GOLabeler, based on LTR (You *et al.*, 2018) won first place in the CAFA3 challenge (Zhou *et al.*, 2019). In this paper, we present a similar method to predict the HPO annotations of human proteins. Now an instance (query) is a human protein, a document is an HPO term (label), and our HPO annotation problem is ranking HPO terms with respect to a protein. As the biggest advantage, LTR allows to integrate multiple types of "evidence", i.e. models trained by different, diverse data sources. Specifically, evidence for each protein can be divided into three types: 1) global evidence: HPO term classifiers trained by protein-protein interaction networks (PPI), Gene Ontology (GO) annotations, InterPro signatures, and amino acid trigrams; 2) local evidence: nearest neighbor (NN) to assign HPO terms of most similar (nearest neighbor) proteins in PPI; 3) prior evidence: HPO frequencies (equivalent to the Naive method in CAFA2).

We extensively evaluated the performance of HPOLabeler through cross-validation and temporal validation (such as CAFA2) over large-scale datasets from the HPO database. We focused on PA, the largest sub-ontology covering over 98% of the total HPO terms. From our experiments, HPOLabeler outperformed all component methods as well as six competing methods. We further empirically found: 1) NN performed well, highlighting the importance of PPI; 2) only HPOLabeler among the competing methods outperformed the Naive method in temporal validation; 3) closer examination revealed that low performance in temporal evaluation might be attributed to the incomplete annotation of new proteins. Finally, we present several typical examples of real protein annotations.

2 Related work

Predicting protein-HPO associations has two categories: 1) filling missing associations (Petegrosso *et al.*, 2016), and 2) predicting HPO annotations of new proteins. We focus on the second category, since only a small fraction of known human proteins have been annotated with HPO terms. Existing methods can be considered from two aspects: 1) data types and/or 2) the way of integrating such data. An early method merged only PPI and ontology with weights after normalization (Wang *et al.*, 2013), which they call PhenoPPIOrth. A supervised learning-based method (specifically with structured support vector machine (SSVM)) used features from PPI, GO and literature by a joint kernel (Kahanda *et al.*, 2015), an extension of GOstruct (Sokolov and Ben-Hur, 2010; Sokolov *et al.*, 2013) into the HPO prediction (PHENOstruct). Also, SSVM was further used for indirect prediction of SSVM→disease→HPO (S→D→H) in (Kahanda *et al.*, 2015). A recent approach of HPO2GO used only one information source, i.e. GO, focusing on reliable co-occurring HPO-GO mappings (Doğan, 2018). According to the hierarchical structure (DAG), the flat classification results can be refined to keep the biological consistency. In this way, two methods called HTD-DAG and TPR-DAG (Notaro *et al.*, 2017*a*,*b*) are developed. In contrast to existing methods, HPOLabeler makes use of LTR to integrate multiple types of evidence, which are generated from heterogeneous data sources and different basic models.

3 Methods

3.1 Notations

Table 1 shows the notations used in this paper.

3.2 Problem formulation

Let $D_l = \{(p_i, T_i)\}_{i=1}^m$ be a given dataset, where human protein p_i has a set of experimentally validated annotations $T_i \subseteq \mathcal{O}$, where $\mathcal{O}$ is the Human Phenotype Ontology and "$\subseteq$" is a consistent subgraph of the ontology (Radivojac, 2013). This means that all ancestors of vertex $v \in T_i$ are in T_i. We denote D_{add} as data additionally applied in prediction.

[1] https://www.ncbi.nlm.nih.gov/medgen/

[2] https://www.omim.org/

Table 1. Table of notations.

Notation	Meaning
$\mathcal{O}$	Human Phenotype Ontology
p	Protein
t	HPO term
$\mathcal{P} = \{p_1, p_2, \cdots, p_m\}$	Set of proteins, where p_i is the i-th protein and m is the cardinality
$\mathcal{P}_S, \mathcal{P}_L, \mathcal{P}_T$	Training sets for component models and LTR, and test set, respectively
m_S, m_L, m_T	Cardinality of $\mathcal{P}_S$, $\mathcal{P}_L$, and $\mathcal{P}_T$, respectively
$\mathbf{x}_i^{(f)} = \left(x_{i,1}^{(f)}, x_{i,2}^{(f)}, \cdots, x_{i,n^{(f)}}^{(f)}\right)^\mathsf{T}$	Feature vector of protein p_i using data source f, where $x_{i,j}^{(f)}$ denotes the j-th feature and $n^{(f)}$ denotes the number of features
$\mathbf{y}_t = (y_{1,t}, y_{2,t}, \cdots, y_{m,t})^\mathsf{T}$	Label vector of t, if p_i is annotated with t then $y_{i,t} = 1$, otherwise 0
$S^{(f)}(p, t)$	Likelihood score between p and t
$\mathbf{x}_t^{(\mathrm{LTR})}$	Feature vector of term t for LTR

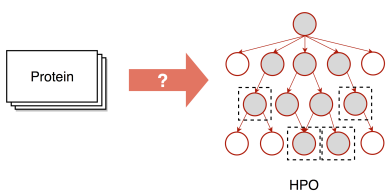

Fig. 2. HPO prediction problem: The left-hand side is just a protein (schematic amino acid sequence). On the other hand, on the right-hand side, the dark nodes are predicted terms, for which four leaf nodes (highlighted by dashed lines) are originally predicted.

We now formulate the problem of HPO prediction: Given D_l and unseen protein p in D_{add}, the problem is to infer a consistent subgraph $\hat{T} \subseteq \mathcal{O}$ that maximizes the probability of correct annotation T of p. More formally, we have

$$\hat{T} = \arg\max_{T \subseteq \mathcal{O}} P(T|p)$$

where $P(T|p)$ is the probability of T being correct as annotations of protein p. Fig. 2 is a schematic picture of this problem.

It is particularly noteworthy that the problem we attempt to tackle in this work is to predict the HPO annotation for a given protein, rather than to predict which protein is most likely related to a given HPO term. The formulations of these two problems are quite different in practice.

3.3 Feature generation for component models

3.3.1 Protein-protein interaction networks (PPI)

PPI refers to specific contacts between two or more protein molecules (Nooren and Thornton, 2003). We generate PPI features from three databases: STRING, GeneMANIA, and BioGRID.

STRING (Szklarczyk *et al.*, 2014): For protein p_i, we can write a real-valued vector $\mathbf{x}_i^{(\mathrm{STR})}$ from STRING:

$$\mathbf{x}_i^{(\mathrm{STR})} = \left(x_{i,1}^{(\mathrm{STR})}, x_{i,2}^{(\mathrm{STR})}, \cdots, x_{i,n^{(\mathrm{STR})}}^{(\mathrm{STR})}\right)^\mathsf{T}, \quad (1)$$

where $x_{i,j}^{(\mathrm{STR})}$ is the score of protein p_j interacting with p_i.

GeneMANIA (Warde-Farley *et al.*, 2010): For protein p_i, feature vector $\mathbf{x}_i^{(\mathrm{GM})}$ from GeneMANIA can be written as:

$$\mathbf{x}_i^{(\mathrm{GM})} = \left(x_{i,1}^{(\mathrm{GM})}, x_{i,2}^{(\mathrm{GM})}, \cdots, x_{i,n^{(\mathrm{GM})}}^{(\mathrm{GM})}\right)^\mathsf{T}, \quad (2)$$

where $x_{i,j}^{(\mathrm{GM})}$ is the score of p_i associated with protein p_j.

BioGRID (Chatr-aryamontri *et al.*, 2017): For protein p_i, a feature vector $\mathbf{x}_i^{(\mathrm{BGD})}$ is a binary vector:

$$\mathbf{x}_i^{(\mathrm{BGD})} = \left(x_{i,1}^{(\mathrm{BGD})}, x_{i,2}^{(\mathrm{BGD})}, \cdots, x_{i,n^{(\mathrm{BGD})}}^{(\mathrm{BGD})}\right)^\mathsf{T}, \quad (3)$$

where $x_{i,j}^{(\mathrm{BGD})} = 1$ if p_i interacts with protein p_j; otherwise 0.

3.3.2 Gene Ontology (GO) annotations

Similar to HPO, GO provides gene functions and uses DAG to represent relationships among gene functions. GO covers three different domains: Biological Process (BP), Cellular Component (CC) and Molecular Function (MF). We extract GO annotations from Gene Ontology Consortium (The Gene Ontology Consortium, 2017) and UniProt-GOA (Huntley *et al.*, 2015), removing records without evidence codes of EXP, IDA, IPI, IMP, IGI, IEP, TAS, and IC. All remaining annotations are used without any redundancy, and "true-path-rule" is also applied for propagating annotations. For protein p_i, we have three binary vectors:

$$\mathbf{x}_i^{(\mathrm{GOXX})} = \left(x_{i,1}^{(\mathrm{GOXX})}, x_{i,2}^{(\mathrm{GOXX})}, \cdots, x_{i,n^{(\mathrm{GOXX})}}^{(\mathrm{GOXX})}\right)^\mathsf{T}, \quad (4)$$

where XX denotes BP, CC or MF, and $x_{i,j}^{(\mathrm{GOXX})}$ is 1 if p_i is annotated by GO term j in subontology XX; otherwise zero.

3.3.3 InterPro (Finn *et al.*, 2017)

We use InterProScan (Jones *et al.*, 2014), which predicts domains and functional sites of the input protein (sequence) by predictive models, as signatures. For protein p_i, binary feature vector $\mathbf{x}_i^{(\mathrm{IPR})}$ is given as:

$$\mathbf{x}_i^{(\mathrm{IPR})} = \left(x_{i,1}^{(\mathrm{IPR})}, x_{i,2}^{(\mathrm{IPR})}, \cdots, x_{i,n^{(\mathrm{IPR})}}^{(\mathrm{IPR})}\right)^\mathsf{T}, \quad (5)$$

where $x_{i,j}^{(\mathrm{IPR})} = 1$ if p_i has signature j in InterPro; otherwise zero.

3.3.4 Trigrams

Regarding a protein as a string with 20 letters (amino acids), we count the frequencies of all possible trigrams (3-mer) in the string, resulting in a feature vector with the size of 8,000 = 20^3. For protein p_i, feature vector $\mathbf{x}_i^{(\mathrm{TRI})}$ is given as follows:

$$\mathbf{x}_i^{(\mathrm{TRI})} = \left(x_{i,1}^{(\mathrm{TRI})}, x_{i,2}^{(\mathrm{TRI})}, \cdots, x_{i,n^{(\mathrm{TRI})}}^{(\mathrm{TRI})}\right)^\mathsf{T}, \quad (6)$$

where each entry is the frequency of the corresponding trigram.

3.4 Component models

LTR uses learning (component) models, which can be trained from the above feature vectors. We used the following three components.

3.4.1 Logistic regression (LR)

For protein p_i and HPO term t, from feature vector $\mathbf{x}_i^{(f)}$ of data source f, the predicted score $S^{(f)}(p_i, t)$ can be written as:

$$S^{(f)}(p_i, t) = \mathcal{L}_t^{(f)}(\mathbf{x}_i^{(f)}) = P\left(y_{i,t} = 1 | \mathbf{x}_i^{(f)}\right), \quad (7)$$

where $\mathcal{L}_t^{(f)}$ is LR for HPO term t and data source f.

LR is used for each of the total eight generated feature sets: STRING, GeneMANIA and BioGRID from PPI, BP, CC and MF from GO, InterPro and Trigrams. This means that eight scores for each protein – HPO term pair.

3.4.2 Nearest neighbor (NN)

Two proteins with strong interactions are likely to be associated with the same phenotype (Goh *et al.*, 2007). For protein p_i and HPO term t, the associated score is given as:

$$S^{(\text{NBR-G})}(p_i,t)=\frac{\sum_{p_j\in N_G(p_i)} d(p_i,p_j)\cdot y_{j,t}}{\sum_{p_j\in N_G(p_i)} d(p_i,p_j)}, \tag{8}$$

where $N_G(p_i)$ is the set of nodes (proteins) adjacent to p_i in PPI G, and $d(p_i,p_j)$ is the interaction score between p_i and p_j.

We used STRING, GeneMANIA, and BioGRID for PPI: STRING and GeneMANIA provide interaction score $d(p_i,p_j)$, while this score is unavailable in BioGRID. For BioGRID, we then compute the interaction score, due to (Chua *et al.*, 2007), as follows:

$$d(p_i,p_j)=\frac{2|N_G[p_i]\bigcap N_G[p_j]|}{|N_G[p_i]-N_G[p_j]|+2|N_G[p_i]\bigcap N_G[p_j]|+1}\times \frac{2|N_G[p_i]\bigcap N_G[p_j]|}{|N_G[p_j]-N_G[p_i]|+2|N_G[p_i]\bigcap N_G[p_j]|+1},$$

where $N_G[p_i]$ is the set of adjacent proteins (including p_i) to p_i in PPI G, and $|Q|$ indicates the cardinally of set Q.

A well-known and partially proved assumption is that proteins with strong interaction in a protein-protein interaction network are likely involved in the same cellular process, and tend to be related to similar disease phenotypes (Xu and Li, 2006; Oti *et al.*, 2006; Gandhi *et al.*, 2006; Köhler *et al.*, 2008). Based on this assumption, we run the NN method on the three networks of STRING, GeneMANIA, and BioGRID. However, it is slow to run NN on other types of datasets with no network information (such as domain, Trigram and GO). In order to compromise the efficiency and effectiveness of our method, we thus use LR for all eight models, but use NN only for PPI .

3.4.3 Naive method (Naive)

We use the number of appearances of HPO term t as the score of all proteins for HPO term t (note that this score is always the same for any protein):

$$S^{(\text{Naive})}(p_i,t)=\frac{|\{p_j\in\mathcal{P}_S|y_{j,t}=1\}|}{m_S} \tag{9}$$

3.5 HPOLabeler

HPOLabeler consists of three steps. Fig. 3 and Algorithm 1 show a schematic flow and pseudocode of the algorithm, respectively.The procedure is as follows. Given a query protein, HPOLabeler generates multiple types of features from nine sources for this particular protein such as PPI, GO,Trigrams, and InterPro. Different types of twelve component methods are then trained by using different features. For example, NN uses each of the three PPI, while LR uses all the eight types of features. The output of each component method is a list of an HPO term-score pair for the query protein. HPOLabeler then selects the top-k_{ont} candidate HPO terms from each of twelve lists, and combines them as the final HPO candidates. At the same time, the predicted scores by the component methods for each HPO term in the final candidates are concatenated as a feature vector. LTR accepts this vector as its input to compute the score for this term again, and ranks all terms in the final candidates. The top-ranked HPO candidates are produced as the final output of HPOLabeler.

3.5.1 Candidate generation

By using the scores from component models, we first sort HPO terms of each protein in descending order of the scores and select top-k_{ont} of HPO terms as candidates. We have twelve models (8 LR, 3 NN and 1 Naive) for each pair of protein – HPO term. In other words, for each protein, the top-k_{ont} candidate HPO terms are chosen from each of the prediction results of twelve models, and then we union them as the final candidates.

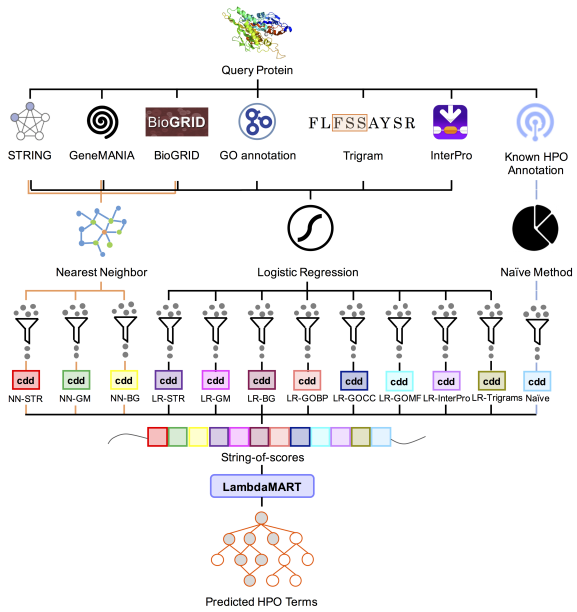

Fig. 3. Overview of HPOLabeler. Given a query protein, eight features are first generated, with three on PPI, i.e. STRING, GeneMANIA and BIOGRID, three from GO, i.e. BP, CC and MF, Trigrams and InterPro. Then three types of component models are trained: NN from each of the three PPI, LR from each of all eight features, and Naive (directly from the input), resulting in 12 models producing 12 scores, which are concatenated into a feature vector by string-of-scores. Finally, LTR (LambdaMART) re-ranks HPO terms by using the feature vector as input.

Algorithm 1 HPOLabeler

Input: Query protein p, HPO $\mathcal{O}$;
Output: A ranked list;
Construct eight feature vectors, by using (1)-(6) ;
Compute $S^{(M)}(p,t)$ for method M and term $t\in\mathcal{O}$, by using (7)-(9);
for each sub-ontology ont of $\mathcal{O}$ **do**
  $\mathcal{C}=\varnothing$;
  for each component method M **do**
    Sort $S^{(M)}(p,t)$ in the descending order where t in ont;
    Select top-k_{ont} terms t from ranked list $S^{(M)}(p,t)$ as $\mathcal{C}^{(M)}$;
    $\mathcal{C}=\mathcal{C}\bigcup\mathcal{C}^{(M)}$;
  Construct $\mathbf{x}_t^{(\text{LTR})}$ for each term $t\in\mathcal{C}$ according to (10);
  Compute $S^{(\text{LTR})}(p,t)$ for each term $t\in\mathcal{C}$ by LambdaMART;
Sort $S^{(\text{LTR})}(p,t)$ in the descending order as the final ranked list;

3.5.2 Feature generation for LTR

For each candidate HPO term, we concatenate the predicted scores from 12 models to generate a feature vector to be input to LTR (we call this concatenation (representation) *string-of-scores*). Specifically, for protein p and HPO term t, the generated feature vector can be written as follows:

$$\mathbf{x}_t^{(\text{LTR})}=\left(\text{——}\ \ S^{(M)}(p,t)\ \ \text{——}\right)^{\mathsf{T}}, \tag{10}$$

where M indicates one of the twelve scores. Note that we normalize the range of each score into $[0,1]$.

3.5.3 Ranking

As a state-of-the-art LTR algorithm, LambdaMART (Burges, 2010) casts the ranking problem as a *pairwise* regression one. By ranking, the algorithm can tell which HPO term is better in a given pair of HPO terms with respect to a query protein. In fact, LambdaMART combines MART (Multiple Additive Regression Trees) (Friedman, 2001) and LambdaRank

(Burges *et al.*, 2007). Specifically, MART is a boosted tree model with its final output as a linear combination of the outputs of a collection of regression trees. For solving the ranking task, LambdaMART modifies gradient boosted decision trees that are originally used for prediction, specifying the gradients derived from LambdaRank. The details of LambdaMART are provided in the Supplementary Material 1.

We use LambdaMART to re-rank the candidate HPO terms, with score feature vector $\mathbf{x}_t^{(\mathrm{LTR})}$ as the input of LambdaMART and then receive a ranked HPO term list with predicted scores, $S^{(\mathrm{LTR})}(p,t)$.

3.5.4 A toy example

To better illustrate the procedure of HPOLabeler, we provide a pictorial toy example in Supplementary Material 2. For details, please refer to Figure S5.

4 Experiments

4.1 Data

We briefly describe the data used below (details are in the supplement).

4.1.1 Cross-validation

We conduct cross-validation on gene-HPO annotations released on 2018-07-27[3]. All other data sources we used, such as GO and InterPro annotations, are up to this date. The version of each data source is reported in Tables S1, and the data split strategy is described in Figure S6 of supplementary material 2. We first map genes in annotations into proteins by using the UniProt ID mapping tool. To ensure the quality of data, the proteins that are not stored in Swiss-Prot are filtered out. The true-path-rule is then applied here to propagate annotations. It is worth noting that we ignore the roots of the whole HPO and the sub-ontologies. Finally, we divide the entire dataset into five equal parts, taking four of them as $\mathcal{P}_S$, and the remaining one in half, with one half as $\mathcal{P}_L$, and another half as $\mathcal{P}_T$. In this way, we carry out five-fold cross-validation of two rounds per fold by swapping $\mathcal{P}_L$ and $\mathcal{P}_T$. This strategy enables each sample to be assigned to the test set once. The results we report below are the average of results of ten rounds. For all ten results, we use the Wilcoxon signed-rank test to check the statistic significance of performance differences of the different methods, where p-value less than 0.05 is deemed as statistically significant.

We further divided all HPO terms into six groups, according to the number of appearances (in brackets): Very rare (1-3), Rare (4-10), Uncommon (11-30), Common (31-100), Very common (100-300), and Extremely common (>300), to examine the properties of HPOLabeler further. Fig. 4 shows the statistics of these six groups. The left pie chart indicates that HPO terms with low appearances, i.e. Very rare, Rare and Uncommon, occupy 37.8%, 20.9% and 17.3%, respectively, of all HPO terms, while the right one shows that these HPO terms are only small parts of all protein-HPO term pairs, i.e. Very rare, Rare and Uncommon, occupying only 1.1%, 2.4% and 5.8%, respectively, of all pairs.

4.1.2 Temporal validation

For the temporal validation, we extract the genes that already have HPO annotations before 2017-02-24[4], as $\mathcal{P}_S$. The genes added from 2017-02-24 to 2018-03-09[5] form $\mathcal{P}_L$, while newly annotated genes added between 2018-03-09 and 2018-12-21[6] consist of $\mathcal{P}_T$. Since some HPO terms are obsolete and redirected to new terms, these new HPO terms are mapped to the old ones released on 2017-02-24 by using their corresponding positions. Moreover, the newly created HPO terms are discarded. As a result, we unify the three datasets into the February 2017 HPO release. To avoid information leakage, all data sources utilized were released before March 2018. The version of each data source is reported in Tables S5, and the data split strategy is described in Figure S7 of supplementary material 2. Table 2 summarizes the statistics of these data.

[3] http://compbio.charite.de/jenkins/job/hpo.annotations/1259/

[4] http://compbio.charite.de/jenkins/job/hpo.annotations/1236/

[5] http://compbio.charite.de/jenkins/job/hpo.annotations/1252/

[6] http://compbio.charite.de/jenkins/job/hpo.annotations/1263/

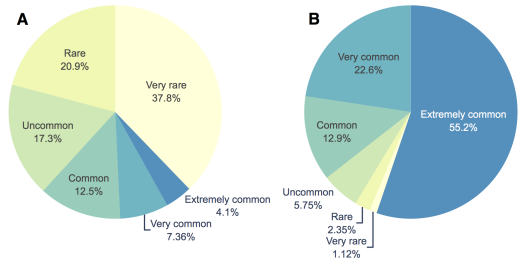

Fig. 4. Statistics on groups of HPO terms in a cross-validation scenario. (A) The proportion of HPO terms in each group. (B) The proportion of HPO annotations related to the terms in each group.

Table 2. Statistics of dataset used for temporal validation.

	$\mathcal{P}_S$ (train)	$\mathcal{P}_L$ (LTR)	$\mathcal{P}_T$ (test)
	∼2017-02	2017-02 ∼ 2018-03	2018-03 ∼ 2018-12
#Proteins	3,334	304	226
#HPO terms	7,394	2,836	2,091
#Annotations per protein	107.1	83.9	61.5

4.2 Evaluation metrics

We used the following three metrics for performance evaluation.

Protein-centric ($F_{\max}$): For cut-off $\tau \in [0,1]$, precision $Pr(\tau)$ and recall $Rc(\tau)$ can be defined as follows:

$$Pr(\tau) = \frac{1}{m(\tau)} \sum_{i=1}^{m(\tau)} \frac{\sum_{t\in\mathcal{O}} \mathbb{I}\left(t \in P_{p_i}(\tau) \bigwedge t \in T_{p_i}\right)}{\sum_{t\in\mathcal{O}} \mathbb{I}\left(t \in P_{p_i}(\tau)\right)},$$

$$Rc(\tau) = \frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{t\in\mathcal{O}} \mathbb{I}\left(t \in P_{p_i}(\tau) \bigwedge t \in T_{p_i}\right)}{\sum_{t\in\mathcal{O}} \mathbb{I}\left(t \in T_{p_i}\right)},$$

where for p_i, $P_{p_i}(\tau)$ is a set of terms with predicted scores no less than τ and T_{p_i} is a set of annotated terms (true labels), $m(\tau)$ is the number of proteins with HPO annotation with predicted scores no less than τ, and N is the total number of proteins. $\mathbb{I}(\cdot)$ is the indicator function. Then maximum F-Measure, $F_{\max}$, can be computed as follows:

$$F_{\max} = \max_{\tau} \left\{ \frac{2 \cdot Pr(\tau) \cdot Rc(\tau)}{Pr(\tau) + Rc(\tau)} \right\}.$$

Note that $F_{\max}$ is an official evaluation metric in CAFA.

Term-centric (AUC): For cut-off τ and HPO term t, sensitivity Sn and specificity Sp can be computed as follows:

$$Sn(\tau) = \frac{\sum_{i=1}^{N} \mathbb{I}\left(t \in P_{p_i}(\tau) \bigwedge t \in T_{p_i}\right)}{\sum_{i=1}^{N} \mathbb{I}\left(t \in T_{p_i}\right)},$$

$$Sp(\tau) = \frac{\sum_{i=1}^{N} \mathbb{I}\left(t \notin P_{p_i}(\tau) \bigwedge t \notin T_{p_i}\right)}{\sum_{i=1}^{N} \mathbb{I}\left(t \notin T_{p_i}\right)}.$$

We then computed AUC_t (area under ROC curve) of term t by using the ROC curve, obtained by plotting $(1 - Sp(\tau), Sn(\tau))$ changing τ, and obtained the final AUC by averaging AUC_t over all terms.

Pairwise (AUPR): The Area Under Precision-Recall curve (AUPR) is computed by regarding a pair of protein-HPO term as an instance. AUPR provides a comprehensive evaluation of the performance of the algorithm.

4.3 Competing methods and implementation details

We compare our model with several baseline methods. The descriptions of the baselines and their parameter settings in the experiments are as follows.

PHENOstruct and S→D→H (Kahanda *et al.*, 2015): We use the same features as ours for training and predicting. The parameters follow its default setting: $C = 1.0$.

SVM & LR: The traditional SVM and LR are implemented by Python 3.7 and scikit-learn 0.19.2 with their default parameters. Each classifier is trained and tested on a single HPO term using the concatenation of the feature vectors from STRING, GeneMANIA, BioGRID, InterPro, GO and Trigram.

HTD-DAG & TPR-DAG (Notaro *et al.*, 2017*a*,*b*): We adopt SVM as the flat learner. Without parameters for HTD-DAG, we set $t = 0.5$ and $w = 0.5$ for TPR-DAG.

HPO2GO (Doğan, 2018): The version of the GO annotations used in the approach is the same as ours. The specific values of the parameters are selected by the algorithm itself from the ranges of $S \in \{0, 0.1, 0.2, \cdots, 0.6\}$ and $n \in \{1, 2, 3, 4, 5\}$.

PhenoPPIOrth (Wang *et al.*, 2013): We downloaded prediction results from their website[7] with $\lambda = 0.8$.

Naive: The computation method is the same as the one described in Section 3.4.3.

Note that for the fair comparisons, we used $\mathcal{P}_S \bigcup \mathcal{P}_L$ as the training set, and still $\mathcal{P}_T$ as the test set for all the competing methods.

For our method, the component LR is implemented by scikit-learn with its default settings. To distinguish the Naive method as a component from that as a competing method, the training dataset is $\mathcal{P}_S$. XGBoost (Chen and Guestrin, 2016) is then adopted to implement LambdaMART by setting the `booster` to `gbtree` and the `objective` to `rank:pairwise`. To avoid over-fitting, the maximum depth of the tree `max_depth` is set to 4. Candidate generation is a critical step in HPOLabeler. A small k_{ont} may result in missing some HPO terms, while a large k_{ont} could reduce efficiency and bring noise into the ranking process. As such, we carried out some prior experiments with cross validation over training dataP_S by varying $k_{ont} \in \{60, 80, 100, 120, 140, 160, 180, 200\}$. We found the performance of HPOLabeler usually achieves the best with $k_{ont} = 120$, and decreases slightly with a higher k_{ont}. By considering this, k_{ont} is set to 120 in our experiments (see Fig. S2 in the supplement for the results of different k_{ont}).

4.4 Experimental results of cross-validation

Table S2 in supplement shows the statistics of data with 3,722 proteins, 8,067 HPO terms, and 119.4 annotations per protein on average.

4.4.1 Component model performance

Table 3 shows the performance of component models. This table indicates that the nearest neighbor with STRING (NN-STRING) achieved the best performance, followed by NN-GeneMANIA. Also, LR with PPI outperformed LR with other data sources, implying that PPI is most informative, which is consistent with the result in Kahanda *et al.* (2015). Besides, NN performed better than LR. In particular, NN-STRING and NN-GeneMANIA achieved AUPR of more than 0.35, which was around 10% better than those of LR-STRING and LR-GeneMANIA. This sizable difference indicates that NN made more effective use of PPI than LR.

[7] http://jjwanglab.org/PhenoPPIOrth/

Table 3. Component model performance of cross-validation.

Component	$F_{\max}$	AUC	AUPR
LR-STRING	0.4174	0.6390	0.2697
LR-GeneMANIA	0.3506	0.7282	0.2605
LR-BioGRID	0.3441	0.5941	0.2677
LR-GO BP	0.3777	0.6741	0.2926
LR-GO CC	0.3643	0.6544	0.2916
LR-GO MF	0.3343	0.6081	0.2403
LR-InterPro	0.3588	0.6041	0.2699
LR-Trigrams	0.2941	0.5136	0.1564
NN-STRING	**0.4213**	**0.7892**	**0.3635**
NN-GeneMANIA	0.4110	0.7274	0.3550
NN-BioGRID	0.3529	0.6407	0.2822
Naive	0.3517	0.5	0.2590

Table 4. Competing method performance. Asterisks for HPOLabeler indicate the performance improvements with $p < 0.001$ by Wilcoxon signed-rank test.

Method	$F_{\max}$	AUC	AUPR
PHENOstruct	0.4228	0.7760	0.3596
S→D→H	0.3476	0.7606	0.2580
SVM	0.4055	0.6831	0.2900
LR	0.4242	0.6690	0.2972
HTD-DAG	0.4134	0.6832	0.2951
TPR-DAG	0.4253	0.6840	0.3170
PhenoPPIOrth	0.1430	0.5731	0.0558
HPO2GO	0.2751	0.5395	0.0936
Naive	0.3517	0.5	0.2591
HPOLabeler	**0.4688***	**0.7956**	**0.4293***

4.4.2 Leave-one-source-out performance check

We examined the effectiveness of each model by the cross-validation result obtained by eliminating each model. We call this "leave-one-source-out", in which a model with the most reduced performance is more informative. The result (see Fig. S3) shows that the performance of the model eliminating PPI was most reduced.

4.4.3 Competing method performance

Table 4 summarizes the performance of competing methods. This table clearly shows HPOLabeler outperformed all competing methods. In particular, in $F_{\max}$ and AUPR, the performance difference between HPOLabeler and all other methods were all significant ($p < 0.001$) by Wilcoxon signed-rank test. In fact, HPOLabeler improved $F_{\max}$ and AUPR of PHENOstruct, the recent state-of-the-art method, by 10.9% and 19.4%, respectively. From Tables 3 and 4, interestingly, even NN-STRING already outperformed most of the competing methods, indicating again that both PPI is most informative and NN is useful for prediction. However, NN-STRING was significantly beaten by HPOLabeler. In addition, by comparing HPOLabeler with SVM and LR, which were trained by the same set of features, the performance of LTR was impressive. Entire results proved that the ensemble framework of LTR with heterogeneous information is highly powerful for the HPO prediction problem.

To more specifically analyze the performance under the term-centric evaluation, we divide the HPO terms into six groups according to their frequency. As can be seen from Fig. 4, the HPO terms annotating only a few proteins account for the majority of the entire HPO. As such, the experiment results on low-frequency groups have little value. Due to the space limitation, we only list the AUCs of four out of the six groups in

Table 5. AUC for four datasets. Asterisks again show statistical significance ($p < 0.001$) by Wilcoxon signed-rank test. Com. means "Common".

Method	Uncommon	Com.	Very Com.	Extremely Com.
PHENOstruct	**0.8161**	0.7888	0.7748	0.7501
S→D→H	0.7925	0.7619	0.7324	0.6895
SVM	0.6690	0.6851	0.6989	0.6937
LR	0.6429	0.6704	0.6974	0.7023
HTD-DAG	0.6716	0.6842	0.6971	0.6928
TPR-DAG	0.6689	0.6849	0.7005	0.7009
PhenoPPIOrth	0.5961	0.5745	0.5562	0.5231
HPO2GO	0.5521	0.5347	0.5267	0.5306
Naive	0.5	0.5	0.5	0.5
HPOLabeler	0.7922	**0.8046***	**0.8082***	**0.7778***

Table 6. Competing method performance of temporal validation.

Method	$F_{\max}$	AUC	AUPR
PHENOstruct	0.3054	0.6362	0.1424
S→D→H	0.1461	0.5473	0.0603
SVM	0.2791	0.5929	0.1077
LR	0.2956	0.5950	0.1119
HTD-DAG	0.2933	0.5956	0.1138
TPR-DAG	0.3002	0.5962	0.1235
PhenoPPIOrth	0.0678	0.5219	0.0121
HPO2GO	0.2075	0.5083	0.0277
Naive	0.3097	0.5	0.2147
HPOLabeler	**0.3415**	**0.6398**	**0.2342**

Table 5, indicating that HPOLabeler outperformed all other methods again. The full results can be found in Tables S3 and S4 in the Supplementary Material. The groups with statistically significant results are Common, Very Common, and Extremely Common, i.e. HPO terms assigned for many proteins. This implies that the performance of HPOLabeler is superior for more common HPO terms.

4.5 Experimental results of temporal validation

4.5.1 Performance of component models

Table S6 in supplement shows the performance of component models. It indicates that NN-STRING was the best in all three metrics, which is consistent with the result of cross-validation.

4.5.2 Performance of competing methods

Table 6 shows the performance results, indicating HPOLabeler outperformed all others, followed by Naive. In other words, HPOLabeler was the only method that beat Naive. This result demonstrates the advantage of HPOLabeler in annotating HPO terms of new proteins. For temporal validation, again we generated six groups, according to the number of appearances of HPO terms. The statistics are provided in Fig. S4. Due to the space limitation, the results are provided in Tables S7 and S8 in the supplement, with consistent cross-validation results.

4.5.3 Discussions

Table 6 shows that except HPOLabeler, Naive was best, with the consistent results of CAFA2 Challenge (Jiang *et al.*, 2016). Naive uses only the distribution of HPO terms in the given data. The advantage of Naive was exploited by the fact that a large number of HPO terms are associated with most of proteins, resulting in a similar HPO term distribution for any protein (Jiang *et al.*, 2016). Additionally, the relatively good performance of Naive implies another reason. Table 2 shows the average number of

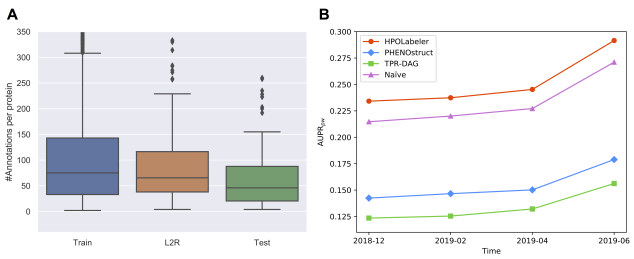

Fig. 5. A. #HPO terms associated with a protein in $\mathcal{P}_S$, $\mathcal{P}_L$ and $\mathcal{P}_T$. B. Four competing methods by AUPR in temporal validation, using annotations released at different times.

annotations per protein on $\mathcal{P}_S$, $\mathcal{P}_L$, and $\mathcal{P}_T$ were 107.1, 83.9, and 61.5, respectively, which are changing heavily, implying three different distributions of HPO terms. Fig. 5A shows box plots on the distribution of the number of HPO terms annotated per protein, where the box becomes lower from left to right. Obviously, newly annotated proteins have a smaller number of HPO annotations than old proteins. Thus we can imagine that HPO terms annotated wrongly might be currently false positives by incomplete annotations and become positives by biological research development in the future. Fig. 5B shows the performance (AUPR) of the competing methods by updating our datasets using recent releases. This figure indicates the increase of AUPR as time goes by, for all four methods shown. Thus again the low performance of temporal validation might be caused by incomplete annotations for relatively new proteins.

4.5.4 HPO term examples becoming from negatives to positives

We checked some annotations that were highly ranked in the predicted list of HPO terms but regarded as negatives (not in the Dec 2018 release of HPO), and examined these annotations by using a more recent database, i.e., Feb. 2019 release. As examples, Table 7 shows three proteins, each with three HPO terms, highly ranked but negatives. Surprisingly, all these nine annotations could be found in the newer release. This means that all HPO terms in Table 7 are eventually positives (see Table S9 for detail). This is not rare cases but more regular. For example, seven out of the top 10 predictions of Q96F07 (second protein in Table 7) were evaluated as negatives but now found as positives. This finding implies incompleteness of HPO annotations on new proteins. Also, this demonstrates that the real accuracy of HPOLabeler in temporal validation would be much higher than those in Table 6.

5 Conclusion

For the HPO annotation problem, we have proposed HPOLabeler, which is able to integrate diverse types of evidence, such as PPI, GO, InterPro, and trigrams, in the framework of LTR. We empirically validated the performance of HPOLabeler, which significantly outperformed all component models, as well as six competing methods, particularly on HPO terms with a higher number of appearances. Further examinations of the experimental results indicates that: 1) PPI is the most informative data source, and 2) lower predictive performance in temporal validation might be caused by incomplete annotations of new proteins. The possible future work on improving the predictive performance further would be to integrate our framework with new machine learning approaches, such as deep learning.

Funding

This work was supported in part by the National Natural Science Foundation of China (Nos. 61572139 and 61872094).

Table 7. Three example proteins, each with three example HPO terms, which were all highly ranked and evaluated as negatives, but appeared in the latest release in February 2019, meaning that all nine HPO terms are eventually positives.

UniProt ID	Protein name	Gene symbol	Disease ID	HPO term ID	HPO term name	Rank
Q08209	Serine/threonine-protein phosphatase 2B catalytic subunit alpha isoform	PPP3CA	ORPHA:442835 OMIM:617711	HP:0000924	Abnormality of the skeletal system	3
				HP:0011842	Abnormality of skeletal morphology	9
				HP:0025031	Abnormality of the digestive system	18
Q96F07	Cytoplasmic FMR1-interacting protein 2	CYFIP2	ORPHA:442835 OMIM:618008	HP:0000152	Abnormality of head or neck	1
				HP:0000234	Abnormality of the head	1
				HP:0000924	Abnormality of the skeletal system	3
P61981	14-3-3 protein gamma	YWHAG	ORPHA:442835 OMIM:617665	HP:0000478	Abnormality of the eye	3
				HP:0000152	Abnormality of head or neck	8
				HP:0000234	Abnormality of the head	9

"Disease ID" refers to the diseases related to the corresponding protein. "HPO term ID" refers to the HPO terms annotated to the corresponding protein.

References

Boycott,K. *et al.* (2013) Rare-disease genetics in the era of next-generation sequencing: discovery to translation. *Nat. Rev. Genet.,* **14** (10), 681–691.

Burges,C. (2010). From RankNet to LambdaRank to LambdaMART: An Overview. Technical report Microsoft Research.

Burges,C. *et al.* (2007) Learning to rank with nonsmooth cost functions. In *Advances in neural information processing systems* pp. 193–200.

Chatr-aryamontri,A. *et al.* (2017) The BioGRID interaction database: 2017 update. *Nucleic Acids Res.,* **45** (D1), D369–D379.

Chen,T. and Guestrin,C. (2016) XGBoost: a scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* pp. 785–794 ACM.

Chong,J. *et al.* (2015) The genetic basis of Mendelian phenotypes: discoveries, challenges, and opportunities. *Am. J. Hum. Genet.,* **97** (2), 199–215.

Chua,H. *et al.* (2007) An efficient strategy for extensive integration of diverse biological data for protein function prediction. *Bioinformatics,* **23** (24), 3364–3373.

Doğan,T. (2018) HPO2GO: prediction of human phenotype ontology term associations for proteins using cross ontology annotation co–occurrences. *PeerJ,* **6**, e5298.

Finn,R. *et al.* (2017) InterPro in 2017–beyond protein family and domain annotations. *Nucleic Acids Res.,* **45** (D1), D190–D199.

Friedman,J. (2001) Greedy function approximation: a gradient boosting machine. *Ann. Stat.,* **29** (5), 1189–1232.

Gandhi,T. *et al.* (2006) Analysis of the human protein interactome and comparison with yeast, worm and fly interaction datasets. *Nat. Genet.,* **38** (3), 285–293.

Goh,K.I. *et al.* (2007) The human disease network. *Proc. Natl. Acad. Sci. U.S.A.,* **104** (21), 8685–8690.

Groza,T. *et al.* (2015) The Human Phenotype Ontology: semantic unification of common and rare disease. *Am. J. Hum. Genet.,* **97** (1), 111–124.

Huntley,R. *et al.* (2015) The GOA database: Gene Ontology annotation updates for 2015. *Nucleic Acids Res.,* **43** (D1), D1057–D1063.

Jiang,Y. *et al.* (2016) An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biol.,* **17** (1), 184.

Jones,P. *et al.* (2014) InterProScan 5: genome–scale protein function classification. *Bioinformatics,* **30** (9), 1236–1240.

Kahanda,I. *et al.* (2015) PHENOstruct: Prediction of human phenotype ontology terms using heterogeneous data sources [version 1; referees: 2 approved]. *F1000Res.,* **4**, 259.

Köhler,S. *et al.* (2008) Walking the interactome for prioritization of candidate disease genes. *Am. J. Hum. Genet.,* **82** (4), 949–958.

Köhler,S. *et al.* (2019) Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources. *Nucleic Acids Res.,* **47** (D1), D1018–D1027.

Legrain,P. *et al.* (2011) The human proteome project: current state and future direction. *Mol. Cell. Proteomics,* **10** (7), M111–009993.

Liu,K. *et al.* (2015) MeSHLabeler: improving the accuracy of large-scale MeSH indexing by integrating diverse evidence. *Bioinformatics,* **31** (12), i339–i347.

Lowe,W.L. and Reddy,T. (2015) Genomic approaches for understanding the genetics of complex disease. *Genome Res.,* **25** (10), 1432–1441.

Nooren,I. and Thornton,J. (2003) Diversity of protein–protein interactions. *EMBO J.,* **22**, 3486–3492.

Notaro,M. *et al.* (2017*a*) Ensembling descendant term classifiers to improve gene–abnormal phenotype predictions. In *Proceedings of the 14th International Conference on Computational Intelligence methods for Bioinformatics and Biostatistics* Springer.

Notaro,M. *et al.* (2017*b*) Prediction of Human Phenotype Ontology terms by means of hierarchical ensemble methods. *BMC Bioinformatics,* **18**, 449.

Oti,M. *et al.* (2006) Predicting disease genes using protein–protein interactions. *J. Med. Genet.,* **43** (8), 691–698.

Petegrosso,R. *et al.* (2016) Transfer learning across ontologies for phenome–genome association prediction. *Bioinformatics,* **33** (4), 529–536.

Radivojac,P. (2013). A (not so) quick introduction to protein function prediction.

Sokolov,A. and Ben-Hur,A. (2010) Hierarchical classification of gene ontology terms using the GOstruct method. *J. Bioinform. Comput. Biol.,* **8** (2), 357–376.

Sokolov,A. *et al.* (2013) Combining heterogeneous data sources for accurate functional annotation of proteins. *BMC Bioinformatics,* **14** (Suppl 3), S10.

Szklarczyk,D. *et al.* (2014) STRING v10: protein–protein interaction networks, integrated over the tree of life. *Nucleic Acids Res.,* **43** (D1), D447–D452.

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

Wang,P. *et al.* (2013) Inference of gene–phenotype associations via protein–protein interaction and orthology. *PLOS One,* **8** (10), e77478.

Warde-Farley,D. *et al.* (2010) The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function. *Nucleic Acids Res.,* **38** (suppl_2), W214–W220.

Xu,J. and Li,Y. (2006) Discovering disease-genes by topological features in human protein–protein interaction network. *Bioinformatics,* **22** (22), 2800–2805.

You,R. *et al.* (2018) GOLabeler: improving sequence-based large-scale protein function prediction by learning to rank. *Bioinformatics,* **34** (14), 2465–2473.

Yuan,Q. *et al.* (2016) DrugE-Rank: improving drug–target interaction prediction of new candidate drugs or targets by ensemble learning to rank. *Bioinformatics,* **32** (12), i18–i27.

Zhou,N. *et al.* (2019) The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. *Genome Biol.,* **20** (1), 1–23.