Novel Comparison of Evaluation Metrics for Gene Ontology Classifiers Reveals Drastic Performance Differences

Ilya Plyusnin[1], Liisa Holm[1,2], Petri Törönen[1]

1 Institute of Biotechnology/University of Helsinki/Helsinki, Finland
2 Department of Biosciences/University of Helsinki, Helsinki, Finland

* Petri.Toronen@Helsinki.fi

Abstract

GO classifiers and other methods for automatic annotations of novel sequences play an important role in modern biosciences. It is thus important to assess the quality of different GO classifiers. Evaluation of GO classifiers depends heavily on the used evaluation metrics. Still, there has been little research on the effect of different metrics on the produced method ranking. Indeed most evaluation metrics are simply borrowed from machine learning without any testing for their applicability to GO classification.

We propose a novel simple comparison of metrics, called Artificial Dilution Series. We start by selecting a set of annotations that are known a priori to be correct. From this set we create multiple copies and introduce different amount of errors to each copy. This creates a "series" of annotation sets with the percentage of original correct annotations ("signal") decreasing from one end of the series to the other. Next we test metrics to see which of them are good at separating annotation sets at different signal levels. In addition, we test metrics with various false positive annotation sets, and show where they rank in the generated signal range.

We compared a large set of evaluation metrics with ADS, revealing drastic differences between them. Especially we show how some metrics consider false positive datasets as good as 100 % correct data sets and how some metrics perform poorly at separating the different error levels. This work A) shows that evaluation metrics should be tested for their performance; B) presents a software that can be used to test different metrics on real-life datasets; C) gives guide lines on what evaluation metrics perform well with Gene Ontology structure; D) proposes improved versions for some well-known evaluation metrics. The presented methods are also applicable to other areas of science where evaluation of prediction results is non-trivial.

Author Summary

Comparison of predictive methods is one of the central tasks in science and bioinformatics is no exception. Currently predictive methods are increasingly needed as biosciences are

producing novel sequences at an ever higher rate. These sequences require Automated Function Prediction (AFP) as manual curation is often impossible. Unfortunately, selecting AFP methods is a confusing task as current AFP publications show a mixed set of functions for method comparison called Evaluation Metrics (metrics for short). Furthermore, many existing popular metrics can generate misleading or unreasonable results in AFP comparison. We argue that the usage of badly performing metrics in AFP comparison is caused by the lack of methods that can be used to benchmark the metrics. We propose a testing method, called Artificial Dilution Series (ADS). It can be used to test any group of metrics on selected real life test dataset. ADS uses selected dataset to create a large set of artificial AFP results, where each AFP result has a controlled amount of errors. We use ADS to compare how different metrics are able to separate generated error proportions. Our results show drastic differences between different metrics.

Introduction

Biosciences generate sequences at a much higher rate than their structure, function or interactions can be experimentally determined. This has created a demand for computational methods that can automatically link novel sequences to their biological function (Automated Function Prediction, AFP, [1]). Development in this field has been exceptionally swift and has generated a large collection of AFP methods [2, 3]. Therefore the comparison (or understanding the comparison) of different methods is an important task a) for bioscientists who need AFP methods in their sequencing projects b) for bioinformaticians who are comparing or developing AFP methods and c) for reviewers who are evaluating novel AFP tools.

Still, the evaluation of the AFP tools is far from trivial, having two major challenges: Articles use different datasets and different evaluation metrics in their result evaluations. This makes it unclear to the end-users how different methods perform and to the method developers how the comparisons should be done. For datasets there are currently some standards forming, like dataset used in MouseFunc project [4] or datasets used by CAFA competitions [2, 3]. Furthermore, our own evaluations have also used a filtered set of well-annotated sequences [5, 6].

However, the main challenge in AFP comparisons is the selection of Evaluation Metric (EvM, also called Performance Metric). EvM is the very function that defines how good and bad results are separated from each other [3, 7–9]. Currently the AFP field uses various EvMs with no clear standards. Table 1 in supplementary S2 Text demonstrates this by listing EvMs used in various articles. Furthermore, although CAFA competitions [2, 3] have generated standards, even these have obtained some criticism. Gillis and Pavlidis [10] stated on CAFA1: *"the primary performance metric ... for CAFA is unsatisfactory. ... by this measure, a null 'prediction method' outperforms most methods."* Kahanda et al. [11] pointed that ordering of methods varies drastically between different EvMs. In addition, also CAFA authors have stated in both CAFA1 [2] and in CAFA2 [3] articles that the EvM for method comparison is still an open research question. This shows that there is a clear need for research on AFP related EvMs.

These EvM problems are related to the Gene Ontology [12] structure that is used to store functional annotations. It represents the following challenges:

- The GO structure has a very large number of separate classes and genes can belong to many of them. This generates a multi-labelled and multiclass classification task.
- Hierarchy in the GO structure causes strong and complex correlations between classes.
- Class sizes vary dramatically with most classes being very small. This causes strong class size imbalance.

We also have additional challenges, caused by the used biological data:

- The set of genes, used in the evaluation, can be quite small when compared to the number of classes [2]. This makes the correct annotation table very sparse.
- The meaning and definition of true negative GO annotations is ambiguous [13, 14].
- Each gene has a varying number of correct annotations.

These challenges point to the selection of EvM being clearly nontrivial. Furthermore, the EvM selection problem is not limited to AFP comparison, but is a problem that occurs in many fields [7, 8]. Weaknesses of EvMs have been discussed, for example, with speech recognition [15] and with the classification of cognitive activities [16]. Finally, EvM often forms the core of the results section in bioinformatic articles and it can be selected so that it favors one's own method [17]. So understanding the strengths and weaknesses of current EvMs and developing new ones should be important for the whole bioinformatics - machine learning community. However, we are currently lacking clear standard on how EvM could be benchmarked for any given application area.

We propose a novel methodology, **Artificial Dilution Series (ADS)**, to address these challenges. ADS checks the performance and stability of any EvM using real-life GO annotations. ADS creates artificial classifier results by taking a set of correct GO predictions and replacing a controlled percentage of correct annotations with wrong ones (creating type 1 and type 2 errors, collectively referred to as noise). The percentage of noise is then increased in a step-wise fashion, creating a set of separate result datasets with a controlled level of noise at each step. This process creates a "dilution series" of the original signal in a collection of altered datasets. This series is then used to test different EvMs to see how well they separate datasets with different signal levels.

Furthermore, we included a secondary test to our analysis by creating various **False Positive (FP) datasets**. We run each EvM with them, and compare the obtained result to scores in the ADS series. A good EvM would place FP sets close to sets from zero signal. FP sets allow monitoring how each EvM ranks for example the naive predictor [2, 3] that was stated to be the main problem of one EvM [10]. Our supplementary text S2 Text compares the proposed tests to existing related research.

We tested many EvMs previously applied to GO annotations, using three datasets: CAFA1 dataset [2], MouseFunc dataset [4] and one consisting of 1000 GO annotated sequences randomly selected from the UniProt database. We also tested many variations of EvMs by: *a*) calculating metric values first for every gene and using the average over the genes as the final value (Gene Centric) *b*) calculating metric values first for every GO class and using the average over the GO classes as the final value (Term Centric) *c*) calculating metric value using simply the list of gene - GO class pairs (Unstructured). Methods section and supplementary text S2 Text explain the EvMs more in detail.

Our results show that *1*) Area Under Curve (AUC) tests often fail with FP tests, *2*) semantic scores [18] are heavily effected by the used semantic score summation method, with the currently popular methods failing either in tests with ADS or with FP sets and *3*) a metric called SimGIC [19] and its variations showed consistently good performance.

We also tested simple modifications on EvMs currently used for evaluation of AFPs. Some of these improved the performance:

1. We tested AUC with Precision Recall curve and showed that it outperforms ROC AUC in most tests.
2. We proposed novel semantic similarity summation methods and showed dramatic improvement in their performance.
3. We combined Smin [20] and SimGIC [19] function and showed that this improved performance.

Examples 2 and 3 represent novel EvMs. These and all tested EvMs are available in distributed software[1].

Our results show that the most consistent performance, across all the datasets, was obtained with Term Centric AUC score for Precision Recall curves and with modified SimGIC functions. Some EvMs showed good performance on two out of three tested datasets. These were Term Centric ROC AUC score and one version of Lin semantics. These results point that different types of GO datasets can impose different challenges, suggesting testing candidate metrics on target data with ADS and FPS. ADS program, all tested EvMs and our analysis scripts are available for such tests.

Materials and Methods

Generating Artificial Dilution Series

Motivation and Requirements

First, we expect that we have a set of GO class annotations for a set of genes, referred here as truth set, T. This includes (gene, GO class) pairs that link genes to selected GO classes. Our aim with Artificial Dilution Series (ADS) is to create several datasets, each representing functional classifications for genes in T. These datasets, referred here as *Artificial Prediction Sets* (APS), imitate a GO prediction set from a typical GO classifier. However, with APS we know the frequency of correct and erroneous annotations. APS's are created artificially without any classifier or similar method from a given set of genes and their correct annotations.

We require the following features from each APS:

1. APS should include the same genes and approximately the same GO classes as the starting data T.

[1] http://ekhidna2.biocenter.helsinki.fi/ADS/

2. APS generation should use the same ontological structure that is used in the evaluation of the real results.

3. APS can have signal and noise GO predictions for genes.

4. Signal GO term prediction, t_{signal}, is either a correct GO term, t_{corr}, observed in the real data for the gene in question (G_i). Or it can be a parent node of the t_{corr} in the GO graph.

5. Noise GO term prediction, t_{noise}, is a GO prediction that is not in the same part of GO tree with any of the t_{corr} for gene G_i.

6. The proportion of noise prediction in each APS should be controlled with noise proportion, p_{noise}.

7. Each prediction should have a prediction score, representing the strength of the prediction.

8. The proportion of noise predictions should be allowed to increase as the prediction score is weakened.

Point 1 and 2 ensure that the generated data has similar class counts and correlation structures as the real datasets. Point 4 is based on the observations with real life GO prediction tools. Those often predict also near-by nodes, that include almost identical set of classes with t_{corr}. Point 5 ensures that t_{noise} should be considered as bad result with any reasonable evaluation metric. Point 6 allows us to generate several datasets for each value of p_{noise}. With 4, 5 and 6 we are creating datasets that should get clearly different results, for different p_{noise}, with a good GO evaluation metric. Points 7 and 8 are based on the observations from the real life predictions, where classifier score is used to rank the results. Top ranks in these predictions are often reliable, whereas the lowest rank are often quite unreliable.

Workflow for ADS pipeline is given in Fig 1. Generation of AP sets is also described as pseudocode in S1 Text. In the first step two GO prediction sets are created for each APS: the *Positive Prediction set*, P_{pos}, and the *Negative Prediction set*, P_{neg}.

Step 1: Create Positive Set

The Positive set, P_{pos}, is created from a copy of T, in two steps: First, we draw a random integer N_{shift} between 0 and the size of P_{pos}. Further, we select N_{shift} random rows from P_{pos} and for every selected row switch GO term, t_{corr}, to semantically similar term. For this, we select a neighborhood of k terms that are immediate parents of the t_{corr} and select one to replace t_{corr}. We refer to this as *shifting to semantic neighbors*. Figure 1 in supplementary text S2 Text demonstrates this step.

Second, we introduce a selected percentage of errors, ϵ, by switching GO terms between genes for a controlled fraction of annotation lines. We refer to this as the *permutation of the positive set*. Here a random pair of annotation lines, $a = (gene_a, t_a)$ and $b = (gene_b, t_b)$, with different gene and GO class names, is selected from P_{pos}. Next, if the GO class t_a is dissimilar to every GO class annotated to $gene_b$ and GO class t_b is dissimilar to every GO class annotated to $gene_a$, the GO labels are interchanged (see Fig 2). The similarity between the two classes, t_x and t_y, is defined here as Jaccard correlation between the two sets created from the ancestral classes of t_x and t_y.

The permutation process is repeated until the required percentage of errors ϵ is created. This introduces a controlled fraction of false positives (wrong predictions) and an equivalent fraction of false negatives (missing correct predictions) among the predictions. We refer to this fraction as the *ADS noise level* and the remaining fraction of correct positive annotations as the *ADS signal level* (1 – ADS noise level).

Step 2: Create Negative Set

The negative set, P_{neg}, is created by assigning genes in T with random GO annotations. Here we collect all the unique gene names that occur in T and for every $gene_x$ we sample a random GO term, t_r, from the GO structure. Again, if t_r is dissimilar to all GO classes linked to $gene_x$, we include the pair $(gene_x, t_r)$ into P_{neg}. This process is repeated 4 times for every gene.

Step 3: Add Prediction Scores and Merge Sets

Next, all the entries in P_{pos} and P_{neg} are assigned prediction scores, sc. Predictions in the positive set are assigned higher scores. For each prediction in P_{pos} we sample a sc from normal distribution with $mean = 1$ and $SD = 0.5$ and add it to the prediction, creating the triplet $(gene_x, t_x, sc)$. For each prediction in P_{neg} we sample sc from normal distribution with $mean = -1$ and $SD = 0.5$ and add it similarly to the prediction. Note that we assign higher scores to entries in P_{pos} than to entries in P_{neg}. If the score values have to be within some region, like [0.1], one could run a sigmoid function on the score values.

Finally, the positive set and the negative set are merged to form an APS $APS = P_{pos} \cup P_{neg}$. What we now have is a set of GO predictions with A) a set of errors, represented by P_{neg}, that can be separated by setting a threshold on sc values and B) a set of errors hidden into P_{pos}, that cannot be separated with a threshold (ADS noise level). Furthermore, A stays constant for all APS's, whereas B is adjusted with ϵ.

For metrics that are not based on gene ontology (ROC AUC, Jaccard index, Fmax, Smin etc.) predictions are extended to include ancestor nodes in GO hierarchy (in line with true path rule [18]). For metrics that are based on semantic similarity this extension is omitted since these metrics understand the dependencies between related GO classes [18].

Step 4: Creating AP Sets with All ADS Signal Levels

Full ADS analysis creates AP sets, with ADS signal levels changing incrementally over the selected signal range, $SR = [x_1, x_2, ..x_k]$, with k different levels. We used SR that spans from 100% to 0% signal in steps of ten percentage points. However, our implementation allows to span any range with any number of steps. At each signal level the APS generation is repeated N times, each time applying shifting to semantic neighborhood and permutation procedure as described above. This way the final ADS series will contain:

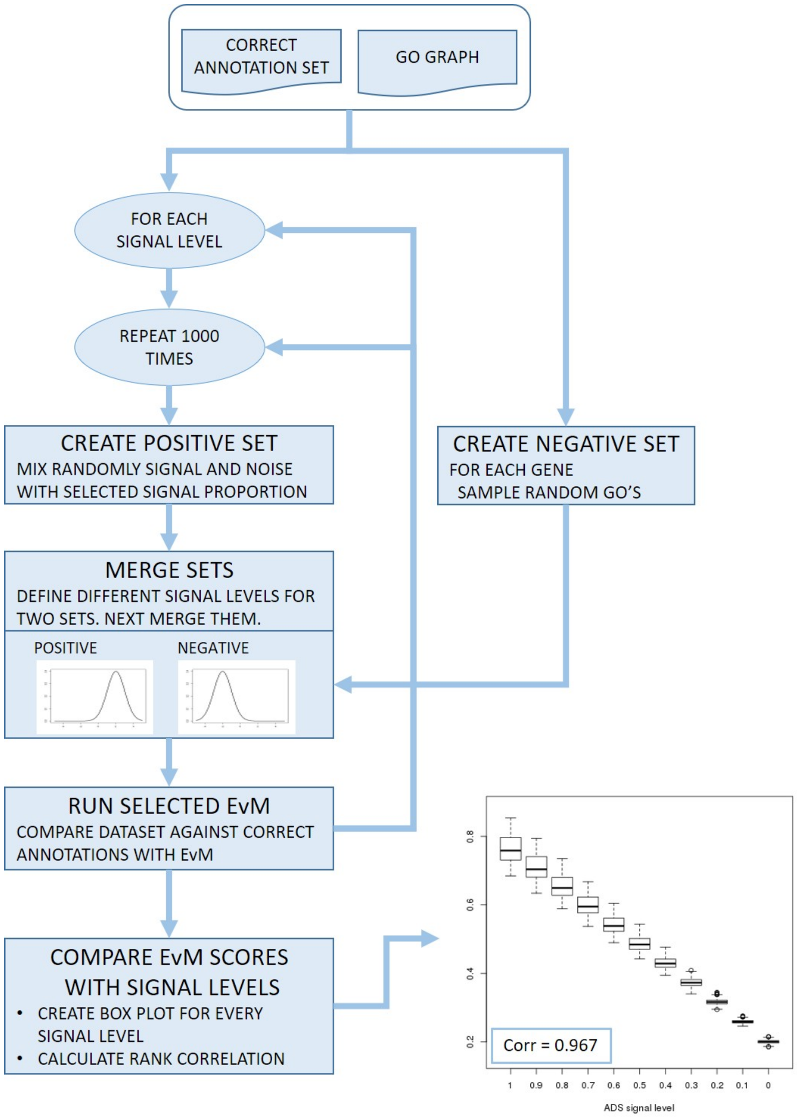
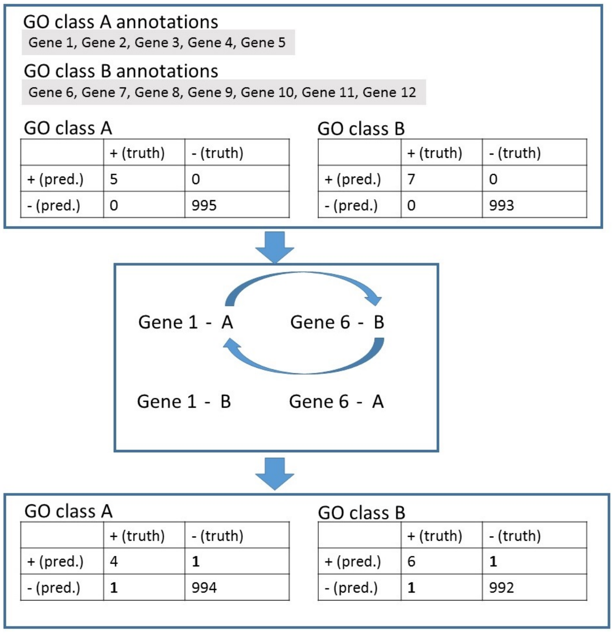

CORRECT ANNOTATION SET
GO GRAPH
FOR EACH SIGNAL LEVEL
REPEAT 1000 TIMES
CREATE POSITIVE SET
MIX RANDOMLY SIGNAL AND NOISE WITH SELECTED SIGNAL PROPORTION
CREATE NEGATIVE SET
FOR EACH GENE SAMPLE RANDOM GO'S
MERGE SETS
DEFINE DIFFERENT SIGNAL LEVELS FOR TWO SETS. NEXT MERGE THEM.
POSITIVE
NEGATIVE
RUN SELECTED EvM
COMPARE DATASET AGAINST CORRECT ANNOTATIONS WITH EvM
COMPARE EvM SCORES WITH SIGNAL LEVELS
• CREATE BOX PLOT FOR EVERY SIGNAL LEVEL
• CALCULATE RANK CORRELATION
0.8
0.6
0.4
0.2
Corr = 0.967
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
ADS signal level

Fig 1. ADS workflow ADS pipeline iterates specified signal levels (e.g. from 100% to 0%) at each level creating a large collection of artificial prediction sets (AP sets). Each AP set is a union of a set containing only false annotations (negative set) and a set containing a controlled fraction of true annotations (positive set). AP sets are compared against correct annotations with the tested Evaluation Metric (EvM). Performance can be illustrated graphically by plotting EvM scores for AP sets at each signal level as box plots. These reveal how stable EvM is against random variation introduced at different signal levels and how well EvM is able to track signal retained in different AP sets. Finally EvM performance is quantified with rank correlation.

- Evenly sampled ADS signal levels between selected signal minimum and maximum.
- *N* separate AP sets representing each signal level.


GO class A annotations
Gene 1, Gene 2, Gene 3, Gene 4, Gene 5
GO class B annotations
Gene 6, Gene 7, Gene 8, Gene 9, Gene 10, Gene 11, Gene 12
Gene 1 - A
Gene 6 - B
Gene 1 - B
Gene 6 - A

GO class A

	+ (truth)	- (truth)
+ (pred.)	5	0
- (pred.)	0	995

GO class B

	+ (truth)	- (truth)
+ (pred.)	7	0
- (pred.)	0	993

GO class A

	+ (truth)	- (truth)
+ (pred.)	4	**1**
- (pred.)	**1**	994

GO class B

	+ (truth)	- (truth)
+ (pred.)	6	**1**
- (pred.)	**1**	992

Fig 2. Effect of permutation step Figure demonstrates the ADS permutation step, where we define the amount of error, on toy data. We start with the known annotations for two GO classes, A and B. This would give us 100% correct prediction. Next, permutation step switches GO-terms for a pair of genes, increasing false positives and false negatives by 1 and decreasing true positives and true negatives by 1. Permutations are repeated until the required noise level is obtained. We require that the interchanged GO classes have only little overlap in the parental path in the GO graph (for details see figure 1 in S2 Text).

- Percentage of annotations shifted to semantic neighbors varying randomly across AP sets at the same signal level.
- Annotations selected for permutation and shifted to semantic neighbors vary across AP sets at the same signal level.
- Proportion of correct or near correct annotations is changing as a function of ADS signal level.

So we have datasets that have variations within the ADS signal level that evaluation metric should be more or less insensitive to and variation across ADS signal levels that evaluation metric should monitor.

Testing Evaluation Metrics with ADS

Generated AP sets can be compared to the selected correct set T that we used as the starting material, using the tested evaluation metric. This gives us an output matrix:

$$Output[i,j] = Metric(T, APS[i,j]), \tag{1}$$

where $Metric$ is the tested evaluation metric, columns, $j = [1, 2, ..k]$, correspond to different ADS signal levels and rows $i = [1, 2..N]$ represents the different repetitions with the same ADS signal level. With good metric we expect to see little variance in the results within one ADS signal level (within $Output$ columns) and clear separation between the different ADS signal levels (between $Output$ columns). We test this in two ways: A) Visually by plotting $Output$ columns as separate box plots at different ADS noise levels (see Fig. 4) and B) Numerically by calculating rank correlation, RC, between $Output$ matrix and ADS signal levels. Boxplot view allows here an overview on the performance of the tested metric. RC monitors how closely EvM is able to predict the correct ranking of the ADS signal levels. One could consider linear correlation but that would favor linear correlation over correct rank.

False Positive datasets

In addition to AP sets we also generated a small number of prediction sets, where all genes were annotated either with exactly the same set of non-informative GO terms or with sets of randomly chosen GO-terms. These represent various cases where some evaluation metrics might be fooled to assign good scores to prediction set that contains no signal. We refer to these sets as the *False Positive Sets* (FPS). We considered the following alternatives for FPS:

1. Each gene has the same very large set of GO terms as predictions. We call this *all positive set*. Most of the terms assigned to each gene are incorrect.
2. Each gene has the very same set of largest classes as predictions. Classes are reported in their size order for each gene. We call this *naïve set*.
3. Each gene has a large set of randomly selected classes. We call this *random set*.

The motivation for the *all positive set* is to test the sensitivity of the evaluation metric to false positive predictions (precision of the classifier is very low). Note that a set where each gene has all GO classes would be totally unfeasible file to analyze. Therefore, we decided to select a large set of the smallest classes. *Random set* shows the performance when both precision and recall are very low. *Naïve set* tests if the evaluation metric can be fooled by bias in null probabilities to consider the results meaningful. Naïve set was originally proposed in the CAFA competitions [2], where it was referred as naive prediction. Note that all of these FP sets are totally decoupled from the annotated sequences and contain no real information about the corresponding genes.

There is a parameter that needs to be considered for FPS generation: the number of GO classes that one reports per gene. We selected this to be quite large (800 GO classes). This emphasizes the potential inability of the metric to separate noise signal in the FP sets from true signal in predictions output by classifiers. We used three false positive sets:

1. Naïve-800: all genes assigned to the 800 most frequent GO classes. Here larger GO class had always a better score.
2. Small-800: all genes assigned to the 800 least frequent GO classes. Here smaller GO class had always a better score.
3. Random-800: all genes assigned by a random sample of 800 GO classes (sampling without replacement).

Naïve-800 represents our naive set, small-800 represents our all positive set and Random-800 is our random set. GO frequencies for FPS sampling were estimated from the UniProt GO annotation[2].

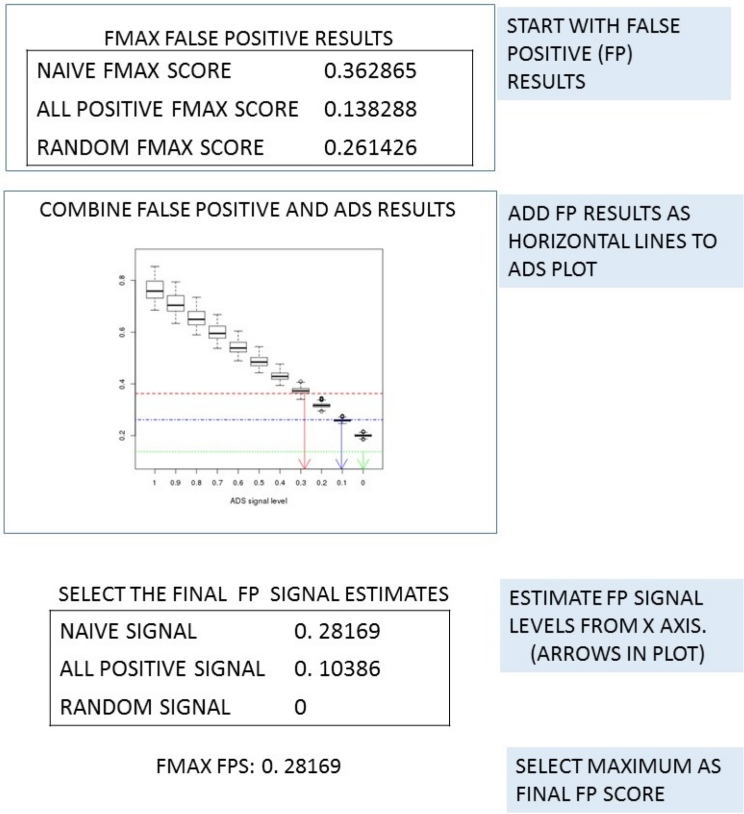
In the visual analysis we plot the score for each FP set as a horizontal line over the box plot of APS scores. A well-performing evaluation metric is expected to assign all FP set scores close to AP sets with minimal or no signal (*signal* 0). In visualizations this would show as a horizontal line at the lowest region of the plot (see fig. 3).

In numerical analysis we summarize FP sets by **1**) Finding the approximate point where the horizontal FP set line crosses the curve, created by the median lines of boxplots (see fig. 3) **2**) choosing the corresponding signal level from the X axis as the FP score for FP set in question and **3**) selecting the maximum over all FP sets as the final FP Score (referred later as FPS). Steps **1** and **2** aim to find the corresponding signal level over ADS sets for each FP set. Note that in step **2** we limit the signal between zero and one. In step **3** we focus on the worst result over all tested FP sets. A good EvM should not be sensitive towards any of FP the sets, making the worst signal (=highest FP score) the best measure for the insensitivity towards these biases.

Tested Evaluation Metrics

We selected representatives from various types of EvMs that are commonly applied to function predictors (see S2 Text). We also included modifications of commonly used EvMs. Tested EvMs can be grouped into different families: **rank based metrics**, **GO semantic**

[2] (ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/UniProt/, 03.2016)



FMAX FALSE POSITIVE RESULTS

NAIVE FMAX SCORE	0.362865
ALL POSITIVE FMAX SCORE	0.138288
RANDOM FMAX SCORE	0.261426

START WITH FALSE POSITIVE (FP) RESULTS

COMBINE FALSE POSITIVE AND ADS RESULTS

ADD FP RESULTS AS HORIZONTAL LINES TO ADS PLOT

SELECT THE FINAL FP SIGNAL ESTIMATES

NAIVE SIGNAL	0. 28169
ALL POSITIVE SIGNAL	0. 10386
RANDOM SIGNAL	0

ESTIMATE FP SIGNAL LEVELS FROM X AXIS. (ARROWS IN PLOT)

FMAX FPS: 0. 28169

SELECT MAXIMUM AS FINAL FP SCORE

Fig 3. Combining FP sets with ADS results Here Fmax results from False Positive (FP) sets are combined with Fmax box plots from ADS. FP results are added as vertical lines to visualization. Robust metric would score FP lines near the lowest box plot. An FS signal estimate is obtained by comparing each FP result with medians of each box. Note that resulting signals are limited between 0 and 1.

similarity based metrics and **group based metrics**. Note that there is a large variety of EvMs available, especially in machine learning [7, 9], and we cannot cover them all. Rather, we try to create an overview of some alternatives.

We evaluated two rank based metrics: Area Under ROC-Curve (AUC or ROC AUC) and Area Under Precision-Recall Curve (AUC-PR). Both strategies capture the amount of false positive and false negative errors at different threshold levels. We evaluated three semantic similarities: Lin [21], Resnik [22] and Ancestor Jaccard (AJacc). AJacc is a novel method. It is simply a Jaccard correlation between two sets of GO classes: ancestor classes for the first and ancestor classes for the second GO class. Group based methods compare a set of predicted and correct GO classes to each other. From this group we evaluated SimGIC [19], Jaccard correlation and Smin [20].

Most of these metrics have been used actively in AFP evaluation (see table 1 in supplementary text S2 Text). Although AUC-PR has not been directly used, it is related to popular Precision-Recall plots. All metrics except rank based metrics can be calculated using different thresholds over prediction scores. We selected the maximum score as the final score for each metric over the set of tested threshold values.

Semantic similarity metrics return a matrix of similarity values between the correct and the predicted GO classes (see table 2 in Supplementary text S2 Text). This matrix needs to be summarised into a single score at each tested threshold. As there is no clear recommendations for this, we tested six alternatives:

A Mean of matrix

B Mean of column maxima

C Mean of row maxima

D Mean of B and C

E Minimum of B and C

F Mean of concatenated row and column maxima

Note that methods A and D have been widely used before [18]. Methods B and C represent intentionally flawed methods, used here as negative controls. B is weak at monitoring false positive predictions and C is weak at monitoring false negatives. Methods E and F represent our novel simple improvements to method D. Our supplementary text S2 Text describes the used semantics and summation methods more in detail.

Most EvMs can be further modified by using the same core function with different *Data Structuring*. By data structuring we refer to the gene-centric, term-centric or unstructured evaluation. In **Gene-Centric evaluation (GC)** we evaluate the predicted set of GO classes against the true set separately for each gene and then summarise these values with the mean over all genes. In **Term-Centric evaluation (TC)** we compare the set of genes assigned to a given GO term in predictions against the true set separately for each GO term and summarise these values with the mean over all GO terms. In **UnStructured evaluation (US)** we compare predictions as a single set of gene-GO pairs disregarding any grouping by shared genes or GO classes.

In total we tested 37 metrics (summarised in S1 Table). Further discussion and metric definitions are given in S2 Text.

Used datasets and run parameters

All metrics were tested on three annotation data sets: the evaluation set used in CAFA I [2], the evaluation set from mouseFunc competition [4] and a random sample of 1000 genes from UniProt GO annotation [3]. These datasets represent three alternatives for evaluating an AFP tool (see Introduction).

These datasets have different features. CAFA data is the smallest one, having 1642 Gene-GO annotations over 698 genes. After propagation to parent classes these annotations map to 3493 GO classes, creating a matrix where only 0.91% of the cells have annotation. UniProt dataset is an intermediate with 5076 Gene-GO annotations over 1000 genes. After propagation these annotations map to 5104 GO classes, creating a matrix with 0.98% of cells having an annotation. MouseFunc data is the largest one with 24879 Gene-GO annotations over 1945 genes. These map to 3334 unique GO classes after propagation,

[3] ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/UniProt/, 03.2016

creating a matrix with 1.42% of cells with annotation. Furthermore, the distribution of annotations is skewed over both the GO classes and genes (for details see S2 Table). Here it is interesting to see how different densities affect the EvM performances.

We tested all metrics with two values for the k (semantic neighborhood size) parameter: $k = 2$ and $k = 4$. RC and FPS scores for these runs are listed in S3 Table and S4 Table. Metric scores for all AP and FP sets are available in S1 File and S2 File.

Results

All discussed results are generated using analysis scripts that are available from our web page (http://ekhidna2.biocenter.helsinki.fi/ADS/). We first show differences within groups of EvMs, then show differences between datasets and finally show best performing methods over all datasets.

ADS shows large differences in results for popular evaluation metrics

We first compare popular evaluation metrics: US AUC, Fmax, Smin, Resnik with summary method D and A and Lin with summary method D to see if ADS can find performance differences. Visualisations for these metrics are plotted in Fig 4 (for exact values see S3 Table).

Results with UniProt data show drastic differences between the tested evaluation metrics. We see a very clear separation across ADS signal levels in the box plots for Fmax ($RC = 0.990$) and Smin ($RC = 0.991$), and next best separation for US AUC. The opposite is shown by three examples of semantic similarity measures: they have very unstable distribution in the obtained scores. Resnik scores show here the worst separation. Note that it is clearly impossible to estimate the original amount of signal based on these semantic similarity measures.

One could argue that the comparison of fully random data and data with strong signal is enough for metric evaluations. Our results, however, show that in most plots the first ($signal = 1$) and last ($signal = 0$) box plot do not fully reveal how good the evaluation metric is. For Lin score and Resnik score two extreme box plots might seem as acceptable with clear separation. However, the intermediate levels reveal surprising instability in the methods.

With FP datasets we see different methods failing. Now AUC based method shows the worst performance as it ranks FP dataset as equally good as $signal = 1$ datasets. Note, that these FP datasets do not convey any real annotation information, but rather represent artefacts related to GO structure. Only Resnik with method A shows good performance with FPD signal. Altogether, only Smin method shows here reasonably good performance in both tests.

In addition, results clearly show that neither of two tests, ADS or FP datasets, is enough to show all weaknesses all by itself. It rather seems that they show orthogonal views on the evaluated methods, and are both important.

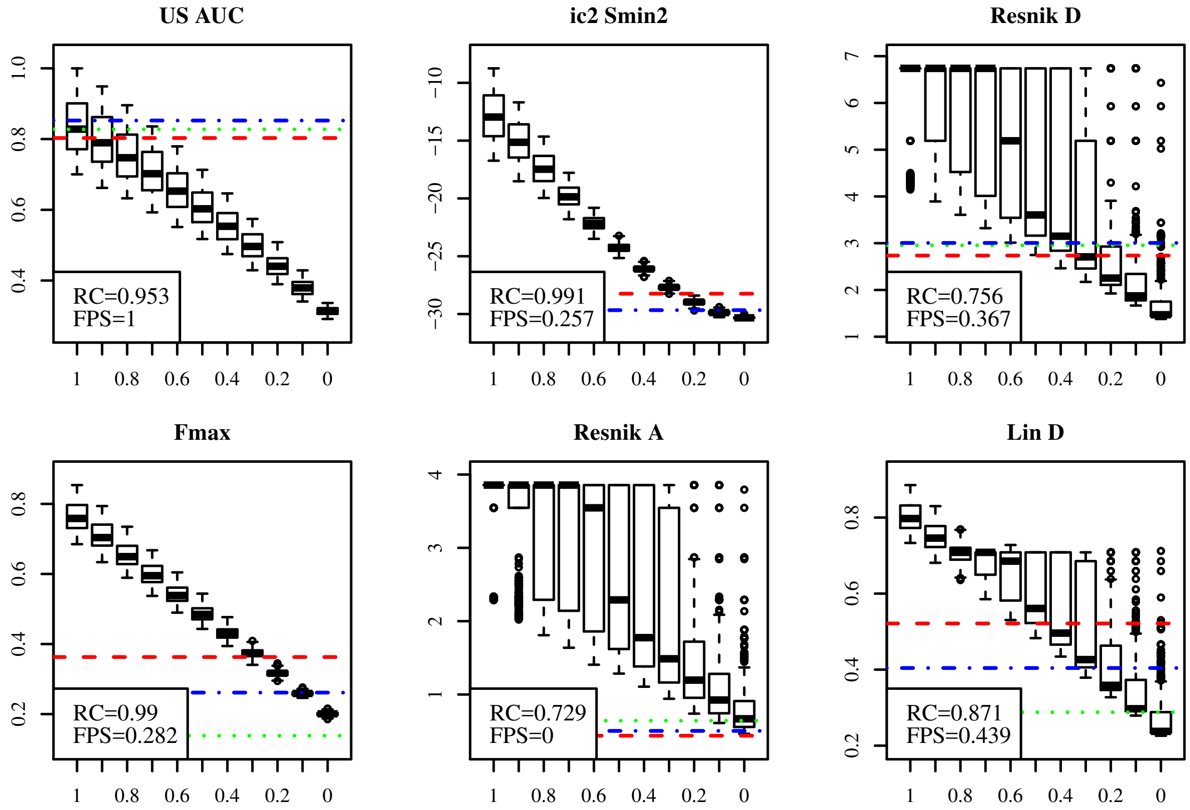


Fig 4. Six popular evaluation metrics compared with ADS Visual analysis of ADS results for six evaluation metrics: US AUC, Fmax, Smin, Resnik score A and D, and Lin score D. Scores for AP (Artificial Prediction) sets at each ADS signal level are plotted as boxplots and scores for FPD (False Positive Data) sets as horizontal lines. RC value shows rank correlation of the AP sets with ADS signals and FPS shows largest signal from FP sets. RC should be high and FPS should be low. Note the drastic differences between methods. Resnik and Lin show weak performance with changing signal range, whereas AUC ranks FP datasets equally good with best signal level. Smin is the only method here that performs reasonably well in both tests. Note that we flipped the sign of Smin results for consistency of the analysis.

ADS confirms known flaws in AUC metric

Next we focus on the performance of AUC methods, using standard ROC AUC (here simply AUC) and AUC under Precision-Recall curve (here AUC-PR). ROC AUC has been used in this field but AUC-PR has not been used. We look at the Unstructured (US), Gene-Centric (GC) and Term-Centric (TC) versions of both AUC methods. We show here results for MouseFunc data.

Standard AUC has been criticised for being a noisy evaluation metric with sensitivity to sample size and class imbalance [7, 23]. Dilution series demonstrates that AUC metrics score somewhat good correlation with the signal level, but fail with FP sets (S3 Table and Fig 5). Surprisingly some of the AUC methods even rank the FP sets as good as $signal = 1$ datasets

(Fig 5, top row). This means that one could not separate a near perfect prediction result from false positive dataset, when using this evaluation metric.

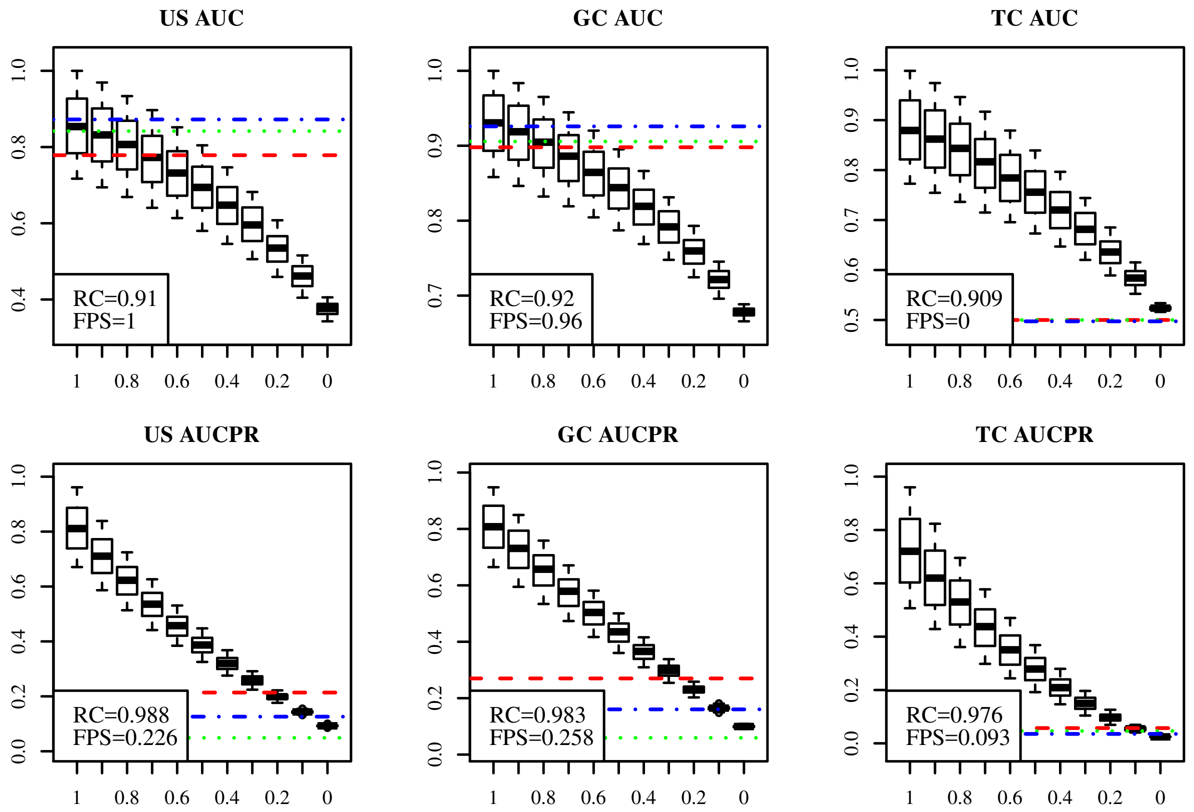


Fig 5. AUC metrics compared with ADS Here we compare standard AUC and AUC-PR (AUC under Precision Recall curve) metric. We test both as unstructured (US), Gene Centric (GC) and Term Centric (TC) versions. Note the very bad performance on FP sets with US AUC and GC AUC. TC AUC-PR shows good performance with both ADS and FP sets. *RC* and *FPS* are explained in previous figure text.

Bad FPS performance of some AUC measures is caused by the identical treatment of small and large classes in US and GC analysis. However, a positive result for large class is more probable from a random dataset. This is worsened by the extreme class size differences in GO data. It should still be noted that this problem was corrected in mouseFunc competition [4] by dividing GO classes to subsets based on class sizes and in CAFA 2 [3] competition by analysing each GO class separately. This later metric is represented here as TC AUC. Indeed, the TC analysis, displayed in the last column, shows clear correction to FP signal. This is seen in all datasets and is in agreement with the theory of the TC method.

When AUC and AUC-PR are compared we see that FP sets get weaker signal with US AUC-PR and GC AUC-PR but the problem still exists. However, in correlation scores AUC-PR metrics consistently outperform AUC metrics. AUC - AUC-PR difference might be explained by problems related to ROC AUC definition (see supplementary S2 Text). So as TC AUC-PR assigns low scores to FP sets and achieves high correlation it is here our recommended method. Furthermore, these results argue strongly against the usage of US

AUC and GC AUC.

ADS reveals drastic differences between semantic summation methods

Here we compare GO semantic similarities, Resnik, Lin and Ancestor Jaccard (AJacc), each coupled with six different semantic summation methods, A-F (see Methods). Results are shown in Fig 6 and S3 Table. Fig 6 is organised so that the columns represent the compared semantic similarities and rows represent summation methods. We look for a combination that delivers good separation in boxplots and ranks FP datasets next to datasets with $signal = 0$.

This comparison shows the power of ADS results. First clear observation is the stronger difference between different summation methods than between compared semantic measures. One can clearly see that *1*) Methods A and C fail at separating signal levels in ADS series and *2*) Methods B and D fail at ranking FP datasets next to the $signal = 0$ datasets. *3*) Only methods E and F show strong performance in both ADS and FP tests. Furthermore, intentionally flawed methods (B and C) were marked as weak either by ADS or FP test.

Second, Resnik shows always weakest performance from the compared semantic similarities. This metric shows frequently the largest scatter in the box plots, when compared to the other semantic similarities. This is in agreement with the fact that Resnik has no upper boundary, whereas Lin and AJacc are bounded from above and below.

Third, the best performance is shown by Lin metric with methods E and F. These achieve high performance here in both ADS and FP tests.

Smin and SimGIC show similar performance with ADS

The third tested group of metrics are the methods that compare *a*) the set of predicted GO classes and their ancestral classes to *b*) the set of correct GO classes and their ancestral classes. Some methods, like SimGIC and SimUI [18], calculate Jaccard correlation between the two sets, whereas Smin calculates a distance between the sets. Most of these metrics include Information Content (IC) weights [18] in their function. We have two alternatives for IC, but limit the results here to IC weights proposed in Smin article [24] (for a discussion about the two versions of information content, ic and $ic2$, please refer to the Supplementary text). We further include gene centric and unstructured versions of these functions:

SimGIC Gene-Centric original version (weighted Jaccard)

SimGIC2 Unstructured modified version of SimGIC

Smin1 Smin original version

Smin2 Gene-Centric version of Smin

GC Jacc Gene-Centric unweighted Jaccard (original SimUI)

US Jacc Unstructured version of SimUI (unstructured Jaccard)

Supplementary text discusses these metrics more in detail.

Results are shown in Fig 7 and S3 Table.We observe that both Smin and SimGIC show high performance in ADS test achieving very similar and nearly perfect correlation to signal

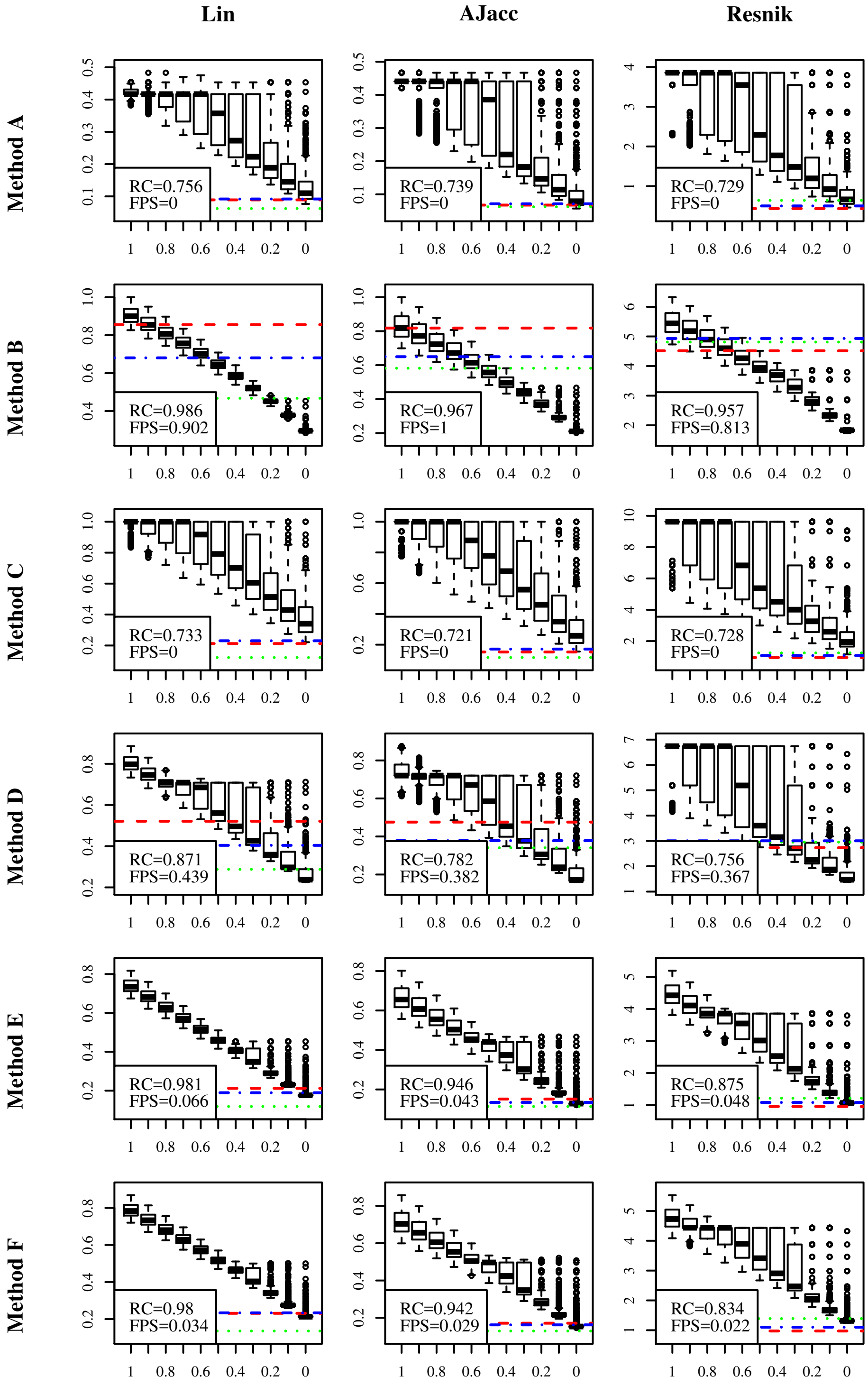


Fig 6. Semantic similarities with different summation methods. Here we represent every combination between three semantics (Resnik, Lin, AJacc), in columns, and six semantic summation methods (A, B, C, D, E, F), in rows. Note that the summation methods affect the performance here more than actual semantic. The novel methods, E and F, outperform the previous standards, A and D.

level. However, in FP test SimGIC shows better performance. Also US Jacc shows quite good performance, but GC Jacc is weaker. Further, we see that introducing gene-centric terms in Smin2 resulted in a notable increase in FP scores in all three data sets. SimGIC2, our unstructured version of SimGIC, showed a minor improvement in correlation scores on all three data sets. In overall the Unstructured versions (Smin1, SimGIC2 and US Jacc) outperformed Gene Centric versions (Smin2, SimGIC, GC Jacc) in correlation values, FPS values or both. Performance of Smin and SimGIC metrics across different datasets was surprisingly similar. So our recommendation is to use Smin1 and SimGIC2. Alternatively, one can use US Jacc, which does not require calculation of information content values and shows quite good performance.

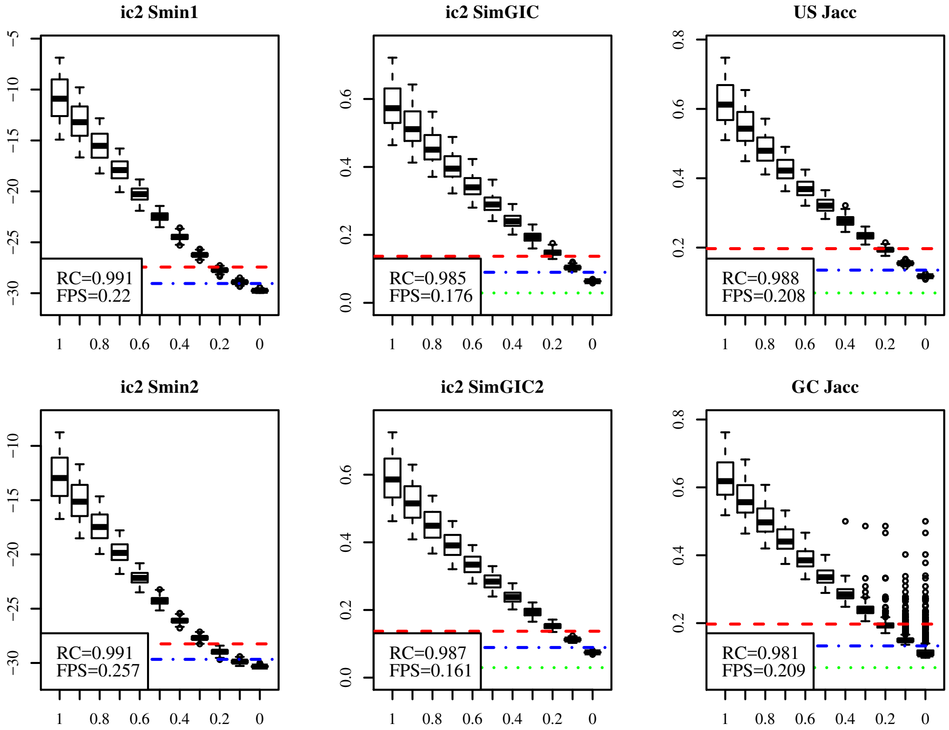


Fig 7. Smin and SimGIC metrics Here we compare two versions of Smin, SimGIC and Jaccard each. Performances are quite similar with the exception of GC Jacc that shows weaker correlation. Smin1 and SimGIC2 are slightly better methods here.

Metric rankings vary between datasets

Next we wanted to compare all the discussed evaluation metrics to see which of them performed best. Fig 8 gives a summary of our results for the UniProt dataset. In this figure we plotted RC against FPS for all metrics tested on the UniProt dataset. Metrics of high quality are expected to appear at the upper left corner of the plot. Metrics that fail with FP tests will be more towards the right. Metrics failing in correlation will be lower in the plot.

Upper panel in fig 8 shows large differences in performances. Especially semantic similarity metrics (shown in blue) show drastic scatter on Y-axis. AUC metrics (red) show

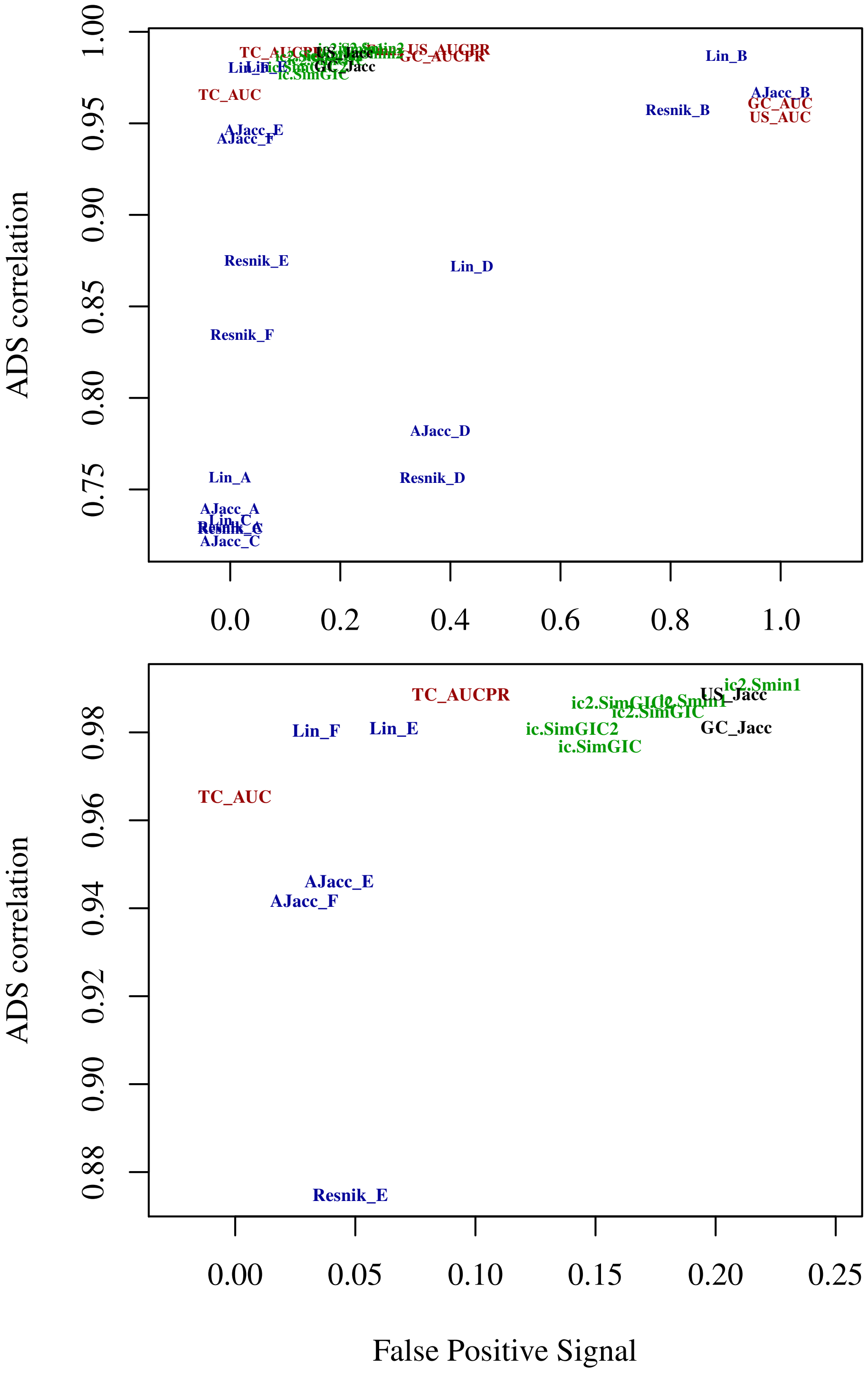


Fig 8. Top metrics for UniProt dataset. US and GC Jacc are plotted in black, Fmax, AUC and AUC-PR in red, Smin and SimGIC in green, semantic similarity metrics in blue. Quality metrics appear at the upper left corner of the plot (high correlation and low False Positive Signal). This part is zoomed in the lower panel.

drastic scatter along X-axis. This is in agreement with the results in earlier chapters. Clearly unstructured and gene-centric Area Under Curve methods (GC AUC, US AUC, US AUC-PR, GC AUC-PR) and semantic similarities based on signal summation methods A, B, C and D should not be used.

Next we look at the metrics that show best performance in our analysis (lower panel in Fig 8). We see good performance here with TC AUC, TC AUC-PR, Lin F, Lin E and SimGIC-Smin cluster (shown in green).

However, when we look at the other two datasets we see a different ranking (Fig. 9). In MouseFunc data we see Semantics (like Lin E and Lin F), simGIC2 and US Jacc as top-performers. Performance of TC AUC-PR and especially TC AUC have gone down. For CAFA dataset we see opposite with TC AUC-PR, TC AUC, US Jacc and simGIC methods showing top-performance. Now semantic methods show clearly weaker performance.

Note that these datasets differed from each other in size and density, with CAFA data being the smallest and sparsest data, UniProt data being intermediate and MouseFunc data being the largest and the most dense dataset. Results suggest that a) Gene Centric semantic methods perform well until the size and sparsity of CAFA data is reached b) Term Centric AUC and AUC-PR perform well except with denser mouse data and c) simGIC methods seem more or less constant. This suggests that selecting the best evaluation metric is actually data-dependent, promoting ADS style tests.

Selecting overall best Evaluation Metrics

Finally we collect the previous results of top-performing methods into Table 1. We show RC (Rank Correlation) and FPS (False Positive Set) values for top performing metrics from each dataset. In addition, we show the same results for selected currently popular methods. Note that although US AUC-PR is not used as such, it corresponds to popular Precision Recall plots, where large area under the curve is what is visually sought.

Again a good method should have high RC values and low FPS values. Therefore, we show in RC tests five highest scores in each column with bold and five lowest scores with underlined italics. In FPS tests we show five lowest scores in bold and five highest scores with underlined italics. So a good metric should have many bold numbers and especially it should not have any underlined italics.

Furthermore we wanted to highlight the consistent good performance of metrics by setting a threshold on RC, t_{RC}, and threshold on FPS, t_{FPS}, and requiring that $RC > t_{RC}$ and $FPS < t_{FPS}$. We highlight the failures of these tests in Table 1 with light red colour. For RC values we observe that top performers would be selected with threshold $t_{RC} = [0.94, 0.98]$. Here we set $t_{RC} = 0.97$. For FPS values we consider that the top performers would be selected with $t_{FPS} = [0.1, 0.25]$. We set $t_{FPS} = 0.15$. We point that these threshold values should be taken with a grain of salt. They can be set differently to emphasise more RC or FPS. However, both tests should be clearly monitored for metric selection.

Top performing metrics

The upper block of Table 1 shows some metrics with encouraging performance. TC AUC-PR shows nice performance with highest ranking RC values. Its FPS values are almost 0.1, but

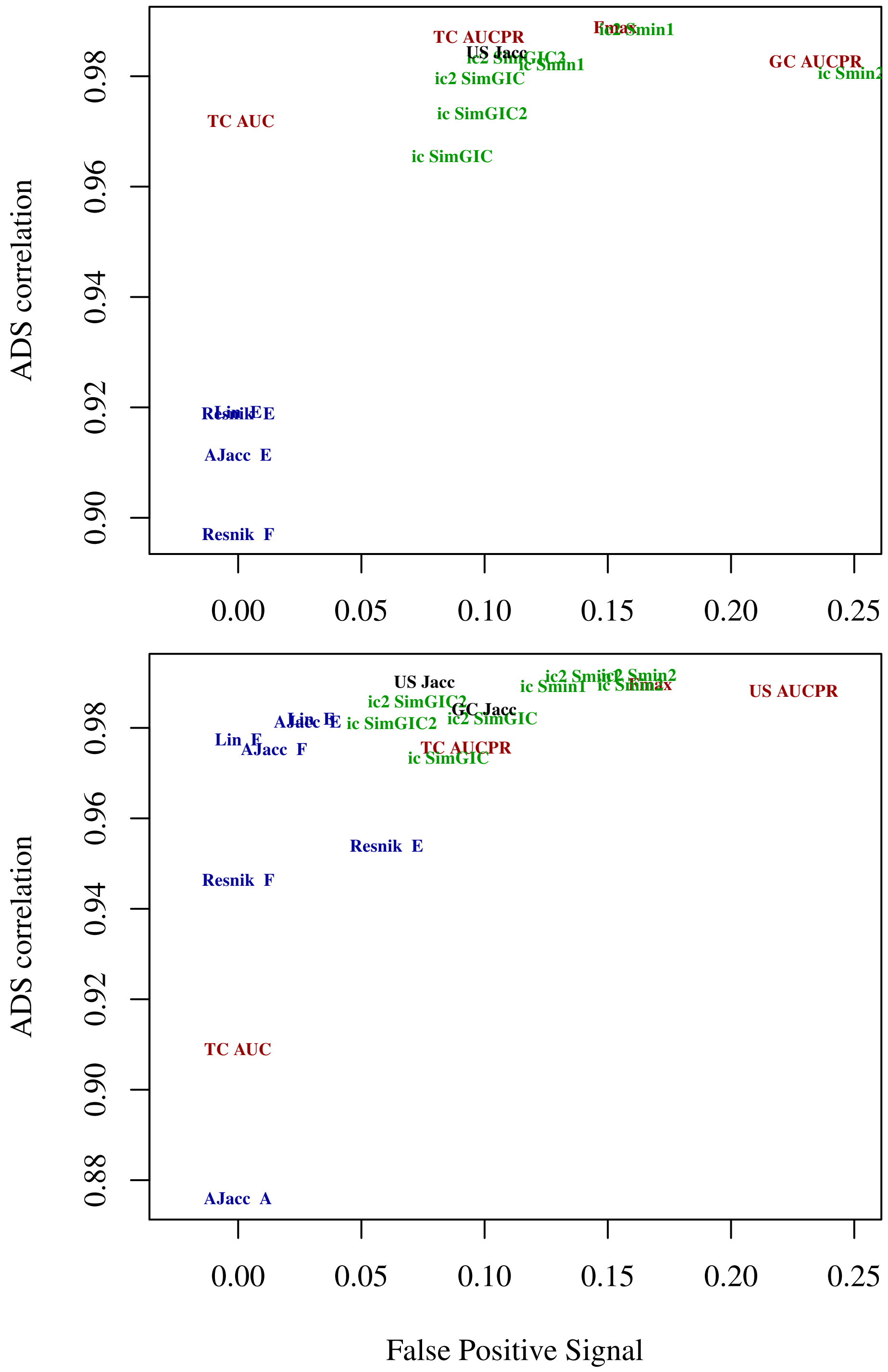


Fig 9. Top metrics for CAFA (top panel) and MouseFunc (lower panel) datasets. Here we repeat the lower visualization in fig 8 on other datasets. Note how the performance of the EvMs varies between the datasets.

we consider this already to be acceptable performance in FPS tests. Therefore we mark it as recommended method. TC AUC shows almost equally good performance with very good FPS results. However, its RC result from mouse data is drastically lower. Still, we mark it as potential recommendation. Lin score with method E shows best consistent performance among the semantics in the upper block. Differences are seen especially in RC values. However, also Lin E fails in RC test on CAFA dataset. Therefore we mark it also only as a potential recommendation. Finally ic2 SimGIC2 and ic SimGIC2 show quite good performance across the tests, with ic SimGIC2 passing through all tests. We mark also ic2 SimGIC2 as recommended method although it fails in FPS test with UniProt data. Note that ic2 SimGIC2 goes only little over the selected threshold (0.15) and in another datasets it stays clearly below t_{FPS}.

We are surprised to see that the ic2 Smin1 [20] has somewhat elevated FPS signal, although it uses *ic* weights to emphasise small classes. It is even more surprising to see that simple US Jacc outperforms ic2 Smin1 in FPS tests. This could be related to internal stabilizing, done in Jaccard correlation. Furthermore, they have quite similar performance in RC tests. Indeed, we mark also US Jacc as potential recommendation, although it fails in FPS test with UniProt data. However, we still point that FPS results for Smin are not drastically bad and that it has the strongest RC results in the table. Finally, as a summary, the best performances were here seen with TC AUC-PR and with two versions of SimGIC2.

Problems with popular metrics

The lower part of Table 1 shows currently popular methods that fail badly. Note that popular Fmax and US AUC-PR show high FPS values and semantic distances show very weak RC values. Although Fmax is preferred for its simplicity, our results show that even the simple US Jacc would be better, as it shows similar RC values but has roughly 30% lower FPS scores. Furthermore, the weighted Jaccards (SimGIC and SimGIC2 metrics) correct FPS error even further. US AUC-PR shows even worse FPS results. Note that again TC AUC-PR would correct this problem while maintaining almost the same level of RC values. In addition the results for semantic similarities in the lower block are very weak, with the weakest RC scores in the comparison.

Overall, these results suggest that TC AUC-PR and two versions of SimGIC2 show consistently best performance. TC AUC, US Jacc and Lin Score E are optional methods that would have to be evaluated on the dataset. In addition, our result discourage the usage of Fmax, US AUC-PR and semantics with summary methods A and D.

Discussion

GO classifiers and other methods for automatic annotations of novel sequences play an important role in modern biosciences. It is thus important to assess the quality of different GO classifiers. This evaluation depends heavily on the used evaluation metrics as they define how methods will be ranked in the evaluation. However, little research has been done on evaluation metrics, although they affect on how the field will develop.

Artificial Dilution Series (ADS) is a novel method to systematically assess how reliably

	Rank Correlation results			False Positive Set results				
	UniProt	cafa2012	mouse	UniProt	cafa2012	mouse	Rec	weakness
TC AUC	0.965	0.972	*0.909*	**0.000**	0.001	**0.000**	(*)	RC in mouse
TC AUC-PR	**0.988**	**0.987**	0.976	0.094	0.098	0.093	*	
Lin score E	0.981	0.919	0.982	0.066	**0.000**	0.030	(*)	RC in CAFA
Lin score F	0.980	*0.833*	0.977	**0.034**	**0.000**	**0.000**		RC in CAFA
AJacc score E	0.946	0.911	0.981	0.043	**0.000**	0.028		weak RC
AJacc score F	*0.942*	*0.837*	0.975	**0.029**	0.004	**0.015**		weak RC
ic2 SimGIC2	0.987	0.983	**0.986**	0.161	0.113	0.073	*	
ic2 SimGIC	0.985	0.980	0.982	0.176	0.098	0.103		FPS elevated
ic SimGIC2	0.981	0.973	0.981	0.140	0.099	0.062	*	
ic SimGIC	0.977	0.966	0.973	0.152	0.087	0.085		FPS elevated
ic2 Smin1	**0.991**	**0.989**	**0.991**	*0.220*	*0.162*	*0.140*		FPS elevated
US Jacc	**0.988**	**0.984**	**0.990**	0.208	0.105	0.076	(*)	FPS elevated
Current popular evaluation metrics								
Fmax	**0.990**	**0.989**	**0.990**	*0.282*	*0.153*	0.167		high FPS
US AUC-PR	**0.990**	**0.987**	**0.988**	*0.398*	*0.287*	*0.226*		high FPS
Lin score A	*0.756*	*0.815*	*0.807*	**0.000**	**0.000**	**0.000**		weak RC
Lin score D	*0.871*	*0.779*	*0.944*	*0.439*	*0.280*	*0.334*		weak in all tests
Resnik score A	*0.729*	*0.829*	*0.779*	**0.000**	**0.000**	**0.000**		weak RC
Resnik score D	*0.756*	0.864	*0.905*	*0.367*	*0.337*	*0.397*		weak in all tests

Table 1. Summary of results for best performing and currently popular metrics. Here we show RC (Rank Correlation) and FPS (False Positive Score) results for best performing methods. We also show same results for some currently popular methods. Good method should have high RC scores and low FPS scores. Rec column shows our selected recommendations (See text for details). Five best results in each column are shown with bold font. Five weakest results in each column are shown with underlined italics. Light red colour highlights cases where metric fails a test (see text for details). Note how methods in lower block show consistent weak performance either in RC or FPS tests.

different evaluation metrics cope with the task of selecting GO classifiers of the highest quality. Thus ADS is a valuable new tool for developing and complementing the existing evaluation methodology. Using ADS we have revealed drastic differences between popular evaluation metrics and metrics that give extremely overoptimistic ratings to false positive annotations.

We demonstrate ADS by testing a large number of existing evaluation metrics and their variations. We were able to show that:

- One should only use Term Centric (TC) AUC methods.
- AUC-PR frequently outperforms ROC AUC.
- Our novel summary methods, E and F, clearly outperform existing semantic similarity summary methods.
- Lin was the best semantic here and Resnik the weakest.

- simGIC and Smin show similar performance but simGIC methods have lower signal on FP tests.
- Among simple evaluation metrics, the unstructured Jaccard correlation outperforms all others, including the popular Fmax metric.

The most consistent good performance is seen with TC AUC-PR, ic2 SimGIC2 and ic SimGIC2. TC AUC shows good performance with two datasets but has weaker RC score with mouseFunc dataset. Lin score with method E shows also good performance on two datasets but has a weaker RC score on CAFA dataset. These latter cases show that selection of best evaluation metrics is dataset dependent. Note that our analysis scripts and ADS are freely available, allowing testing with any GO datasets.

One motivation for the ADS project is the development of better evaluation metrics for GO classifier comparisons. We tested, for example, the following variations on existing metrics:

1. We modified the summary methods for semantic similarities.
2. We included Area Under Precision Recall curve (AUC-PR) besides the Area Under ROC curve.
3. We tested SimGIC function with unstructured calculation.

Here 2, 3 and and some alternatives in 1 improved performance. All these modifications are small and simple to implement to existing evaluation codes. Furthermore, we distribute a separate stand alone C++ program that calculates all the evaluation metrics tested here. This allows end users to apply these metrics in practice.

Our set of compared evaluation metrics included also metrics that were expected to show weaker performance in tests. These were included as negative controls in our analysis to ensure that ADS system truly works. Indeed, we were able to recognize that US AUC and GC AUC show very bad performance with FPS tests. This is in agreement with the results by Ferri et al [7] with AUC, pointing to its weakness with un-even class sizes. Also semantic summation method B showed bad performance with FPS tests and semantic summation method C showed bad performance with ADS RC values. These two methods were expected to show bad performance (see supplementary S2 Text). Our results marked these as very weak methods, pointing that our analysis works.

The importance of a good evaluation metric is best demonstrated by problems of flawed metrics. A metric with sensitivity to our naive FPS signal, EvM_{naive}, would allow ”cheating” in method comparison. For example, when the classifier lacks predictions for a gene G it could gain better result from EvM_{naive} simply by returning a set of naive predictions for G. However, the best alternative in this case should be that the classifier does not return any prediction. Also, a bad EvM_{naive} would evaluate the correct GO class as an equally good prediction as any of its parents. This would promote predicting large unspecific GO classes for all genes, although the small classes would represent more biological information. Likewise, a metric penalising false positives weakly would promote methods to predict as many classes as possible or a metric penalising false negatives weakly would promote predicting as few GO classes as possible. Note that these artefacts are often hard to observe without the comparisons we represent.

ADS is designed for comparison of evaluation metrics that are used in GO class prediction for proteins. However, similar problems with evaluation metrics occur in other areas of biosciences. Complex hierarchical classifications are used for example with diseases (International Classification of Diseases and health related problems, ICD-10) for classifying disease genes. Also the usage of ontologies in biosciences is growing [25], creating more similar classification challenges. Furthermore, similar hierarchical structures are seen outside biosciences (for example in WordNet [26]). Furthermore, evaluation metrics are shown to fail even without hierarchical structures simply when there is a class imbalance problem [16]. Note that most real-life datasets have class imbalance.

The motivation for the ADS project came from our own GO classifier work [5,6], where we noted that standard classifier metrics gave misleading results. Indeed, we expect that the method developing community and researchers doing comparisons between GO prediction methods will benefit most from this work. However, the benefits do not end there. Currently it can be difficult for readers and reviewers to estimate the importance of new predictive bioinformatics methods as different articles can use different metrics and different test datasets. Indeed, the reliability of scientific findings [27,28], over-optimistic results [29] and every method claiming to be the new best method [17] have generated discussion. We hope that the ADS project would improve this situation by developing robust and transparent standards to selection of evaluation metrics to bioinformatics publications.

Conclusion

Science needs various predictive methods, making the comparison and evaluation of these methods an important challenge. The central component of these comparisons is the applied Evaluation Metric. However in the case of Automated Function Predictors the selection of the evaluation metric is a challenging task, potentially favoring wrong methods. In this article we have described Artificial Dilution Series (ADS) which is, to our knowledge, the first method for testing various classifier evaluation metrics using real datasets as a platform for controlled amounts of embedded signal. ADS allows a simple testing of evaluation metrics to see how easily they can separate different signal levels. Furthermore we do testing with different False Positive Sets that represent various datasets that the evaluation metric might be fooled to consider as meaningful. Our results show clear performance differences between compared evaluation metrics. We also test some modifications of existing evaluation metrics and show that some of them improve the performance. This work provides a platform for selecting evaluation metrics for specific Gene Ontology datasets. Also this platform can considerably simplify the development of novel evaluation metrics. Furthermore, the same principle should be applicable to other datasets inside and outside of biosciences.

Acknowledgements

IP was supported by Emil Aaltonen Foundation and University of Helsinki. PT was supported by Helsinki Institute of Life Science (HiLife). We would like to thank Rishi Das Roy and especially Alan Medlar for valuable comments and fruitful discussions.

Supporting Information

S1 Text ADS pseudocode Pseudocode for generations of ADS.

S2 Text Supplementary text Text shows an example on how the signal and noise is defined in ADS. Text also shows labels, definitions and discussion of tested metrics. Finally, we discuss the summary methods for semantic similarities and motivations for our novel summary methods.

S1 File Metric scores Metric scores for all generated AP and FP sets ($k = 2$).

S2 File Metric scores Metric scores for all generated AP and FP sets ($k = 4$).

S1 Table Abbreviations and features for all compared EvMs We represent a summary table of all the compared EvMs. Table shows the used abbreviation, core function, used data summary method, used threshold function over the classifier prediction score, used IC weighting and summary method for semantic similarities. In addition we mark the EvMs that have been popular in AFP evaluation, ones that are partially novel and ones that we consider to be simple. We also mark EvMs that we expect to perform badly as negative controls. More detailed description of these EvMs is in suppl. text S2 Text.

S2 Table Features of the three used datasets Table explains the size and density of the three used datasets (CAFA1, MouseFunc, UniProt). Datasets are explained in main text. Table shows skewed distribution of GO classes per gene by representing selected quantiles from this distribution. This is shown separately for each dataset with and without the propagation to parent classes. We also show the same size distribution for the GO classes, again showing the skewness of the size distribution.

S3 Table ADS results RC and FPS scores for tested metrics ($k = 2$).

S4 Table ADS results RC and FPS scores for tested metrics ($k = 4$).

References

1. Friedberg I. Automated protein function prediction—the genomic challenge. Briefings in bioinformatics. 2006;7(3):225–242.

2. Radivojac P, Clark WT, Oron TR, Schnoes AM, Wittkop T, Sokolov A, et al. A large-scale evaluation of computational protein function prediction. Nature methods. 2013;10(3):221–227.

3. Jiang Y, Oron TR, Clark WT, Bankapur AR, D'Andrea D, Lepore R, et al. An expanded evaluation of protein function prediction methods shows an improvement in accuracy. Genome biology. 2016;17(1):184.

4. Peña-Castillo L, Tasan M, Myers CL, Lee H, Joshi T, Zhang C, et al. A critical assessment of Mus musculus gene function prediction using integrated genomic evidence. Genome biology. 2008;9(1):S2.

5. Koskinen P, Törönen P, Nokso-Koivisto J, Holm L. PANNZER: high-throughput functional annotation of uncharacterized proteins in an error-prone environment. Bioinformatics. 2015;31(10):1544–1552.

6. Törönen P, Medlar A, Holm L. PANNZER2: a rapid functional annotation web server. Nucleic Acids Research. 2018;46(W1):W84–W88. doi:10.1093/nar/gky350.

7. Ferri C, Hernández-Orallo J, Modroiu R. An experimental comparison of performance measures for classification. Pattern Recognition Letters. 2009;30(1):27–38.

8. Hand DJ. Assessing the performance of classification methods. International Statistical Review. 2012;80(3):400–414.

9. Japkowicz N, Shah M. Evaluating learning algorithms: a classification perspective. Cambridge University Press; 2011.

10. Gillis J, Pavlidis P. Characterizing the state of the art in the computational assignment of gene function: lessons from the first critical assessment of functional annotation (CAFA). In: BMC bioinformatics. vol. 14. BioMed Central; 2013. p. S15.

11. Kahanda I, Funk CS, Ullah F, Verspoor KM, Ben-Hur A. A close look at protein function prediction evaluation protocols. GigaScience. 2015;4(1):41.

12. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, et al. Gene Ontology: tool for the unification of biology. Nature genetics. 2000;25(1):25–29.

13. Eisner R, Poulin B, Szafron D, Lu P, Greiner R. Improving Protein Function Prediction using the Hierarchical Structure of the Gene Ontology. In: 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. Institute of Electrical and Electronics Engineers (IEEE); 2005.Available from: `https://doi.org/10.1109%2Fcibcb.2005.1594940`.

14. Fu G, Wang J, Yang B, Yu G. NegGOA: negative GO annotations selection using ontology structure. Bioinformatics. 2016;32(19):2996–3004.

15. Valverde-Albacete FJ, Peláez-Moreno C. Two information-theoretic tools to assess the performance of multi-class classifiers. Pattern Recognition Letters. 2010;31(12):1665–1671.

16. Valverde-Albacete FJ, Peláez-Moreno C. 100% classification accuracy considered harmful: The normalized information transfer factor explains the accuracy paradox. PloS one. 2014;9(1):e84217.

17. Norel R, Rice JJ, Stolovitzky G. The self-assessment trap: can we all be better than average? Molecular systems biology. 2011;7(1):537.

18. Pesquita C, Faria D, Falcao AO, Lord P, Couto FM. Semantic similarity in biomedical ontologies. PLoS comput biol. 2009;5(7):e1000443.

19. Pesquita C, Faria D, Bastos H, Falcao A, Couto F. Evaluating GO-based semantic similarity measures. In: Proc. 10th Annual Bio-Ontologies Meeting. vol. 37; 2007. p. 38.

20. Clark WT, Radivojac P. Information-theoretic evaluation of predicted ontological annotations. Bioinformatics. 2013;29(13):i53–i61.

21. Lin D, et al. An information-theoretic definition of similarity. In: Icml. vol. 98. Citeseer; 1998. p. 296–304.

22. Resnik P, et al. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. J Artif Intell Res(JAIR). 1999;11:95–130.

23. Hanczar B, Hua J, Sima C, Weinstein J, Bittner M, Dougherty ER. Small-sample precision of ROC-related estimates. Bioinformatics. 2010;26(6):822–830.

24. Clark WT, Radivojac P. Information-theoretic evaluation of predicted ontological annotations. Bioinformatics. 2013;29(13):i53–i61.

25. Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nature biotechnology. 2007;25(11):1251.

26. Fellbaum C. WordNet: An Electronic Lexical Database. Bradford Books; 1998.

27. Ioannidis JP. Why most published research findings are false. PLoS medicine. 2005;2(8):e124.

28. Ioannidis JP. How to make more published research true. PLoS medicine. 2014;11(10):e1001747.

29. Boulesteix AL. Over-optimism in bioinformatics research. Bioinformatics. 2010;26(3):437–439.