An Introduction to the SemDist package
Version 0.99.0

Contents

1 Overview and Background

The methods in this package assess the performance of a predictor of multi-label annotations, such that labels are drawn from an ontology structured as a directed acyclic graph (DAG). The package was designed to work specifically with the Gene Ontology (GO) [Ashburner et al., 2000], but functionality has been embedded that will allow users to define their own ontology in addition to probability distributions associated with "entities" in the given ontology. The basic inputs to the package are sets of true GO terms associated with each data-point (be it a gene or protein), and sets of predicted annotations with associated confidence values in the interval $[0, 1]$. The SemDist package facilitates the calculation of several statistics relating to the performance of a given predictor, all of which utilize the concept of information accretion as their foundation [Clark and Radivojac, 2013].

1.1 Metric definitions

Briefly, the concept of information accretion arises from interpreting an ontology as a Bayesian network. In this framework, the joint probability of a set of "states" in the network (in this case true or false values associated to terms) is calculated by considering the conditional probability of each individual variable given its parents.

Very simply, information accretion is calculated by considering the conditional probability of a term given its parents,

$$ia(v) = \frac{1}{Pr(v|\mathcal{P}(v))}, \tag{1}$$

where $\mathcal{P}(v)$ refers to the parents of vertex v. Using information accretion the information content of a set of terms T, representing a consistent subgraph from the overall DAG, can then be calculated as

$$i(T) = \sum_{v \in T} ia(v). \tag{2}$$

Although the SemDist package calculates several information accretion based metrics users might be more familiar with such as precision, recall, and specificity; the hallmarks of the package are the previously introduced metrics misinformation and remaining uncertainty.

Specifically, the *remaining uncertainty* about a protein's true annotation corresponds to the information about the protein that is not yet provided by the graph P. More formally, we express remaining uncertainty (ru) as

$$ru(T, P) = \sum_{v \in T \setminus P} ia(v), \tag{3}$$

which is simply the total information content of the nodes in the ontology that are contained in true annotation T but not in the predicted annotation P.

Note that, in a slight abuse of notation, we apply set operations to graphs to manipulate only the vertices of these graphs.

The *misinformation* introduced by the classifier corresponds to the total information content of the nodes along incorrect paths in the prediction graph P. More formally, misinformation is expressed as

$$mi(T, P) = \sum_{v \in P \setminus T} ia(v), \tag{4}$$

which quantifies how misleading a predicted annotation is.

For a more thorough, formal treatment of the theory behind these metrics, see Clark and Radivojac [2013] and Clark [2014].

Equations (3) and (4) illustrate calculating remaining uncertainty and misinformation in a very simple case where there is a single data point and no probabilities associated with each predicted term. SemDist performs calculations by averaging over the entire set of proteins used in evaluation, i.e.

$$ru(\tau) = \frac{1}{n} \sum_{i=1}^{n} ru(T_i, P_i(\tau)) \tag{5}$$

and

$$mi(\tau) = \frac{1}{n} \sum_{i=1}^{n} mi(T_i, P_i(\tau)) \tag{6}$$

where n is the number of proteins in the data set, T_i is the true set of terms for protein x_i, and $P_i(\tau)$ is the set of predicted terms for protein x_i given decision threshold τ.

Once the set of terms with scores greater than or equal to τ is determined, the set $P_i(\tau)$ is composed of the unique union of the ancestors of all predicted terms. As the decision threshold is moved from its minimum (0) to its maximum value (1), the pairs of $(ru(\tau), mi(\tau))$ will result in a curve in 2D space. We refer to such a curve using $(ru(\tau), mi(\tau))_\tau$. Removing the normalizing constant $(\frac{1}{n})$ from the above equations would result in the total remaining uncertainty and misinformation associated with a database of proteins and a set of predictions.

Finally, to provide a single performance measure which can be used to rank and evaluate protein function prediction algorithms, we introduced *semantic distance* as the minimum distance from the origin to the curve $(ru(\tau), mi(\tau))_\tau$. More formally, the semantic distance S_k is defined as

$$S_k = \min_{\tau} (ru^k(\tau) + mi^k(\tau))^{\frac{1}{k}}, \tag{7}$$

where k is a positive integer. Setting $k = 2$ results in the minimum Euclidean distance on the curve from the origin.

We have also generalize the definitions of precision and recall into their information formulation. Here, precision and recall can be expressed as

$$pr(T, P(\tau)) = \frac{\sum_{v \in T \cap P(\tau)} ia(v)}{\sum_{v \in P(\tau)} ia(v)} \tag{8}$$

and

$$rc(T, P(\tau)) = \frac{\sum_{v \in T \cap P(\tau)} ia(v)}{\sum_{v \in T} ia(v)}. \tag{9}$$

Precision $pr(\tau)$ and recall $rc(\tau)$ can then be calculated as averages over the database of proteins for each threshold τ, as in Equations Equations (5) and (6).

Our framework also facilitates calculating specificity

$$sp(T, P(\tau)) = \frac{\sum_{v \in T^{\mathbf{c}} \cap P^{\mathbf{c}}(\tau)} ia(v)}{\sum_{v \in T^{\mathbf{c}}} ia(v)} \tag{10}$$

where $T^{\mathbf{c}}$ represents the complement of set T. Combining specificity and recall (sensitivity) facilitate plotting an ROC curve and calculating AUROC.

Finally, we facilitate the calculation of semantic similarity,

$$ss(T, P(\tau)) = \sum_{v \in T \cap P(\tau)} ia(v), \tag{11}$$

a formulation similar to that defined by Resnik [1995] and Lord et al. [2003].

This package allows for straightforward calculation of these metrics. It provides a set of methods for calculating the information accretion for each term in a given ontology in addition to providing pre-calculated values for a set of organisms.

2 Installation and Requirements

Install the SemDist package on your version of R with the biocLite and library functions. The package requires R version 2.10.0 or greater and Bioconductor version 2.14 or greater. The annotate, AnnotationDbi, and GO.db packages are also required for functionality.

```
> biocLite("SemDist")
> library(SemDist)
```

3 Calculating information accretion values

The simplest way to use SemDist is to use one of the provided data sets containing information accretion values. Information accretion values were pre-

calculated for each organism using Bioconductor AnnotationData Packages [Bioconductor, 2014]. These pre-calculated values can be found in the `data/` directory of the SemDist package. (Note that in addition, all other test files used in the following examples can be accessed by the user in the `extdata/` directory). Calculations for human protein annotations were performed using only annotations supported by experimental evidence codes (all other species data were calculated with all available annotations).

Information accretion values can be recalculated by the user or calculated using different evidence codes using the `computeIA` function.

```
> # Examine built-in IA data
> data("Info_Accretion_mouse_CC", package="SemDist")
> head(IAccr)

GO:0000015 GO:0000109 GO:0000110 GO:0000118 GO:0000120 GO:0000123
  4.569856   6.111738   2.700440   2.975517   0.000000   2.409170

> # Calculate IA, specify evidence codes
> # Requires downloading annotation data package
> biocLite("org.Hs.eg.db")
> IAccr <- computeIA("MF", "human", evcodes=c("EXP", "IC"))
```

Users can define their own ontology and annotations. The `computeIA` function can then calculate information accretion for each term, or entity, in the custom ontology according to Equation (1).

Ontologies should be specified using a tab-delimited file indicating every parent-child relationship in the ontology; with the first column designating the parent name and the second column identifying the child name. The annotations file should have the same format as the true annotations file used above where annotations represent terms drawn from the specified ontology.

```
> # Calculate IA, specify ontology and annotations
> ontfile <- system.file("extdata", "mfo_ontology.txt",
+                        package="SemDist")
> annotations <- system.file("extdata", "MFO_LABELS_TEST.txt",
+                            package="SemDist")
> IAccr <- computeIA("my", "values", specify.ont=TRUE,
+                 myont=ontfile, specify.annotations=TRUE,
+                 annotfile=annotations)
```

It is also possible to specify your own ontology and use Bioconductor AnnotationData annotations, although it should be pointed out that using a version of the ontology other than that associated with the AnnotationData annotations may result in terms not being found.

```
> # Calculate IA, specify ontology
> IAccr <- computeIA("MF", "human", specify.ont=TRUE,
+          myont=ontfile, specify.annotations=FALSE)
```

When the ontology is specified but annotations are not, annotations are taken from the specified organism. (Note that this may require the installation of databases not provided in the package). The recommended method is to specify both, as is the case in the second example.

The information accretion data generated in this way can be used in the `findRUMI` and `RUMIcurve` functions (which will be discussed below).

4 Getting Started with RU-MI Calculations

Before any of the evaluation metrics in this package can be utilized, two user-provided data sets are needed in the working directory:

1. A file containing true annotations of the protein sequences being looked at.
2. A file containing the annotations predicted by the function predictor along with a $[0-1]$ probability assigned to each annotation.

These files should take the form of tab-delineated text, with the first column being sequence identifiers, the second column being GO identifiers (of the form "GO:0005615"), and the third column (in the case of the prediction file) being a valid number in the interval $[0, 1]$.

```
> # Sample true, prediction files included with SemDist:
> truefile <- system.file("extdata", "MFO_LABELS_TEST.txt",
+                         package="SemDist")
> predfile <- system.file("extdata", "MFO_PREDS_TEST.txt",
+                         package="SemDist")
> predictions <- read.table(predfile, colClasses = "character")
> head(predictions)
```

```
         V1         V2    V3
1 IN04_YEAST GO:0003674 0.464
2 IN04_YEAST GO:0030528 0.464
3 IN04_YEAST GO:0016564 0.441
4 IN04_YEAST GO:0016563 0.464
5 IN04_YEAST GO:0030234 0.400
6 IN04_YEAST GO:0060589 0.400
```

Given these basic data sets, remaining uncertainty and misinformation can be calculated according to Equation (3) and Equation (4) using the `findRUMI` function in addition to the package's built in data sets.

As a simple example we show how one can calculate remaining uncertainty and misinformation for a single threshold ($\tau = 0.75$),

```
> rumiTable <- findRUMI("MF", "human", 0.75, truefile, predfile)
> avgRU <- mean(rumiTable$RU)
> avgMI <- mean(rumiTable$MI)
```

where information accretion data from the molecular function portion of the ontology terms annotating human sequences were used (denoted `"MF"` and `"human"` in the function call).

The returned `rumiTable` variable should contains a data frame with the remaining uncertainty values for each sequence in the first column and the corresponding misinformation values in the second column. Single-number metrics can be obtained by averaging across the entire set.

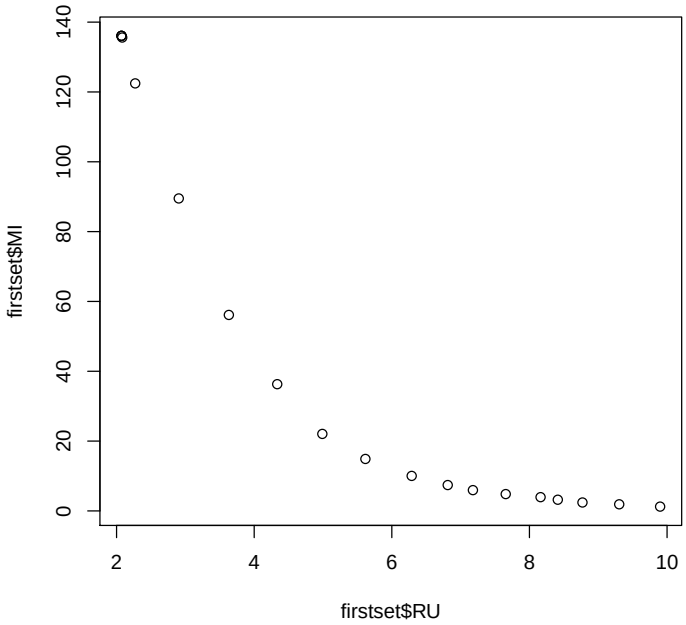
As mentioned in the above section, user-created IA variables from `computeIA` output can be used to specify the information accretion values used in the calculation of `RU` and `MI` via the `IAccr` argument:

```
> # A sample IA file that comes with the SemDist package.
> myIA <- system.file("extdata", "myIA.rda", package="SemDist")
> load(myIA)
> # "MF" and "human" are present only for naming purposes
> # since IA is taken from myIA.rda
> rumiTable <- findRUMI("MF", "human", 0.75, truefile, predfile,
+                           IAccr = IA)
```

5 Creating a RU-MI Curve

The average values of remaining uncertainty and misinformation across multiple thresholds in the interval $[0, 1]$ as specified by Equations (5) and (6) can be calculated. Using the same true/prediction files and specifying a set increment of 0.05 which τ is increased

```
> avgRUMIvals <- RUMIcurve("MF", "human", 0.05, truefile, predfile)
> firstset <- avgRUMIvals[[1]]
> plot(firstset$RU, firstset$MI)
```



The `RUMIcurve` function will accept multiple prediction files (as a character vector) as input, and actually returns a list containing the values calculated for each set of predictions; hence the need for indexing into the first element to get the curve data. The `firstset` variable contains a data frame with the average RU and MI values for each threshold the function calculated for. In this case, the increment was set to 0.05.

The plot visualizes how remaining uncertainty and misinformation change as the decision threshold is varried. If multiple predictions are input they can be quickly compared.

Semantic distance can be obtained in a straightforward manner from the points in the RU-MI curve according to Equation (7):

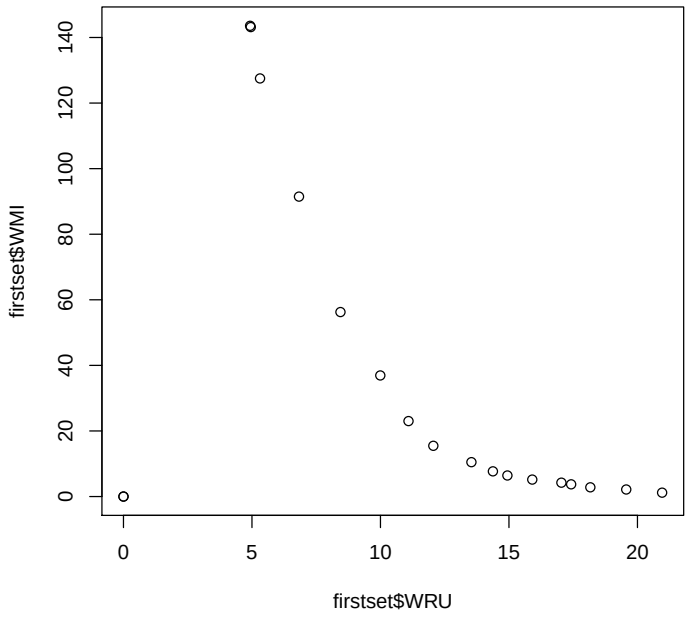
```
> # Minimize distance from origin over all points in RUMI curve:
>
> semdist <- min(sqrt(firstset$MI^2 + firstset$RU^2))
> semdist

[1] 9.004228
```

6 Additional metrics

Related metrics can also be output by the `RUMIcurve` function. The data frames that are returned also contain an `"SS"` column which contains the semantic

similarity for a given threshold calculated as specified by Equation (11). Setting the boolean parameter `add.prec.rec` to TRUE will cause precision, recall, and specificity values to be included as well. Doing the same for `add.weighted` will output weighted remaining uncertainty, weighted misinformation, and weighted semantic similarity. The weighted metrics assign a weight to each RU or MI value for all the sequences based on how much of the total information accretion that sequence accounts. See Equation (8), Equation (9), and Equation (10) for the formal definitions of these calculations.

```
> rumiout <- RUMIcurve("MF", "human", 0.05, truefile, predfile,
+                      add.prec.rec=TRUE, add.weighted=TRUE)
> firstset <- rumiout[[1]]
> plot(firstset$WRU, firstset$WMI)
```



7 Acknowledgments

The code in the SemDist package was based in part on the source code of the GOSemSim package [Yu et al., 2010].

References

Michael Ashburner et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature Genetics*, 25(1):25–29, 2000.

Bioconductor. AnnotationData Packages, 2014. URL `http://master.bioconductor.org/packages/release/data/annotation/`.

Wyatt Travis Clark. *Information-Theoretic Evaluation for Computational Biomedical Ontologies*. Springer International Publishing, 2014.

Wyatt Travis Clark and Predrag Radivojac. Information-theoretic evaluation of predicted ontological annotations. *Bioinformatics*, 29(13):i53–i61, 2013.

Phillip W. Lord, Robert D. Stevens, Andy Brass, and Carole A. Goble. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283, 2003.

Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

Guangchuang Yu, Fei Li, Yide Qin, Xiaochen Bo, Yibo Wu, and Shengqi Wang. GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics*, 26(7):976–978, 2010.