Improving X!Tandem on Peptide Identification From Mass Spectrometry by Self-boosted Percolator

Pengyi Yang, Jie Ma, Penghao Wang, Yunping Zhu, Bing B. Zhou, and Yee Hwa Yang

Abstract—A critical component in mass spectrometry (MS)-based proteomics is an accurate protein identification procedure. Database search algorithms commonly generate a list of peptide-spectrum matches (PSMs). The validity of these PSMs is critical for downstream analysis since proteins that are present in the sample are inferred from those PSMs. A variety of post-processing algorithms have been proposed to validate and filter PSMs. Among them, the most popular ones include a semi-supervised learning (SSL) approach known as Percolator and an empirical modeling approach known as PeptideProphet. However, they are predominantly designed for commercial database search algorithms i.e. SEQUEST and MASCOT. Therefore, it is highly desirable to extend and optimize those PSM post-processing algorithms for open source database search algorithms such as X!Tandem. In this study, we propose a Self-boosted Percolator for post-processing X!Tandem search results. We find that the SSL algorithm utilized by Percolator depends heavily on the initial ranking of PSMs. Starting with a poor PSM ranking list may cause Percolator to perform suboptimally. By implementing Percolator in a cascade learning manner, we can progressively improve the performance through multiple boost runs, enabling many more PSM identifications without sacrificing false discovery rate (FDR).

Index Terms—Proteomics, Mass spectrometry, Percolator, X!Tandem, Peptide-spectrum match (PSM), Peptide identification, Semi-supervised learning.

1 Introduction

Mass spectrometry (MS)-based high-throughput proteomics studies aim to identify the entire proteome present in a cell, tissue or organism at a specific time or condition. Currently, tandem MS-based technologies are the preferred method in proteomics as relatively high sensitivity and specificity can be achieved [1]. One computational challenge is to infer peptides from the spectra produced by the mass spectrometer. There are three main approaches for peptide identification; the database search approach [2], the spectral library search approach [3], [4], and the *de novo* sequencing approach [5]. The *de novo* sequencing approach is often only applicable to very high precision mass spectrometry [5] and the remaining two approaches are more common. The library search approach relies on the initial results from the database search, and the *de novo* sequencing approach can benefit from incorporating database search results [6]. Thus, improvement on the database search approach will also enhance the library search approach and the *de novo* sequencing approach. This suggests that it is important that our initial focus for improving peptide identification results is to concentrate on achieving better and more efficient database search results.

In the database search approach, a search algorithm is applied to produce a list of peptide-spectrum matches (PSMs), in which the peptides and proteins are inferred. Popular database search algorithms include SEQUEST [2], MASCOT [7], X!Tandem [8], OMSSA [9], and Paragon [10]. Several studies have reviewed and compared their performance on different datasets [11], [12].

All these algorithms involve comparing observed spectra to a list of theoretical enzymatic digested peptides from a specified protein database. The comparison is based on a “search score” measuring the degree of similarity between the observed spectra to a theoretical spectrum generated from enzymatic digested peptide. Each pair of observed spectra and a theoretical peptide is known as a peptide-spectrum match (PSM). Each PSM is assigned a search score, and different algorithms vary in their definition of the score. For example, SEQUEST calculates an Xcorr score for each PSM by evaluating the correlation between the experimental spectrum and the theoretically constructed spectrum from the database [2];

- *P. Yang and B. Zhou are with School of Information Technologies, University of Sydney, NSW 2006, Australia.*
- *J. Ma and Y. Zhu are with State Key Laboratory of Proteomics, Beijing Proteome Research Center, Beijing Institute of Radiation Medicine, Beijing 102206, China.*
- *Y. Yang and P. Wang are with School of Statistics and Mathematics, University of Sydney, NSW 2006, Australia.*
- *Corresponding to:*
  E-mail: jean.yang@sydney.edu.au
  E-mail: bing.zhou@sydney.edu.au

X!Tandem [8] counts the number of matched ions (also refer to as peaks) and then calculates a score using the matched ions and their intensities.

Each search score is an indication of the quality of match between the theoretical peptides and the observed spectra. One typically expects that the higher the score, the more likely that the PSM is a correct match, that is, the observed spectrum is correctly identified as the corresponding peptide of the PSM. Due to the varying quality of the spectra, the characteristics of the search algorithm and scoring metrics, and the incompleteness of the protein database, typically, only a fraction of the PSMs are correct [13]. Moreover, the search scores are often not directly interpretable in terms of statistical significance [14]. Therefore, it is necessary to determine a critical value above which ranking scores are to be considered significant. This filtering process is also seen as an independent validation of the PSM and thus the whole process is often known as PSM post-processing.

For PSM post-processing, algorithms such as PeptideProphet [15] and Percolator [16] are probably the most popular ones. PeptideProphet utilizes a linear discriminant analysis (LDA) model to score PSMs and fits an expectation maximization (EM) model from which a posterior probability for each PSM being a correct peptide identification is generated. Percolator uses a semi-supervised learning (SSL) algorithm for training a support vector machine (SVM) iteratively. The training data is filtered subsequently with a predefined false discovery rate (FDR) threshold, and the SVM model from the last iteration is used for classifying PSMs.

Both Percolator and PeptideProphet were originally designed for SEQUEST [15], [16]. Recent extensions to PeptideProphet include the incorporation of more flexible models (e.g. variable component mixture model) [17] and other database search algorithms [18]. In comparison, the extensions of Percolator include a wrapper interface for MASCOT [19], and the reformulation of the learning algorithm [20].

While these validation and filtering algorithms have been found to be very useful, they are predominantly designed for commercial database search algorithms i.e. SEQUEST and MASCOT. So far, there has been no extension of Percolator for open source search algorithms such as X!Tandem. Therefore, it is highly desirable to extend and optimize these PSM *post-processing* algorithms for open source algorithms, given their increasing popularity in the proteomics community [18].

In this study, we propose a Self-boosted Percolator for post-processing X!Tandem search results. We discover that the current Percolator algorithm relies heavily on decoy PSMs and their rankings in the initial PSM list [19]. The iterative FDR filtering of PSMs is the key to enhance the discriminant ability of the final SVM model. If the decoy PSMs are poorly ranked in the initial PSM list, the performance of the algorithm may degrade, resulting in a suboptimal SVM model and reduced PSM classification accuracy. One potential solution could be to apply the SVM model from Percolator to re-rank the PSM list and rerun Percolator on the re-ranked PSM list. By repeating the learning and re-ranking process a few times, the algorithm "boosts" itself to a stable state, reducing the noise in the initial ranking to the minimum and generating better PSM post-processing results. For overfitting assessment, we extend the *nonsense database* search approach proposed by Bern and Kil [21], from which a comparison of identifications from target and decoy databases can be objectively conducted. This is especially useful for benchmarking post-processing algorithms that utilizing decoy database explicitly in scoring PSMs such as the semi-supervised learning procedure used by Percolator and the proposed Self-boosted Percolator.

In summary, our contributions are (1) extending Percolator for X!Tandem, (2) identifying a potential inefficiency in Percolator learning process, (3) implementing a cascade learning procedure to boost the performance of Percolator, and (4) introducing a nonsense database search approach for objective benchmarking on algorithms that using information in decoy database explicitly. Our experiments on two benchmark datasets generated from complex samples show substantial improvement on X!Tandem search results, enabling many more PSM identifications without sacrificing FDR.

2 Materials and Methods

2.1 Evaluation Datasets

Two benchmark datasets that were utilized to assess Percolator in its original study [16] were included in this study for method evaluation. They were denoted as the **Yeast** dataset and the **Worm** dataset according to the organisms from which the samples were derived (refer to Supplement of [16] for details). Specifically, we utilized the datasets generated from trypsin digestion. The corresponding target databases were obtained from the authors (http://noble.gs.washington.edu/proj/percolator/) and the decoy databases were built by reversing the sequences in the target databases, respectively.

2.2 Database Searching

We used the concatenated target-decoy database search approach, in which the reverse protein sequences were combined with the target database [22]. The estimated false discovery rate (FDR) was calculated as follows:

$$\mathrm{FDR} = 2 \times \frac{N_{\mathrm{D}}}{N_{\mathrm{D}} + N_{\mathrm{T}}} \qquad (1)$$

where N_D and N_T are the number of decoy and target matches from the concatenated database, respectively, which pass the predetermined filtering threshold. The q-value is defined as the minimal FDR at which a PSM is accepted.

A key aspect of post-processing algorithms is their ability in controlling overfitting. Here, we extended the nonsense database search approach used in [21] for benchmarking false positive identifications. Specifically, we created a nonsense database from the target database using the procedure described in [21] and combined the target and the nonsense databases. The decoy database was then created by reversing the combined database which contains both target and nonsense protein sequences (Figure 1). The level of overfitting of a given algorithm was assessed by comparing the number of reported PSMs in nonsense group and the reverse of nonsense group at different cutoff, as ideally they should be approximately the same. This assessment ensures that the semi-supervised learning procedure utilized by Percolator and Self-boosted Percolator does not bias to reducing only PSMs from reverse database as this will cause an unrealistic under-estimation on number of false positive PSMs in target database.

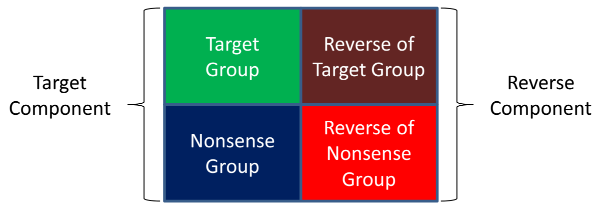


Fig. 1. A modified target-decoy approach with a paired nonsense database. The search database comprises of four groups including the original target database (denoted as target group), a nonsense group derived from the target database (nonsense group), the reverse database of the target database (reverse of target group), and the reverse database of the nonsense database (reverse of nonsense group). The combination of the target group and the nonsense group is treated as the target database by a given post-processing algorithm while the other two groups together form the reverse database.

Raw spectra files were searched against the corresponding concatenated databases using X!Tandem (2009.10.01.1 from TPP v4.4). The average mass was used for both peptide and fragment ions, with fixed modification (Carbamidomethyl, +57.02 Da) on Cys and variable modification (Oxidation, +15.99 Da) on Met. Tryptic cleavage at Lys or Arg only was selected and up to two missed cleavage sites were allowed. The mass tolerance for precursor ions and fragments were 3.0 Da and 1.0 Da for all datasets.

2.3 Extending Percolator for X!Tandem

We extended Percolator for filtering X!Tandem search results. Specifically, Percolator extracts a set of discriminant features from the data and each PSM is represented as a vector $\mathbf{x}_i$ and a class label $y_i (i = 1, ..., M)$ where M is the total number of PSMs. Each component in $\mathbf{x}_i$ is a feature $x_{ij} (j = 1, ..., N)$ interpreted as the j^{th} feature of the i^{th} PSM, where N is the dimension of the feature space.

A linear SVM with a soft margin is trained to generate a credibility score for each PSM. Linear SVMs with a soft margin are robust tools for data classification [23]. The hyperplane in SVM is formed by optimizing the following objective function with constraints:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \mathcal{C}\sum_{i=1}^{M}\xi_i \tag{2}$$

$$\text{subject to}: \ y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle) + b \geqslant 1 - \xi_i \tag{3}$$

where $\mathbf{w}$ is the weight vector, ξ_i are slack variables that allow misclassification, $\mathcal{C}$ determines the penalty of misclassification, and b is the bias.

The key component in Percolator is to label each PSM so as to train a SVM. Since we do not know in advance which PSMs are correct/incorrect identifications, a target-decoy approach is used to construct positive and negative PSMs for SVM training. Particularly, a subset of PSMs regarded as "correct identifications" from the target database are used as positive training examples while all PSMs from the decoy database are used as the negative examples. In order to build a high-quality training dataset, the Percolator algorithm attempts to iteratively remove potential false positive identifications from the target database (Algorithm 1). This is done by calculating a FDR in each iteration and filtering the target hits that appear below the expected FDR threshold (Algorithm 2).

Algorithm 1 Percolator

```
1: Input: PSM list $L$; feature set $\mathcal{F}$
2: Output: PSM probability list $L^*$
3: $L' \leftarrow L$;
4: while number of filtered target PSMs $> 0$ do
5:     $< L', C > \leftarrow$ filterPSM($L'$);
6:     $svm \leftarrow$ trainSVM($L'$, $\mathcal{F}$, $C$);
7:     $L' \leftarrow$ rank($svm$, $L'$, $\mathcal{F}$);
8: end while
9: // use the SVM model from the last iteration to
   re-score PSM list
10: $L^* \leftarrow$ computeProbability($svm$, $L$, $\mathcal{F}$);
11: return $L^*$;
```

From search results of X!Tandem, we extracted 14 features for training SVM in Percolator. Table 1 summarizes the features used by our Percolator for

Algorithm 2 filterPSM

```
1:  Input: PSM list $L$
2:  Output: filtered list $L'$; class label $C$
3:  $p \leftarrow 0$; // a pointer that go through the PSM list
4:  while $FDR < 0.01$ do
5:      $p \leftarrow p + 1$;
6:      if $L[p] \in targets$ then
7:          $L'[p] \leftarrow L[p]$;
8:          $C[p] \leftarrow PositiveLabel$;
9:      else
10:         $L'[p] \leftarrow L[p]$;
11:         $C[p] \leftarrow NegativeLabel$;
12:     end if
13:     $FDR \leftarrow$ computeFDR($L$, $p$);
14: end while
15: // collect the rest of decoys as negative examples
16: while $L[p] \neq null$ do
17:     $p \leftarrow p + 1$;
18:     if $L[p] \in decoys$ then
19:         $L'[p] \leftarrow L[p]$;
20:         $C[p] \leftarrow NegativeLabel$;
21:     end if
22: end while
23: return $< L', C >$;
```

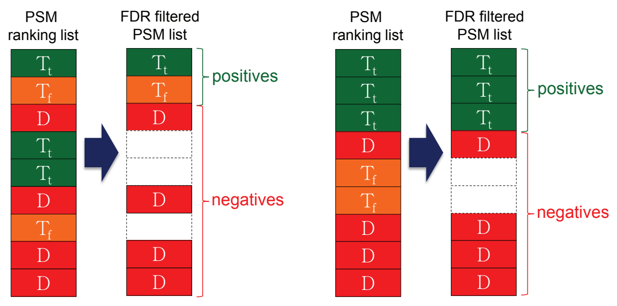
X!Tandem. These features were selected according to previous studies on Percolator for SEQUEST and MASCOT [16], [19]. Features that explore protein level information were excluded to avoid potential overfitting [24], [25].

Following the same configuration as in Percolator for SEQUEST and MASCOT [16], [19], we implemented the iterative PSM filtering procedure (Algorithm 1 and 2). The result of Percolator is a list of PSM scores reported by the trained SVM model from the last iteration.

2.4 Semi-supervised Learning on Creating Training Dataset

In Percolator, the training set is built by removing ambiguous PSMs from the target database using a FDR threshold (Algorithm 2). However, since the FDR is estimated by using PSMs from the decoy database, the rankings of the decoy PSMs determine how many PSMs from the target database will be removed and which of them will be used as positive training examples in each iteration.

As an example, assume that the PSM list in Figure 2a is the initial ranking using PSM search scores of a database search algorithm whereas the PSM list in Figure 2b is the re-ranking after further processing. Identifications from the target database are denoted as "T", from which true positive identifications and false positive identifications are denoted as "T_t" and "T_f", respectively. Any identification from the decoy database is denoted as "D" . In both cases (Figure 2a,b), by estimating FDR (Equation 1) and using any threshold smaller than 0.5, we will remove any PSMs from the target database that appear below one or more PSMs from the decoy database. Therefore, the resulting training set from Figure 2a includes only two positive training examples where one of them is a false positive identification that will be treated incorrectly by SVM as a positive example. In contrast, the resulting training set from Figure 2b includes three positive training examples and all of them are true identifications.



(a) Initial PSM list (b) Re-ranked PSM list

Fig. 2. Schematic illustration of PSM list on creating training dataset. (a) Initial PSM list ranked by search score from database search algorithm. (b) A re-ranked PSM list from further processing. $\mathrm{T_t}$ and $\mathrm{T_f}$ are true positive and false positive identifications from target database. D denotes identification from decoy database. Empty rectangles indicate that the corresponding PSM is removed after FDR filtering.

In this study, we evaluated the number of PSMs included for SVM training during the semi-supervised learning and the boost learning procedures. The FDR threshold of 0.01 is used for PSM filtering in each semi-supervised learning iteration.

2.5 Self-boosted Percolator

It is evident from Section 2.4 that the SSL algorithm used by Percolator for SVM training is sensitive to the initial PSM ranking list. That is, a poor initial ranking will have a reduced number of target PSMs passing the predefined FDR filtering threshold, causing an under-representation of positive training examples. This under-representation of positive training examples persists through the iteration of the training process since once a target PSM is removed by FDR filtering, it will not be considered in followup interactions.

One way to overcome this deficiency is to repeat the Percolator training and filtering process multiple times each on the PSM ranking list generated in its previous run. The assumption is that if Percolator could improve the ranking of PSMs, then by each time repeating the Percolator training on the PSM ranking list generated in its previous run, we can obtain more

TABLE 1
Summary of features used by Percolator for X!Tandem search result.

Feature	Description
Hyperscore	the first Hyperscore reported by X!Tandem
Δscore	the difference between the first Hyperscore and the second score
expect	the expectation reported by X!Tandem
ln(rHyper)	the natural logarithm of the rank of the match based on the Hyperscore
mass	the observed monoisotopic mass of the identified peptide
Δmass	the difference in calculated and observed mass
abs(Δmass)	the absolute value of the difference in calculated and observed mass
ionFrac	the fraction of matched b and y ions
enzN	a Boolean value indicating if the peptide is preceded by a tryptic site
enzC	a Boolean value indicating if the peptide has a tryptic C-terminus
enzInt	the number of missed internal tryptic sites
pepLen	the length of the matched peptide, in residues
charge	the predicted charge state of the peptide
retention	the retention of the mass spectrometer

target PSMs with potentially less false positives. We call this cascade learning procedure "self-boosting" and the algorithm "Self-boosted Percolator" (Algorithm 3).

Algorithm 3 Self-boosted Percolator

```
1: Input: Initial PSM list $L$, number of boost runs $b$
2: Output: PSM probability list $L^*$
3: $L' \leftarrow L$;
4: while $b > 0$ do
5:     $L' \leftarrow$ Percolator($L'$);
6:     $b \leftarrow b - 1$;
7: end while
8: $L^* \leftarrow L'$;
9: return $L^*$;
```

2.6 Performance Comparison on PSM Post-Processing

The performance of Self-boosted Percolator was compared with PeptideProphet and Percolator. The results from the database search algorithms (without further processing) were used as the baselines. Specifically, we calculated the number of accepted PSMs reported by each PSM filtering algorithm with respect to the estimated FDR (denoted as q-value) threshold ranging from (0, 0.2]. For each algorithm, the level of overfitting was assessed using the method described in Section 2.2.

For PeptideProphet, we used TPP v4.4 [26]. The database search outputs from X!Tandem are preprocessed by msconvert.exe to generate mzXML files for running PeptideProphet. For Percolator, it is to run the Self-boosted Percolator with the boost runs set to 1. This is algorithmically equivalent to the original implementation of the Percolator algorithm for MASCOT and SEQUEST.

3 Results and Discussion

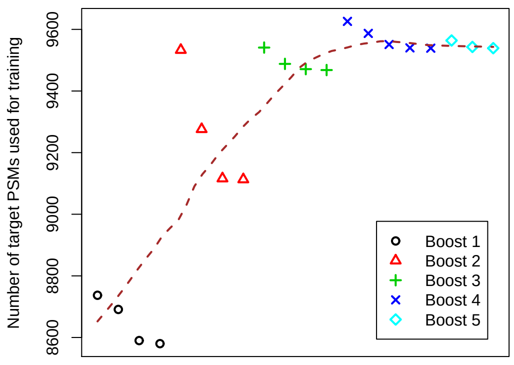
3.1 Percolator is Sensitive to PSM Ranking

We evaluated the number of target PSMs included in each boost run of Percolator. Figure 3a shows the result from the Yeast dataset. As can be seen, Percolator included ~8600 target PSMs as positive training examples after performing FDR filtering in the initial step. The number increased to ~9100 after FDR filtering in the second boost run and plateaued at ~9500 in the fifth boost run. A similar trend was observed in processing the Worm dataset (Figure 3b) where the number of PSMs utilized for model training increased from the initial ~7000 to ~9000 in the fifth boost run. Notice that FDR was controlled at the same level (i.e. 1%) among each boost run. These results suggest that the original Percolator algorithm is sensitive to the initial PSM ranking, and the self-boosted Percolator is able to overcome this inefficiency by extracting increasingly more target PSMs from each boost run for SVM model training and PSM re-ranking.

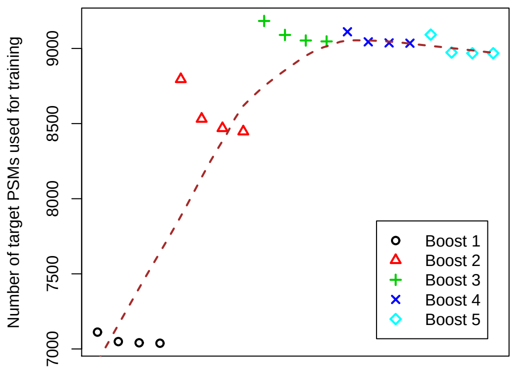
In Figure 3, multiple iterations of filtering within each boost run are denoted by points with the same shape. Within each boost run, target PSMs are filtered iteratively by a predefined FDR threshold (1% in our experiments). It is clear that within each boost run, the SSL algorithm of Percolator generally converges after a few iterations. Note that the iterative filtering of SSL dose not increase the number of target PSMs for SVM training.

3.2 PSM Post-processing

The motivation of extracting more target PSMs through self-boosting is to create a more robust and accurate PSM filtering model which could lead to the identification of more PSMs without sacrificing FDR. Figure 4a and c show the performance of Self-boosted Percolator in comparison with PeptideProphet and Peculator without self-boosting in the Yeast and the Worm datasets, respectively. We observed that in both



(a) Self-boosting on Yeast dataset



(b) Self-boosting on Worm dataset

Fig. 3. Self-boosting of Percolator on (a) the Yeast dataset and (b) the Worm dataset. For each dataset, 5 boost runs were conducted. Within a boost run, FDR filtering iterations were denoted by points with the same shape. For each dataset, a locally weight regression line was fitted to all points.

datasets Self-boosted Percolator identified consistently more PSMs at any given q-value thresholds compared to PeptideProphet and Percolator without self-boosting. In general, the performance of Percolator (without self-boosting) was relatively better than PeptideProphet. This is consistent with the result obtained by käll *et al.* [16]. Using the E-value reported from X!Tandem for PSM filtering gave lower sensitivity compared to those achieved using post-processing algorithms. It is evident that Self-boosted Percolator is robust to the noise of initial PSM ranking and can fully recover the performance of Percolator without self-boosting.

To verify that the additional PSM identifications reported by each algorithm retain the same level of FDR, we repeated the database search for each dataset using the modified target-decoy approach with a paired nonsense database as described in Section 2.2. Figure 4b plots the first 1500 PSMs (approximately correspond to a q-value of 0.1 in the target database) identified from the reverse nonsense group against its target nonsense group using the Yeast dataset with its corresponding database. The same was done for the Worm dataset and the results are shown in Figure 4d. It is observed that all lines closely resemble the 45 degree lines, indicating that X!Tandem raw score, PeptideProphet, Percolator, and Self-boosted Percolator did not bias to penalizing only PSMs from the reverse database. This is especially important for Percolator and Self-boosted Percolator as the information of reverse database were explicitly used to train the SVM model.

4 Conclusion

In this study, we proposed a Self-boosted Percolator for post-processing X!Tandem search results. We found that the learning procedure used by Percolator relies heavily on the guidance of the decoy PSMs and their ranking among target PSMs. The iterative FDR filtering of PSMs is the key to enhance the discriminant ability of the final SVM model. If the decoy PSMs are poorly ranked in the initial PSM list, the performance of the SVM model may degenerate. We propose to overcome the deficiency of the original Percolator algorithm by using a cascade learning approach where the performance is boosted by using the PSM ranking from the previous boost run as the input of the next boost run. The consistent improvement of performance on two benchmark datasets with samples derived from complex organisms indicates that the proposed Self-boosted Percolator is effective for improving X!Tandem on peptide identification from tandem mass spectrometry.

Availability

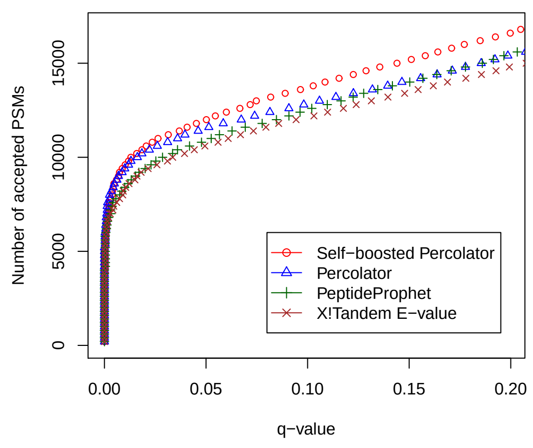
Self-boosted Percolator is freely available from: ***http://code.google.com/p/self-boosted-percolator***

Acknowledgments

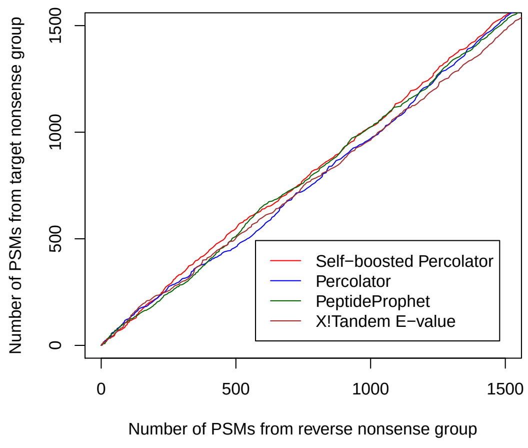
Pengyi Yang is supported by the NICTA International Postgraduate Award (NIPA) and the NICTA Research Project Award (NRPA). This work is funded in part by ARC Discovery Grant DP0984267 (Penghao Wang, Yee Hwa Yang) and FT0991918 (Yee Hwa Yang).

References

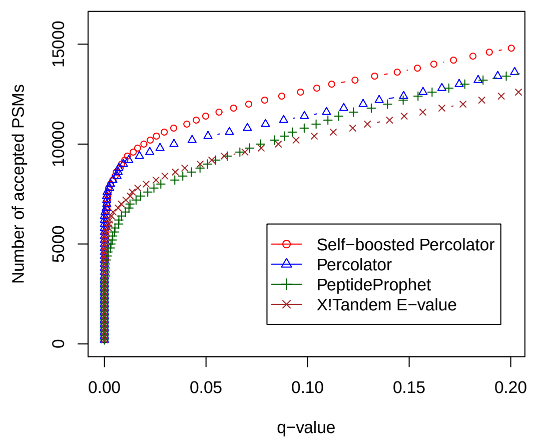
[1] X. Han, A. Aslanian, and J. Yates III, “Mass spectrometry for proteomics,” *Current Opinion in Chemical Biology*, vol. 12, no. 5, pp. 483–490, 2008.



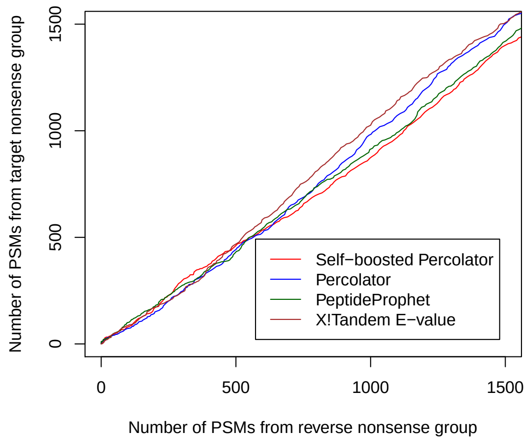
(a) Yeast database



(b) Yeast with paired nonsense database



(c) Worm database



(d) Worm with paired nonsense database

Fig. 4. The number of accepted PSMs is determined at each q-value threshold on X!Tandem search results using X!Tandem E-value, PeptideProphet, Percolator without self-boosting, and Self-boosted Percolator. (a) Accepted PSMs across different q-value threshold in the Yeast dataset. (b) False positive identifications in the Yeast dataset from using target-decoy approach with a paired nonsense database. (c) Accepted PSMs across different q-value threshold in the Worm dataset. (d) False positive identifications in the Worm dataset from using target-decoy approach with a paired nonsense database.

[2] J. Eng, A. McCormack, and J. Yates III, “An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database,” *Journal of the American Society for Mass Spectrometry*, vol. 5, no. 11, pp. 976–989, 1994.

[3] R. Craig, J. Cortens, D. Fenyo, and R. Beavis, “Using annotated peptide mass spectrum libraries for protein identification,” *Journal of Proteome Research*, vol. 5, no. 8, pp. 1843–1849, 2006.

[4] H. Lam, E. Deutsch, J. Eddes, J. Eng, N. King, S. Stein, and R. Aebersold, “Development and validation of a spectral library searching method for peptide identification from MS/MS,” *Proteomics*, vol. 7, pp. 655–667, 2007.

[5] A. Frank, M. Savitski, M. Nielsen, R. Zubarev, and P. Pevzner, “De novo peptide sequencing and identification with precision mass spectrometry,” *Journal Proteome Research*, vol. 6, no. 1, pp. 114–123, 2007.

[6] M. Bern, Y. Cai, and D. Goldberg, “Lookup peaks: a hybrid of de novo sequencing and database search for protein identification by tandem mass spectrometry,” *Analytical Chemistry*, vol. 79, no. 4, pp. 1393–1400, 2007.

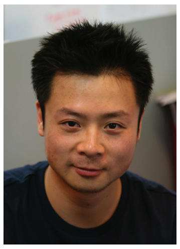
[7] D. Perkins, D. Pappin, D. Creasy, and J. Cottrell, “Probability-based protein identification by searching sequence databases using mass spectrometry data,” *Electrophoresis*, vol. 20, no. 18, pp. 3551–3567, 1999.

[8] R. Craig and R. Beavis, “TANDEM: matching proteins with tandem mass spectra,” *Bioinformatics*, vol. 20, no. 9, pp. 1466–1467, 2004.

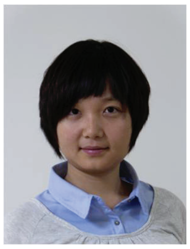
[9] L. Geer, S. Markey, J. Kowalak, L. Wagner, M. Xu, D. Maynard, X. Yang, W. Shi, and S. Bryant, “Open mass spectrometry search algorithm,” *Journal of Proteome Research*, vol. 3, no. 5, pp. 958–964, 2004.

[10] I. Shilov, S. Seymour, A. Patel, A. Loboda, W. Tang, S. Keating, C. Hunter, L. Nuwaysir, and D. Schaeffer, “The paragon algorithm, a next generation search engine that uses sequence temperature values and feature probabilities to identify peptides from tandem mass spectra,” *Molecular & Cellular Proteomics*, vol. 6, no. 9, pp. 1638–1655, 2007.

[11] B. Balgley, T. Laudeman, L. Yang, T. Song, and C. Lee, "Comparative evaluation of tandem MS search algorithms using a target-decoy search strategy," *Molecular & Cellular Proteomics*, vol. 6, no. 9, p. 1599, 2007.
[12] E. Kapp, F. Schütz, L. Connolly, J. Chakel, J. Meza, C. Miller, D. Fenyo, J. Eng, J. Adkins, G. Omenn *et al.*, "An evaluation, comparison, and accurate benchmarking of several publicly available ms/ms search algorithms: sensitivity and specificity analysis," *Proteomics*, vol. 5, no. 13, pp. 3475–3490, 2005.
[13] A. Nesvizhskii, F. Roos, J. Grossmann, M. Vogelzang, J. Eddes, W. Gruissem, S. Baginsky, and R. Aebersold, "Dynamic spectrum quality assessment and iterative computational analysis of shotgun proteomic data," *Molecular & Cellular Proteomics*, vol. 5, no. 4, p. 652, 2006.
[14] M. Kallberg and H. Lu, "An improved machine learning protocol for the identification of correct sequest search results," *BMC Bioinformatics*, vol. 11, no. 1, p. 591, 2010.
[15] A. Keller, A. Nesvizhskii, E. Kolker, and R. Aebersold, "Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search," *Analytical Chemistry*, vol. 74, no. 20, pp. 5383–5392, 2002.
[16] L. Käll, J. Canterbury, J. Weston, W. Noble, and M. MacCoss, "Semi-supervised learning for peptide identification from shotgun proteomics datasets," *Nature Methods*, vol. 4, no. 11, pp. 923–925, 2007.
[17] H. Choi, D. Ghosh, and A. Nesvizhskii, "Statistical validation of peptide identifications in large-scale proteomics using the target-decoy database search strategy and flexible mixture modeling," *Journal of Proteome Research*, vol. 7, no. 01, pp. 286–292, 2007.
[18] E. Deutsch, L. Mendoza, D. Shteynberg, T. Farrah, H. Lam, N. Tasman, Z. Sun, E. Nilsson, B. Pratt, B. Prazen *et al.*, "A guided tour of the trans-proteomic pipeline," *Proteomics*, vol. 10, no. 6, pp. 1150–1159, 2010.
[19] M. Brosch, L. Yu, T. Hubbard, and J. Choudhary, "Accurate and sensitive peptide identification with Mascot Percolator," *Journal of Proteome Research*, vol. 8, no. 6, pp. 3176–3181, 2009.
[20] M. Spivak, J. Weston, L. Bottou, L. Käll, and W. Noble, "Improvements to the percolator algorithm for peptide identification from shotgun proteomics data sets," *Journal of Proteome Research*, vol. 8, no. 7, pp. 3737–3745, 2009.
[21] M. Bern and Y. Kil, "Comment on "unbiased statistical analysis for multi-stage proteomic search strategies"," *Journal of proteome research*, vol. 10, no. 4, pp. 2123–2127, 2011.
[22] J. Elias and S. Gygi, "Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry," *Nature Methods*, vol. 4, no. 3, pp. 207–214, 2007.
[23] A. Ben-Hur, C. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch, "Support vector machines and kernels for computational biology," *PLoS Computational Biology*, vol. 4, no. 10, p. e1000173, 2008.
[24] L. Everett, C. Bierl, and S. Master, "Unbiased statistical analysis for multi-stage proteomic search strategies," *Journal of Proteome Research*, vol. 9, no. 2, pp. 700–707, 2010.
[25] J. Zhang, L. Xin, B. Shan, W. Chen, M. Xie, D. Yuen, W. Zhang, Z. Zhang, G. Lajoie, and B. Ma, "Peaks db: De novo sequencing assisted database search for sensitive and accurate peptide identification," *Molecular & Cellular Proteomics*, 10.1074/mcp.M111.010587, 2011.
[26] A. Keller, J. Eng, N. Zhang, X. Li, and R. Aebersold, "A uniform proteomics ms/ms analysis platform utilizing open xml file formats," *Molecular Systems Biology*, vol. 1, no. 1, 2005.



Pengyi Yang received the MS degree in computer science in 2008. He is currently working toward a PhD degree in the School of Information Technologies, University of Sydney. He is also associated with the School of Statistics and Mathematics, University of Sydney, and the Diabetes and Obesity Program, Garvan Institute of Medical Research, New South Wales. His research interests include applications of machine learning algorithms and statistical models for systems biology. He is currently funded by the NICTA International Postgraduate Award and the NICTA Research Project Award.

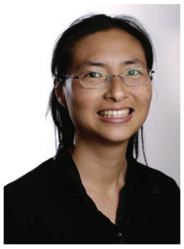


Jie Ma received the PhD degree in biochemistry and molecular biology from Beijing Institute of Radiation Medicine, China, in 2010. She is now an assistant professor in Beijing Proteome Research Center. Her research interests are in the areas of proteome bioinformatics, and focuses on improving the analysis process of tandem mass data.

Penghao Wang received his PhD degree in School of Information Technologies from University of Sydney, Australia, in 2009. He currently works as a post-doc researcher at the School of Mathematics and Statistics at University of Sydney, Australia. His research interests lie in developing algorithms for bioinformatics and distributed computing.

Yunping Zhu is the professor of Beijing Proteome Research Center (BPRC) and Beijing Institute of Radiation Medicine (BIRM). Mainly engages in the research of proteome bioinformatics and liver systems biology. His lab greatly supported the Chinese Human Liver Proteome Project (CNHLPP). He has more than eighty papers published in Mol. Cell. Proteomics, J. Proteome Res., Proteomics and BMC Bioinformatics, etc.

Bing B. Zhou received the BS degree from Nanjing Institute of Technology, Nanjing, China, and the PhD degree in computer science from Australian National University. He is currently an associate professor in the School of Information Technologies, University of Sydney, Sydney. His research interests include parallel/distributed computing, grid computing, peer-to-peer systems, parallel algorithms, and bioinformatics. He has a number of publications in leading international journals and conference proceedings. His research has been funded by the Australian Research Council through several Discovery Project grants.



Yee Hwa (Jean) Yang has a bachelor's degree in statistics from the University of Sydney and completed her doctoral studies in the Department of Statistics at the University of California, Berkeley where she worked under the supervision of Terry Speed on the design and analysis of microarray experiments. She relocated back to Sydney 5 years ago and currently a Future Fellow at the School of Mathematics and Statistics, University of Sydney.

Her research work has centered on the development of statistical methodology and the application of statistics to problems in genomics, proteomics and biomedical research. In particular, her focus is on developing methods for integrating expression studies and other biological metadata such as miRNA expression, sequence information and clinical data. As a statistician who works in the bioinformatics area, she enjoys research in a collaborative environment, working closely with scientific investigators from diverse backgrounds.