Clin Proteom (2009) 5:23–36
DOI 10.1007/s12014-009-9024-5

An Unsupervised, Model-Free, Machine-Learning Combiner for Peptide Identifications from Tandem Mass Spectra

Nathan Edwards · Xue Wu · Chau-Wen Tseng

Published online: 20 March 2009

Abstract As the speed of mass spectrometers, sophistication of sample fractionation, and complexity of experimental designs increase, the volume of tandem mass spectra requiring reliable automated analysis continues to grow. Software tools that quickly, effectively, and robustly determine the peptide associated with each spectrum with high confidence are sorely needed. Currently available tools that postprocess the output of sequence-database search engines use three techniques to distinguish the correct peptide identifications from the incorrect: statistical significance re-estimation, supervised machine learning scoring and prediction, and combining or merging of search engine results. We present a unifying framework that encompasses each of these techniques in a single model-free machine-learning framework that can be trained in an unsupervised manner. The predictor is trained on the fly for each new set of search results without user intervention, making it robust for different instruments, search engines, and search engine parameters. We demonstrate the performance of the technique using mixtures of known proteins and by using shuffled databases to estimate false discovery rates, from data acquired on three different instruments with two different ionization technologies. We show that this approach outperforms machine-learning techniques applied to a single search engine's output, and demonstrate that combining search engine results provides additional benefit. We show that the performance of the commercial Mascot tool can be bested by the machine-learning combination of two open-source tools X!Tandem and OMSSA, but that the use of all three search engines boosts performance further still. The Peptide identification Arbiter by Machine Learning (PepArML) unsupervised, model-free, combining framework can be easily extended to support an arbitrary number of additional searches, search engines, or specialized peptide–spectrum match metrics for each spectrum data set. PepArML is open-source and is available from http://peparml.sourceforge.net.

Keywords Bioinformatics · Peptide identification · Tandem mass spectra · Machine-learning

Electronic supplementary material The online version of this article (doi: 10.1007/s12014-009-9024-5) contains supplementary material, which is available to authorized users.

N. Edwards (✉)
Department of Biochemistry and Molecular & Cellular Biology, Georgetown University Medical Center, Washington, DC, USA
e-mail: nje5@georgetown.edu

X. Wu · C.-W. Tseng
Department of Computer Science, University of Maryland, College Park, MD, USA

Introduction

Peptide identification by tandem mass spectrometry is widely used for protein characterization in complex samples. High-throughput proteomics strategies such as the liquid chromatography–tandem mass spectrometry (LC-MS/MS)-based shotgun [1] and MudPIT [2] workflows generate tens of thousands of tandem mass spectra in a single acquisition, identifying peptide sequences and, by association, the protein content of the analytical samples. As the speed of mass spectrometers, the sophistication of sample fractionation, and the com-

plexity of experimental designs increase, the volume of tandem mass spectra requiring reliable automated analysis will continue to grow. Consequently, software tools that quickly, effectively, and robustly determine the peptide associated with each spectrum with high confidence are sorely needed.

In this high-throughput LC-MS/MS setting, peptides are associated with tandem mass spectra by search engines that compare peptide sequences with each spectrum and rate the quality of the match with a score. Each search engine generates a list of zero, one, or in some cases, as many as ten of the best scoring peptides for each tandem mass spectrum, along with various measures of the quality of each peptide–spectrum match. There are a large variety of these search engines available [3–13], ranging from mature commercial offerings such as SEQUEST [3] and Mascot [4] to more recent open-source offerings such as X!Tandem [9], OMSSA [10], and MyriMatch [13]. All of these tools utilize the same basic algorithmic framework but vary widely in the details, particularly in the methodologies used to score and rank the candidate peptide sequences. Further, many of these tools provide estimates of statistical significance for each peptide identification, but each tool uses different significance estimation techniques. Despite more than 10 years of research on algorithms, scoring, statistical significance, and software tools for peptide identification by sequence database search, there is little, if any, consensus on the optimal scoring function for evaluating the quality of a peptide assignment to a given spectrum. Particularly problematic, given the variation in scoring and statistical significance techniques, is distinguishing correct peptide identifications from those that are incorrect.

In addition to the array of sequence database search engines now available, a variety of techniques have been applied to their outputs in order to improve the reliability of peptide identification. Broadly, these tools treat the search engines as "black boxes," attempting to do a better job at discerning correct peptide identifications from incorrect identifications than is possible using only a search engine's score or E-value. These black-box tools employ one or more of various techniques, including statistical significance re-estimation, supervised and semisupervised machine learning scoring and prediction, and combining or merging of search engine results.

The popular decoy database strategy [14, 15] for estimating false discovery rates (FDR) and the unsupervised expectation-maximization (E-M) algorithm phase of Peptide Prophet [16] use empirical properties of the underlying search engine results as a null-model to re-estimate the statistical significance or likelihood associated with peptide identification results. Other tools re-estimate the significance of peptide identification results using a theoretical model [17].

Supervised machine learning techniques [15, 16, 18–20] have been applied to existing and novel features derived from search engine results to derive score functions that demonstrate superior sensitivity and specificity for specific data sets. The supervised machine learning techniques are often combined with statistical significance re-estimation techniques [15, 16] to normalize the class prediction probabilities.

Lastly, result combiners [15, 21, 22] rely on peptide identification results from multiple search engines to improve sensitivity and specificity, under the premise that search engine agreement increases the likelihood that the identification is correct. These are often used in conjunction with statistical significance re-estimation techniques [15, 22], which are used to normalize the scores from each search engine to make them comparable.

The effectiveness of machine learning in combining various measures of peptide identification confidence from a single search engine into a single correctness likelihood or prediction has already been demonstrated [15, 16, 18–20]. In each case, the predictors are first trained using tandem mass spectra generated from controlled mixtures of known protein standards, for which peptide identifications that correspond to peptides from the known proteins are presumed correct. The trained predictor can then be applied to previously unseen peptide identifications to distinguish correct from incorrect identifications. Various machine learning technologies have been applied in this way, from linear discriminant analysis (LDA) [16] to support vector machines (SVM) [19], neural networks [20], and random forest [15, 18]. In each case, these techniques demonstrate a considerable boost in peptide identification sensitivity and specificity, as compared to the original search engine's score or E-value.

A major concern with supervised machine learning techniques, however, is the question of generalization of the trained machine learning model to peptide identification results from a different instrument or search engine. Unsupervised training, which can be applied on the fly to each new data set, can adapt to the particular features of a given set of results as needed. Peptide Prophet [16], in particular, utilizes a supervised training phase to establish the optimal linear weighting of the various peptide identification properties and scores, and an unsupervised empirical fit of a bimodal mixture model to the discriminant scores to establish the likelihood of correct peptide identifications. Peptide

Prophet's unsupervised significance re-estimation helps to make its probabilities quite robust, as long as the underlying linear discriminant model generalizes well enough to distinguish correct and incorrect identifications well. Semisupervised machine-learning has recently emerged [23, 24] as a solution to concerns about the ability of supervised machine-learning to generalize well. Semisupervised learning uses decoy database hits as known false peptide identifications to guide the machine learning approach.

A second significant concern with black-box search engine output re-analysis techniques is the requirement that correct (and incorrect) peptide identifications respect some underlying mathematical model. Sometimes, these model assumptions are explicit, as in the use of LDA [16] and SVM [19] machine learning techniques, while in other cases, they are implicit, such as the assumption built into most result combiners that incorrect peptide identifications from different search engines are uncorrelated [15, 21, 22].

In this work, we describe the application of a model-free, result combining, unsupervised machine-learning approach to the problem of distinguishing correct from incorrect peptide identifications that outperforms each of these basic techniques. Mixtures of standard proteins and estimated FDR are used throughout to ensure that the performance of our peptide identification results arbiter is evaluated objectively. We demonstrate that these techniques outperform search engine E-values when applied to the results from a single search engine, and that using machine learning to combine results from multiple search engines provides additional benefit.

We first show the effectiveness of our machine learning combiner in a supervised setting, in which the true-positive proteins are known in advance, and then show how to use a heuristic, iterative training procedure to achieve comparable performance in an unsupervised fashion. Our model-free, unsupervised machine learning algorithm is applied anew to each set of search engine results, making it robust across instruments, search engines, search engine parameter settings, and protein sequence databases without user intervention.

We apply this technique to three data sets derived from synthetic protein mixtures, analyzed by three different instruments utilizing two distinct ionization techniques. We use peptide identification results from three different search engines: Mascot [4], X!Tandem [9], and OMSSA [10]. We demonstrate that, in each case, the use of additional metrics of peptide identification confidence generated by each search engine can boost identification reliability over using the search engines' E-values alone. We also show that the performance of the machine learning technique applied to the results of just two open-source search engines, X!Tandem and OMSSA, is almost as good as that obtained when using results from all three.

Experimental Procedures

Tandem Mass Spectra

We use three MS/MS data sets created from known protein mixtures on three different instruments utilizing a variety of ionization, mass measurement, and fragmentation technologies. We label these data sets S17, AURUM, and OMICS throughout. A description of MS/MS search engines and sequence database search parameters, used to determine true-positive peptide assignment statistics, follows.

S17 1,389 MS/MS spectra from the Sashimi project data repository (http://sashimi.sourceforge.net) data set `17mix_test2`, representing a tryptic digest of 17 standard proteins analyzed using an electrospray ionization quadrupole time-of-flight mass spectrometer (Q-TOF Ultima) (Micromass/Waters, Manchester, United Kingdom). The S17 data set contains 241 (17.35%) true-positive spectra assignable to peptides from the expected or contaminant proteins.

AURUM 10,097 MS/MS spectra from the Aurum 1.0 data set [25]. Spectra were generated using a matrix assisted laser desorption/ionization time-of-flight (MALDI TOF-TOF) instrument (Applied Biosystems 4700, Foster City, CA, USA) from 246 commercially sourced human proteins synthetically expressed in *Escherichia coli*, checked for purity, digested with trypsin, and spotted, one protein per MALDI spot, for analysis. The AURUM data set contains 4,101 (40.61%) true-positive spectra assignable to peptides from the expected or contaminant proteins.

OMICS 19,000 MS/MS spectra from the data set described in Keller et al. [26] representing a tryptic digest of 18 standard proteins analyzed by electrospray ionization–ion trap mass spectroscopy (Thermo Finnigan, San Jose, CA, USA). The OMICS data set contains 2,890 (15.2%) true-positive spectra assignable to peptides from the expected or contaminant proteins.

MS/MS Search Engines and Sequence Database Search

We use three sequence database search engines to search the tandem mass spectra: X!Tandem [8], Release 2008.02.01.1; MASCOT [4], version 2.2.03; and OMSSA [10], version 2.1.1. We used the following search parameters for all data sets: precursor mass tolerance of 2.0 Da, fixed cysteine carbamidomethyl modification, and variable methionine oxidation modifications. Fully tryptic peptides with, at most, one missed cleavage were specified for S17, AURUM, and OMICS. Tandem's refinement mode was not used. The data sets S17, AURUM, and OMICS were searched using fragment mass tolerances of 0.2, 0.4, and 0.6 Da, respectively.

Two protein sequence databases were used: UniProtKB/Swiss-Prot (version 53.0) and Human International Protein Index (IPI-Human) (version 3.32). Data sets S17 and OMICS were searched against Swiss-Prot, while data set AURUM was searched against IPI-Human. For each sequence database, we create two decoy databases of independently shuffled sequences. Decoy searches were conducted independently of the target searches, and the search results from the decoy databases were used for estimating FDR as described in the experimental procedures to follow.

Searching tandem mass spectra data sets against protein sequence databases, each search engine generates zero, one, or more peptide assignments for each spectrum (we will refer to these peptide–spectrum assignments as *peptide IDs* for the remainder of this paper). The peptide IDs differ between search engines, as each search engine has its own heuristics for selecting candidate peptides and filtering noisy spectra, as well as different scoring functions for assessing the quality of the peptide–spectrum match and different techniques for assessing the statistical significance of a peptide ID score.

We extract the top-ranked peptide, for each spectrum, from each search engine's results. Peptide IDs corresponding to an expected protein (in the synthetic protein mix used to generate each data set) are considered correct, regardless of E-value, score, or number of agreeing search engines. These peptide IDs are then considered *true* assignments. In addition, we check for contaminant proteins with highly statistically significant identifications in all search engines, also labeling their peptide IDs true. The peptide IDs not associated with proteins in the synthetic protein mix or confidently identified contaminant proteins are considered *false* assignments.

Use of Shuffled Decoy Databases

We use decoy database search results in a number of ways. As described by others, we use decoy peptide identifications to estimate the FDR of each of our various algorithms for selecting the best peptide ID for each spectrum. In addition, a number of our algorithms require that scores or E-values be calibrated, where the calibration is achieved using decoy peptide IDs. This requires some care since we cannot use the same decoy peptide IDs as algorithmic input data *and* to evaluate the performance of the algorithm—this will overestimate the algorithm's performance. For this reason, decoy sequence databases are constructed by randomly shuffling the order of the amino-acids in each protein sequence. We create two independently shuffled decoy sequence databases and compute decoy peptide IDs for each. One set of decoy peptide IDs is used exclusively for evaluating the performance of peptide ID selection algorithms, while the other is used internally to calibrate search engine scores (voting heuristic combiner) or prediction confidences (unsupervised machine learning). We point out that independent decoy peptide IDs can only be achieved by shuffling, rather than reversing, protein sequences.

Voting Heuristic Search Engine Combiner

To test whether machine learning is necessary to achieve good performance, we designed a simple heuristic search engine combiner based on the widely used notion that, when different search engines give the same peptide identification, the identification is more likely to be correct than the single search engine E-values or scores would otherwise suggest [15, 21, 22].

We considered a variety of heuristic result-combining schemes based on consensus identifications. Ultimately, we selected a voting scheme that seemed to consistently outperform the other consensus heuristics we tried. Briefly, each search engine's E-value is calibrated by the number of decoy peptide IDs, called decoy hits, in the search engine's decoy search results with the same or better E-value. Each peptide ID then has a minimum decoy hit count over the one or more search engines with the same peptide ID. Peptide IDs are ranked according to the number of agreeing search engines (votes, decreasing), and then by the minimum of their decoy hit counts (minimum decoy hits, increasing). The minimum decoy hits value is also used as the score of the peptide ID selection for determining ROC curves and other performance metrics.

Machine Learning Search Engine Combiner

The Peptide identification Arbiter by Machine Learning (PepArML) result combiner models the peptide identification problem as a classification problem. PepArML classifies each spectrum's peptide IDs, generated by one or more search engines, as either *true* or *false* with some confidence. For each spectrum, the peptide ID classified as true with highest confidence is the predicted correct peptide ID.

In addition to a score and E-value, search engines compute additional metrics characterizing peptide IDs. Using a machine learning framework makes it possible to use these additional metrics to predict the correctness of peptide IDs with little additional effort. We construct a feature vector for each peptide–spectrum pair consisting of scores and E-values from each search engine, plus any additional metrics supplied by the search engine. When search engines assign spectra to different peptides, the score, E-value, and other metrics computed by a search engine may be absent for a particular peptide–spectrum pair, in which case the values are set to sentinels. An additional sentinel feature, per search engine, is set to indicate the presence or absence of the search engine's features for a particular peptide–spectrum pair. Scores computed directly from the spectrum or peptide, or both, can be easily added to the feature vector. We use a number of spectrum- and peptide–specific features that are available for all peptide–spectrum pairs. We do not, at this time, make any effort to eliminate features that are correlated. The peptide ID feature vector is shown in Table 1.

We chose the random forest machine-learning technique, implemented by the Weka [27] machine-learning package, for the PepArML combiner due to the number of missing values, and the heterogeneous mix of categorical, integer, and scale-free real values in a typical training data set. Early in the project, we experimented with other machine-learning techniques, including LDA, logistic regression, SVM, AdaBoost, and naive Bayes, and found the random forest technique to generally outperform the others. However, since a thorough examination of the strengths and weaknesses of a carefully tuned use of each technique is beyond the scope of this work, we do not suppose to suggest these techniques could not be made to work well too. Having chosen the random forest technique, we found that careful tuning of the training parameters was necessary to reliably achieve good performance. To

Table 1 Peptide identification features and their InfoGain (Rank) with respect to each data set

Search engine	Feature	S17	AURUM	OMICS
–	Precursor m/z	0.00 (24)	0.01 (21)	0.10 (18)
	Charge state	0.20 (13)	0.00 (25)	0.02 (24)
	Peptide molecular weight	0.09 (22)	0.01 (21)	0.33 (6)
	Molecular weight delta	0.28 (5)	0.29 (11)	0.14 (14)
	Peptide length	0.02 (23)	0.00 (25)	0.09 (19)
	# of agreeing search engines	0.36 (1)	0.39 (7)	0.38 (3)
	Missed cleavages	0.09 (20)	0.09 (19)	0.07 (20)
	# Tryptic termini	0.00 (24)	0.00 (23)	0.00 (25)
	Tryptic N-terminal	0.00 (24)	0.00 (23)	0.00 (25)
	Tryptic C-terminal	0.00 (24)	0.00 (25)	0.00 (27)
Tandem	# of matched b-ions	0.14 (16)	0.28 (12)	0.10 (17)
	b-ion score	0.16 (14)	0.20 (15)	0.07 (20)
	E-value	0.31 (4)	0.48 (1)	0.26 (7)
	Hyperscore	0.27 (8)	0.46 (2)	0.20 (9)
	Sum of matched intensity	0.12 (17)	0.21 (14)	0.13 (15)
	# of matched y-ions	0.24 (10)	0.39 (8)	0.14 (13)
	y-ion score	0.21 (12)	0.25 (13)	0.17 (11)
	Sentinel	0.12 (18)	0.16 (16)	0.07 (23)
Mascot	E-value	0.32 (2)	0.43 (3)	0.33 (5)
	# of matched peaks	0.16 (15)	0.16 (17)	0.11 (16)
	# of matched ions	0.27 (9)	0.33 (10)	0.17 (10)
	Score	0.32 (3)	0.42 (4)	0.33 (4)
	Sentinel	0.09 (21)	0.13 (18)	0.07 (22)
OMSSA	E-value	0.27 (7)	0.40 (5)	0.41 (2)
	# of matched ions	0.21 (11)	0.35 (9)	0.25 (8)
	p value	0.28 (6)	0.40 (6)	0.41 (1)
	Sentinel	0.09 (19)	0.09 (19)	0.14 (12)

train the random forest classifier, we first use the information gain, called InfoGain by Weka, to eliminate features with little predictive power, retaining features with InfoGain at least 0.01. The tree-building algorithm considers a random 25% of the remaining features at each node split. We use 100 random trees, with the prediction confidence of a peptide ID determined by the number of random trees classifying the peptide ID as true.

We trained classifiers using all possible combinations of search engines. Classifiers using only the features of Mascot, OMSSA, and Tandem are denoted C-M, C-O, and C-T, respectively. Classifiers C-MO, C-TM, and C-TO use features from Mascot and OMSSA, Tandem and Mascot, and Tandem and OMSSA, respectively. Classifier C-TMO uses features from all three search engines.

Sampling Structure for Supervised Learning Training and Evaluation

In order to get an unbiased estimate of the generalization error of the machine-learning predictor, we use fivefold stratified cross-validation by spectra. Each spectrum is designated true if at least one of its peptide IDs is known true, and false otherwise. The true and false spectra are each partitioned randomly into five similarly sized groups. In each cross-validation iteration, one group of true and false spectra is withheld, and the peptide IDs corresponding to the remaining spectra are used to train a machine-learning classifier. Once trained, the classifier is applied to the target *and* decoy peptide IDs corresponding to the withheld spectra and the predictions are used to evaluate the performance of supervised learning.

The peptide ID training data sets formed by cross-validation are typically quite unbalanced, with many more false peptide IDs than true peptide IDs. We uniformly down-sample the majority (false) class of training instances before training so that the number of false peptide IDs is, at most, five times the number true peptide IDs. The factor five, like the random forest training parameters, is the result of some tuning effort but seems to work well across many data sets.

We apply each of these structured sampling procedures external to the feature selection and Weka random forest training algorithm so that an unbiased, conservative estimate of the generalization performance of the machine-learning training can be obtained. For the unsupervised learning algorithm to follow, in which we *do not use the identity of the known true proteins* and *do not require the model to generalize*, we evaluate the algorithm implemented using the fivefold cross-validation procedure *and* without cross-validation.

Unsupervised Learning PepArML

For tandem mass spectra data sets derived from synthetic protein mixes, training peptide IDs can be deemed true or false based on the known protein content of the samples. In practice, however, the proteins of a sample are not known in advance. We have developed an unsupervised training procedure that can be applied in this case.

The unsupervised training procedure relies on two key observations. First, typically, many of the proteins in a sample can be confidently identified without fancy result combiners or machine learning, even if some of its peptides or spectra cannot. Second, machine learning techniques can often be successfully trained even if the training labels contain some errors. We use an iterative procedure in which putative true-positive proteins are selected using the input search engine results. The putative true-positive proteins are used to label peptide IDs, as if the selected proteins were known to be correct. Machine learning is then carried out and peptide IDs predicted true or false with some prediction confidence. The predictions, and their confidence values, are then used to select a new putative true-positive protein set, and the procedure is iterated until convergence.

Figure 1 shows a schematic representation of unsupervised PepArML training. We outline each step of the unsupervised learning procedure in more detail.

Select putative true-positive proteins by consensus First, consensus peptide IDs for which all search engines agree are identified. Proteins containing at least three nonoverlapping consensus peptides are selected. When consensus peptides are contained in more than one protein, all proteins meeting the criteria are selected. The selected proteins become the initial putative true-positive protein set.

Iteratively refine the putative true-positive proteins Given putative true-positive proteins, all peptide IDs associated with these proteins are labeled true, with all other peptide IDs labeled false. A (new) classifier is trained on the labeled data as previously described. Target and decoy peptide ID predictions are obtained from the classifier, and estimated FDR computed for each prediction confidence value. Peptide IDs confidently predicted as true with estimated FDR of at most 10% can then be determined. Proteins with at least two such distinct peptides are selected for addition to the putative true-positive pro-

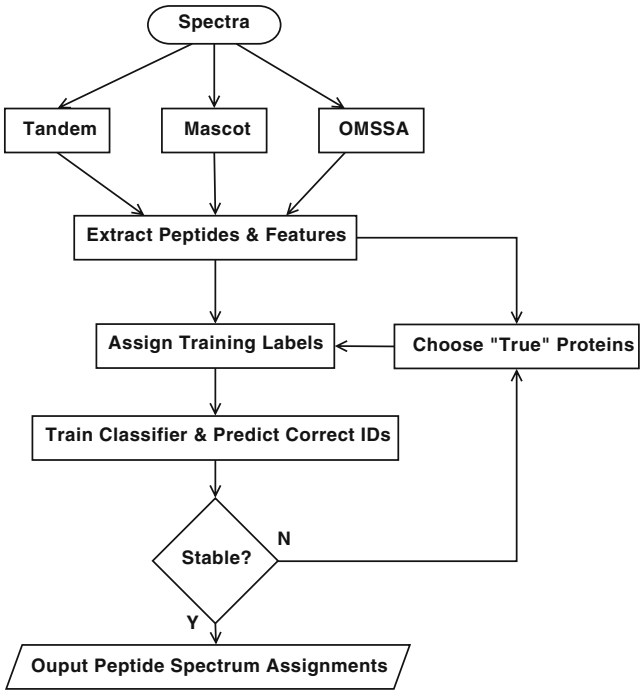

Fig. 1 Schematic representation of unsupervised PepArML training

tein set. Previously selected proteins with no confidently identified peptide IDs are removed from the putative true-positive protein set.

Termination The iterative procedure continues until the putative true-positive proteins do not change. The parameters used to select the putative true-positive proteins are intended to be somewhat conservative and are similar to those required by some journals for the publication of protein identification results. These settings work well for each of our data sets, terminating in three to four iterations in most cases. We stress that the putative true-positive proteins need not be completely correct in order for this iterative training procedure to be successfully carried out, though we find that conservative putative true protein selection parameters work better than aggressive parameters. In the "Results and Discussion" section, we show that this procedure can robustly handle omission of some true-positive proteins.

Performance Evaluation

We use a number of standard performance metrics to evaluate the peptide ID predictions from the search engines, the voting heuristic, and the machine learning combiner, by reference to the *ground truth* for each data set (correctness of each peptide ID as determined by the corresponding synthetic protein mixture). Given predictions on the feature vectors comprising a data set, we select a subset of the predictions by thresholding some confidence value, such as the search engine E-value or classification confidence. For the selected predictions, we can count the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) with respect to the known ground truth.

Sensitivity, defined as TP/(TP+FN), measures the proportion of known true peptide IDs that are correctly selected. Sensitivity is also known as true-positive rate. Specificity, defined as TN/(FP+TN), measures the proportion of known false peptide IDs that are correctly not selected. The false-positive rate is defined as (1−specificity). The true FDR, defined as FP/(FP+TP), measures the proportion of selected peptide IDs that are known to be false. To compute true FDR requires knowledge of ground truth, which is not available for real data sets. We estimate the FDR using decoy peptide IDs. The total number of selected peptide IDs above a confidence threshold (number of positives—NP), which is equal to (FP+TP), can be determined without knowledge of the ground truth. The number of decoy peptide IDs above a confidence threshold (decoy positives—DP) represents an estimate of FP, and furnishes estimated FDR, defined here as DP/NP.

We use a number of different metrics to evaluate each method of predicting the true peptide IDs:

- Receiver operating characteristic (ROC) curves. ROC curves are graphical plots of the sensitivity vs. (1-specificity) for all possible threshold values. Equivalently, ROC curves plot the true-positive rate vs. the false-positive rate. ROC curves provide a global view of the sensitivity/specificity trade-off of a particular predictor, evaluating the predictor's ability to separate true from false peptide IDs.
- Area under ROC (AUROC). AUROC is a single-value summary of a ROC curve, where higher values are desirable. A predictor that separates true and false peptide IDs perfectly will have a square ROC curve and an AUROC of 1.
- Sensitivity for given true FDR. Specificity-based performance metrics can sometimes give a misleading impression as to the performance of a predictor when the number of false data points is much larger than the number of true data points, as is the case for our peptide ID data sets. We compute the sensitivity at 10% true FDR.
- Sensitivity for given estimated FDR. The sensitivity at a given true FDR is uncomputable for data sets for which the ground truth is not available. Therefore, we evaluate the sensitivity of each predictor

at a threshold corresponding to a fixed estimated FDR level of 10%, to better evaluate the predictor performance as it would be used on a real data set.

All performance metrics other than the ROC curves are recorded for 20 independent trials, and presented as boxplots to show the distribution of values arising from the random behavior of the cross-validation, down-sampling, and random forest classifier training. ROC curves are plotted for one representative trial.

In order to benchmark our results against other machine-learning techniques, we also evaluated the performance of the Peptide Prophet algorithm (part of the TPP package, version 4.1.1) applied to the Mascot peptide IDs. However, due to the way in which the Peptide Prophet algorithm is implemented, we cannot compute estimated FDR via the decoy search method. Therefore, for Peptide Prophet results, we use $(1 - p)$, where p is the Peptide Prophet probability, as the estimated FDR. We note that, while this reflects the likely use of Peptide Prophet on a real data set, it creates a potential apples-to-oranges comparison of sensitivity at 10% estimated FDR in the results to follow.

Results and Discussion

Search Engine and Voting Heuristic Performance

At first glance, the labeling methodology described in the "Experimental Procedures" section appears to create data sets too easily classified since all true peptide identifications must appear as the top-ranked peptide ID in at least one search engine's results. Nevertheless, a cursory examination of each data set shows considerable room for improvement over Tandem, Mascot, or OMSSA alone. Table 2 shows the sensitivity of each search engine at 10% estimated FDR.

We observe that sensitivity varies wildly across search engines and data sets, with a different search engine achieving the best sensitivity for each data set. We observe that a significant number of the true peptide IDs cannot be recovered, even at this generous significance cutoff. Examining the peptide identification results more carefully, we observe that a significant number of true peptide IDs have poor individual search engine scores, a setting in which we hope that search engine agreement might help distinguish true from false peptide identifications. Finally, we see that the voting heuristic dominates the best individual search engine for each data set, though only by a few percent.

Table 2 Sensitivity (%) of Tandem, Mascot, OMSSA, and Voting heuristic at 10% estimated FDR

	Tandem	Mascot	OMSSA	Voting heuristic
S17	42.7	**46.1**	43.6	52.7
AURUM	**80.2**	74.1	68.9	82.0
OMICS	41.5	46.7	**56.9**	59.6

Best single search engine sensitivity for each data set indicated in boldface

Supervised Learning PepArML Performance

Next, we examine the ability of machine learning to combine search engine scores and features. Classifier and search engine ROC curves for each data set are shown in Figs. 2 (OMICS), S1 (S17), and S4 (AURUM), using six ROC plots for each data set. In each plot, the y axis represents true-positive rate (sensitivity), and the x axis represents false-positive rate (1-specificity). The three plots on the left of each figure present ROC curves for classifiers C-T, C-M, and C-O (solid lines) and for search engines Tandem, Mascot, and OMSSA (dotted lines). The three plots on the right of each figure present ROC curves for classifiers C-TM, C-MO, and C-TO (solid lines). ROC curves for C-TMO (dash-dotted lines) and the voting heuristic (dashed lines) are included in all plots for comparison.

Classifier and search engine sensitivity (percent) for each data set at 10% estimated FDR is shown in Fig. 3. Sensitivity at 10% and 20% true and estimated FDR is shown for each data set in Figs. S2, S5, and S7. Sensitivity at 5%, 10%, and 15% false-positive rate and AUROC is shown in Figs. S3, S6, and S8. Classifiers are arranged along the x axis, with boxplots summarizing the results of 20 independent trials. The performance of Peptide Prophet applied to Mascot results (denoted M*) is also shown next to the Mascot boxplot.

A thoughtful evaluation of these performance plots provides considerable insight into the relative performance of individual search engines, voting heuristics, and machine learning for peptide identification.

First, the application of the machine learning to individual search engine scores improves performance considerably. In some cases, such as the OMICS data set, the improvement is dramatic, with C-T yielding 420 additional peptide identifications at 10% estimated FDR, a 35% increase over Tandem's E-value alone. In other cases, machine-learning applied to individual search engine scores appears to have little benefit. Furthermore, the relative benefit is not consistent across data sets, with C-O showing little improvement over OMSSA's E-value for data set OMICS, but significant improvement for data set S17.

Surprisingly, Peptide Prophet applied to the Mascot results generally reduces the sensitivity at fixed estimated FDR. We conjecture that the reduced performance may be due the failure of Peptide Prophet to fit models in data sets S17 and OMICS to charge state 1+ and 3+, and 1+, spectra, respectively. However, similarly degraded performance is seen for the AURUM data set, which consists entirely of 1+ MALDI spectra. It is also possible that more conservative estimated FDR values are responsible, but a similar pattern is observed with respect to true FDR and the AUROC. Our application of Peptide Prophet to the Mascot results used the most recent release of the TPP (version 4.1.1) and appropriate settings as described in the documentation—and as such, we believe these results are representative of what would be obtained on real data sets.

Second, combining results from multiple search engines using machine learning is very effective. Examining the performance of C-TMO, C-TM, C-TO, and C-MO, we see that these classifiers generally achieve much better sensitivity than the voting heuristic. Interestingly, while classifiers combining pairs of search engine results generally dominate their constituent single search engine classifiers, sometimes a single search engine classifier is so good that it beats the classifier that combines the two other search engines. Classifier C-T demonstrates this for data set AURUM. Nevertheless, we see that, when we observe the performance over 20 independent runs, the classifier C-TMO significantly outperforms all of the other supervised learning classifiers, the voting heuristic combiner, and the individual search engines. Compared to the worst search engine E-value for each data set, the C-TMO classifier

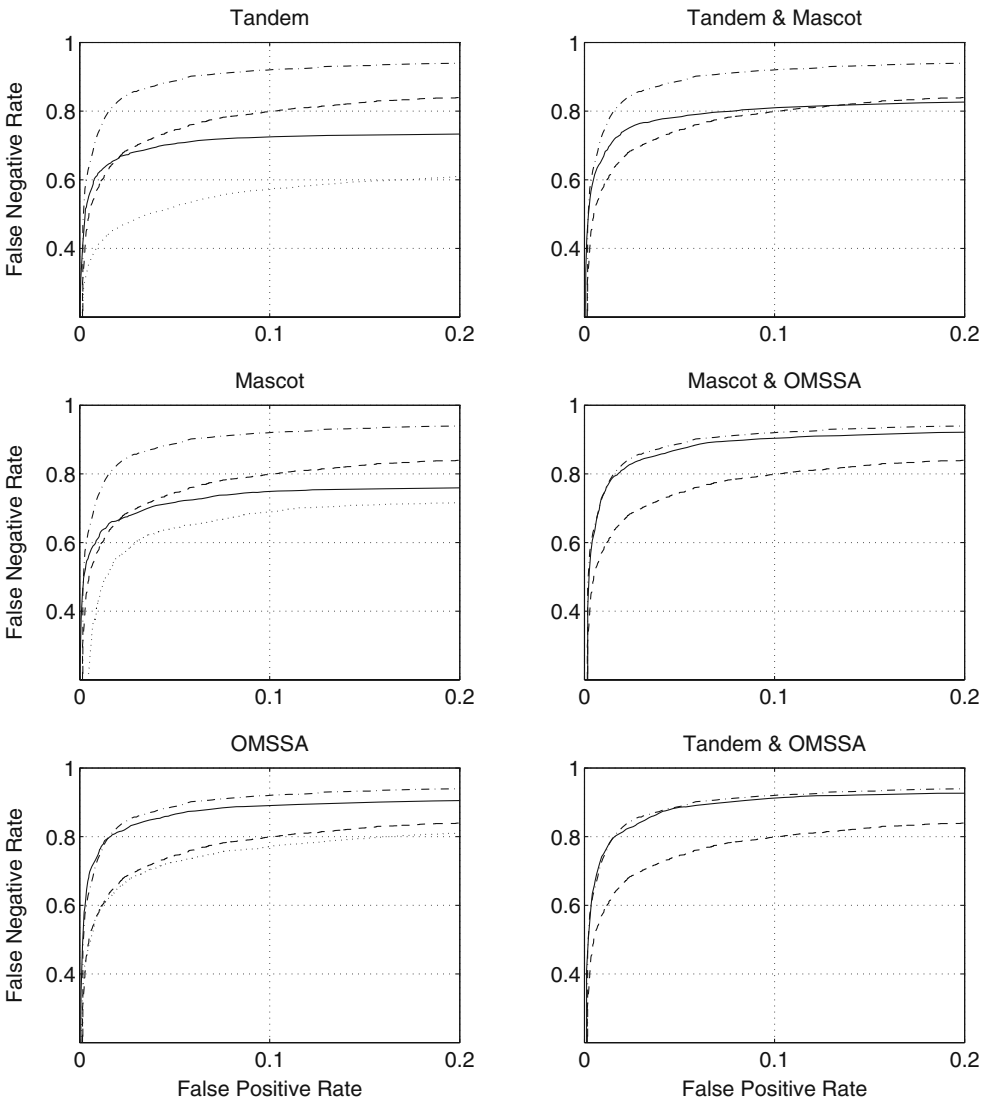

Fig. 2 ROC curves for data set OMICS. Classifiers (*solid line*) and search engines (*dotted line*) are presented in each plot. C-TMO (*dash-dotted line*) and voting heuristic combiner (*dashed line*) are included in all plots

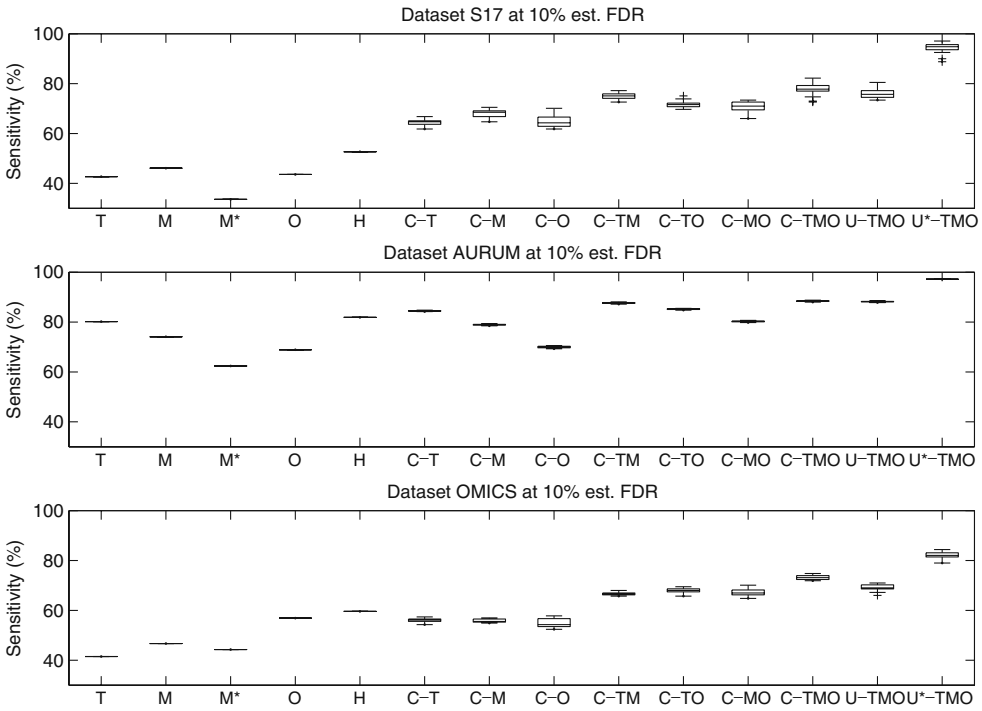

Fig. 3 Sensitivity (%) at 10% estimated FDR for data sets S17, AURUM, and OMICS. Search engines Tandem, Mascot, and OMSSA are denoted *T*, *M*, and *O*; Peptide Prophet applied to Mascot results is denoted *M**; heuristic combiner is denoted *H*; supervised machine-learning classifiers (using fivefold cross validation) are denoted *C-T* through *C-TMO*; and unsupervised machine-learning combiner using fivefold cross validation and no cross validation are denoted *U-TMO* and *U***-TMO*. Boxplots summarize 20 independent runs

improves the sensitivity at 10% estimated FDR from 43% (Tandem) to 78% for data set S17, from 69% (OMSSA) to 88% for data set AURUM, and from 42% (Tandem) to 73% for data set OMICS.

Third, in a number of cases, combining the results of two search engines performs almost as well as combining the results of all three search engines. In particular, the performance of C-TM is generally quite close to that of C-TMO, except for data set OMICS, where the performance of C-TO is closest to the performance of C-TMO. This is particularly interesting in the case of C-TO, which is comprised of the results of two free, open-source search engines. It is not unexpected that the use of three search engines results in marginal improvement over the results of two search engines in some cases, though it is unclear which pair of search engines to choose a priori to achieve the best results. The C-TMO classifier is consistently better than the paired classifiers across all data sets.

Fourth, it is clear that the randomized cross-validation, down-sampling, and random forest construction results in some variation in the resulting performance of the classifiers, and that this variation is greatest for the machine-learning combiners with more features. For similarly performing classifiers, this variation can sometimes change their relative performance ranking, but the essential characteristics of relative performance of the classifiers is retained.

Fifth, despite the rather small size of data set S17, with each cross-validation fold containing around 200 true-positive peptide IDs, the random forest training procedure has no trouble constructing a good peptide identification predictor. In fact, from these results, it seems likely that the characteristics of the underlying data set are more relevant to the performance of the trained predictor, as the AURUM results (large data set) look more impressive than the results for the small S17 data set or the very large OMICS data set. The very high sensitivity for the AURUM data set may also be in part due to the use of IPI Human rather than SwissProt for searching, as the smaller database is less likely to contain high scoring random or homologous peptides. Nevertheless, it is clear from the S17 results that classifiers can be reliably trained on small spectral data sets.

Supervised Learning Feature Evaluation

We assess the discriminating ability of each element of the feature vector using the information gain metric. The information gain of a feature measures the reduction in entropy or randomness when the data set is subdivided according to the feature's values. Information gain, called *InfoGain* by *Weka*, is one of many good techniques for assessing the relative importance of each feature in good classifiers, and it is a property of the data set, independent of the particular classification algorithm used. Larger InfoGain values indicate that a feature is likely to be more useful in making accurate predictions.

Table 1 shows InfoGain values computed by Weka for each feature of the feature vector. As might be expected, each search engine's E-value is very good at distinguishing true from false peptide identifications, although the relative importance of the different search engines' E-values vary considerably across data sets. We note that the variation in InfoGain values is *not* due to random variation induced by the down-sampling and cross-validation as the standard-deviation of the largest InfoGain values, computed over ten observations of five cross-validation folds, is less than 0.015 for the small data set S17 and less than 0.005 for the larger data sets OMICS and AURUM.

The number of agreeing search engines is also very powerful, providing significant support for the use of consensus or voting heuristics in peptide identification, although, again, the relative importance varies with data set. The utility of the model-free machine-learning technique becomes apparent when examining the other, less obvious features. The mass difference between experimental and theoretical peptide molecular weight is quite powerful in data sets S17 and AURUM, but much less so in data set OMICS, perhaps reflecting instruments providing more accurate precursor measurements. Surprisingly, the peptide molecular weight is a good indicator of true vs. false peptide identifications in data set OMICS, which typically occurs when the data set contains many spectra matching very short, rarely correct, peptides.

As expected, a number of features, such as peptide length and number of tryptic termini, are uninformative given the search parameters used. The tiny amount of signal (InfoGain is non-zero, but less than 0.01) attributable to the tryptic N-terminal feature is due to the differences in search engines' definition of tryptic peptides—Mascot and OMSSA consider the peptide at the N-terminus of a protein, with the initial methionine removed, to be a tryptic peptide, while Tandem considers this a semitryptic peptide. Unfortunately, these types of search engine quirks can reduce the number of consensus peptide IDs and make search engine combiners work much harder to discern true from false peptide IDs.

Unsupervised Learning PepArML Performance

Table 1 shows that there is considerable variation in InfoGain and relative rank for each feature between data sets. This suggests that a classifier trained on one data set may perform poorly when applied to another. To understand the possible extent of the problem, we applied the classifier trained on data set S17 to a data set from a Thermo Finnigan LTQ instrument and evaluated the results (data not shown). We found that the S17 classifier generalized extremely poorly, performing worse than individual search engine E-values. Upon reflection, we realized that, not only would the machine learning algorithm chose suboptimal features for a data set from a different instrument, but the weights and thresholds estimated by training would not be valid if the characteristics of the feature vector changed significantly. To understand this phenomenon, recall that E-values change with sequence database size and precursor mass tolerance and that search engine scores tend to depend heavily on the instrument fragment mass tolerance. To make a machine learning approach useful in practice, we must be able to train on the specific characteristics of each individual data set, despite the lack of ground truth for its peptide IDs. The unsupervised PepArML training procedure is able to make an educated, and in practice, quite good, guess at the true labels for each peptide ID, making it possible to train a classifier. Iteration is merely an attempt to refine the guessed labels.

In order for the unsupervised training procedure to be useful in practice, we need to show that its performance is comparable to that of the supervised training procedure. The left side of Fig. 4 shows the ROC curves of the supervised C-TMO classifier (dash-dotted line), voting heuristic (dashed line), unsupervised learning classifier with fivefold cross-validation U-TMO (solid line), and unsupervised learning classifier with no cross-validation U*-TMO (dotted line) applied to each data set. Figure 3 shows the percent sensitivity at 10% estimated FDR for U-TMO and U*-TMO in comparison to supervised learning classifiers. The ROC curves demonstrate almost no difference between the supervised and unsupervised learning under the same cross-validation regime, with a significant performance boost once we use the unsupervised learning procedure to *fit* a predictor to the data, rather than asking it to generalize. The sensitivity boxplots indicate that there is a small penalty in using the unsupervised learning procedure under the fivefold cross validation regime, though the U-TMO classifier still bests most of the other supervised learning classifiers. As with the ROC curves, it is clear that, once we *fit* the predictor with respect to our putative true proteins, we can boost the classification performance beyond that of supervised learning.

The success of the unsupervised learning procedure, as good or better than the supervised learning classifiers on each of these data sets, suggests that it is able to select proteins so the resulting true/false labels approximate the supervised learning labels well. It is not unreasonable to suppose that protein sequence homology or peptide sequence redundancy might dis-

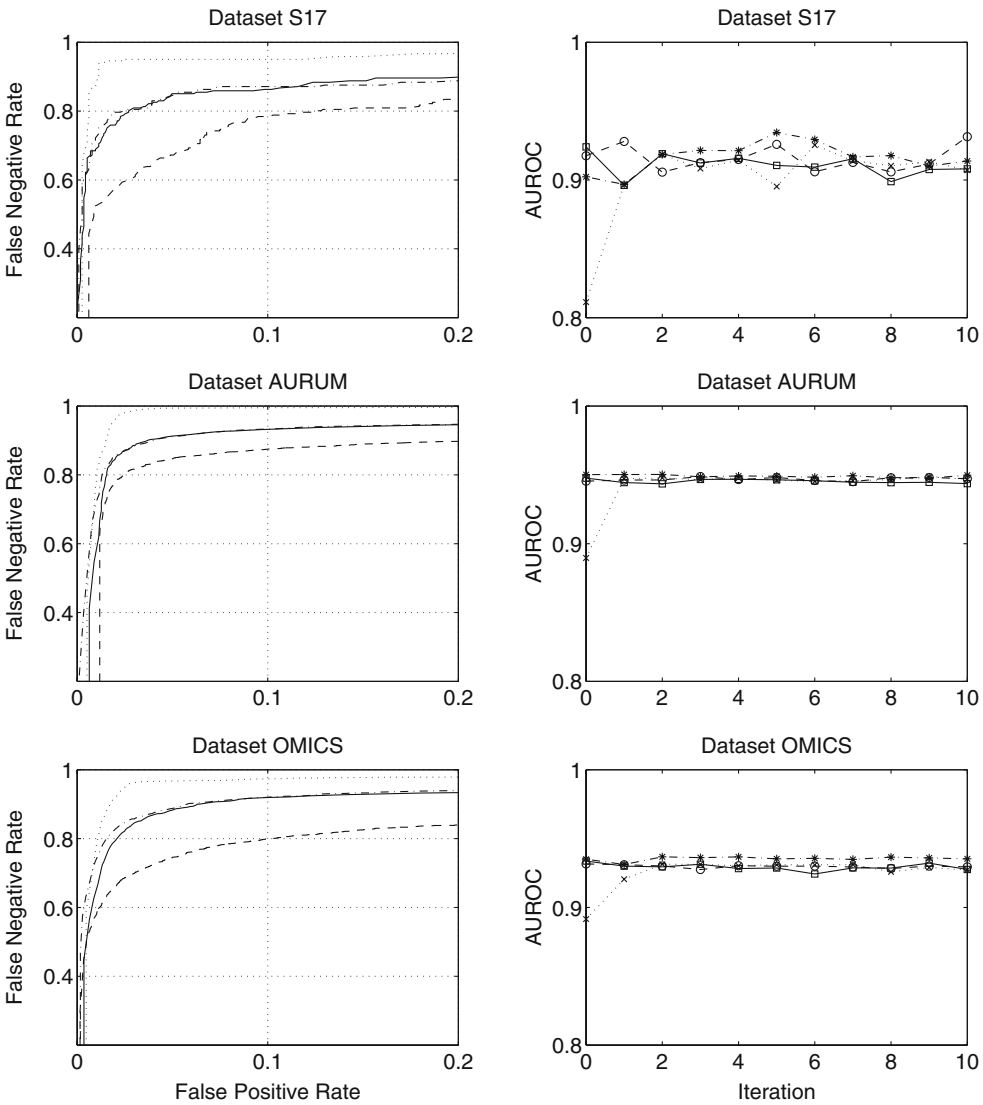

Fig. 4 *Left column*: ROC curves for data sets S17, AURUM, and OMICS—C-TMO (*dash-dotted line*), heuristic combiner (*dashed line*), U-TMO (*solid line*), and U*-TMO (*dotted line*) shown. *Right column*: AUROC after each training iteration for data sets S17, AURUM, and OMICS—true protein labels at every iteration (*dash-dotted* and *asterisks*), true protein labels for initialization only (*solid* and *boxes*), heuristic protein labels for initialization (*dashed* and *circles*), and single best consensus protein labels for initialization (*dotted* and *crosses*)

rupt the unsupervised learning procedure. To test this, we searched data sets S17 and OMICS against SwissProt, which contains considerable sequence homology. Despite the inclusion of homologous proteins in the putative true protein set, the unsupervised learning achieved similar performance to the supervised learning classifier whose true protein set did not include homologous proteins.

We also checked whether data sets representing a large number of proteins with relatively few peptide IDs per protein might derail the unsupervised learning procedure. The AURUM data set is one such data set, containing the MS/MS spectra of peptides from hundreds of proteins. The unsupervised procedure had no trouble labeling the peptide IDs well enough to achieve good performance on this data set either.

We checked whether peptide IDs from known present low-abundance proteins with just one high-quality peptide ID would be lost due to the unsupervised learning procedure. We found that high-quality "one-hit-wonder" peptide IDs are correctly given high confidence predictions by the unsupervised learning classifier, as these represent a relatively small number of incorrectly labeled training examples and do not prevent the machine-learning algorithm from building a good predictor.

Unsupervised Learning Initialization and Convergence

While the previous section demonstrates that the unsupervised learning PepArML procedure achieves similar performance to the supervised learning approach on all three data sets, its rate of convergence and tolerance for a poor initial putative true protein set needs to be established. If too many iterations are required, or if small errors in the initial protein set result in

nonsense predictions, then the appropriateness of the technique for real experimental data sets could be in doubt.

Figure 4, right column, demonstrates the robustness of the unsupervised learning procedure to the initial protein set and its stability in comparison to supervised learning. The x axis represents the training iteration of the unsupervised procedure. The y axis represents the AUROC of the classifier after each iteration, evaluated with respect to the known true protein set. We conduct ten iterations of the machine-learning training algorithm for U-TMO (fivefold cross-validation) under various peptide ID-labeling regimes. First, we use the true protein labels at each iteration (dash-dotted and asterisks). Second, we initialize with true protein labels, and then use the unsupervised learning procedure to update the protein set (solid and box). Third, we use the unsupervised learning procedure initial protein set and update (dashed and circle). Finally, we use the single putative true protein with the most consensus peptide IDs as the initial protein set and update using the unsupervised learning procedure (dotted and crosses). It is clear that there is little difference, for these data sets, between ten iterations of supervised learning and the unsupervised learning procedure applied as described in the "Experimental Procedures" section. Even when we initialize with just one protein, the predictor is able to achieve an AUROC value equivalent to the supervised learning classifier after just one iteration. Generally, the unsupervised learning procedure terminates in about three iterations for these data sets. As demonstrated in Fig. 4, the unsupervised learning procedure is able get started even if only a small number of the true-positive peptide IDs can be correctly labeled.

Conclusions

In this paper, we have demonstrated a highly sensitive and specific new technique for separating true from false peptide identifications on three synthetic protein mixture data sets from both electrospray and MALDI instruments. PepArML uses machine learning to combine the search results from many search engines, achieving better results than machine-learning or result combining alone. PepArML uses the model-free random forest machine learning technique, which ensures that the relative contributions of search engine agreement and peptide–spectrum match scores can be used optimally for each combination of spectra, search engines, and search engine parameters. We have demonstrated that PepArML can be trained effectively, with no loss of performance, in an unsupervised manner, making it possible to learn the properties of each data set on the fly. Unsupervised PepArML removes the need for extensive libraries of pretrained models based on experimental spectra from synthetic protein mixtures from all instrument, search engine, and parameter combinations. Unsupervised PepArML training also alleviates concerns about suboptimal machine learning models being applied beyond their ability to generalize effectively.

PepArML does not rely on specialized features or difficult-to-compute scores, but these can be easily added to the model if desired. Similarly, PepArML can be used with any number of different search engines, or even multiple searches from the same search engine. The model-free nature of PepArML combining even makes it possible to combine results from disparate peptide identification techniques, such as spectral matching, alongside search engine results, or to use paired searches with conservative and aggressive search parameters.

The unsupervised training procedure could be manipulated in a variety of ways to exercise more control over PepArML learning. Users could hand-select the initial set of putative true proteins, or apply a species constraint to the putative true protein set, if the sample is known to come from a particular organism. It would also be straightforward to incorporate peptide IDs to decoy peptides as known false identifications just as other semisupervised learning approaches have done. However, we believe that the addition of known false peptide IDs is less powerful in this context than correctly guessing true labels on a smaller number of spectra.

The excellent performance of PepArML applied to the results of Tandem and OMSSA, both open-source, freely available search engines, raises the tantalizing possibility that a PepArML-based metasearch engine might offer superior identification performance than costly commercial search engines. Such a metasearch engine could wrap Tandem, OMSSA, and other free search engines behind a single user interface.

It remains to be seen whether the iterated unsupervised learning procedure proposed here can be applied, as described, across the rich variety of experimental data sets. Our experiments to investigate the robustness of the procedure are encouraging, but it is possible our simple technique for selecting putative true proteins, in particular, may find too few true-positive proteins for successful training. In future work, we plan to investigate whether protein identification tools, such as Protein Prophet [28], or an E-M approach, which would fit naturally into the iterative PepArML framework,

might provide additional robustness for PepArML when applied to experimental data sets.

We believe that the PepArML machine-learning framework has the potential to solve the critical problems of combining multiple search engine results and the use of machine-learning tools beyond training data sets, resulting in robust, effective, and reliable tools for peptide spectrum assignment.

Acknowledgements This work is supported by NIH/NCI Grant R01 CA126189. We thank Brian Balgley and Paul Rudnick for the use of an unpublished synthetic protein mixture MS/MS data set used in early PepArML development.

References

1. Yates JR. Mass spectrometry and the age of the proteome. J Mass Spectrom. 1998;33(1):1–19.
2. Washburn MP, Wolters D, Yates JR. Large-scale analysis of the yeast proteome by multidimensional protein identification technology. Nat Biotechnol. 2001;19(3):242–7.
3. Yates JR, Eng JK, Clauser KR, Burlingame AL. Search of sequence databases with uninterpreted high-energy collision-induced dissociation spectra of peptides. J Am Soc Mass Spectrom. 1996;7:1089–98.
4. Perkins DN, Pappin DJ, Creasy DM, Cottrell, JS. Probability-based protein identification by searching sequence databases using mass spectrometry data. Electrophoresis. 1999;20: 3551–67.
5. Zhang W, Chait BT. Profound: an expert system for protein identification using mass spectrometric peptide mapping information. Anal Chem. 2000;72(11):2482–9.
6. Colinge J, Masselot A, Giron M, Dessingy T, Magnin J. Olav: towards high-throughput tandem mass spectrometry data identification. Proteomics. 2003;3(8):1454–63.
7. Bafna V, Edwards NJ. SCOPE: a probabilistic model for scoring tandem mass spectra against a peptide database. Bioinformatics. 2001;17:13–21.
8. Craig R, Beavis RC. Tandem: matching proteins with tandem mass spectra. Bioinformatics. 2004;20:1466–7.
9. Craig R, Beavis RC. A method for reducing the time required to match protein sequences with tandem mass spectra. Rapid Commun Mass Spectrom. 2003;17;2310–6.
10. Geer LY, Markey SP, Kowalak JA, Wagner L, Xu M, Maynard DM, Yang X, Shi W, Bryant SH. Open mass spectrometry search algorithm. J Proteome Res. 2004;3: 958–64.
11. Zhang N, Aebersold R, Schwikowski B. ProbID: a probabilistic algorithm to identify peptides through sequence database searching using tandem mass spectral data. Proteomics. 2002;2(10):1406–12.
12. MacCoss MJ, Wu CC, Yates JR. Probability-based validation of protein identifications using a modified SEQUEST algorithm. Anal Chem. 2002;72(21):5593–9.
13. Tabb DL, Fernando CG, Chambers MC. Myrimatch: highly accurate tandem mass spectral peptide identification by multivariate hypergeometric analysis. J Proteome Res. 2007; 6(2):654–61.
14. Elias JE, Haas W, Faherty BK, Gygi SP. Comparative evaluation of mass spectrometry platforms used in large-scale proteomics investigations. Nat Methods. 2005;2:667–75.
15. Higgs RE, Knierman MD, Bonnerfreeman A, Gelbert LM, Patil ST, Hale JE. Estimating the statistical significance of peptide identifications from shotgun proteomics experiments. J Proteome Res. 2007;6(5):1758–67.
16. Keller A, Nesvizhskii AI, Kolker E, Aebersold R. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. Anal Chem. 2002;74(20):5383–92.
17. Moore RE, Young MK, Lee TD. Qscore: an algorithm for evaluating SEQUEST database search results. J Am Soc Mass Spectrom. 2002;13(4):378–86.
18. Ulintz PJ, Zhu J, Qin ZS, Andrews PC. Improved classification of mass spectrometry database search results using newer machine learning approaches. Mol Cell Proteomics. 2006;5(3):497–509.
19. Anderson DC, Li W, Payan DG, Noble WS. A new algorithm for the evaluation of shotgun peptide sequencing in proteomics: support vector machine classification of peptide MS/MS spectra and SEQUEST scores. J Proteome Res. 2003; 2(2):137–46.
20. Baczek T, Bucinski A, Ivanov AR, Kaliszan R. Artificial neural network analysis for evaluation of peptide MS/MS spectra in proteomics. Anal Chem. 2004;76(6):1726–32.
21. Resing KA, Meyer-Arendt K, Mendoza AM, Aveline-Wolf LD, Jonscher KR, Pierce KG, Old WM, Cheung HT, Russell S, Wattawa JL, Goehle GR, Knight RD, Ahn NG. Improving reproducibility and sensitivity in identifying human proteins by shotgun proteomics. Anal Chem. 2004;76 (13);3556–68.
22. Searle BC, Turner M, Nesvizhskii AI. Improving sensitivity by probabilistically combining results from multiple MS/MS search methodologies. J Proteome Res. 2008;7(1):245–53.
23. Choi H, Nesvizhskii AI. Semisupervised model-based validation of peptide identifications in mass spectrometry-based proteomics. J Proteome Res. 2008;7(1):254–65.
24. Käll L, Canterbury JD, Weston J, Noble WS, Maccoss MJ. Semi-supervised learning for peptide identification from shotgun proteomics datasets. Nat Methods. 2007;4(11): 923–5.
25. Falkner JA, Veine DM, Kachman M, Walker A, Strahler JR, Andrews PC. Validated MALDI-TOF/TOF mass spectra for protein standards. J Am Soc Mass Spectrom. 2007;18(5): 850–5.
26. Keller A, Purvine S, Nesvizhskii AI, Stolyar S, Goodlett DR, Kolker E. Experimental protein mixture for validating tandem mass spectral analysis. OMICS J Integr Biol. 2002;6(2):207–212.
27. Witten IH, Frank E. Data mining: practical machine learning tools and techniques. 2nd edn. San Francisco: Morgan Kaufmann; 2005.
28. Nesvizhskii AI, Keller A, Kolker E, Aebersold R. A statistical model for identifying proteins by tandem mass spectrometry. Anal Chem. 2003;75:4646–58.