BIOINFORMATICS

Vol. 24 ISMB 2008, pages i348–i356
doi:10.1093/bioinformatics/btn189

Modeling peptide fragmentation with dynamic Bayesian networks for peptide identification

Aaron A. Klammer[1], Sheila M. Reynolds[2], Jeff A. Bilmes[2,3], Michael J. MacCoss[1], William Stafford Noble[1,3,*]

[1]Department of Genome Sciences, [2]Department of Electrical Engineering and [3]Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA

ABSTRACT

Motivation: Tandem mass spectrometry (MS/MS) is an indispensable technology for identification of proteins from complex mixtures. Proteins are digested to peptides that are then identified by their fragmentation patterns in the mass spectrometer. Thus, at its core, MS/MS protein identification relies on the relative predictability of peptide fragmentation. Unfortunately, peptide fragmentation is complex and not fully understood, and what is understood is not always exploited by peptide identification algorithms.

Results: We use a hybrid dynamic Bayesian network (DBN)/support vector machine (SVM) approach to address these two problems. We train a set of DBNs on high-confidence peptide-spectrum matches. These DBNs, known collectively as Riptide, comprise a probabilistic model of peptide fragmentation chemistry. Examination of the distributions learned by Riptide allows identification of new trends, such as prevalent a-ion fragmentation at peptide cleavage sites C-term to hydrophobic residues. In addition, Riptide can be used to produce likelihood scores that indicate whether a given peptide-spectrum match is correct. A vector of such scores is evaluated by an SVM, which produces a final score to be used in peptide identification. Using Riptide in this way yields improved discrimination when compared to other state-of-the-art MS/MS identification algorithms, increasing the number of positive identifications by as much as 12% at a 1% false discovery rate.

Availability: Python and C source code are available upon request from the authors. The curated training sets are available at http://noble.gs.washington.edu/proj/intense/. The Graphical Model Tool Kit (GMTK) is freely available at http://ssli.ee.washington.edu/bilmes/gmtk.

Contact: noble@gs.washington.edu

1 INTRODUCTION

A major goal in biology is the identification and characterization of the cell's entire protein complement, or proteome. Toward this end, tandem mass spectrometry (MS/MS)-based technologies offer the ability to rapidly identify proteins in complex mixtures (Mann *et al.*, 2001; Yates, 1998). An essential step in MS/MS is the fragmentation of a protonated peptide and detection of the resulting fragment ions in the form of a mass spectrum. Due to the complex chemistry of peptide fragmentation, the pattern of peaks in such a spectrum can be predicted only qualitatively: an exact prediction of spectrum peak heights, or even which peaks will be present or absent, has proven elusive (Dancik *et al.*, 1999; Elias *et al.*, 2004; Wan and Chen, 2005; Zhang, 2004).

One motivation for modeling peptide fragmentation is to aid in the assignment of peptide sequences to observed fragmentation spectra. However, because of the complexity of peptide fragmentation, designers of peptide spectrum identification software have often relied heavily on expert knowledge to design simple heuristics (Eng *et al.*, 1994; Field *et al.*, 2002) or to set probabilities within a larger model (Bafna and Edwards, 2001). Only recently models trained on actual mass spectrometry data have been pursued (Dancik *et al.*, 1999; Elias *et al.*, 2004; Havilio *et al.*, 2003; Wan and Chen, 2005). These peptide identification methods typically use relatively simple models that leave out large portions of known fragmentation pathways, such as neutral losses (Elias *et al.*, 2004; Wan and Chen, 2005), or incorporate most of the known fragmentation pathways in a black box model that is not easily interpreted or extended (Zhang, 2004).

The method presented here builds upon and extends this previous work in an effort to address the limitations of existing fragmentation models and search methods. We test two closely related hypotheses: the first is that an improved model of peptide mass spectrum peak intensity, trained on actual MS/MS data, will provide insight into the complex chemistry of protonated peptide fragmentation; the second is that such a model will be useful for improving identification of unknown peptide fragmentation spectra, especially in conjunction with a sequence database search. We address these hypotheses using a machine learning tool known as the dynamic Bayesian network (DBN).

A Bayesian network is a type of graphical model (Lauritzen, 1996), a mathematical tool in which a graph is used to express important factorization properties about families of probability distributions. These properties allow computationally efficient dynamic programming algorithms to carry out important tasks such as parameter estimation and pattern recognition. Without the expression of factorization, such algorithms would be intractable. Bayesian networks also provide a visual, intuitive, yet mathematically formal graphical description of such probabilistic models, something that can be of enormous assistance when designing a model to solve a given problem.

A DBN is a type of Bayesian network (Heckerman, 1995) that is ideally suited to sequential data, such as acoustic speech signals in speech recognition or DNA and protein sequences in biological sequence analysis. Because DBNs subsume hidden Markov models (HMMs), and because HMMs have been widely and successfully used in a variety of sequence analysis tasks, it is likely that the much more powerful family of DBNs may further advance the field of

*To whom correspondence should be addressed.

bioinformatics. A DBN is constructed using a fixed-length *template* which is unrolled in order to model a sequence of any arbitrary length. The fact that the DBN is described using only a finite number of parameters, but can describe a sequence of unbounded length, is one of the powerful aspects of DBNs. A detailed description of the types of DBNs used in this work, but for the problem domain of speech recognition, can be found in (Bilmes and Bartels, 2005). Because of their ability to model large complex phenomena, DBNs are particularly appropriate for modeling peptide fragment ion intensity. In addition, the DBN's governing parameters (which are automatically learned in this work) are highly interpretable, making them well-suited to provide scientific insight.

Our fragmentation model, called *Riptide*, consists of a collection of DBNs that capture physical properties of peptide fragmentation. Riptide's design is motivated by the widely accepted *mobile proton model* of peptide fragmentation (Dongre *et al*., 1996; Wysocki *et al*., 2000). According to this model, peptide fragmentation by collision-induced dissociation under low-energy conditions is caused by migration of a proton to a location on the peptide backbone and subsequent fragmentation of the peptide into *b*- and *y*-ions. This fragmentation event can be influenced by numerous factors (Dongre *et al*., 1996). The most closely studied factor influencing cleavage is the effect of adjacent amino acid residues on the probability of cleavage occurring at a particular backbone amide bond. This effect is modeled in detail in the Riptide DBNs. In addition, the primary fragmentation event into *b*- and *y*-ions (corresponding to sequences N-term and C-term to the fragmentation position) is often accompanied by a number of additional fragmentation events: the formation of an *a*-ion by a loss of carbon monoxide from the *b*-ion; the loss of NH_3 or H_2O (Kinter and Sherman, 2000). Riptide explicitly models these ion formations, both alone and jointly with their precursor ions. This feature is missing in other machine learning approaches, which only model *b*- and *y*-ions (Elias *et al*., 2004; Wan and Chen, 2005).

Riptide's probabilistic parameters are trained from previously identified tandem mass spectra. In order to avoid contamination of the training set with incorrectly identified spectra, we generated high-confidence identifications using a combination of seven peptide-identification algorithms, paying special attention to controlling false discoveries. The resulting collection of 1208 peptide-spectrum matches (PSMs) is freely available at http://noble.gs.washington.edu/proj/intense.

The Riptide model detects both known and potentially novel trends in peptide fragment intensity within these high-quality PSMs. Of these trends, perhaps the most provocative is the tendency towards higher intensity *a*-ion peaks and *a*-ion neutral loss peaks from cleavage sites C-term to hydrophobic residues. In addition to providing scientific insight into peptide fragmentation chemistry, the probabilities assigned by the Riptide models are useful for improving peptide identification. We demonstrate that, when feature vectors comprised of Riptide probabilities are used as input to either an SVM or the semi-supervised learning algorithm Percolator, they improve peptide identification at a 1% false discovery rate (FDR) by 10.9 and 12.4%, respectively.

2 APPROACH

Although the details of the Riptide model are complex, the inputs to and outputs from the Riptide training and testing procedure are quite simple (Fig. 1). We start with a collection of high-confidence PSMs generated as described in Section 3.2. These PSMs are used to train the Riptide model, which consists of a collection of DBNs that model the probability distributions governing peptide fragment ion intensities. The resulting Riptide model is then evaluated on a set of test PSMs, generating for each PSM a feature vector of probabilities. These vectors can then be used as input to analysis software, assigning scores to the PSMs. Examples of analysis software include support vector machines (SVMs) or the semi-supervised learning algorithm Percolator of Käll *et al*. (2007) (Section 4).

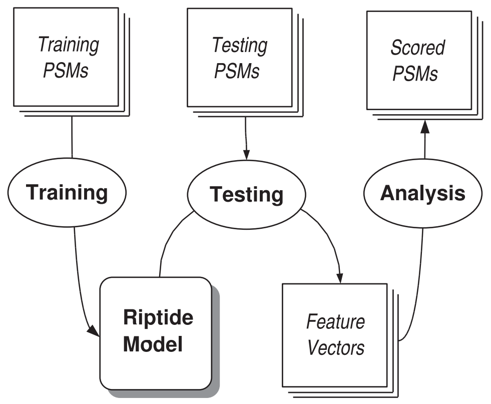


Fig. 1. Experimental overview. We start with a collection of high-confidence PSMs. These training PSMs are used to train the Riptide model, which consists of a collection of DBNs that model the probability distributions governing peptide fragment ion intensities. Riptide is used to evaluate testing PSMs to produce a vector of features for each PSM, each feature related to a probability assigned to the PSM by one of the Riptide DBNs. Finally, these feature vectors can be analyzed by additional algorithms (such as SVMs) to produce scores for the test PSMs.

2.1 Riptide training

Training the Riptide model proceeds in two main steps, portrayed in Figure 2. The first step starts with the high-confidence PSMs, produced as described in Section 3.2. Each of the spectra for these positive PSMs is also associated with a randomly generated peptide to create a set of negative PSMs. We use these two classes of PSMs (positive and negative) to train a set of 'positive' and 'negative' dynamic Bayesian networks (Fig. 2A). These trained DBNs are then used to evaluate the test PSMs, yielding for each PSM and each type of DBN a pair of probabilities (positive and negative). In addition to the raw probabilities, we also include the ratio between them, which we found helped in discrimination (Section 2.3). Thus, each of the original training PSMs is represented as a length three vector, with three scalar values for each kind of DBN in the original Riptide training (Fig. 2B, right). These vectors can then be used during testing as input to either an SVM or the Percolator (Section 4).

2.2 Bayesian networks

At the core of the Riptide algorithm are two types of DBNs that model the probability distributions governing spectrum ion intensities. One section of a DBN template is called a *frame*. Three frames for each model are shown in Figure 3 using standard DBN diagramming semantics. Nodes in the model represent random variables, solid edges signify potential dependencies between these

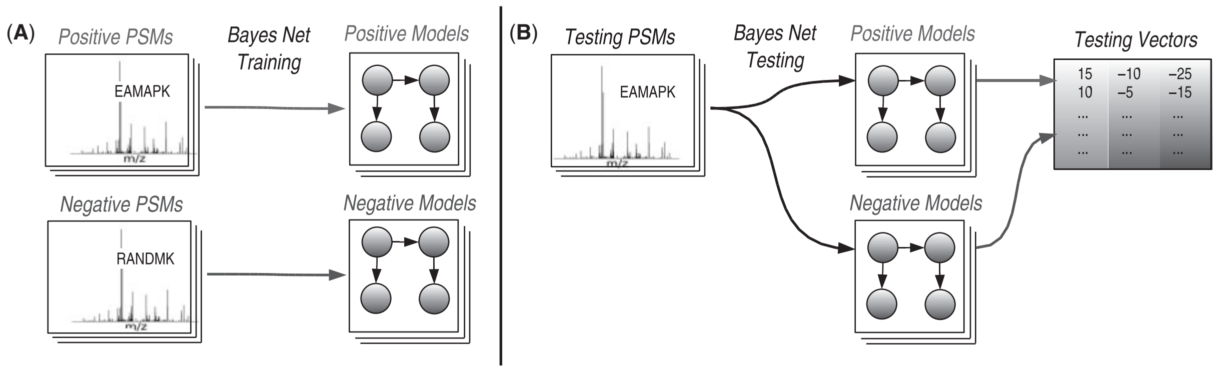


Fig. 2. Riptide overview. The process of training and evaluating Riptide consists of two main stages. (A) The process begins with high-confidence PSMs (positive PSMs, top). Each of the spectra for these positive PSMs is also associated with a randomly generated peptide to create a set of negative PSMs (bottom). These two classes of PSMs are used to train a set of DBNs, one positive and one negative for each ion series. (B) The trained DBNs are then used to evaluate testing PSMs, yielding for each PSM and each type of DBN a pair of probabilities (positive and negative) as well as the ratio between them, a log odds ratio. Thus, each of the testing PSMs is represented as a vector of scalar values, three values for each DBN. These vectors are then used during testing as input to either an SVM or the Percolator algorithm (Section 4).

variables, and dashed edges signify switching edges (Bilmes and Bartels, 2005).

The first network type is the *single-ion* model, which captures information about the influence of peptide chemistry on individual ion series (Fig. 3A). Intuitively, this model learns probability relationships that can be expressed as '*b*-ions N-term to proline tend to have high intensity' or '*y*-ions C-term to aspartate have low intensity'. The second type of network is the *paired-ion* model, which captures information about relationships between pairs of ions of related types (Fig. 3B). These models learn probability relationships that can be expressed as '*b*- and *y*-ions that result from the same cleavage location tend to have intensities of similar height.' Taken together, these two kinds of models are capable of capturing a rich set of probabilistic relationships governing fragment ion intensities. We train specific instances of each model type for different ion series and pairs of ion series. For example, Riptide contains a single-ion model trained on *b*-ions alone, and a paired-ion model trained on *b*- and *y*-ions jointly. Additional details about each of these model types are given subsequently.

2.2.1 Single ion models A graphical representation of the single-ion model is shown in Figure 3A. It models the relationship between a spectrum and a particular series of fragment ions from an associated peptide. Each frame corresponds to a single fragmentation location in the peptide and the intensity of a peak in the mass spectrum associated with that fragmentation. For example, a model of the *b*-ions resulting from the peptide EAMPK would contain four frames, with the first frame corresponding to the *b*-ion resulting from fragmentation at the amide bond between the peptide fragments E and AMPK.

For a given set of training PSMs, one single-ion model is trained for each of the 18 different ion series. These include nine singly charged and nine doubly charged ion series, the latter denoted with a '++'. For each charge state, we model three primary ion series (*b*, *y* and *a*), each of these primary series with a loss of water (denoted b°, y° and a°) and a loss of ammonia (denoted b^*, y^* and a^*). Because we train a separate model on negative and positive sets of PSMs, this procedure results in 36 single-ion models (3 ion types × 3 loss types × 2 charge-states × 2 training sets).

At the center of each frame of the single-ion model is a random variable that represents ion intensity as the percentile rank of the observed peak in the mass spectrum using a number between 0 and 1, derived as described in Section 3.1. We model this variable using a mixture of three Gaussians, for a total of eight free parameters (three means, three variances and two weights). Three Gaussians were found to match natural ion intensity distributions well: most distributions have a single large peak, tapering into a broad background distribution modeled by the other two Gaussians.

The intensity variable is dependent on several other variables, which can be divided into two groups. The first group consists of two variables that influence the probability of a peak being detected in the spectrum. As physical instruments, mass spectrometers are only capable of detecting ions within a finite range of m/z values. Thus, the first variable indicates whether the particular peak is within the detectable portion of the mass spectrum, while the second indicates whether a peak is indeed detected. We do not wish to assign low probability to a peptide with an undetected ion if that ion is physically undetectable; hence, when both of these variables are false, we set the center intensity variable to unity. On the other hand, if an ion is detectable but not detected, then we penalize it during testing by using the (usually low) probability for a zero intensity ion. Finally, if an ion is both detectable and detected, then we use the corresponding intensity value to train (or test) the appropriate one-dimensional three-component Gaussian mixture at the center intensity node.

The second group of three variables influence the intensity of the fragment ion assuming that it is detected. Two variables represent the flanking amino acids immediately to the left (N-term) and right (C-term) of the fragmentation position in the peptide. These positions have been shown to have a strong effect on the probability of detecting an ion resulting from the cleavage position at that position (Tabb *et al.*, 2004; Wysocki *et al.*, 2000). The third variable represents the position of the peak along the m/z axis of the spectrum relative to the intact peptide m/z (using an integer between 0 and 4). This variable accounts for the bias towards center fragmentation

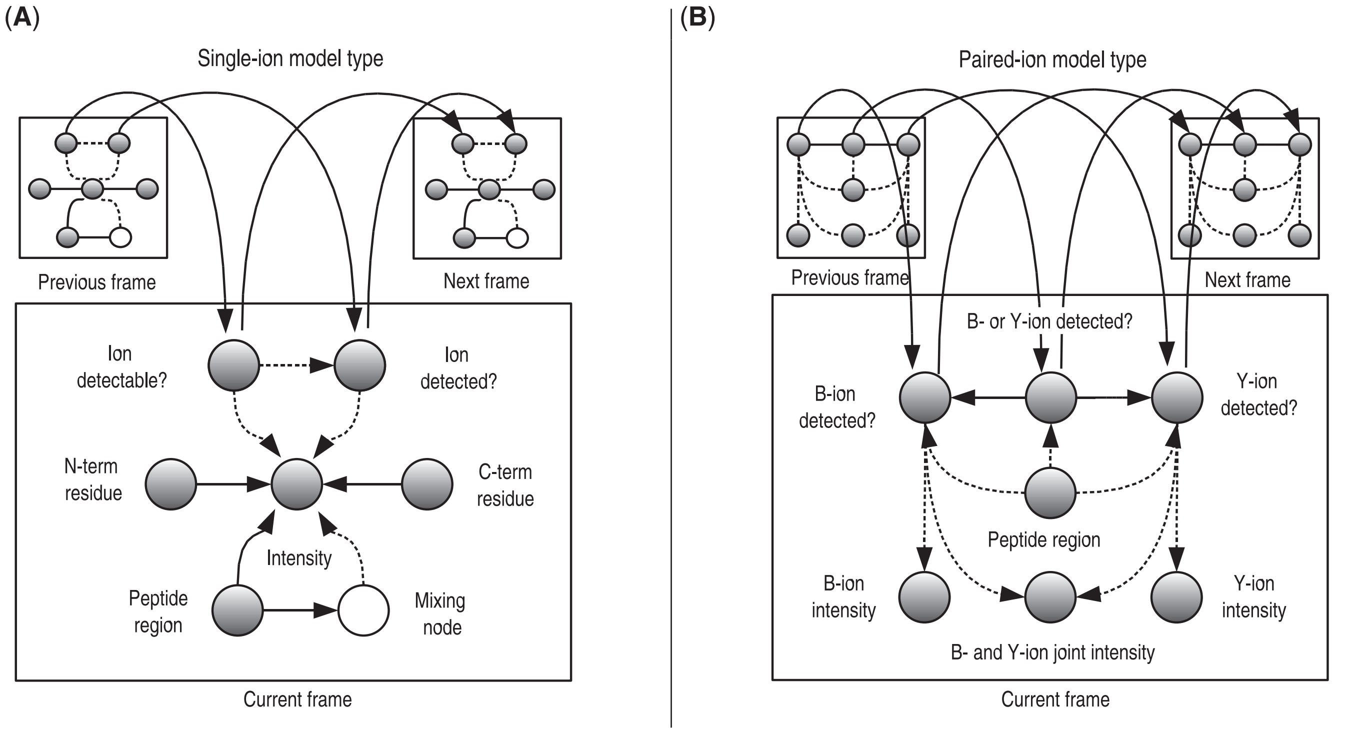


Fig. 3. The two types of Riptide DBNs. The first class of DBNs (A) models distributions of ion fragment intensities individually, conditioning on the amino acids flanking the cleavage site that fragments to produce that particular ion. The second class of DBNs (B) models the distributions of ion fragment intensities in pairs, incorporating dependencies between related ions that result from fragmentation at the same site. Observed nodes are shaded gray; hidden nodes are not shaded. The nodes at the top of each center frame are connected to identical nodes in previous and subsequent frames. Solid lines indicate conditional dependencies, while dashed lines indicate a switching parent relationship, a special form of conditional dependency.

in peptides. These three variables are mixed together using a hidden mixing variable, the distribution of which depends on the region of the mass spectrum. The mixing procedure allows what would be a very high number of training parameters (20 left-flanking residues $\times$ 20 right flanking residues $\times$ 5 peptide regions $\times$ 8 Gaussian parameters $=16000$) to a much smaller number ($(20+20+5)*8=360$), using the switching parent mechanism. Thus, the mixing node allows us to train a much richer model on much fewer data than would otherwise be possible. An edge connects the peptide region to this mixing node because intensities from fragmentations in different regions of the peptide will have slightly different dependencies on flanking residues and peptide position. For example, the extreme C-term and N-term regions of the peptide will tend to have low intensities regardless of flanking residues (Havilio *et al.*, 2003), whereas towards the center of the peptide differences in intensity are more influenced by flanking residues.

2.2.2 Paired-ion models A graphical representation of the paired-ion model type is shown in Figure 3B. This type of model attempts to capture the pairwise relationships between related ions of different types. Some pairs of ion types are closely related because they result from the same fragmentation event (e.g. b_i and y_{n-i}, for a peptide of length n). This is because under low-energy conditions, the b-ion and y-ion fragments co-exist in a loose complex; the two members of this dimer compete for the proton, with assignment of charge being determined by the proton affinities of the two ions (Paizs and Suhai, 2004). Fragment cannot be detected if they are not charged, so this competition matters for the detected ion intensities. Other pairs of ion types are related because one type can produce the other upon secondary fragmentation (b and b°). Still others are related because they represent different charge states of the same ion (b and b^{++}). Thus, we train paired-ion models for each of the following 15 pairs of related ions: b/y, b/b°, y/y°, b/a, b/b^*, y/y^*, b/b^{++}, y/y^{++}, $b^\circ/b^{\circ++}$, $y^\circ/y^{\circ++}$, a/a^{++}, b^*/b^{*++}, y^*/y^{*++}, y/a and b^{++}/y^{++}. For clarity, Figure 3 shows the model for b- and y-ions only; other pairs of ions are modeled analogously. As for the single-ion models, one model is trained on each ion series for a positive and negative set of training PSMs (Fig. 2), producing 30 trained models. Like the single-ion models, each frame in a paired-ion model corresponds to a single fragmentation location in a peptide and models the intensities of a pair of peaks in a mass spectrum associated with that fragmentation. For example, the first frame of a model of the $+1$ b- and y-ions resulting of the peptide EAMPK would model the b-ion E and the y-ion AMPK.

The three variables along the bottom of Figure 2 represent peak intensity. Two of these variables model the intensities of the ions individually and are essentially identical to the intensity variable described above for the single-ion models. These variables also use a mixture of three 1D Gaussians and are used if only one or the other ion is detected but not both. On the other hand, if both ions are detected, then the center variable models the ions jointly using a mixture of nine 2D Gaussians. Regardless of which pattern of ions is detected, the distribution corresponding to the undetected pattern of ions is given a unity score. Whether ions are detected or not is indicated using the three variables directly above each of the intensity variables. Finally, the three variables indicating whether ions are detected or not are dependent on the peptide region, which is identical to the corresponding node in the single ion models. The dependence of these three variables on peptide region captures the fact that some pairs of ion types (b and y) from the center of a peptide are more likely to be observed simultaneously in the spectrum.

2.3 Using Riptide to evaluate PSMs

The final Riptide model consists of 66 dynamic Bayesian networks, including a positive and negative model for each of 18 single-ion series and 15 pairs of ion series ((18 + 15) * 2 = 66). Once these networks have been trained, they can be used to assign a probability to the ion series from any given PSM. Evaluating a PSM using one of the models described above yields the joint probability of the observed values for a particular ion series intensity pattern i and peptide p given the trained model M, $Pr(i,p|M)$. Each ion series has two probabilities assigned to it: one for the model trained with positive PSMs, and one for the model trained with negative PSMs. We use these two probabilities to calculate a log odds ratio for each ion series and PSM. Hence, the final measure of how well a given PSM ion series and for a particular peptide matches expectation is given by

$$LOR(i,p) = \log\left(\frac{\Pr(i,p|M_+)}{\Pr(i,p|M_-)}\right), \tag{1}$$

where M_+ and M_- are the positive and negative models, respectively. Evaluating the log odds ratios for each of the 33 positive and 33 negative models yields a vector of an additional 33 values for each PSM. Thus, the final vector summarizing each PSM is 99 elements long. We use these vectors as input to other algorithms such as SVMs, described in more detail in Section 4.

3 METHODS

3.1 Spectrum preprocessing

Before any particular spectrum is analyzed, we transform the intensities by sorting the peaks in ascending order by intensity and calculating, for that peak, the fraction of peaks that are less than or equal to that intensity. We use these fractional representations of peak intensities to train the Riptide dynamic Bayesian networks. Thus, in a hypothetical spectrum with 10 peaks, the peak with the highest intensity would be assigned a normalized rank of 1.00, and the peak with the lowest intensity would be assigned a normalized rank of 0.10. This intensity transformation is similar to the relative rank value used in (Wan and Chen, 2005) and reduces the effect of variations in dynamic range and noise.

3.2 Training dataset

To generate the MS/MS data, an aqueous soluble protein sample from *Escherichia coli* lysate was reduced, carbamidomethylated and digested with trypsin in the presence of an acid-labile detergent (RapiGest, Waters Corp., Milford, MA). The resulting peptides were analyzed by μLC–MS/MS using the multi-dimensional protein identification technology (Washburn *et al.*, 2001) on a ThermoFinnigan Orbitrap LTQ mass spectrometer, yielding a total of 112 329 spectra.

We wanted to avoid learning spurious relationships from PSMs with MS/MS spectra that contained heterogeneous populations of peptides. To select spectra from homogeneous populations of peptides, we used the isotope detection algorithm HardKlör (Hoopmann *et al.*, 2007). We included MS/MS spectra that had an associated MS spectrum with only one isotope distribution within a window of 3 m/z of the precursor ion over three out of four MS spectra, yielding a total of 51 179 MS/MS spectra.

The spectra were searched against the *E.coli* protein sequence database using several algorithms to mitigate the bias resulting from any one algorithm or algorithm class: SEQUEST (Eng *et al.*, 1994; Yates *et al.*, 1995), OMMSA (Geer *et al.*, 2004), ProbID (Zhang *et al.*, 2002), PepNovo (Frank and Pevzner, 2005), Lutefisk (Taylor and Johnson, 1997), Inspect (Tanner *et al.*, 2005) and GutenTag (Tabb *et al.*, 2003). These algorithms were chosen to represent the diversity of existing MS/MS analysis software and according to source code and executable availability. Parameters for each algorithm were set as appropriate to search all peptides (regardless of enzyme specificity) with a precursor mass tolerance of $+/-2.5$ Da. PSMs from each algorithm were accepted if they met the following criteria: a minimum length of six amino acids, a charge of +2, fully tryptic (ending in K or R) with no missed cleavages. The algorithms GutenTag and SEQUEST had additional filters of a minimum DeltCN of 0.20 and 0.10, respectively. Many of the algorithms return multiple PSMs for a particular spectrum; in these cases, the top PSM (according to the primary scoring method for that algorithm) that matched the above criteria was selected as the PSM for that algorithm and spectrum. The FDR for each algorithm was estimated by searching the spectra against a randomly shuffled sequence database, consisting of randomly generated proteins with the same amino acid frequencies and length distribution as the original sequence database. PSMs are sorted by the primary scoring metric, and the FDR at a given primary scoring threshold is equal to the number of identifications to the shuffled database divided by the number of identifications to the real database above that threshold. The handful of short peptides that occurred in both the real and shuffled sequence databases were ignored for the purpose of calculating FDR. Thresholds for accepting PSMs generated by each algorithm were set consistent with an FDR of 1%.

In some cases, the PSMs contain contradictory assignments of different peptides by different algorithms to the same spectrum. We remove these contradictory PSMs from the set. Finally, we require each PSM to be confirmed by at least two algorithms, and each peptide to be confirmed by at least two spectra. The resulting 1208 charge +2 PSMs were used to train Riptide.

3.3 Testing datasets

For validation, we use a publicly available tandem MS dataset from Klammer *et al.* (2007), available as the 60cm dataset at http://noble.gs.washington.edu/proj/retention/data/data.html. The dataset consists of a collection of 18 149 spectra derived from a yeast whole-cell lysate as described previously (Klammer *et al.*, 2007). This data set was used to demonstrate the model's ability to generalize across different peptide sets.

The yeast protein sequence database used to search the test dataset was first digested to tryptic peptides *in silico*, by cleaving protein sequences after K or R except when followed by P, and allowing internal missed cleavages. The resulting peptides are then indexed by their mass in Daltons (Da) rounded to the nearest integer. For each test spectrum, we created a list of candidate peptides by rounding the spectrum's mass in Da (assuming charge of +2) to the nearest integer, and extracting all peptides within $+/-3$ Da of the rounded mass. For a sequence database of sufficient size, this list of candidate peptides will be prohibitively large; hence, to winnow this list further, we apply a subsequent filtering step akin to the SEQUEST Sp score (Eng *et al.*, 1994).

3.4 SVM training

For the SVM, we use a Gaussian kernel, and hyperparameters C and σ. C is the soft-margin penalty, or the penalty for misclassified examples, and σ is the width of the Gaussian used. These hyperparameters are selected using 5-fold nested cross-validation, where the parameter with the largest area under the receiver operating characteristic (ROC) curve is selected. The

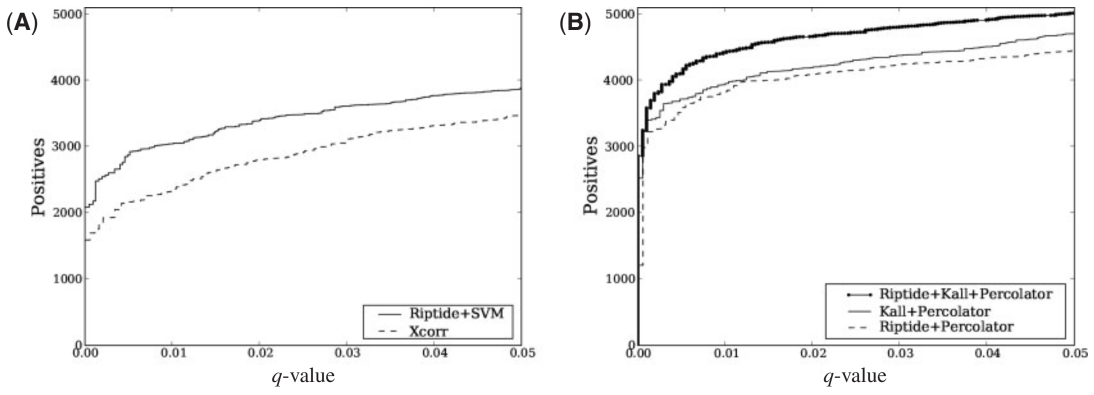


Fig. 4. Positive peptide identifications as a function of *q* -value (a measure of FDR). The Riptide scoring function is compared with the SEQUEST scoring function *Xcorr*, to test the utility of the SVM normalized discriminant score function (A). In addition, the Riptide DBN feature vectors are used as input to the algorithm Percolator (Käll *et al.*, 2007), and are compared with the original Percolator features (B).

SVM was implemented using the publicly available software package PyML http://pyml.sourceforge.net).

4 RESULTS

4.1 Validation with a sequence database search

We test Riptide in the context of a three-stage computational pipeline, in which (1) candidate PSMs are generated by a reimplementation of SEQUEST. (C.Y.Park *et al.*, In Press); (2) these PSMs areevaluated by Riptide, and (3) the resulting feature vectors are post-processed either by an SVM or by the semi-supervised learning algorithm known as Percolator (Käll *et al.*, 2007). We train Riptide and the SVM using target and decoy PSMs from *E.coli* (Section 3.2), and we then measure the ability of the pipeline to discriminate between target and decoy PSMs, using spectra generated from a yeast whole-cell lysate (Section 3.3).

An SVM is a binary classifier that projects feature vectors into a high-dimensional space and learns an optimal separating hyperplane between positive and negative examples in that space (Section 3.4). In this case, we use an SVM to learn to discriminate between positive and negative PSMs, using the 99-dimensional feature vectors generated by Riptide. After SVM training, test set PSMs are scored using the discriminant value of the SVM classifier, which is the distance between the test PSM's feature vector and the SVM hyperplane. If the scoring function is working well, then correct PSMs will be assigned positive discriminant scores, and incorrect PSMs will be assigned negative scores.

Figure 4A compares the performance of Riptide+SVM with the performance of XCorr, the score function used by SEQUEST [re-implemented in the software package Crux (C.Y.Park *et al.*, In Press)]. To generate thefigure, we searched each spectrum in the test set against a shuffled decoy version of the same protein sequence database (Klammer *et al.*, 2007). We use the number of matches to the decoy database at a particular score threshold to estimate the rate of false identifications among the target PSMs (Käll *et al.*, 2008). For each PSM, we then compute a *q* value, which is defined as the minimal FDR threshold at which the PSM is deemed significant (Storey and Tibshirani, 2003). Each series in the figure plots the number of target PSMs identified as a function of *q*-value threshold. We selected this mode of evaluation because it closely matches the goal of the typical mass spectrometrist: identifying the largest number of peptides with the lowest rate of false identifications. Riptide with the static SVM outperforms SEQUEST by 10.8% at a 1% FDR. In this experiment, the Riptide DBNs failed on many short (length seven or less) peptides, so they are not included in the analysis. If these peptides are included, performance deteriorates dramatically.

In addition to testing the static SVM post-processor, we test Riptide in conjunction with the semi-supervised learning algorithm, Percolator (Käll *et al.*, 2007). Percolator uses an SVM to iteratively learn to discriminate between correct and incorrect PSMs by using PSMs from a decoy dataset as a proxy for incorrect PSMs in the target dataset. As originally described, Percolator uses a collection of 20 features, including several features derived from the algorithm SEQUEST: e.g. *Xcorr*, *Sp*, *deltCN* as well as features describing the trypticity of the peptide termini, among others. We tested three variants of Percolator: using the original 20 features, using Riptide's 99 features and using all 119 features. The results are shown in Figure 4B. When Riptide's feature vectors are combined with those used in the original Percolator publication, we obtain 12.4% improvement in positives at 1% false discovery (Fig. 4B).

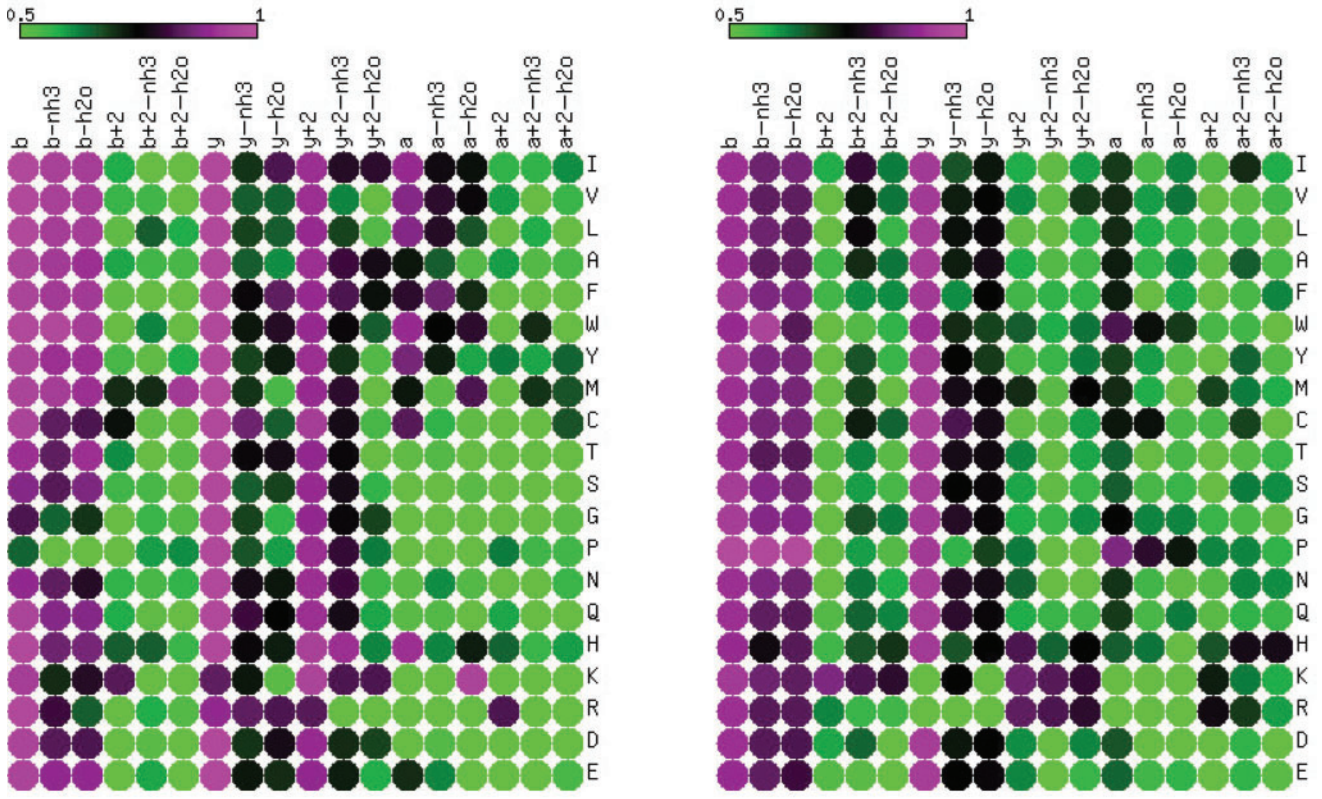
In Figure 4B, the original features from (Käll *et al.*, 2007) outperform the Riptide features when both are used alone. This result is likely due to the use of information in generating Percolator features unavailable to Riptide, in particular, protein-level information. Thus, Riptide-derived features will likely benefit discrimination most when combined with high-level, non-fragmentation based-information.

4.2 Analysis of learned fragmentation probabilities

An additional benefit of using DBNs in the Riptide model is that the probability distributions learned by the networks can be readily interpreted to produce scientific insights. We examine the probability distributions governing ion fragment intensities learned by the single-ion and paired-ion types of Riptide models in Figure 6.

In Figure 6A, we examine the distributions of intensities learned for particular residues and ion types by the single-ion models.

(A)



(B)

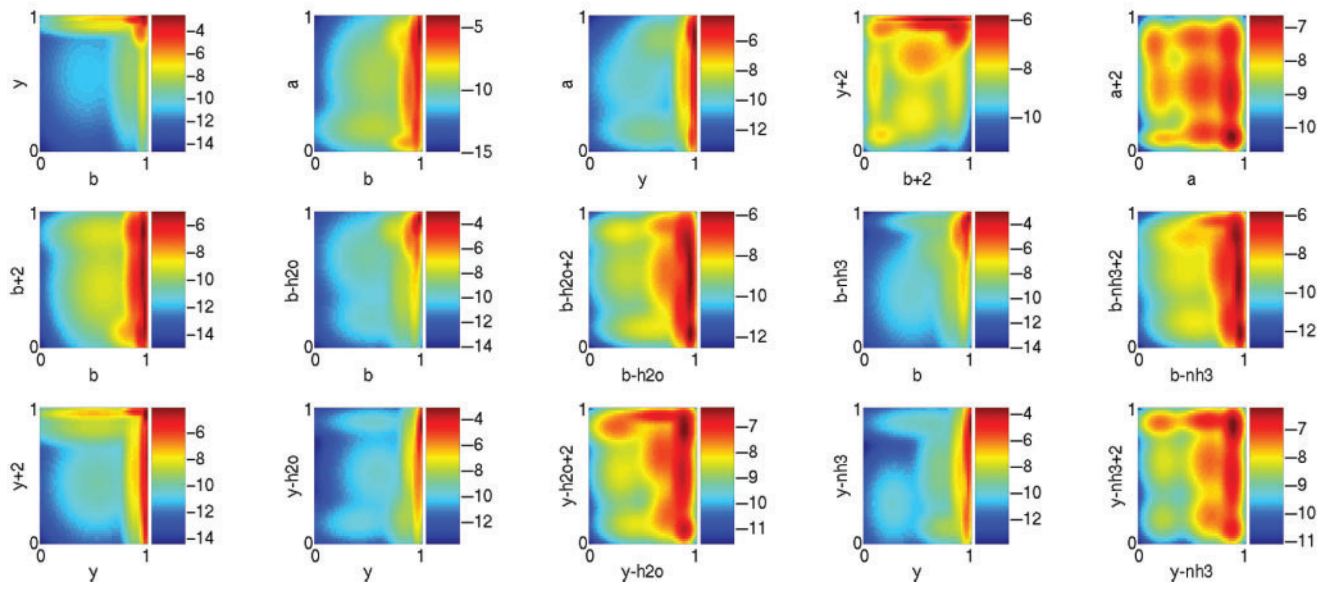


Fig. 6. Learned parameters of the Riptide model. (**A**) Displays the mean peak intensities for different residues and ion types learned using the Riptide single-ion models. Each cell shows the mean normalized intensity value for a particular ion series and flanking residue. For the left heat map, residues designated are those to the left of the amide bond fragmented to produce ions of that type (i.e. the amide bond is itself C-term to the residue), while for the right heat map, the residues designated are those to the right of the fragmented amide bond (i.e. the amide bond is itself N-term to the residue). The top image was created using matrix2png (Pavlidis and Noble, 2003). (**B**) Displays the 2D Gaussian distributions of peak intensities for pairs of ions learned using the Riptide paired-ion models. Each plot shows the joint distribution of ion intensities resulting from the same amide bond cleavage; thus, for example, points on the b/y plot corresponds to b_i/y_{n-i} pairs, for a peptide of length n. The color bar scale indicates natural log probability.

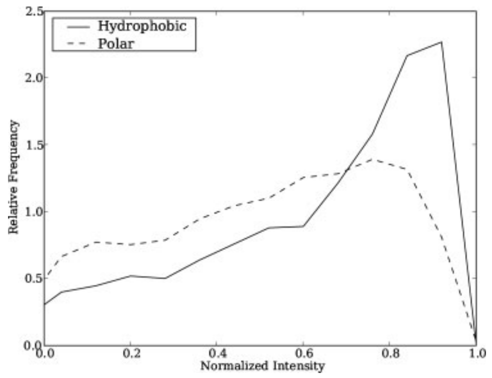


Fig. 5. Distribution of a-ion intensities from fragmentation at sites C-term to hydrophobic or polar residues. Difference is significant at $P \ll 0.0001$.

Each plot shows the mean intensity for each of the 18 single-ion models and the 20 residues that can be to the left and right (N-term and C-term) of the cleaved amide bond. Several expected trends can be detected: the high intensities of b- and y-ions; the high intensities of cleavages N-term to P, and corresponding low intensities of cleavages C-term to P. Other trends that have not been as widely noted in the peptide fragmentation literature are also present: the increased prevalence of $+2$ y-ions when basic residues are C-term to the fragmentation location; and the increased intensity of singly charged a-ions when hydrophobic residues are N-term to the fragmentation site. The first effect makes sense from physical principles. We examined the latter effect more closely by looking at the raw distribution of a-ion intensities near hydrophobic residues. Specifically, we compared a-ion intensities that resulted from fragmentation events C-term to the hydrophobic residues YILMFWC (where C is modified with iodoacetamide) with the polar residues RKDENQ (Fig. 5). The two distributions are significantly different from one another (Kolmogorov–Smirnov test, $P \ll 0.0001$).

In Figure 6B, we examine the 2D distributions learned by the joint intensity node in the paired-ion models. Here we analyze how pairs of ion intensities resulting from the same amide bond cleavage depend on each other. Again, several expected trends can be detected: prominence of b- and y-ions, with relatively higher prominence for y-ions (panel b/y); relatively low values for a and a^{++} ions, but with preference for a (panel a/a^{++}). Other suggestive trends can also be found: the apparent correlation between b-ions and their respective neutral losses (diagonal plumes on the right of panels b/b° and b/b^{*}); the qualitative difference between plots showing b-ion neutral losses and their charge $+2$ species, on the one hand, and y-ion neutral losses and their charge $+2$ species, on the other ($b^{\circ}/b^{\circ++}$ and b^{*}/b^{*++} versus. $y^{\circ}/y^{\circ++}$ and y^{*}/y^{*++}). The y-ions possess a prominent plume at the top of the plots, showing increased prevalence of $+2$ neutral losses at low $+1$ charge states, relative to b-ions.

5 DISCUSSION

We have presented Riptide, which models peptide fragmentation chemistry using a collection of DBNs trained from high-quality PSMs. Riptide can provide insights into fragmentation biochemistry, and feature vectors produced by Riptide can be used as input to further machine learning algorithms to improve peptide identification.

The Riptide models generalizes well across PSMs from different organisms: we train our model on PSMs from *E.coli* and test on PSMs from the yeast *Saccharomyces cerevisiae*. This good generalization is aided by the DBN machinery's ability to control model complexity through switching parents, dramatically reducing the number of trainable parameters. It is unlikely that a model taking into account, for example, C-term and N-term flanking amino acids could be trained on a few thousand spectra without some analogous parameter-reduction machinery.

Of course, Riptide will likely not generalize well across all types of MS/MS peptide fragmentation data. For example, using different methods of activating peptide ions, such as electron transfer dissociation (ETD) (Mikesh *et al*., 2007) or electron collision dissociation (ECD) (Zubarev, 2004), would likely require retraining the model. Furthermore, very long or very short peptides (as noted in Section 4.1) may also exhibit different chemistries that subvert the Riptide model. However, one of the benefits of the learning approach used here is that Riptide is not static and can improve as data improves and as technology and protocols change. For example, in this study we focused on fragmentation of tryptic peptides of charge state $+2$, because these are the most common peptides in the samples we analyze with collision induced dissociation. But different samples generated from different proteases or analyzed with different fragmentation technologies could be used to train the Riptide models. A related advantage of the machine learning approach is that new DBNs can be applied to arbitrary ion series. In this work, we focused on collision-induced dissociation fragmentation spectra from $+2$ peptides. An obvious extension would be to apply the DBNs to different charge states, such as $+1$ and $+3$ or higher. Also, ETD and ECD have been shown to be useful in proteomics, but produce prevalent c- and z-ions, rather than b and y. Given appropriate training data, Riptide could learn fragmentation patterns from these ion series.

In a sense, the two overall goals of Riptide—learning about peptide fragmentation biochemistry and improving our ability to identify spectra—are at odds with respect to each other. This tension correlates with the observation that, in general, DBNs admit two different methods of parameter training. On the one hand, there is *generative training*, where optimizing the objective function means that the corresponding joint probability distribution should best describe the data. As a simple example, given a DBN representation of the joint distribution of intensity and peptides $Pr(i,p|\theta)$, where θ are model parameters, generative model training adjusts θ so that this joint distribution is as accurate as possible. *Discriminative training*, on the other hand, adjusts the parameters of the model so that classification accuracy is as high as possible. For example, using Bayes rule, we can form the posterior $Pr(p|i,\theta)=Pr(i,p|\theta)/Pr(i|\theta)$ and then choose the p that maximizes this posterior. Adjusting the parameters θ to minimize the error rate of a so-formed Bayes decision rule would constitute discriminative training. Generative training is computationally cheap relative to discriminative training. Therefore, in this work we have simulated a discriminative training procedure by explicitly training positive and negative models separately. This latter choice was also motivated by the desire to obtain interpretable probabilistic parameters, which a model trained solely on positive PSMs allows. In future work, we plan to experiment by using a fully discriminative Riptide model for peptide identification and using a separate, fully generative model for investigating fragmentation phenomena.

Although Riptide is relatively fast in real time (on the order of a minute per spectrum for the databases considered here), it is slow compared to other commonly used PSM evaluation metrics, such as $Xcorr$. This is tolerable, because there is a long history in MS/MS analysis software of using fast preliminary scores to pre-filter peptides before handing them off to the sensitive, yet expensive, final scoring routines. The running time for Riptide to score a given spectrum scales approximately as $O(lN_pN_ilog(N_s))$, where l is the average length of a peptide, N_p is the number of candidate peptides for that spectrum, N_i is the number of ion series under consideration, and N_s is the number of peaks in the particular spectrum.

Currently Riptide is implemented in a combination of C++ and Python code, using the GMTK package for dynamic Bayesian network analysis. GMTK is freely available, and the C++ code is available from the authors upon request. In the near future, we plan to migrate Riptide to C and integrate the code into the sequence database search package Crux. (C.Y. Park *et al*., In Press). Ultimately, the Crux package will incorporate the probabilities produced by Riptide for PSMs into probabilities for protein identification.

ACKNOWLEDGEMENTS

Funding: This work was supported by National Institutes of Health awards R01 EB007057 and P41 RR11823.

Conflict of Interest: none declared.

REFERENCES

Bafna,V. and Edwards,N. (2001) SCOPE: a probabilistic model for scoring tandem mass spectra against a peptide database. *Bioinformatics*, **17**, S13–S21.

Bilmes,J. and Bartels,C. (2005) Graphical model architectures for speech recognition. *IEEE Signal Proc. Mag.*, **22**, 89–100.

Dongre,A.R. *et al*. (1996) Influence of peptide composition, gas-phase basicity, and chemical modification on fragmentation efficiency: evidence for the mobile proton model. *J. Am. Chem. Soc.*, **118**, 8365–8374.

Dancik,V. *et al*. (1999) *De novo* peptide sequencing via tandem mass spectrometry. *J. Comput. Biol.*, **6**, 327–342.

Eng,J.K. *et al*. (1994) An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass Spectr.*, **5**, 976–989.

Elias,J.E. *et al*. (2004) Intensity-based protein identification by machine learning from a library of tandem mass spectra. *Nature Biotechnology*, **22**, 214–219.

Field,H.I. *et al*. (2002) Radars, a bioinformatics solution that automates proteome mass spectral analysis, optimises protein identification, and archives data in a relational database. *Proteomics*, **2**, 36–47.

Frank,A. and Pevzner,P. (2005) Pepnovo: de novo peptide sequencing via probabilistic network modeling. *Anal. Chem.*, **77**, 964–973.

Geer,L.Y. *et al*. (2004) Open mass spectrometry search algorithm. *J. Proteome Res.*, **3**, 958–964.

Heckerman,D. (1995) A tutorial on learning with Bayesian Networks. *Technical report*, Microsoft Corporation, Redmond.

Havilio,M. *et al*. (2003) Intensity-based statistical scorer for tandem mass spectrometry. *Anal. Chem.*, **75**, 435–444.

Hoopmann,M.R. *et al*. (2007) High speed data reduction, feature detection, and MS/MS spectrum quality assessment of shotgun proteomics datasets using high resolution mass spectrometry. *Anal. Chem.*, **79**, 5620–5632.

Kinter,M. and Sherman,N.E. (2000) Protein sequencing and identification using tandem mass spectrometry. Wiley-Interscience, New York, NY.

Klammer,A.A. *et al*. (2007) Improving tandem mass spectrum identification using peptide retention time prediction across diverse chromatography conditions. *Anal. Chem.*, **79**, 6111–6118.

Käll,L. *et al*. (2007) A semi-supervised machine learning technique for peptide identification from shotgun proteomics datasets. *Nat. Methods*, **4**, 923–925.

Käll,L. *et al*. (2008) Assigning significance to peptides identified by tandem mass spectrometry using decoy databases. *J. Proteome Res.*, **7**, 29–34.

Lauritzen,S. (1996) *Graphical Models*. Oxford Science Publications, Oxford University Press, Princeton, NJ.

Mann,M. *et al*. (2001) Analysis of proteins and proteomes by mass spectrometry. *Ann. Rev. Biochem.*, **70**, 437–473.

Mikesh,L.M. *et al*. (2007) The utility of ETD mass spectrometry in proteomic analysis. *Biochim. Biophys. Acta.*, **1764**, 1811–1822.

Pavlidis,P. and Noble,W.S. (2003) Matrix2png: a utility for visualizing matrix data. *Bioinformatics*, **19**, 295–296.

Paizs,B. and Suhai,S. (2004) Fragmentation pathways of protonated peptides. *Mass Spectro. Rev.*, **24**, 508–548.

Park,C.Y. *et al*. (2008) Rapid and accurate peptide identification from tandem mass spectra. *J. Proteome Res.*, In Press.

Storey,J.D. and Tibshirani,R. (2003) Statistical significance for genome-wide studies. *Pro. Natl. Acad. Sci.USA*, **100**, 9440–9445.

Taylor,J.A. and Johnson,R.S. (1997) Sequence database searches via *de novo* peptide sequencing by tandem mass spectrometry. *Rapid commun. Mass Spectr.*, **11**, 1067–1075.

Tabb,D.L. *et al*. (2003) Gutentag: high-throughput sequence tagging via an empirically derived fragmentation model. *Anal. Chem.*, **75**, 6415–6421.

Tabb,D.L. *et al*. (2004) Influence of basic residue content on fragment ion peak intensities in low-energy collision-induced dissociation spectra of peptides. *Anal. Chem.*, **76**, 1243–48.

Tanner,S. *et al*. (2005) InsPecT: Identification of posttranslationally modified peptides from tandem mass spectra. *Anal. Chem.*, **77**, 4626–4639.

Wysocki,V.H. *et al*. (2000) Mobile and localized protons: a framework for understanding peptide dissociation. *J. Am. Soc. Mass Spectr.*, **35**, 1399–1406.

Washburn,M.P. *et al*. (2001) Large-scale analysis of the yeast proteome by multidimensional protein identification technology. *Nat. Biotechnol.*, **19**, 242–247.

Wan,Y. and Chen,T. (2005) PepHMM: a hidden Markov model based scoring function for mass spectrometry database search. *Anal.l Chem.*, **78**, 432–437.

Yates,III,J.R. *et al*. (1995) Method to correlate tandem mass spectra of modified peptides to amino acid sequences in the protein database. *Anal. Chem.*, **67**, 1426–1436.

Yates,III,J.R. (1998) Mass spectrometry and the age of the proteome. *Anal. Chem.*, **33**, 1–19.

Zhang,N. *et al*. (2002) ProbID: a probabilistic algorithm to identify peptides through sequence database searching using tandem mass spectral data. *Proteomics*, **2**, 1406–1412.

Zubarev,R.A. (2004) Electron-capture dissociation tandem mass spectrometry. *Curr. Opin. Biotechnol.*, **15**, 12–16.

Zhang,Z. (2004) Prediction of low-energy collision-induced dissociation spectra of peptides. *Anal. Chem.*, **76**, 3908–3922.