BagReg: Protein inference through machine learning

Can Zhao\textsuperscript{a,1}, Dao Liu\textsuperscript{b,1}, Ben Teng\textsuperscript{a}, Zengyou He\textsuperscript{a,*}

\textsuperscript{a} School of Software, Dalian University of Technology, Dalian, China
\textsuperscript{b} Baidu.com, Inc., No. 10, Shangdi 10th Street, Haidian District, Beijing, China

**Abstract**
Protein inference from the identified peptides is of primary importance in the shotgun proteomics. The target of protein inference is to identify whether each candidate protein is truly present in the sample. To date, many computational methods have been proposed to solve this problem. However, there is still no method that can fully utilize the information hidden in the input data.

In this article, we propose a learning-based method named BagReg for protein inference. The method firstly artificially extracts five features from the input data, and then chooses each feature as the class feature to separately build models to predict the presence probabilities of proteins. Finally, the weak results from five prediction models are aggregated to obtain the final result. We test our method on six public available data sets. The experimental results show that our method is superior to the state-of-the-art protein inference algorithms.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction
Proteomics is an emerging discipline which is focused on the large-scale studies of proteins expressed in an organism (Altelaar et al., 2012). An explicit goal of proteomics is the prompt and accurate identification of all proteins in a cell or tissue. Protein identification is the foundation of proteomics research, which is of primary importance to the success of down-stream analysis tasks such as protein quantification and biomarker discovery.

Mass spectrometry-based shotgun proteomics is a strategy which offers fast, high-throughput characterization of complex protein mixtures. In shotgun proteomics, the protein identification procedure has two main steps: peptide identification and protein inference. More precisely, the experimental tandem mass spectra are firstly searched against a protein sequence database to obtain a set of peptide spectrum matches, and then the identified peptides are assembled into a set of confident proteins. Fig. 1 gives an illustration of protein identification process. In this article, we focus on the protein inference problem.

To date, many computational approaches for protein inference have been proposed. The details of these existing methods and the challenges of protein inference problem have been summarized and discussed in several recent reviews (Claassen, 2012; Huang et al., 2012; Li and Radivojac, 2012). In general, the input for protein inference can be modeled as a bipartite graph in which one side is a set of peptides and the other is a set of candidate proteins. There are many important features that are embedded in the input graph for each protein, such as spectra count, scoring information, the number of shared peptides, etc. Many protein inference algorithms have taken advantage of some of these features to identify whether one protein is present in the sample or not.

Unfortunately, existing protein inference methods cannot fully utilize available latent information in the input graph. For example, ProteinProphet (Nesvizhskii et al., 2003) employs an EM-like iterative procedure to estimate protein probabilities, but the information of spectra count is ignored. Similarly, methods like MSBayesPro (Li et al., 2009) and ProteinLP (Huang and He, 2012) just consider the best matched spectrum for each peptide and discard spectra with lower scores that may be the real spectra generated from the target peptide.

The aforementioned observations motivate our research. We propose a new protein inference model based on machine learning to fully utilize the information hidden in the input graph. Our method is called BagReg, which can be divided into three major phases: feature extraction, prediction model construction and prediction result combination.

In feature extraction, we generate five features from the input graph for each protein: the number of matched peptides, the number of unique peptides, the number of matched spectra, the maximal score of matched peptides and the average score of matched peptides. These features have a common characteristic:
the protein identification procedure has two main steps: peptide identification and protein inference.

In prediction model construction, we build five classification models to predict the presence probabilities of proteins. In each model, we take one feature as the class/target feature and use the remaining four features as dependent features. Thanks to the positive correlation between protein presence probability and the value of extracted features, we can sort proteins according to the class feature values in a decreasing order. The training data set is constructed by selecting a subset of proteins with higher (lower) class feature values as the positive (negative) data set, respectively. Classification methods such as logistic regression (Lee and Silvapulle, 1988) are first exploited to construct a predictive model on the training data and then the classifier is applied to predict the presence probabilities of proteins.

In prediction result combination, we aggregate the prediction probabilities from five models to generate a consensus presence probability for each protein. Here we simply use the average of probabilities from five models to generate a consensus presence probability for each protein. Here we simply use the average of probabilities from five models to generate a consensus presence probability for each protein. The high confidence in peptide identification will accordingly increase our confidence on the presence of its parent proteins.

These five features can be directly obtained from the input data, and their values are numeric and easy to be calculated. Besides, there is a positive correlation between these feature values and the presence probabilities of proteins in the biological sample. That is, proteins with higher feature values are more likely to be present in the sample than those with lower feature values. This characteristic brings much convenience to the construction of training set. In addition, by taking all these five features into account during the learning process, the algorithm can eliminate the negative impact caused by ‘one-hit wonder’ (the protein that has only one matched peptide) and degenerate peptides (the peptides that are shared by multiple proteins) to some extent. This is because ‘one-hit wonder’ is determined by the number of matched peptides and degenerate peptides are closely related with the number of unique peptides.

Here we have to clarify that we only use this five but no more features, since our objective is to show the feasibility of proposed new framework for protein inference. We believe there would be other features that might be included, while these five features are good enough to demonstrate the feasibility of our algorithm.

2.2. Prediction model construction

Inspired by the bagging algorithm (Breiman, 1996) which improves the stability and accuracy by aggregating the results generated by several classifiers from the same input data, we perform several parallel learning processes in prediction model construction. Since we have already obtained five features that are all positively correlated with the presence probabilities of proteins,
Fig. 3. The feature extraction process. We extract five features from the original input data for each protein: the number of matched peptides (MP), the number of unique peptides (UP), the number of matched spectra (MS), the maximal score of matched peptides (MSP) and the average score of matched peptides (AMP).

we can construct the training set based on each of the features. After that, we choose a classification method to construct a prediction model on the training data and apply the classification model to predict presence probabilities for all proteins. Fig. 4 gives an illustration of a component learning process.

2.2.1. Training set construction

Before constructing the prediction model, we need to collect training data set for building the model first. It is crucial to generate accurate and diverse classifiers in different learning processes, under which condition we can confidently regard the results predicted by the classifiers as valuable references.

Since the proteins come with no label, we have to set up the labels artificially. In each single learning process, we select one feature as the class feature and other four are regarded as dependent features. It helps to generate different classifiers and explains the reason why we make the number of learning processes to be identical with the feature number (i.e. five features correspond to five class features). As previously stated in feature extraction section, there is a positive correlation between the feature value and the protein presence probability. Therefore, we first sort all candidate proteins based on the class feature value. Then, we select a portion of top ranked proteins as the positive training set and a portion of proteins at the end of the sorted list as the negative training set.

Fig. 4. Single component learning process. Each separate learning process accomplishes a typical supervised learning procedure. We firstly build a classifier and then apply it to generate prediction results. The model construction phase involves constructing the training set and building the classifier (steps 1–3). And the prediction phase involves the construction of testing set and applying the model (steps 4–6).
Because the distribution of the data set is unknown, it is difficult to determine the size of training data automatically. In this paper, we use a parameter $c$ to control the size of both the positive data set and negative data set. For instance, if $c = 20\%$, then we select top 20\% of proteins at the beginning of sorted list to construct the positive training data. Similarly, those 20\% proteins with lowest class feature values are used as the negative training data.

2.2.2. Prediction model selection

After constructing the training set, a classification model is ready to be built. Here we can consider any classification model that could produce a probability as the prediction result. To generate a presence probability for each protein, the testing set is the whole data set. In this paper, we choose two widely used classifiers: logistic regression and Bayesian network (Cooper and Herskovits, 1992) to construct a classification model on the training set and then predict the presence probabilities for all proteins.

Logistic regression is a probabilistic classification model, which is a typical approach in supervised learning (Bishop et al., 2006). The prediction probability that one sample belongs to a certain class is modeled as a function of the explanatory features, using a logistic function:

$$F(h) = \frac{e^h}{1 + e^h},$$

where base $e$ denotes the exponential function, and $h$ is a linear combination of explanatory features, as described in the following formula:

$$h = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k,$$

where $x_5$ are the observed variables, namely the remaining features except the designate class feature in the paper, and $\beta_1$ is the regression coefficient. Since we choose one feature as the class/target feature and use the remaining four features as dependent features, the number of features involved in the linear combination is 4, i.e. $k = 4$.

Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (Bishop et al., 2006). The network can be used to find out updated knowledge of the state of a subset of variables when other variables (the evidence variables) are observed through computing the posterior distribution. According to the Bayes’ rule, the posterior probability can be expressed as:

$$p(c_i = 1|\mathbf{x}_1, \ldots, \mathbf{x}_k; \theta) = \frac{p(\mathbf{x}_1, \ldots, \mathbf{x}_k, c_i; \theta)}{\sum_{c_i} p(\mathbf{x}_1, \ldots, \mathbf{x}_k, c_i; \theta)},$$

where $c_i$ is the indicator variable, $c_i = 1$ denotes that protein $i$ is present in the sample, otherwise protein $i$ is absent. And $\theta$ is the parameter of the classifier. In order to get the optimal protein configuration, the maximum a posteriori (MAP) rule is adopted:

$$\max p(c_i = 1|\mathbf{x}_1, \ldots, \mathbf{x}_k; \theta).$$

After model training and prediction, each protein should get a predictive score indicating its existence possibility. Since we perform five learning processes, each protein should be assigned with five weak scores in the end.

2.3. Prediction result combination

Since each candidate protein is assigned with five scores, we need to integrate these results in an unbiased manner. Many methods can be employed in this phase. In this article, we use a simple but widely used approach to get the final score of each protein via calculating the arithmetic mean of its five weak scores. This means that each of the five scores is assigned with the same weight when considering its contribution to the existence of each protein.

3. Experiments

We apply our algorithm to six data sets and compare its performance with three existing protein inference methods. The results show that our algorithm beats other methods on most of the data sets in terms of the identification accuracy.

3.1. Data sets

In the experiment, we use six data sets: 18 mixtures (Moore et al., 2002), Sigma49 (Tabb et al., 2007), Yeast (Ramakrishnan et al., 2009a), DME (Brunner et al., 2007), HumanMD (Ramakrishnan et al., 2009b) and HumanEKC (Ramakrishnan et al., 2009a). Each of the first three data sets has a protein reference set as its ground-truth protein set and the other three data sets do not have reference sets. More details on these data sets can be found in Huang and He (2012).

We use the target-decoy strategy in the performance evaluation. In the target-decoy strategy, the tandem mass spectra are first searched against a mixed protein database containing all target protein sequences and an equal number of decoy sequences. And then an identified protein is considered as a true identification if it comes from the target protein database.

In peptide identification, we use X!Tandem (v2010.10.01.1) (Craig and Beavis, 2004) to search the protein database with its default parameters. If one data set has a reference set, the tandem mass spectra are only searched against the target protein database. Otherwise, the spectra are searched against both the target and decoy protein databases. Peptide identification probabilities are obtained with PeptideProphet (Trans-Proteomic Pipeline v4.5) (Keller et al., 2002).

3.2. Parameter setting

In the training set construction of BagReg, we use a parameter $c$ to control the size of both the positive data and negative data. Here, we set $c = 20\%$. In the prediction model selection phase, we choose logistic regression and Bayesian network as the classifier. In the experiments, we use the implementation in Weka (Hall et al., 2009) for these two classifiers with their default parameters.

3.3. Results

In the experiment, since all the inference algorithms generate protein probabilities as output, we use the receiver operating characteristic (ROC) curve to compare and evaluate the protein inference results. The ROC curve uses the false positive rate as its x-axis and the true positive rate as its y-axis. The area under the curve (AUC) is used as a single numeric indicator of overall performance.

We compare our BagReg algorithm with three protein inference algorithms: ProteinProphet, MSBayesPro and ProteinLP with their default parameters. The comparison results are presented in Figs. 5 and 6, where the classifier used in BagReg is logistic regression and Bayesian network, respectively.

From the comparison results in these two figures, we have the following observations.

Firstly, our method performs the best 3 times and 5 times in terms of AUC values in Figs. 5 and 6, respectively. This indicates that our method can achieve better identification performances than other protein inference algorithms.

Secondly, our method performs the worst on the yeast data set while ProteinProphet, MSBayesPro and ProteinLP have similar performance. The reason is that there are only few candidate proteins that are false positives according to reference set of yeast data. Thus, we will include too many true positives in the negative training set, leading to the construction of an inaccurate prediction model.
Fig. 5. Performance comparison among four different protein inference algorithms. Here the classifier used in BagReg is logistic regression.
Fig. 6. Performance comparison among four different protein inference algorithms. Here the classifier used in BagReg is Bayesian network.
Fig. 7. The prediction results before ensemble, where each of the five features is used as the target feature. Here the classifier used is Bayesian network.
A Bayesian network is used as the classifier. Fig. 7 presents the performance, we analyze the prediction results before the ensemble when the classifier used.

The correlation with the number of matched spectra. As a result, the classifier such as Bayesian network that incorporates the dependence relationship into consideration is more likely to yield better performance than other classifiers. While the performance gap is not very significant, indicating that our method is robust with respect to the classifier used.

To test which feature is more effective for accurate prediction, we analyze the prediction results before the ensemble when Bayesian network is used as the classifier. Fig. 7 presents the performance curves when each of the five features is served as the target feature. In general, there are no features that can always achieve the best performance over all data sets. Furthermore, we can observe that these features are complementary to each other with respect to prediction accuracy. Therefore, the arithmetic mean, a simple but widely used approach, is a good choice for aggregating the prediction results.

To test the sensitivity of our algorithm to the training data size, we vary the training data size parameter $c$ from 0.05 to 0.35 and plot the AUC values in Fig. 8. It shows that the identification result is stable with respect to $c$ when this parameter falls into the range $[0.15, 0.35]$. Though we cannot determine the optimal training data size automatically, we suggest to set this parameter $c = 0.2$ or $c = 0.3$ as a rule of thumb in practice since such settings exhibit good performances on average.

4. Conclusions

In this paper, we tackle the protein inference problem from a machine learning perspective and propose a novel method called BagReg. Different from previous algorithms in this field, we transform the unsupervised protein inference problem into a supervised classification problem through automatic feature extraction and training data construction. The main advantage is that our method can improve the prediction precision of protein inference and fully utilize the latent information in the input data. According to the experimental results, our method has better performances than those competitive methods.

In fact, there are already some learning-based approaches to solve protein inference problem such as Barista (Spivak et al., 2012), ProteinLasso (Huang et al., 2013) and Re-Faction (Yang et al., 2012). Barista directly optimizes the total number of identified proteins by training a supervised learning algorithm to introduce a ranking on the combined set of target and decoy proteins. ProteinLasso formulates the protein inference problem as a constrained Lasso regression problem and then solves it with a fast pathwise coordinate descent algorithm. Re-Faction constructs a support vector machine (SVM) regression model on proteomics data generated from samples fractionated by gel electrophoresis to assign proteins to expected protein fractions. This information is then used to appropriately assign peptides to specific proteins. Clearly, our BagReg method differs from existing learning-based approaches in both the key idea and underlying methodology.

Actually, many research efforts in other bioinformatics applications have demonstrated that the use of different features and different training sets may lead to significant different prediction performances (e.g. Gong et al., 2014; Liu et al., 2014a,b; Wei et al., 2014). Therefore, we will investigate the feasibility and advantage of incorporating more features into the prediction model of BagReg in the future work. Moreover, new effective training data selection strategies will be further studied as well.

Acknowledgements

This work was partially supported by the Natural Science Foundation of China under Grant No. 61003176 and the Fundamental Research Funds for the Central Universities of China (DUT14QY07).

References


