Bioinformatics, 31, 2015, i365–i374
doi: 10.1093/bioinformatics/btv264
ISMB/ECCB 2015

OXFORD

Protein (multi-)location prediction: utilizing interdependencies via a generative model

Ramanuja Simha[1], Sebastian Briesemeister[2], Oliver Kohlbacher[2] and Hagit Shatkay[1,3,4,*]

[1]Department of Computer and Information Sciences, University of Delaware, Newark, DE, USA, [2]Applied Bioinformatics, Center for Bioinformatics, University of Tuebingen, Germany, [3]Center for Bioinformatics and Computational Biology, University of Delaware, Newark, DE, USA and [4]School of Computing, Queen's University, Kingston, ON, Canada

*To whom correspondence should be addressed.

Abstract

Motivation: Proteins are responsible for a multitude of vital tasks in all living organisms. Given that a protein's function and role are strongly related to its subcellular location, protein location prediction is an important research area. While proteins move from one location to another and can localize to multiple locations, most existing location prediction systems assign only a single location per protein. A few recent systems attempt to predict multiple locations for proteins, however, their performance leaves much room for improvement. Moreover, such systems do not capture dependencies among locations and usually consider locations as independent. We hypothesize that a multi-location predictor that captures location inter-dependencies can improve location predictions for proteins.

Results: We introduce a *probabilistic generative model* for protein localization, and develop a system based on it—which we call *MDLoc*—that utilizes inter-dependencies among locations to predict multiple locations for proteins. The model captures location inter-dependencies using Bayesian networks and represents dependency between features and locations using a mixture model. We use iterative processes for learning model parameters and for estimating protein locations. We evaluate our classifier MDLoc, on a dataset of single- and multi-localized proteins derived from the DBMLoc dataset, which is the most comprehensive protein multi-localization dataset currently available. Our results, obtained by using MDLoc, significantly improve upon results obtained by an initial simpler classifier, as well as on results reported by other top systems.

Availability and implementation: MDLoc is available at: http://www.eecis.udel.edu/~compbio/mdloc.

Contact: shatkay@udel.edu.

1 Introduction

Proteins are responsible for a multitude of diverse vital tasks in all living organisms (Rost *et al.*, 2003). Given that a protein's function and role are strongly related to its subcellular location, protein location prediction is an important research area (Alberts *et al.*, 2002; Nair and Rost, 2008). Furthermore, the location of a protein helps understand the protein's prospective utility as a drug target (Bakheet and Doig, 2009). Methods for determining protein locations include experimental as well as high-throughput computational ones. The experimental methods accurately determine protein locations, but are typically time consuming and are typically not cost effective for finding locations for a large number of proteins. Such methods include mass spectrometry (Dreger, 2003) and green fluorescence detection (Simpson *et al.*, 2000). On the other hand, the computational methods are fast, and can potentially predict locations for proteins whose actual locations have not yet been experimentally determined. Most of the prediction systems represent proteins using sequence-derived features and utilize machine learning methods (e.g. Blum *et al.*, 2009; Emanuelsson *et al.*, 2000; Nakai and Kanehisa, 1991; Shatkay *et al.*, 2007).

Proteins move from one location to another and localize to multiple subcellular compartments (Murphy, 2010; Pohlschroder *et al.*, 2005). For instance, the enzyme *TREX1*, which assists in DNA repair, is

primarily present in the cytoplasm but is also transported to the nucleus in response to DNA damage (Tomicic *et al.*, 2013). Thus, predicting multiple locations for proteins is important, as protein movement across locations enables the protein to serve multiple distinct functions. Nevertheless, all prediction systems mentioned earlier and most current systems assign only a single location per protein. Since proteins localize systematically, and translocation occurs only among specific locations for the purpose of a particular subcellular function, our hypothesis is that modeling inter-depedencies among locations can assist in predicting locations of proteins more accurately.

Posing the problem using computational, machine-learning terms, assigning multiple locations to proteins is a *multi-label classification* task. Traditional single-label classification assigns a single *label* (location) to each *instance* (protein), and is addressed by methods such as Support Vector Machines (Scholkopf and Smola, 2002), naïve Bayes or neural networks (Russell and Norvig, 2010). Multi-label classification, on the other hand, aims to associate each instance with possibly multiple classes. Some of the simplest and commonly used approaches transform the multi-label classification task into one or more single-label classification task(s) (Tsoumakas *et al.*, 2010); such approaches do not capture label inter-dependencies. More sophisticated multi-label classification approaches attempt to capture label inter-dependencies and incorporate them into the classification process. Such multi-label classification methods have not yet been employed in the context of protein location prediction.

In this article, we present a new, dependency-based probabilistic generative model for eukaryotic protein localization, and develop a multi-location prediction system—which we call *MDLoc*, to predict locations of multiply localized proteins. As was done before (Briesemeister *et al.*, 2010a; King and Guda, 2007; Li *et al.*, 2012), we use sequence-derived features and Gene Ontology (GO) terms to represent proteins. Here we introduce a new model using Bayesian networks to directly address and capture inter-dependencies among locations. Furthermore, we present the concept of *location dependency sets* and use a mixture model to represent *feature dependency on location-combinations*. The new system uses a generative model and an iterative procedure for estimating its parameters, and effectively improves the estimation process of multi-locations. Our method is based on iteratively learning parameters of the location-Bayesian-network and the mixture model, while re-inferring the location estimates of the proteins in each iteration. This improves on our preliminary system, which comprised a collection of Bayesian network classifiers, where location inter-dependencies were not learnt as part of the model but rather captured based on simple estimates of location values (Simha and Shatkay, 2014).

We evaluate MDLoc on a dataset derived from the DBMLoc dataset (Zhang *et al.*, 2008), which is the most comprehensive protein multi-localization dataset currently available, using multiple runs of 5-fold cross-validation. We show that the performance of MDLoc on multi-localized proteins improves over earlier results for a top performing system, YLoc^{+} (Briesemeister *et al.*, 2010a). The improved results obtained by MDLoc demonstrate the advantage of utilizing location inter-dependencies and feature dependencies on locations in the prediction process.

The rest of the article proceeds as follows: Section 2 surveys methods for protein multi-location prediction. In Section 3, we introduce the concept of *location dependency sets* and provide relevant notations; we also present our new probabilistic generative model for protein localization, which captures dependencies between protein-features and locations. In Section 4, we discuss the model parameters, the learning procedure used for finding them, and the inference technique used for predicting multiple protein locations. Experiments and results are presented in Section 5, followed by conclusions and future directions.

2 Related work

A number of recent location prediction systems attempt to predict multiple locations for proteins, however their performance leaves much room for improvement. While most use sequence-derived features (e.g. amino acid composition) and GO terms to represent proteins and to predict protein locations, a few are based solely on sequence-based similarity. The former class of methods incorporate one or more of the following classifiers: *k*-nearest neighbors (*k*-NN, Chou *et al.*, 2011), Support Vector Machines (Li *et al.*, 2012), naïve Bayes (Briesemeister *et al.*, 2010a) and neural networks (Emanuelsson *et al.*, 2000). KnowPred$_{site}$ (Lin *et al.*, 2009) is an example of the latter, similarity based, class of methods.

Systems that use *k*-NN adaptations to predict multiple locations for proteins include WoLF PSORT (Horton *et al.*, 2007), Euk-mPLoc (Chou and Shen, 2007), iLoc-Euk (Chou *et al.*, 2011) and an ensemble system (Li *et al.*, 2012). WoLF PSORT outputs for a query protein the location-combination that is most frequent among the protein's *k*-NN in the training set; the predictions are thus restricted to location-combinations already present in the set. Both iLoc-Euk and Euk-mPLoc compute a score for each candidate location, based on the query protein; iLoc-Euk outputs locations having the highest scores; the number of locations is the same as that associated with the query protein's nearest neighbor in the dataset; Euk-mPLoc assigns the protein to locations whose score lies within a certain deviation from the highest score.

All the methods described thus far treat locations as *independent* from one another and do not utilize possible inter-dependencies among locations in the prediction process. A few systems, however, have tried to make use of location inter-dependencies to predict multiple locations for proteins. For example, the classifier by He *et al.* (2012) attempts to use pairwise location- *correlation* in the prediction process, but does not use more complex inter-dependencies. YLoc^{+} (Briesemeister *et al.*, 2010a) introduces a new class for each location-combination represented in the training dataset and uses a naïve Bayes classifier to predict a probability distribution over these new classes. Thus, each classifier prediction is restricted to location-combinations in the training set. YLoc^{+}'s performance was evaluated using the most comprehensive protein multi-localization dataset and is the highest among current multi-location prediction systems. In our earlier preliminary work (Simha and Shatkay, 2014), we used a collection of Bayesian network classifiers to predict multiple locations of proteins. The simplified model used did not incorporate location-interdependencies into the iterative learning process, but rather utilized one-time estimates of location values to establish interdependencies. The performance of that classifier was comparable to that of YLoc^{+} when using the same dataset, but did not improve on it.

In the next section, we present a new probabilistic generative model for protein localization that directly incorporates the learning of location inter-dependencies into the iterative learning process. Additionally, we introduce the concept of *location dependency sets*, which enables us to capture feature dependencies on location-combinations in a mixture model setting. The resulting system *MDLoc*, shows significant improvement, according to all evaluation metrics, compared with previously reported performance for protein multi-location prediction.

3 A probabilistic generative model for protein localization

As we and others have done before (Briesemeister *et al.*, 2010a; Garg and Raghava, 2008; Simha and Shatkay, 2014), we represent each protein P as a weighted feature vector, $\vec{f}^P = \langle f_1^P, \ldots, f_d^P \rangle$ where d is the number of features. Let $S = \{s_1, \ldots, s_q\}$ be the set of q subcellular components in the cell. Each protein P localizes to at least one—and possibly more than one—location. The locations of each protein P are represented by a location indicator vector, $\vec{l}^P = \langle l_1^P, \ldots, l_q^P \rangle$ of 0/1 values, where $l_i^P = 1$ if P localizes to s_i, and $l_i^P = 0$ otherwise. We view each location indicator l_i^P as a value taken by a random variable L_i and each feature f_j^P as a value taken by a random variable F_j. Given a protein P, represented as a vector $\vec{f}^P$, the multi-localization task amounts to assigning a (correct) 0/1 value to each of the entries l_i^P.

3.1 Modeling location inter-dependency

We use Bayesian networks to model inter-dependencies among subcellular locations. A Bayesian network consists of a directed acyclic graph $G = (L, E)$ whose set of nodes L corresponds to random variables and set of edges E indicates dependencies among the variables. In our case, nodes represent location variables denoted $L = \{L_1, \ldots, L_q\}$. Each variable L_i corresponds to a location s_i within the cell and takes on a 0/1 value. Figure 1 shows an example Bayesian network we learn over location variables. A directed edge, for instance, from *membrane* to *cytoplasm* represents the assertion that knowing that a protein localizes to the *membrane* influences the level of belief about the protein localizing to the *cytoplasm*. According to the conditional independence relationship encoded in the Bayesian network, each variable L_i is conditionally independent of its non-descendants given its parents $\mathrm{Pa}(L_i)$ (for additional details see Russell and Norvig, 2010). The joint distribution of the location variables can thus be calculated as:

$$\Pr(L_1, \ldots, L_q) = \prod_{i=1}^{q} \Pr(L_i | \mathrm{Pa}(L_i)). \quad (1)$$

3.2 Capturing location-feature dependency

The value of each feature represents a certain characteristic of a protein, such as the relative abundance of each amino acid in the protein's amino-acid composition (King and Guda, 2007). In our experiments, we use the exact same features used by Briesemeister *et al.* (2010a), as explained in Section 5.1. For the purpose of predicting locations for a protein, we view a protein as though it was generated through a stochastic process, in which each of its feature values was determined. The value of each feature variable F_j $(1 \leq j \leq d)$ is assigned based on the values taken by one or more location random variables; that is, each feature value may *depend on multiple locations and not just on one*. For instance, consider a feature capturing the abundance of *tryptophan (Trp)* residues in the amino acid composition of a protein; we denote the random variable associated with this feature by F_{Trp}. The value of this feature varies greatly between proteins known to localize to the *membrane* vs. those that are known to localize to both the *membrane* and the *cytoplasm*. Specifically, the probability of a *membrane* protein to have more than three *Trp* residues (formally denoted as the conditional probability: $\Pr(F_{\mathrm{Trp}} > 3 | L_{\mathrm{Mem}} = 1)$), is 0.36 [this high probability agrees with the well-established importance of *Trp*'s role in membrane proteins (Schiffer *et al.*, 1992)]. In contrast, the probability of proteins known to be multi-localized to both the *membrane* and the *cytoplasm* to have more than three *Trp* residues $(\Pr(F_{\mathrm{Trp}} > 3 | L_{\mathrm{Mem}} = 1, L_{\mathrm{Cyt}} = 1))$, is only 0.15 (the probability values are calculated based on the dataset described in Section 5.1). Thus, the feature value depends on more than a single location value. To accurately capture the dependency between protein features and location-combinations, we view each feature value as depending on a *set of location indicator values*.

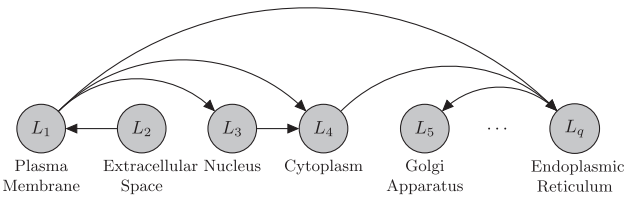

Fig. 1. An example location-Bayesian-network that we learn. Directed edges represent dependencies between the connected nodes. The location associated with each variable is shown below the corresponding node

Recall that we view a protein as represented by (i.e. comprised of) a set of features. As such, we view each possible location of a protein P as depending on a set of locations to which proteins with similar feature values (including P itself) are likely to be localized. We thus introduce the concept of *location dependency sets*. For a location s_i, its *dependency set* comprises the minimal set of locations $\{s_{i_1}, \ldots, s_{i_m}\}$ such that the likelihood of a protein to localize to s_i depends on (i.e is correlated or anti-correlated with) its likelihood of to localize to each of $\{s_{i_1}, \ldots, s_{i_m}\}$. Using the Bayesian network framework, we note that a dependency as described above between the locations s_{i_j} and s_i can be represented as a directed edge from the graph node L_{i_j} to L_i. Given a Bayesian network that represents the dependencies among locations in this way, we can thus denote the *location dependency set* for each location variable L_i as the parents of L_i in the Bayesian network. As such, we define q location dependency sets, one set per location,

$$LS_1 = \{L_1\} \cup \mathrm{Pa}(L_1), \ldots, LS_q = \{L_q\} \cup \mathrm{Pa}(L_q), \quad (2)$$

where $\mathrm{Pa}(L_i)$ $(1 \leq i \leq q)$ denotes the parents of location variable L_i in a Bayesian network.

Given a Bayesian network G, the steps involved in protein generation are discussed in the rest of this section. We use a *coin-toss model* to set location indicator values, and two *die-roll processes* to set feature values. For each feature, one die roll is used to select a location dependency set, and another to assign the actual feature value. We next describe each of the steps in detail.

3.3 Setting location values

As part of the generative process for a protein P, we view the value of a location indicator l_i^P $(1 \leq i \leq q)$ as set by tossing a coin C_i; if the coin comes up *Heads*, the location indicator l_i^P is set to 1; otherwise $l_i^P = 0$. The probability of C_i to come up *Heads* is: $\Pr(L_i = 1 | \mathrm{Pa}(L_i))$. Values comprising the location indicator vector $\vec{l}^P$ are thus set by tossing the location-specific coins in a sequence one after the other. We assume that there is a specific order in which the coins are tossed. To establish the order, we use a topological ordering of location variables in the Bayesian network G denoted as $L_{\mathrm{t}1}, \ldots, L_{\mathrm{tq}}$, where each parent in the network appears before its descendant; an example of such an ordering of nodes based on the network in Figure 1 is $L_2, L_1, L_3, L_4, L_q, L_5$. Consequently, coin $C_{\mathrm{t}1}$ is tossed first, and based on its outcome, the location indicator value $l_{\mathrm{t}1}^P$ is set, then $C_{\mathrm{t}2}$ is tossed and $l_{\mathrm{t}2}^P$ is set, and so on, until C_{tq} is tossed and l_{tq}^P is set.

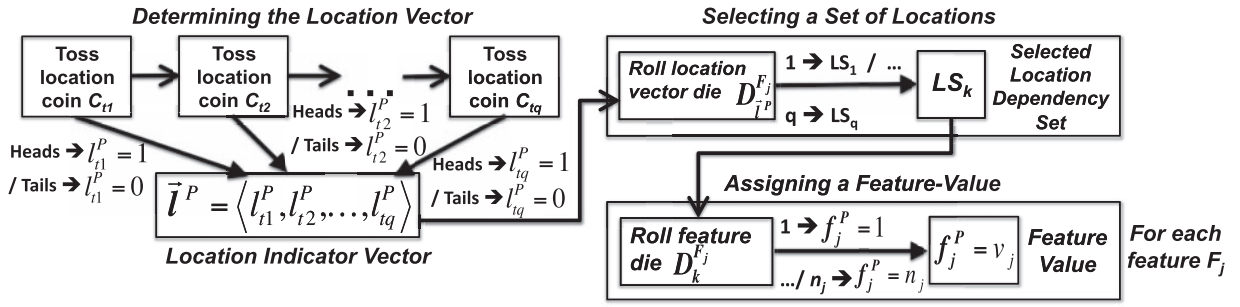

Fig. 2. The generative process for a protein *P*. First, location coins, $C_{t1}, \ldots, C_{tq}$, are tossed (top left); based on the outcomes, location indicator values, $l^P_{t1}, \ldots, l^P_{tq}$, are chosen (bottom left). Collectively, these values make up the location indicator vector $\vec{l}^P$. For each feature F_j, the die $D^{F_j}_{\vec{l}^P}$ is then tossed to select a location dependency set (top right); based on the selected set LS_k, the feature die $D^{F_j}_k$ is tossed to pick the feature-value f^P_j (bottom right)

3.4 Setting feature values

We further view each feature value f^P_j $(1 \leq j \leq d)$ as selected from among n_j possible distinct values by adhering to the following steps:

1. *A dependency set is selected:* A location dependency set is chosen based on a probability distribution over q such sets [see Equation (2) for the sets definitions]. For each feature F_j, let λ^{F_j} be a random variable that takes on the values $1, \ldots, q$, where a value i $(1 \leq i \leq q)$ indicates that the ith location dependency set is selected. We denote the event of selecting the ith set, LS_i, by $\lambda^{F_j} = i$. Given a location indicator vector $\vec{l}$ (selected in the previous step), to select a location dependency set, a die $D^{F_j}_{\vec{l}}$ with q faces is rolled. If the die $D^{F_j}_{\vec{l}}$ lands with the ith face up, the set LS_i is selected. The probability of $D^{F_j}_{\vec{l}}$ to come up as i is: $\theta^{\vec{l}}_i(F_j) = \Pr(\lambda^{F_j} = i | \vec{l})$.
2. *A feature-value is assigned:* Based on the values taken by variables in a selected location set, the feature value is chosen. Given that the set LS_k was selected, we assume that a die $D^{F_j}_k$ with n_j faces is rolled to pick a value for feature F_j. If the die $D^{F_j}_k$ lands with the v_jth face up, the feature value is set to v_j. The probability of $D^{F_j}_k$ to come up as v_j is: $\Pr(F_j = v_j | l_k, \mathrm{Pa}(L_k))$, where F_j is the random variable associated with the jth feature.

Based on this model, each feature value f^P_j is set independently of other features to construct the complete feature vector of the protein P, $\vec{f}^P = \langle f^P_1, \ldots, f^P_d \rangle$.

The generative process for a protein P is summarized as shown in Figure 2: First, the *location coins* are tossed in the order $C_{t1}, C_{t2}, \ldots, C_{tq}$—as shown on the left side of the figure. If the coin C_{ti} $(1 \leq i \leq q)$ comes up *Heads*, the location indicator l^P_{ti} is set to 1; otherwise $l^P_{ti} = 0$. Collectively, this results in choosing the location indicator vector $\vec{l}^P$. Next, for each feature F_j $(1 \leq j \leq d)$, the *location vector die* $D^{F_j}_{\vec{l}^P}$ is rolled; if the die lands with the kth face up, the set LS_k is selected—as shown on the top-right side of the figure. Based on the selected set LS_k, the *feature die* $D^{F_j}_k$ is rolled; if the die lands with the v_jth face up, the feature value f^P_j is set to v_j—as shown on the bottom-right side of the figure.

We note that our generative model makes the following two *independence assumptions*:

1. The feature values $f^P_1, \ldots, f^P_d$ of a protein P, are *conditionally independent* of each other given the protein's location indicator vector $\vec{l}^P$, formally:

$$\Pr(\vec{f}^P | \vec{l}^P) = \prod_{j=1}^{d} \Pr(f^P_j | \vec{l}^P). \quad (3)$$

While this assumption may oversimplify the underlying biological mechanisms, it works well in practice and has proven useful before (Briesemeister *et al.*, 2010a). Moreover, our model carefully accounts for inter-dependencies among locations, as well as among locations and features, thus indirectly capturing interdependencies among features.

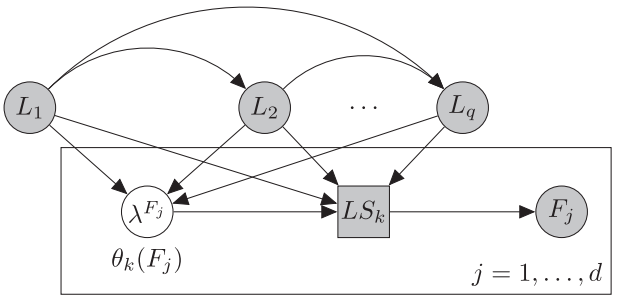

Fig. 3. The probabilistic graphical model for the generation of protein features. Directed edges represent dependencies between nodes. Locations and features are shown as circles and location sets as squares. Shaded nodes represent observed variables and unshaded nodes represent latent variables. The variable λ^{F_j} takes on a value *k*, indicating the selection of the set LS_k, with a probability $\theta_k(F_j)$. The rectangular plate notation is used to represent replication of features and location sets with the same dependencies

2. Given the values taken by a location variable L_k and its parents $\mathrm{Pa}(L_k)$ in a *selected* location dependency set LS_k, the feature value for a protein, f^P_j, is *conditionally independent* of all other location values, formally:

$$\Pr(f^P_j | \lambda = k, l^P_1, \ldots, l^P_q) = \Pr(f^P_j | l^P_k, \mathrm{Pa}(L_k)). \quad (4)$$

Figure 3 shows the protein generation process using the standard notation of a probabilistic graphical model. Nodes represent random variables and directed edges represent dependencies among variables. The values of location and feature random variables are governed by a probability distribution and as such are denoted using circles. In contrast, the value of each location dependency set variable LS_k is assigned deterministically based on the values of the location variable L_k and its parents $\mathrm{Pa}(L_k)$, and is denoted as a square. The variables representing locations, features, and location dependency sets are *observed* and hence are shown as shaded; the rest of the variables are *latent* and are shown unshaded. The latent variable λ^{F_j} takes on a value k, indicating the selection of the location set LS_k, with a probability $\theta_k(F_j)$. As was shown in Figure 1, edges among location variables capture inter-dependencies among locations. The rectangular plate notation is used to represent replication of feature and location set variables with the same dependencies. The lack of feature–feature edges captures the conditional independencies among features given location sets.

Under the independence assumptions and the structure of our model described earlier, the joint probability of the location indicator vector $\vec{l}^P$ and the feature vector $\vec{f}^P$ is expressed as:

$$\Pr(\vec{l}^P, \vec{f}^P) = \Pr(\vec{l}^P)\Pr(\vec{f}^P|\vec{l}^P) = \\ = \prod_{i=1}^{q}\Pr(L_i = l_i^P|\mathrm{Pa}(L_i)) \times \prod_{j=1}^{d}\sum_{k=1}^{q}\theta_k^{\vec{l}^P}(F_j)\Pr(F_j = f_j^P|L_k = l_k^P, \mathrm{Pa}(L_k)), \quad (5)$$

where each term corresponds to a parameter of the generative model as described below:

a. $\prod_{i=1}^{q}\Pr(L_i = l_i^P|\mathrm{Pa}(L_i))$ is the factorization of the joint probability $\Pr(\vec{l}^P) = \Pr(L_1 = l_1^P, \ldots, L_q = l_q^P)$, over the individual q location indicator values;
b. $\Pr(F_j = f_j^P|L_k, \mathrm{Pa}(L_k))$ denotes the conditional probability of a feature value f_j^P $(1 \le j \le d$, where d is the total number of features), given the values taken by a location variable L_k and its parents $\mathrm{Pa}(L_i)$ comprising the location dependency set LS_k (under the current model G);
c. $\theta_k^{\vec{l}^P}(F_j)$ denotes the probability that the location dependency set LS_k was selected for a given feature F_j and a location indicator vector $\vec{l}^P$.

4 Model learning and protein multi-location prediction

In this section, we introduce the procedure used for learning the structure and the parameters of our generative model and for predicting multiple locations for proteins. We present an expectation maximization (EM) algorithm to estimate the hidden parameters and explain the inference technique used for multi-location prediction.

As our goal is to predict multiple locations for proteins, we use the probabilistic generative model presented in Section 3 to predict a 0/1 value for each location variable L_i. To obtain the model, we use an iterative process (see Fig. 4) in which the structure of a Bayesian network and the parameters of the generative model [shown in Equation (5)] are learned. Each iteration consists of first learning a network structure and estimating its parameters, and following the learning by performance assessment of the resulting model by using it to infer the locations of proteins in the training dataset. This process is continued until a stopping criterion is met, namely, until the prediction performance of the learned model on the proteins from the training-set does not improve between two successive iterations. Typically the process does not require more than ten iterations to complete. To measure prediction performance in each iteration, we use the F_1-score metric, which is formally defined later in Section 5.2. We next discuss the procedures used for learning the structure and the parameters of our model.

4.1 Model learning

In each iteration of the learning process, we obtain a Bayesian network structure of locations using the software package BANJO (Smith *et al.*, 2006) and estimate the model parameters shown in the previous section in Equation (5). The initial Bayesian network structure is learned from protein locations in the training set, and iteratively updated to reflect the most-recently estimated locations.

To estimate the model parameters described in components (a) and (b) of Equation (5), we calculate the maximum likelihood estimates from frequency counts in the training dataset. As for component (c) there, the location set probability $\theta_k^{\vec{l}}(F_j)$ for a given location indicator vector $\vec{l}$ and a feature F_j cannot be directly computed from the dataset. We thus use an EM algorithm (Dempster *et al.*, 1977) to estimate the *hidden* parameter $\theta^{\vec{l}}(F_j)$, as described next.

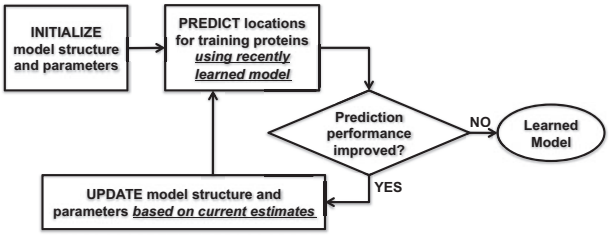

Fig. 4. A summary of our model-learning process. The rectangular boxes represent steps in the learning process, the diamond indicates checking for a stopping criterion, and the oval represents the output, which in our case is the learned model. Directed edges indicate the order among steps

In the *E-step*, for each protein P and each of its feature values f_j^P in the training set, we compute the probability of a location set LS_k to be used to determine the protein's feature value as:

$$\Pr(\lambda = k|f_j^P, \vec{l}^P) = \frac{\theta_k^{\vec{l}^P}(F_j)\Pr(f_j^P|l_k^P, \mathrm{Pa}(L_k))}{\sum_{k=1}^{q}\theta_k^{\vec{l}^P}(F_j)\Pr(f_j^P|l_k^P, \mathrm{Pa}(L_k))}. \quad (6)$$

In Equation (6), for each location indicator vector $\vec{l}^P$ and feature F_j, the distribution $\theta^{\vec{l}^P}(F_j)$ over q location sets is initialized as uniform; thus initially $\theta_k^{\vec{l}^P}(F_j) = 1/q$ for all k, $(1 \le k \le q)$. The conditional probability, $\Pr(f_j^P|l_k^P, \mathrm{Pa}(L_k))$, of a feature value f_j^P given the location set LS_k (where $LS_k = L_k \cup Pa(L_k)$) is initialized to the maximum likelihood estimate computed using the training dataset.

In the *M-step*, we re-estimate all the model parameters. For each location indicator vector $\vec{l}$ and feature F_j, the probability of a location dependency set LS_k is re-estimated as:

$$\theta_k^{\vec{l}}(F_j) = \frac{\sum_{v_j}\sum_{\{P|\vec{l}^P=\vec{l}, f_j^P=v_j\}}\Pr(\lambda^{F_j} = k|f_j^P, \vec{l}^P)\Pr(f_j^P|\vec{l}^P)}{\sum_{k=1}^{q}\sum_{v_j}\sum_{\{P|\vec{l}^P=\vec{l}, f_j^P=v_j\}}\Pr(\lambda^{F_j} = k|f_j^P, \vec{l}^P)\Pr(f_j^P|\vec{l}^P)}, \quad (7)$$

where v_j is a feature value of F_j and k denotes the selection of the dependency set LS_k. That is, in the numerator, for each feature, F_j, we go over all feature values v_j that F_j takes, and all proteins in the set that have this feature value; we sum the probability of having used the dependency set LS_k to generate feature value v_j—weighted by the probability of observing that feature value. The denominator is a normalization factor ensuring that probabilities sum to 1. The probability of a set LS_k to be selected for determining f_j^P, $\Pr(\lambda^{F_j} = k|f_j^P, \vec{l}^P)$, is calculated in the *E-step* [see Equation (6)]. Note that $\theta_k^{\vec{l}}(F_j)$ is computed separately for each feature since feature-values are determined independently of each other during protein generation.

To re-estimate the conditional probability $\Pr(F_j = v_j|LS_k)$, we introduce the notation $\vec{l}^P_{LS_k}$ to denote the *restriction* of the location indicator vector $\vec{l}^P$ to only those locations that are members in the location dependency set LS_k. The conditional probability is then calculated as:

$$\Pr(F_j = v_j|L_k, \mathrm{Pa}(L_k)) = \Pr(F_j = v_j|\mathrm{LS}_k) = \\ \frac{\sum_{\{P|\vec{l}^P_{LS_k}=\vec{l}_{LS_k}, f_j^P=v_j\}}\Pr(\lambda^{F_j} = k|f_j^P, \vec{l}^P)\Pr(f_j^P|\vec{l}^P)}{\sum_{v_j}\sum_{\{P|\vec{l}^P_{LS_k}=\vec{l}_{LS_k}, f_j^P=v_j\}}\Pr(\lambda^{F_j} = k|f_j^P, \vec{l}^P)\Pr(f_j^P|\vec{l}^P)}.$$

This re-estimation formula is similar to the one shown in Equation (7), but taking into account only those proteins in the training set that are localized to the locations included in the dependency set LS_k.

The process of alternating between the E-step and the M-step is carried out until convergence is reached, i.e. until changes to the hidden parameter values between iterations are no greater than 0.05. Throughout the estimation process, we use Laplace smoothing to avoid overfitting, by adding fractional pseudocounts to observed counts of events (Russell and Norvig, 2010). The smoothing parameter (α) is set to 0.5, which is close to the count of rare events and almost insignificant compared with counts of frequent ones. We next present the inference procedure that we use for predicting protein locations.

4.2 Multiple location prediction

Given a protein P, represented as a feature vector $\vec{f}^P$, our task is to predict its location indicator vector $\vec{l}^P$, i.e. we need to assign a 0/1 value to each of its location indicators l_i^P $(1 \leq i \leq q)$. Under the Bayesian network model, this task translates to inferring the value of the random variable L_i, which in turn depends on the values of its parent nodes $\mathrm{Pa}(L_i)$. We thus infer the values of the location dependency set $\{L_i\} \cup \mathrm{Pa}(L_i)$. The inference procedure aims to assign values to L_i and to $\mathrm{Pa}(L_i)$ such that the conditional probability, $\Pr(L_i, \mathrm{Pa}(L_i)|\vec{f}^P)$ is maximized.

To infer these values, we follow an iterative process. We start by initializing all location indicators in $\vec{l}^P$ to 0. For any value-assignment, l_i to L_i, we denote by $\vec{\mathrm{Pa}}(l_i)$ the values assigned to all parents of L_i. We also denote by $\overline{L}_i$ the current value assignment for all location random variables in the network other than L_i and $\mathrm{Pa}(L_i)$. In each iteration, we consider in turn each of the random variables L_i. For all possible value assignments, l_i, $\vec{\mathrm{Pa}}(l_i)$, to L_i and $\mathrm{Pa}(L_i)$, respectively, we calculate the conditional probability, $\Pr(L_i = l_i, \mathrm{Pa}(L_i) = \vec{\mathrm{Pa}}(l_i)|\vec{f}^P, \overline{L}_i)$. The value assignment to L_i that produces the highest probability is the one used as the current estimate for L_i. As noted earlier, the process typically requires about ten iterations to reach convergence.

We next describe our experiments and the results obtained using the protein generation model.

5 Experiments and results

We implemented our algorithms for learning parameters of the generative model and for inferring locations using Python. We have applied our system MDLoc to the largest available dataset of multi-localized proteins, previously used for training YLoc$^+$ (Briesemeister *et al.*, 2010a). Next, we describe the dataset and the evaluation methods we use, followed by experiments and results obtained using MDLoc. We also provide several specific examples demonstrating the utility of incorporating location inter-dependencies into the prediction process.

5.1 Data

In our experiments, we use a dataset first constructed for an extensive comparison of multi-location prediction systems as part of the evaluation of YLoc$^+$ (Briesemeister *et al.*, 2010a). It contains 5447 single-localized proteins, originally published by Höglund *et al.* (2006), and 3056 multi-localized proteins, originally published as part of the DBMLoc dataset (Zhang *et al.*, 2008). As in a true prediction scenario it is not known a priori whether a protein may localize to a single or to multiple locations, we train our system on the combined set of proteins, thus enabling it to handle the actual prediction task. The dataset is already homology-reduced, i.e. proteins sharing >80% sequence identity with another protein in the dataset were removed. We compare the performance of our system to that of others using only multi-localized proteins (3056 proteins) because the only results publicly available for the other systems were obtained on this dataset (Briesemeister *et al.*, 2010a). The single-localized proteins are from the following locations (abbreviations and number of proteins per location are given in parentheses): cytoplasm (*cyt*, 1411 proteins); endoplasmic reticulum (*ER*, 198); extra cellular space (*ex*, 843); golgi apparatus (*gol*, 150); lysosome (*lys*, 103); mitochondrion (*mi*, 510); nucleus (*nuc*, 837); membrane (mem, 1238); peroxisome (*per*, 157). The multi-localized proteins are from the following pairs of locations: *cyt_nuc*: 1882 proteins; *ex_mem*: 334; *cyt_mem*: 252; *cyt_mi*: 240; *nuc_mi*: 120; *ER_ex*: 115; *ex_nuc*: 113. Note that all the multi-location subsets used have over 100 representative proteins. We use the exact same representation of a 30-dimensional feature vector as used for evaluating YLoc$^+$ (for further details see Briesemeister *et al.*, 2010b): (i) thirteen features derived directly from the protein sequence data; (ii) nine features constructed using pseudo-amino acid composition (Chou, 2001); (iii) two *annotation-based* features constructed using two distinct groups of PROSITE patterns; (iv) six *annotation-based* features based on GO-annotations.

5.2 Experimental setting and performance measures

We compare the performance of MDLoc to that of our preliminary system (Simha and Shatkay, 2014) and to other systems, specifically, YLoc$^+$ (Briesemeister *et al.*, 2010a), Euk-mPLoc (Chou and Shen, 2007), WoLF PSORT (Horton *et al.*, 2007) and KnowPred$_{\mathrm{site}}$ (Lin *et al.*, 2009), whose results on the multi-localized proteins are described in a previously published comprehensive study by Briesemeister *et al.* (2010a). The comparison uses the exact same dataset from that study, and employs multiple runs of stratified 5-fold cross-validation. That is, we ran 5-fold-cross-validation five complete times (25 runs in total), using a different five-way split each time. The use of multiple runs with multiple splits helps validate the stability and the significance of the results. The total training time for our system for the 25 training experiments is about 8 hours (wall-clock), when running on a standard Dell Poweredge machine with 32 AMD Opteron 6276 processors.

To formally define the evaluation measures we use, let D be a dataset containing proteins. For a given protein P, let $M^P = \{s_i \mid l_i^P = 1, \text{where } 1 \leq i \leq q\}$ be the set of locations to which protein P localizes according to the dataset, and let $\hat{M}^P = \{s_i \mid \hat{l}_i^P = 1, \text{where } 1 \leq i \leq q\}$ be the set of locations that a classifier predicts for P, where $\hat{l}_i^P$ is the 0/1 prediction obtained for location s_i. We use *adapted* measures of multi-label precision and recall denoted Pre_{s_i} and Rec_{s_i} and defined as follows (Briesemeister *et al.*, 2010a):

$$\mathrm{Pre}_{s_i} = \frac{1}{|\{P \in D | s_i \in \hat{M}^P\}|} \times \sum_{P \in D | s_i \in \hat{M}^P} \frac{|M^P \cap \hat{M}^P|}{|\hat{M}^P|};$$

$$\mathrm{Rec}_{s_i} = \frac{1}{|\{P \in D | s_i \in M^P\}|} \times \sum_{P \in D | s_i \in M^P} \frac{|M^P \cap \hat{M}^P|}{|M^P|}.$$

We also use the *adapted* measure of *accuracy* proposed by Tsoumakas *et al.* (2010) for evaluating multi-label classification. Some of these measures have also been previously used for multi-location evaluation (Briesemeister *et al.*, 2010a; He *et al.*, 2012).

The multi-label accuracy and the F_1-*label* score used for the evaluation of YLoc$^+$ (Briesemeister *et al.*, 2010a) are computed as:

$$\text{Acc} = \frac{1}{|D|}\sum_{P\in D}\frac{|M^P \cap \hat{M}^P|}{|M^P \cup \hat{M}^P|} \text{ and } F_1\text{-}label = \frac{1}{|S|}\sum_{s_i\in S}\frac{2\times \text{Pre}_{s_i}\times \text{Rec}_{s_i}}{\text{Pre}_{s_i}+\text{Rec}_{s_i}}.$$

Finally, to evaluate the correctness of predictions made for each location s_i, we use the *standard precision* and *recall* measures, denoted by Pre-Std$_{s_i}$ and Rec-Std$_{s_i}$ and defined as: Pre-Std$_{s_i}$ = TP/(TP + FP) and Rec-Std$_{s_i}$ = TP/(TP + FN), where *TP* (*true positives*) denotes the number of proteins that localize to s_i and are predicted to localize to s_i, *FP* (*false positives*) denotes the number of proteins that do not localize to s_i but are predicted to localize to s_i, and *FN* (*false negatives*) denotes the number of proteins that localize to s_i but are not predicted to localize to s_i. The F_1-score for location s_i is defined as:

$$F_1\text{-score}_{s_i} = \frac{2\times \text{Pre-Std}_{s_i}\times \text{Rec-Std}_{s_i}}{(\text{Pre-Std}_{s_i}+\text{Rec-Std}_{s_i})}.$$

5.3 Classification results

In this section, we compare the performance of our system with that of existing location prediction systems over the commonly used set of multi-localized proteins. We also report experiments using the *combined set* of single and multi-localized proteins as mentioned in Section 5.1. Our analysis includes an examination of the per-location break-up of the results. Additionally, we focus on several specific examples demonstrating the benefit of incorporating location interdependency into our prediction system.

Table 1A shows the F_1-*label* score and the *accuracy* obtained by our current system MDLoc compared with those obtained by other multi-location predictors [YLoc$^+$, Euk-mPLoc, WoLF PSORT and KnowPred$_{site}$ as reported by Briesemeister *et al.* (2010a) in Table 3] and by our preliminary system (Bayesian network classifiers, denoted BNCs, Simha and Shatkay, 2014), using the same set of multi-localized proteins and evaluation measures. The table shows that MDLoc performs better than the existing top-systems, including YLoc$^+$ which has the best performance reported so far and whose predictions are based only on location-combinations in the training set. In contrast, MDLoc is not limited to the location-combinations in the training set, as it represents dependency of features on location-combinations in a generalizable manner, and directly captures inter-dependencies among locations. The only other system that attempts to capture such dependencies is our preliminary system BNCs.

To illustrate the use of interdependency, consider the protein *Securin* which is included in our dataset and localizes to both the cytoplasm (*cyt*) and the nucleus (*nuc*). *Securin*, initially present in the cytoplasm, translocates to the nucleus in response to DNA damage (Kim *et al.*, 2007). While MDLoc assigns it to both the *cyt* and the *nuc*, YLoc$^+$ assigns it to the *nuc* only. Our system utilizes the dependency between *nuc* and *cyt* (represented by a directed edge between the two locations, see Fig. 1) to make an accurate multi-location prediction. Location dependencies reflect intrinsic relationships that locations share with each other, and in this case, it is well-known that proteins shuttle continuously between the nucleus and the cytoplasm to control a variety of functions such as cell cycle progression (Gama-Carvalho and Carmo-Fonseca, 2001). MDLoc's benefit from capturing the interdependency between *cyt* and *nuc* is also reflected in its significantly higher *Multilabel-Precision* and *Multilabel-Recall* (Pre$_{s_i}$ and Rec$_{s_i}$, respectively) for the *cyt* and the *nuc* as shown in Table 1B. As another example, consider *Protransforming growth factor alpha (TGF-alpha)*, a protein that assists in cell growth (See NCBI's Gene database, http://www.ncbi.nlm.nih.gov/gene/7039), localizes to both the extracellular space (*ex*) and the plasma membrane (*mem*), and is correctly assigned by MDLoc to both. Here MDLoc employs the well-known dependency between the extracellular space and the plasma membrane, as reflected for instance in the exocytic trafficking pathway (Tokarev *et al.*, 2000), and in the transition of proteins such as *hsp 90-alpha* (initiated by *TGF-alpha*) from the extracellular space to the plasma membrane in response to stress (Cheng *et al.*, 2008). Again, the value of utilizing interdependencies is demonstrated in MDLoc's significantly improved precision in terms of *Multilabel-Precision* (Pre$_{s_i}$) on the *ex* and *mem* proteins (while still retaining a similar level of recall, Rec$_{s_i}$, to that of YLoc$^+$).

Table 1. Multi-location prediction results, averaged over 25 runs of 5-fold cross-validation, for *multi-localized* proteins only

(A)

	MDLoc	BNCs	**YLoc$^+$**	Euk-mPLoc	WoLF PSORT	KnowPred$_{site}$
F_1-label	**0.71 (± 0.02)**	0.66 (± 0.02)	0.68	0.44	0.53	0.66
Acc	**0.68 (± 0.01)**	0.63 (± 0.01)	0.64	0.41	0.43	0.63

(B)

		cyt (2374)	*p*-value	**nuc (2115)**	*p*-value	**mem (586)**	*p*-value	**ex (562)**	*p*-value	**mi (360)**	*p*-value
Rec$_{si}$	MDLoc	**0.750 (±0.012)**	≪0.001	**0.776 (±0.014)**	≪0.001	0.527 (±0.022)	0.01	0.547 (±0.035)	0.01	0.519 (±0.026)	0.04
	YLoc$^+$	0.712 (±0.009)		0.728 (±0.011)		**0.543 (±0.018)**		**0.573 (±0.026)**		**0.536 (±0.031)**	
Pre$_{si}$	MDLoc	**0.911 (±0.008)**	≪0.001	**0.929 (±0.008)**	0.03	**0.807 (±0.036)**	≪0.001	**0.833 (±0.044)**	≪0.001	**0.832 (±0.042)**	≪0.001
	YLoc$^+$	0.893 (±0.010)		0.924 (±0.008)		0.764 (±0.029)		0.740 (±0.053)		0.765 (±0.033)	
Rec-Std$_{si}$	MDLoc	**0.817 (±0.021)**	≪0.001	**0.746 (±0.028)**	≪0.001	0.588 (±0.042)	0.04	0.385 (±0.058)	0.3	0.388 (±0.062)	0.03
	YLoc$^+$	0.786 (±0.020)		0.684 (±0.015)		**0.614 (±0.042)**		**0.401 (±0.037)**		**0.429 (±0.060)**	
Prec-Std$_{si}$	MDLoc	**0.942 (±0.009)**	0.01	0.904 (±0.014)	0.02	**0.794 (±0.039)**	≪0.001	**0.830 (±0.046)**	≪0.001	**0.784 (±0.057)**	≪0.001
	YLoc$^+$	0.935 (±0.009)		**0.914 (±0.014)**		0.730 (±0.047)		0.771 (±0.055)		0.670 (±0.055)	

Standard deviations are shown in parentheses (if available). The highest values are shown in boldface. (**A**) Overall F_1-*label* scores and overall accuracy (*Acc*) obtained using our current system MDLoc, our preliminary system (denoted BNCs, Simha and Shatkay, 2014), YLoc$^+$ (Briesemeister *et al.*, 2010a), Euk-mPLoc (Chou and Shen, 2007), WoLF PSORT (Horton *et al.*, 2007) and KnowPred$_{site}$ (Lin *et al.*, 2009). The four rightmost columns are taken directly from Table 3 in the article by Briesemeister *et al.* (2010a). (**B**) Per location scores: *Multilabel-Precision* (Pre$_{s_i}$) and *Recall* (Rec$_{s_i}$), as well as *standard precision* (Pre-Std$_{s_i}$) and *recall* (Rec-Std$_{s_i}$), for each location s_i, for MDLoc and YLoc$^+$. Results for YLoc$^+$ were reproduced using our five-way splits. The *p*-values indicate the statistical significance of the differences between the values obtained from MDLoc and from YLoc$^+$.

Table 2. Multi-location prediction results, per location, averaged over 25 runs of 5-fold cross-validation, for the *combined set* of single- and multi-localized proteins

		cyt (3785)	*p*-value	nuc (2952)	*p*-value	ex (1405)	*p*-value	mem (1824)	*p*-value	mi (870)	*p*-value
Rec_{s_i}	MDLoc	**0.825 (±0.009)**	≪0.001	**0.830 (±0.010)**	≪0.001	**0.780 (±0.020)**	≪0.001	**0.822 (±0.012)**	≪0.001	**0.773 (±0.013)**	≪0.001
	BNCs	0.795 (±0.011)		0.784 (±0.017)		0.737 (±0.022)		0.780 (±0.014)		0.730 (±0.025)	
Pre_{s_i}	MDLoc	**0.819 (±0.013)**	0.03	0.822 (±0.014)	0.02	0.864 (±0.020)	≪0.001	0.872 (±0.014)	≪0.001	0.861 (±0.024)	0.001
	BNCs	0.809 (±0.018)		**0.832 (±0.013)**		**0.912 (±0.019)**		**0.900 (±0.012)**		**0.885 (±0.023)**	
$Rec\text{-}Std_{s_i}$	MDLoc	**0.867 (±0.015)**	0.1	**0.808 (±0.021)**	≪0.001	**0.715 (±0.030)**	≪0.001	**0.842 (±0.017)**	≪0.001	**0.719 (±0.028)**	≪0.001
	BNCs	0.861 (±0.014)		0.736 (±0.031)		0.652 (±0.024)		0.805 (±0.017)		0.664 (±0.034)	
$Prec\text{-}Std_{s_i}$	MDLoc	**0.854 (±0.014)**	0.001	0.783 (±0.020)	0.6	0.839 (±0.028)	≪0.001	0.882 (±0.014)	≪0.001	0.843 (±0.026)	0.001
	BNCs	0.840 (±0.011)		**0.786 (±0.026)**		**0.906 (±0.022)**		**0.900 (±0.015)**		**0.873 (±0.034)**	

The table shows the same measures used in Table 1B obtained over the combined dataset using our current system MDLoc, and using our preliminary system (denoted BNCs) (Simha and Shatkay, 2014). The highest values are shown in boldface. The *p*-values indicate the statistical significance of the differences between the values obtained from MDLoc and those obtained from BNCs. Standard deviations are shown in parentheses.

Table 3. Multi-location prediction results, *per location-combination*, obtained using one run of 5-fold cross-validation, for multi-localized proteins only

		cyt_nuc (1882)	ex_mem (334)	cyt_mem (252)	cyt_mi (240)	nuc_mi (120)	ER_ex (115)	ex_nuc (113)
Both locations correct	MDLoc	**1253 (66.6%)**	**34 (10.2%)**	**31 (12.3%)**	**36 (15%)**	**15 (12.5%)**	**35 (30.4%)**	51 (45.1%)
	BNCs	976 (51.9%)	16 (4.8%)	15 (6%)	25 (10.4%)	11 (9.2%)	16 (13.9%)	**54 (47.8%)**
First location correct	MDLoc	**1603 (85.2%)**	**87 (26%)**	**186 (73.8%)**	164 (68.3%)	**43 (35.8%)**	**66 (57.4%)**	**73 (64.6%)**
	BNCs	1578 (83.8%)	60 (18%)	174 (69%)	**165 (68.8%)**	37 (30.8%)	**66 (57.4%)**	68 (60.2%)
Second location correct	MDLoc	**1481 (78.7%)**	**258 (77.2%)**	**82 (32.5%)**	**99 (41.3%)**	**67 (55.8%)**	**51 (44.3%)**	**72 (63.7%)**
	BNCs	1240 (65.9%)	246 (73.7%)	68 (27%)	85 (35.4%)	64 (53.3%)	27 (23.5%)	68 (60.2%)

For each combination, the table shows the number of proteins with correct predictions for both locations, for the first of the two locations, and for the second of the two locations, using MDLoc and using our preliminary system (BNCs, Simha and Shatkay, 2014). The highest values are shown in boldface.

As an example for MDLoc's ability to handle proteins whose *location-combination* is not included in the training set, consider *Transmembrane emp24 domain-containing protein 7 (emp24)*. It localizes to the ER and transports secretory proteins to the golgi complex (*gol*) (Belden and Barlowe, 1996). (The tables shown do not include *ER* and *gol* proteins, as the number of proteins from either of these locations in the dataset is very small.) MDLoc assigns *emp24* to both the *ER* and the *gol*, whereas YLoc$^+$ assigns it to the *ER* only. As indicated before, MDLoc makes use of the dependency which captures the relationship between the *ER* and the *gol*, both of which act as components in the exocytic trafficking pathway (Tokarev *et al.*, 2000). We thus see that MDLoc is not restricted to predicting only pre-defined location-combinations.

Table 1B shows the per-location prediction results for multi-localized proteins obtained by MDLoc compared with those obtained by YLoc$^+$ (Briesemeister *et al.*, 2010a). Per-location predictions for the other systems are not shown here as they are not publicly available. Results are shown for the five locations with the largest number of associated proteins. For each location s_i, we show *Multilabel-Precision* (Pre_{s_i}) and *Multilabel-Recall* (Rec_{s_i}) as well as *standard precision* ($Pre\text{-}Std_{s_i}$) and *recall* ($Rec\text{-}Std_{s_i}$). For the cytoplasm and the nucleus, which have a large number of proteins, the precision and recall values obtained using MDLoc are significantly higher in most cases than those obtained using YLoc$^+$. For locations with much fewer proteins, while the recall values when using MDLoc are marginally lower than when using YLoc$^+$, MDLoc's precision values are typically significantly higher than those of YLoc$^+$. We note that YLoc$^+$ assigns each protein to *all* the locations whose probability exceeds a pre-defined threshold; as such, the number of locations it assigns exceeds that to which the protein actually localizes resulting in a lower precision. In contrast, MDLoc does not simply assign a protein to each location whose probability is higher, but rather, it simultaneously considers a *set* of locations and assigns each protein to the set whose overall probability is high, leading to a higher precision.

Table 2 shows the per-location prediction results on the *combined dataset of both single- and multi-localized proteins* obtained by MDLoc, in comparision to those obtained by BNCs (Simha and Shatkay, 2014). While MDLoc's precision values are somewhat lower than those of BNCs, MDLoc's recall is typically higher. MDLoc simultaneously infers the probability of a *set* of locations; in contrast, BNCs uses an independent Bayesian network structure to infer the probability of each location separately. As such, the likelihood of BNCs to correctly assign the combination of several locations to a protein is much lower than its probability to correctly assign a single location, which directly translates into a relatively low recall measure. When using MDLoc, the increase in recall values for almost all cases is higher than the decrease in the precision values, except in the case of the extracellular space (*ex*). Notably, proteins in the extracellular space all originate from or are bound toward another location within the cell and as such predicting them as extracellular is challenging for most prediction systems.

Moreover, MDLoc assigns some proteins hitherto known to localize only to a single location into multiple locations. It is likely that at least some of these additional predicted locations are indeed correct and can be the subject of an experimental validation. For instance, *Calreticulin (Cal)* is currently annotated by SwissProt as localized to the *ER* only. However, MDLoc assigns it to both the *ER* and the *ex*, and work by Gold *et al.* (2010) suggests that it indeed relocates from the *ER* to the *ex*.

We also examine the statistically significant differences in the *Multilabel-Recall* for the location with the highest number of

multi-localized proteins (cytoplasm, 2374 proteins) and the location with the lowest number (endoplasmic reticulum, 115 proteins). The *Multilabel-Recall* for cytoplasm (Rec_{cyt}) increases from 0.80 when classifying using BNCs, to 0.83 when using MDLoc. Similarly, the *Multilabel-Recall* for endoplasmic reticulum (Rec_{ER}, not shown in Table 2) increases from 0.64 to 0.69. This analysis demonstrates the advantage of using MDLoc for predicting protein locations, not just for locations that have a large number of associated proteins but also for locations that are associated with relatively few proteins.

Table 3 shows the prediction results obtained using MDLoc in contrast to those obtained using BNCs (Simha and Shatkay, 2014) for all location-combinations, using multi-localized proteins only. For each location combination in the dataset, we show the number of proteins with correct predictions for both locations, as well as for the first of the two locations, and for the second, separately. For almost all combinations, the number of proteins whose location is correctly predicted by MDLoc is significantly higher than the corresponding number when using BNCs. We examine the predictions for the location-combination with the highest number of proteins (cytoplasm and nucleus—1882 proteins) and its constituent locations (cytoplasm—1411 and nucleus—837 proteins). As can be seen from the table, the number of *multi-localized* proteins whose combined-location is correctly predicted increases significantly from 976 when classifying using BNCs, to 1253 when using MDLoc. The increase shows that location inter-dependencies learnt using MDLoc help to improve predictions for multi-localized proteins.

6 Conclusion and future work

We presented a new probabilistic generative model for protein localization based on Bayesian networks and a mixture model, and developed a system MDLoc, to predict multiple locations for proteins. MDLoc takes advantage of the location inter-dependencies and location-feature dependency to provide a generalizable method for predicting multiple locations for proteins. Our results demonstrate the utility of using location inter-dependencies in the prediction process, and show that the performance of MDLoc improves over current state-of-the-art reported results.

MDLoc significantly improves over our own preliminary method which used a relatively simple collection of Bayesian network classifiers (Simha and Shatkay, 2014) whose performance was on par with that of $YLoc^{+}$ (Briesemeister *et al.*, 2010a). In our previous method, location inter-dependencies were not learnt as part of the model but rather captured based on simple estimates of location values. In contrast, MDLoc uses a generative model comprising Bayesian networks to directly address and capture inter-dependencies among locations, and a mixture model to represent *feature dependency on location-combinations*. We iteratively learn a Bayesian network over location variables while estimating the locations using expectation maximization.

Our future work includes exploring alternative ways to learn the mixture model parameters, to evaluate the model learned in each iteration of our current process, and to perform multi-location inference. We will also conduct experiments testing our system's performance on more complex location-combinations. Having a larger set of multi-localized proteins from plant-, fungi- and animal-specific organelles will also enable us to explore the possibility of building a model for each taxonomic group.

As another direction, we will also experiment with features other than the ones previously used by $YLoc^{+}$, utilizing multiple data-sources, which is likely to be more appropriate for representing proteins in the context of multi-location prediction.

Acknowledgements

We are grateful to J. Simmons, C. Shannon and H. Wei for assisting us with the development of a website to enable web access for MDLoc.

Conflict of Interest: none declared.

References

Alberts,B. *et al.* (2002) *Molecular Biology of the Cell.* Vol. 4. Garland Science, New York.

Bakheet,T. and Doig,A. (2009) Properties and identification of human protein drug targets. *Bioinformatics,* **25**, 451–457.

Belden,W. and Barlowe,C. (1996) Erv25p, a component of copii-coated vesicles, forms a complex with Emp24p that is required for efficient endoplasmic reticulum to golgi transport. *J. Biol. Chem.*, **271**, 26939–26946.

Blum,T. *et al.* (2009) MultiLoc2: integrating phylogeny and Gene Ontology terms improves subcellular protein localization prediction. *BMC Bioinformatics,* **10**, 274.

Briesemeister,S. *et al.* (2010a) Going from where to why – interpretable prediction of protein subcellular localization. *Bioinformatics,* **26**, 1232–1238.

Briesemeister,S. *et al.* (2010b). YLoc—an interpretable web server for predicting subcellular localization. *Nucleic Acids Res.,* **38**(Web Server issue), W497–W502.

Cheng,C. *et al.* (2008) Transforming growth factor alpha (TGFalpha)-stimulated secretion of HSP90alpha: using the receptor LRP-1/CD91 to promote human skin cell migration against a TGFbeta-rich environment during wound healing. *Mol. Cell. Biol.,* **28**, 3344–3358.

Chou,K. (2001) Prediction of protein cellular attributes using pseudo-amino acid composition. *Cell Mol. Life Sci.,* **43**, 246–255.

Chou,K. *et al.* (2011) iLoc-Euk: a multi-label classifier for predicting the subcellular localization of singleplex and multiplex eukaryotic proteins. *PLoS One,* **6**, e18258.

Chou,K. and Shen,H. (2007) Euk-mPLoc: a fusion classifier for large-scale eukaryotic protein subcellular location prediction by incorporating multiple sites. *J. Proteome Res.,* **6**, 1728–1734.

Dempster,A. *et al.* (1977) Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Series B,* **39**, 1–38.

Dreger,M. (2003) Proteome analysis at the level of subcellular structures. *Eur. J. Biochem.*, **270**, 589–599.

Emanuelsson,O. *et al.* (2000) Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *J. Mol. Biol.,* **300**, 1005–1016.

Gama-Carvalho,M. and Carmo-Fonseca,M. (2001) The rules and roles of nucleocytoplasmic shuttling proteins. *FEBS Lett.*, **498**, 157–163.

Garg,A. and Raghava,G. (2008) ESLpred2: Improved method for predicting subcellular localization of eukaryotic proteins. *BMC Bioinformatics,* **9**, 503.

Gold,L. *et al.* (2010) Calreticulin: non-endoplasmic reticulum functions in physiology and disease. *FASEB J.,* **24**, 665–683.

He,J. *et al.* (2012) Imbalanced multi-modal multi-label learning for subcellular localization prediction of human proteins with both single and multiple sites. *PLoS One,* **7**, e37155.

Höglund,A. *et al.* (2006) MultiLoc: prediction of protein subcellular localization using N-terminal targeting sequences, sequence motifs, and amino acid composition. *Bioinformatics,* **22**, 1158–1165.

Horton,P. *et al.* (2007) WoLF PSORT: protein localization predictor. *Nucleic Acids Res.,* **35** (Web Server issue), W585–W587.

Kim,D. *et al.* (2007). Securin induces genetic instability in colorectal cancer by inhibiting double-stranded DNA repair activity. *Carcinogenesis,* **28**, 749–759.

King,B. and Guda,C. (2007) ngLOC: an n-gram-based Bayesian method for estimating the subcellular proteomes of eukaryotes. *Genome Biol.,* **8**, R68.

Li,L. *et al.* (2012) Prediction of protein subcellular multi-localization based on the general form of Chou's pseudo amino acid composition. *Protein Pept. Lett.,* **19**, 375–387.

Lin,H. *et al.* (2009) Protein subcellular localization prediction of eukaryotes using a knowledge-based approach. *BMC Bioinformatics,* **10** (Suppl. 15), 8.

Murphy,R. (2010) Communicating subcellular distributions. *Cytometry A.*, **77**, 686–692.

Nair,R. and Rost,B. (2008) Protein subcellular localization prediction using artificial intelligence technology. *Funct. Proteomics,* **484**, 435–463.

Nakai,K. and Kanehisa,M. (1991) Expert system for predicting protein localization sites in gram-negative bacteria. *Proteins,* **11**, 95–110.

Pohlschroder,M. *et al.* (2005) Diversity and evolution of protein translocation. *Annu. Rev. Microbiol.,* **59**, 91–111.

Rost,B. *et al.* (2003) Automatic prediction of protein function. *Cell Mol. Life Sci.,* **60**, 2637–2650.

Russell,S. and Norvig,P. (2010) *Artificial Intelligence—A Modern Approach,* 3rd edn. Pearson Education, New Jersey, USA, 3rd edition.

Schiffer,M. *et al.* (1992) The function of tryptophan residues in membrane proteins. *Protein Eng.,* **5**, 213–214.

Scholkopf,B. and Smola,A. (2002) *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, MA, USA.

Shatkay,H. *et al.* (2007) SherLoc: high-accuracy prediction of protein subcellular localization by integrating text and protein sequence data. *Bioinformatics,* **23**, 1410–1417.

Simha,R. and Shatkay,H. (2014) Protein (multi-)location prediction: using location inter-dependencies in a probabilistic framework. *Algorithm Mol. Biol,* **9**, 8.

Simpson,J. *et al.* (2000) Systematic subcellular localization of novel proteins identified by large-scale cDNA sequencing. *EMBO Rep.,* **1**, 287–292.

Smith,A. *et al.* (2006) Computational inference of neural information flow networks. *PLoS Comput. Biol.,* **2**, e161.

Tokarev,A. *et al.* (2000) Overview of Intracellular Compartments and Trafficking Pathways. *Landes Bioscience*, Texas, USA.

Tomicic,M. *et al.* (2013) Human three prime exonuclease TREX1 is induced by genotoxic stress and involved in protection of glioma and melanoma cells to anticancer drugs. *Biochim. Biophys. Acta* **1833**, 1832–1843.

Tsoumakas,G. *et al.* (2010). Mining multi-label data. In: Maimon,O. and Rokach,L. (eds). *Data Mining and Knowledge Discovery Handbook,* 2nd edn. Springer, Heidelberg, 667–685.

Zhang,S. *et al.* (2008) DBMLoc: a database of proteins with multiple subcellular localizations. *BMC Bioinformatics,* **9**, 127.