Selective Sampling with a Hierarchical Latent Variable Model

Hiroshi Mamitsuka

Institute for Chemical Research, Kyoto University
Gokasho Uji, 611-0011, Japan
mami@kuicr.kyoto-u.ac.jp

Abstract. We present a new method which combines a hierarchical stochastic latent variable model and a selective sampling strategy, for learning from co-occurrence events, i.e. a fundamental issue in intelligent data analysis. The hierarchical stochastic latent variable model we employ enables us to use existing background knowledge of observable co-occurrence events as a latent variable. The selective sampling strategy we use iterates selecting plausible non-noise examples from a given data set and running the learning of a component stochastic model alternately and then improves the predictive performance of a component model. Combining the model and the strategy is expected to be effective for enhancing the performance of learning from real-world co-occurrence events. We have empirically tested the performance of our method using a real data set of protein-protein interactions, a typical data set of co-occurrence events. The experimental results have shown that the presented methodology significantly outperformed an existing approach and other machine learning methods compared, and that the presented method is highly effective for unsupervised learning from co-occurrence events.

1 Introduction

In this paper, we focus on two-mode and co-occurrence events, such as product pairs in purchasing records and co-occurred words in texts. A data set of such co-occurrence events is one of the most frequently seen types of data sets in real world, and learning from co-occurrence events is a fundamental issue in intelligent data analysis. One of the most important property of co-occurrence events is that they are only positive examples, i.e. unsupervised examples in nature. An effective existing approach for learning from co-occurrence events is then based on model-based clustering [1]. In general, model-based clustering has a latent variable corresponding to a latent cluster and obtains one or more clusters of co-occurrence events by clustering co-occurrence events through estimating model parameters. Then, when two events are newly given, we can assign a latent cluster to the events with a likelihood, which can be computed by the estimated model parameters.

M.R. Berthold et al. (Eds.): IDA 2003, LNCS 2810, pp. 352–363, 2003.

We present a new methodology to improve the predictive ability of the existing model-based clustering approach for co-occurrence events. Our method combines a hierarchical latent variable model and a selective sampling strategy for unsupervised learning. The hierarchical latent variable model for co-occurrence events was presented in [2] and has two types of latent variables hierarchically. In general, an event has its own existing background knowledge, and in most cases, the background knowledge can be given as a set of classes, to one or more of which an event belongs. With the background knowledge, we can build a hierarchical latent variable model, in which one of the two types of latent variables and the other latent variable correspond to an existing class of events and a latent cluster of the classes, respectively. Estimating the parameters of the latent variable model results in performing clustering class pairs, instead of clustering event pairs, and we can obtain co-occurrence classes after estimating the parameters. Using the estimated model, arbitrary two events are predicted to be co-occurrence events if the two events belong to one or more pairs of two co-occurrence classes. The space complexity of the model is only roughly linear in the number of given events, and the parameters can be estimated by time-efficient EM (Expectation-Maximization) algorithm. We expect that the hierarchical model improves the predictive performance of existing approaches for co-occurrence events, since the model enables us to use existing classes of co-occurrence events.

We further focus on a selective sampling strategy for unsupervised learning and combine the hierarchical latent variable model and the strategy by employing the learning algorithm of the hierarchical model as the component subroutine of the strategy. The selective sampling was originally proposed for unsupervised noisy data in [3] and has been empirically shown that it is effective to improve the predictive performance of its component subroutine for noisy data. It repeats selecting plausible non-noise candidates by using previously obtained component models and running a component subroutine on the selected set of examples to obtain a new component model alternately. The final hypothesis is given by combining all obtained stochastic models, and the final likelihood for an example is obtained by averaging the likelihoods computed by all stochastic models. We can expect that the selective sampling also improves the predictive performance of existing approaches for real-world co-occurrence events, which is inevitably noisy.

The purpose of this paper is to empirically evaluate the effectiveness of the presented methodology using real data of co-occurrence events, comparing with an existing approach and other machine learning techniques, including a variety of simpler probabilistic models and support vector machines (SVMs). In our experiments, we focus on a real protein-protein interaction data set, a typical co-occurrence data set. We note that predicting unknown protein-protein interactions is an important and challenging problem in current molecular biology. We used the data set to perform five-fold cross-validation and generated synthetic negative examples for each of the five runs in the cross-validation to evaluate the ability of our method for discriminating positive from negative examples as the prediction accuracy of our method.

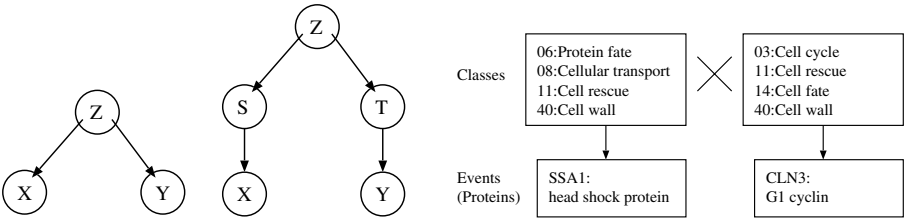

Fig. 1. Graphical models of (a) AM and (b) HAM, and (c) an example of co-occurrence events and their classes

Experimental results have shown that the presented method drastically improved the performance obtained by an existing approach and other methods tested. The component hierarchical model significantly outperformed other methods compared, and the selective sampling strategy further improved the prediction accuracy obtained by the component stochastic model. Over all, we have empirically shown that our method is highly effective for learning from real-world co-occurrence events, and we expect that our method is useful for predicting other numerous co-occurrence events found in a variety of application domains.

2 Methods

In this paper, we use the following notation. We denote a variable by a capitalized letter, e.g. X, and the value of a corresponding variable by that same letter in lower case, e.g. x. Now let X and Y be observable random variables taking on values $x_1, ..., x_L$ and $y_1, ..., y_M$, respectively, each of which corresponds to a discrete value (i.e. event). More concretely, x and y correspond to the first and the second event of an example of co-occurrence events. Let N be the total number of given examples (co-occurrence events). Let S and T be latent random variables taking on values $s_1, ..., s_U$ and $t_1, ..., t_V$, respectively, each of which corresponds to an existing latent class (background knowledge) of observable events. More concretely, for given co-occurrence events, s corresponds to a class taken by the first event and t corresponds to that by the second event. Let Z be a discrete-valued latent variable taking on values $z_1, ..., z_K$, each of which corresponds to a latent cluster. Let $n(x, y)$ be a binary value defined as follows: If x and y co-occur in a given data set then $n(x, y)$ is 1, otherwise it is 0. Let $v_{x,y}(s, t)$ be also a binary value defined as follows: If x and y belong to classes s and t respectively in given data then $v_{x,y}(s, t)$ is 1, otherwise it is 0.

In our experiments, we used protein-protein interactions as co-occurrence events. In this case, a protein corresponds to an event x (and y), and a protein class corresponds to an event class s (and t).

2.1 Hierarchical Latent Variable Model for Co-occurrence Events

Hierarchical Aspect Model (HAM). We first explain an existing model-based clustering approach for modeling co-occurrence events. The approach, which is called aspect model (AM for short), has been applied to some fields, including collaborative filtering recommender systems and semantic analysis in natural language processing [1]. An AM for X and Y with K clusters has the following form:

$$p(x, y; \theta) = \sum_{k}^{K} p(x|z_k; \theta)p(y|z_k; \theta)p(z_k; \theta).$$

The graphical model of AM is shown in Figure 1 (a).

In order to use existing classes of co-occurrence events, we extend the model to a hierarchical latent variable model for X and Y with S, T and K as follows:

$$p(x, y; \theta) = \sum_{l,m,k}^{U,V,K} p(x|s_l; \theta)p(y|t_m; \theta)p(s_l|z_k; \theta)p(t_m|z_k; \theta)p(z_k; \theta).$$

We call the model HAM, standing for *hierarchical aspect model* [2], and a graphical model of HAM is shown in Figure 1 (b).

Figure 1 (c) shows an example of co-occurrence events and their classes, which is derived from the real data of protein-protein interactions used in our experiments. In the figure, each of two events, i.e. proteins named as SSA1 and CLN3, belongs to a different set of four classes. We note that estimating the parameters of HAM results in clustering class pairs to obtain co-occurrence classes, instead of clustering event pairs as done in AM. Thus newly given two events will be predicted as co-occurrence events if the two events belong to co-occurrence classes. In addition, the larger the number of co-occurrence classes to which the two events belong, the more confidently the two events will be predicted as co-occurrence events. In the example shown in Figure 1 (c), we can consider 16 ($= 4 \times 4$) possible combinations of class pairs, and the proteins, SSA1 and CLN3, are predicted as co-occurrence events more strongly if a larger number of class pairs are co-occurrence classes. We emphasize that this model naturally combines co-occurrence events and their existing classes and is applicable to any co-occurrence events found in a variety of real-world applications.

Estimating Model Parameters. When the number of clusters K and training data D are given, a possible criterion for estimating probability parameters of HAM is the maximum likelihood (ML): $\theta^{ML} = arg\max_\theta \log p(D; \theta)$, where $\log p(D; \theta) = \sum_i \sum_j n(x_i, y_j) \log p(x_i, y_j; \theta)$.

We use a time-efficient general scheme, EM (Expectation-Maximization) algorithm [4], to obtain the ML estimators of HAM. The algorithm starts with initial parameter values and iterates both an expectation step (E-step) and a maximization step (M-step) alternately until a certain convergence criterion is

satisfied. The following equations derived by us are consistent with those of a hierarchical mixture model shown in [5].

In E-step, we calculate the latent class conditional probability, namely the probability that a given example (x, y) is generated by a cluster z and latent classes s and t given parameters θ:

$$w(z_k, s_l, t_m | x_i, y_j; \theta) = \frac{p(x_i|s_l;\theta)p(y_j|t_m;\theta)p(s_l|z_k;\theta)p(t_m|z_k;\theta)p(z_k;\theta)}{\sum_{k,l,m} p(x_i|s_l;\theta)p(y_j|t_m;\theta)p(s_l|z_k;\theta)p(t_m|z_k;\theta)p(z_k;\theta)}.$$

In M-step, we take a corresponding summation over $w(z_k, s_l, t_m|x_i, y_j; \theta)$ with $n(x_i, y_j)$:

$$\theta_{x_i|s_l} \propto \sum_{j,k,m} n(x_i, y_j)\ v_{x_i,y_j}(s_l, t_m)\ w(z_k, s_l, t_m|x_i, y_j; \theta_{old}).$$

$$\theta_{y_j|t_m} \propto \sum_{i,k,l} n(x_i, y_j)\ v_{x_i,y_j}(s_l, t_m)\ w(z_k, s_l, t_m|x_i, y_j; \theta_{old}).$$

$$\theta_{s_l|z_k} \propto \sum_{i,j,m} n(x_i, y_j)\ v_{x_i,y_j}(s_l, t_m)\ w(z_k, s_l, t_m|x_i, y_j; \theta_{old}).$$

$$\theta_{t_m|z_k} \propto \sum_{i,j,l} n(x_i, y_j)\ v_{x_i,y_j}(s_l, t_m)\ w(z_k, s_l, t_m|x_i, y_j; \theta_{old}).$$

$$\theta_{z_k} \propto \sum_{i,j,l,m} n(x_i, y_j)\ v_{x_i,y_j}(s_l, t_m)\ w(s_l, t_m|x_i, y_j; \theta_{old}).$$

The required memory size of the EM procedure depends only on the parameters of the model, i.e. $p(x|s), p(y|t), p(s|z), p(t|z)$ and $p(z)$. Note that K, i.e. the size of latent clusters, can be given as a certain constant, and the size of $p(s|z)$ and $p(t|z)$ is linear in the number of given events. Note further that the size of $p(x|s)$ and $p(y|t)$ is also practically linear in the number of events, since the number of classes to which an event belongs is limited to a certain small number. Overall, the total space complexity of HAM is roughly linear in the number of given events. In this sense, the HAM is a space-efficient model.

2.2 Selective Sampling with Hierarchical Aspect Models (SsHAM)

In order to further improve the predictive accuracy of HAM, we employ a selective sampling strategy for unsupervised learning [3]. It has been shown that the performance advantage of the selective sampling has more pronounced for noisier data sets.

The selective sampling uses a learning algorithm of an arbitrary stochastic model as a component subroutine, assuming that given an input x, the subroutine returns its likelihood $L(x)$. It further assumes that given a set of examples including x, x is regarded as an outlier (i.e. a noise) if $L(x)$ is the lowest among the likelihoods given to the example set. Under these assumptions, it repeats selecting plausible non-noise examples from a given whole set of examples and running a component learning algorithm on the selected examples alternately. In

Input: A set of given examples: E
Learning algorithm of hierarchical aspect model: A
Number of iterations: I
Number of examples selected at each iteration: N_s
Number of noise candidates to be removed at each iteration: N_n
Initialization: 0. Randomly choose N_s examples as E_0 from E
For $t = 0, ..., I-1$
1. Run A on E_t with random initial parameter values and obtain a trained model.
2. For each $(x, y) \in E$, compute its likelihood $L_t(x, y)$ using the trained model.
3. For each $(x, y) \in E$, compute average $\bar{L}(x, y)$ by $\frac{\sum_t L_t(x,y)}{t+1}$.
4. Select $\langle (x^*_1, y^*_1), ..., (x^*_{N_n}, y^*_{N_n}) \rangle$, whose $\bar{L}(x^*_1, y^*_1), ..., \bar{L}(x^*_{N_n}, y^*_{N_n})$ are the smallest, and remove them from E as $E'_{t+1} = remove(E, \langle (x^*_1, y^*_1), ..., (x^*_{N_n}, y^*_{N_n}) \rangle)$.
5. Randomly choose N_s examples from E'_{t+1} as E_{t+1}.

Output: Output all obtained component stochastic models.

Fig. 2. Algorithm: Selective sampling with hierarchical aspect models (SsHAM).

the example selection, the strategy first removes examples whose likelihoods are the lowest as noises (i.e. outliers) from all examples and then selects examples from the remaining examples randomly as a new set of training examples. The likelihood for a newly given test example is computed by averaging over the likelihoods computed by all models obtained as the final output. We combine the strategy and HAM by using the learning algorithm of HAM as a subroutine of the selective sampling strategy, and we call this combination 'SsHAM', standing for *selective sampling for hierarchical aspect models*. The pseudocode of SsHAM is shown in Figure 2.

2.3 Methods Compared in Our Experiments

In order to compare our method with other methods in our experiments, we generate a training data set using the data of co-occurrence events and a set of classes to which an event belongs. The data set is a table, in which each record corresponds to an example of co-occurrence events, i.e. a pair (x, y) that satisfies $n(x, y) = 1$, each attribute corresponds to a class pair and attribute values are all binary and represented by $v_{x,y}(s, t)$. The data set used in our experiments is a one-class data set, and we can run any unsupervised learning method on this set. We note that the complexity of the attribute size of the data set is quadratic in the size of event classes. Thus we cannot run a learning method whose computational complexity rises drastically with increasing the size of attributes. For example, learning Baysian belief networks in general falls into this type of unsupervised learning methods.

Simple Probabilistic Models. We first tested a simple method, which computes the probability that arbitrary two classes appear in given data. We consider

three variations in using the probabilities: summing over them (SSum), averaging over them (SAve) and selecting the maximum (SMax). The procedure of the method can be summarized as follows:

1. For all s and t, $p(s,t)$ is computed as follows:
$$p(s,t) = \frac{q(s,t)}{\sum_{s,t} q(s,t)},$$
where $q(s,t) = \sum_{i,j} n(x_i, y_j) \cdot v_{x_i,y_j}(s,t)$.
2. For any new events x and y, $\hat{p}(x,y)$, i.e. the probability that they are co-occurrence events, is calculated as follows:
$$\text{SSum:} \quad \hat{p}(x,y) = \sum_{x \in s, y \in t} p(s,t).$$
$$\text{SAve:} \quad \hat{p}(x,y) = \frac{\sum_{x \in s, y \in t} p(s,t)}{\sum_{x \in s, y \in t} 1}.$$
$$\text{SMax:} \quad \hat{p}(x,y) = \max_{x \in s, y \in t} p(s,t).$$

One-Class Support Vector Machine. We then tested 'one-class support vector machine (one-class SVM, hereafter OC.SVM)' proposed by [6]. Learning OC.SVM is unsupervised learning, and thus it is trained by positive examples only. In our experiments, we used LIBSVM ver.2.36, which has an option to run the OC.SVM and can be downloaded from http://www.csie.ntu.edu.tw/~cjlin/libsvm/. In prediction, for each example, LIBSVM gives a binary output, indicating whether the example belongs to the class of training data or not.

Support Vector Classifier. We finally tested a support vector classifier (hereafter, SVC), a supervised learning approach with significant success in numerous real-world applications. When we use the SVC in our experiments, we randomly generated negative training examples, which has the same size as that of positive training examples, not to overlap with any positive training examples and test examples. We then run both SVM^{light} [7] and the LIBSVM, and the two softwares outputted approximately the same performance results for all cases in our experiments.

For both OC.SVM and SVC, we used linear kernels in our experiments. We tested other types of kernels, such as a polynomial kernel, within the limits of main memory, but no significant improvements were obtained.

3 Experimental Results

3.1 Data

In our experiments, we focus on the data set of protein-protein interactions as a typical co-occurrence data set. We focus on yeast proteins, by which high-throughput bio-chemical experiments have been actively done these few years,

and used the MIPS [8] comprehensive yeast genome database, which contains the recent results of the experiments. From the MIPS data base, we obtained 3,918 proteins and 8,651 physical protein-protein interactions, each protein of which is one of the 3,918 proteins.

We further used the MIPS database to obtain a set of protein classes. We focus on the functional classification catalogue of the MIPS database. The information of the catalogue takes a tree structure, in which each node corresponds to a protein class and a protein falls into one or more nodes. To obtain a set of classes of proteins from the tree, the problem is how we cut the tree, using given training data. We employ the idea by [9] which cuts a thesaurus based on the MDL (Minimum Description Length) principle [10] to cut the tree for classifying proteins. By adjusting a parameter in the MDL, we obtained some sets of protein classes which range from 30 to 400 in their size. The 30 classes correspond to the nodes which have a root as their parent, and the set of 400 classes is the largest set under the condition that each class has one or more proteins.

3.2 Five-Fold Cross Validation

The evaluation was done by five-fold cross validation. That is, we split the data set into five blocks of roughly equal size, and in each trial four out of these five blocks were used as training data, and the last block was reserved for test data. The results were then averaged over the five runs.

To evaluate the predictive ability of each of the methods, we used a general manner for supervised learning. That is, for each test data set, we generated negative examples, the number of which is the same as that of positive examples, and examined whether each method can discriminate positive from negative examples. In this 50:50 setting, a prediction accuracy obtained by a random guessing (and a predictor, which outputs only one label) is maximally 50%, and we can check the performance of each of the methods by how it is better than 50%. We used two types of negative examples, both of which are randomly generated not to overlap with any positive example. The first type of examples are randomly generated from the 3,918 proteins. The second type of examples are randomly generated from a set of proteins, which are used in positive examples, i.e. the proteins contained in the 8,651 protein-protein interactions. Hereafter we call the two test data sets containing the first and second type of negative examples as Data1 and Data2, respectively.

In our experiments, we tested eight methods, i.e. SsHAM, HAM, SVC, OC.SVM, SSum, SAve, SMax and AM (aspect model), with varying the size of protein classes from 30 to 400, for Data1 and Data2. Throughout the experiments, we fixed $K = 100$ for AM and $K = 100$, $I = 10$, $N = N_n + N_s$ and $\frac{N_n}{N} = 0.1$ for SsHAM (and HAM). All our experiments were run on a Linux workstation with Intel Xeon 2.4GHz processor and 4G bytes of main memory. The actual computation time of HAM was within two minutes for all cases. Thus the computational burden of HAM is extremely small.

Table 1. Average prediction accuracies and t-values for Data1.

#classes	SsHAM	HAM	SVC	OC.SVM	SSum	SAve	SMax	AM
30	**79.5**	79.0	64.2	57.9	57.9	56.9	56.7	
		(1.76)	(**24.7**)	(**29.5**)	(**24.6**)	(**26.8**)	(**28.8**)	
100	**79.1**	78.3	67.0	59.3	59.8	58.4	58.1	
		(**6.72**)	(**23.1**)	(**26.8**)	(**22.0**)	(**29.8**)	(**24.9**)	
200	**78.5**	77.6	68.1	58.8	58.9	58.9	58.7	47.6
		(**5.39**)	(**19.0**)	(**21.6**)	(**18.8**)	(**21.5**)	(**18.8**)	(**41.0**)
300	**77.8**	76.6	68.7	61.3	63.2	64.6	64.2	
		(**10.8**)	(**20.7**)	(**22.9**)	(**16.8**)	(**22.0**)	(**14.0**)	
400	**77.5**	76.2	68.6	61.6	62.7	64.8	64.3	
		(**8.20**)	(**19.1**)	(**25.7**)	(**18.2**)	(**25.9**)	(**15.3**)	

Table 2. Average prediction accuracies and t-values for Data2.

#classes	SsHAM	HAM	SVC	OC.SVM	SSum	SAve	SMax	AM
30	67.0	**67.2**	59.7	52.6	53.2	53.0	52.7	
		(1.71)	(**23.7**)	(**47.0**)	(**39.2**)	(**68.5**)	(**85.5**)	
100	67.4	**67.5**	61.3	53.2	53.5	53.8	52.5	
		(0.14)	(**15.4**)	(**31.5**)	(**41.4**)	(**38.9**)	(**54.9**)	
200	**68.0**	67.6	62.6	52.8	53.3	55.2	53.1	49.4
		(1.64)	(**14.8**)	(**32.2**)	(**28.8**)	(**29.2**)	(**36.3**)	(**63.0**)
300	**66.6**	66.4	62.9	55.4	56.3	60.0	58.4	
		(0.84)	(**11.2**)	(**53.3**)	(**43.3**)	(**20.0**)	(**34.5**)	
400	**66.1**	65.8	62.6	55.3	55.4	59.1	58.4	
		(0.45)	(**11.6**)	(**34.3**)	(**45.4**)	(**19.3**)	(**24.2**)	

3.3 Average Prediction Accuracies

We first evaluated our method with other methods by average prediction accuracies obtained for Data1 and Data2. All of the methods except for SVC are trained by positive examples only, and predictions of all of methods are done for each of both positive and negative examples as a likelihood or a score. The prediction accuracy is obtained by first sorting examples according to their predicted likelihoods (or scores) and then computing an accuracy at a threshold which maximally discriminates positive from negative examples. The average prediction accuracy is obtained by averaging the accuracy over five runs in cross-validation.

We further used the 't' values of the (pairwise) mean difference significance test for statistically comparing the accuracy of SsHAM with that of another method. The t values are calculated using the following formula: $t = \frac{|ave(D)|}{\sqrt{\frac{var(D)}{n}}}$, where we let D denote the difference between the accuracies of the two methods for each data set in our five trials, $ave(W)$ the average of W, $var(W)$ the variance

of W, and n the number of data sets (five in our case). For $n = 5$, if t is greater than 4.604 then it is more than 99% statistically significant that SsHAM achieves a higher accuracy than the other.

Tables 1 and 2 show the average prediction accuracies and t-values (in parentheses) for SsHAM, HAM, SVC, OC.SVM, SSum, SAve, SMax and AM. As shown in the tables[1], SsHAM greatly outperformed the other methods, being statistically significant for all cases in both Data1 and Data2. For Data1, the maximum average prediction accuracy of SsHAM achieved roughly 80%, whereas that of other methods except HAM was less than 70%. We emphasize that the performance of SsHAM should be compared with other unsupervised learning approaches, because the accuracy of SVC may be improved as increasing the size of negative examples or changing the types of negative examples. Surprisingly, for Data1, the maximum average prediction accuracy of other unsupervised learning methods except HAM was lower than 65%, and for each case, the performance difference between SsHAM and one of other unsupervised learning approaches except HAM and AM reached approximately 15 to 20%. Moreover, the performance advantage of SsHAM over AM, an existing model-based clustering method, was roughly 30%. These results indicate that our methodology is highly effective for learning from co-occurrence events, unsupervised data in nature. This performance advantage of SsHAM over other methods was further confirmed for Data2. In particular, for Data2, the prediction accuracies obtained by other unsupervised methods except HAM was less than 60% except for one case, but the highest prediction accuracy of SsHAM reached 68%. This result also shows that our method is especially effective for learning from co-occurrence events.

We next check the predictive performance of HAM and the sampling strategy separately. For the two data sets, HAM significantly outperformed other methods compared, and SsHAM improved the performance of HAM in eight out of all ten cases, being statistically significant in four out of the eight cases. More exactly, SsHAM significantly outperformed HAM in four out of five cases for Data1, but for Data2, SsHAM outperformed HAM in only three cases and the performance difference between them was statistically insignificant in all cases. From this result, we can say that the advantage of the selective sampling strategy varies, depending on a given data set. Another possible explanation for this result is that the training data set used this time (particularly Data2) may not contain so many noises (outliers), because it has been shown that the selective sampling strategy is especially effective for noisy data.

3.4 ROC Curves

We then examined ROC (Receiver Operating Characteristic) curves obtained by each of the methods. An ROC curve is drawn by plotting 'sensitivity' against '1 - specificity'. The sensitivity is defined as the proportion of correctly predicted

[1] Approximate class sizes are shown. Exact sizes are 30, 88, 202, 297 and 398.

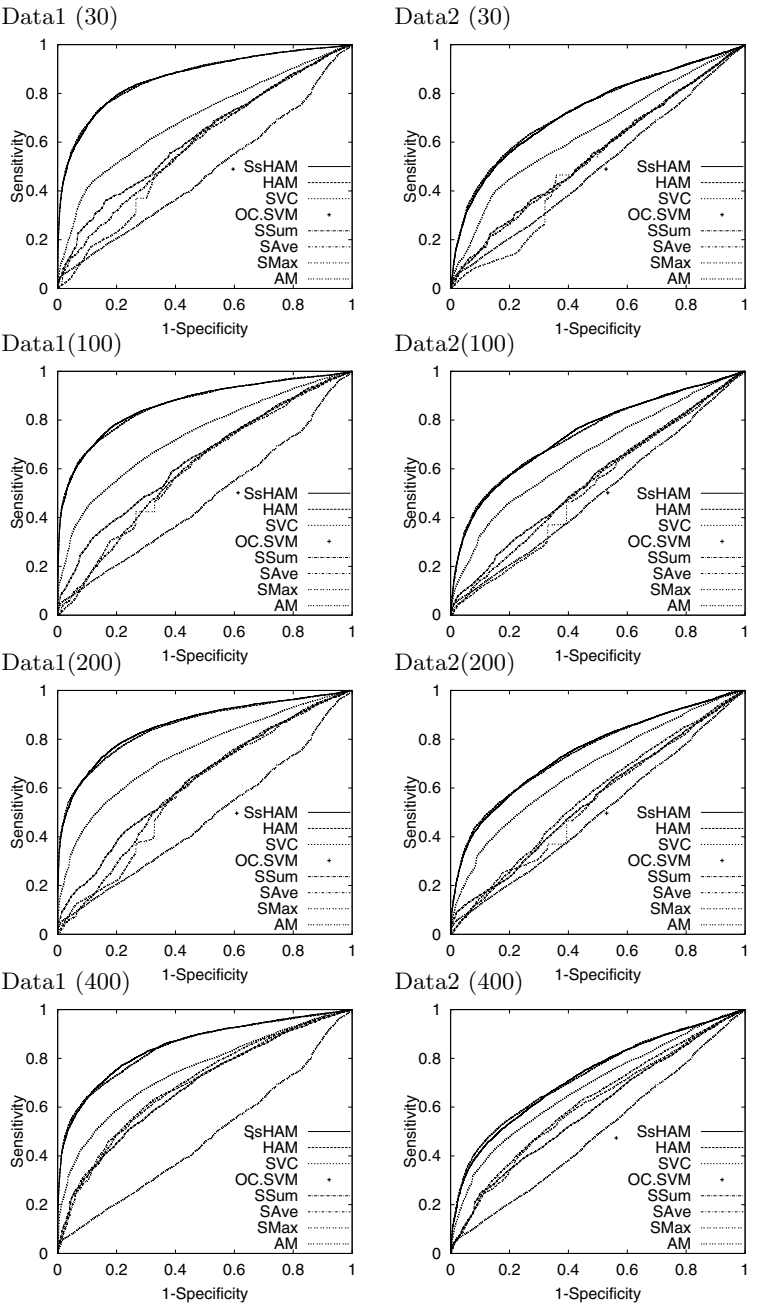

Fig. 3. ROC curves.

examples out of positive examples, and the specificity is defined as the proportion of correctly predicted examples out of negative examples.

Figure 3 shows the ROC curves obtained by all of methods tested for Data1 and Data2, for 30, 100, 200 and 400 protein classes (The sizes are shown in the parentheses.). The experimental findings obtained by the average prediction accuracies are confirmed from these curves.

4 Concluding Remarks

We have presented a new methodology which combines a selective sampling strategy and a hierarchical latent variable model, for the problem of modeling and predicting co-occurrence events. We have empirically shown that the presented method is highly effective for learning from co-occurrence events, using real data sets of protein-protein interactions, i.e. typical and real-world co-occurrence data sets. We believe that the presented method is successfully applicable to other numerous co-occurrence events in a variety of real-world applications as well as the protein-protein interactions used in our experiments.

References

1. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. Machine Learning **42** (2001) 177–196
2. Mamitsuka, H.: Hierarchical latent knowledge analysis for co-occurrence data. In: Proceedings of the Twentieth International Conference on Machine Learning, Morgan Kaufmann (2003)
3. Mamitsuka, H.: Efficient unsupervised mining from noisy data sets. In: Proceedings of the Third SIAM International Conference on Data Mining, SIAM (2003) 239–243
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B **39** (1977) 1–38
5. Bishop, C.M., Tipping, M.E.: A hierarchical latent variable model for data visualization. IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998) 281–293
6. Schölkopf, B., et al.: Estimating the support of a high-dimensional distribution. Neural Computation **13** (2001) 1443–1471
7. Joachims, T.: Making large-scale SVM learning practical. In Schölkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods – Support Vector Learning. MIT Press (1999)
8. Mewes, H.W., et al.: MIPS: A database for genomes and protein sequences. Nucleic Acids Research **30** (2002) 31–34
9. Li, H., Abe, N.: Generalizing case frames using a thesaurus and the MDL principle. Computational Linguistics **24** (1998) 217–244
10. Rissanen, J.: Modeling by shortest data description. Automatica **14** (1978) 465–471