Prediction of Chemical Toxicity by Network-based SVM on ES-cell Validation System

Wataru Fujibuchi[1] w.fujibuchi@aist.go.jp
Sachiyo Aburatani[1] s.aburatani@aist.go.jp
Junko Yamane[2] yamane-j@m.u-tokyo.ac.jp
Satoshi Imanishi[2] imanishi@m.u-tokyo.ac.jp
Hiromi Akanuma[3] akanuma.hiromi@nies.go.jp
Hideko Sone[3] hsone@nies.go.jp
Seiichiroh Ohsako[2] ohsako@m.u-tokyo.ac.jp

[1] Computational Biology Research Center, Advanced Industrial Science and Technology (AIST), 2-4-7 Aomi, Koto-ku, Tokyo 135-0064, Japan
[2] Center for Disease Biology and Integrative Medicine, The University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8654, Japan
[3] Research Center for Environmental Risk, National Institute for Environmental Studies, 16-2 Onogawa, Tsukuba 305-8506, Japan

Keywords: Bayesian network, kernel SVM, chemical toxicity, gene expression, stem cell

1 Introduction

Stem cell-based chemical informatics becomes increasingly important in biomedical fields, especially in drug discovery research. Here we describe our advanced bioinformatics technology to predict toxicities of chemicals including methylmercury (causes "Minamata" disease) and thalidomide (causes baby deformities) using the ES cell-based chemical validation system. In our analysis, we find that the Support Vector Machine (SVM)[1] prediction accuracy is increased by inferred gene networks when the weights of network edges are used as SVM features.

2 Method and Results

With RT-PCR gene expression data from ES cells exposed to 15 chemicals that are categorized into "neurotoxic", "tumor-genesis", and "others", we compared two SVM prediction methods: 1) gene expression data only 2) gene expression data + gene network edge weights. For RT-PCR analysis, we chose 9 (+1 internal standard) genes known as important to neuron development and performed RT-PCR experiments with 5 doses and 4 time points, yielding a total of 9x5x4=180 dimensions of data as SVM features. For network analysis, we start with the existing Bayesian network (BN) inference algorithm called TAO-Gen[2,3] and have improved it to the parallel (replica-exchange) version called RX-TAOgen, in order to obtain stable networks. A total of 9x9=81 gene-to-gene weights (β in Fig.1) on network edges are obtained and used as additional features for SVM. The whole scheme is shown in Fig. 1.

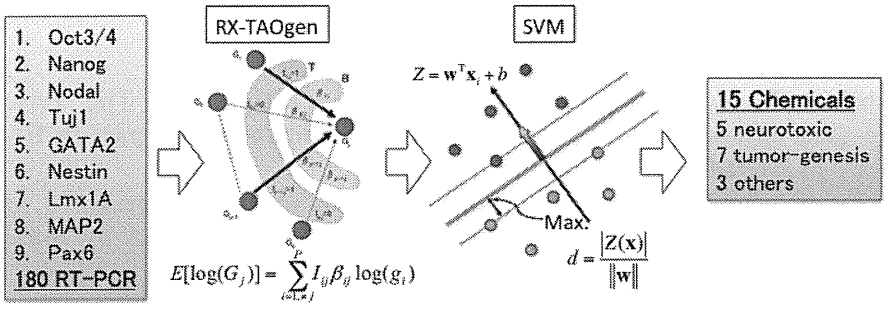

Fig.1 The whole scheme of toxic prediction using Bayesian networks as a part of input vectors of SVM.

2.1 Prediction Accuracy

For each of 15 chemicals, two RT-PCR experiments are performed, thus a total of 30 data are obtained. To evaluate the prediction accuracy, we performed a leave-two-out-prediction (LTOP) test, where two repeat data for each of 15 chemicals are used as test data at one cycle of prediction. In the SVM analysis, we tested three well-known kernels (linear, polynomial, and RBF) and our maximum entropy kernel[4] that is distance-based and the final kernel matrix is obtained by maximizing the kernel entropy. The accuracy is calculated by the ratio of the number of trues (true positives + true negatives) in all of the test data. The results are shown in Table 1.

Table 1: The number of data and accuracies for toxic chemical predictions.
(SVM: support vector machine, BN: Bayesian network)

Toxic category	Data (total of 30)	SVM	BN+SVM
Neurotoxic	10	90.0%	**93.3%**
Tumor-genesis	14	96.7%	**100.0%**
Others	6	–	–

2.2 Network Signatures

According to the results, the use of BN weights as additional features to SVM increases accuracies in both categories of toxic chemicals. The inferred BN structures are quite stable due to the high-tuned Gibbs-sampling method in RX-TAOgen; nevertheless, the obtained structures vary among chemicals. The network examples for 5 neurotoxic and 7 tumor-genesis chemicals are shown in Fig.2.

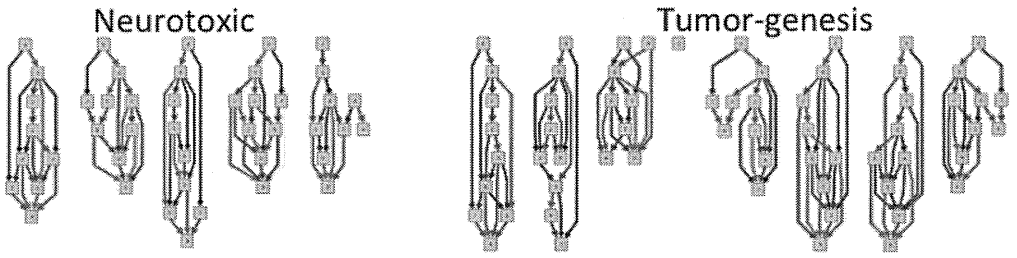

Fig.2 Examples of BN structures inferred from RT-PCR data for two chemical categories. Nine genes are indicated by rectangles and large weights are shown as edges.

3 Discussion

As the BN structures vary in Fig. 2 and it is difficult to recognize any distinct signatures in the same category by human eye, we performed a statistical comparison of the network weights of a pair of inferred networks. Interestingly, we observe that there are more correlations in weights among the same category than the different categories. For example, the average Pearson correlation coefficient of edge weights within 5 neurotoxic chemicals is 0.90, while that of 5 neurotoxic and 7 tumor-genesis chemicals is 0.86. Together, we conclude that the BN structures are weakly conserved in the toxic category, which may give additional information to SVM, resulted in the increase of accuracy.

References

[1] Vapnik, V., *The Nature of Statistical Learning Theory*, Springer, 1995.
[2] Yamanaka, T., Toyoshiba, H., Sone, H., Parham, F.M., and Portier C.J., The TAO-Gen algorithm for identifying gene interaction networks with application to SOS repair in E. coli., *Environ. Health. Perspect.*, 112:1614-21, 2004.
[3] Toyoshiba, H., *et al.*, Inference for Bayesian network via Gibbs sampling, Pre-print.
[4] Fujibuchi, W. and Kato, T., Classification of heterogeneous microarray data by maximum entropy kernel., BMC Bioinformatics, 8:267, 2007.

Regular Paper

Learning similarity functions for multi-platform gene expression data

Abstract

The existence of several technologies for measuring gene expression and the growing number of available large-scale gene expression microarrays motivate the need for cross-platform analysis tools. Cross-platform analysis of microarray data is an important problem, which heavily relies on the choice of a similarity function. For a classification task, a good similarity function should improve the prediction performance. It should also be easy to compute, and provide new biological insights of the data. However in practice, choosing a good similarity function for multi-platform microarray data is a difficult problem. In this work, our goal is to improve the performance of microarray search engines such as CellMontage. Therefore, we focus the ranking task rather than the classification task. Our ranking-based approach compares favourably to several similarity functions, including the Pearson and Spearman Correlation coefficients, the Euclidean distance, Linear Discriminant Analysis, and Neighbourhood Component Analysis. Experiments show that our method can be used to differentiate different types of cells with high accuracy, including induced pluripotent stem cells, embryonic stem cells, and cancer cells.

1. Introduction

The use of microarray data has become a common approach for studying various biological phenomena. When analyzing and comparing microarray samples, it is essential to accurately quantify to which extent two samples are similar. Therefore, the choice of good similarity functions, or equivalently good distance functions, is crucial. However, it is often difficult to choose a distance function relevant to the biological process of interest. In addition, microarray datasets are usually characterized by a small number of samples and a high number of genes, most of them being irrelevant to the biological process of interest. This introduces noise which may hinder the use of standard statistical methods. And last, the existence of different microarray technologies may introduce a bias in the gene expression measurements, making the comparison of samples from different platforms difficult.

Commonly used distance functions and similarity functions such as Euclidean distance, Pearson correlation coefficient, and Spearman's rank correlation coefficient do not provide answers to the previously mentioned problems. Instead, they consider that all genes are equally relevant and the resulting similarity scores are not specific to the biological process of interest. This motivates the need for automatically learning good distance metrics from the available data. Distance learning algorithms attempt to learn a distance function specific to a given prediction task, while discarding uninformative genes[1),2)]. Distance metrics can also be used for visualization and interpretation purposes, and therefore are also important for understanding the data. However, most of these approaches work in a classification setting, i.e. optimize a classification error[2)–6)]. In this work, our goal is to improve the ranking performance in multi-platform microarray databases such as CellMontage[7)]. The quality of such systems rely on their ability to correctly rank the samples rather than their ability to classify them as relevant or not (see e.g.[8),9)]). Therefore, we present a new distance learning approach which focuses on the ranking setting rather than the classification setting. Previous studies in multi-platform analysis have focused on

the unsupervised setting[10] and the classification setting[11]. However, to our knowledge, this work is the first study for ranking multi-platform microarray data. The rest of the paper is organized as follows. In section 2 we present our approach for distance learning in the ranking setting. We explain our model and the implementation's details. In section 3, we evaluate our approach on four microarray datasets. We explain our experimental protocol, present related methods, and analyze the results. We conclude and give suggestions to improve our approach in section 4.

2. A ranking-based distance learning model

2.1 Model

Pairwise ranking error

First, let $S = \{(x_i, y_i)\}_{i=1...n}$ be the database containing the microarray samples, where $x_i \in R^d$ is a gene expression vector and $y_i \in [1, \ldots, c]$ is a class label representing the cell type. We consider an information retrieval (IR) setting[12]. A user submits a query to the database. The goal is to present to the user a list of the database's samples, sorted by their similarity to the query: the most similar appear at the top of the list, and the less similar appear at the bottom.

One way to measure the quality of the system's answer is to compute a ranking error. Let's consider the sample $(x_i, y_i) \in S$ as the query. The user is only interested in other samples similar to the query, i.e. samples with the same class label y_i. Now let's consider two samples $(x_j, y_j) \in S$ and $(x_k, y_k) \in S$, such that x_i and x_j have the same cell type (i.e. $y_j = y_i$), and x_k has a different cell type (i.e. $y_k \neq y_i$). As x_i and x_j belong to same class, we expect x_i to be closer to x_j than it is to x_k. If we consider the Euclidean distance, this means that $||x_i - x_j||^2$ should be smaller than $||x_i - x_k||^2$. When it is not the case, there is a ranking error. We can now consider the following quantity:

$$[\![||x_i - x_j||^2 > ||x_i - x_k||^2]\!] = \begin{cases} 1 & \text{if } x_i \text{ is closer to } x_k \text{ than } x_j \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $[\![p]\!] = 1$ if the predicate p is true, 0 otherwise. In other words, this

quantity equals 1 when there is a ranking error, 0 otherwise. Considering (x_i, y_i) as the query, we can now compute the ranking error E over the whole dataset:

$$E = \sum_{(x_i,y_i)\in S} \frac{1}{\alpha_i} \sum_{\substack{j\in C_i \\ j\neq i}} \sum_{k\notin C_i} [\![||x_i - x_j||^2 > ||x_i - x_k||^2]\!]$$

where $C_i = \{j = 1 \dots n | y_j = y_i\}$ is the set of indices corresponding to class y_i and α_i is the number of pairs (j, k) such that x_i and x_j belongs to the same class while x_i and x_k belong to different classes: $\alpha_i = |\{(y_j, y_k) | j \in C_i, j \neq i, k \notin C_i\}|$. With such a normalization we have $E_i \in [0, 1]$, with $E_i = 0$ (resp. 1) if all the pairs are correctly (resp. incorrectly) ordered.

Learning a Mahalanobis distance

The previous ranking error E uses the standard Euclidean distance function for ranking the samples: $d(x_i, x_j)^2 = (x_i - x_j)^T(x_i - x_j) = ||x_i - x_j||^2$. However for a given ranking problem, this may not be appropriate and lead to a high ranking error. Therefore, our goal is to learn an appropriate distance function from the available data. To this end, we will consider the Mahalanobis distance: $d(x_i, x_j)^2 = (x_i - x_j)^T A^T A(x_i - x_j) = (Ax_i - Ax_j)^T(Ax_i - Ax_j) = ||Ax_i - Ax_j||^2$ parameterized by the matrix $A \in \mathbb{R}^{k\times d}$. It is clear that the Mahalanobis distance is equivalent to a linear projection with A, followed by the standard Euclidean distance. Replacing the Euclidean distance with the Mahalanobis distance in equation 2.1, the ranking error now writes:

$$E(A) = \sum_{(x_i,y_i)\in S} \frac{1}{\alpha_i} \sum_{\substack{j\in C_i \\ j\neq i}} \sum_{k\notin C_i} [\![||Ax_i - Ax_j||^2 > ||Ax_i - Ax_k||^2]\!]$$

Equation 2.1 contains boolean predicates which are not differentiable. Therefore the cost function $E(A)$ is difficult to minimize. As[13] suggested, we will use $[\![x < 0]\!] \leq \exp(-x)$, and consider the following upper bound:

$$F(A) = \sum_{(x_i,y_i)\in S} \frac{1}{\alpha_i} \sum_{\substack{j\in C_i \\ j\neq i}} \sum_{k\notin C_i} \exp\left(||Ax_i - Ax_j||^2 - ||Ax_i - Ax_k||^2\right)$$

The new cost function 2.1 is now differentiable, and is easier to minimize with simple gradient descent strategies. However, the computation of $F(A)$ is expensive, as it requires to consider triplets of examples (x_i, x_j, x_k). Since we

want to use our approach with large-scale datasets, a more efficient computation is required. Denoting $x_{ij} = x_i - x_j$, we can rewrite $F(A)$ as:

$$F(A) = \sum_{(x_i,y_i)\in S} \frac{1}{\alpha_i} \left(\sum_{\substack{j\in C_i \\ j\neq i}} \exp ||Ax_{ij}||^2 \right) \left(\sum_{k\notin C_i} \exp -||Ax_{ik}||^2 \right) \tag{2}$$

With this form, the computational complexity of $F(A)$ is now $O(n^2k + ndk)$. We will use standard gradient descent strategies to minimize F. Let's differentiate the cost function F with respect to the projection matrix A:

$$\frac{\partial F}{\partial A} = -2A \sum_{(x_i,y_i)\in S} \frac{1}{\alpha_i} \left(\left(\sum_{\substack{j\in C_i \\ j\neq i}} \exp ||Ax_{ij}||^2 \right) \left(\sum_{k\notin C_i} x_{ik}x_{ik}^T \exp -||Ax_{ik}||^2 \right) + \left(\sum_{\substack{j\in C_i \\ j\neq i}} x_{ij}x_{ij}^T \exp ||Ax_{ij}||^2 \right) \left(\sum_{k\notin C_i} \exp -||Ax_{ik}^2 \right) \right)$$

The computational complexity of the derivative $\partial F/\partial A$ is $O(n^2dk)$.

3. Experiments

3.1 Experimental protocol

Ranking error

Let S be the test set and T be the training set. In order to evaluate how good a distance function is for the ranking task, we compute the normalized ranking error on the test set. It is defined as follows:

$$E(\mathbf{A}) = \sum_{(\mathbf{x}_i,y_i)\in S} \frac{1}{\alpha_i} \sum_{\substack{j\in C_i \\ (\mathbf{x}_j,y_j)\in T}} \sum_{\substack{k\notin C_i \\ (\mathbf{x}_k,y_k)\in T}} [\![||\mathbf{Ax}_i - \mathbf{Ax}_j||^2 > ||\mathbf{Ax}_i - \mathbf{Ax}_k||^2]\!] * \frac{1}{0.5}$$

We divided the ranking error $[\![||\mathbf{Ax}_i - \mathbf{Ax}_j||^2 > ||\mathbf{Ax}_i - \mathbf{Ax}_k||^2]\!]$ by 0.5, which corresponds to the expected error of a random predictor. Hence, a perfect ranking algorithm will achieve $E = 0$ on the test set, while a random predictor will achieve $E = 1$. For all the following experiments, we performed a 20-fold cross-validation and computed the average ranking error on the test sets.

Pre-processing

In microarray datasets, the number of genes is typically much higher than the number of samples. This situation may lead to overfitting problems and hinder further analysis. By removing uninformative genes, feature selection methods have proven useful in this context[14]. In this work, we used the Pearson correlation coefficient to select the most important genes prior to distance learning[15].

3.2 Comparison to other methods

We compared our ranking-based approach to two other well known distance learning approaches: Linear Discriminant Analysis and Neighbourhood Components Analysis. Linear Discriminant Analysis (LDA) learns a linear projection such that the classes are well separated in the projected space (see e.g.[3]). The algorithm maximizes the variance between the classes while minimizing the variance within the classes. The computational complexity of LDA is dominated by the computation of the inverse matrix and the resolution of the eigenvalue problem, resulting in $O(D^3 + D^2N)$.

Neighbourhood Components Analysis (NCA)[1] learns a linear projection that optimizes a stochastic variant of the classification error in the nearest neighbour scheme. The computational complexity of NCA is $O(N^2K + NDK)$. In [15], experimental evaluation shows that NCA compares favourably to LDA and PCA with respect to the classification accuracy. In addition to LDA and NCA, we also used three standard distance functions: the Euclidean distance (ED), the Pearson correlation coefficient (PCC) and the Spearman's rank correlation coefficient (SRC)[16].

3.3 Results

In this section, we present ranking results on four datasets. We generated each dataset according to a biological task. With the first two datasets, we want to study the relationships between different cell types and identify the cell types across several platforms. With the third dataset, we study the relationship between cancer cells and ES cells and try to identify the cancer cells from the ES cells and brain cells. With the fourth dataset, we want to study iPS cells

and identify from which cell types they are derived.

Multiple cell types datasets

To study multiple cell types across several platforms, we generated two multi-platform microarray datasets as follows. First, we ranked the CellMontage*1 platforms according to the number of samples they contain. For dataset A, we selected the six largest platforms. Then, we selected the seven most represented cell types in these platforms: muscle, liver, kidney, brain, lung, skin, and intestine. This resulted in 2,778 samples, with 913 genes common to the six platforms. For dataset B, we selected the three largest platforms and the three most represented cell types in these platforms: muscle, liver, and kidney. This resulted in 1,064 samples, with 8,229 genes common to the three platforms. In both datasets, each cell type is considered as a class label. The ranking results are shown in table 1.

For dataset A, the best performance of ED is 0.47 with 1000 genes, that of PCC is 0.26 with 100 genes, and that of SRC is 0.83 with 10 genes. For dataset B, the best performance of ED is 0.56 with 1000 genes, that of PCC is 0.51 with 1000 genes, and that of SRC is 0.86 with 10 genes. We can see that PCC achieves the best prediction accuracy out of these three methods on both datasets, and seems to be a good method when combined with gene selection. However, the performance of the three methods highly depend on the number of selected genes. The reason is that ED, PCC, and SRC are unsupervised methods, and do not select genes specific to the ranking task. Therefore, it is difficult to select the optimal number of genes. On both datasets, NCA achieves better results for all values of the projection rank. The best performance of NCA is 0.13 with 2 and 10 dimensions on dataset A, and 0.29 with 2 dimensions on dataset B. The improvement of NCA over ED, PCC, and SRC was expected since it is supervised method, which uses the class labels to compute a good distance function. Finally, our ranking-based approach achieves the highest prediction accuracy on both datasets. The ranking error is 0.02 for

⋆1 http://cellmontage.cbrc.jp/

Gene filtering	Distance function	Projection rank	Dataset A	Dataset B
10	ED	-	0.81	0.72
10	PCC	-	0.68	0.68
10	SRC	-	0.83	0.86
100	ED	-	0.49	0.64
100	PCC	-	0.26	0.54
100	SRC	-	0.97	0.95
1000	ED	-	0.47	0.56
1000	PCC	-	0.35	0.51
1000	SRC	-	0.98	0.97
-	NCA	2	0.13	0.29
-	NCA	5	0.14	0.36
-	NCA	10	0.13	0.37
-	Ranking	2	**0.02**	0.07
-	Ranking	5	**0.02**	**0.05**
-	Ranking	10	**0.02**	0.06

Table 1 Ranking results between cell types on two different datasets.

all values of the rank projection on the dataset A, while the best ranking error is 0.05 with 5 dimensions. These results suggest that there is a compromise for choosing the optimal rank projection. If the rank is too small, the resulting projection is too constrained and can not achieve optimal accuracy. However, if the rank projection is too high, the model may overfit the training data and perform poorly on the test data. Overall, our approach achieves a high-quality identification of cell types in a multi-platform context.

Cancer cells dataset

Next, we want to study the difference between cancer cells, ES cells and brain cells. We selected a platform containing these three cell types, resulting in 140 samples and 24,353 genes. Each of the three cell types is considered as a class label. Because of the high number of genes, we used gene selection as pre-processing before using distance learning methods. The ranking results are shown in table 2.

Gene filtering	Distance function	Projection rank	Ranking error
-	ED	-	0.57
-	PCC	-	0.56
-	SRC	-	1.0
10	ED	-	0.61
10	LDA	1	0.34
10	NCA	1	0.04
10	Ranking	1	**0.01**
1000	ED	-	0.42
1000	LDA	1	0.46
1000	NCA	1	0.06
1000	Ranking	1	**0.02**

Table 2 Ranking results between cancer cells, ES cells, and brain cells.

Without pre-processing, ED and PCC achieve similar performance, while the performance of SRC is close to random. Again, these three methods can't offer any performance guarantee, since they ignore the class labels. In contrast, LDA, NCA and our ranking-based approach are supervised methods and achieve better prediction accuracies. LDA achieves 0.34 with 10 genes, while NCA achieve 0.04 with 10 genes. Even though both methods optimize a classification error, NCA achieves a high prediction accuracy. This suggests that the optimal distance functions for ranking and for classification are similar. Nevertheless, direct optimization of the ranking error can further improve the result, as our ranking-based approach achieves the best ranking performance, 0.01 with 10 genes. This suggests that these 10 genes seem to contain enough information to identify cancer cells, ES cells and brain cells. However, those 10 genes can't be used directly, as the poor performance of ED shows. This experiment shows that our approach automatically computes the best linear combination of genes to achieve the highest prediction performance.

Induced plutipotent stem cells dataset

With this experiment, we want to study the differences between iPS cells of

Gene filtering	Distance function	Projection rank	Ranking error
-	ED	-	0.70
-	PCC	-	0.68
-	SRC	-	0.93
-	NCA	1	0.58
-	Ranking	1	**0.06**
100	ED	-	0.74
100	PCC	-	0.63
100	SRC	-	0.75
100	LDA	1	0.35
100	NCA	1	0.41
100	Ranking	1	**0.14**

Table 3 Ranking results between types of different iPS cells.

different origins. We selected 12 platforms containing samples composed of iPS cells. The iPS cells are derived from various cell types, including foreskin fibroblasts, neural cells, embryonic cells, and mesenchymal cells. This resulted in 73 samples and 16,319 genes. When generating the dataset, we considered the foreskin fibroblasts iPS cells as one class, and all the other iPS cells as another class. Therefore, our goal was to predict if an iPS cell is derived from foreskin fibroblasts or from another cell type. The ranking results are shown in table 3.

When we consider all the genes, ED and PCC achieve similar results, while the performance of SRC is close to random prediction. LDA could not be used due to the high number of genes. NCA achieves better results (ranking error of 0.58) by making use of the class labels. Our ranking-based approach achieves the best results with 0.06. When we select the 100 most informative, the performance of SRC improves and becomes comparable to that of ED and PCC. LDA and NCA, which both optimize a classification error, achieve good prediction performance with ranking errors of respectively 0.35 and 0.41. Once again, our ranking-based approach achieves the lowest ranking error with 0.14. Interestingly, the use of all the genes achieves better performance than the use of the 100 selected

genes. This shows the weakness of gene selection methods such as Pearson correlation coefficient which considers each gene separately. It ignores genes that are uninformative when used alone but become informative when used in combination to other genes. This underlines the advantage of our method, which considers all the genes simultaneously and computes a combination optimal for the ranking task.

4. Conclusion

In this work, we presented a ranking-based approach to automatically compute good distance functions (or equivalently, good similarity measures) for multi-platform microarray data. First we defined a ranking error between vectors, and considered a linear model, i.e. a Mahalanobis distance. Then we proposed our distance learning algorithm to optimize this ranking error. In order to analyze large scale multi-platform datasets, we also proposed an efficient implementation of our algorithm. Then we evaluated our approach on four microarray datasets.

We compared our approach with three standard distance functions as well as two distance learning algorithms: Euclidean distance, Pearson correlation coefficient, Spearman's rank correlation coefficient, Linear Discriminant Analysis and Neighbourhood Components Analysis. The results showed that ED, PCC, and SRC show poor prediction performance, independently of the preprocessing. As we explained previously, these methods ignore the class labels. Therefore, they are not appropriate for the ranking task. LDA and NCA show better prediction performance than ED, PCC, and SRC. This improvement was expected, as LDA and NCA are supervised methods. However, they optimize a classification error and therefore, are not optimal for the ranking task. In addition, LDA is computationally and memory intensive when the number of genes is high, which makes it difficult to use with large datasets. Our ranking-based approach achieves the best ranking errors on the four datasets and for all preprocessing strategies. This shows that our approach can be used to differentiate different types of cells with high accuracy, including induced pluripotent stem

cells, embryonic stem cells, and cancer cells. In addition, our approach is simple to implement, efficient, and can be used for better understanding of the dataset.

In our future works, we want to extend our approach in different ways. First, our approach is limited by its linear nature. We wish to extend it to non-linear distance learning using the kernel trick[4]. The second limitation is related to the cost function we optimized: we defined a ranking error which uniformly penalizes all the pairs incorrectly ordered for a given list. With this ranking error, a wrong prediction of the highest ranked samples has the same cost as a wrong prediction of the lowest ranked samples. However, when querying a database, we are mainly interested in the results in the top of the list, and the quality prediction in the bottom of the list is less important. In our future works we wish to optimize other ranking errors, which focus on the ranking accuracy of higher ranked samples (e.g. the normalized discounted cumulative gain)[17]. Last, we want to use our approach to discover the roles of each gene for a given cell type. One possible way to do this is to interpret the genes weights of the projection matrix as importance scores for a given phenotype.

References

1) Goldberger, J., Roweis, S., Hinton, G. and Salakhutdinov, R.: Neighbourhood components analysis, *Advances in Neural Information Processing Systems 17*, MIT Press, pp.513–520 (2004).

2) Weinberger, K.Q. and Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification, *Advances in Neural Information Processing Systems*, pp.1473–1480 (2006).

3) Duda, R.O., Hart, P.E. and Stork, D.G.: *Pattern Classification*, Wiley-Interscience Publication (2000).

4) Globerson, A. and Roweis, S.: Metric learning by collapsing classes, *Advances in Neural Information Processing Systems*, Vol.18, p.451 (2006).

5) Shalev-Shwartz, S., Singer, Y. and Ng, A.Y.: Online and batch learning of pseudo-metrics, *ICML '04: Proceedings of the twenty-first international conference on machine learning*, New York, NY, USA, ACM, p.94 (2004).

6) Sugiyama, M.: Local Fisher discriminant analysis for supervised dimensionality reduction, *ICML '06: Proceedings of the 23rd international conference on machine learning*, New York, NY, USA, ACM, pp.905–912 (2006).

7) Fujibuchi, W., Kiseleva, L., Taniguchi, T., Harada, H. and Horton, P.: CellMontage: similar expression profile search server, *Bioinformatics*, Vol. 23, pp.3103–4 (2007).

8) Freund, Y., Iyer, R., Schapire, R.E. and Singer, Y.: An efficient boosting algorithm for combining preferences, *Journal of Machine Learning Research*, Vol.4, pp.933–969 (2003).

9) Har-peled, S., Roth, D. and Zimak, D.: Constraint classification for multiclass classification and ranking, *Proceedings of the 16th Annual Conference on Neural Information Processing Systems, NIPS-02*, MIT Press, pp.785–792 (2003).

10) Tamayo, P., Scanfeld, D., Ebert, B., Gillette, M., Roberts, C. and Mesirov, J.: Metagene projection for cross-platform, cross-species characterization of global transcriptional states, *PNAS*, Vol.104, pp.5959–5964 (2007).

11) Warnat, P., Eils, R. and Brors, B.: Cross-platform analysis of cancer microarray data improves gene expression based classification of phenotypes, *BMC Bioinformatics*, Vol.6 (2005).

12) Manning, C.D., Raghavan, P. and Schütze, H.: *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA (2008).

13) Freund, Y. and Schapire, R.E.: Experiments with a New Boosting Algorithm (1996).

14) Guyon, I. and Elisseeff, A.: An introduction to variable and feature selection, *J. Mach. Learn. Res.*, Vol.3, pp.1157–1182 (2003).

15) Weston, J., Elisseeff, A., Schölkopf, B. and Tipping, M.: Use of the zero norm with linear models and kernel methods, *J. Mach. Learn. Res.*, Vol.3, pp.1439–1461 (2003).

16) Mount, D. W.: *Bioinformatics: Sequence and Genome Analysis*, Cold Spring Harbor Laboratory (2004).

17) Burges, C.J., Ragno, R. and Le, Q.V.: Learning to Rank with Nonsmooth Cost Functions, *Advances in Neural Information Processing Systems 19*

(Schölkopf, B., Platt, J. and Hoffman, T., eds.), MIT Press, Cambridge, MA, pp.193–200 (2007).

A ranking-based alternative to the discriminant analysis framework

Jean-François Pessiot Wataru Fujibuchi

Computational Biology Research Center (CBRC), Advanced Industrial Science and Technology (AIST), Tokyo 135-0064, JAPAN

1 Introduction

In statistics and machine learning, discriminant methods are used when two or more classes need to be characterized or separated from each other. One of the most popular method is the Linear Discriminant Analysis (LDA), also known as Fischer Discriminant Analysis. The LDA is a feature extraction method which identifies a linear combination of features that maximizes the variance between the classes while minimizing the variance within the classes. The LDA has been successfully used to identify important features of a dataset, or as a pre-processing step prior to the use of classification methods [5].

Approaches such as LDA implicitely take place in a classification setting, i.e. the new feature space should help improve the classification accuracy. However, ranking the data vectors is sometimes more important than predicting the labels themselves [3]. In the information retrieval (IR) setting, for example, a user submits a query to an IR system. The IR system should then return a list of documents ranked with respect to their relevance to the query [9]. The most relevant documents should appear at the top of the list and the less relevant ones should appear at the bottom of the list. Online search engines, such as Google [1], are typical examples of this setting. The quality of such systems relies on their ability to correctly rank the web pages rather than their ability to classify them as relevant or not. This problem has attracted a lot of attention in the last years (see e.g. [1,8]).

In this work, our goal is to compute a feature space in which the Euclidean distance induces few ranking errors. We illustrate this idea in figure , which shows two different feature spaces used to represent the query and the documents. In each feature space, we use the Euclidean distance to classify and rank the documents. The documents close to the query are classified as "relevant," while those far from the query are classified as "irrelevant." The classification decision function is shown by dashed lines and the classification errors are shown by purple arrows. The ranking errors are shown by gray curves. We can see that with the feature space $\mathcal{F}$, the IR system makes two classification errors and five ranking errors. With the feature space $\mathcal{G}$, the IR system makes two classification errors and only one ranking error. Therefore, these two feature spaces are equivalent with respect to classification accuracy, but have different performance with respect to ranking accuracy. This example underscores the need to learn a feature space specifically designed to improve the ranking error.

2 A ranking-based model

Our goal is to learn an appropriate feature space from the available data, i.e. a feature space which leads to a small ranking error. Let's consider a set of examples $S = \{(\mathbf{x}_i, y_i)\}_{i=1\ldots N}$, where each example is composed of a vector $\mathbf{x}_i \in \mathbb{R}^D$ and a class label $y_i \in [1, \ldots, C]$, and D is the number of dimensions of the initial feature space. We define a ranking error as the quantity:

$$[\![\|\mathbf{x}_i - \mathbf{x}_j\|^2 > \|\mathbf{x}_i - \mathbf{x}_k\|^2]\!] = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ is closer to } \mathbf{x}_k \text{ than } \mathbf{x}_j \\ 0 & \text{otherwise} \end{cases}$$

[1] http://www.google.com

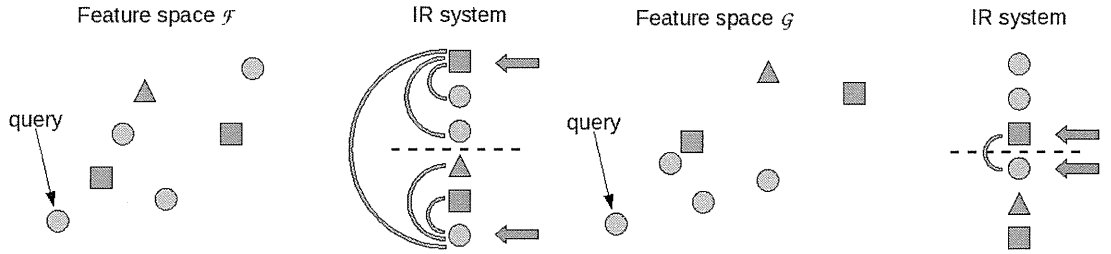

Figure 1: The query and the documents are represented in two different feature spaces. Each color represents a document class. The classification errors are shown by purple arrows, while the ranking errors are shown by gray curves. Left: with the feature space $\mathcal{F}$, the IR system makes two classification errors and five ranking errors. Right: with the feature space $\mathcal{G}$, the IR system makes two classification errors and one ranking error

where $[\![p]\!] = 1$ if the predicate p is true, 0 otherwise. Now let's consider the projection matrix $\mathbf{A} \in \mathbb{R}^{K \times D}$, where $K \leq D$ is the number of dimensions of the new feature space. The total ranking error induced by the projection matrix A writes:

$$E(\mathbf{A}) = \sum_{(\mathbf{x}_i, y_i) \in S} \frac{1}{\alpha_i} \sum_{\substack{j \in C_i \\ j \neq i}} \sum_{k \notin C_i} [\![\|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j\|^2 > \|\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_k\|^2]\!] \tag{1}$$

where $C_i = \{j = 1 \dots N | y_j = y_i\}$ is the set of indices corresponding to class y_i, and α_i is the number of pairs (j, k) such that $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class while $\mathbf{x}_i$ and $\mathbf{x}_k$ belong to different classes: $\alpha_i = |\{(y_j, y_k) | j \in C_i, j \neq i, k \notin C_i\}|$. A good projection matrix should induce a low ranking error. Therefore, our goal is to minimize the ranking error $E(\mathbf{A})$. We implemented two gradient descent based algorithms to minimize $E(\mathbf{A})$, called PARCA and Logistic PARCA.

3 Experiments

We evaluated our approach using two standard text datasets in the text mining literature: 20Newsgroups and Reuters[2]. For both datasets, we removed non alphanumerical words as well as common words using a stop-words list. We also removed non-informative words using information gain and kept only the top 1,000 words as the final vocabulary (see e.g. [10]). Text data were then represented using the classical bag-of-words representation, where each document is a vector of word counts over the vocabulary space [9]. The 20 Newsgroups dataset contains text messages posted on 20 Usenet newsgroups. After pre-processing, the total number of documents is 18,595 and each class contains between 448 and 1,000 documents. The Reuters dataset contains articles from the Reuters newswire, classified into different categories. We removed categories containing less than 100 documents. After pre-processing, the dataset contains 12,887 documents and 13 categories. Each category contains between 105 and 6,386 documents.

We compared two variants of our ranking-based method (PARCA and Logistic PARCA) against various other methods, including Euclidean distance, Linear Discriminant Analysis, Neighbourhood Components Analysis (NCA) [7], Information-theoretic metric learning (ITML) [4], and Principal Component Analysis (PCA) [5]. In figure , we show the ranking performance on the 20 Newsgroups and Reuters datasets. Due to the size of these datasets, we only considered training ratios between 2% and 10%. For both datasets and for all training ratios, the initial Euclidean distance performs better than PCA. The poor performance of PCA stems from its unsupervised nature. Because PCA ignores the class labels, the first two principal components are not relevant for the ranking task.

Despite being a supervised method, LDA shows poor performance. For both datasets, LDA performs worse than the Euclidean distance for most values of the training ratio. This suggests that LDA has

[2]http://www.cs.technion.ac.il/~ronb/thesis.html

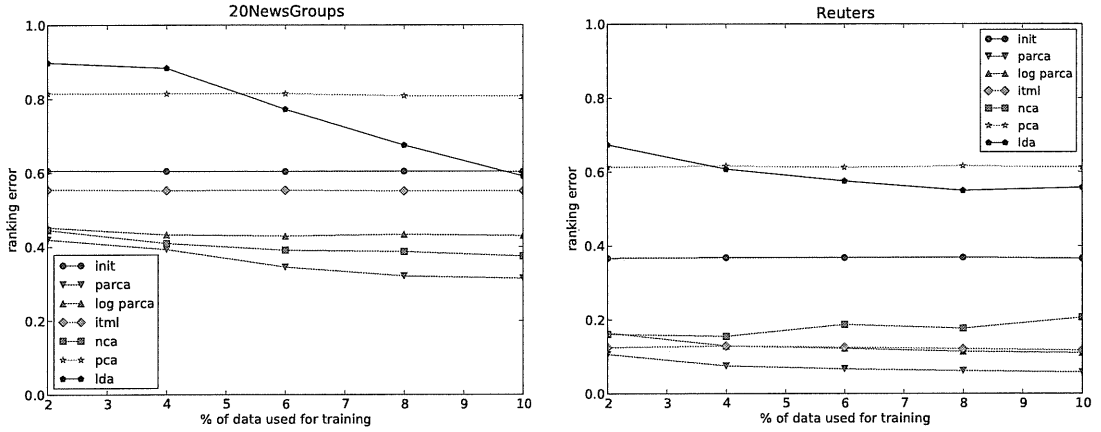

Figure 2: Ranking performance on two text datasets: 20 Newsgroups, Reuters and WebKB

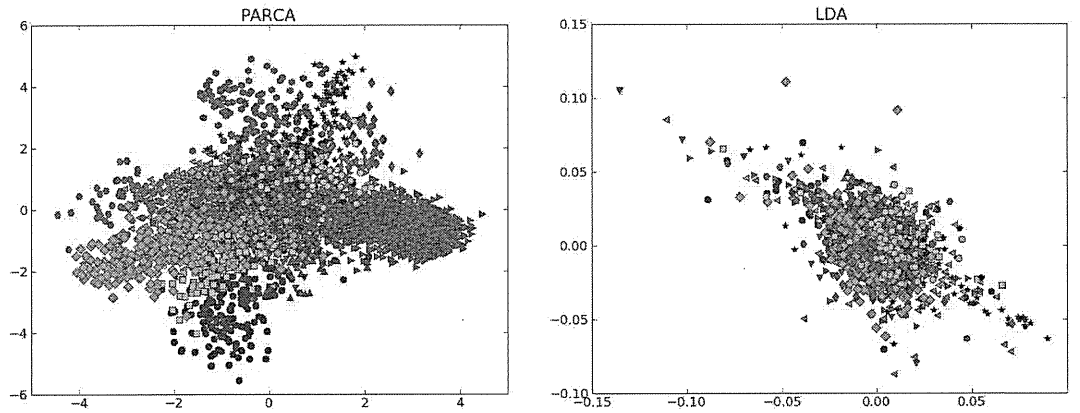

Figure 3: Visualization results on the Reuters test set, for 2% of training data. Each colored symbol represents one of the 13 classes

a severe tendency to overfit when few training data are available. Hence, it seems that when only few training data are available, LDA should not be used for the ranking task. On 20 Newsgroups, ITML performs only slightly better than the Euclidean distance. However, it shows good ranking accuracy on the Reuters dataset.

PARCA achieves the best ranking error on both datasets for all the training ratios, while NCA and Logistic PARCA show similar performance. PARCA always performs slightly better than Logistic PARCA. The performance of NCA improves as more data become available for training, except on the Reuters dataset. This suggests that the classification error optimized by NCA is sometimes irrelevant for ranking. The performance of PARCA always improves as more data become available for training. As we noted previously, PARCA only needs a few training data to achieve a low ranking error.

In figure , we show the visualization results on the Reuters test sets for 2% of training data. We can see that PARCA is the best method at separating the classes, while LDA shows severe overfitting problems and tends to collapse the classes together.