

Gaussian Logic and Its Applications in Bioinformatics

Ondřej Kuželka
Czech Technical University in
Prague, Faculty of Electrical
Engineering
Technická 2, 16627
Prague, Czech Republic
kuzelon2@fel.cvut.cz

Andrea Szabóová
Czech Technical University in
Prague, Faculty of Electrical
Engineering
Technická 2, 16627
Prague, Czech Republic
szaboand@fel.cvut.cz

Filip Železný
Czech Technical University in
Prague, Faculty of Electrical
Engineering
Technická 2, 16627
Prague, Czech Republic
zelezny@fel.cvut.cz

ABSTRACT

We describe a novel statistical relational learning framework capable to work efficiently with combinations of relational and numerical data which is especially valuable in bioinformatics applications. We show how this model can be applied to modelling of gene expression data and to problems from proteomics.

Categories and Subject Descriptors

I.2.6 [Artificial Intelligence]: Learning; J.3 [Life and Medical Sciences]: Biology and genetics

1. INTRODUCTION

Modelling of relational domains which contain substantial part of information in the form of real valued variables is an important problem with applications in bioinformatics. In this paper we describe a relatively simple framework for learning in rich relational domains containing numerical data. The system relies on multivariate normal distribution and exploits regularities in covariance matrices for construction of models capable to deal with variable number of numerical random variables.

2. GAUSSIAN LOGIC

Let $n \in \mathbb{N}$. If $\vec{v} \in \mathbb{R}^n$ then v_i ($1 \leq i \leq n$) denotes the i -th component of \vec{v} . If $I \subseteq [1; n]$ then $\vec{v}_I = (v_{i_1}, v_{i_2}, \dots, v_{i_{|I|}})$ where $i_j \in I$ ($1 \leq j \leq |I|$). To describe training examples as well as learned models, we use a conventional first-order logic language \mathcal{L} whose alphabet contains a distinguished set of constants $\{r_1, r_2, \dots, r_n\}$ and variables $\{R_1, R_2, \dots, R_m\}$ ($n, m \in \mathbb{N}$). An r -substitution ϑ is any substitution as long as it maps variables (other than) R_i only to terms (other than) r_j . For the largest k such that $\{R_1/r_{i_1}, R_2/r_{i_2}, \dots, R_k/r_{i_k}\} \subseteq \vartheta$ we denote $I(\vartheta) = (i_1, i_2, \dots, i_k)$. A (Herbrand) interpretation is a set of ground atoms of \mathcal{L} . $I(H)$ ($I(\varphi)$) denotes the naturally ordered set of indexes of all constants r_i found in an interpretation H (\mathcal{L} -formula φ).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM-BCB '11, August 1-3, Chicago, IL, USA
Copyright 2011 ACM 978-1-4503-0796-3/11/08... \$10.00

Our learning examples have both *structure* and *real parameters*. An example may e.g. describe a measurement of the expression of several genes; here the structure would describe functional relations between the genes and the parameters would describe their measured expressions. The structure will be described by an interpretation, in which the constants r_i represent uninstantiated real parameters. The parameter values will be determined by a real vector. Formally, an example is a pair $(H, \vec{\theta})$ where H is an interpretation, $\vec{\theta} \in \Omega_H$, and $\Omega_H = \mathbb{R}^{I(H)}$. The pair $(H, \vec{\theta})$ may also be viewed as a non-Herbrand interpretation of \mathcal{L} , which is the same as H except for including R in its domain and assigning θ_i to r_i .

Examples are assumed to be sampled from the distribution $P(H, \Omega_H) = \int_{\Omega_H} f_H(\vec{\theta}|H) P(H) d\vec{\theta}$ which we want to learn. Here, $P(H)$ is a discrete probability distribution on the countable set of Herbrand interpretations of \mathcal{L} . If \mathcal{L} has functions other than constants, we assume that $P(H)$ is non-zero only for finite H . $f_H(\vec{\theta}|H)$ are the conditional densities of the parameter values. The advantage of this definition is that it cleanly splits the possible-world probability into the discrete part $P(H)$ which can be modeled by state-of-the-art approaches such as Markov Logic Networks (MLN's) [3], and the continuous conditional densities $f_H(\vec{\theta}|H)$ which we elaborate here. In particular, we assume that $f(\vec{\theta}|H) = N(\vec{\mu}_H, \Sigma_H)$, i.e., $\vec{\theta}$ is normally distributed with mean vector $\vec{\mu}_H$ and covariance matrix Σ_H . The indexes H emphasize the dependence of the two parameters on the particular Herbrand interpretation that is parameterized by $\vec{\theta}$.

We explore a method, in which parameters determining $P(H, \Omega_H)$ can be estimated using the entire training set. The type of $P(H, \Omega_H)$ is obviously not known; note that it is generally not a Gaussian mixture since the $\vec{\theta}$ in the normal densities $f_H(\vec{\theta}|H)$ have, in general, different dimensions for different H . However, our strategy is to learn *Gaussian features* of the training set. A Gaussian feature (*feature*, for short) is a \mathcal{L} -formula φ , which for each example $(H, \vec{\theta})$ extracts some components of $\vec{\theta}$ into a vector $\vec{u}(\varphi)$, such that $\vec{u}(\varphi)$ is approximately normally distributed across the training sample. For each feature φ , $\vec{\mu}_{\vec{u}(\varphi)}$ and $\Sigma_{\vec{u}(\varphi)}$ are then

estimated from the entire training sample. A set of such learned features φ can be thought of as a constraint-based model determining an approximation to $P(H, \Omega_H)$. We define *Gaussian features* more precisely in a moment after we introduce *sample sets*.

Given an example $e = (H, \vec{\theta})$ and a feature φ , the *sample set* of φ and e is the multi-set $\mathcal{S}(\varphi, e) = \{\vec{\theta}_{I(\varphi)} \mid H \models \varphi\}$

where ϑ are r-substitutions grounding all free variables in φ , and $H \models \varphi\vartheta$ denotes that $\varphi\vartheta$ is true under H .

Now we can formally define *Gaussian features*. Let φ be a \mathcal{L} -formula, $\{e_i\}$ be a set of examples drawn independently from a given distribution and let $\vec{\theta}_i$ be vectors, each drawn randomly from $\mathcal{S}(\varphi, e_i)$. We say that φ is a *Gaussian feature* if $\vec{\theta}_i$ is multivariate-normally distributed.

Given a non-empty sample set $\mathcal{S}(\varphi, e)$, we define the *mean vector* as $\mu(\varphi, e) = \frac{1}{|\mathcal{S}(\varphi, e)|} \sum_{\vec{\theta} \in \mathcal{S}(\varphi, e)} \vec{\theta}$ and the Σ -matrix as $\Sigma(\varphi, e) = \frac{1}{|\mathcal{S}(\varphi, e)|} \sum_{\vec{\theta} \in \mathcal{S}(\varphi, e)} (\vec{\theta} - \mu(\varphi, e)) (\vec{\theta} - \mu(\varphi, e))^T$.

Finally, using the above, we define estimates over the entire training set as $\hat{\mu}_\varphi = \frac{1}{m} \sum_{i=1}^m \mu(\varphi, e_i)$ and $\hat{\Sigma}_\varphi = \frac{1}{m} \sum_{i=1}^m (\Sigma(\varphi, e_i) + \mu(\varphi, e_i) \mu(\varphi, e_i)^T) - \hat{\mu}_\varphi \hat{\mu}_\varphi^T$. It can be shown that these estimates are consistent and unbiased (asymptotically unbiased, respectively) despite the fact that samples in individual sample sets are not independent [6].

Importantly, using the training-set-wide estimates, we can derive estimates of parameters $\vec{\mu}_{H_n}$ and Σ_{H_n} of the densities $f_{H_n}(\vec{\theta} \mid H_n)$ for a relational structure H , even if H does not occur in the training set.

In general, the problem of estimating the mean $\mu(\varphi, e_i)$ is an NP-hard problem (it subsumes the well-known NP-complete problem of θ -subsumption). However, it is tractable for a class of features, *conjunctive tree-like features* [5] for which we devised efficient (polynomial-time) algorithms. Intuitively, a tree-like conjunction can be imagined as a tree with the exception that whereas trees are graphs, conjunctions correspond in general to hypergraphs. Computation of μ -vectors and Σ -matrices of tree-like conjunctive features can be done in time polynomial in the combined size of the feature and the respective example.

Finally, let us briefly describe methods for constructing a set of features that *give rise* to models capable to appropriately model a given set of examples. These methods are specialized for working with tree-like features because estimation of the parameters is tractable for them. The feature construction algorithm for tree-like features is based on an algorithm from [5]. It shares most of the favourable properties of the original algorithm like detection of redundant features¹.

3. EXPERIMENTS

We start by describing experiments with gene expression data. We wanted to find out whether it is possible to improve covariance matrix estimation on gene sets using fea-

¹More details can be found in the technical report [6].

GL Train	Target	w/l against BL	GL	BL
GDS1975	GDS1975	694/306	-459	-488
GDS1220	GDS1220	827/173	-302	-630
GDS1375	GDS1375	445/555	-334	-409
GDS1975	GDS1220	807/193	-303	-630
GDS1220	GDS1375	423/577	-336	-409
GDS1375	GDS1975	740/260	-459	-488

Table 1: Experimental Results: The first column displays the training datasets used for Gaussian logic, the second column displays the target datasets on which 10-fold cross-validation is performed, the third column displays the number of wins and losses against the baseline method [7] (BL). The fourth and fifth column display average log-likelihoods (per fold) on unseen target data (estimated by 10-fold cross-validation) for the two methods.

tures learned on completely different sets of genes from possibly different datasets (e.g. from different tissues or corresponding to different diseases). To this end we utilized the algorithms presented in previous sections for construction and selection of a set of Gaussian features. We, however, did not use directly the models (mean vectors and covariance matrices) created according to these features for modelling of the data but we used them as *shrinkage targets* for covariance-matrix estimation [7].

Before we get to the details of our experimental protocol, we give here a brief description of KEGG pathways. Each KEGG pathway is a description of some biological process (a metabolic reaction, a signalling process etc.). It contains a set of genes annotated by relational description which contains relations among genes such as *compound*, *phosphorylation*, *activation*, *expression*, *repression* etc. The relations do not necessarily refer to the processes involving the genes per se but they may refer to relations among the products of these genes. For example, the relation *phosphorylation* between two genes A, B is used to indicate that a protein coded by the gene A adds phosphate group(s) to a protein coded by the gene B . This is what makes the task of modelling gene expression (i.e. regulation networks of gene activities) using only this kind of relational knowledge a rather challenging task.

We performed the experiments with three datasets obtained from GEO (Gene Expression Omnibus [4]). We normalized the data so, in fact, we worked with correlation matrices rather than with general covariance matrices. We selected 50 smallest pathways from each gene-expression dataset and, on half of them, we constructed a set of tree-like features and estimated their parameters. Then we greedily selected a subset of these features on the other half of the 50 pathways. We performed the experiments constructing the Gaussian features on one dataset and then applying them on *different* pathways from another dataset and also to *different* pathways from the same dataset. As a baseline method we used the shrinkage-based method from [7]. The experimental results are displayed in Table 1. Our novel method based on shrinkage with covariance-matrix targets given by Gaussian features performs best. It is also fair to say that the most important feature (which contributed most to the accuracy) was a feature that added an average correlation for all pairs of genes. Nevertheless, the other discovered more complex

features (e.g. $g(A, R_1)$, $cmpd(A, B)$, $cmpd(B, C)$, $g(C, R_2)$ or $g(A, R_1)$, $inhib(A, B)$, $expr(B, C)$, $g(C, R_2)$) further contributed to the accuracy as well.

The second set of experiments dealt with estimation of DNA-binding propensity of proteins. Proteins which possess the ability to bind to DNA play a vital role in the biological processing of genetic information like DNA transcription, replication, maintenance and the regulation of gene expression. The process of DNA-binding has not been completely understood yet. It has been shown that electrostatic properties of proteins such as total charge, dipole moment and quadrupole moment or properties of charged patches located on proteins' surfaces are good features for predictive classification (e.g. [1], [2]). Szilágyi and Skolnick [8] created a logistic regression classifier based on 10 selected features, including electrostatic properties, to predict DNA-binding propensity of proteins from their low-resolution structures. We will use this method of Szilágyi et al. as a method for comparisons.

We used Gaussian logic to create a model for distributions of positively charged amino acids. We split each protein into consecutive non-overlapping *windows*, each containing l_w amino acids. For each window of a protein P we computed the value a_i^+/l_w where a_i^+ is the number of positively charged amino-acids in the window i . Then for each protein P we constructed an example $e_P = (H_P, \vec{\theta}_P)$ where

$$H_P = w(1, r_1), next(1, 2), \dots, next(n_P - 1, n_P), w(n_P, r_P)$$

$$\vec{\theta}_P = (a_1^+/l_w, a_2^+/l_w, \dots, a_{n_P}^+/l_w)$$

In the experiments that we performed, we constructed only one feature $F_{non} = w(A, R_1)$ for non-DNA-binding proteins since we do not expect this class of proteins to be very homogeneous. For DNA-binding proteins, we constructed a more complex model by selecting a set of features using a greedy search algorithm. The greedy search algorithm optimized classification error on training data. Classification was performed by comparing, for a tested protein, the likelihood-ratio of the two models (DNA-binding and non-DNA-binding) with a threshold selected on the training data. We estimated the accuracy of this method using 10-fold cross-validation (always learning parameters and structure of the models and selecting the threshold and window length l_w using only the data from training folds) on a dataset containing 138 DNA-binding proteins (PD138 [8]) and 110 non-DNA-binding proteins (NB110 [1]). The estimated accuracies (*Gaussian Logic*) are shown in Table 2.

Method	Accuracy [%]
Szilágyi et al.	81.4
Baseline Gaussian logic	78.7
Gaussian logic	81.9

Table 2: Accuracies estimated by 10-fold cross-validation on PD138/NB110.

The Gaussian-logic method performs only slightly better than the method of Szilagyi et al. [8] but uses much less information. Next, we were interested in the question whether

the machinery of Gaussian logic actually helped improve the predictive accuracy in our experiments or whether we could obtain the same or better results using only the very simple feature $F = w(A, R_1)$ also to model the DNA-binding proteins. The results corresponding to this latter simplified setting are also shown in Table 2 (*Baseline Gaussian Logic*) and indicate that Gaussian logic was really helpful in this case.

4. CONCLUSIONS AND FUTURE WORK

In this paper we have introduced a novel relational learning system capable to work efficiently with combinations of relational and numerical data. The experiments gave us some very promising results. Furthermore, there are other possible applications of Gaussian logic in bioinformatics which were not discussed in this paper. For example, finding patterns that generally correspond to highly correlated sets of genes may have applications in predictive classification of gene expression data.

Acknowledgement: O.K. was supported by the Czech Grant Agency through project 103/11/2170. A.S. was supported by project ME10047 granted by the Czech Ministry of Education. F.Ž. was supported by the Czech Grant Agency through project 201/09/1665.

5. REFERENCES

- [1] S. Ahmad and A. Sarai. Moment-based prediction of dna-binding proteins. *Journal of Molecular Biology*, 341(1):65 – 71, 2004.
- [2] N. Bhardwaj, R. E. Langlois, G. Zhao, and H. Lu. Kernel-based machine learning protocol for predicting DNA-binding proteins. *Nucleic Acids Research*, 33(20):6486–6493.
- [3] P. Domingos, S. Kok, D. Lowd, H. Poon, M. Richardson, and P. Singla. Probabilistic inductive logic programming. chapter Markov logic, pages 92–117. Springer-Verlag, 2008.
- [4] R. Edgar, M. Domrachev, and A. Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic Acids Research*, 1, 2002.
- [5] O. Kuželka and F. Železný. Block-wise construction of tree-like relational features with monotone reducibility and redundancy. *Machine Learning*, 83:163–192, 2011.
- [6] O. Kuželka and F. Železný. Gaussian logic for bioinformatics, 2011, *Available online: http://ida.felk.cvut.cz/users/kuzelka/gl_bio.pdf*.
- [7] J. Schäffer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 2005.
- [8] A. Szilágyi and J. Skolnick. Efficient prediction of nucleic acid binding function from low-resolution protein structures. *Journal of Molecular Biology*, 358(3):922 – 933, 2006.