Learning Monadic and Dyadic Relations: Three Case Studies in Systems Biology

Michiel Stock[1], Tapio Pahikkala[2], Antti Airola[2], Tapio Salakoski[2], and Bernard De Baets[1], Willem Waegeman[2]

[1] Department of Mathematical Modelling, Statistics and Bioinformatics, Ghent University, Coupure links 653, B-9000 Ghent, Belgium
firstname.surname@UGent.be
[2] University of Turku and Turku Centre for Computer Science, Joukahaisenkatu 3-5 B, FIN-20520, Turku, Finland
firstname.surname@utu.fi

Abstract. In molecular biology and other subfields of biology one can encounter many machine learning problems where the goal consists of predicting relations or interactions between pairs of objects. In this article we elaborate on three applications that represent such a learning scenario: predicting functional relationships between enzymes in bioinformatics, predicting protein-ligand interactions in computational drug design and predicting heterotroph-methanotroph interactions in microbial ecology. All three case studies are analyzed using an extension of a general kernel-based framework that we proposed recently. From a mathematical perspective, we both consider monadic and dyadic relations, and we use Kronecker product feature mappings to couple feature representations of paired objects, which correspond to vertices in a graph.

1 Introduction

Relational data can be observed with many faces in systems biology: interactions between proteins, regulation between genes, binding of chemical compounds, etc. Usually such data cannot be appropriately analyzed with traditional methods for vectorial data. In this article we discuss the challenges that occur when trying to predict relations between objects from feature representations of those objects. To make this more precise, let us first introduce the three case studies that we will consider:

– As a first application we consider the problem of ranking a database of enzymes according to their catalytic similarity to a query protein. This catalytic similarity, which serves as the relation of interest, represents the relationship between enzymes w.r.t. their biological function. For newly discovered enzymes, this catalytic similarity is usually not known, so one can think of trying to predict it using machine learning algorithms and kernels that describe the structure-based or sequence-based similarity between enzymes.

- In a second application we also use a protein query, not against a database of proteins but for finding ligands that can bind the protein with high affinity. Extracting relevant features for these two types of molecules, this framework could serve as a post-processing method for docking algorithms, greatly aiding drug design. From a technical standpoint, this problem differs from the previous one. Relations are here dyadic instead of monadic, since two different types of objects are considered.
- A third application originates from the field of microbial ecology, where it is known that bacteria form complex ecological networks, exchanging metabolites, nutrients and information. More specifically, the interaction between methanotrophic bacteria and heterotrophic bacteria forms a key interest for biologists. The question we are interested in is how different kinds of each group influence each others growth. In wet lab experiments different combinations of both groups are incubated and their mutual growth density is measured. We construct a model that ranks the bacteria of one group for their predicted cooperation with our reference bacteria from the other functional group. As features we use a metabolic and phylogenetic representation of the different organisms. This model could give environmental technologists a powerful tool for synthetic ecology.

From a machine learning perspective, the above three applications have a lot in common, and the learning scenario can be formally summarized as follows. Given a dataset of known relations between pairs of objects and a feature representation of these objects in terms of variables that might characterize the relations, the goal usually consists of inferring a statistical model that takes two objects as input and predicts whether the relation of interest occurs for these two objects. Moreover, since one aims to discover unknown relations, a good learning algorithm should be able to construct a predictive model that can generalize towards unseen data, i.e., pairs of objects for which at least one of the two objects was not used to construct the model. As a result of the transition from predictive models for single objects to pairs of objects, new advanced learning algorithms need to be developed, resulting in new challenges with regard to model construction, computational tractability and model assessment.

As relations between objects can be observed in many different forms, this general problem setting provides links to several subfields of machine learning, like statistical relational learning [1], graph mining [2], metric learning [3] and preference learning [4]. More specifically, from a graph-theoretic perspective, learning a relation can be formulated as learning edges in a graph where the nodes represent information of the data objects; from a metric learning perspective, the relation that we aim to learn should satisfy some well-defined properties like positive definiteness, transitivity or the triangle inequality; and from a preference learning perspective, the relation expresses a (degree of) preference in a pairwise comparison of data objects.

Our methods are based on generating a joint feature representation of pairs of objects by using the Kronecker product pairwise kernel (KPPK). In the most simple case, methods such as regularized least squares can then be used to make

predictions on pairs of objects. It can be shown that the KPPK can learn any arbitrary relation [5], though it is not guaranteed to be the most optimal kernel. By using kernels we can deal with complex data structures such as sequences, graphs and trees, which are frequently encountered in systems biology.

When the ranking error is optimized, the above framework can be used for conditional ranking. Here a set of objects are ranked, conditioned on another object, the query. This means we are now treating the problem as a kind of information retrieval setting where we want to find the most relevant objects. We explore several performance measures such as the ranking error, mean average precision, ROC and CROC curves, to find the most meaningful evaluation criterion for a particular problem.

Despite clear connections with (fuzzy) logic and statistical relational learning, our framework differs substantially differs from existing logical or relational learning papers. In our setting we assume that there is an underlying relation such as *similarity of enzymes* or *binding affinity of molecules* defined between the data points. The values for this relation are observed between some, or in some cases, between all the training data points. The relation to be learned is based directly on this underlying relation. In contrast, existing frameworks such as [6–11] typically consider relational and logical information as additional information, rather than as the direct source of the labels. That is, the relational graph as such does not define the rankings over the training data points, but simply provides us with additional information about the relationships between the data points.

2 A general framework for learning relations

2.1 Using pairs of objects as instances

Relational learning is equivalent to inferring labels on edges between vertices of a graph. When considering relations between two objects one can define monadic and dyadic relations. The former only deals with objects of the same 'type', for example inferring interactions between proteins, as depicted in Figure 1. The dyadic setting on the other hand treats prediction about a pair of objects of a different type, for example whether a small molecule can bind a protein. This setting can be viewed in Figure 2. Let $Q(v, u)$ denote a binary relation on an object space $\mathcal{V} \times \mathcal{U}$, then one can distinguish the following basic settings [5]:

- Crisp relations, when $Q : \mathcal{V} \times \mathcal{U} \to \{0, 1\}$, this corresponds to a binary classification with the pair of objects as input.
- real-valued relations when the relation is of the form $Q : \mathcal{V} \times \mathcal{U} \to \mathbb{R}$, thus resulting in a regression type of learning setting.
- Ordinal-valued relations: when a certain order can be derived, but the actual values or differences between the values do not matter.
- Monadic relations: in this case we have the restriction $\mathcal{V} = \mathcal{U}$.

Assuming the data is structured as a bipartite graph $G = (\mathcal{V}, \mathcal{U}, \mathcal{E}, Q)$ where $\mathcal{V}$ and $\mathcal{U}$ correspond to two different sets of nodes v and u, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{U}$ represents

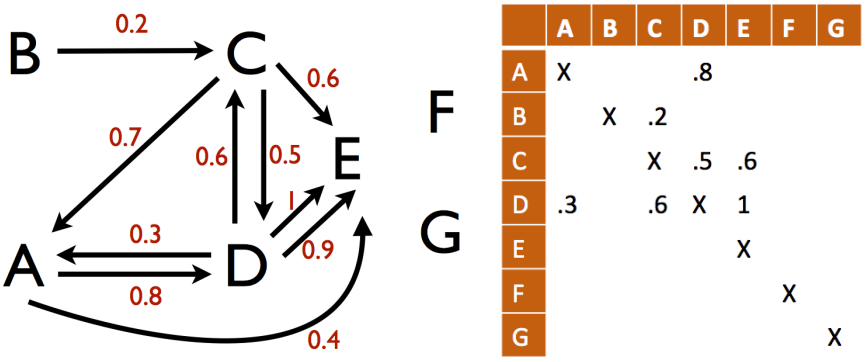

Fig. 1. Relational learning presented as learning the labels on a graph (in this case a real value between 0 and 1). Since the machine learning framework is based on features of the individual objects, a generalization can be made to make predictions of pairs where one (for example $\{A, F\}$) or both (for example $\{F, G\}$) of the objects do not appear in the training set. After [5].

the set of edges e, for which training labels are provided in terms of relations. These relations are represented by training weights y_e on the edges, generated by the unknown underlying relation $Q : \mathcal{V} \times \mathcal{U} \rightarrow [0, 1]$. In areas like logic, relational calculus and fuzzy logic, these relations are confined to the interval $[0, 1]$, in order to analyze relational properties such as transitivity. However, it is always possible to make an extension to real-valued relations $h : \mathcal{V} \times \mathcal{U} \rightarrow \mathbb{R}$ by using an increasing mapping $\sigma : \mathbb{R} \rightarrow [0, 1]$ such that

$$Q(v, u) = \sigma(h(v, u)), \quad \forall (v, u) \in \mathcal{V} \times \mathcal{U}. \tag{1}$$

The learning problem is formulated as the selection of a suitable function $h \in \mathcal{H}$, with $\mathcal{H}$ a certain hypothesis space, in particular a reproducing kernel Hilbert space (RKHS). Thus the hypothesis $h : \mathcal{V} \times \mathcal{U} \rightarrow \mathbb{R}$ is denoted as

$$h(e) = \mathbf{w}^T \Phi(e), \tag{2}$$

with $\mathbf{w}$ a vector of parameters to be estimated from the training data and Φ a joint feature mapping for edges in the graph.

To derive a relevant kernel for pairs of objects, a feature mapping based on the Kronecker product is proposed to express pairwise interactions between features of nodes:

$$\Phi(e) = \Phi(v, u) = \phi(v) \otimes \phi(u), \tag{3}$$

where ϕ denotes the feature mapping of the individual nodes. This feature mapping is shown[3] to correspond to the the Kronecker product pairwise kernel [13]

$$K^{\Phi}_{\otimes}(e, \bar{e}) = K^{\Phi}_{\otimes}(v, u, \bar{v}, \bar{u}) = K^{\phi}(v, \bar{v}) K^{\phi}(u, \bar{u}), \tag{4}$$

[3] Remember the mixed-product property for the Kronecker product: $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$

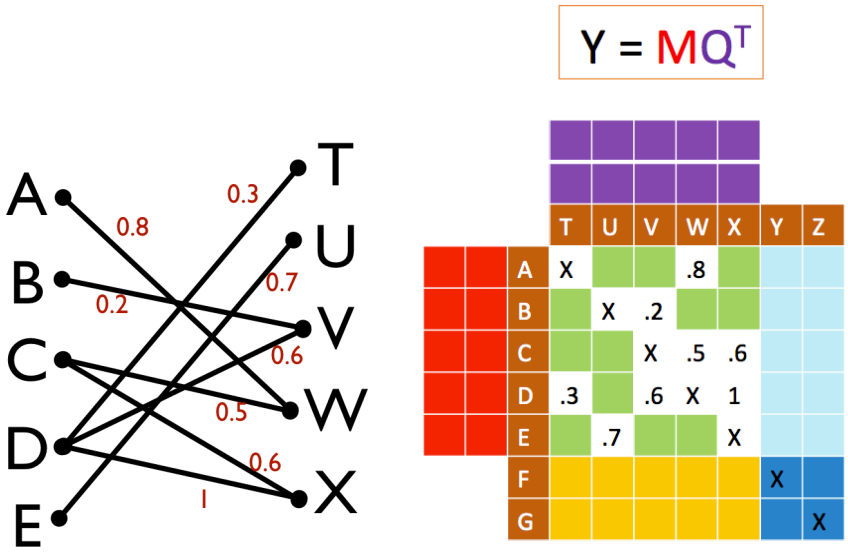

Fig. 2. Relational learning of dyadic data, presented as learning the labels on a graph (in this case a real value between 0 and 1). Features of the individual objects are used to make predictions about the labels. In addition, latent features can be obtained by means of matrix factorization methods. This is shown with the matrices $\boldsymbol{M}$ and $\boldsymbol{Q}$, purple and red, constituting a factorization of the original label matrix $\boldsymbol{Y}$ [12].

with K^{ϕ} the kernel corresponding to ϕ or, stated differently, the node kernel.

This is a very beautiful and efficient way to generalize kernel methods to deal with pairs of objects. By simply taking the Kronecker product of the kernel or kernels for the nodes one obtains a kernel representing the edges that can simply be plugged in your favorite kernel algorithm to learn the label of the edges. [5] prove that the Kronecker product pairwise kernel is indeed universal and can be used to learn arbitrary relations.

Theorem *Let us assume that the spaces $\mathcal{V}$ and $\mathcal{U}$ are compact metric spaces. If a continuous kernel K^{ϕ} is universal on $\mathcal{V}$ and $\mathcal{U}$, then $K^{\Phi}_{\otimes}$ defines a universal kernel on $\mathcal{E}$.*

This is by no means equivalent to saying that this kernel is the most optimal kernel for learning relations. The theorem only states that the Kronecker product pairwise kernel can be used to approximate any relation, given enough suitable data to train the algorithm. However, this does not exclude the possibility of obtaining performance gains by considering other pairwise kernels. Several of such pairwise kernels have been introduced in bioinformatics, including the metric learning pairwise kernel [14] and the Cartesian kernel [15]. We refer to [5] for an in-depth discussion of the differences between such kernels.

2.2 Conditional ranking

Let us further elaborate on the second case study in the introduction to introduce conditional ranking algorithms. Suppose a company wants to design a new drug to cure a certain disease. They have a set of promising organic molecules, some of them being potentially suitable as a drug. From experiments a target protein is found on which the drug should bind and act as an inhibitor. To minimize side effects, there is also a set of proteins that are similar in function, but not involved in the disease for which the compound should preferably have a low affinity. Given a dataset of known protein ligand interactions, it is possible, using the methods described above, to construct a model to predict a value quantifying binding of a ligand to a protein. This model could be used to search for the best compounds in the database, so that only these compounds need to be investigated during clinical trials. Though this may sound very reasonable, one can see this may not be the best approach for this problem. The model constructed is used to predict a binding value, but this is not something one is directly interested in. The researcher wants an ordered list of the best or worst binding compounds for a given protein. The model is optimized to predict an accurate value (usually by minimizing the squared residuals), not to be able to perform an optimal ranking. Thus the appropriate solution for this problem would be to construct a model that can rank a database of molecules according to their binding properties conditioned on a target protein.

Given the relations $Q(v, u)$ and $Q(v, u')$ we compose the ranking of u and u' conditioned on v as:

$$u \succeq_v u' \Leftrightarrow Q(v, u) \geq Q(v, u'). \tag{5}$$

To obtain a model that can correctly rank the nodes, it is desirable to use a loss function that punishes error in the ranking, the ranking loss

$$\mathcal{L}(h, T) = \sum_{v \in V} \sum_{e, \bar{e} \in E_v : y_e < y_{\bar{e}}} I(h(e) - h(\bar{e})), \tag{6}$$

with $I(x)$ the Heaviside function, returning one when its argument is positive, zero when its argument is negative en $1/2$ when its argument is zero. E_v denotes the set of all edges starting from, or the set of all edges ending at the node v, depending on the specific task. Since Equation 6 is neither convex nor differentiable, we use an approximation of the ranking loss

$$\mathcal{L}(h, T) = \sum_{v \in V} \sum_{e, \bar{e} \in E_v} (y_e - y_{\bar{e}} - h(e) + h(\bar{e}))^2. \tag{7}$$

Thus the mean difference between conditional ranking and a regression-based framework is the use of a ranking-based loss function. This framework was described in [16]. It can be implemented efficiently and it easily scales to millions of edges, using a closed-form analytic shortcut.

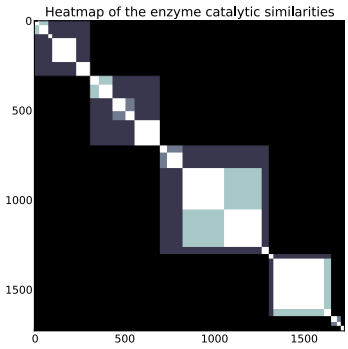

Fig. 3. Heatmap of the catalytic similarity between the enzymes in the analyzed dataset. Rows and columns are ordered according to the main EC classes.

3 Experimental results

For each of the three application mentioned in the introduction we have been able ot obtain quite promising results using the framework outlined above. Due to lack of space it is impossible to provide an elaborate description of all three case studies. Hence we will focus here on the first application, where the goal consists of predicting functional relationships between enzymes, as defined by the enzyme commission (EC) hierarchy. Further details about the other two case studies can be found in [17].

Our models were built and tested using a dataset of 1730 enzymes with known protein structures. All the enzyme structures had a resolution of at least 2.5 Å, they had a binding site volume between 350 and 3500 $Å^3$, and they were fully EC annotated. For evaluation purposes our database contained at least 20 observations for every EC number, leading to a total of 21 different EC numbers comprising members of all 6 top level codes. A heat map of the catalytic similarity of the enzymes is given in Figure 3. This catalytic similarity will be our relation of interest, constituting the output of the algorithm. As input we consider five state-of-the-art kernel matrices for enzymes, denoted cb (CavBase similarity), mcs (maximum common subgraph), lpcs (labeled point cloud superposition), wp (fingerprints) and wfp (weighted fingerprints). More details about the generation of these kernel matrices can be found in [18].

The dataset was randomized and split in four equal parts. Each part was withheld as a test set while the other three parts of the dataset were used for training and model selection. This process was repeated for each part so that every instance was used for training and testing (thus, four-fold outer cross-validation). In addition, a 10-fold inner cross validation loop was implemented for estimating the optimal regularization parameter λ, as recommended by [19]. The value of the hyperparameter was selected from a grid containing all the powers of 10 from 10^{-4} to 10^5. The final model was trained using the whole

training set and the median of the best hyperparameter values over the ten folds. We used the implementation RLScore in Python to train the models.

We benchmark our algorithm against an unsupervised procedure that is commonly used in bioinformatics for retrieval of enzymes. Given a specific enzyme query and one of the above similarity measures, a ranking is constructed by computing the similarity between the query and all other enzymes in the database. Enzymes having a high similarity to the query appear on top of the ranking, those exhibiting a low similarity end up at the bottom. More formally, let us represent the similarity between two enzymes by $K : \mathcal{V}^2 \to \mathbb{R}$, where $\mathcal{V}$ represents the set of all potential enzymes. Given the similarities $K(v, u)$ and $K(v, u')$ we compose the ranking of u and u' conditioned on the query v as:

$$u \succeq_v u' \Leftrightarrow K(v, u) \geq K(v, u'). \tag{8}$$

This approach follows in principle the same methodology as a nearest neighbor classifier, but rather a ranking than a class label should be seen as the output of the algorithm.

The quality of the obtained ranking can be evaluated by comparison with a *ground truth* ranking that is based on the EC numbers of the enzymes in the ranking (provided that their EC numbers are known). This ground truth ranking can be deduced from the catalytic similarity (i.e. ground truth similarity) between the query and all database enzymes. To this end, we count the number of successive correspondences from left to right, starting from the first digit in the EC label of the query and the database enzymes, and stopping as soon as the first mismatch occurs. For example, an enzyme query with number EC 2.4.2.23 has a similarity of two with a database enzyme labeled EC 2.4.99.12, since both enzymes belong to the family of glycosyl transferases. The same query manifests a similarity of one with an enzyme labeled EC 2.8.2.23. Both enzymes are transferases in this case, but they show no further similarity in the chemistry of the reactions to be catalyzed.

Table 1 gives a global summary of the results obtained for the unsupervised and the supervised ranking approach, respectively. All models score relatively well for all the performance measures. One can observe that supervised ranking outperforms unsupervised ranking for all four performance measures and all five kernels. The statistical significance of the differences was confirmed by a paired Wilcoxon test and a conservative Bonferroni correction for multiple hypotheses testing ($p < 10^{-6}$). Moreover, for all kernels and performance measures, supervised ranking decreases the standard deviation of the error, implying that the models become more stable.

Three important reasons can be put forward for explaining the improvement in performance. First of all, the traditional benefit of supervised learning plays an important role. One can expect that supervised ranking methods outperform unsupervised ranking methods, because they take ground-truth rankings into account during the training phase to steer towards retrieval of enzymes with a similar EC number. Conversely, unsupervised methods solely rely on the characterization of a meaningful similarity measure between enzymes, while ignoring EC numbers.

	cb	fp	wfp	mcs	lpcs
RA	**0.9062** (0.0603)	0.8815 (0.0689)	*0.8467* (0.0884)	0.8923 (0.0692)	0.8877 (0.0607)
MAP	**0.9321** (0.1531)	0.7207 (0.235)	*0.2836* (0.2132)	0.8846 (0.1578)	0.7339 (0.2074)
AUC	**0.9636** (0.0795)	0.8655 (0.1387)	*0.7527* (0.1468)	0.9393 (0.0919)	0.8794 (0.1126)
nDCG	**0.9922** (0.0329)	0.9349 (0.1424)	*0.3202* (0.3282)	0.9812 (0.0498)	0.9471 (0.1112)
RA	0.9951 (0.017)	0.995 (0.015)	**0.9981** (0.01)	*0.9944* (0.0112)	0.9952 (0.0156)
MAP	**0.9991** (0.0092)	0.9954 (0.0432)	0.9981 (0.0167)	0.9989 (0.0076)	*0.9835* (0.0797)
AUC	**0.9976** (0.0005)	0.9967 (0.0184)	0.9975 (0.0016)	0.9975 (0.0024)	*0.9934* (0.0368)
nDCG	**0.9968** (0.0171)	0.9942 (0.0424)	0.9979 (0.0173)	0.987 (0.0398)	*0.9812* (0.0673)

Table 1. A summary of the results obtained for unsupervised and supervised ranking (above and below double horizontal line, respectively). For each combination of model, kernel and performance measure, the average of the value over the different queries and folds is given by the standard deviation between parentheses. In every row the best ranking model is marked in bold, while the worst model is indicated in italic.

Second, we also advocate that supervised ranking methods have the ability to preserve the hierarchical structure of EC numbers in their predicted rankings. Figure 4 supports this claim. It summarizes the values used for ranking one fold of the test set obtained by the different models. So, for supervised ranking it visualizes the values $h(v, u)$, for unsupervised ranking it visualizes $K(v, u)$. Each row of the heatmap corresponds to one query. For the supervised models one notices a much better correspondence with the ground truth given in Figure 3. Furthermore, the different levels of catalytic similarity can be better distinguished.

A third reason for improvement by the supervised ranking method can be found in the exploitation of dependencies between different similarity values. Roughly speaking, if one is interested in the similarity between enzymes v and u, one can try to compute the similarity in a direct way, or derive it from the similarity with a third enzyme z. In the context of inferring protein-protein interaction and signal transduction networks, both methods are known as the direct and indirect approach, respectively [14, 20]. We argue that unsupervised ranking boils down to a direct approach, while supervised ranking should be interpreted as indirect. Especially when the kernel matrix contains noisy values, one can expect that the indirect approach allows for detecting the *back bone* entries and correcting the noisy ones.

4 Discussion

In this paper we discussed the applicability of kernel methods for learning relations or interactions in systems biology and related fields. Using such methods we obtained very promising results recently on three case studies. Due to lack of space we only discussed one of these case studies in detail. We showed that retrieval of enzymes w.r.t. functionality can be substantially improved by ap-

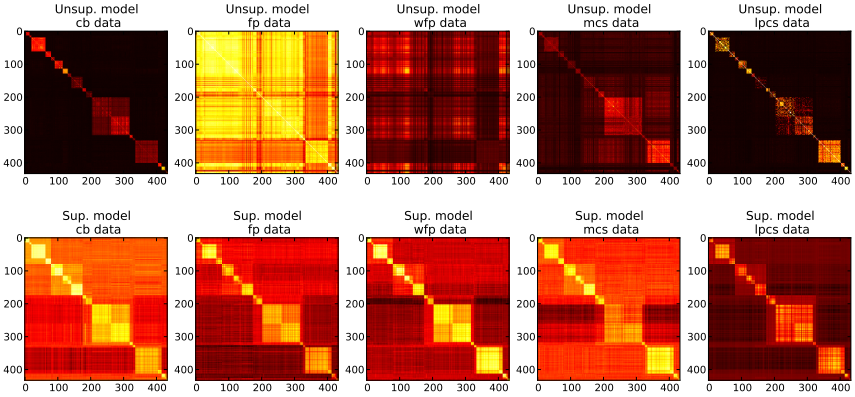

Fig. 4. Heat maps of the values used for ranking the database during one fold in the testing phase. Each row of the heat map corresponds to one query. This figure can be compared with the ground truth of Figure 3

plying a supervised ranking method that takes advantage of ground-truth EC numbers during the training phase. Supervised ranking outperformed unsupervised ranking in a consistent manner for all kernels and performance measures we considered. While the unsupervised approach succeeded quite well in returning exact matches to a query, the hierarchical structure of EC numbers was better preserved in the rankings predicted by the supervised approach. As such, supervised ranking can be interpreted as a correction mechanism for existing unsupervised methods that rely on the notion of similarity.

Acknowledgments

T.P. and A.A. are supported for this work by the Academy of Finland and W.W. by the Research Foundation of Flanders.

References

1. L. De Raedt. *Logical and Relational Learning*. Springer, 2009.
2. J.-P. Vert and Y. Yamanishi. Supervised graph inference. In *Advances in Neural Information Processing Systems*, volume 17, 2005.
3. E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side information. In *Advances in Neural Information Processing Systems*, volume 16, pages 521–528, 2002.
4. E. Hüllermeier and J. Fürnkranz. *Preference Learning*. Springer, 2010.
5. W. Waegeman, T. Pahikkala, A. Airola, T. Salakoski, M. Stock, and Bernard De Baets. A Kernel-based Framework for Learning Graded Relations from Data. *IEEE Transactions on Fuzzy Systems*, to appear., 2012.

6. F. Geerts, H. Mannila, and E. Terzi. Relational link-based ranking. In *Proceedings of the Conference on Very Large Databases*, pages 552–563, 2004.
7. S. Agarwal. Ranking on graph data. In *Proceedings of the International Conference on Machine Learning, Pittsburgh, PA, USA*, pages 25–32, 2006.
8. T. Qin, T. Liu, X. Zhang, D. Wang, W. Xiong, and H. Li. Learning to rank relational objects and its application to web search. In *Proceedings of the World Wide Web Conference*, 2008.
9. K. Kersting and Z. Xu. Learning preferences with hidden common cause relations. In *In Proceedings of the European Conference on Machine Learning.*, pages 676–691, 2009.
10. Z. Xu, K. Kersting, and T. Joachims. Fast active exploration for link-based preference learning using gaussian processes. In *Proceedings of the European Conference on Machine Learning*, pages 499–514, 2010.
11. M. Ng, X. Li, and Y. Ye. Multirank: co-ranking for objects and relations in multi-relational data. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 1217–1225, 2011.
12. Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Major components of the gravity recommendation system. *ACM SIGKDD Explorations Newsletter*, 9(2):80, December 2007.
13. Asa Ben-Hur and William Stafford Noble. Kernel methods for predicting protein-protein interactions. *Bioinformatics*, 21 Suppl 1:i38–46, June 2005.
14. J.-P. Vert, J Qiu, and W S Noble. A new pairwise kernel for biological network inference with support vector machines. *BMC Bioinformatics*, 8 (Suppl 1:S8, 2007.
15. H. Kashima, S. Oyama, Y. Yamanishi, and K. Tsuda. On pairwise kernels: An efficient alternative and generalization analysis. In Thanaruk Theeramunkong, Boonserm Kijsirikul, Nick Cercone, and Tu Bao Ho, editors, *PAKDD*, volume 5476 of *Lecture Notes in Computer Science*, pages 1030–1037. Springer, 2009.
16. T. Pahikkala, W. Waegeman, A. Airola, T. Salakoski, and B. De Baets. Conditional ranking on relational data. In J. Balcazar, F. Bonchi, A. Gionis, and M. Sebag, editors, *Proceedings of the European Conference on Machine Learning*, volume 6322 of *Lecture Notes in Computer Science*, pages 499–514. Springer Berlin / Heidelberg, 2010.
17. M. Stock. Learning pairwise relations in bioinformatics: Three case studies. Master's thesis, Ghent University, 2012.
18. M. Stock, T. Fober, E. Hüllermeier, S. Glinca, G. Klebe, C. Heitzer, T. Pahikkala, A. Airola, B. De Baets, and W. Waegeman. Supervised ranking for enhanced retrieval of enzyme functions. *Bioinformatics*, 2012. submitted.
19. S Varma and R Simon. Bias in Error Estimation when Using Cross-Validation for Model Selection. *Bioinformatics*, 7:91, 2006.
20. P Geurts, N Touleimat, M Dutreix, and F D'Alché-Buc. Inferring biological networks with output kernel trees. *BMC Bioinformatics*, 8(2):S4, 2007.