A Sparse Regularized Least-Squares Preference Learning Algorithm

Evgeni TSIVTSIVADZE, Tapio PAHIKKALA, Antti AIROLA, Jorma BOBERG, and Tapio SALAKOSKI

Turku Centre for Computer Science (TUCS)
Department of Information Technology
University of Turku, Joukahaisenkatu 3-5 B
20520 Turku, Finland
email: firstname.lastname@utu.fi

Abstract. Learning preferences between objects constitutes a challenging task that notably differs from standard classification or regression problems. The objective involves prediction of ordering of the data points. Furthermore, methods for learning preference relations usually are computationally more demanding than standard classification or regression methods. Recently, we have proposed a kernel based preference learning algorithm, called RankRLS, whose computational complexity is cubic with respect to the number of training examples. The algorithm is based on minimizing a regularized least-squares approximation of a ranking error function that counts the number of incorrectly ranked pairs of data points. When nonlinear kernel functions are used, the training of the algorithm might be infeasible if the amount of examples is large. In this paper, we propose a sparse approximation of RankRLS whose training complexity is considerably lower than that of basic RankRLS. In our experiments, we consider parse ranking, a common problem in natural language processing. We show that sparse RankRLS significantly outperforms basic RankRLS in this task. To conclude, the advantage of sparse RankRLS is the computational efficiency when dealing with large amounts of training data together with high dimensional feature representations.

1. Introduction

Recently, learning preference relations has received a lot of attention in machine learning research. Generally, the task can be cast as learning a function that is capable of ranking data points according to some given preference relation. The ranking algorithms are widely used in many areas such as information retrieval [6], natural language processing [2], et cetera.

In many cases the preference learning problem is reduced to classification of data point pairs, where one pair is preferred to the other one [5]. A major drawback associated with this approach is that the number of data point pairs grows quadratically with respect to the size of the dataset, making the training of a preference learner too expensive for large datasets. In [7], we proposed RankRLS whose computational complexity is the same as that of the standard RLS regression [12], even though RankRLS takes account of the data point pairs instead of the individual data points. A similar algorithm was proposed independently by [3]. The computational complexity of the kernel version of RankRLS is $O(m^3)$, where m is the size of the training set. In practise, also this may be infeasible for very large training sets. In this paper, we propose sparse RankRLS, a sparse regularized least-squares algorithm for learning preference relations. The computational complexity of sparse RankRLS is $O(mr^2)$. In fact, r can be selected to be much smaller than m and in some cases it can be considered as a constant. Thus, our algorithm can efficiently perform ranking using a large amount of training data together with high dimensional feature representations.

Sparse RankRLS can be used to learn pairwise preferences from scored data. In our setting, every data point provided to the algorithm consists of an input and its real valued score. The algorithm can be used for both object ranking and label ranking tasks (see e.g. [4] for in depth discussion about these types of tasks). However, in this paper we consider only label ranking. Therefore, we define every input to consist of an object and its label.

As an example, we consider the task of parse ranking, a common problem in natural language processing. In this case, each input consists of a sentence and its parse. That is, the sentence and the parse are considered as an object and a label, respectively. We are given a set of sentences and each sentence is associated with a set of parses. The task is to find the correct ordering of the parses of a sentence. We are not interested in preferences between parses associated with different sentences. Thus, the only relevant input pairs are the ones that are associated with the same sentence. As another example one can consider information retrieval task where we are given a set of web-search results obtained with a set of queries. The aim is to rank them according to the user preference. In this case, we are not interested in the order of the web documents obtained from different queries.

Training of the existing kernel based ranking algorithms, such as RankSVM [5], may be infeasible when the size of the training set is large. This is especially the case when nonlinear kernel functions are used. In this study, we suggest that sparse RankRLS makes it possible to take advantage of much more data in the training process than the non-sparse kernel methods do.

2. Ranking Task

We construct a training set from a given set of m data points. A data point $z = (x, y)$ consist of an input $x \in \mathcal{X}$ and its score $y \in \mathbb{R}$, where $\mathcal{X}$, called the input space, can be any set. We say that a data point $z = (x, y)$ is preferred to $z' = (x', y')$ if $y > y'$ and vice versa. We call data points tied if $y = y'$. In this paper, we consider label ranking (see e.g. [4]), where every input consists of an object and its label. An input pair is considered relevant if both inputs are associated with the same object. In parse ranking tasks, for example, each object is a sentence and the labels associated with it are parses generated for the sentence. The score of an input indicates how well the parse included in the input matches the correct parse of the sentence.

Following [7], we define an undirected graph whose vertices correspond to the training inputs. Two vertices in the graph are connected with an edge if the corresponding pair of inputs is relevant to the task. Let $W \in \mathbb{R}^{m\times m}$ denote the adjacency matrix of the graph. That is, $W_{i,j} = 1$ when the vertices indexed by i and j are connected, and $W_{i,j} = 0$ otherwise. Further, let $X = (x_1, \ldots, x_m) \in (\mathcal{X}^m)^T$ be a sequence of inputs, where $(\mathcal{X}^m)^T$ denotes the set of row vectors whose elements belong to $\mathcal{X}$. We also define $Y = (y_1, \ldots, y_m)^T \in \mathbb{R}^m$ to be a sequence of the corresponding scores. Finally, we consider a training set to be the triple $S = (X, Y, W)$.

Let us denote $\mathbb{R}^{\mathcal{X}} = \{f : \mathcal{X} \to \mathbb{R}\}$, and let $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{X}}$ be the hypothesis space. Further, let $f(X) = (f(x_1), \ldots, f(x_m))^T$. We measure how well a hypothesis $f \in \mathcal{H}$ is able to predict the direction of preference for the input pairs that are relevant to the task with the following function known as the disagreement or ranking error:

$$d(f(X), Y, W) = \frac{1}{N} \sum_{i,j=1}^{m} W_{i,j} \frac{1}{2} \left| \operatorname{sign}\big(y_i - y_j\big) - \operatorname{sign}\big(f(x_i) - f(x_j)\big) \right|, \quad (1)$$

where $N = \sum_{i,j=1}^{m} W_{i,j}$ and $\operatorname{sign}(\cdot)$ is the signum function.

3. Regularization

In order to construct an algorithm that selects a hypothesis f from $\mathcal{H}$, we have to define an appropriate cost function that measures how well the hypotheses fit the training data. We would also like to avoid too complex hypotheses that overfit at the training phase and are not able to generalize to unseen data. To give a formal representation of these aims, we follow [13] and consider the framework of regularized kernel methods in which $\mathcal{H}$ is so-called reproducing kernel Hilbert space (RKHS) defined by a positive definite kernel function k. The kernel functions (see e.g. [14]) are defined as follows. Let $\mathcal{F}$ denote the feature vector space. For any mapping $\Phi : \mathcal{X} \to \mathcal{F}$, the inner product $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$ of the mapped data points is called a kernel function. Using RKHS as our hypothesis space, we define the learning algorithm as

$$\mathcal{A}(S) = \operatorname*{argmin}_{f \in \mathcal{H}} J(f),$$

where

$$J(f) = c(f(X), Y, W) + \lambda \|f\|_k^2, \quad (2)$$

$f(X) = (f(x_1), \ldots, f(x_m))^T$, c is a real valued cost function, and $\lambda \in \mathbb{R}_+$ is a regularization parameter controlling the tradeoff between the cost on the training set and the complexity of the hypothesis. By the generalized representer theorem ([13]), the minimizer of (2) has the following form:

$$f(x) = \sum_{i=1}^{m} a_i k(x, x_i), \quad (3)$$

where $a_i \in \mathbb{R}$. Using this notation, we rewrite $f(X) = KA$ and $\|f\|_k^2 = A^T K A$, where $A = (a_1, \ldots, a_m)^T$.

It would be natural to use the disagreement error (1) as a cost function, however, it is well known that this leads to intractable optimization problem. Thus, following [7], we use a least-squares approximation of (1), that is, regressing the differences $y_i - y_j$ with $f(x_i) - f(x_j)$:

$$c(f(X), Y, W) = \frac{1}{2} \sum_{i,j=1}^{m} W_{i,j}((y_i - y_j) - (f(x_i) - f(x_j)))^2. \tag{4}$$

Note also that when using (4), not only the sign of $y_i - y_j$ but also its magnitude is included in the objective function (2).

Let $L = D - W$ be the Laplacian matrix [1] of the graph W, where D is the degree matrix of W. That is, D is a diagonal matrix whose entries are defined as

$$D_{i,i} = \sum_{j=1}^{m} W_{i,j}. \tag{5}$$

We observe that for any vector $p \in \mathbb{R}^m$ and an undirected weighted graph W with m vertices, we can write

$$\frac{1}{2} \sum_{i,j=1}^{m} W_{i,j}(p_i - p_j)^2 = p^T Dp - p^T Wp = p^T Lp.$$

Therefore, by selecting $p = Y - KA$, we rewrite the cost function (4) in a matrix form as

$$c(f(X), Y, W) = (Y - KA)^T L(Y - KA).$$

The RankRLS algorithm can be presented in matrix form as

$$\mathcal{A}(S) = \operatorname*{argmin}_{A} J(A),$$

where

$$J(A) = (Y - KA)^T L(Y - KA) + \lambda A^T KA. \tag{6}$$

As shown in [7], the minimizer of (6) is

$$A = (KLK + \lambda K)^{-1} KLY. \tag{7}$$

The computational complexity of the matrix inversion operation involved in (7) is $O(m^3)$. In practice, this makes the RLS training procedure infeasible when the amount of data available is large. Next, we propose a solution for this problem.

4. Sparse RankRLS

In this section, we propose an algorithm that is based on a similar kind of idea as the subset of regressors method [9,15] for the standard regularized least-squares regression. For in depth discussion of this type of techniques, we refer to [11].

Let $M = \{1, \ldots, m\}$ be the index set in which the indices refer to the examples in the training set and let $R \subseteq M$, $|R| = r$. By Z_{RM} we denote the submatrix of $Z \in \mathbb{R}^{m \times m}$ that contains only the rows indexed by R. Further, Z_{RR} denotes a submatrix of Z having only rows and columns indexed by R.

Now we consider instead of (3) a solution that allows only the training instances indexed by R to have nonzero coefficient, that is,

$$f(x) = \sum_{i \in R} a_i k(x, x_i).$$

We call the training examples indexed by R basis vectors. The problem of finding this type of hypothesis can be solved by finding the coefficients a_i, where $i \in R$. We observe that $f(X) = K_{MR}A$ and $\|f\|_k^2 = A^T K_{RR} A$, where a coefficient vector $A \in \mathbb{R}^r$, determines the sparse approximation of the minimizer of (2). Using these definitions, we present a method we call sparse RankRLS:

$$\mathcal{A}(S) = \operatorname*{argmin}_{A \in \mathbb{R}^r} J(A)$$

and

$$J(A) = (Y - K_{MR}A)^T L (Y - K_{MR}A) + \lambda A^T K_{RR} A.$$

We take the derivative of $J(A)$ with respect to A:

$$\begin{aligned} \frac{d}{dA} J(A) &= -2K_{RM}L(Y - K_{MR}A) + 2\lambda K_{RR}A \\ &= -2K_{RM}LY + (2K_{RM}LK_{MR} + 2\lambda K_{RR})A \end{aligned}$$

We set the derivative to zero and solve with respect to A:

$$A = (K_{RM}LK_{MR} + \lambda K_{RR})^{-1} K_{RM}LY. \tag{8}$$

The calculation of the solution (8) requires multiplications with a $m \times m$ matrix L which might be infeasible in practise. However, it can be performed efficiently using the following method.

Let $B \in \mathbb{R}^{m \times q}$, where q is number of objects in the training set. The value of $B_{i,j}$ is 1 when the ith input is associated with the jth object and 0 otherwise. Then matrix W can be written as $W = BB^T$. Now, the multiplication $K_{RM}LK_{MR}$ can be written as

$$K_{RM}LK_{MR} = K_{RM}(DK_{MR} - B(B^T K_{MR})),$$

where D is the diagonal degree matrix whose entries are defined in (5). The multiplication $K_{RM}LY$ can be done analogously. The computational complexity of the inversion operation used on $r \times r$ matrices is $O(r^3)$. The complexity of the matrix multiplications is $O(mr^2)$, because B contains only m nonzero elements. Selecting r to be much smaller than m, the overall training complexity of the sparse RankRLS algorithm is $O(mr^2)$.

Clearly, the selection of the index set R may have an influence on results obtained by our method. Different approaches for selecting R are discussed, for example, in [12]. There, it was found that simply selecting the elements of R randomly performs no worse than more sophisticated methods.

4.1. Efficient Training via Decompositions

One advantage of using sparse RankRLS instead of other ranking methods is efficient selection of regularization parameter. Using Cholesky decompositions for K_{RR}, we can rewrite the solution (8) as follows:

$$A = (K_{RM}LK_{MR} + \lambda CC^T)^{-1}K_{RM}LY,$$

where $K_{RR} = CC^T$. Now,

$$\begin{aligned}(K_{RM}LK_{MR} + \lambda CC^T)^{-1} &= (CC^{-1}K_{RM}LK_{MR}(C^T)^{-1}C^T + \lambda CC^T)^{-1}\\ &= (C^T)^{-1}(C^{-1}K_{RM}LK_{MR}(C^T)^{-1} + \lambda I)^{-1}C^{-1}\\ &= (C^T)^{-1}(V\Lambda V^T + \lambda I)^{-1}C^{-1}\\ &= (C^T)^{-1}V\hat{\Lambda}_\lambda V^T C^{-1},\end{aligned}$$

where $V\Lambda V^T$ is the eigen decomposition of $C^{-1}K_{RM}LK_{MR}(C^T)^{-1}$ with V, Λ being the eigenvector matrix and diagonal matrix containing the corresponding eigenvalues, respectively, and $\hat{\Lambda}_\lambda = (\Lambda + \lambda I)^{-1}$. Therefore, we rewrite the solution (8) as follows:

$$A = (C^T)^{-1}V\hat{\Lambda}_\lambda V^T C^{-1}K_{RM}LY.$$

The decompositions and the inversion of C can be calculated in $O(r^3)$ time, and hence the overall training complexity is not increased. The computational cost of calculating $\hat{\Lambda}_\lambda$ is $O(r)$, since $(\Lambda + \lambda I)$ is a diagonal matrix. When the matrices $V^TC^{-1}K_{RM}LY \in \mathbb{R}^{r\times 1}$ and $(C^T)^{-1}V \in \mathbb{R}^{r\times r}$ are stored in memory, the subsequent training with different values of regularization parameters can be performed in $O(r^2)$ time.

5. Experiments

We evaluate the performance of sparse RankRLS on the task of ranking of the parses of an unseen sentence. We use the BioInfer corpus [10] which consists of 1100 manually annotated sentences. A detailed description of the parse ranking problem and the data used in the experiments is given in [16]. Each sentence is associated with a set of candidate parses. The manual annotation of the sentence, present in the corpus, provides the correct parse. Further, each candidate parse is associated with a goodness score that indicates how close to the correct parse it is. The correct ranking of the parses associated with the same sentence is determined by this score. While the scoring induces a total or-

Basic RLS Regressor	Sparse RLS Regressor	Basic RankRLS	Sparse RankRLS
0.27	0.24	0.23	0.21

Table 1. Comparison of the parse ranking performances of RLS regressor, sparse RLS regressor, basic RankRLS, and sparse RankRLS using the disagreement error (1) as the performance evaluation measure.

der over the whole set of parses, the preferences between parses associated with different sentences are not considered in the parse ranking task.

As a similarity measure for parses, we use the best performing graph kernel considered in [8]. The disagreement error (1) is used to measure the performance of the ranking algorithms. The error is calculated for each sentence separately and the performance is averaged over all sentences. We have previously shown that basic RankRLS significantly outperforms the basic RLS regressor in the parse ranking task [7]. As shown in Section 4, with sparse RankRLS it is possible to take advantage of much more training data than with basic RankRLS only with a small increase in computational complexity. In our experiments, we test how beneficial this is by comparing sparse RankRLS with basic RankRLS. Furthermore, we make a comparison with basic and sparse RLS regressors. As the basis vectors of the sparse algorithms, we use the training examples of the non-sparse ones, while adding more examples as non-basis vectors. We select 500 sentences for the training of rankers. For basic RankRLS we randomly select 5 parses per sentence, and for sparse RankRLS we use additional 15 randomly selected parses per sentence as non-basis vectors. Thus, $m = 10000$ and $r = 2500$ for the sparse methods.

The algorithms have the regularization parameter λ that controls the trade-off between the minimization of the training error and the complexity of the learned function. Further, kernel function has parameters too. We evaluate the performance of sparse RankRLS as well as other baseline methods by performing a 10-fold cross-validation on the sentence level so that all parses generated from the same sentence would always be in the same fold. We use 500 sentences for the parameter estimation and the rest are reserved for the final evaluation. The appropriate values of the regularization and the kernel parameters are determined by grid search with 10-fold cross-validation on the parameter estimation data. The parameter selection is performed separately for each experiment.

Finally, the algorithms are trained on the whole parameter estimation data set with the best found parameter values and tested with the sentences reserved for the final validation. The results of the validation are presented in Table 1. The results show that the sparse RankRLS algorithm notably outperforms RLS regressor, sparse RLS regressor, and basic RankRLS. Furthermore, to test the statistical significance of the performance differences between the sparse RankRLS algorithm and the other methods, we conducted Wilcoxon signed-ranks tests. The sentences reserved for the final validation are considered as independent trials. We observed that the performance differences are statistically significant ($p < 0.05$).

6. Conclusion

We propose sparse RankRLS, a sparse regularized least-squares algorithm, for learning preference relations. The computational complexity of the algorithm is $O(mr^2)$, where m is the number of training examples, and r is much smaller than m. We formulate the algorithm within the kernel framework. The key feature of the algorithm is the ability to efficiently train the ranker with large amounts of data in high dimensional feature spaces, thus improving the ranking performance. This is achieved by finding a sparse solution to the regularized least-squares problem. In our experiments, we consider parse ranking task. It is shown that sparse RankRLS significantly outperforms basic RLS regressor, sparse RLS regressor, and basic RankRLS.

Acknowledgments

This work has been supported by Tekes, the Finnish Funding Agency for Technology and Innovation. We would like to thank CSC, the Finnish IT center for science, for providing us extensive computing resources.

References

[1] R. A. Brualdi and H. J. Ryser. *Combinatorial Matrix Theory.* Cambridge University Press, 1991.

[2] M. Collins. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning*, pages 175–182, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[3] C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In Z. Ghahramani, editor, *Proceedings of the 24th Annual International Conference on Machine Learning*, pages 169–176. Omnipress, 2007.

[4] J. Fürnkranz and E. Hüllermeier. Preference learning. *Künstliche Intelligenz*, 19(1):60–61, 2005.

[5] R. Herbrich, T. Graepel, and K. Obermayer. Support vector learning for ordinal regression. In *Proceedings of the Ninth International Conference on Articial Neural Networks*, pages 97–102, London, 1999. Institute of Electrical Engineers.

[6] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, pages 133–142, New York, NY, USA, 2002. ACM Press.

[7] T. Pahikkala, E. Tsivtsivadze, A. Airola, J. Boberg, and T. Salakoski. Learning to rank with pairwise regularized least-squares. In T. Joachims, H. Li, T.-Y. Liu, and C. Zhai, editors, *SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, pages 27–33, 2007.

[8] T. Pahikkala, E. Tsivtsivadze, J. Boberg, and T. Salakoski. Graph kernels versus graph representations: a case study in parse ranking. In T. Gärtner, G. C. Garriga, and T. Meinl, editors, *Proceedings of the ECML/PKDD'06 workshop on Mining and Learning with Graphs (MLG'06)*, Berlin, Germany, 2006.

[9] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9), 1990.

[10] S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, and T. Salakoski. BioInfer: A corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8:50, 2007.

[11] J. Quinonero-Candela, C. E. Rasmussen, and C. K. I. Williams. Approximation methods for gaussian process regression. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large-Scale Kernel Machines*, pages 203–224. MIT Press, Cambridge, Ma, USA, 09 2007.

[12] R. Rifkin, G. Yeo, and T. Poggio. Regularized least-squares classification. In J. Suykens, G. Horvath, S. Basu, C. Micchelli, and J. Vandewalle, editors, *Advances in Learning Theory: Methods, Model and Applications*, volume 190 of *NATO Science Series III: Computer and System Sciences*, pages 131–154, Amsterdam, 2003. IOS Press.

[13] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In D. Helmbold and R. Williamson, editors, *Proceedings of the 14th Annual Conference on Computational Learning Theory*, pages 416–426, Berlin, Germany, 2001. Springer.

[14] B. Schölkopf and A. J. Smola. *Learning with kernels.* MIT Press, Cambridge, Massachusetts, 2002.

[15] A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 911–918, San Francisco, Ca, 2000. Morgan Kaufmann Publishers Inc.

[16] E. Tsivtsivadze, T. Pahikkala, S. Pyysalo, J. Boberg, A. Mylläri, and T. Salakoski. Regularized least-squares for parse ranking. In A. F. Famili, J. N. Kok, J. M. Peña, A. Siebes, and A. J. Feelders, editors, *Advances in Intelligent Data Analysis VI*, pages 464–474. Springer, 2005.