Co-Regularized Least-Squares for Label Ranking

Evgeni Tsivtsivadze, Tapio Pahikkala, Jorma Boberg, Tapio Salakoski, and Tom Heskes

Abstract Situations when only a limited amount of labeled data and a large amount of unlabeled data are available to the learning algorithm are typical for many real-world problems. To make use of unlabeled data in preference learning problems, we propose a semisupervised algorithm that is based on the multiview approach. Our algorithm, which we call Sparse Co-RankRLS, minimizes a least-squares approximation of the ranking error and is formulated within the co-regularization framework. It operates by constructing a ranker for each view and by choosing such ranking prediction functions that minimize the disagreement among all of the rankers on the unlabeled data. Our experiments, conducted on real-world dataset, show that the inclusion of unlabeled data can improve the prediction performance significantly. Moreover, our semisupervised preference learning algorithm has a linear complexity in the number of unlabeled data items, making it applicable to large datasets.

1 Introduction

Semisupervised learning algorithms have gained more and more attention in recent years as unlabeled data is typically much easier to obtain than labeled one. *Multi-view* learning algorithms, such as co-training [1], split the attributes into independent sets and an algorithm is learnt based on these different "views". The goal of the learning process consists in finding for every view a prediction function (for the

E. Tsivtsivadze (✉) and T. Heskes
Institute for Computing and Information Sciences, Radboud University Nijmegen
Toernooiveld 1, 6525 ED Nijmegen, The Netherlands
e-mail: evgeni@science.ru.nl, t.heskes@science.ru.nl

T. Pahikkala, J. Boberg, and T. Salakoski
Turku Centre for Computer Science (TUCS),
Department of Information Technology, University of Turku,
Joukahaisenkatu 3-5 B, 20520 Turku, Finland
e-mail: firstname.lastname@it.utu.fi

J. Fürnkranz and E. Hüllermeier (eds.), *Preference Learning*,
DOI 10.1007/978-3-642-14125-6_6,

learning task) performing well on the labeled data of the designated view such that all prediction functions agree on the unlabeled data. Closely related to this approach is the *co-regularization* framework described in [20], where the same idea of agreement maximization between the predictors is central. Briefly stated, algorithms based upon this approach search for hypotheses from different Reproducing Kernel Hilbert Spaces [19], namely views, such that the training error of each hypothesis on the labeled data is small and, at the same time, the hypotheses give similar predictions for the unlabeled data. Within this framework, the disagreement is taken into account through a co-regularization term. Empirical results show that the co-regularization approach works well for classification [20], regression [2], and clustering [3] tasks. Moreover, theoretical investigations demonstrate that the co-regularization approach reduces the Rademacher complexity by an amount that depends on the "distance" between the views [18, 21].

We consider the problem of learning a function capable of arranging data points according to a given preference relation [8]. Training of existing kernel-based ranking algorithms, such as RankSVM [10], may be infeasible when the size of the training set is large. This is especially the case when nonlinear kernel functions are used. Recently, a sparse preference learning algorithm, called *Sparse RankRLS*, that can take advantage of a large amount of data in the training process, has been proposed [23]. In this paper, we formulate a co-regularized version of RankRLS, called *Sparse Co-RankRLS*, and aim to improve the performance of RankRLS by making it applicable to situations when only a small amount of labeled data, but a large amount of unlabeled data, is available.

We evaluate our algorithm on a *parse ranking task* [5, 24] that is a common problem in natural language processing. In this task, the aim is to rank a set of parses associated with a single sentence, based on some goodness criteria giving a score to the parse. In our experiments, we consider the case when both scored and a large amount of unscored data is available to the learning algorithm. We demonstrate that Sparse Co-RankRLS is computationally efficient when trained on large datasets and the obtained results are significantly better than the ones obtained with the standard RankRLS algorithm. We consider the parse ranking task as label ranking. However, in the parse ranking task the labels (i.e. the parses of a sentence) are instance-specific. That is, for each sentence, we have a different set of labels, while in the conventional label ranking setting labels are not instance specific.

2 Problem Setting

Let $\mathcal{X}$ be a set of instances and $\mathcal{Y}$ be a set of labels. The learning scenario we consider is *label ranking* [6,8], i.e., we want to predict for any instance $\boldsymbol{x} \in \mathcal{X}$ (e.g., a person) a preference relation $\mathcal{P}_{\boldsymbol{x}} \subseteq \mathcal{Y} \times \mathcal{Y}$ among the set of labels $\mathcal{Y}$, where each label $y \in \mathcal{Y}$ can be thought of as an alternative. An element $(y, y') \in \mathcal{P}_{\boldsymbol{x}}$ means

that the instance $\boldsymbol{x}$ prefers the label y compared to y', also written as $y \succ_{\boldsymbol{x}} y'$. [1] We assume that the (true) preference relation $\mathcal{P}_{\boldsymbol{x}}$ is transitive and asymmetric for each instance $\boldsymbol{x} \in \mathcal{X}$. As training information, we are given a finite set $\{(\boldsymbol{q}_i, s_i)\}_{i=1}^n$ of n data points, where each data point $(\boldsymbol{q}_i, s_i) = ((\boldsymbol{x}_i, y_i), s_i) \in (\mathcal{X} \times \mathcal{Y}) \times \mathbb{R}$ consists of an instance-label tuple $\boldsymbol{q}_i = (\boldsymbol{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ and its score $s_i \in \mathbb{R}$. We say that the pair of data points $((\boldsymbol{x}, y), s)$ and $((\boldsymbol{x}', y'), s')$ is *relevant*, iff $\boldsymbol{x} = \boldsymbol{x}'$. Considering relevant pair $((\boldsymbol{x}, y), s)$ and $((\boldsymbol{x}, y'), s')$, we say that instance $\boldsymbol{x}$ *prefers* label y to y', iff $s > s'$. If $s = s'$, the labels are called *tied*. Accordingly, we write $y \succ_{\boldsymbol{x}} y'$ if $s > s'$ and $y \sim_{\boldsymbol{x}} y'$ if $s = s'$.

A *label ranking function* is a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ mapping each instance-label tuple $(\boldsymbol{x}, y)$ to a real value representing the (predicted) relevance of the label y with respect to the instance $\boldsymbol{x}$. This induces for any instance $\boldsymbol{x} \in \mathcal{X}$ a transitive preference relation $\mathcal{P}_{f,\boldsymbol{x}} \subseteq \mathcal{Y} \times \mathcal{Y}$ with $(y, y') \in \mathcal{P}_{f,\boldsymbol{x}} \Leftrightarrow f(\boldsymbol{x}, y) > f(\boldsymbol{x}, y')$. Ties can be broken arbitrarily. Informally, the goal of our ranking task is to find a label ranking function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ such that the ranking $\mathcal{P}_{f,\boldsymbol{x}} \subseteq \mathcal{Y} \times \mathcal{Y}$ induced by the function for any instance $\boldsymbol{x} \in \mathcal{X}$ is a good "prediction" for the true (unknown) preference relation $\mathcal{P}_{\boldsymbol{x}} \subseteq \mathcal{Y} \times \mathcal{Y}$.

In order to incorporate the relevance information, we define an undirected preference graph which is determined by its adjacency matrix W such that for each i and j $(1 \leq i, j \leq n, i \lneqq j)$ $[W]_{i,j} = 1$, if $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ is relevant, and $[W]_{i,j} = 0$ if $(\boldsymbol{q}_i, \boldsymbol{q}_j)$ is not relevant. To avoid loops, we set $[W]_{i,i} = 0$ for $i = 1, \ldots, n$, although an instance-label tuple is relevant to itself. Furthermore, let $Q = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n)^t \in (\mathcal{X} \times \mathcal{Y})^n$ be the vector of instance-label training tuples and $S = (s_1, \ldots, s_n)^t \in \mathbb{R}^n$ the corresponding vector of scores. Given these definitions, our training set is the triple $\mathcal{T} = (Q, S, W)$.

Let us define $\mathbb{R}^{\mathcal{Q}} = \{f : \mathcal{Q} \to \mathbb{R}\}$ with $\mathcal{Q} = \mathcal{X} \times \mathcal{Y}$ and let $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{Q}}$ be the hypothesis space of possible ranking functions. To measure how well a hypothesis $f \in \mathcal{H}$ is able to predict the preference relations $\mathcal{P}_{\boldsymbol{x}}$ for all instances $\boldsymbol{x} \in \mathcal{X}$, we consider the following cost function that captures the amount of incorrectly predicted pairs of relevant training data points:

$$d(f, \mathcal{T}) = \frac{1}{2} \sum_{i,j=1}^{n} [W]_{i,j} \left| \operatorname{sign}\big(s_i - s_j\big) - \operatorname{sign}\big(f(\boldsymbol{q}_i) - f(\boldsymbol{q}_j)\big) \right|, \tag{1}$$

where $\operatorname{sign}(\cdot)$ denotes the signum function

$$\operatorname{sign}(r) = \begin{cases} 1, & \text{if } r > 0 \\ -1, & \text{if } r \leq 0 \end{cases}.$$

It is well-known that the use of cost functions such as (1) leads to intractable optimization problems. Therefore, we consider the following least-squares approximation,

[1] As described in [8], one can distinguish between *weak preference* ($\succeq$) and *strict preference* ($\succ$), where $y \succ_{\boldsymbol{x}} y' \Leftrightarrow (y \succeq_{\boldsymbol{x}} y') \wedge (y' \not\succeq_{\boldsymbol{x}} y)$; furthermore, $y \sim_{\boldsymbol{x}} y' \Leftrightarrow (y \succeq_{\boldsymbol{x}} y') \wedge (y' \succeq_{\boldsymbol{x}} y)$.

which in fact regresses the differences $s_i - s_j$ with $f(\boldsymbol{q}_i) - f(\boldsymbol{q}_j)$ of relevant training data points $\boldsymbol{q}_i$ and $\boldsymbol{q}_j$:

$$c(f, \mathcal{T}) = \frac{1}{2} \sum_{i,j=1}^{n} [W]_{i,j} \Big((s_i - s_j) - (f(\boldsymbol{q}_i) - f(\boldsymbol{q}_j)) \Big)^2. \tag{2}$$

Note that the above cost function c also takes the extent of discrepancy between the predicted preference $(f(\boldsymbol{q}_i) - f(\boldsymbol{q}_j))$ and the training preference $(s_i - s_j)$ of pairs of relevant training data points into account.

3 Regularized Least-Squares Ranking

The co-regularized ranking algorithm presented in this paper stems from the results developed in [12] and [23]. For completeness, we briefly review these results in this section.

We aim to construct an algorithm that selects a hypothesis f from $\mathcal{H}$ which minimizes (2) and which is, at the same time, not too "complex", i.e., which does not overfit at training phase and is therefore able to generalize to unseen data. We consider the framework of regularized kernel methods [19], in which $\mathcal{H}$ is a so-called *Reproducing Kernel Hilbert Space* (RKHS) defined by a positive definite kernel function.

3.1 Regularization Framework

Let $k : \mathcal{Q} \times \mathcal{Q} \to \mathbb{R}$ be a positive definite kernel defined on the set $\mathcal{Q}$. Then we define $\mathcal{H}$ as

$$\mathcal{H} = \Big\{ f \in \mathbb{R}^{\mathcal{Q}} \mid f(\cdot) = \sum_{j=1}^{\infty} \beta_j k(\cdot, \boldsymbol{q}_j), \beta_j \in \mathbb{R}, \boldsymbol{q}_j \in \mathcal{Q}, \|f\|_{\mathcal{H}} < \infty \Big\}, \tag{3}$$

where $\|\cdot\|_{\mathcal{H}}$ denotes the norm in $\mathcal{H}$. Using the RKHS $\mathcal{H}$ as our hypothesis space, we consider the optimization problem

$$\mathcal{A}(\mathcal{T}) = \operatorname*{argmin}_{f \in \mathcal{H}} J(f), \tag{4}$$

where $J(f) = c(f, \mathcal{T}) + \lambda \|f\|^2_{\mathcal{H}}$ 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 [19], the minimizer of (4) has the form

$$f^*(\cdot) = \sum_{i=1}^{n} a_i k(\cdot, \boldsymbol{q}_i) \tag{5}$$

with appropriate coefficients $a_i \in \mathbb{R}$. Hence, we can focus on functions $f \in \mathcal{H}$ having the above form. Defining the kernel matrix $K \in \mathbb{R}^{n \times n}$ with entries of the form $[K]_{i,j} = k(\boldsymbol{q}_i, \boldsymbol{q}_j)$ and $f(Q) = (f(\boldsymbol{q}_1), \ldots, f(\boldsymbol{q}_n))^t \in \mathbb{R}^n$, we can write $f(Q) = KA$ and $\|f\|^2_{\mathcal{H}} = A^t KA$, where $A = (a_1, \ldots, a_n)^t \in \mathbb{R}^n$ is a corresponding coefficient vector.[2]

3.2 RankRLS

Let $\mathcal{L} = D - W$ be the Laplacian matrix [4], where D denotes the diagonal matrix with elements of the form $[D]_{i,i} = \sum_{j=1}^{n} [W]_{i,j}$. Using a slightly different notation, it is shown in [12] that the cost function (2) can be rewritten as

$$c(f, \mathcal{T}) = (S - KA)^t \mathcal{L}(S - KA). \tag{6}$$

Considering this representation of the cost function c, we get the following optimization problem called *RankRLS* in [12]:

$$\mathcal{A}(\mathcal{T}) = \operatorname*{argmin}_{A \in \mathbb{R}^n} J(A), \tag{7}$$

where $J(A) = (S - KA)^t \mathcal{L}(S - KA) + \lambda A^t KA$. Using the fact that $\mathcal{L}$ is positive semidefinite [15] and assuming that K is positive definite, it is easy to see that the Hessian matrix $H(J) = 2K^t \mathcal{L}K + 2\lambda K$ of J is positive definite. Thus, J is strictly convex and the global minimum of J can be obtained by setting the first derivative $\frac{\mathrm{d}}{\mathrm{d}A} J(A) = -2K^t \mathcal{L}(S - KA) + 2\lambda KA$ to zero and by solving the resulting system of equations with respect to A. The optimal solution for (7) is

$$A = (K\mathcal{L}K + \lambda K)^{-1} K\mathcal{L}S = (\mathcal{L}K + \lambda I)^{-1} \mathcal{L}S, \tag{8}$$

where I denotes the identity matrix. The computational complexity of the matrix inversion in (8) is $\mathcal{O}(n^3)$.

Fact 1 *[12] For fixed $\lambda \in \mathbb{R}^+$, the solution of the RankRLS optimization problem (7) can be found in $\mathcal{O}(n^3)$ time.*

3.3 Sparse RankRLS

Similarly to [14] and [22], an approximation algorithm aiming at reducing the cubic running time of the RankRLS approach is developed in [23]: The cost function c is evaluated over *all* points, but only a subset of the coefficients $a_1, \ldots, a_n$ is allowed

[2] Unless stated otherwise, we assume that a kernel matrix K is positive definite, i.e., $B^t KB > 0$ for all $B \in \mathbb{R}^n$, $B \neq 0$. This can be ensured, for example, by performing a small diagonal shift.

to be nonzero, thus an approximation of the optimization problem is considered. Let $R = \{i_1, \ldots, i_r\} \subseteq \{1, \ldots, n\}$ be a subset of indices. Then, we only allow the coefficients $a_{i_1}, \ldots, a_{i_r}$ to be nonzero in (5), i.e., we search for minimizers $\widehat{f} \in \mathcal{H}$ having the form

$$\widehat{f}(\cdot) = \sum_{j=1}^{r} a_{i_j} k(\cdot, \boldsymbol{q}_{i_j}). \tag{9}$$

By defining $\bar{K} \in \mathbb{R}^{n \times r}$ to be the submatrix of $K \in \mathbb{R}^{n \times n}$ that only contains the columns indexed by R and by defining $\widehat{K} \in \mathbb{R}^{r \times r}$ to be the submatrix of $\bar{K}$ only containing the rows indexed by R, we can express $\widehat{f}(Q) = (\widehat{f}(\boldsymbol{q}_1), \ldots, \widehat{f}(\boldsymbol{q}_n))^t \in \mathbb{R}^n$ as $\widehat{f}(Q) = \bar{K}\widehat{A}$ and $\|\widehat{f}\|^2_{\mathcal{H}} = \widehat{A}^t \widehat{K} \widehat{A}$, where $\widehat{A} = (a_{i_1}, \ldots, a_{i_r})^t \in \mathbb{R}^r$. Given these notations, the approximation presented in [23], called *Sparse RankRLS*, can be formulated as

$$\mathcal{A}(\mathcal{T}) = \operatorname*{argmin}_{\widehat{A} \in \mathbb{R}^r} \widehat{J}(\widehat{A}), \tag{10}$$

where $\widehat{J}(\widehat{A}) = (S - \bar{K}\widehat{A})^t \mathcal{L}(S - \bar{K}\widehat{A}) + \lambda \widehat{A}^t \widehat{K} \widehat{A}$. Setting the derivative of $\widehat{J}$ to zero and solving the resulting system of equations with respect to $\widehat{A}$ leads to

$$\widehat{A} = (\bar{K}^t \mathcal{L} \bar{K} + \lambda \widehat{K})^{-1} \bar{K}^t \mathcal{L} S. \tag{11}$$

The overall training complexity of the Sparse RankRLS algorithm is $\mathcal{O}(nr^2)$, see [23] for more details.

Fact 2 *([23]) For fixed $\lambda \in \mathbb{R}^+$, the solution of the Sparse RankRLS optimization problem (10) can be found in $\mathcal{O}(nr^2)$ time.*

Hence, selecting r to be much smaller than n results in a significant acceleration of the training procedure. Clearly, the selection of the index set R may have an influence on results obtained by the above approximation approach. Various methods for subset selection have been proposed (see e.g.[17, 25]), however, for simplicity and computational efficiency, we consider random selection of data points contained in R.

3.4 Constructing Kernels with Subsets of Regressors

Considering the Sparse RankRLS algorithm, the label predictions for the training data points can be obtained by $\bar{K}\widehat{A}$. Using the Woodbury matrix identity [9] and (11) and by defining $\widetilde{K} = \frac{1}{\lambda} \bar{K} \widehat{K}^{-1} \bar{K}^t$, we can reformulate this expression as follows:

$$\begin{aligned} \bar{K}\widehat{A} &= \bar{K}(\bar{K}^t \mathcal{L} \bar{K} + \lambda \widehat{K})^{-1} \bar{K}^t \mathcal{L} S \\ &= \bar{K} \left(\frac{1}{\lambda} \widehat{K}^{-1} - \frac{1}{\lambda} \widehat{K}^{-1} \bar{K}^t \left(\frac{1}{\lambda} \mathcal{L} \bar{K} \widehat{K}^{-1} \bar{K}^t + I \right)^{-1} \frac{1}{\lambda} \mathcal{L} \bar{K} \widehat{K}^{-1} \right) \bar{K}^t \mathcal{L} S \end{aligned}$$

$$
\begin{aligned}
&= (\widetilde{K} - \widetilde{K}(\mathcal{L}\widetilde{K} + I)^{-1}\mathcal{L}\widetilde{K})\mathcal{L}S \\
&= (\widetilde{K}(I - (\mathcal{L}\widetilde{K} + I)^{-1}\mathcal{L}\widetilde{K})\mathcal{L}S \\
&= (\widetilde{K}((\mathcal{L}\widetilde{K} + I)^{-1}(\mathcal{L}\widetilde{K} + I) - (\mathcal{L}\widetilde{K} + I)^{-1}\mathcal{L}\widetilde{K})\mathcal{L}S \\
&= \widetilde{K}(\mathcal{L}\widetilde{K} + I)^{-1}(\mathcal{L}\widetilde{K} + I - \mathcal{L}\widetilde{K})\mathcal{L}S \\
&= \widetilde{K}(\mathcal{L}\widetilde{K} + I)^{-1}\mathcal{L}S.
\end{aligned}
$$

Note that because $\mathcal{L}$ is positive semidefinite and $\widetilde{K}$ is positive definite, their product $\mathcal{L}\widetilde{K}$ contains only nonnegative eigenvalues [11]. Hence, $\mathcal{L}\widetilde{K} + I$ is invertible. Further, the last term can be rewritten as $\widetilde{K}(\mathcal{L}\widetilde{K} + I)^{-1}\mathcal{L}S = \check{K}(\mathcal{L}\check{K} + \lambda I)^{-1}\mathcal{L}S$, where $\check{K} = \bar{K}\widehat{K}^{-1}\bar{K}^t \in \mathbb{R}^{n\times n}$. These derivations show that the Sparse RankRLS algorithm operating with a kernel function k is essentially equivalent to the standard RankRLS algorithm operating with a modified kernel $\check{k}$. In the following section, we use this fact for constructing different Hilbert spaces by taking different sets of basis vectors.

4 Co-Regularized Least Squares Ranking

The co-regularization approach is based on the idea of constructing M prediction functions from M different RKHSs such that the error of each function on the labeled data is small and, at the same time, the functions give similar predictions for the unlabeled data. These RKHSs can stem from different data point descriptions (i.e., different features), from different kernel functions, and/or from different subsets of the data. Note that the case of different data point descriptions can be obtained by applying the kernel functions only to appropriate subsets of features. Further, as depicted in Sect. 3.4, taking different subsets of the data leads to different RKHSs. Hence, we will consider M RKHSs $\mathcal{H}_1, \ldots, \mathcal{H}_M$ along with their corresponding kernel functions $k_v : \mathcal{Q} \times \mathcal{Q} \to \mathbb{R}, 1 \leq v \leq M$, to incorporate the co-regularization approach.

4.1 Co-Regularized RankRLS

Considering our ranking task, we have a training set $\mathcal{T} = (Q, S, W)$ originating from a set $\{(\boldsymbol{q}_i, s_i)\}_{i=1}^n$ of data points *with* scoring information, where $Q = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n)^t \in \mathcal{Q}^n$, $S = (s_1, \ldots, s_n)^t \in \mathbb{R}^n$, and W is the matrix incorporating the relevance information. Moreover, we have a training set $\widetilde{\mathcal{T}} = (\widetilde{Q}, \widetilde{W})$ from a set $\{\boldsymbol{q}_{n+i}\}_{i=1}^l$ of data points *without* scoring information, $\widetilde{Q} = (\boldsymbol{q}_{n+1}, \ldots, \boldsymbol{q}_{n+l})^t \in \mathcal{Q}^l$, and an appropriate adjacency matrix $\widetilde{W}$. To avoid misunderstandings with the definition of the label ranking task, we will use the terms "scored" instead of "labeled" and "unscored" instead of "unlabeled".

In the ranking task, we search for a vector $\mathbf{f} = (f_1, \ldots, f_M) \in \mathcal{H}_1 \times \ldots \times \mathcal{H}_M$ of prediction functions which minimizes

$$J(\mathbf{f}) = \sum_{v=1}^{M} c(f_v, \mathcal{T}) + \lambda \sum_{v=1}^{M} \|f_v\|^2_{\mathcal{H}_v} + \nu \sum_{v,u=1}^{M} \tilde{c}(f_v, f_u, \widetilde{\mathcal{T}}), \tag{12}$$

where $\lambda, \nu \in \mathbb{R}^+$ are regularization parameters and $\tilde{c}$ is the loss function measuring the disagreement between the prediction functions of the views on the unscored data:

$$\tilde{c}(f_v, f_u, \widetilde{\mathcal{T}}) = \frac{1}{2} \sum_{i,j=1}^{l} \widetilde{[W]}_{i,j} \Big(\big(f_v(\boldsymbol{q}_{n+i}) - f_v(\boldsymbol{q}_{n+j}) \big) - \big(f_u(\boldsymbol{q}_{n+i}) - f_u(\boldsymbol{q}_{n+j}) \big) \Big)^2.$$

Applying the representer theorem [19] in this context shows that the minimizers $f_v^* \in \mathcal{H}_v$ of (12) for $v = 1, \ldots, M$ have the form

$$f_v^*(\cdot) = \sum_{i=1}^{n} a_i^{(v)} k_v(\cdot, \boldsymbol{q}_i) + \sum_{i=1}^{l} a_{n+i}^{(v)} \, k_v(\cdot, \boldsymbol{q}_{n+i}) \tag{13}$$

with adequate coefficients $a_1^{(v)}, \ldots, a_{n+l}^{(v)} \in \mathbb{R}$. Using matrix notations, we can reformulate (12) as

$$\begin{aligned} J(\mathbf{A}) = &\sum_{v=1}^{M} (S - L_v A_v)^t \mathcal{L}_L (S - L_v A_v) + \lambda \sum_{v=1}^{M} {A_v}^t K_v A_v \\ &+ \nu \sum_{v,u=1}^{M} (U_v A_v - U_u A_u)^t \mathcal{L}_U (U_v A_v - U_u A_u), \end{aligned} \tag{14}$$

where $A_v = (a_1^{(v)}, \ldots, a_{n+l}^{(v)})^t \in \mathbb{R}^{n+l}$, and $\mathbf{A} = (A_1^t, \ldots, A_M^t)^t \in \mathbb{R}^{M(n+l)}$. The matrix $L_v \in \mathbb{R}^{n \times (n+l)}$ has entries of the form $[L_v]_{i,j} = k_v(\boldsymbol{q}_i, \boldsymbol{q}_j)$ and the matrix $U_v \in \mathbb{R}^{l \times (n+l)}$ has entries of the form $[U_v]_{i,j} = k_v(\boldsymbol{q}_{n+i}, \boldsymbol{q}_j)$. Stacking both matrices up gives the matrix K_v:

$$K_v = \begin{pmatrix} L_v \\ U_v \end{pmatrix} \in \mathbb{R}^{(n+l) \times (n+l)}.$$

Further, $\mathcal{L}_L \in \mathbb{R}^{n \times n}$ and $\mathcal{L}_U \in \mathbb{R}^{l \times l}$ denote the Laplacian matrices corresponding to W and $\widetilde{W}$, respectively. Hence, we have the following optimization problem:

$$\mathcal{A}(\mathcal{T}, \widetilde{\mathcal{T}}) = \underset{\mathbf{A} \in \mathbb{R}^{M(n+l)}}{\operatorname{argmin}} \; J(\mathbf{A}). \tag{15}$$

4.2 Sparse Co-Regularized RankRLS

Similar to the non-co-regularized case, the above optimization problem could be difficult to solve due to the computations involving the complete kernel matrices. Hence, as in Sect. 3, we aim at solving an approximation of the above optimization problem by only allowing a subset of the coefficients in (13) to be nonzero for each view. This corresponds to taking submatrices of the original matrices, i.e., for each view v we define $\bar{L}_v \in \mathbb{R}^{n \times r}$ to be the submatrix of L_v that only contains the columns corresponding to r selected basis vectors $\boldsymbol{q}_{c_v(1)}, \ldots, \boldsymbol{q}_{c_v(r)}$. Here, the number $c_v(i) \in \{1, \ldots, n+l\}$ denotes the index (column) of the ith selected vector of view v. Accordingly, we define $\bar{U}_v \in \mathbb{R}^{l \times r}$ to be the submatrix of U_v that only contains the columns corresponding to $\boldsymbol{q}_{c_v(1)}, \ldots, \boldsymbol{q}_{c_v(r)}$. Finally, we define $\widehat{K}_v \in \mathbb{R}^{r \times r}$ to be the kernel matrix with elements $\left[\widehat{K}_v\right]_{i,j} = k_v(\boldsymbol{q}_{c_v(i)}, \boldsymbol{q}_{c_v(j)})$. Hence, we obtain the following optimization problem, which we call *Sparse Co-RankRLS*:

$$\mathcal{A}(\mathcal{T}, \widetilde{\mathcal{T}}) = \operatorname*{argmin}_{\widehat{\mathbf{A}} \in \mathbb{R}^{Mr}} \widehat{J}(\widehat{\mathbf{A}}), \tag{16}$$

where

$$\begin{aligned} \widehat{J}(\widehat{\mathbf{A}}) = &\sum_{v=1}^{M} \left(S - \bar{L}_v \widehat{A}_v\right)^t \mathcal{L}_L \left(S - \bar{L}_v \widehat{A}_v\right) + \lambda \sum_{v=1}^{M} \widehat{A}_v{}^t \widehat{K}_v \widehat{A}_v \\ &+ \nu \sum_{v,u=1}^{M} \left(\bar{U}_v \widehat{A}_v - \bar{U}_u \widehat{A}_u\right)^t \mathcal{L}_U \left(\bar{U}_v \widehat{A}_v - \bar{U}_u \widehat{A}_u\right), \end{aligned} \tag{17}$$

$\widehat{A}_v = (a^{(v)}_{c_v(1)}, \ldots, a^{(v)}_{c_v(r)})^t \in \mathbb{R}^r$, and $\widehat{\mathbf{A}} = (\widehat{A}^t_1, \ldots, \widehat{A}^t_M)^t \in \mathbb{R}^{Mr}$. For ease of notation, we consider the same number of basis vectors for each view. Given this matrix formulation of our optimization problem, we can follow the framework described in [2] to find a closed form for the solution: Taking the partial derivative of $\widehat{J}(\widehat{\mathbf{A}})$ with respect to $\widehat{A}_v$ we get

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\widehat{A}_v} \widehat{J}(\widehat{\mathbf{A}}) = &-2\bar{L}_v^t \mathcal{L}_L (S - \bar{L}_v \widehat{A}_v) + 2\lambda \widehat{K}_v \widehat{A}_v \\ &-4\nu \sum_{u=1, u \neq v}^{M} \bar{U}_v^t \mathcal{L}_U (\bar{U}_u \widehat{A}_u - \bar{U}_v \widehat{A}_v). \end{aligned}$$

By defining $G_v^\nu = 2\nu(M-1)\bar{U}_v^t \mathcal{L}_U \bar{U}_v$, $G_v^\lambda = \lambda \widehat{K}_v$ and $G_v = \bar{L}_v^t \mathcal{L}_L \bar{L}_v$, we can rewrite the above term as

$$\frac{\mathrm{d}}{\mathrm{d}\widehat{A}_v}\widehat{J}(\widehat{\mathbf{A}}) = 2(G_v + G_v^{\nu} + G_v^{\lambda})\widehat{A}_v - 2\bar{L}_v^t\mathcal{L}_L S$$
$$-4\nu \sum_{u=1,u\neq v}^{M} \bar{U}_v^t\mathcal{L}_U\bar{U}_u\widehat{A}_u.$$

At the optimum we have $\frac{\mathrm{d}}{\mathrm{d}\widehat{A}_v}\widehat{J}(\widehat{\mathbf{A}}) = 0$ for all views, thus we get the exact solution by solving

$$\begin{pmatrix} \bar{G}_1 & -2\nu\bar{U}_1^t\mathcal{L}_U\bar{U}_2 \ldots \\ -2\nu\bar{U}_2^t\mathcal{L}_U\bar{U}_1 & \bar{G}_2 \quad \ldots \\ \vdots & \vdots \quad \ddots \end{pmatrix} \begin{pmatrix} \widehat{A}_1 \\ \widehat{A}_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \bar{L}_1^t\mathcal{L}_L S \\ \bar{L}_2^t\mathcal{L}_L S \\ \vdots \end{pmatrix}$$

with respect to $\widehat{A}_1, \ldots, \widehat{A}_M$, where $\bar{G}_v = G_v + G_v^{\nu} + G_v^{\lambda}$. The left-hand side matrix is positive definite and therefore invertible (see Appendix). By defining

$$B = \begin{pmatrix} G_1 & 0 & \ldots \\ 0 & G_2 & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix} \quad D = \begin{pmatrix} G_1^{\lambda} & 0 & \ldots \\ 0 & G_2^{\lambda} & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix} E = \begin{pmatrix} \bar{L}_1^t\mathcal{L}_L S \\ \bar{L}_2^t\mathcal{L}_L S \\ \vdots \end{pmatrix}$$

$$C = \begin{pmatrix} G_1^{\nu} & -2\nu\bar{U}_1^t\mathcal{L}_U\bar{U}_2 \ldots \\ -2\nu\bar{U}_2^t\mathcal{L}_U\bar{U}_1 & G_2^{\nu} \quad \ldots \\ \vdots & \vdots \quad \ddots \end{pmatrix}$$

we can formulate the solution of the system as follows:

$$\widehat{\mathbf{A}} = (B + C + D)^{-1}E. \tag{18}$$

The computational complexity of constructing the vector E is $\mathcal{O}(Mnr)$. Further, the matrices B, C, and D can be constructed in $\mathcal{O}(Mr^2n)$, $\mathcal{O}(M^2r^2l)$, and $\mathcal{O}(Mr^2)$, respectively. The resulting matrix $(B + C + D) \in \mathbb{R}^{Mr\times Mr}$ can be inverted in $\mathcal{O}(M^3r^3)$. Hence, our algorithm scales linearly in the number of unscored data items. Note that the multiplications involving the Laplacian matrices $\mathcal{L}_L$ and $\mathcal{L}_U$ can be accelerated using the approach described in [23]. Assuming $l \geq n$, we have shown the following theorem:

Theorem 1. *For fixed parameters $\lambda, \nu \in \mathbb{R}^+$ and assuming $l \geq n$, the solution of the Sparse Co-RankRLS optimization problem (16) can be found in $\mathcal{O}(M^3r^3 + M^2r^2l)$ time.*

5 Efficient Regularization Parameter Selection

When performing experiments, the recurrent matrix inversion in (18) for each combination of the regularization parameters λ and ν could be time-consuming. Therefore, we propose a procedure which accelerates this parameter selection process. Writing D as $D = \lambda \acute{D}$ with an appropriate (positive definite) matrix $\acute{D}$ and rewriting $\acute{D}$ as $\acute{D} = PP^t$ using the Cholesky decomposition [9], we obtain

$$\begin{aligned}(B + C + D)^{-1} &= (B + C + \lambda \acute{D})^{-1} \\ &= (PP^{-1}(B + C)(P^t)^{-1}P^t + \lambda PP^t)^{-1} \\ &= (P^t)^{-1}(P^{-1}(B + C)(P^t)^{-1} + \lambda I)^{-1}P^{-1}.\end{aligned}$$

Further, the matrix $P^{-1}(B + C)(P^t)^{-1}$ can be eigen decomposed to $V \Lambda V^t$, where Λ is a diagonal matrix containing the eigenvalues and V is the matrix composed of the eigenvectors [9]. Hence, we get

$$\begin{aligned}(B + C + D)^{-1} &= (P^t)^{-1}(V \Lambda V^t + \lambda I)^{-1}P^{-1} \\ &= (P^t)^{-1}V(\Lambda + \lambda I)^{-1}V^t P^{-1}\end{aligned}$$

and the solution in (18) can be rewritten as

$$\widehat{\mathbf{A}} = (P^t)^{-1}V(\Lambda + \lambda I)^{-1}V^t P^{-1}E.$$

Thus, by fixing the parameter ν, we can efficiently search for the second regularization parameter λ. The decompositions and the inversion of P can be calculated in $\mathcal{O}(M^3 r^3)$ time, and hence, the overall training complexity is not increased. The computational cost of calculating $(\Lambda + \lambda I)^{-1}$ is $\mathcal{O}(Mr)$, since it is a diagonal matrix. If the matrices $V^t P^{-1}E \in \mathbb{R}^{Mr \times 1}$ and $(P^t)^{-1}V \in \mathbb{R}^{Mr \times Mr}$ are stored in memory, the subsequent training with different values of λ can be performed in $\mathcal{O}(M^2 r^2)$ time.

6 Experiments

We evaluate the performance of the Sparse Co-RankRLS algorithm[3] on the task of ranking given parses for an unseen sentence. For this purpose, we use the BioInfer corpus [16] which consists of 1,100 manually annotated sentences. A detailed description of the parse ranking problem and the data used in the experiments is given in [24]. Each sentence is associated with a set of candidate parses. The manual

[3] Python implementation of the algorithm and the dataset are available on request.

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

Using the definitions presented in Sect. 2, we consider each sentence as an instance and the parses generated for the sentence as the labels associated with it. The score of an input indicates how well the parse included in the input matches the correct parse of the sentence. We have previously demonstrated that the RankRLS algorithm performs comparably to some state-of-the-art ranking methods [12]. In this section, we will compare the performance of the Sparse Co-RankRLS algorithm with that of the RankRLS algorithm.

6.1 Experimental Setup

From the 1,100 sentences of the BioInfer corpus, we randomly select 500 and 600 sentences for the training and final validation phase, respectively. To simulate a semisupervised setting, we consider that only half of sentence-parse pairs in the training set are scored, while the remaining sentence-parse pairs do not have the scoring information associated with them. For the evaluation of the Sparse Co-RankRLS method, we set the number M of views to 2. Further, we randomly select 20 sentence-parse pairs from the training data set as basis vectors for the first view and repeat this procedure for the second view. According to Sect. 4, we select different basis vectors for each view.

Both algorithms have the regularization parameter λ that controls the tradeoff between the minimization of the training error and the complexity of the learnt function(s). In addition, the Sparse Co-RankRLS algorithm has the regularization parameter ν that controls the agreement between the predictions of the different views. As a similarity measure for parses, we use the best performing graph kernel with the appropriate parameter considered in [13]. The values of the regularization parameters for RankRLS as well as for Sparse Co-RankRLS are estimated during a fivefold cross-validation procedure, with the splits being performed on the sentence level ensuring that all parses associated with the same sentence are present in the same fold. In the semisupervised setting each fold consists of one tenth of scored and unscored data present in the training set. For the cross-validation phases, we randomly select parses for each sentence to be associated with it, out of which 30 parses are used for training the model and 30 for testing. Finally, we use 30 parses per sentence for the final validation procedure.

Table 1 Comparison of the parse ranking performances of the standard RankRLS and the Sparse Co-RankRLS algorithms using a normalized version of the disagreement error (1) as performance evaluation measure. The results of the Sparse Co-RankRLS algorithm are obtained by averaging the predictions of the two views

Standard RankRLS	Sparse Co-RankRLS
0.348	0.326

6.2 Results

The normalized version of 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.

The algorithms are trained on the whole parameter estimation data set with the best found parameter values and tested with the 600 sentences reserved for the final validation. The results of the validation are presented in Table 1. They show that the Sparse Co-RankRLS algorithm notably outperforms the RankRLS method. We note that random selection of the basis vectors for both methods has an influence on the performance of the learning algorithm. To avoid variations in the final results obtained with particular set of basis vectors, we perform complete experiment 5 times, selecting different sets for basis vectors in all of the experiments and report averaged results.

Furthermore, to test the statistical significance of the performance difference between the Sparse Co-RankRLS and RankRLS algorithms, we conduct the Wilcoxon signed-ranks test [7]. The sentences reserved for the final validation are considered as independent trials. We observe that the performance differences are statistically significant ($p < 0.05$).

6.3 Learning Curve

To evaluate performance of the Sparse Co-RankRLS algorithm with respect to the number of unscored sentence-parse pairs used for training, we divide 4,000 sentence-parse pairs into 4 parts containing 1,000, 2,000, 3,000, and 4,000 unscored data, respectively. The algorithm is trained using complete scored training set with best found parameters that were estimated with fivefold cross-validation procedure. The separate test set is used for final validation of the algorithm. Each of the unscored data sets is re-sampled 5 times and obtained results are averaged. The outcomes of the experiments are presented in Fig. 1.

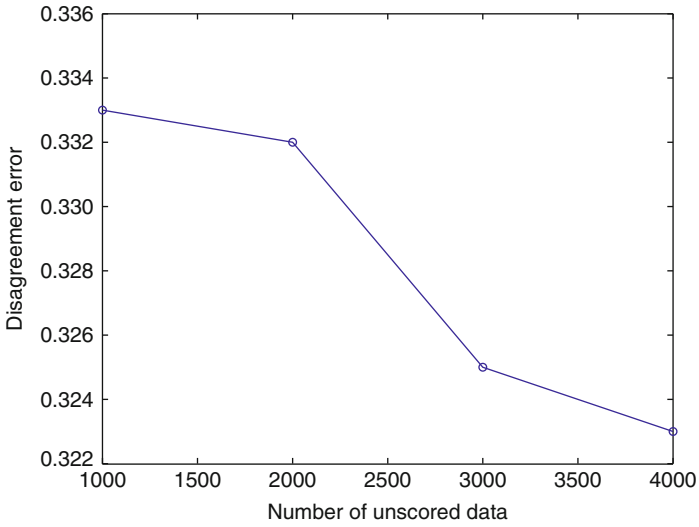

Fig. 1 The plot shows the relation between the disagreement error and the amount of unscored data

7 Conclusions

We propose Sparse Co-RankRLS, a semisupervised regularized least-squares algorithm, for learning preference relations. The computational complexity of the algorithm is $\mathcal{O}(M^3r^3 + M^2r^2l)$, where l is the number of unscored training examples, M is the number of views, and r is the number of basis vectors. We formulate the algorithm within the co-regularization framework, which aims at improving the prediction performance by minimizing the disagreement of all prediction hypotheses on the unscored data. In our experiments, we consider a parse ranking task and show that the Sparse Co-RankRLS algorithm significantly outperforms the standard RankRLS algorithm on this task.

Due to the fact that our semisupervised preference learning algorithm has a linear complexity in the number of unscored examples, it is primarily applicable in cases when only a small amount of scored but a large amount of unscored data is available for training. In the future, we aim to evaluate our Sparse Co-RankRLS algorithm on various tasks where scored data is scarce.

Acknowledgements We acknowledge support from the Netherlands Organization for Scientific Research (NWO), in particular a Vici grant (639.023.604). We also thank CSC, the Finnish IT center for science for providing us with computing resources.

References

1. A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, in *Proceedings of the 11th Annual Conference on Computational Learning Theory* (ACM, New York, NY, USA, 1998), pp. 92–100
2. U. Brefeld, T. Gärtner, T. Scheffer, S. Wrobel, Efficient co-regularised least squares regression, in *Proceedings of the 23rd International Conference on Machine learning* (ACM, New York, NY, USA, 2006), pp. 137–144
3. U. Brefeld, T. Scheffer, Co-em support vector learning, in *Proceedings of the 21st International Conference on Machine learning* (ACM, New York, NY, USA, 2004), p. 16
4. R.A. Brualdi, H.J. Ryser, *Combinatorial Matrix Theory* (Cambridge University Press, 1991)
5. M. Collins, N. Duffy, New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron, in *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (Association for Computational Linguistics, Morristown, NJ, USA, 2001), pp. 263–270
6. O. Dekel, C.D. Manning, Y. Singer, Log-linear models for label ranking, in *Advances in Neural Information Processing Systems*, vol. 16, ed. by S. Thrun, L. Saul, B. Schölkopf (MIT, Cambridge, MA, 2004), pp. 497–504
7. J. Demšar, Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
8. J. Fürnkranz, E. Hüllermeier, Preference learning. Künstliche Intelligenz **19**(1), 60–61 (2005)
9. G.H. Golub, C.F. Van Loan, *Matrix Computations* (Johns Hopkins University Press, 1996)
10. R. Herbrich, T. Graepel, K. Obermayer, Support vector learning for ordinal regression, in *Proceedings of the Ninth International Conference on Articial Neural Networks* (Institute of Electrical Engineers, London, 1999), pp. 97–102
11. R. Horn, C.R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, 1985)
12. T. Pahikkala, E. Tsivtsivadze, A. Airola, J. Järvinen, J. Boberg, An efficient algorithm for learning to rank from preference graphs. Mach. Learn. **75**(1), 129–165 (2009)
13. T. Pahikkala, E. Tsivtsivadze, J. Boberg, T. Salakoski, Graph kernels versus graph representations: a case study in parse ranking, in *Proceedings of the ECML/PKDD'06 workshop on Mining and Learning with Graphs*, ed. by T. Gärtner, G.C. Garriga, T. Meinl, Berlin, Germany (pp. 181–188) (2006)
14. T. Poggio, F. Girosi, Networks for approximation and learning. Proc. IEEE **78**(9), 1481–1497 (1990)
15. A. Pothen, H.D. Simon, K.-P. Liou, Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl. **11**(3), 430–452 (1990)
16. S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, T. Salakoski, BioInfer: A corpus for information extraction in the biomedical domain. BMC Bioinformatics, **8**, 50 (2007)
17. J. Quinonero-Candela, CE. Rasmussen, A unifying view of sparse approximate Gaussian process regression. J. Mach. Learn. Res. **6**, 1939–1959 (2005)
18. D. Rosenberg, P.L. Bartlett, The rademacher complexity of co-regularized kernel classes, in *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, ed. by M. Meila, X. Shen (2007), pp. 396–403
19. B. Schölkopf, R. Herbrich, A.J. Smola, A generalized representer theorem, in *Proceedings of the 14th Annual Conference on Computational Learning Theory*, ed. by David P. Helmbold, B. Williamson (Springer, London, 2001), pp. 416–426
20. V. Sindhwani, P. Niyogi, M. Belkin, A co-regularization approach to semi-supervised learning with multiple views, in *Proceedings of ICML Workshop on Learning with Multiple Views* (2005)
21. V. Sindhwani, D. Rosenberg, An rkhs for multi-view learning and manifold co-regularization, in *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, ed. by A. McCallum, S. Roweis (Omnipress, Helsinki, Finland, 2008), pp. 976–983
22. A.J. Smola, B. Schölkopf, Sparse greedy matrix approximation for machine learning, in *Proceedings of the 17th International Conference on Machine Learning*, ed. by Pat Langley (Morgan Kaufmann Publishers, San Francisco, Ca, USA, 2000), pp. 911–918

23. E. Tsivtsivadze, T. Pahikkala, A. Airola, J. Boberg, T. Salakoski, A sparse regularized least-squares preference learning algorithm, in *10th Scandinavian Conference on Artificial Intelligence (SCAI 2008)*, vol. 173, ed. by A. Holst, P. Kreuger, P. Funk (IOS, 2008), pp. 76–83
24. E. Tsivtsivadze, T. Pahikkala, S. Pyysalo, J. Boberg, A. Mylläri, T. Salakoski, Regularized least-squares for parse ranking, in *Advances in Intelligent Data Analysis VI*, ed. by A. Fazel Famili, J.N. Kok, J.M. Peña, A. Siebes, A.J. Feelders (Springer, 2005), pp. 464–474
25. P. Vincent, Y. Bengio, Kernel matching pursuit. *Mach. Learn.* **48**(1–3), 165–187 (2002)

Appendix

We will show that the matrix

$$\begin{pmatrix} \bar{G}_1 & -2\nu\bar{U}_1^t\mathcal{L}_U\bar{U}_2\ \ldots \\ -2\nu\bar{U}_2^t\mathcal{L}_U\bar{U}_1 & \bar{G}_2 \quad \ldots \\ \vdots & \vdots \quad \ddots \end{pmatrix}$$

is positive definite. To prove that, we decompose the above matrix into a sum of matrices

$$X_1 = \begin{pmatrix} \bar{G}_1 - 2\nu(M-1)\bar{U}_1^t\mathcal{L}_U\bar{U}_1 & 0 & \ldots \\ 0 & \bar{G}_2 - 2\nu(M-1)\bar{U}_2^t\mathcal{L}_U\bar{U}_2\ \ldots \\ \vdots & \vdots \quad \ddots \end{pmatrix}$$

and

$$X_2 = \begin{pmatrix} \nu(M-1)\bar{U}_1^t\mathcal{L}_U\bar{U}_1 & -\nu\bar{U}_1^t\mathcal{L}_U\bar{U}_2 \quad \ldots \\ -\nu\bar{U}_2^t\mathcal{L}_U\bar{U}_1 & \nu(M-1)\bar{U}_2^t\mathcal{L}_U\bar{U}_2\ \ldots \\ \vdots & \vdots \quad \ddots \end{pmatrix}.$$

The matrix X_1 is positive definite as each block matrix is positive definite (we require the matrix $\widehat{K}_v$ to be positive definite). Further, the matrix X_2 is positive semidefinite as we can write it as a sum of positive semidefinite matrices of the form

$$\begin{pmatrix} 0 \cdots & 0 & \cdots & 0 & \cdots 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 \cdots & \nu \bar{U}_i^t \mathcal{L}_U \bar{U}_i & \cdots & -\nu \bar{U}_i^t \mathcal{L}_U \bar{U}_j & \cdots 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 \cdots & -\nu \bar{U}_j^t \mathcal{L}_U \bar{U}_i & \cdots & \nu \bar{U}_j^t \mathcal{L}_U \bar{U}_j & \cdots 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 \cdots & 0 & \cdots & 0 & \cdots 0 \end{pmatrix} = X_{(i,j)}^t X_{(i,j)},$$

where $X_{(i,j)} = \big(0, \ldots, 0, \sqrt{\nu} P \bar{U}_i, 0, \ldots, 0, -\sqrt{\nu} P \bar{U}_j, 0, \ldots, 0\big)$. Here, the positive semidefinite matrix $\mathcal{L}_U$ is decomposed as $\mathcal{L}_U = P^t P$ using the Cholesky decomposition [9].