Semi-supervised Learning with Explicit Relationship Regularization

Kwang In Kim
Lancaster University

James Tompkin
Harvard SEAS

Hanspeter Pfister
Harvard SEAS

Christian Theobalt
MPI for Informatics

Abstract

In many learning tasks, the structure of the target space of a function holds rich information about the relationships between evaluations of functions on different data points. Existing approaches attempt to exploit this relationship information implicitly by enforcing smoothness on function evaluations only. However, what happens if we explicitly regularize the relationships between function evaluations? Inspired by homophily*, we regularize based on a smooth* relationship function*, either defined from the data or with labels. In experiments, we demonstrate that this significantly improves the performance of state-of-the-art algorithms in semi-supervised classification and in spectral data embedding for constrained clustering and dimensionality reduction.*

1. Introduction

Regularization attempts to prevent overfitting in ill-posed problems. It is commonly applied in semi-supervised learning tasks: Given a sparse labeling on u data points with s labels $\{(x_i, y_i)\}_{i=1}^{s}$, our goal is to learn a function f which maps from an input space M to a target space N. The lack of labels is compensated for by exploiting unlabeled data points to provide additional information, e.g., on the geometry of and/or probability distribution on M, from which the data are generated. Regularization tries to measure and limit the complexity of proposed f solutions by preferring smaller training errors and placing restrictions on smoothness. This established approach helps solve a variety of learning problems, such as image and shape classification, tracking, and retrieval (e.g., [21, 19, 5, 17]).

The target space N has a structure which may be defined implicitly or, in some applications, explicitly through pair-wise similarity or dissimilarity potentials. However, current regularization methods operate only on the function itself, and do not *explicitly* consider the potentially rich informative structure of N as something which can be used for regularization. Regularizing the structure — or the *relationships* — is inspired by *homophily*, which is actively used to predict relationships within social networks [13, 1, 9]: individuals with similar mutual friends, or local structure, are more likely to influence one another, e.g., if two individuals A and B are friends then they tend to have mutual friends, and if A has an enemy C, then B is also likely to be an enemy of C. We demonstrate that a priori knowledge of the smoothness of a relationship between entities can be exploited in inference on the entity itself.

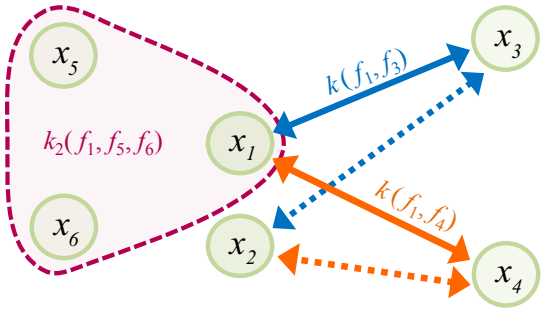

Figure 1. If two data points x_1 and x_2 are close on the domain M of f, then conventional regularizers enforce that the corresponding function values f_1 and f_2 in co-domain N of f are similar ($f_i \equiv f(x_i)$). We assume that relationships between pairs of function evaluations f_i and f_j are represented by smooth functions $k(f_i,f_j)$, e.g., a similarity measure. Our regularizer explicitly enforces that $k(f_1,f_j)$ and $k(f_2,f_j)$ are similar for any j. For instance, if $k(f_1,f_3)$ is large as f_1 and f_3 are similar, but $k(f_1,f_4)$ is small as f_1 and f_4 are dissimilar (solid arrows), then our algorithm enforces that $k(f_2,f_3)$ and $k(f_2,f_4)$ are large and small, respectively (dotted arrows), as x_1 and x_2 are close in M. The same principle applies to high-order relationships: if $k_2(f_1, f_5, f_6)$ represents a ternary relationship, e.g., a third-order correlation, the similarity of $k_2(f_1,f_5,f_6)$ and $k_2(f_2,f_5,f_6)$ is enforced.

One example that benefits from this principle occurs when *relationship labels* are provided. In semi-supervised or *constrained* spectral clustering [12, 14, 18], the labels are provided not on the underlying cluster assignment function f but on the binary relationships k between the function evaluations, as *must-link* or *cannot-link* labels. These are exploited by applying conventional regularization on f with the condition that the constraints are satisfied. However, in this case, the relationship itself can also be a natural object to regularize (Fig. 1). Applying homophily, if (x_1,x_3) *must link*, i.e., if they belong to the same cluster, then a *relationship function* k on N is defined such that $k(f(x_1),f(x_3))$ is positive. For point x_2, which is close to x_1 in M, we expect the relationship function $k(f(x_2),f(x_3))$ to be positive also.

In general, the relationship itself is not formally defined or observed; however, in many applications, certain relationships are manifested through a smooth function, where the number of arguments corresponds to the relationship degree, e.g., a distance metric is a function of two arguments. k can be defined either directly from the data or from labels; either way, once the relationship is defined, regularization is independent of the existence of labels and therefore applies generally to any learning problem.

1.1. Function-only and implicit relationships

We begin with a regularized empirical risk minimization framework where $f: M \to N$ minimizes the energy functional:

$$\mathcal{E}(f) = \sum_{i=1,\ldots,s} l(y_i, f(x_i)) + \lambda \mathcal{R}(f), \tag{1}$$

where λ is a regularization parameter, $\mathcal{R}: N^M \to \mathbb{R}^+$ is the regularization functional that measures the *complexity* of the input function, and $l: N \times N \to \mathbb{R}^+$ is the loss function. For simplicity, we assume that $N = \mathbb{R}^n$ and adopt the squared loss: $l(a,b) = \|a-b\|^2$, but our framework can be easily extended to other convex loss functions. Extension to non-Euclidean N is also possible as discussed in Sec. 2.2.

While a variety of semi-supervised learning algorithms can potentially benefit from our approach (see [4] for a comprehensive survey), we focus on the successful class of graph Laplacian-based approaches. One of the best-established classes of regularizers is based on applying differential operators to f:

$$\mathcal{R}_D(f) = \int_M \|[Df](x)\|^2 dV(x), \tag{2}$$

where domain M is the Riemannian manifold as is common in semi-supervised learning, and $dV(x)$ is the natural volume element of M. If D is the first-order differential operator $\frac{d}{dx}$, then $\mathcal{R}_D$ is the familiar *harmonic energy* functional [2, 16]:

$$\mathcal{R}^{\mathrm{h}}(f) = \int_M \|[\nabla f](x)\|^2_{T^*_x} dV(x), \tag{3}$$

with Riemannian connection ∇ in M, and cotangent space $T^*_x := T^*_x(M)$ of M at x [11].

Roughly, this energy functional applies a differential operator to the input function and measures the corresponding squared norm. Minimizing this energy functional leads to a *smooth* function with smaller first-order magnitudes. When M is only indirectly observed through data point clouds, $\mathcal{R}^{\mathrm{h}}$ is instantiated based on the graph Laplacian [2], the performance of which has been demonstrated in numerous applications.

Harmonic energy can be regarded as a first-order regularizer since it directly penalizes only variations of f. For relationships, denoted by double brackets, e.g., $[\![A,B]\!]$, this roughly corresponds to minimizing the pair-wise deviations between self-relationships $[\![f(x+\mathrm{d}x)]\!]$ and $[\![f(x)]\!]$, where $[\![A]\!]$ is simply as informative as A, with no consideration of relationships between entities.[1]

If we apply this first-order operator ∇ twice to f, i.e., $D = \nabla^2$, we minimize the resulting second-order energy and penalize the deviations of the two pair-wise deviations $[\![f(x+\mathrm{d}x), f(x)]\!]$ and $[\![f(x-\mathrm{d}x), f(x)]\!]$. This can be regarded as an example of a second-order relationship regularizer, with the relationship defined as the difference between two entities. Higher-order relationship regularizers then enforce smoothness on relationships involving more than two entities by increasing the order of D. For instance, the state-of-the-art *p-iterated Laplacian semi-norm* [20] measures smoothness of $(p-1)$-th order relationships.

$$\mathcal{R}^{\mathrm{p}}(f) = \int_M f(x)[\Delta^p f](x) dV(x). \tag{4}$$

However, existing differential operator-based regularizers focus only on *local* relationships. By construction, $Df(x)$ is defined for an arbitrarily small open set containing x, and so it does not explicitly enforce smoothness over any pair $[\![f(x), f(x')]\!]$ and $[\![f(x''), f(x''')]\!]$ of relationships when all four input points x, x', x'', x''' do not lie within a small neighborhood — even when x and x'' are close. This property is shared by established regularizers in Euclidean space (i.e., M is Euclidean): For instance, the well-known Gaussian kernel regularizer corresponds to Eq. 2 with D being a combination of powers of the Laplacian operator [15].

Implicitly, any existing regularization functional regularizes any high-order relationships, as smoothness on f implies smoothness on pairs $[\![f(x), f(x')]\!]$. While apparently redundant, we will show experimentally that adding *explicit* control over relationship regularization increases utility over existing function-only regularizers.

The success of local high-order derivative-based regularizers supports this claim: In 1D space, minimizing the first-order derivative norm as a regularizer implicitly minimizes all high-order derivative norms, as the only null space of the first-order derivative operator is the space of constant functions (as these have zero high-order derivatives). Nevertheless, the use of high-order derivative-based regularizers, e.g., thin plate spline and Gaussian regularizers, is strongly supported by their empirical performances.

That high-order derivative-based regularizers can be considered as local high-order relationship regularizers, coupled with the success of these approaches over first-order (or non-relationship) regularizers, leads us to investigate the potential of 'longer-range' relationship regularization. Among this various set of apparently-redundant regularizers, which leads to improved performance? We explore this potential and empirically validate that *explicitly* exploiting rich structural information on non-local relationships improves existing regularization algorithms.

2. Relationship regularization

To begin, we focus on a specific class of relationships and discuss the ideal case where we know M exactly. In Section 2.3, we present a practical algorithm for when M is indirectly represented as a sampled point cloud $\mathcal{X} = \{x_1, \ldots, x_u\}$.

2.1. Class of relationships

In many problems, N has relationship structure that is either canonically specified by the problem or is given implicitly. In classification, the target space is the discrete space of class memberships. In this case, the natural relationship $[\![f(x), f(x')]\!]$ is binary: either *same class* or *different class*. In matching, $[\![f(x), f(x')]\!]$ is either *match* or *no match*.[2] In Markov random fields (MRF), N can be explicitly provided with a pair-wise potential $p: N \times N \to \mathbb{R}$, or an n-ary potential $q: N^n \to \mathbb{R}$ [10]. In many cases, these

[1] A mathematically-precise relationship definition is obtained by equating the relationship with a set function $F: 2^M \to \mathbb{R}$. We do not adopt this definition since we focus on specific relationships instantiated through smooth kernels as defined in Sec. 2.1. In this sense, $[\![A]\!]$ can be identified with a set function defined on singletons, equivalent to a regular function on M.

[2] f may not be explicitly defined as the primary object in the relationship.

relationships represent similarity between pairs or n-tuples of entities; in general, any non-metric relationship can be defined, e.g., *left of* or *on top of* for generating topographic maps.

These relationships can be represented by an n-th order *relationship function* k defined on N^n, where n is application specific. In principle, any relationship function can be regularized; for numerical optimization, we focus on k that is *smooth* wrt. the input arguments (i.e., $k \in C^\infty(N^n)$). Specifically, for semi-supervised learning, we use a Gaussian relationship function k:

$$k(f(x),f(x')) = \exp\left(-\frac{(f(x)-f(x'))^2}{\sigma_f^2}\right) \tag{5}$$

where $\sigma_f^2 > 0$. We assume that $f \in C^\infty(M)$, which we regularize as aided by relationships. We obtain the final class membership $\{-1,1\}$ by thresholding the output space.

2.2. Regularization on relations

Our proposed regularizer assumes the general cases where N is a Riemannian manifold (though many examples, including our demonstrations, are Euclidean in N). First, we discuss a straightforward approach which is not computationally practical for large problems. Then, we develop this intuition further to arrive at a computationally-affordable solution.

We construct the regularizer of f based on the regularization of relationship k on the evaluations of f. First, we construct the *pullback function* [11] f^*k of k based on f:

$$f^*k(x,x') := k(f(x),f(x')). \tag{6}$$

This operation casts k, originally defined on N^2, into a function defined on M^2 so that it can be regularized based on the differential structure on M^2: Since $f^*k \in C^\infty(M^2)$, we can immediately extend the harmonic energy $\mathcal{R}^{\mathrm{h}}$ and the p-iterated Laplacian semi-norm $\mathcal{R}^{\mathrm{p}}$ as defined now on M^2 by noting that f^*k can be regarded as a single-argument function on the product manifold M^2: 1) The tangent space for the point (x,x') is defined based on the direct sum: $T_{(x,x')} := T_x \oplus T_{x'}$; 2) The Riemannian metric is defined by $g_{M^2}(x_1+x_2,x_1'+x_2') := g_M(x_1+x_2)+g_M(x_1'+x_2')$, which fixes the natural volume element $dV(x,x')$; 3) Based on 1) and 2), the differential structure ∇_{M^2} follows naturally from ∇_M.

The resulting new energy is in the same form as $\mathcal{R}^{\mathrm{h}}$ (Eq. 3) except that its domain is now M^2 instead of M:

$$\mathcal{R}_k^{\mathrm{prod}}(f^*k) = \int_{M^2} \|[\nabla f^*k](x,x')\|^2_{T^*_{(x,x')}} dV(x,x'). \tag{7}$$

The biggest obstacle to apply this straightforward construction to semi-supervised learning is its high computational complexity. When approximating $\mathcal{R}^{\mathrm{h}}$ and $\mathcal{R}^{\mathrm{p}}$ based on a sampled point cloud of size u, the corresponding approximations are calculated based on $u \times u$ matrices (Sec. 2.3). For the product manifold M^2, the approximations now require building regularization matrices of size $u^2 \times u^2$, which become infeasible even for moderate u.

Our approach is to make the roles of x and x' asymmetric in the regularization. For a given pair-wise relationship function k, we construct an auxiliary single-argument function h and the corresponding pullback function f^*h as:

$$h_{y'}(y) := k(y,y') \in C^\infty(N), \tag{8}$$

$$f^*h_{x'}(x) := h_{f(x')}(f(x)) \in C^\infty(M). \tag{9}$$

Now, we define new extensions of harmonic energy functional and p-th iterated Laplacian energy functional as:

$$\mathcal{R}_k^{\mathrm{h}}(f) = \int_M \int_M \|\nabla f^*h_{x'}(x)\|^2_{T_x^*} dV(x) dV(x'), \tag{10}$$

$$\mathcal{R}_k^{\mathrm{p}}(f) = \int_M \int_M h_{x'}(x)[\Delta^p f^*h_{x'}(x)] dV(x) dV(x'). \tag{11}$$

For each fixed x' in the function, $f^*h_{x'}(x)$ encodes the relationship between $f(x)$ and $f(x')$, and since $f^*h_{x'}(x)$ is a function of a single variable $x \in M$, $\nabla f^*h_{x'}(x)$ lies in $T_x^*(M)$. This makes the interpretation of Eqs. 10 and 11 also straightforward: the inner integral measures the variation of $f^*h_{x'}(x)$ that corresponds to pair-wise relations between the fixed x' and each value of x. In particular, when $k(a,b)$ measures the Euclidean distance between a and b, the inner integral is zero only when the distances between each pair $[\![f(x),f(x')]\!]$ are identical for all $x \in M$. This does not require that k is zero. Then, the outer integral averages x' over the entire M.

For an n-th order relationship function q, the corresponding $\mathcal{R}_q$'s can be defined similarly through an n-times iterated integration: For each case, a pull-back function similar to $f^*h_{x'}(x)$ is defined as a C^∞ function on M. An important advantage of this asymmetrization is that now the corresponding approximate regularization matrices retain the sizes of $u \times u$ (see Sec. 2.3) and accordingly they afford practical applications.

It should also be noted that currently, our regularizer does not exploit the potential differential structure of the target manifold N. While the differential structure of N is irrelevant in most applications we foresee, for interested readers, we note that in principle, our regularizer can take this structure into account by pulling it back to M, i.e., to use the *pullback connection* $f^*\nabla^N$ [16].

2.3. Approximating $\mathcal{R}_k$ from a sampled point cloud

In many practical applications, M is not directly observed but indirectly represented as a sampled point cloud $\mathcal{X} = \{x_1,\ldots,x_u\}$ and accordingly, we approximate $\mathcal{R}_k$ based on evaluations of f on $\mathcal{X}$. For a given relationship function k, our approximate regularization functional to $\mathcal{R}_k^{\mathrm{h}}$ is defined as:

$$\widetilde{\mathcal{R}_k^{\mathrm{h}}}(\mathbf{f}) = tr[K^\top L K], \tag{12}$$

where $tr[\cdot]$ is the trace, $K_{ij} := k(f(x_i),f(x_j))$, and $L(u \times u)$ is the graph Laplacian:

$$L = D - W, \tag{13}$$

where $W_{ij} = \exp\left(-\frac{\|x_i - x_j\|}{\sigma_x^2}\right)$ when x_i, x_j are k-nearest neighbors and 0 when not, σ_x^2 is a hyper-parameter, and D is a diagonal matrix containing the column sums of W. For exposition, we use the unnormalized graph Laplacian. However, our results straightforwardly extend to normalized graph Laplacian cases, which we use for all experiments (Sec. 4.3).

By noting that the i-th column $K_{[:,i]}$ of K corresponds to a

discrete approximation of $f^* h_{x_i}(\cdot)$, the convergence of $\widetilde{\mathcal{R}^{\mathrm{h}}_k}$ to $\mathcal{R}_k$ can be easily established based on the convergence results of the graph Laplacian to the Laplace-Beltrami operator [2, 6].

Proposition 1. Let M be a connected, compact submanifold of $\mathbb{R}^M$ without boundary and $\mathcal{X}_u = \{x_1,...,x_u\}$ be sampled from a uniform distribution on M. Then, for $f \in C^\infty(M)$ and $k \in C^\infty(N \times N)$ and $\sigma_x^2(u) = u^{-\frac{1}{m+2+\alpha}}$ with $\alpha > 0$,

$$\lim_{u\to\infty} \frac{\widetilde{\mathcal{R}^{\mathrm{h}}_k}(\mathbf{f})}{u^3(\sigma_x^2(u))^{m/2+1}} = \frac{\mathcal{R}^{\mathrm{h}}_k(f)}{V(M)^2}, \tag{14}$$

in probability, where $V(M)$ is the volume of M.

Proof. The proof is similar to that of Theorem 4 by Zhou and Belkin [20]. Since $f \in C^\infty(M)$ and $k \in C^\infty(N \times N)$, $f^* h_{x'} \in C^\infty(M)$. Then, applying the convergence result of graph Laplacian to $f^* h_{x_i}$ for a fixed x_i [2], we have $\forall x_j \in \mathcal{X}$ in probability,

$$\lim_{u\to\infty} \frac{[LK_{[:,i]}]_j}{u(\sigma_x^2(u))^{m/2+1}} = \Delta f^* h_{x_i}(x_j). \tag{15}$$

For Eq. 14, we apply the law of large numbers and then Green's identity [11] for a compact manifold without boundary to Eq. 15:

$$\int_M f \Delta g dV(x) = -\int_M \langle \nabla f, \nabla g \rangle_{T^*_x} dV(x). \quad \square \tag{16}$$

For simplicity, we assume a uniform sample distribution on M. However, this result extends to non-uniform underlying probability distributions P on M via Hein et al. [6]. In this case, the integrand in Eq. 10 is weighted by the corresponding density.

Similarly to $\mathcal{R}^{\mathrm{h}}_k$, the approximate regularization functional to $\mathcal{R}^{\mathrm{p}}_k$ is defined as:

$$\widetilde{\mathcal{R}^{\mathrm{p}}_k}(\mathbf{f}) = tr[K^\top L^p K]. \tag{17}$$

Given Prop. 2.3 conditions, the convergence of $\widetilde{\mathcal{R}^{\mathrm{p}}_k}$ to $\mathcal{R}^{\mathrm{p}}_k$ follows from Eq. 16 and the fact that $\Delta f \in C^\infty(M)$ for $f \in C^\infty(M)$.

3. Semi-supervised learning

Given the two regularizers $\mathcal{R}$ and $\mathcal{R}_k$ (Eqs. 3 and 10 or Eqs. 4 and 11) and the loss function (l; Eq. 1), we state our semi-supervised learning algorithm:

$$\begin{aligned}\mathcal{E}^k(\mathbf{f}) &= (\mathbf{f}-\mathbf{t})^\top H(\mathbf{f}-\mathbf{t}) + \lambda_1 \mathbf{f}^\top G \mathbf{f} + \lambda_2 tr[K^\top G K] \\ &\approx \sum_{i=1,...,s} l(y_i, f(x_i)) + \lambda_1 \mathcal{R}^{\mathrm{h}}(f) + \lambda_2 \mathcal{R}^{\mathrm{h}}_k(f),\end{aligned} \tag{18}$$

where $\mathbf{f} = [f(x_1),...,f(x_u)]^\top$, H is a diagonal matrix, $H_{ii} = 1$ if i-th data point is labeled (0 otherwise), λ_1 and λ_2 are regularization hyper-parameters, and G is L or L^p. For $\mathbf{t}$, if the i-th data point is labeled, $\mathbf{t}_i$ is the corresponding label y_i, or otherwise 0.

While the first two summands in $\mathcal{E}^k$ are convex with respect to $\mathbf{f}$, the third term is non-convex. We minimize $\mathcal{E}^k$ based on conjugate gradient (CG) descent. We set the initial solution $\mathbf{f}^0$ as the minimizer of $\mathcal{E}^k$ with λ_2 held fixed at 0, which can be analytically computed. Hence, the entire optimization process is deterministic.

With the Gaussian relationship function (Eq. 5), the gradient of each summand for the t-th function evaluation is:

$$\frac{\partial(\mathbf{f}-\mathbf{t})^\top H(\mathbf{f}-\mathbf{t})}{\partial \mathbf{f}} = 2H(\mathbf{f}-\mathbf{t}) \tag{19}$$

$$\frac{\partial \mathbf{f}^\top G \mathbf{f}}{\partial \mathbf{f}} = 2G\mathbf{f} \tag{20}$$

$$\frac{\partial tr[K^\top G K]}{\partial \mathbf{f}_t} = 2tr[K^\top G \frac{\partial K}{\partial \mathbf{f}_t}], \tag{21}$$

where $\mathbf{f} = [f(x_1),...,f(x_u)]^\top$ and

$$\frac{\partial K_{ij}}{\partial \mathbf{f}_t} = \begin{cases} -\frac{2(\mathbf{f}_i - \mathbf{f}_j)}{\sigma_f^2} K_{ij} & \text{if } i = t \\ -\frac{2(\mathbf{f}_j - \mathbf{f}_i)}{\sigma_f^2} K_{ij} & \text{else if } j = t. \end{cases} \tag{22}$$

For (binary) classification problems, $y_i \in \{-1,1\}$. In Sec. 3.2, we discuss the dimensionality reduction problem where the output dimensionality n is larger than 1 and accordingly $f(x)$ is a vector.

3.1. Sparsity

Our empirical explicit relationship regularizer enforces smoothness across every possible pairwise evaluation of the function f. This leads to a dense matrix K in Eq. 18. For large-scale problems, we can construct a sparse version of the regularizer by discarding the smoothness enforcement over the relationships that are evaluated for distant points, and focus only on local neighborhoods (not to be confused with the locality of the regularizer, i.e., neighborhood for graph Laplacian):

$$\mathcal{E}^k_S(\mathbf{f}) = \lambda_2 \sum_i \sum_{jk} (K_{ij} - K_{ik})^2 W_{jk} g_{ij} g_{ik}, \tag{23}$$

where $g_{ij} = 1$ if x_i and x_j are in a specified neighborhood $\mathcal{N}_K$ and $g_{ij} = 0$, otherwise. When the neighborhood size is infinite (i.e., $g = 1$), $\mathcal{E}^k_S$ is the same as the original regularizer in Eq. 18. Otherwise, $\mathcal{E}^k_S$ enforces smoothness only for relationships that are defined for function evaluations of close input points.

3.2. Relationship labels and spectral embedding

For some applications, the relationships K themselves are natural variables of interest, and so training labels can be user provided. For instance, in spectral embedding such as for clustering and dimensionality reduction, e.g., in scientific visualization, where $f(x) \in \mathbb{R}^n$ with n being the desired dimensionality, the absolute value of the function f may be irrelevant while the relative *spread* of the data are important. The user might provide expert rules to define which data points should be close to each other (*must-link*) or not (*cannot-link*). We can exploit this by penalizing the deviation of K from the given relationship label T:

$$\mathcal{E}^k_Q(\mathbf{f}) = \|(K - T).Q\|^2_{\mathcal{F}}, \tag{24}$$

where $Q_{ij} = 1$ if the label T_{ij} is provided for a pair (i,j), and $Q_{ij} = 0$ otherwise. $T_{ij} = 1$ when $f(x_i)$ and $f(x_j)$ should be close to each other in the embedding space, and $T_{ij} = 0$ otherwise. $A.B$ is element-wise multiplication of two matrices A and B, and $\|A\|_{\mathcal{F}}$ is the Frobenius norm of A. In this case, our new energy functional is constructed as follows:

$$\mathcal{E}^k(\mathbf{f}) = \|\mathbf{f} - \mathbf{t}\|^2 + \lambda_2 tr[K^\top G K] + \lambda_3 \mathcal{E}^k_Q(\mathbf{f}), \tag{25}$$

where we set the label $\mathbf{t}$ and the initial search solution $\mathbf{f}^0$ of the optimization as the results of standard spectral embedding obtained from a graph Laplacian-based algorithm: $\mathbf{f}^0 = [\mathbf{e}_2,...,\mathbf{e}_n]$ with $\mathbf{e}_i$ being the i-th eigenvector of L. Since each output $f(x)$ is a vector, our relationship function is adapted accordingly:

$$k(f(x),f(x')) = \exp\left(-\frac{\|f(x)-f(x')\|^2}{\sigma_f^2}\right). \quad (26)$$

Minimizing Eq. 24 over $\mathbf{f}$ is different from independently minimizing it for each output dimension since the outputs are tied across the dimensions through the relationship labels (Eq. 25), and the regularizer (Eq. 12) is truly vector valued.

4. Experiments

We compare the performance of our explicit relationship regularization (ERR, Eqs. 10 and 11) by adapting two existing implicit relationship regularizations (IRR, Eqs. 3 and 4): classic graph Laplacian [2] and state-of-the-art iterated graph Laplacian [20]. To our knowledge, no algorithms exist which attempt to explicitly regularize relationships, even though they may implicitly attempt to do so (Sec. 1.1). The purpose of our experiments is to show the improvement that can come from explicit relationship regularization, using standard and state-of-the-art approaches as evidence. As such, we conducted a semi-supervised learning experiment for pattern classification with a set of standard machine learning databases. Code will be made available on the web.

4.1. Semi-supervised classification

We use seven standard binary classification datasets for semi-supervised learning covering image digits (USPS), EEG signals (BCI), newsgroup categories (Text, Pcmac, Real-sim) and news reports (CCAT, GCAT) [4, 20]. We randomly divide each dataset into three subsets: 50 labeled data points, 50 data points for validation for hyper-parameter selection, and the remaining unlabeled data points are used for evaluation. We average error rates for 10 experiments with different sets of labeled examples. To demonstrate sparsity for large datasets (Sec. 3.1), we use the 60,000 point large MNIST dataset, with binary labels obtained in the same way as for the USPS dataset [4]. Here, $|\mathcal{N}_K| = 200$, while the number of labeled and validation data points were fixed at 300 each. Due to the large size of the problem, the iterated graph Laplacian was not applicable for neither IRR nor ERR since taking the power of a sparse (Laplacian) matrix tends to produce a denser matrix.

Binary classification allows direct comparison of regularization performance and disregards multi-class combination method effects. However, to gain an insight into multi-class classification performance, we performed experiments with a 10-class dataset of 2,000 data points sampled from MNIST. For training and validation, we used 50 labels for each class. To facilitate representing the multi-class outputs, we learn a vector-valued function f and the corresponding relationship function k as defined in Eq. 26.

For IRR, there are three parameters: σ_x^2, k_N, the k-nearest neighborhood size for the graph Laplacian construction (Eq. 13), and regularization parameter λ_1. For ERR (Eq. 10), there are two more to be tuned: σ_f^2 for the Gaussian similarity relationship function k (Eq. 5), and regularization parameters λ_2. We first find bounds for σ_x^2, k_N, and λ_1 around the optimal for IRR; then, we optimize σ_f^2 and λ_2 for ERR. This resulted in the total number of parameter evaluations for ERR being only slightly larger than that of IRR. For $\mathcal{R}_k^{\mathrm{p}}$ (Eq. 11) there is an additional hyper-parameter p that we fix at 2 throughout the entire set of experiments.

Performance For all but one dataset, the error rate of ERR was lower than that of IRR when parameters were automatically chosen (Table 1). This demonstrates the possible improvement of ERR over IRR and supports our claim that explicitly exploiting relationship information is useful. However, automatically optimizing the parameters with a limited number of labeled points can lead to overfitting (as observed in worse performance for ERR on BCI). Automatic tuning of hyper-parameters is still an open problem in semi-supervised learning where only a limited number of labeled examples are provided.

We also report the performance of both algorithms when best-case (BC) hyper-parameters are provided (odd row blocks), and the performance difference between ERR and IRR is more pronounced. This indicates that ERR can potentially lead to larger improvements over IRR when the parameters are tuned properly (e.g., through user interaction). If the error rate surface with respect to the hyper-parameters is *smooth*, then the user could decide the next search point based on the information gathered thus far. Our preliminary experiments showed that the error rate surface with respect to hyper-parameter *is* smooth. Accordingly, the active sampling strategy can indeed be exercised (Table 1).

4.2. Spectral embedding

Our algorithm is a general regularizer for Riemannian manifolds, and also supports explicit relationship labels. We use dimensionality reduction and clustering applications to show this with MNIST, full USPS, and standard UCI clustering datasets (Diabetes, Iris, Wine, Breast Cancer Wisconsin (BCW), and Pendigits). *Must-link* and *cannot-link* labels are based on ground truths for selected pairs. Note that relationship labels are *weak* in that having a positive or negative label T_{ij} for a pair f_i and f_j does not reveal the corresponding class information for either y_i or y_j.

In general, for unsupervised learning such as clustering and dimensionality reduction, automatic tuning of hyper-parameters is infeasible as there is no ground-truth information. Following experimental convention [3], we set $k_N = 10$ and σ_x^2 adaptively based on the average Euclidean distance of a point to its k_N neighbors. In practice, the remaining hyper-parameters should be user tuned. To facilitate this process, we reduce the number of hyper-parameters to two, by first setting $\lambda_1 = 0$ (see Eq. 25) and tying λ_2 and λ_3 by a new parameter λ_2': We set the weight λ_3 of relationship labels at a relatively large value 10 as these user labels should be regarded as quasi-hard constraints. The overall contribution of the s_R relation labels is controlled by λ_2', replacing λ_2 by λ_2/s_R. Figure 2 shows that parameter tuning is feasible as performance varies smoothly with respect to the parameter space.

Again, while the hyper-parameters might be tuned based on user inspection in practice, to facilitate numerical evaluation for each dataset we randomly selected $s_R = 250$ labels and optimized

Table 1. Classification performance as error rate for implicit and explicit relationship regularization (IRR and ERR), versus both graph Laplacian ($\widetilde{\mathcal{R}}_k^{\mathrm{h}}$) and iterated graph Laplacian ($\widetilde{\mathcal{R}}_k^{\mathrm{p}}$) regularizers, with added best-case parameters (BC; Sec. 4.1). Bold marks the best results. The performance improvement of ERR over IRR is calculated as the reduction of error rate (RER) in %.

		USPS	Text	BCI	Pcmac	Real-sim	CCAT	GCAT	MNIST	MNIST (multi-class)
Graph Laplacian $\widetilde{\mathcal{R}}_k^{\mathrm{h}}$	IRR	10.81	43.13	**42.98**	14.97	15.48	26.08	12.61	10.43	8.72
	ERR	**6.76**	**35.13**	43.38	**11.62**	**12.71**	**25.92**	**12.16**	**5.24**	**7.03**
	RER (%)	37.46	18.55	-0.93	22.38	17.89	0.06	3.57	49.79	19.38
	IRR (BC)	9.59	37.91	40.03	13.61	14.32	20.80	8.90	8.68	7.04
	ERR (BC)	**4.44**	**22.39**	**38.95**	**8.90**	**10.23**	**19.63**	**8.39**	**4.90**	**6.14**
	RER (%)	53.70	40.94	2.70	34.61	28.56	5.63	5.73	43.58	12.78
Iterated Graph Laplacian $\widetilde{\mathcal{R}}_k^{\mathrm{p}}$	IRR	4.80	29.05	**41.74**	11.95	12.36	24.20	10.97	N/A as matrix too dense	7.46
	ERR	**3.71**	**23.84**	42.35	**10.38**	**11.52**	**21.31**	**9.48**		**6.74**
	RER (%)	22.71	17.94	-1.46	13.14	6.80	11.94	9.75		9.72
	IRR (BC)	3.77	24.40	38.18	10.07	11.35	18.94	7.99		6.79
	ERR (BC)	**2.33**	**22.21**	**37.58**	**7.51**	**9.68**	**16.70**	**7.26**		**6.14**
	RER (%)	38.20	8.98	1.57	25.42	14.71	11.83	9.14		9.65

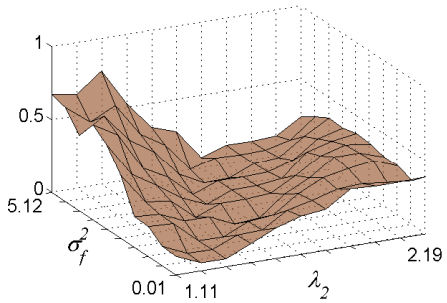

Figure 2. Clustering performance (error rate) of the proposed algorithm on USPS dataset with hyper-parameters σ_f^2 and λ_2 ($\lambda_2' * S_R$) that vary in multiplicative intervals 2 and 3, respectively.

σ_f^2 and λ_2' based on their respective ground-truth error measures (Sec. 4.2.1). These parameter values are fixed across *all* s_R values. For each value of s_R, we randomly sampled half the number of must-link and cannot-link labels, averaging error rates across 10 experiments. For comparison, we tuned the hyper-parameters of all competing algorithms (as described shortly) for each dataset and for each value of s_R, based on the ground-truth error rate, which is an advantage over our fixed parameters across s_R values.

4.2.1 Clustering

From the optimized $\mathbf{f}^*$, the final cluster label is assigned to each data point by applying k-means clustering on $\mathbf{f}^*$. Since k-means optimization is non-convex, we run it ten times with random initialization and choose the result that minimizes the *normalized cut (NCut)* [3] as it can be calculated without requiring any labels. We compare with the original spectral clustering, and three state-of-the-art algorithms which exploit explicit relationship labels: Constrained Clustering via Spectral Regularization (CCSR) [12] and Flexible Constrained Spectral Clustering (CSP) [18] both optimize spectral energy ($\mathcal{R}$) but under hard and soft constraints respectively (must-link and cannot-link), while Constrained 1-Spectral Clustering (COSC) [14] minimizes a continuous ($L1$) relaxation of the NCut under the same constraints. These algorithms significantly outperform existing (relationship-) constrained approaches, as well as unconstrained clustering algorithms [12, 14, 18].

One major difference between those algorithms and ours is that they regularize $\mathbf{f}$ with constraints, while our algorithm explicitly regularizes relationships. We also compare with the more classical Spectral Learning algorithm (SL) that encodes the constraints into the weight matrix in building the graph Laplacian [7]. For CCSR and CSP, we used the code provided by the authors on their websites. Since CSP is designed for binary clustering, we only report the corresponding results of binary datasets (Diabetes, BCW). The clustering error is defined by summing the occurrences of errors for each cluster: a data points is counted as an error if its label is different from the dominant label of the cluster to which it belongs.

Performance All algorithms that exploit relationship labels significantly improved over original spectral clustering (Table 2). The CSP and CCSR were especially good for BCW when the number of labels s_R is small. However, they failed to show steady performance increases as s_R increases. Further, for Diabetes, both algorithms showed much higher error rates than other algorithms. On average, SL showed better performance over CSP and CCSR. However, for some datasets, it showed significant error rate increases when s_R is too large, which shows application limitation. Overall, COSC and our algorithm (ERR) demonstrated steady decreases of error rates as s_R increases. However, except for one case (BCW for $s_R = 500$), our algorithm outperformed COSC by a large margin. For USPS, the error rates of COSC stayed high even when $s_R = 1,000$: in the original spectral clustering result, multiple classes are merged into a single cluster, which leads to a single class dominating in multiple clusters. Classes 1 and 4 dominated in two clusters, respectively, and accordingly, classes 6 and 10 are absorbed. While ERR restored all classes when $s_R = 500$, COSC failed even when $s_R = 1{,}000$.

4.2.2 Dimensionality reduction

The target dimensionality n was set at 2 for all experiments, e.g., for visualization applications, though any dimensionality is possible. We measured the error rate based on leave-one-out 1-nearest

Table 2. Clustering performance as error rate for different constrained clustering algorithms.

# labels (s_R)		Diabetes	BCW	USPS	MNIST	Iris	Wine	Pendigits
	Original	23.25	33.02	34.77	13.05	29.71	34.96	29.89
50	CSP	**30.21**	3.25		N/A — CSP is binary only			
	SL	34.80	34.99	18.96	30.72	1.80	32.64	15.69
	CCSR	30.99	**2.75**	47.55	59.20	2.27	29.49	18.78
	COSC	33.58	9.59	18.01	24.04	5.27	36.57	19.67
	ERR	33.50	6.34	**13.27**	**19.88**	**1.53**	**21.52**	**12.28**
100	CSP	31.08	5.24		N/A — CSP is binary only			
	SL	34.01	32.11	18.11	29.16	1.47	23.65	14.23
	CCSR	29.26	**2.77**	37.78	47.19	2.07	29.04	17.41
	COSC	32.15	5.39	18.32	25.91	1.67	29.61	13.75
	ERR	**27.85**	3.95	**12.40**	**17.85**	**0.87**	**9.89**	**8.60**
250	CSP	29.91	2.99		N/A — CSP is binary only			
	SL	28.26	12.91	**5.17**	17.39	0.13	2.42	6.37
	CCSR	29.05	2.78	20.84	34.69	2.00	28.65	13.52
	COSC	12.38	0.92	18.12	19.60	0.13	4.27	3.13
	ERR	**12.36**	**0.64**	10.17	**15.20**	**0.00**	**0.45**	**1.65**
500	CSP	28.19	3.05		N/A — CSP is binary only			
	SL	17.77	6.25	8.24	12.98	**0.00**	**0.00**	5.81
	CCSR	28.98	2.87	16.16	28.86	2.07	27.87	12.79
	COSC	2.84	**0.13**	17.30	13.49	**0.00**	0.06	1.12
	ERR	**1.86**	0.15	**5.14**	**12.83**	**0.00**	**0.00**	**1.09**
1,000	CSP	26.43	2.80		N/A — CSP is binary only			
	SL	1.54	0.44	15.40	24.67	**0.00**	**0.00**	28.24
	CCSR	29.34	2.97	11.69	23.96	1.93	27.02	12.29
	COSC	0.39	**0.00**	10.63	9.79	**0.00**	**0.00**	0.76
	ERR	**0.04**	**0.00**	**3.45**	**7.67**	**0.00**	**0.00**	**0.67**

neighbor classification: For each point, we find its nearest neighbor and use the corresponding retrieved class label as the predicted label and measured the error rate. For comparison, we show the results of CCSR and SL. While both CCSR and SL were originally developed for clustering, they first perform spectral embedding to a given target dimension and then apply conventional clustering therein. Their embedding parts can be used for dimensionality reduction by choosing the target dimension accordingly.

Performance All algorithms improve over the original spectral dimensionality reduction (Table 3), demonstrating the utility of relationship labels. CCSR was especially good for BCW, but it did not show noticeable improvement as s_R increases. ERR and SL both showed steady error rate decreases while ERR significantly outperformed SL, demonstrating the utility of explicit relationship regularization. Figure 3 shows an example embedding.

4.3. Complexity

For all experiments, following conventions, the graph Laplacians are normalized. We set the number of conjugate gradient (CG) steps to 50. This provides a moderate trade-off between the performance and accuracy: While we observed a steady increase in accuracy as the number of CG steps increased for pattern classification experiments, the rate of increase dropped significantly past 50. As indicated by the form of the energy functional (Eq. 12), when sparsity in relationships is not enforced (see Eq. 23), the time complexity of each gradient step is cubic in the number of data points. For pattern classification experiments with the USPS dataset (with 1500 data points), it took approximately 1.6 seconds for 50 CG step on NVIDIA GeForce 680 GPU, and 25 seconds on Intel Xeon 3.6GHz CPU; while the IRR took approximately 0.3 seconds on the same CPU: IRR can be solved analytically, while ERR must be solved iteratively.

Table 4. Performance vs. sparsity ($|\mathcal{N}_K|$) for MNIST subsets ($s=100, u=2{,}000$). GPU optimization negates the need for sparsity for these problem sizes.

$\lvert\mathcal{N}_K\rvert$	25	50	100	200	full ERR	IRR
Error (%)	9.76	9.08	8.64	8.02	7.82	10.10
Time CPU (sec.)	3	10	21	38	35	1
Time GPU (sec.)			-		3	-

4.4. Sparsity

To gain an insight into the sparsity/performance trade-off, we performed experiments on a small subset ($u = 2{,}000$) of the MNIST dataset such that direct performance comparison with dense regularization is possible (Table 4). Performance degrades gracefully as $|\mathcal{N}_K|$ decreases. For this small dataset, the processing time of the sparse system when $|\mathcal{N}_K| = 200$ is longer than the full ERR due to the sparsification overhead. However, the complexity grows roughly linearly with respect to u, and thus sparsity makes ERR applicable to large-scale datasets. In Table 1, we show the results of the full MNIST dataset with $|\mathcal{N}_K| = 200$.

5. Discussion

We have only evaluated the binary relationship function k with the single parameter σ_f^2, and different potential relationship function types could be explored. Further, we have only investigated

Table 3. Leave-one-out classification performance as error rate for different dimensionality reduction algorithms.

# labels (s_R)		Diabetes	BCW	USPS	MNIST	Iris	Wine	Pendigits
	Original	46.35	9.37	29.29	34.48	4.67	28.09	15.92
50	SL	39.40	6.50	28.68	33.29	3.60	31.35	12.88
	CCSR	36.59	**3.91**	42.62	33.70	2.73	34.55	9.04
	ERR	**33.95**	4.77	**5.34**	**23.29**	**3.07**	**24.49**	**2.80**
100	SL	37.49	7.13	27.93	33.15	3.67	30.28	13.04
	CCSR	37.21	**4.04**	38.39	34.98	2.80	33.60	8.81
	ERR	**30.63**	4.10	**5.35**	**22.41**	**2.33**	**16.74**	**3.07**
250	SL	36.93	7.10	25.85	31.05	1.87	20.34	11.45
	CCSR	37.38	4.09	29.24	37.43	3.33	33.09	8.92
	ERR	**24.92**	**3.41**	**5.30**	**10.43**	**0.93**	**9.38**	**2.60**
500	SL	24.88	3.63	22.48	27.02	0.60	2.58	10.82
	CCSR	37.72	4.07	32.27	46.42	3.33	31.57	9.02
	ERR	**16.39**	**1.65**	**5.11**	**6.62**	**0.27**	**0.90**	**2.58**
1,000	SL	11.78	1.39	17.25	22.68	**0.00**	0.11	10.03
	CCSR	38.06	4.04	36.93	47.25	3.33	31.35	9.53
	ERR	**9.53**	**0.79**	**4.90**	**6.31**	**0.00**	**0.00**	**2.20**

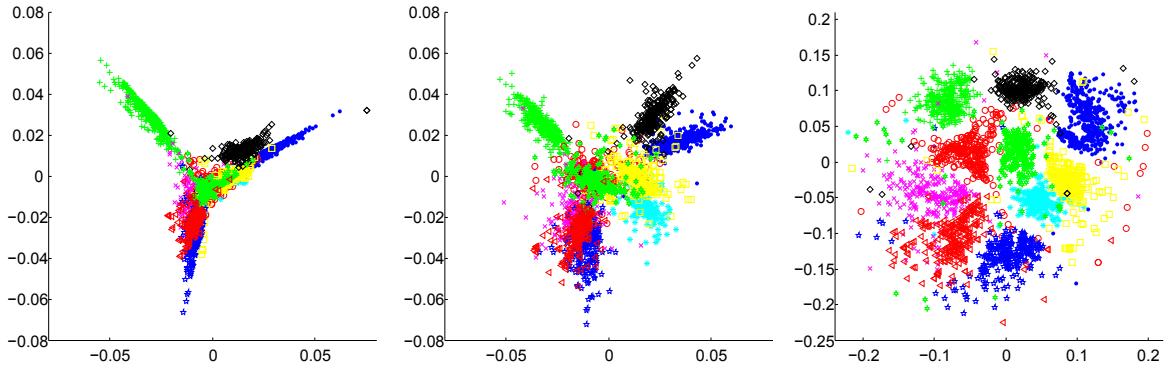

Figure 3. Embedding results for full 10-class USPS dataset ($s_R = 100$); plots show only 2,000 data points for better visibility. *Left:* Spectral embedding ($\mathbf{t}$ in Eq. 25). *Middle:* Minimizing 1) deviation from $\mathbf{t}$; 2) training error for relationship labels (Eq. 24), and 3) conventional graph Laplacian regularization energy ($\mathcal{E}_M^k$ and $\mathcal{R}$: Eqs. 24 and 18 with $\lambda_2 = 0$). *Right:* Our proposal ($\mathcal{E}_M^k$ and $\mathcal{R}_k$: Eq. 25). Error rates (left to right): 28.30, 27.53, and 0.63.

binary relationship functions, and n-ary relationship functions are possible. In this case, the K matrix in Eq. 18 is replaced by a tensor, and the problem complexity increases, though it may still be possible to handle these cases by enforcing sparsity (Sec. 3.1).

For the specific case of binary relationship functions regularized by the graph Laplacian (which corresponds to pair-wise regularization), our regularization energy functional (Eq. 23) can be regarded as a construction of a ternary relationship function: One can define a ternary clique as a summand of Eq. 23:

$$q(f_i, f_j, f_k) = (K_{ij} - K_{ik})^2 W_{jk} g_{ij} g_{ik}. \quad (27)$$

In this way, our algorithm can be viewed as a special case of an MRF. While, in general, the optimization with a ternary relationship function is computationally very demanding, the asymmetric roles of three arguments in our clique (see the last paragraph of Sec. 2.2) leads to a computationally affordable algorithm. In this respect, one of our main contributions is a method to construct a high-order clique from low-order cliques and the corresponding practical algorithm for semi-supervised learning.

In our semi-supervised learning experiments, we chose hyper-parameters based on separate validation sets. Heuristics can help set some hyper-parameters, e.g., for spectral embedding, we set σ_x^2 based on the average Euclidean distance of a point to its k_N neighbors (Sec. 4.2). For USPS, the corresponding average clustering error rate was around 20% higher than when varying and manually selecting σ_x^2. This suggests that the heuristic can trade accuracy with hyper-parameter optimization time.

6. Conclusion

We have investigated *explicit relationship regularization*, which, in addition to regularizing the function in semi-supervised learning, now regularizes the relationships between function evaluations through smooth relationship functions. This approach improves performance by a large margin in semi-supervised classification and in constrained spectral clustering applications, and facilitates a related algorithm in semi-supervised dimensionality reduction. We believe semi-supervised learning and constrained clustering algorithms will increase in importance in vision, e.g., recent works in pose estimation [17], and video segmentation [8]. Future work should consider what role our explicit relationship regularization plays on the effect of the statistical model, e.g., error bound.

Acknowledgements

Kwang In Kim thanks EPSRC EP/M00533X/1 and EP/M006255/1, James Tompkin and Hanspeter Pfister thank NSF CGV-1110955, and James Tompkin and Christian Theobalt thank the Intel Visual Computing Institute.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *Proc. ICCV*, 2009. 1

[2] M. Belkin and P. Niyogi. Towards a theoretical foundation for Laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2005. 2, 4, 5

[3] T. Bühler and M. Hein. Spectral clustering based on the graph p-Laplacian. In *Proc. ICML*, pages 81–88, 2009. 5, 6

[4] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006. 2, 5

[5] S. Ebert, D. Larlus, and B. Schiele. Extracting structures in image collections for object recognition. In *Proc. ECCV*, pages 720–733, 2010. 1

[6] M. Hein, J.-Y. Audibert, and U. von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proc. COLT*, pages 470–485, 2005. 4

[7] S. D. Kamvar, D. Klein, and C. D. Manning. Spectral learning. In *Proc. IJCAI*, pages 561–566, 2003. 6

[8] A. Khoreva, F. Galasso, M. Hein, and B. Schiele. Learning must-link constraints for video segmentation based on spectral clustering. In *Proc. GCPR*, pages 701–712, 2014. 8

[9] K. I. Kim, J. Tompkin, M. Theobald, J. Kautz, and C. Theobalt. Match graph construction for large image databases. In *Proc. ECCV*, pages 272–285, 2012. 1

[10] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289, 2001. 2

[11] J. M. Lee. *Riemannian Manifolds- An Introduction to Curvature*. Springer, New York, 1997. 2, 3, 4

[12] Z. Li, J. Liu, and X. Tang. Constrained clustering via spectral regularization. In *Proc. CVPR*, pages 421–428, 2009. 1, 6

[13] M. McPherson, L. Smith-Lovin, , and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001. 1

[14] S. Rangapuram and M. Hein. Constrained 1-spectral clustering. *JMLR W&CP (Proc. AISTATS)*, 22:1143–1151, 2012. 1, 6

[15] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002. 2

[16] F. Steinke, M. Hein, and B. Schölkopf. Nonparametric regression between general Riemannian manifolds. *SIAM Journal on Imaging Sciences*, 3(3):527–563, 2010. 2, 3

[17] D. Tang, T.-H. Yu, and T.-K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression rorests. In *Proc. ICCV*, pages 3224–3231, 2013. 1, 8

[18] X. Wang and I. Davidson. Flexible constrained spectral clustering. In *Proc. SIGKDD*, pages 563–572, 2010. 1, 6

[19] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, pages 1330–328, 04. 1

[20] X. Zhou and M. Belkin. Semi-supervised learning by higher order regularization. *JMLR W&CP (Proc. AISTATS)*, pages 892–900, 2011. 2, 4, 5

[21] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proc. ICML*, pages 912–919, 2003. 1