Semisupervised Neural Networks for Efficient Hyperspectral Image Classification

Frédéric Ratle, Gustavo Camps-Valls, *Senior Member, IEEE*, and Jason Weston

***Abstract*—A framework for semisupervised remote sensing image classification based on neural networks is presented. The methodology consists of adding a flexible embedding regularizer to the loss function used for training neural networks. Training is done using stochastic gradient descent with additional balancing constraints to avoid falling into local minima. The method constitutes a generalization of both supervised and unsupervised methods and can handle millions of unlabeled samples. Therefore, the proposed approach gives rise to an operational classifier, as opposed to previously presented transductive or Laplacian support vector machines (TSVM or LapSVM, respectively). The proposed methodology constitutes a general framework for building computationally efficient semisupervised methods. The method is compared with LapSVM and TSVM in semisupervised scenarios, to SVM in supervised settings, and to online and batch k-means for unsupervised learning. Results demonstrate the improved classification accuracy and scalability of this approach on several hyperspectral image classification problems.**

***Index Terms*—Graph Laplacian, hyperspectral image classification, Laplacian support vector machine (LapSVM), neural networks, regularization, semisupervised learning (SSL), support vector machine (SVM), transductive SVM (TSVM).**

I. Introduction

REMOTE sensing image classification is a challenging task because only a small number of labeled pixels are typically available, and thus, classifiers tend to overfit the data [1]. In recent years, supervised kernel classifiers, such as support vector machines (SVMs) [2], have demonstrated very good performance in multispectral, hyperspectral, and multisource image classification [3]–[5]. However, when little labeled information is available, the underlying probability distribution function of the image is not properly captured, and a risk of overfitting still persists.

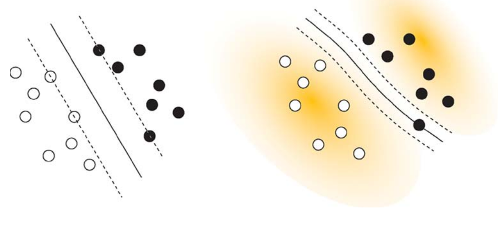
Fig. 1. (Left) Classifier obtained using labeled data. (Right) Classifier obtained using labeled data plus unlabeled data distribution (shaded).

In this context, semisupervised learning (SSL) naturally appears as a promising tool for combining labeled and unlabeled information, thus increasing the accuracy and robustness of class predictions [6], [7]. The traditional SSL methods are based on generative models, which estimate the conditional density, and have extensively been applied in remote sensing image classification [8]. Recently, more attention has been paid to *discriminative* approaches, such as 1) the transductive SVM (TSVM) [9], [10], which simultaneously maximizes the margin for labeled and unlabeled samples; 2) graph-based methods, in which each pixel spreads its label information to its neighbors until a global steady state is achieved on the whole image [11], [12]; and 3) the Laplacian SVM (LapSVM) [13]–[15], which deforms the kernel matrix of a standard SVM with the relations found by building the graph Laplacian. Fig. 1 illustrates a typical SSL situation.

Despite the good performance of these kernel methods, some shortcomings are observed. For example, the TSVM does not constitute a convex problem, and thus, several alternative formulations have been proposed, usually introducing additional free parameters to tune. The graph-based methods based on label propagation rely on inverting a huge matrix of the side size of the labeled and unlabeled pixels. Although efficient methods for matrix inversion can be applied, such as the Nyström method or the incomplete Cholesky factorization, the computational cost is still prohibitive for operational remote sensing applications. The LapSVM constitutes a reasonable tradeoff between accuracy and cost, generalizes the SVM and other clustering and graph-based methods, and provides a final prediction model. Nevertheless, none of these methods can handle millions of unlabeled pixels in a timely manner. This is certainly the main Achilles' heel of any kernel method and particularly relevant in SSL since performance is usually improved when the number of unlabeled samples used is large [16]. In fact, one should ideally use all image pixels as unlabeled samples to properly model the underlying data distribution. We should stress here that, for other standard discriminative machine learning tools, like classification trees or artificial

Manuscript received December 15, 2008; revised May 8, 2009 and September 8, 2009. First published February 17, 2010; current version published April 21, 2010. This work was supported in part by the Spanish Ministry of Education and Science under Project TEC2009-13696, Project CSD2007-00018, and Project AYA2008-05965-C04-03 and in part by the Swiss National Science Foundation under Grant 105211-115965.

F. Ratle was with the Institute of Geomatics and Analysis of Risk, University of Lausanne, 1015 Lausanne, Switzerland. He is now with Nuance Communications International BVBA, 9820 Merelbeke, Belgium (e-mail: frederic.ratle@gmail.com).

G. Camps-Valls is with the Department Enginyeria Electrònica Escola Tècnica Superior d'Enginyeria, Universitat de València, 46100 Burjassot (València), Spain (e-mail: gustavo.camps@uv.es).

J. Weston was with NEC Labs America, Princeton, NJ 08540 USA. He is now with Google Research, New York, NY 10011 USA (e-mail: jweston@google.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2009.2037898

neural networks [17], [18], this is not a problem. This paper is concerned with extending Laplacian and transductive methods to make them fully operational.

In the last years, kernel methods have progressively replaced neural networks as the state-of-art classification methods, mainly because of 1) the ease of interpretation of the free parameters; 2) training is made by solving a convex optimization problem (i.e., no local minima); 3) very good performance on high-dimensional problems is observed; and 4) the ability to handle noisy data sets. However, when confronted with "large-scale" problems, e.g., very large data sets or images, kernel methods (either supervised or semisupervised) tend to become difficult to use. Indeed, methods to solve convex programs do not scale well, i.e., they are at least quadratic in the number of samples used. Online schemes for training SVMs have been proposed elsewhere [19], [20], but they are too complicated for a nonexpert user. This is even more true in the context of kernel-based SSL, where the resulting loss function is hard to optimize with an online algorithm. Furthermore, the number of kernels can possibly grow as more data are collected; thus, new points can become support vectors, which increase the computational cost for both training and testing.

In this paper, we advocate the use of *semisupervised neural networks* (SSNNs) to deal with large-scale remote sensing classification problems. The methodology consists of adding a regularizer to the loss function used for training neural networks, similar to LapSVM or TSVM. The stochastic gradient descent (SGD) is modified with additional balancing constraints to avoid falling into local minima, which is a common problem with transduction schemes. The proposal constitutes a general framework for building computationally efficient semisupervised methods, and it generalizes other supervised and unsupervised methods. We compare the results to TSVM and LapSVM, as well as purely supervised and unsupervised algorithms, and demonstrate the improved classification accuracy and scalability of this approach on several hyperspectral images.

The rest of this paper is outlined as follows: Section II fixes notation and briefly revises the main concepts and properties of the manifold-based regularization framework for SSL. We pay attention to the basic elements of the LapSVM, which is particularly relevant for the exposition of the proposed algorithm. Section III presents the proposed method for semisupervised image classification. Section IV describes the data collection, the experimental setup, and the obtained results. Finally, Section V concludes with some remarks and further research directions.

II. Manifold-Based Regularization Framework

Regularization helps to produce smooth decision functions that avoid overfitting to the training data. Since the work of Tikhonov [21], many regularized algorithms have been proposed to control the capacity of the classifier [2], [22]. Regularization has been applied to both linear and nonlinear algorithms in the context of remote sensing image classification and becomes strictly necessary when few labeled samples are available compared with the high dimensionality of the problem. For example, linear discriminant algorithms need to be regularized to obtain a proper estimation of the sample covariance matrix [23], [24]. In the case of neural networks, different forms of regularization are typically applied (growing, pruning, or weight decay) to prevent the network from falling in local minima and overfitting the training data [18], [25]. In the last decade, the most paradigmatic case of regularized nonlinear algorithm is the SVM: in this case, maximizing the margin is equivalent to applying a kind of regularization to model weights [2], [3]. These regularization methods are particularly appropriate when a low number of samples is available but are not concerned on the geometry of the marginal data distribution. This has recently been treated within a more general regularization framework that includes Tikhonov's as a special case. In this section, we first review the basics of this framework and fix notation. Then, an algorithmical example—the LapSVM—is given, and a related case—the TSVM—is revised in this framework. We end the section by pointing out some shortcomings of this approach, which will be solved by our proposal in the following section.

A. *Semisupervised Regularization Framework*

The classical regularization framework has recently been extended to the use of unlabeled samples [14] as follows: Notationally, we are given a set of l labeled samples $\{\mathbf{x}_i\}_{i=1}^{l}$ with corresponding class labels y_i and a set of u unlabeled samples $\{\mathbf{x}_i\}_{i=l+1}^{l+u}$, where $\mathbf{x}_i \in \mathbb{R}^N$, and $y_i \in \{-1, +1\}$. In our case, $\mathbf{x}_i$ represents the N-band spectrum of pixel i. Let us now assume a general-purpose decision function f. The regularized functional to be minimized is defined as

$$\mathcal{L} = \frac{1}{l}\sum_{i=1}^{l} V(\mathbf{x}_i, y_i, f) + \gamma_L \|f\|_{\mathcal{H}}^2 + \gamma_M \|f\|_{\mathcal{M}}^2 \qquad (1)$$

where V represents a generic cost function of the committed errors on the labeled samples, γ_L controls the complexity of f in the associated Hilbert space $\mathcal{H}$, and γ_M controls its complexity in the intrinsic geometry of the marginal data distribution. For example, if the probability distribution is supported on a low-dimensional manifold, $\|f\|_{\mathcal{M}}^2$ penalizes f along that manifold $\mathcal{M}$. Note that this SSL framework allows us to develop many different algorithms just by playing around with the loss function V and the regularizers $\|f\|_{\mathcal{H}}^2$ and $\|f\|_{\mathcal{M}}^2$.

B. *LapSVM*

Here, we briefly review the LapSVM as an instantiation of the previous framework. More details can be found in [13], [14], and its application to remote sensing data classification in [15].

1) Cost Function of the Errors: The LapSVM uses the same hinge loss function as the traditional SVM, i.e.,

$$V(\mathbf{x}_i, y_i, f) = \max\left(0, 1 - y_i f(\mathbf{x}_i)\right) \qquad (2)$$

where f represents the decision function implemented by the selected classifier, and the predicted labels are $y_* = \operatorname{sgn}(f(\mathbf{x}_*))$. Hereafter, unlabeled or test samples are highlighted with $*$.

2) Regularized Decision Function: The decision function used by the LapSVM is $f(\mathbf{x}_*) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}_*)\rangle + b$, where $\boldsymbol{\phi}(\cdot)$ is a nonlinear mapping to a higher-dimensional Hilbert space $\mathcal{H}$, and $\mathbf{w}$ and b define a linear decision function in that space. The decision function is given by $f(\mathbf{x}_*) = \sum_{i=1}^{l+u} \alpha_i K(\mathbf{x}_i, \mathbf{x}_*) + b$, where $\mathbf{K}$ is the kernel matrix formed by kernel functions $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j)\rangle$. The key point here is that, without explicitly considering the mapping $\boldsymbol{\phi}$, a nonlinear classifier can be constructed by selecting the proper kernel. The regularization term can fully be expressed in terms of the corresponding kernel matrix and the expansion coefficients $\boldsymbol{\alpha}$ as

$$\|f\|^2_{\mathcal{H}} = \|\mathbf{w}\|^2 = (\boldsymbol{\Phi}\boldsymbol{\alpha})^\top(\boldsymbol{\Phi}\boldsymbol{\alpha}) = \boldsymbol{\alpha}^\top\mathbf{K}\boldsymbol{\alpha}. \tag{3}$$

3) Manifold Regularization With LEs: Essentially, for manifold regularization, the LapSVM relies on Laplacian eigenmaps (LEs), which try to map nearby inputs (pixels) to nearby outputs (corresponding class labels), thus preserving the neighborhood relations between samples.[1] Therefore, the geometry of the data is modeled with a graph in which nodes represent both labeled and unlabeled samples connected by weights W_{ij} [6]. Regularizing the graph follows from the *smoothness* (or *manifold*) assumption and intuitively is equivalent to penalize the "rapid changes" of the classification function evaluated between close samples in the graph as

$$\|f\|^2_{\mathcal{M}} = \frac{1}{(l+u)^2}\sum_{i,j=1}^{l+u} W_{ij}(f(\mathbf{x}_i) - f(\mathbf{x}_j))^2 = \frac{\mathbf{f}^\top\mathbf{L}\mathbf{f}}{(l+u)^2} \tag{4}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian, whose entries are sample and graph dependent $L(f(\mathbf{x}_i), f(\mathbf{x}_j), W_{ij})$; $\mathbf{D}$ is the diagonal degree matrix of $\mathbf{W}$ given by $D_{ii} = \sum_{j=1}^{l+u} W_{ij}$ and $D_{ij} = 0$ for $i \neq j$; the normalizing coefficient $1/(l+u)^2$ is the natural scale factor for the empirical estimate of the Laplace operator [14]; and $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{l+u})]^\top = \mathbf{K}\boldsymbol{\alpha}$, where we have deliberately dropped the bias term b.

C. TSVM

The TSVM, originally proposed in [9] and further extended to deal with the peculiarities of remote sensing data in [10], aims at choosing a decision boundary that maximizes the margin on both labeled and unlabeled data. The TSVM optimizes a loss function similar to (1), but $\gamma_M\|f\|^2_{\mathcal{M}}$ is replaced by a term related to the distance of unlabeled samples to the margin. The TSVM functional to be minimized is

$$\mathcal{L} = \frac{1}{l}\sum_{i=1}^{l} V(\mathbf{x}_i, y_i, f) + \gamma_L\|f\|^2_{\mathcal{H}} + \lambda\sum_{j=l+1}^{l+u} L^*\left(f\left(\mathbf{x}_i^*\right)\right) \tag{5}$$

where l and u are the number of labeled and unlabeled examples, λ is a free parameter that controls the relevance of unlabeled samples, and L^* is the symmetric hinge loss function

$$L^*\left(f\left(\mathbf{x}_i^*\right)\right) = \max\left(0, 1 - |f(\mathbf{x}^*)|\right). \tag{6}$$

The optimization of L^* can be seen as "self-learning," i.e., we use the prediction for $\mathbf{x}^*$ for training the mapping for that same example. Minimizing (6) pushes away unlabeled samples from the margin, either negative or positive, thus the absolute value. When the number of labeled examples is small, balancing constraints are necessary to ensure that the samples are not pushed on only one side of the classification function, which would lead to extremely poor generalization performance.

D. Shortcomings and Limitations

Note that in both LapSVM and TSVM, the decision function $f(\mathbf{x})$ is a SVM. These algorithms give excellent results but still suffer from a high computational burden if a naive implementation is used. Effectively, the main problem with this approach is that typically very few unlabeled samples can be used for training, and thus, the marginal data distribution is not efficiently modeled in practical cases.

For the case of the TSVM, many heuristics have been proposed to reduce its computational cost. In [26], a mixed-integer programming was proposed to find the labeling with the lowest objective function. The optimization, however, is intractable for large data sets. In [27], a heuristic that iteratively solves a convex SVM objective function with alternate labeling of unlabeled samples was proposed. Unfortunately, the algorithm is only capable of dealing with a few thousand samples. Another implementation, i.e., the ∇TSVM, still has a cubic cost and requires storing a $l+u$ sample kernel matrix, which precludes its use in remote sensing [28]. Finally, the concave convex procedure-TSVM scales better—almost quadratically—but it is still impractical for more than 200 000 samples [29].

Several approaches have already tried to alleviate the computational cost of LapSVM, either by using a sparsified manifold regularizer [30] or by using an L_1 penalization term and a regularization path algorithm [31]. These approaches, however, do not make the use of LapSVM practical for several millions of unlabeled samples. The latter case is precisely the situation where using LapSVM provides significant benefits compared with the purely supervised approach. A second and important problem with LapSVM is related to the use of a functional form of the LE, which gives rise to a constrained optimization problem.

In this paper, we propose an alternative methodology to solve the aforementioned problems. Note that by replacing the SVM by a neural network, significant progress can be made regarding the scalability of the classifier. On the other hand, by replacing the LE with a proper loss function for embedding, similar pixels are forced to be mapped closely and dissimilar pixels to be separated. These two improvements solve the previous problems and yield a potentially useful method for remote sensing operational applications.

III. SSL With Neural Networks

It is possible to apply the principles of graph-based SSL to neural networks. This can provide more efficient manners of exploiting a higher number of both labeled and unlabeled pixels. Neural networks can be trained by SGD [32] and can therefore easily scale to millions of samples, which is infeasible with SVM-based methods.

Following the semisupervised regularization framework revised in Section II, we propose to minimize

$$\mathcal{L} = \frac{1}{l}\sum_{i=1}^{l} V(\mathbf{x}_i, y_i, f) + \gamma_{\mathcal{M}}\frac{1}{(l+u)^2}\sum_{i,j=1}^{l+u} L(f_i, f_j, W_{ij}) \tag{7}$$

[1]In our case, nearby points are those pixels spectrally similar, and thus, the assumption is applied to the (high-) dimensional space of image pixels.

where the edge weights W_{ij} define pairwise similarity relationships between unlabeled examples.

Note that this problem, like TSVM's objective, is nonconvex in the nonlinear case, and there is no simple optimization scheme for solving it, even for linear models such as kernel machines. To solve it, we propose to minimize this function in the primal by SGD and use a multilayer perceptron for solving the nonlinear case.

The general scheme of SSNNs proposed in this paper is described in Algorithm 1. Basically, the algorithm is provided with labeled and unlabeled samples and for each iteration makes gradient steps to optimize first the loss function of errors (labeled information) V and then the regularizer (unlabeled information) L. Note that supervised and unsupervised neural networks are particular cases of this approach. By allowing the weighting of V and L within the same neural net, the model provides a general learning framework for classification problems.

```
Algorithm 1 General training scheme of an SSNN.
  Input: labeled pixels $\{\mathbf{x}_i, y_i\}$ and unlabeled pixels $\{\mathbf{x}_j^*\}$
  repeat
    Pick a random labeled example $\{\mathbf{x}_i, y_i\}$
    Make a gradient step to optimize $V(\mathbf{x}_i, y_i, f)$
    Pick a random unlabeled pair of neighbors $\mathbf{x}_i^*$ and $\mathbf{x}_j^*$
    Make a gradient step to optimize $L(f(\mathbf{x}_i), f(\mathbf{x}_j), 1)$
    Pick a random unlabeled pair $\mathbf{x}_k^*$ and $\mathbf{x}_l^*$
    Make a gradient step to optimize $L(f(\mathbf{x}_k^*), f(\mathbf{x}_l^*), 0)$
  until Stopping criterion
```

The implementation of the proposed algorithm requires the definition of a loss function for both labeled and unlabeled samples, a neural network topology, an optimization algorithm, and balancing constraints. In the following sections, we analyze these issues in detail. Generally speaking, in this framework, neighbors are defined according to a k nearest-neighbor algorithm.

A. Loss Function for Supervised Classification

In our approach, we use the hinge loss function in (2) for V, like SVM, LapSVM, or TSVM do. Neural networks traditionally use a squared loss function that is appropriate for Gaussian noise distributions. When this assumption does not necessarily hold, the use of the hinge loss function may be more appropriate. In fact, it has been advocated that for the classification setting, the use of entropic or hinge loss functions can be more suitable [9], [33].

B. Loss Function for Unsupervised Classification

LapSVM implements a *functional* version of LEs: rather than learning a one-to-one mapping between the input space and the embedding space, it learns a function (a SVM) that preserves neighborhood relations. Nevertheless, the optimization of the LE loss term $W_{ij}\|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|^2$ can be difficult. Indeed, one has to enforce the following constraints: $\mathbf{f}^\top\mathbf{D}\mathbf{f} = \mathbf{I}$ and $\mathbf{f}^\top\mathbf{D}\mathbf{1} = \mathbf{0}$, which ensure that the new features have zero mean and unit variance. The presence of such constraints makes the optimization difficult, and we seek a loss function that would allow an unconstrained optimization.

In this sense, the use of the following alternative loss function for embedding has recently been proposed in [34]:

$$L(f_i, f_j, W_{ij}) = \begin{cases} \sum_{ij} \|f_i - f_j\|^2, & \text{if } W_{ij} = 1 \\ \sum_{ij} \max\left(0, m - \|f_i - f_j\|\right)^2, & \text{if } W_{ij} = 0 \end{cases} \tag{8}$$

where $W_{ij} = 1$ if i and j are deemed similar and 0 otherwise, and m is a margin. The authors in [34] implemented this loss in a neural network and called the method Dimensionality reduction by Learning an Invariant Mapping (DrLIM). DrLIM encourages similar examples to be mapped closely and dissimilar examples to be separated by at least the distance m. This prevents the embedding from collapsing and makes the use of constraints unnecessary. Loss functions of this type have recently been applied in [35], where the LapSVM approach is generalized to networks of several layers.

We can adapt this type of objective function directly to the task of classifying the unlabeled data, as presented in [36] and [37], yielding an approach interestingly related to TSVM. We propose that, rather than performing an optimization over the *coordinates* of the samples in the embedding, one can *directly* optimize the labels of unlabeled data by directly encoding into the algorithm that neighbors with $W_{ij} > 0$ should have the same class assignment. This can be done by optimizing a general objective function such as

$$L(f_i, f_j, W_{ij}) = \begin{cases} \sum_{ij} \eta^{(+)} V(\mathbf{x}_i, f(\mathbf{x}_i), c) & \text{with } c = \operatorname{sign}(f_i + f_j) \text{ if } W_{ij} = 1 \\ \sum_{ij} -\eta^{(-)} V(\mathbf{x}_i, f(\mathbf{x}_i), c) & \text{with } c = \operatorname{sign}(f_j) \text{ if } W_{ij} = 0 \end{cases} \tag{9}$$

where we have the following:

1) $\mathbf{W}$ is a pairwise similarity matrix defined *a priori*, as previously done.
2) $V(\cdot)$ is the hinge loss function [see (2)]. For the multiclass case, we can sum over the classes Ω, that is, $V(\mathbf{x}, f(\mathbf{x}), y) = \sum_{c=1}^{\Omega} \max(0, 1 - y(c) f_c(\mathbf{x}))$, where $y(c) = 1$ if $y = c$ and -1 otherwise.
3) $\eta^{(+)}$ and $-\eta^{(-)}$ are learning rates. The classifier should be trained to classify $\mathbf{x}_i$ and $\mathbf{x}_j$ in the same class if $W_{ij} = 1$, with learning rate $\eta^{(+)}$, and should be trained to push them in different classes if $W_{ij} = 0$, with learning rate $-\eta^{(-)}$.

Intuitively, (9) assigns a pair of neighbors to the cluster with the most confident label from the pair. Examples $\mathbf{x}_j$ that are not neighbors of $\mathbf{x}_i$, i.e., when $W_{ij} = 0$, are encouraged to fall into different clusters. This principle is illustrated in Fig. 2.

Note that, if only the L cost is used [see (9)], then the method only works with unlabeled samples, thus performing unsupervised learning. This can be done by setting $\gamma_{\mathcal{M}}$ arbitrarily large in (7). Therefore, the proposed method constitutes a generalization of both supervised and unsupervised approaches. The *NCutEmb* approach, as described in [37], provides algorithms for the binary and the multiclass case in an unsupervised setting. In the binary case, (9) describes the loss that is minimized. A neural network $f(\mathbf{x})$ with one output $y \in \{\pm 1\}$ and trained online via SGD is used. For the multiclass case, two algorithmical variants were provided, i.e., $NCutEmb^{max}$ and $NCutEmb^{all}$, which differ by the way the "winning" class is chosen. They are also evaluated in this paper. $NCutEmb^{max}$

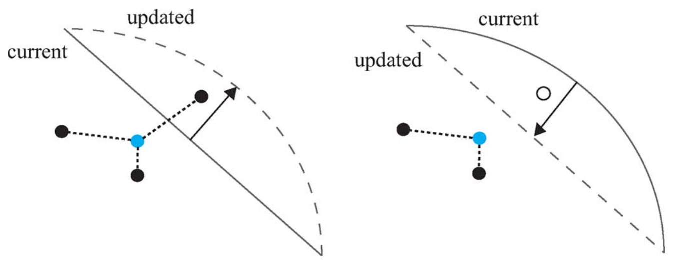

Fig. 2. Unsupervised learning using NCutEmb. A hyperplane that separates neighbors is pushed to classify them in the same class (left; the classifier cuts through an edge of the graph and is pushed upward), whereas a hyperplane that classifies nonneighbors in the same class is modified to separate them (right; the hyperplane is pushed to separate the unconnected points).

is very similar to the binary case: we push neighbors toward the most confident label. With $NCutEmb^{all}$, we simultaneously push toward all clusters, with one learning rate η per class. Each learning rate is weighted by the outputs of the neural network. For a deeper analysis of the multiclass method, see [37].

C. Neural Network Architecture

We propose to use a multilayer neural network model whose neuron j in layer $l+1$ yields

$$x_j^{l+1} = g\left(\sum_i w_{i,j}^l x_i^l + w_{bj}^l\right) \tag{10}$$

where $w_{i,j}^l$'s are the weights connecting neuron i in layer $l-1$ to neuron j in layer l, w_{bj}^l's are the bias terms of neuron j in layer l (we fixed it to $+1$), and g is a nonlinear activation function (in our case, the hyperbolical tangent was used). As in the LapSVM case, we denote the output (prediction) of the model for the sample $\mathbf{x}_i$ as $f(\mathbf{x}_i)$ and the vector of all predictions as $\mathbf{f} = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{l+u})]^\top$. For the multiclass case, the network has an output node for each class, but each output shares the same hidden neurons, as is often done in neural networks.

The use of neural networks for SSL has several advantages:

1) Kernel methods are computationally demanding as the number of samples increases.
2) Regularization can easily be controlled by limiting the number of hidden neurons in the network.
3) The model is readily applicable to new pixels as they become available.
4) One can easily encode *prior* knowledge via the architecture of the neural network.

In this paper, we make use of very simple network architectures. Essentially, we select the best model among 1) a linear network and 2) a one-hidden-layer network (with 20, 40, 60, or 80 hidden units). This is a straightforward way of controlling the model's complexity.

D. Stochastic Gradient Optimization

Gradient descent works by moving toward the minimum of the loss function by taking a step proportional to the negative of the loss function's gradient. Usually, this requires computing the *average value* of the gradient over the whole data set. This technique is referred to as *batch* gradient descent. Batch gradient can be a rather costly operation when working with a very large number of pixels. To circumvent this problem, training is done here using SGD [32], [38]–[40]. Unlike batch gradient descent, SGD processes one example at a time. This requires much less calculations and can often yield better generalization properties, as it finds an *approximate* minimum rather than an exact one. This technique has been applied to the field of remote sensing in [3], [25], and [41]–[45]. With SGD, parameters w_{ij}^l's are updated using

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla \mathcal{L} \tag{11}$$

where $\mathbf{w}^t$ is the weight vector at training epoch t, and $\mathcal{L}$ refers to any differentiable loss function [in our case the function in (7)]. Training with SGD allows handling very large databases since every update involves one (or a pair of) example and linearly grows in time with the size of the data set. Moreover, the convergence of the algorithm is ensured for low enough values of η.

E. Balancing Constraints

In the proposed formulation, minimizing the loss function L can lead to collapse in the embedding (in the case of manifold regularization) or to classify all the samples in a single class (in the case of manifold transduction). Similarly, V can be trapped in a local minimum by ignoring the possibly uneven distribution of samples per class. These problems can be circumvented by constraining the solution. Several options can be considered [36], [37]:

1) *no-bal*: No balancing constraint, which can be appropriate when training and test data have similar class proportions.
2) *ignore-bal*: Keep a number of examples in the cache and count how many have been attributed to each class c_i. If a new example $\mathbf{x}^*$ is given a prediction c^* that has been seen too often in the cache, then ignore that example.This constraint can be implemented in different ways. In the case of well-balanced classes, one can impose that the number of examples in the cache belonging to class c_i should not exceed M/K, where M is the size of the cache, and K is the number of classes. This type of constraint can be used for unsupervised learning [37]. We designate it as p_u (uniform distribution).

   In the SSL, one can take advantage of the prior label distribution in the data set to estimate the proportion of each class to allow in the cache. In [36], the authors refer to this distribution as p_{trn}. This method might lead to poor results if the distribution in the training (labeled) set is very different from that of the test (unlabeled) set.

These three types of balancing constraints ("no-bal", p_u, and p_{trn}) are used in this paper.

F. Remarks

Since the proposed method can give rise to a nonconvex problem (if a hidden layer is used), and because of the scalability problem with nonlinear kernel machines, we have made the following algorithmic choices: 1) minimization is done in the primal, making online learning possible; 2) the neural network approach provides faster training and testing rates than competing kernel methods; and 3) noting that there is a risk of falling into local minima, a set of possible balancing constraints have been included in the method.

Note that the method implements regularization in two ways. First, the use of unlabeled samples helps to better model the marginal distribution, thus leading to smoother solutions. Second, by including balancing constraints in the standard conjugate gradient algorithm, one is introducing another form of regularization, as a proper estimation of the class density is forced. Thus, even in the case of not considering unlabeled samples, the proposed method performs some kind of supervised regularized neural network, which has recently demonstrated excellent results compared with other (kernel-based) methods in remote sensing image classification [3], [25].

IV. Experimental Results

This section presents the application of the proposed techniques to the classification of two standard hyperspectral images in several ill-posed scenarios, i.e., where the number of features is larger than the number of labeled samples.

A. Data Collection

Two labeled hyperspectral images have been considered:

1) The first data set is the classical 220-band AVIRIS image taken over Indiana's Indian Pine test site in June 1992. The image is 145 $\times$ 145 pixels, contains 16 crop-type classes, and a total of 10 366 labeled pixels. This image is a classical benchmark to validate model accuracy and constitutes a very challenging classification problem because of the strong mixture of the classes' signatures and unbalanced number of labeled pixels per class. The six most representative classes were selected: "Corn" (2268), "Corn-min till" (834), "Grass/Pasture" (497), "Grass/Trees" (747), "Hay-windrowed" (489), "Soybean" (4050), and "Woods" (1294). Before training the classifiers, we removed 20 noisy bands covering the region of water absorption. The image is available at http://dynamo.ecn.purdue.edu/~biehl/.
2) The second image was acquired by the AVIRIS instrument over the Kennedy Space Center (KSC), Florida, on March 23, 1996, with a total of 224 bands of 10-nm width with center wavelengths from 400 to 2500 nm. The data were acquired from an altitude of 20 km and have a spatial resolution of 18 m. After removing low SNR bands and water absorption, a total of 176 bands remain for analysis. The data set originally contained 13 classes representing the various land cover types of the environment, containing many different marsh subclasses that we decided to merge in a fully representative marsh class. Thus, we finally used ten classes: "Water" (761), "Mud flats" (243), "Marsh" (898), "Hardwood swamp" (105), "Dark/broadleaf" (431), "Slash pine" (520), "CP/Oak" (404), "CP Hammock" (419), "Willow" (503), and "Scrub" (927). Note that in this case, more samples are available per class, and class distributions are not as inbalanced as in the first class distributions. More information can be found in http://www.csr.utexas.edu/.

Data were scaled to the range [0, 1] before training the classifiers.

B. Model Development and Experimental Setup

In the experiments, we analyze the performance of the semisupervised approach and the particular case of the unsupervised and supervised approaches. In the semisupervised scenario, we will compare the proposed SSNN with both TSVM and LapSVM. In the unsupervised scenario, we compare the unsupervised part of the method L with k-means working in both batch and online modes. Batch k-means updates each cluster center using the mean of the data points assigned to that cluster at the previous iteration. This involves computing the mean of every cluster at each iteration. Online k-means (also referred to as stochastic k-means; see, e.g., [32]) updates the cluster centers by using one point $\mathbf{x}_i$ at a time, which is drawn randomly. The cluster centers μ_i are usually updated using a rule such as $\mu_{t+1} \leftarrow \mu_t + \eta(\mathbf{x} - \mu_t)$, where η is a learning rate. Finally, results for the standard supervised SVM are also provided for mere comparison purposes.

The graph Laplacian $\mathbf{L}$ consisted of $l + u$ nodes connected using k nearest neighbors, and we computed the edge weights W_{ij} using the Euclidean distance among samples. For the kernel methods, we used the radial basis function (RBF) kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$, where $\sigma \in \mathbb{R}^+$.

As previously stated, the proposed method contains supervised and unsupervised learning as particular cases. In this sense, the following results are presented:

1) Semisupervised classification with different amounts of labeled pixels per class.
2) Semisupervised classification with labeled pixels chosen at random.
3) Supervised classification with different numbers of pixels *per* class and samples chosen at random.
4) Unsupervised classification with the NCutEmb method ("max" and "all" versions).

When appropriate, we try to compare different types of balancing constraints ("no-bal," "ignore-bal" with uniform p_u, or training p_{trn} distribution).

C. Experiment 1: AVIRIS Indian Pines

Table I presents the results for the Indian Pines image. We show the results for a varying number of labeled pixels (45, 100, 200) randomly selected from the whole image, as well as when using five labeled pixels *per* class. Note that by following a random selection procedure, some classes could be over/underrepresented or even missing: in the experiments, we avoided these cases by ensuring that every class is present in the labeled set. In the following, *no-bal* designates "no balancing constraint," p_u is a balancing constraint with equal class sizes, and p_{trn} is a balancing constraint with class sizes estimated from the training (labeled) set. Note that we also included the results obtained by the TSVM and LapSVM for the case of $l = 200$ and all unlabeled samples $u = 9345$. Naive implementation of the TSVM cannot cope with such a big amount of data, so we used the UniverSVM package,[2] which can be used for training in reasonable times for less than 20 000 samples, but cannot cope with millions of samples as our method (cf. Section IV-F). When only five labeled samples

[2]Available at http://www.kyb.tuebingen.mpg.de/bs/people/fabee/universvm.html.

TABLE I
SEMISUPERVISED RESULTS FOR THE INDIAN PINES IMAGE (SIX CLASSES). SEVERAL ACCURACY MEASURES ARE INCLUDED: PER-CLASS (C1–C6) ACCURACY, OA (OA[%]), AND KAPPA STATISTIC (κ). WE USE THREE TYPES OF BALANCING CONSTRAINTS. WE TRY 1) RANDOMLY SELECTED LABELED SAMPLES FROM THE DATA SET (45, 100, AND 200) AND 2) FIVE SAMPLES CHOSEN FROM EACH CLASS. IN THE LATTER CASE, $p_u = p_{trn}$

Samples	5/class		45			100			200			TSVM	LapSVM
	no-bal	p_{trn}	no-bal	p_u	p_{trn}	no-bal	p_u	p_{trn}	no-bal	p_u	p_{trn}	$l = 200$	
Corn (C1)	27.73	30.29	35.46	35.46	44.28	56.62	56.62	52.47	66.74	55.16	55.16	53.84	57.38
Grass/Pastures (C2)	0.0	78.18	65.05	65.05	54.55	66.26	66.26	66.26	65.98	65.98	64.95	60.18	60.44
Grass/Trees (C3)	97.85	97.18	98.92	98.92	93.68	99.59	99.59	98.11	99.45	99.45	98.50	97.11	98.21
Hay-windrowed (C4)	99.79	99.79	100.0	100.0	99.79	99.59	99.59	99.79	99.79	99.79	99.79	97.34	97.24
Soybean (C5)	59.85	47.08	50.84	50.84	51.36	52.17	52.17	55.08	61.27	61.27	61.16	56.36	59.85
Woods (C6)	98.14	98.99	96.43	96.43	99.46	95.23	95.23	97.26	97.17	97.17	98.50	91.23	98.18
OA[%]	**57.41**	**60.81**	**62.32**	**62.32**	**63.87**	**60.19**	**66.74**	**67.17**	**71.02**	**71.02**	**72.23**	**63.44**	**66.10**
κ	**0.47**	**0.53**	**0.55**	**0.55**	**0.57**	**0.61**	**0.61**	**0.62**	**0.65**	**0.65**	**0.67**	**0.57**	**0.59**

TABLE II
UNSUPERVISED RESULTS FOR THE INDIAN PINES IMAGE (SIX CLASSES) USING k-MEANS, NCutEmb^{max}, AND NCutEmb^{all}. PER-CLASS ACCURACY, OA (OA[%]), AND KAPPA STATISTIC (κ) ARE INCLUDED

Method	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**OA[%]**	κ
Batch k-means	29.15	0.0	83.94	99.18	53.59	99.07	**55.48**	**0.46**
Online k-means	29.92	0.0	83.94	99.18	53.39	99.00	**55.55**	**0.46**
NCutEmb^{max}	46.44	0.0	96.12	100.0	60.02	99.15	**61.12**	**0.53**
NCutEmb^{all}	48.00	0.0	96.35	100.0	57.35	100.00	**60.52**	**0.52**

TABLE III
SUPERVISED RESULTS FOR THE INDIAN PINES IMAGE (SIX CLASSES) WITH THE LABELED SAMPLES USED IN THE SEMISUPERVISED SECTION. PER-CLASS ACCURACY, OA (OA[%]), AND KAPPA STATISTIC (κ) ARE INCLUDED WE HAVE USED THE SAME NEURAL NET THAT WAS USED FOR SSL FOR COMPARISON PURPOSES

Samples	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**OA[%]**	κ
200	71.37	53.15	98.09	99.58	61.28	98.66	**72.39**	**0.67**
100	59.30	55.85	95.94	98.76	48.90	98.44	**65.74**	**0.59**
45	60.99	23.98	91.13	99.59	43.51	97.75	**58.05**	**0.50**
5/class	47.62	8.56	87.63	100.0	56.02	98.37	**56.47**	**0.46**

are used, a random sample selection from the whole image yields better results (between 2% and 5% improvement). As more labeled samples are used, results obviously improve and overall accuracies raise from about 60% to 72%. It is also noticeable that the best balancing constraint is p_{trn} in all cases, probably because the *prior* label distribution in the data set is used to estimate the proportion of each class to allow in the cache. The use of this balancing turns to be important since the uneven class distribution systematically induces poor results for specific classes (e.g., class 2 with no balancing constraint).

With regard to the results obtained by TSVM and LapSVM, it is very important to note that the obtained results are worse than those obtained with the SSNNs in any of the training modes (*sup*, *no-bal*, p_u, and p_{trn}) when using the same number of labeled samples (l = 200), and are only similar to our results with half the labeled samples (neural network columns with l = 100). Differences are numerically and statistically significant (see Section IV-E for a deeper analysis), thus suggesting that the SSNN exploits more efficiently the unlabeled information than the semisupervised kernel methods.

We will now consider limit cases of the semisupervised setting

1) *Unsupervised learning* ($l = 0$). With regard to the unsupervised results (Table II), we compare the results using k-means (batch and online versions) and the NCutEmb methods ("max" and "all" versions). The overall accuracy (OA) achieved with NCutEmb is good (61.12%), and about 5% improvement compared with k-means is obtained. Additionally, NCutEmb even outperforms the SSNN for the case of five pixels per class (cf. Table I). However, class 2 is ignored, as in the "no-bal" case, which confirms the importance of introducing regularization through balancing class priors. Moreover, class 2 corresponds to Grass/Pasture, which presents several subcategories in this image. This could make the task difficult for an online unsupervised algorithm based on neighborhood relations, as one could expect different categories of, for instance, Grass to be neighbors.
2) *Supervised learning* ($u = 0$). From Table III, we can observe that the SSL approach improves purely supervised results when we have 45 or 100 labeled pixels, but the difference between SSL and supervised neural networks is negligible when using 200 labeled samples (cf. Table I). This behavior can be explained by noting that the more supervised information is available, the more accurate the classification is in general. However, a clear saturation effect is typically observed. A similar curve is observed as we increase the number of unlabeled information. The method uses both labeled samples to fix a *support* for the class distribution and the unlabeled samples to characterize (parametrize) the data *marginal distribution*. In general, SSL methods require a higher number of unlabeled samples than labeled to provide a noticeable improvement in the classification accuracy, as reported in [16], [46], and [47]. However, including a high number of unlabeled samples in the formulation of kernel methods is not straightforward and usually implies an extremely high computational cost. This problem has been mitigated with the use of neural networks.

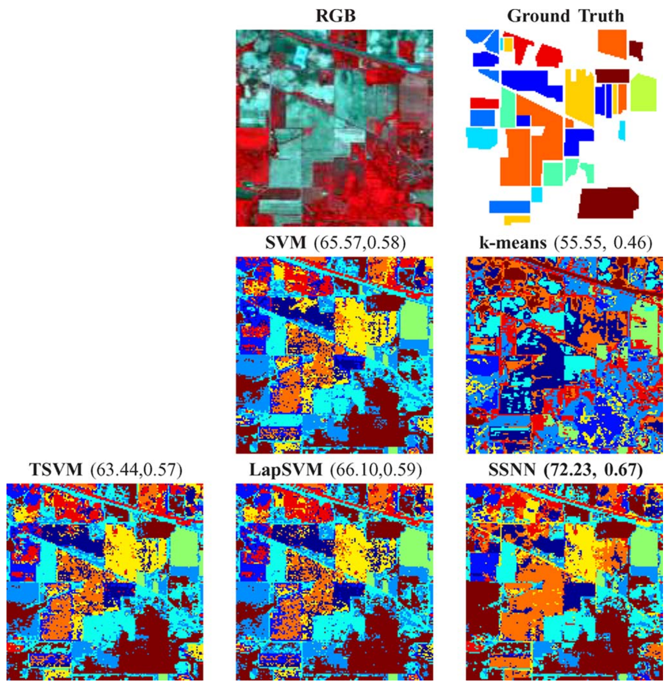

Fig. 3. RGB composition and classification maps with the best SVM, k-means, TSVM, and LapSVM ($l = 200, u = 1000$), along with the best SSNN for the AVIRIS Indian Pines image. OA and kappa statistic are given in brackets.

Fig. 3 shows the classification maps obtained with several state-of-the-art semisupervised classifiers, such as the TSVM and LapSVM, along with the proposed SSNN. In addition, the supervised SVM and the unsupervised k-means algorithms are included for comparison. Accuracy and kappa statistics are indicated. In general, it is observed that numerical results are confirmed by visual inspection of the classification maps. Poor results are obtained with k-means, whereas a supervised SVM yields high accuracy, only slightly improved by their semisupervised versions. The LapSVM slightly improves the results of the TSVM, but they yield similar classification maps. On the contrary, SSNN produces more homogenous areas and better classification maps; see, for instance, the high spatial homogeneity on different crop fields, whereas TSVM/LapSVM yields a noisy classification.

D. Experiment 2: KSC Image

Results for the semisupervised method for the KSC image are shown in Table IV. Unlike in the Indian Pines image, when only five labeled samples are used, a random sample selection yields slightly worse results than even sample selection. As more labeled samples are used, results obviously improve, and accuracies raise to 87%. Again, the best balancing constraint is the p_{trn} in all cases. The use of this balancing turns to be very important, since the uneven class distribution systematically induces very poor results for specific classes, e.g., classes 3 and 4. The test site consists of a series of impounded estuarine wetlands of the northern Indian River Lagoon (IRL). Detection of land cover for this environment is difficult due to the similarity of spectral signatures for certain vegetation types. The superclass marsh (constituted by similar subclasses "Cattail," "Graminoid," "Salt," and "Spartina") is particularly difficult, as previously reported elsewhere [48], where the area under the receiver operating curve offered by advanced target detectors was 6%–7% lower for this class than for other classes in the scene.

TSVM proves to be less accurate than the other tested methods. Indeed, SSNN achieves an improvement of 8% in OA using the same number of labeled samples. Good results are obtained with LapSVM, but the accuracies are slightly lower than those obtained with the SSNN. An improvement of about 4% for the SSNN trained with p_{trn} is observed, and very similar results between SSNNs and LapSVM are obtained with half the number of labeled samples (see columns in Table IV for SSNN, $l = 100$).

The two limit cases of SSL are studied again:

1) *Unsupervised learning* $(l = 0)$. Table V presents the unsupervised results. *NCutEmb* ignores two classes but achieves a much better performance than k-means in terms of accuracy and kappa statistic, leading to an improvement of about 10% using the NCutEmb$^{\max}$ and 8% using the NCutEmb$^{\text{all}}$. Similar to the semisupervised case, worst results are obtained for classes like the complex marsh superclass and the difficult case of hardwood swamp induced by the low number of labeled samples.
2) *Supervised learning* $(u = 0)$. By comparing Tables IV and VI, one can observe that SSL improves purely supervised results when less than 100 labeled pixels are used. Certainly, a small increase of the labeled information becomes more relevant in this (easier) problem than it was observed for the Indian Pines image.

Fig. 4 shows the classification maps, which generally match the numerical results. The LapSVM improves the results of the TSVM, whereas more homogenous areas and better classification maps are observed for the SSNN. See, for instance, the high spatial coherence on the IRL waters, in which TSVM/LapSVM yields a noisy classification, whereas SSNN achieves spatially homogeneous results. This suggests that exploiting unlabeled samples properly is crucial for modeling the data complexity (e.g., similar water spectra are confused when using a low number of unlabeled samples). Another interesting example is that of the inhomogeneous area of impounded estuarine wetlands southeast of the image. Here, small marsh classes are correctly detected with the proposed method, whereas overregularized solutions are obtained with TSVM/LapSVM probably due to the weak support used for data modeling.

E. On the Statistical Significance of the Results

We have performed here a statistical analysis of the differences between all the considered methods in this paper: semisupervised (SSNN, TSVM, LapSVM), unsupervised (k-means, $NCutEmb$), and supervised SVM. The comparison is done through the McNemar's test [49], [50]. The McNemar's test, which is based on the standardized normal test statistic,[3] assesses whether statistical differences between methods exist and also serves to estimate the statistical gain of one classifier versus the other. The difference in accuracy between two

[3]Computing the z-score of classifiers 1 and 2 reduces to $z = (f_{12} - f_{21})/\sqrt{f_{12} + f_{21}}$, where f_{12} represents the number of samples correctly classified by 1 and incorrectly classified by 2.

TABLE IV
SEMISUPERVISED RESULTS FOR THE KSC IMAGE (TEN CLASSES). SEVERAL ACCURACY MEASURES ARE INCLUDED: PER-CLASS ACCURACY (C1–C10), OA (OA[%]), AND KAPPA STATISTIC (κ). WE USE TWO TYPES OF BALANCING CONSTRAINTS: 1) RANDOMLY SELECTED LABELED SAMPLES FROM THE DATA SET (65, 100, AND 200) AND 2) FIVE PIXELS CHOSEN FROM EACH CLASS. IN THE LATTER CASE, $p_u = p_{trn}$. THE NUMBER OF HIDDEN UNITS (0, 20, 40, 60, OR 80) AND THE LEARNING RATE ([0.0001, 0.001]) WERE SELECTED

Samples	5/class		65			100			200			TSVM	LapSVM
	no-bal	p_{trn}	no-bal	p_u	p_{trn}	no-bal	p_u	p_{trn}	no-bal	p_u	p_{trn}	$l = 200$	
Water (C1)	90.09	90.75	92.73	92.73	93.39	88.81	88.81	89.88	91.19	91.19	90.11	85.74	89.95
Mud flats (C2)	84.87	89.50	98.74	98.74	99.58	100.0	100.0	89.79	99.57	99.57	87.18	86.91	85.58
Marsh (C3)	49.54	36.19	30.62	30.62	28.54	36.21	36.21	42.68	48.61	48.61	56.92	56.79	56.57
Hardwood swamp (C4)	0.0	77.45	32.35	32.35	25.49	30.39	30.39	87.25	0.0	0.0	89.11	87.45	88.58
Dark/broadleaf (C5)	88.97	91.31	91.55	91.55	90.38	90.52	90.52	85.78	91.04	91.04	90.56	89.97	90.34
Slash pine (C6)	83.30	83.11	80.97	80.97	82.33	79.34	79.34	93.18	86.51	86.51	95.44	92.77	94.52
CP/Oak (C7)	92.27	89.03	92.52	92.52	88.78	92.50	92.50	86.75	92.60	92.60	93.11	91.68	90.94
CP Hammock (C8)	87.80	89.51	93.17	93.17	97.07	95.33	95.33	97.54	88.66	88.66	96.98	93.73	96.92
Willow (C9)	85.45	87.68	86.67	86.67	90.71	86.53	86.53	93.67	91.39	91.39	98.53	97.15	97.52
Scrub (C10)	98.14	98.47	98.58	98.58	98.14	98.46	98.46	98.02	98.43	98.43	99.66	97.94	99.05
OA[%]	**81.69**	**81.68**	**80.86**	**80.86**	**80.86**	**81.19**	**81.19**	**84.74**	**83.52**	**83.52**	**87.89**	**79.66**	**83.11**
κ	**0.80**	**0.80**	**0.79**	**0.79**	**0.79**	**0.79**	**0.79**	**0.83**	**0.82**	**0.82**	**0.87**	**0.81**	**0.83**

TABLE V
UNSUPERVISED RESULTS FOR THE KSC IMAGE (TEN CLASSES) USING k-MEANS, NCutEmb^{max}, AND NCutEmb^{all}. PER CLASS ACCURACY, OA (OA[%]), AND KAPPA STATISTIC (κ) ARE GIVEN

Method	Batch k-means	Online k-means	NCutEmb^{max}	NCutEmb^{all}
C1	89.49	88.57	95.14	96.32
C2	61.32	66.26	100.0	95.06
C3	48.13	37.57	26.11	22.58
C4	0.0	0.0	0.0	0.0
C5	0.0	33.41	93.27	90.95
C6	71.92	96.54	72.12	75.77
C7	39.60	41.58	89.11	64.11
C8	95.70	85.68	88.54	95.94
C9	80.72	77.14	76.34	71.37
C10	99.03	99.46	98.27	98.60
OA[%]	**67.42**	**70.75**	**77.41**	**75.13**
κ	**0.64**	**0.67**	**0.75**	**0.72**

TABLE VI
SUPERVISED RESULTS FOR THE KSC IMAGE (TEN CLASSES) WITH THE LABELED PIXELS USED IN THE SEMISUPERVISED SECTION. PER-CLASS ACCURACY, OA (OA[%]), AND KAPPA STATISTIC (κ) ARE PROVIDED. WE USE THE SAME MODELS AS THOSE USED WITH SSL FOR COMPARISON PURPOSES

Samples:	5/class	65	100	200
C1	86.66	95.38	94.81	93.22
C2	89.50	99.58	90.64	93.16
C3	35.41	28.35	45.92	56.37
C4	0.0	31.37	88.24	63.37
C5	84.98	88.97	86.26	89.59
C6	64.66	80.39	92.98	93.25
C7	89.03	75.06	85.75	92.60
C8	98.54	94.15	95.58	94.46
C9	41.62	88.48	90.41	96.01
C10	99.45	99.67	99.89	99.89
OA[%]	**73.61**	**79.63**	**85.89**	**87.87**
κ	**0.71**	**0.77**	**0.84**	**0.86**

classifiers is said to be statistically significant if $|z| > 1.96$, and the sign of z indicates which is the most accurate classifier.

Table VII presents the results for the Indian Pines image for the case of $n = 200$. Looking at the first row in the table, SSNN yields better significance than all the methods but LapSVM. Nevertheless, differences are only statistically significant ($|z| > 1.96$) when confronted to unsupervised or supervised methods. It is worth to note that there is no statistical difference between supervised and SSNNs, which suggests that the balancing constraints are a good method to impose regularization, even in supervised learning [differences emerge between SVM and Neural Network (NN)]. Similarly, the proposed unsupervised

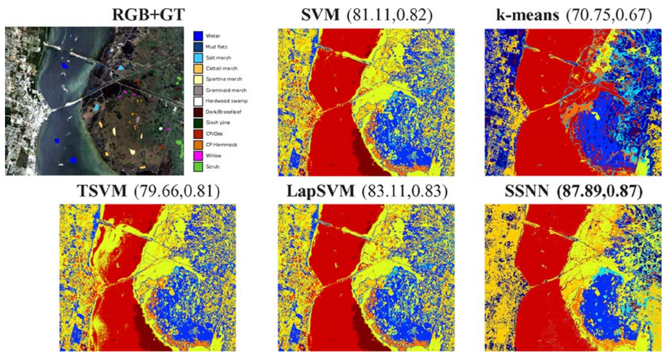

Fig. 4. RGB composition and classification maps with the best SVM, k-means, TSVM, and LapSVM ($l = 200, u = 1000$), along with the best SSNN for the KSC image. OA and kappa statistic are given in brackets.

NCutEmb outperforms k-means, yet not statistically ($z = 0.99$). Table VIII shows the corresponding results for the KSC image. A similar behavior is observed. Summarizing, in both situations, the proposed method is significantly different (and in most of the cases better) than the state-of-the-art methods.

F. On the Computational Cost

Classification using SSNNs is in all cases computationally cheaper than kernel-based SSL. With very large images, the difference could be of several orders in magnitude. SSNNs can efficiently exploit all unlabeled samples in an image. For example, it can process 10^4 samples in no more than a few minutes on a standard computer and linearly scales with the number of samples.

In Table IX, we present the typical training times for all the algorithms involved in the comparisons: SSNN, NN, k-means, NCutEmb, LapSVM, and TSVM. As can be observed, the SSNN presents almost no computational overhead when compared with the NN or a simple clustering algorithm such as k-means. On the other hand, kernel-based SSL algorithms can multiply this cost by a large factor.

V. DISCUSSION AND CONCLUSION

This paper has presented a framework for SSNNs. The proposed methods allow computationally efficient remote sensing image classification. We compared the results to standard

TABLE VII
McNEMAR'S TEST FOR THE INDIAN PINES IMAGE (SIX CLASSES) BETWEEN ALL CONSIDERED METHODS: SSNN, SUPERVISED NEURAL NETWORK, SVM, NCutEmb, k-MEANS, TSVM, AND LAPSVM. WE PERFORM THE COMPARISON FOR $n = 200$ LABELED PIXELS

	SSNN	NN	SVM	NCutEmb	k-means	TSVM	LapSVM
SSNN	–	0.61	4.11	30.90	36.85	0.45	-0.29
NN	-0.61	–	3.83	31.67	35.52	0.16	-0.58
SVM	-4.11	-3.83	–	27.04	24.23	-6.91	-7.79
NCutEmb	-30.90	-31.67		–	0.99	-27.04	-27.67
k-means	-36.85	-35.52	-24.23	-0.99	–	-27.56	-28.22
TSVM	-0.45	-0.16	6.91	27.04	27.56	–	-1.44
LapSVM	0.29	0.58	7.79	27.67	28.22	1.44	–

TABLE VIII
McNEMAR'S TEST FOR THE KSC IMAGE (TEN CLASSES) BETWEEN ALL CONSIDERED METHODS: SSNN, SUPERVISED NEURAL NETWORK, SVM, NCutEmb, k-MEANS, TSVM, AND LAPSVM. WE PERFORM THE COMPARISON FOR $n = 200$ LABELED PIXELS (SAME SETTING AS TSVM AND LAPSVM)

	SSNN	NN	SVM	NCutEmb	k-means	TSVM	LapSVM
SSNN	–	0.43	2.30	18.19	27.34	2.27	1.44
NN	-0.43	–	1.86	21.4	-29.77	0.84	0.0
SVM	-2.30	-1.86	–	-13.24	-21.95	2.0	4.41
NCutEmb	-18.19	-21.4	13.24	–	13.02	-14.18	-15.07
k-means	-27.34	-29.77	21.95	-13.02	–	-22.81	-23.53
TSVM	-2.27	-0.84	2.0	14.18	22.81	–	-2.33
LapSVM	-1.44	0.0	4.41	15.07	23.53	2.33	–

TABLE IX
AVERAGE TRAINING TIMES (IN SECONDS, AVERAGED ON TEN RUNS) FOR SSNN, NN, SVM, k-MEANS, NCutEmb, LAPSVM, AND TSVM. THE TIMES ARE GIVEN FOR A STANDARD PC WITH A 2-GHz CPU

	SSNN	NN	SVM	k-means	NCutEmb	LapSVM	TSVM
Indian Pines	228	214	158	83	227	1722	1509
Kennedy Space Center	193	173	180	55	188	1733	1909

SSL methods (TSVM, LapSVM) as well as purely supervised (SVM) and unsupervised (k-means) algorithms. The proposed methods demonstrate in general an improved classification accuracy and computational efficiency on two hyperspectral images. In the case of obviating the unlabeled samples, the method reduces to a supervised neural network with balancing constraints, which performs some kind of regularization of the solution. Interestingly, in these cases, the (supervised) network outperforms kernel-based (semisupervised) algorithms, thus suggesting that TSVM and LapSVM do not exploit the unlabeled information as efficiently as the proposed method, and being computationally more expensive, the number of unlabeled samples to achieve comparable results is far too big. In the case of working in unsupervised mode, the network largely outperforms standard algorithms.

Essentially, the neural network adopts different kinds of regularization by including the unlabeled information and through the use of balancing constraints. An important aspect of the proposed SSNN is that developing variants is easy and intuitive, for example, through other graph algorithms for unlabeled sample exploitation, active learning methods for sample selection, or with other neural-network architectures.

Although the proposed framework opens in our opinion new and interesting research possibilities for efficient image classification, a small number of shortcomings remain and need to be addressed in future research. Most importantly, the class balancing needs to be studied in a more formal manner. It has been observed that class accuracies can become zero even if their spectral composition is significantly different from other classes. As it is the case in the vast majority of SSL literature, we have optimized our models on the OA and not on the kappa statistic. This induces a solution that may be not balanced if some classes only include a low number of pixels. Using the kappa statistic as the optimization goal would yield balanced per-class performance at the expense of lower values of accuracy. In this paper, different balancing constraints have been compared but these have not yet solved this specific problem. Future research should focus on optimizing a proper functional that would yield the right balance between the accuracy and the kappa index. Additionally, we observed in some cases a nonmonotonic improvement of class accuracies with respect to the number of labeled pixels, which is counterintuitive. This can be due to several factors: the selection of training samples, neural network initialization, and the accuracy selection criterion. The impact of these parameters on the solution is an issue to be studied, particularly when very few labeled samples are used. Active learning could certainly play a role here to alleviate these problems. Other forms of regularization, such as including the spatial information, would also help in making the method more stable. Despite all these shortcomings, we still feel that the method opens a wide range of possible applications and establishes an appropriate theoretical framework for the development of new semisupervised methods.

ACKNOWLEDGMENT

The authors would like to thank Prof. M. Crawford for providing the KSC data set, Prof. Landgrebe for providing the AVIRIS Indian Pines data set, and D. Tuia of the University of Lausanne (Switzerland) for designing Fig. 1.

References

[1] G. F. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Trans. Inf. Theory*, vol. IT-14, no. 1, pp. 55–63, Jan. 1968.

[2] B. Schölkopf and A. Smola, *Learning With Kernels—Support Vector Machines, Regularization, Optimization and Beyond*. Cambridge, MA: MIT Press, 2002.

[3] G. Camps-Valls and L. Bruzzone, "Kernel-based methods for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 6, pp. 1351–1362, Jun. 2005.

[4] G. Camps-Valls, L. Gómez-Chova, J. Muñoz-Marí, J. Vila-Francés, and J. Calpe-Maravilla, "Composite kernels for hyperspectral image classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 1, pp. 93–97, Jan. 2006.

[5] G. Camps-Valls, L. Gómez-Chova, J. Muñoz Marí, M. Martínez-Ramón, and J.L. Rojo-Álvarez, "Kernel-based framework for multi-temporal and multi-source remote sensing data classification and change detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 6, pp. 1822–1835, Jun. 2008.

[6] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*, 1st ed. Cambridge, MA: MIT Press, 2006.

[7] X. Zhu, "Semi-supervised learning literature survey," Comput. Sci., Univ. Wisconsin-Madison, Madison, Tech. Rep. 1530, 2005.

[8] Q. Jackson and D. Landgrebe, "An adaptive classifier design for high-dimensional data analysis with a limited training data set," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 12, pp. 2664–2679, Dec. 2001.

[9] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

[10] L. Bruzzone, M. Chi, and M. Marconcini, "A novel transductive SVM for the semisupervised classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 11, pp. 3363–3373, Nov. 2006.

[11] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems, NIPS2004*, vol. 16. Vancouver, BC, Canada: MIT Press, Dec. 2004.

[12] G. Camps-Valls, T. Bandos, and D. Zhou, "Semi-supervised graph-based hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 10, pp. 3044–3054, Oct. 2007.

[13] V. Sindhwani, P. Niyogi, and M. Belkin, "Beyond the point cloud: From transductive to semi-supervised learning," in *Proc. 22nd ICML*, 2005, pp. 824–831.

[14] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Dec. 2006.

[15] L. Gómez-Chova, G. Camps-Valls, J. Muñoz-Marí, and J. Calpe-Maravilla, "Semi-supervised image classification with Laplacian support vector machines," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 3, pp. 336–340, Jul. 2008.

[16] V. Castelli and T. Cover, "The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 2101–2117, Nov. 1996.

[17] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[18] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall, 1999.

[19] J. Kivinen, A. Smola, and R. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[20] A. Bordes and L. Bottou, "The Huller: A simple and efficient online SVM," in *Proc. ECML*, 2005, pp. 505–512.

[21] A. N. Tikhonov, "Regularization of incorrectly posed problems," *Sov. Math., Dokl.*, vol. 4, no. 6, pp. 1624–1627, Jan. 1963.

[22] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Adv. Comput. Math.*, vol. 13, no. 1, pp. 1–50, Apr. 2000.

[23] A. Berge, A. C. Jensen, and A. H. S. Solberg, "Sparse inverse covariance estimates for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 5, pp. 1399–1407, May 2007.

[24] T. V. Bandos, L. Bruzzone, and G. Camps-Valls, "Classification of hyperspectral images with regularized linear discriminant analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 3, pp. 862–873, Mar. 2009.

[25] G. Camps-Valls and A. Rodrigo-González, "Classification of satellite images with regularized AdaBoosting of RBF neural networks," in *Speech, Audio, Image and Biomedical Signal Processing Using Neural Networks*, B. Prasad and S. R. M. Prasanna, Eds. Berlin, Germany: Springer-Verlag, 2008, pp. 309–326.

[26] K. Bennet and A. Demiriz, "Semi-supervised support vector machines," in *NIPS 12*. Cambridge, MA: MIT Press, Dec. 1998.

[27] T. Joachims, *Making Large-Scale Support Vector Machine Learning Practical*. Cambridge, MA: MIT Press, 1999, pp. 169–184.

[28] O. Chapelle and A. Zien, "Semi-supervised classification by low density separation," in *Proc. AISTATS*, 2005, pp. 57–64.

[29] R. Collobert, F. Sinz, J. Weston, and L. Bottou, "Large scale transductive SVMs," *J. Mach. Learn. Res.*, vol. 7, pp. 1687–1712, Aug. 2006.

[30] I. W. Tsang and J. T. Kwok, "Large-scale sparsified manifold regularization," in *Proc. NIPS*, 2006, pp. 1401–1408.

[31] G. Gasso, K. Zapien, and S. Canu, *L1-Norm Regularization Path for Sparse Semi-Supervised Laplacian SVM,* 2007. [Online]. Available: http://eprints.pascal-network.org/archive/00003875/

[32] L. Bottou, "Stochastic learning," in *Advanced Lectures on Machine Learning*, vol. 3176, *Lecture Notes in Artificial Intelligence*, O. Bousquet and U. von Luxburg, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 146–168. [Online]. Available: http://leon.bottou.org/papers/bottou-mlss-2004

[33] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[34] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. CVPR*, 2006, pp. 1735–1742.

[35] F. Karlen, J. Weston, F. Ratle, and R. Collobert, "Deep learning via semi-supervised embedding," in *Proc. ICML*, 2008, pp. 1168–1175.

[36] M. Karlen, J. Weston, A. Erken, and R. Collobert, "Large scale manifold transduction," in *Proc. ICML*, 2008, pp. 448–455.

[37] F. Ratle, J. Weston, and M. Miller, "Large-scale clustering through functional embedding," in *Proc. ECML*, 2008, pp. 266–281.

[38] L. Bottou and L. Yann, "On-line learning for very large data sets," *Appl. Stoch. Models Bus. Ind.*, vol. 21, no. 2, pp. 137–151, Mar. 2005.

[39] L. Bottou and Y. LeCun, "Large-scale on-line learning," in *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press, 2004.

[40] Y. LeCun, L. Bottou, G. Orr, and K. Müller, "Efficient backprop," in *Neural Networks—Tricks of the Trade*. Berlin, Germany: Spinger-Verlag, 1998.

[41] J. D. Paola and R. A. Schowengerdt, "A review and analysis of back-propagation neural networks for classification of remotely-sensed multi-spectral imagery," *Int. J. Remote Sens.*, vol. 16, no. 16, pp. 3033–3058, Nov. 1995.

[42] D. L. Civco, "Artificial neural networks for land-cover classification and mapping," *Int. J. Geophys. Inf. Syst.*, vol. 7, no. 2, pp. 173–186, Mar. 1993.

[43] H. Bischof and A. Leona, "Finding optimal neural networks for land use classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 36, no. 1, pp. 337–341, Jan. 1998.

[44] L. Bruzzone and D. Fernandez-Prieto, "A technique for the selection of kernel-function parameters in RBF neural networks for classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 1179–1184, Mar. 1999.

[45] G. Camps-Valls, L. Gómez-Chova, J. Calpe, E. Soria, J. D. Martín, L. Alonso, and J. Moreno, "Robust support vector method for hyperspectral data classification and knowledge discovery," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 7, pp. 1530–1542, Jul. 2004.

[46] V. Castelli and T. M. Cover, "On the exponential value of labeled samples," *Pattern Recognit. Lett.*, vol. 16, no. 1, pp. 105–111, Jan. 1995.

[47] K. Sinha and M. Belkin, "The value of labeled and unlabeled examples when the model is imperfect," in *NIPS 2007*, vol. 20. Cambridge, MA: MIT Press, 2008.

[48] L. Capobianco, A. Garzelli, and G. Camps-Valls, "Target detection with semisupervised kernel orthogonal subspace projection," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 11, pp. 3822–3833, Nov. 2009.

[49] L. L. Lapin, *Probability and Statistics for Modern Engineering*. Long Grove, IL: Waveland Press, 1998.

[50] G. M. Foody, "Thematic map comparison: Evaluating the statistical significance of differences in classification accuracy," *Photogramm. Eng. Remote Sens.*, vol. 70, no. 5, pp. 627–633, May 2004.

Frédéric Ratle received the B.S. degree in engineering physics and the M.S. degree in applied sciences (in the field of optimization and statistical methods) from the Ecole Polytechnique de Montreal, Montreal, QC, Canada, in 2003 and 2005, respectively, and the Ph.D. degree in machine learning and data analysis from the University of Lausanne, Lausanne, Switzerland, in 2008.

He is currently a Scientist with Nuance Communications International BVBA, Merelbeke, Belgium, applying machine learning to speech and language processing.

Gustavo Camps-Valls (M'04–SM'07) was born in València, Spain, in 1972. He received the B.Sc. degree in physics, the B.Sc. degree in electronics engineering, and the Ph.D. degree in physics from the Universitat de València, València, in 1996, 1998, and 2002, respectively.

He is currently an Associate Professor with the Department of Electronics Engineering, Universitat de València, where he teaches electronics, advanced time series processing, and machine learning for remote sensing. He conducts and supervises research within the frameworks of several national and international projects and is an Evaluator of project proposals and scientific organizations. He is the author (or coauthor) of 60 international journal papers, more than 80 international conference papers, and several international book chapters, and is the Editor of the books *Kernel Methods in Bioengineering, Signal and Image Processing* (IGI, 2007) and *Kernel Methods for Remote Sensing Data Analysis* (Wiley, 2009). His research interests are tied to the development of machine learning algorithms for signal and image processing with special focus on remote sensing data analysis.

Dr. Camps-Valls has been a member of the Data Fusion Technical Committee of the IEEE Geoscience and Remote Sensing Society since 2007 and of the Machine Learning for Signal Processing Technical Committee of the IEEE Signal Processing Society since 2009. He is a Referee of many international journals and conferences, and currently serves on the Program Committees of the International Society for Optical Engineers (SPIE) Europe, the International Geoscience and Remote Sensing Symposium, the International Workshop on Artificial Neural Networks, Machine Learning for Signal Processing, and the International Conference on Image Processing.

Jason Weston received the Ph.D. degree from the University of London, London, U.K., in 2000.

From 2000 to 2002, he was a Researcher with Biowulf Technologies, New York, where he was applying machine learning to bioinformatics. From 2002 to 2003, he was a Research Scientist with the Max Planck Institute for Biological Cybernetics, Tübingen, Germany. From 2004 to June 2009, he was a research staff member with NEC Labs America, Princeton, NJ. Since July 2009, he has been a Research Scientist with Google Research, New York. His current research focuses on various aspects of statistical machine learning and its applications, particularly in text and images.