An Efficient and Versatile Variational Method for High-dimensional Data Classification

Xiaohao Cai[1*], Raymond. Chan[2*], Xiaoyu Xie[3*] and Tieyong Zeng[4*]

[1]School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, UK.
[2]Department of Operations and Risk Management and School of Data Science, Lingnan University, Tuen Mun, Hong Kong.
[3]Applied Mathematics, Brown University, Providence, RI 02912, US.
[4]Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong.

*Corresponding author(s). E-mail(s): x.cai@soton.ac.uk; raymond.chan@ln.edu.hk; Xiaoyu_Xie@Brown.edu; zeng@math.cuhk.edu.hk;

Abstract

High-dimensional data classification is a fundamental task in machine learning and imaging science. In this paper, we propose an efficient and versatile multi-class semi-supervised classification method for classifying high-dimensional data and unstructured point clouds. To begin with, a warm initialization is generated by using a fuzzy classification method such as the standard support vector machine or random labeling. Then an unconstraint convex variational model is proposed to purify and smooth the initialization, followed by a step which is to project the smoothed partition obtained previously to a binary partition. These steps can be repeated, with the latest result as a new initialization, to keep improving the classification quality. We show that the convex model of the smoothing step has a unique solution and can be solved by a specifically designed primal-dual algorithm whose convergence is guaranteed. We test our method and compare it with the state-of-the-art methods

on several benchmark data sets. Thorough experimental results demonstrate that our method is superior in both the classification accuracy and computation speed for high-dimensional data and point clouds.

Keywords: Semi-supervised clustering, point cloud classification, variational methods, graph Laplacian

1 Introduction

Data classification is a fundamental task in remote sensing, machine learning, computer vision, and imaging science [1–5]. The task, simply speaking, is to group the given data into different classes such that, on one hand, data points within the same class share similar characteristics (e.g. distance, edges, intensities, colors, and textures); on the other hand, pairs of different classes are as dissimilar as possible with respect to certain features. In this paper, we focus on the task of multi-class semi-supervised classification. The total number of classes K of the given data sets is assumed to be known, and a few samples, namely the training points, in each class, have been labeled. The goal is therefore to infer the labels of the remaining data points using the knowledge of the labeled ones.

For data classification, previous methods are generally based on graphical models, see e.g. [1, 2, 6] and references therein. In a weighted graph, the data points are vertices and the edge weights signify the affinity or similarity between pairs of data points, where the larger the edge weight is, the closer or more similar the two vertices are. The basic assumption for data classification is that vertices in the graph that are connected by edges with large weight should belong to the same class. Since a fully connected graph is dense and has the size as large as the square of the number of vertices, it is computationally expensive to work on it directly. In order to circumvent this, some cleverly designed approximations have been developed. For example, spectral approaches are proposed in [7, 8] to efficiently calculate the eigendecomposition of a dense graph Laplacian. In [9, 10], the nearest neighbor strategy was adopted to build up a sparse graph where most of its entries are zero, and therefore is computationally efficient.

In the literature, various studies for semi-supervised classification have been performed by computing a local minimizer of some non-convex energy functional or minimizing a relevant convex relaxation. To name just a few, we have the diffuse interface approaches using phase field representation based on partial differential equation techniques [11, 12], the MBO scheme established for solving the diffusion equation [7, 8, 13], and the variational methods based on graph cut [1, 2]. In particular, the convex relaxation models and special constraints on class sizes were investigated in [1]. In [2], some novelty region-force terms were introduced in the variational models to enforce the affinity between vertices and training samples. To the best of our knowledge, all these proposed

variational models have the so-called *no vacuum and overlap constraint* on the labeling functions, which gives rise to non-convex models with NP-hard issues. By allowing labeling functions to take values in the unit simplex, the original NP-hard combinatorial problem is rephrased into a continuous setting, see e.g. [1–4, 14–17] for various continuous relaxation techniques (e.g. the ones based on solving the eigenvalue problem, convex approximation, or non-linear optimization) and references therein.

Image segmentation can also be viewed as a special case of the data classification problem [4, 18], since the pixels in an image can be treated as individual points. Various studies and many algorithms have been considered for image segmentation. In particular, variational methods are among the most successful image segmentation techniques, see e.g. [19–25]. The Mumford-Shah model [19], one of the most important variational segmentation models, was proposed to find piecewise smooth representations of different segments. It is, however, difficult to solve since the model is non-convex and non-smooth. Then substantially rich follow-up works were conducted, and many of them considered compromise techniques such as: (i) simplifying the original complex model, e.g. finding piecewise constant solutions instead of piecewise smooth solutions [26–28]); (ii) performing convex approximations, e.g. using convex regularization terms like total variation [29, 30]; or (iii) using the smoothing and thresholding (SaT) segmentation methodology [17, 22, 31–33]; for more details please refer to e.g. [34, 34–41] and references therein. Moreover, various applications were put forward for instance in optical flow [42], tomographic imaging [43], and medical imaging [44–49].

In this paper, we propose a semi-supervised data classification method inspired by the SaT segmentation methodology [17, 22, 31–33]. The SaT methodology has been shown to be very promising in terms of segmentation quality and computation speed for images corrupted by many different types of blurs and noises. Briefly speaking, the SaT methodology includes two main steps: the first step is to obtain a smooth approximation of the given image through minimizing some convex models; and the second step is to get the segmentation results by thresholding the smooth approximation, e.g. using thresholds determined by the K-means algorithm [50]. Since the models used are convex, the non-convex and NP-hard issues in many existing variational segmentation methods (e.g. the Mumford-Shah model and its piecewise constant versions mentioned above) are naturally avoided.

Our proposed data classification method mainly contains two steps with a warm initialization. The warm initialization is a fuzzy classification result which can be generated by any standard classification method such as the support vector machine (SVM) [51], or by labeling the given data randomly if no proper method is available for the given data (e.g. the data set is too large or complicated). Its accuracy is not critical since our proposed method will improve the accuracy significantly from this starting point.

With the warm initialization, the first step which is also the key point of our method is to find a set of smooth labeling functions, where each gives

the probability of every point being in a particular class. They are obtained by minimizing a properly-chosen convex objective functional. In detail, the convex objective functional contains K independent convex sub-minimization problems, where each corresponds to one labeling function, with no constraints between these K labeling functions. For each sub-minimization problem, the model is formed by three terms: (i) the data fidelity term restricting the distance between the smooth labeling function and the initialization; (ii) the graph Laplacian (ℓ_2-norm) term, and (iii) the total variation (ℓ_1-norm) built on the graph of the given data. The graph Laplacian and the total variation terms regularize the labeling functions to be smooth but at the same time close to a representation on the unit simplex.

After obtaining the set of labeling functions, the second step of our method is just to project the fuzzy classification results obtained at step one onto the unit simplex to obtain a binary classification result. This step can be done straightforwardly. To improve the classification accuracy, these two steps can be repeated iteratively, where at each iteration the result at the previous iteration is used as a new initialization.

The main advantage of our method is threefold. Firstly, it performs outstandingly in computation speed, since the proposed model at the first step is convex and the K sub-minimization problems are independent of each other (with no constraint on the K labeling functions). The parallelism strategy can thus be applied straightforwardly to improve the computation performance further. On the contrary, the standard start-of-the-art variational data classification methods e.g. [2, 7, 12] have the constraint on unit simplex in their minimization models, and thus the non-convex or NP-hard issues can affect seriously the efficiency of these methods, even though some convex relaxations may be applied. Secondly, in addition to the multi-class classification problem, our method can also be used to tackle other problems like the one-class classification problem [52] (see Section 5.6) benefiting from its robustness in dealing with extremely unbalanced data sets. Thirdly, our method is generally superior in classification accuracy, due to its flexibility of merging the warm initialization and the two-step iterations which are tractable to improve the accuracy gradually. Note again that we are solving a convex model in the first step of each iteration, which guarantees a unique global minimizer. In contrast, there is, however, no guarantee that the results obtained by the standard start-of-the-art variational data classification methods e.g. [2, 7, 12] are global minimizers. The effectiveness of iterations in our proposed method will be shown in the experiments. For most cases, the clustering accuracy would be increased by a significant margin compared to the first initialization and generally outperforms the state-of-the-art variational classification methods.

The paper is organized as follows. In Section 2, the basic notation used throughout the paper is introduced. Our method for data sets classification is proposed in Section 3. In Section 4, we present the algorithm for solving the proposed model and its convergence proof. In Section 5, we test our method on benchmark data sets and compare it with the start-of-the-art methods. Conclusions are drawn in Section 6.

2 Basic notation

Let $G = (V, E, w)$ be a weighted undirected graph representing a given point cloud, where V is the vertex set (in which each vertex represents a point) containing N vertices, E is the edge set consisting of pairs of vertices, and $w : E \to \mathbb{R}_+$ is the weight function defined on the edges in E. The weights $w(\boldsymbol{x}, \boldsymbol{y})$ on the edges $(\boldsymbol{x}, \boldsymbol{y}) \in E$ measure the similarity between the two vertices $\boldsymbol{x}$ and $\boldsymbol{y}$; the larger the weight is, the more similar (e.g. closer in distance) the pair of the vertices is.

There are many different ways to define the weight function. Let $d(\cdot, \cdot)$ be a distance metric. Several particularly popular definitions of weight functions are as follows: (i) radial basis function

$$w(\boldsymbol{x}, \boldsymbol{y}) := \exp(-d(\boldsymbol{x}, \boldsymbol{y})^2/(2\xi)), \quad \forall (\boldsymbol{x}, \boldsymbol{y}) \in E, \tag{1}$$

for a prefixed constant $\xi > 0$; (ii) Zelnic-Manor and Perona weight function

$$w(\boldsymbol{x}, \boldsymbol{y}) := \exp\left(\frac{-d(\boldsymbol{x}, \boldsymbol{y})^2}{\mathrm{var}(\boldsymbol{x})\mathrm{var}(\boldsymbol{y})}\right), \quad \forall (\boldsymbol{x}, \boldsymbol{y}) \in E, \tag{2}$$

where $\mathrm{var}(\cdot)$ denotes the local variance; and (iii) the cosine similarity

$$w(\boldsymbol{x}, \boldsymbol{y}) := \frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle \langle \boldsymbol{y}, \boldsymbol{y} \rangle}}, \quad \forall (\boldsymbol{x}, \boldsymbol{y}) \in E, \tag{3}$$

where $\langle \cdot, \cdot \rangle$ represents the inner product.

Let $W = (w(\boldsymbol{x}, \boldsymbol{y}))_{(\boldsymbol{x}, \boldsymbol{y}) \in E} \in \mathbb{R}^{N \times N}$, the so-called affinity matrix, which is usually assumed to be a symmetric matrix with non-negative entries. Let a diagonal matrix be $D = (h(\boldsymbol{x}, \boldsymbol{y}))_{(\boldsymbol{x}, \boldsymbol{y}) \in E} \in \mathbb{R}^{N \times N}$, where its diagonal entries are equal to the sum of the entries on the same row in W, i.e.,

$$h(\boldsymbol{x}, \boldsymbol{y}) := \begin{cases} \sum_{\boldsymbol{z} \in V} w(\boldsymbol{x}, \boldsymbol{z}), & \boldsymbol{x} = \boldsymbol{y}, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Let $\boldsymbol{u} = (u(\boldsymbol{x}))^\top_{\boldsymbol{x} \in V} \in \mathbb{R}^N$, i.e., an N-length column vector. Define the graph Laplacian as $L = D - W$, and the gradient operator ∇ on $u(\boldsymbol{x}), \forall \boldsymbol{x} \in V$, as

$$\nabla u(\boldsymbol{x}) := (w(\boldsymbol{x}, \boldsymbol{y})[u(\boldsymbol{x}) - u(\boldsymbol{y})])_{(\boldsymbol{x}, \boldsymbol{y}) \in E}. \tag{5}$$

The ℓ_1-norm of an N-length vector is defined as

$$\begin{aligned} \|\nabla \boldsymbol{u}\|_1 &:= \sum_{\boldsymbol{x} \in V} |\nabla u(\boldsymbol{x})| \\ &= \sum_{(\boldsymbol{x}, \boldsymbol{y}) \in E} |w(\boldsymbol{x}, \boldsymbol{y})[u(\boldsymbol{x}) - u(\boldsymbol{y})]|. \end{aligned} \tag{6}$$

The ℓ_2-norm (also known as Dirichlet energy) is defined as

$$\begin{aligned}\|\nabla \boldsymbol{u}\|_2^2 &:= \frac{1}{2}\boldsymbol{u}^\top L\boldsymbol{u} \\ &= \frac{1}{2}\sum_{(\boldsymbol{x},\boldsymbol{y})\in E} w(\boldsymbol{x},\boldsymbol{y})[u(\boldsymbol{x})-u(\boldsymbol{y})]^2. \end{aligned} \tag{7}$$

Note, however, that working with the fully connected graph E—like the settings in Eq. (5), (6) and (7)—can be highly computational demanding.

In order to reduce the computational burden, one often only considers the set of edges with large weights. In this paper, the k-nearest-neighbor (k-NN) of a point $\boldsymbol{x}$, $\mathcal{N}(\boldsymbol{x})$, is used to replace the whole edge set starting from the point $\boldsymbol{x}$ in E. Besides the computational saving, one additional benefit of using k-NN graph is its capability to capture the local property of points lying close to a manifold. With the k-NN graph, the definitions in Eq. (5), (6) and (7) become

$$\nabla u(\boldsymbol{x}) = (w(\boldsymbol{x},\boldsymbol{y})[u(\boldsymbol{x})-u(\boldsymbol{y})])_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})}, \tag{8}$$

$$\begin{aligned}\|\nabla \boldsymbol{u}\|_1 &:= \sum_{\boldsymbol{x}\in V}|\nabla u(\boldsymbol{x})| \\ &= \sum_{\boldsymbol{x}\in V}\sum_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})}|w(\boldsymbol{x},\boldsymbol{y})[u(\boldsymbol{x})-u(\boldsymbol{y})]|, \end{aligned} \tag{9}$$

and

$$\begin{aligned}\|\nabla \boldsymbol{u}\|_2^2 &:= \frac{1}{2}\boldsymbol{u}^\top L\boldsymbol{u} \\ &= \frac{1}{2}\sum_{\boldsymbol{x}\in V}\sum_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})} w(\boldsymbol{x},\boldsymbol{y})[u(\boldsymbol{x})-u(\boldsymbol{y})]^2, \end{aligned} \tag{10}$$

respectively, see e.g. [2, 12] for more detail.

3 Proposed data classification method

3.1 Preliminary

Given a point cloud V containing N points in $\mathbb{R}^M$. We aim to partition V into K classes $V_1, \cdots, V_K$ based on their similarities (the points in the same class possess high similarity), with a set of training points $T = \{T_j\}_{j=1}^K \subset V$, $|T| = N_T$. Note that $T_j \subset V_j$ for $j = 1, \ldots, K$. In other words, we aim to assign the points in $V \setminus T$ certain labels between 1 to K using the training set T in which the labels of points are known, and the partition satisfies no

vacuum and overlap constraint, i.e.,

$$V = \bigcup_{j=1}^{K} V_j \quad \text{and} \quad V_i \cap V_j = \emptyset, \quad \forall i \neq j, 1 \leq i, j \leq K. \tag{11}$$

In the rest of the paper, we denote the points in V needed to be labeled as $S = V \setminus T$, and call S the test set in V.

The constraint (11) can be described by a binary matrix function $U := (\boldsymbol{u}_1, \cdots, \boldsymbol{u}_K) \in \mathbb{R}^{N \times K}$ (also called partition matrix), with $\boldsymbol{u}_j = (u_j(\boldsymbol{x}))_{\boldsymbol{x} \in V}^{\top} \in \mathbb{R}^N : V \to \{0, 1\}$ defined as

$$u_j(\boldsymbol{x}) := \begin{cases} 1, & \boldsymbol{x} \in V_j, \\ 0, & \text{otherwise}, \end{cases} \qquad \forall \boldsymbol{x} \in V, j = 1, \ldots, K. \tag{12}$$

Clearly, the above definition yields $\sum_{j=1}^{K} u_j(\boldsymbol{x}) = 1, \forall \boldsymbol{x} \in V$. The constraint (12) is also known as the indicator constraint on the unit simplex. Since the binary representation in the constraint (12) generally requires solving a non-convex model with NP-hard issue, a common strategy—the convex unit simplex—is considered as an alternative

$$\begin{aligned} &\sum_{j=1}^{K} u_j(\boldsymbol{x}) = 1, \quad \forall \boldsymbol{x} \in V, \\ &\text{s.t.} \quad u_j(\boldsymbol{x}) \in [0, 1], \ j = 1, \ldots, K. \end{aligned} \tag{13}$$

Note, importantly, that the convex constraint (13) can overcome the NP-hard issue and make some subproblems convex, but generally the whole model can still be non-convex. Therefore, solving a model with constraint (11), (12) or (13) can be time consuming, see e.g. [2, 12] for more detail.

If a result satisfying the constraint (13) is not completely binary, a common way to obtain an approximate binary solution satisfying the constraint (12) is to select the binary function as the nearest vertex in the unit simplex by the magnitude of the components, i.e.,

$$\begin{aligned} &(u_1(\boldsymbol{x}), \cdots, u_K(\boldsymbol{x})) \mapsto \boldsymbol{e}_i, \\ &\text{where } i = \underset{j}{\operatorname{argmax}} \{u_j(\boldsymbol{x})\}_{j=1}^{K}, \forall \boldsymbol{x} \in V. \end{aligned} \tag{14}$$

Here, $\boldsymbol{e}_i$ is the K-length unit normal vector, which is 1 at the i-th component and 0 for all other components.

3.2 Proposed method

In this section, we present our novel method for data (e.g. point clouds) classification inspired by the SaT strategy which has been validated very effective in

image segmentation. Our method can be summarized briefly as follows: first, a classification result is obtained as a warm initialization by using a classical and fast, but need not be very accurate classification method such as SVM [51]; then, a proposed two-step iteration scheme is implemented until no change in the labels of the test points could be made between consecutive iterations. Specifically, at the first step, we propose to minimize a novel convex model free of constraint (cf. those constraints (11), (12) and (13)) to obtain a fuzzy partition, say U, while keeping the training labels unchanged. At the second step, a binary result is obtained by just applying the binary rule in Eq. (14) directly on the fuzzy partition obtained at the previous step. This binary result could be the final classification result for the original classification problem or, if necessary, be set as a new initialization to search a better one in the same manner. In the following, we give the details of each step.

Initialization. Given a point cloud V containing N points in $\mathbb{R}^M$ and training set T containing N_T points with correct labels, we use SVM, which is a standard and fast clustering method as an example, to obtain the first clustering. Let the partition matrix be $\hat{U} = (\hat{\boldsymbol{u}}_1, \cdots, \hat{\boldsymbol{u}}_K) \in \mathbb{R}^{N\times K}$, where $\hat{\boldsymbol{u}}_j = (\hat{u}_j(\boldsymbol{x}))^\top_{\boldsymbol{x}\in V} \in \mathbb{R}^N$ for $j = 1, \ldots, K$. One could acquire an initialization by any other methods which have better performance than SVM. If no proper method is available (e.g. the data set is too large), then an initialization generated by setting labels to the test points randomly can be used as an alternative.

Step one. We now put forward our convex model to find a fuzzy partition U with initialization $\hat{U} = (\hat{\boldsymbol{u}}_1, \cdots, \hat{\boldsymbol{u}}_K)$, i.e.,

$$\underset{U}{\operatorname{argmin}} \sum_{j=1}^{K} \left\{ \frac{\beta}{2} \|\boldsymbol{u}_j - \hat{\boldsymbol{u}}_j\|_2^2 + \frac{\alpha}{2} \boldsymbol{u}_j^\top L \boldsymbol{u}_j + \|\nabla \boldsymbol{u}_j\|_1 \right\}, \tag{15}$$

where the first term is the data fidelity term constraining the fuzzy partition not far away from the initialization; the second term is related to $\|\nabla \boldsymbol{u}\|_2^2$ with graph Laplacian L; the last term is the total variation constructed on the graph; and $\alpha, \beta > 0$ are regularization parameters. Specifically, the second term in model (15) is used to impose smooth features on the labels of the points, and the last term is used to force the points with similar information to group together.

It is worth emphasizing that we already have the labels on the points in the training set T, with $\bar{U} = (\bar{\boldsymbol{u}}_1, \cdots, \bar{\boldsymbol{u}}_K) \in \mathbb{R}^{N_T\times K}$ being the partition matrix on T, where $\bar{\boldsymbol{u}}_j = (\bar{u}_j(\boldsymbol{x}))^\top_{\boldsymbol{x}\in T} \in \mathbb{R}^{N_T}$ for $j = 1, \ldots, K$. Therefore, we only assign labels to points in the test set S, i.e., we have

$$\hat{u}_j(\boldsymbol{x}) = \bar{u}_j(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in T,\ j = 1, \ldots, K. \tag{16}$$

Let $\hat{\boldsymbol{u}}_{S_j}$ represent the part of $\hat{\boldsymbol{u}}_j$ defined on the test set S, and then we have

$$\hat{\boldsymbol{u}}_j = (\hat{\boldsymbol{u}}_{S_j}^\top, \bar{\boldsymbol{u}}_j^\top)^\top, \quad j = 1, \ldots, K. \tag{17}$$

Analogous notations are used for the partition matrix $U = (\boldsymbol{u}_1, \cdots, \boldsymbol{u}_K)$, with

$$\boldsymbol{u}_j = (\boldsymbol{u}_{S_j}^\top, \bar{\boldsymbol{u}}_j^\top)^\top, \quad j = 1, \ldots, K. \tag{18}$$

In Section 4, Eq. (17) and (18) are going to be used to derive an efficient algorithm to solve the minimization problem (15).

The following Theorem 1 proves that our proposed model (15) has a unique solution.

Theorem 1. *Given $\hat{U} \in \mathbb{R}^{N\times K}$ and $\alpha, \beta > 0$, the proposed model (15) has a unique solution $U \in \mathbb{R}^{N\times K}$.*

Proof According to [53, Chapter 9], we know that a strongly convex function has a unique minimum. The conclusion follows directly from the strong convexity of the proposed model (15). □

Many algorithms can be used to solve model (15) efficiently due to the convexity of the model without constraint. For example, the split-Bregman algorithm [54], which is specifically devised for ℓ_1 regularized problems; the primal-dual algorithm [55], which is designed to solve general saddle point problems; and the powerful alternative, ADMM algorithm [56]. In particular, model (15) actually contains K independent sub-minimization problems, where each corresponds to a labeling function $\boldsymbol{u}_j$, and therefore the parallelism strategy is ideal to apply. This is one of the important advantages of our method for large data sets. The algorithm aspects to solve our proposed convex model (15) are detailed in Section 4.

Step two. This step is to project the fuzzy partition result U obtained at step one to a binary partition. Here, formula (14) is applied to the fuzzy partition U to generate a binary partition, which naturally satisfies no vacuum and overlap constraint (11). We remark that compared to the computation time at step one, the time at step two is negligible.

Normally, the classification work is complete after we obtain a binary partition matrix at step two. However, since the way of obtaining an initialization in our scheme is open and the quality of the initialization could be poor, we suggest going back to step one with the latest obtained partition as a new initialization and repeating the above two steps until no more change in the partition matrix is observed. More precisely, we set U as $\hat{U}$ and repeat steps one and two again to obtain a new U. Then the final classification result is the converged stationary partition matrix, say U^*. Moreover, to accelerate the convergence speed, we update β in model (15) by a factor of 2 if we are to repeat the steps. This will obviously enforce the closeness between two consecutive clustering results during iterations, which will ensure the algorithm converges fast. We stop the algorithm when no changes are observed in the clustering result compared to the previous one. We remark that a few iterations (~ 10) are generally enough in practice, see the experimental results in Section 5 for more detail.

Note, importantly, that our classification method here is totally different from other variational methods like [2, 12] which need to minimize variational models with constraints like (11), (12), (13), or other kinds of constraints (e.g. minimum and maximum number of points imposed on individual classes V_i). Even though our proposed model (15) has no constraint, the final classification result of our method naturally satisfies the no vacuum and overlap constraint (11). Therefore, our method, namely SaT (inheriting the name of the SaT methodology) classification method for high-dimensional data, is much easier to solve in each iteration. Its whole procedure is summarized in Algorithm 1.

Algorithm 1 SaT classification method for high-dimensional data

Initialization: Generate initialization $\hat{U}$ by e.g. SVM method.
Output: Binary partition U^*.

For $l = 0, 1, \ldots,$ until the stopping criterion reached (e.g. $\|U^{(l)} - U^{(l+1)}\| = 0$)
Step one: Compute fuzzy partition U by solving model (15).
Step two: Compute binary partition $U^{(l+1)}$ by using formula (14) on U.
Set $\hat{U} = U^{(l+1)}$ and $\beta = 2\beta$.
Endfor
Set $U^* = U^{(l+1)}$.

4 Algorithm aspects

In this section, we present an algorithm to solve the proposed convex model (15) based on the primal-dual algorithm [55].

4.1 Primal-dual algorithm

Let X_i be a finite dimensional vector space equipped with a proper inner product $\langle \cdot, \cdot \rangle_{X_i}$ and norm $\|\cdot\|_{X_i}$, $i = 1, 2$. Let map $\mathcal{K} : X_1 \to X_2$ be a bounded linear operator. The primal-dual algorithm is, generally speaking, to solve the following saddle-point problem

$$\min_{\boldsymbol{x} \in X_1} \max_{\tilde{\boldsymbol{x}} \in X_2} \Big\{ \langle \mathcal{K}\boldsymbol{x}, \tilde{\boldsymbol{x}} \rangle + \mathcal{G}(\boldsymbol{x}) - \mathcal{F}^*(\tilde{\boldsymbol{x}}) \Big\}, \tag{19}$$

where $\mathcal{G} : X_1 \to [0, +\infty]$ and $\mathcal{F} : X_2 \to [0, +\infty]$ are proper, convex and lower-semicontinuous functions, and $\mathcal{F}^*$ represents the convex conjugate of $\mathcal{F}$. Given proper initializations, the primal-dual algorithm to solve problem (19) can be summarized in the following iterative way of updating the primal and dual variables, i.e.,

$$\tilde{\boldsymbol{x}}^{(l+1)} = (I + \sigma \partial \mathcal{F}^*)^{-1}(\tilde{\boldsymbol{x}}^{(l)} + \sigma \mathcal{K}\boldsymbol{z}^{(l)}), \tag{20}$$

$$\boldsymbol{x}^{(l+1)} = (I + \tau \partial \mathcal{G})^{-1}(\boldsymbol{x}^{(l)} - \tau \mathcal{K}^* \tilde{\boldsymbol{x}}^{(l+1)}), \tag{21}$$

$$\boldsymbol{z}^{(l+1)} = \boldsymbol{x}^{(l+1)} + \theta(\boldsymbol{x}^{(l+1)} - \boldsymbol{x}^{(l)}), \tag{22}$$

where $\theta \in [0,1], \tau, \sigma > 0$ are algorithm parameters.

4.2 Algorithm to solve our proposed model

We first define some useful notations which will be used to present our algorithm.

4.2.1 Preliminary

For ease of explanation, in the following, when we say $(i,j) \in E$, the i and j represent the i-th and j-th vertices in E, respectively. Let

$$E' = \big\{(i,j) \mid i < j, \forall (i,j) \in E\big\}. \tag{23}$$

The graph Laplacian $L = D - W \in \mathbb{R}^{N\times N}$ can be decomposed as

$$L = \sum_{(i,j)\in E'} L_{ij}, \tag{24}$$

where

$$L_{ij} = \begin{array}{c} \\ i \\ \\ j \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} & i & & j & \end{array} \\ \begin{pmatrix} & \vdots & & \vdots & \\ \cdots & w(i,j) & \cdots & -w(i,j) & \cdots \\ & \vdots & & \vdots & \\ \cdots & -w(i,j) & \cdots & w(i,j) & \cdots \\ & \vdots & & \vdots & \end{pmatrix} \end{array} \in \mathbb{R}^{N\times N} \tag{25}$$

is a matrix with only four nonzero entries which locate at positions $(i,i),(i,j),(j,i)$ and (j,j). Let $E' = E'_a \cup E'_b \cup E'_c$, where

$$E'_a = \big\{(i,j) \mid i,j \in S, \forall (i,j) \in E'\big\}, \tag{26}$$
$$E'_b = \big\{(i,j) \mid i,j \in T, \forall (i,j) \in E'\big\}, \tag{27}$$
$$E'_c = E' \setminus (E'_a \cup E'_b). \tag{28}$$

Then the decomposition L in Eq. (24) can be rewritten as

$$L = \sum_{(i,j)\in E'_a} L_{ij} + \sum_{(i,j)\in E'_b} L_{ij} + \sum_{(i,j)\in E'_c} L_{ij}. \tag{29}$$

Note that, the terms $\sum_{(i,j)\in E'_a} L_{ij}$ and $\sum_{(i,j)\in E'_b} L_{ij}$ only have nonzero entries which are associated to the test set S and the training set T, respectively. Let

$$\begin{aligned}
&\sum_{(i,j)\in E'_a} L_{ij} = \begin{pmatrix} L_S & 0 \\ 0 & 0 \end{pmatrix}, \quad \sum_{(i,j)\in E'_b} L_{ij} = \begin{pmatrix} 0 & 0 \\ 0 & \bar{L} \end{pmatrix}, \\
&\sum_{(i,j)\in E'_c} L_{ij} = \begin{pmatrix} L_1 & L_3 \\ L_3^\top & L_2 \end{pmatrix},
\end{aligned} \tag{30}$$

where $L_S, L_1 \in \mathbb{R}^{(N-N_T)\times(N-N_T)}$ are related to the test set S, $\bar{L}, L_2 \in \mathbb{R}^{N_T\times N_T}$ are related to the training set T, and $L_3 \in \mathbb{R}^{(N-N_T)\times N_T}$. Then we have

$$L = \begin{pmatrix} L_S + L_1 & L_3 \\ L_3^\top & \bar{L} + L_2 \end{pmatrix}. \tag{31}$$

According to Eq. (8), the gradient operator ∇ can be regarded as a linear transformation between $\mathbb{R}^N$ and $\mathbb{R}^{N\times(k-1)}$ (where $k = |\mathcal{N}(\boldsymbol{x})|$). For a vector $\boldsymbol{u}_j = (\boldsymbol{u}_{S_j}^\top, \bar{\boldsymbol{u}}_j^\top)^\top$ defined in Eq. (18), let

$$\begin{aligned}
&\mathcal{A}_S(\boldsymbol{u}_{S_j}) = \nabla \begin{pmatrix} \boldsymbol{u}_{S_j} \\ \mathbf{0} \end{pmatrix} \in \mathbb{R}^{N\times(k-1)}, \\
&H_j = \nabla \begin{pmatrix} \mathbf{0} \\ \bar{\boldsymbol{u}}_j \end{pmatrix} \in \mathbb{R}^{N\times(k-1)}.
\end{aligned} \tag{32}$$

Clearly, $\mathcal{A}_S : \mathbb{R}^{N-N_T} \to \mathbb{R}^{N\times(k-1)}$ is an operator corresponding to the test set S, and H_j is the gradient matrix corresponding to the training set T which is fixed since $\bar{\boldsymbol{u}}_j$ is fixed. Then, we have

$$\begin{aligned}
\nabla \boldsymbol{u}_j &= \nabla \begin{pmatrix} \boldsymbol{u}_{S_j} \\ \bar{\boldsymbol{u}}_j \end{pmatrix} = \nabla \begin{pmatrix} \boldsymbol{u}_{S_j} \\ \mathbf{0} \end{pmatrix} + \nabla \begin{pmatrix} \mathbf{0} \\ \bar{\boldsymbol{u}}_j \end{pmatrix} \\
&= \mathcal{A}_S(\boldsymbol{u}_{S_j}) + H_j.
\end{aligned} \tag{33}$$

4.2.2 Algorithm

Substituting the decomposition of L in Eq. (31), ∇ in Eq. (33), $\hat{\boldsymbol{u}}_j$ in Eq. (17) and $\boldsymbol{u}_j$ in Eq. (18) into the proposed minimization model (15) yields

$$\begin{aligned}
\underset{\{\boldsymbol{u}_{S_j}\}_{j=1}^K}{\operatorname{argmin}} \sum_{j=1}^{K} \Big\{ &\frac{\beta}{2}\|\hat{\boldsymbol{u}}_{S_j} - \boldsymbol{u}_{S_j}\|_2^2 + \frac{\alpha}{2}\boldsymbol{u}_{S_j}^\top L_S \boldsymbol{u}_{S_j} \\
&+ \alpha \boldsymbol{u}_{S_j}^\top L_3 \bar{\boldsymbol{u}}_j + \|\mathcal{A}_S(\boldsymbol{u}_{S_j}) + H_j\|_1 \Big\}.
\end{aligned} \tag{34}$$

Note, obviously, that solving the above model (34) is equivalent to solving K sub-minimization problems corresponding to each $\boldsymbol{u}_{S_j}$, $j = 1, \ldots, K$,

indicating that our proposed model inherently benefits from the parallelism computation. For $1 \leq j \leq K$, let

$$\mathcal{G}_j(\boldsymbol{u}_{S_j}) = \frac{\beta}{2}\|\hat{\boldsymbol{u}}_{S_j} - \boldsymbol{u}_{S_j}\|_2^2 + \frac{\alpha}{2}\boldsymbol{u}_{S_j}^\top L_S \boldsymbol{u}_{S_j} + \alpha \boldsymbol{u}_{S_j}^\top L_3 \bar{\boldsymbol{u}}_j, \tag{35}$$

$$\mathcal{F}_j(\tilde{\boldsymbol{x}}) = \|\tilde{\boldsymbol{x}} + H_j\|_1. \tag{36}$$

Using the definition of the ℓ_1-norm given in Eq. (9), the conjugate of $\mathcal{F}_j$, $\mathcal{F}_j^*$, can then be calculated as

$$\begin{aligned}\mathcal{F}_j^*(\boldsymbol{p}) &= \sup_{\tilde{\boldsymbol{x}} \in \mathbb{R}^{N\times(k-1)}} \langle \tilde{\boldsymbol{x}}, \boldsymbol{p}\rangle - \|\tilde{\boldsymbol{x}} + H_j\|_1 \\ &= -\langle \boldsymbol{p}, H_j\rangle + \chi_P(\boldsymbol{p}),\end{aligned} \tag{37}$$

where $P = \{\boldsymbol{p} \in \mathbb{R}^{N\times(k-1)} : \|\boldsymbol{p}\|_\infty \leq 1\}$, and $\chi_P(\boldsymbol{p})$ is the characteristic function of set P with value 0 if $\boldsymbol{p} \in P$, otherwise $+\infty$.

Using the primal-dual formulation (19) with the definitions of $\mathcal{G}_j$ and $\mathcal{F}_j^*$ respectively given in Eq. (35) and (37), then the minimization problem (34) corresponding to each $\boldsymbol{u}_{S_j}$ can be reformulated as

$$\operatorname*{argmin}_{\boldsymbol{u}_{S_j}} \max_{\boldsymbol{p}} \Big\{ \langle \mathcal{A}_S(\boldsymbol{u}_{S_j}), \boldsymbol{p}\rangle + \mathcal{G}_j(\boldsymbol{u}_S) + \langle \boldsymbol{p}, \boldsymbol{h}_j\rangle - \chi_P(\boldsymbol{p}) \Big\}. \tag{38}$$

To apply the primal-dual method, it remains to compute $(I + \sigma\partial\mathcal{F}_j^*)^{-1}$ and $(I + \tau\partial\mathcal{G}_j)^{-1}$. Firstly, for $\forall \tilde{\boldsymbol{x}} \in \mathbb{R}^{N\times(k-1)}$, we have

$$\begin{aligned}&(I + \sigma\partial\mathcal{F}^*)^{-1}(\tilde{\boldsymbol{x}}) \\ =& \operatorname*{argmin}_{\boldsymbol{p}\in\mathbb{R}^{N\times(k-1)}} \mathcal{F}_j^*(\boldsymbol{p}) + \frac{1}{2\sigma}\|\boldsymbol{p} - \tilde{\boldsymbol{x}}\|_2^2 \\ =& \operatorname*{argmin}_{\boldsymbol{p}\in\mathbb{R}^{N\times(k-1)}} \chi_P(\boldsymbol{p}) + \frac{1}{2\sigma}\|\boldsymbol{p} - \tilde{\boldsymbol{x}}\|_2^2 - \langle \boldsymbol{p}, H_j\rangle \\ =& \operatorname*{argmin}_{\boldsymbol{p}\in\mathbb{R}^{N\times(k-1)}} \chi_P(\boldsymbol{p}) + \frac{1}{2\sigma}\|\boldsymbol{p} - \tilde{\boldsymbol{x}} - \sigma H_j\|_2^2 \\ =& \iota_P(\tilde{\boldsymbol{x}} + \sigma H_j),\end{aligned} \tag{39}$$

where the operator $\iota_P(\cdot)$ is the pointwise projection operator onto the set P, i.e., $\forall p \in \mathbb{R}$,

$$\iota_P(p) = \begin{cases} 1, & |p| > 1, \\ p, & \text{otherwise.} \end{cases} \tag{40}$$

Secondly, for $\forall \boldsymbol{x} \in \mathbb{R}^{N-N_T}$, we have

$$(I + \tau\partial\mathcal{G}_j)^{-1}(\boldsymbol{x})$$

$$= \underset{\boldsymbol{u}_{S_j} \in \mathbb{R}^{N-N_T}}{\operatorname{argmin}} \mathcal{G}_j(\boldsymbol{u}_{S_j}) + \frac{1}{2\tau}\|\boldsymbol{u}_{S_j} - \boldsymbol{x}\|_2^2. \tag{41}$$

Using the definition of $\mathcal{G}_j(\boldsymbol{u}_{S_j})$ given in Eq. (35), problem (41) becomes solving the following linear system

$$(\alpha L_S + \beta I + \frac{1}{\tau} I)\boldsymbol{u}_{S_j} = \beta \hat{\boldsymbol{u}}_{S_j} + \frac{1}{\tau}\boldsymbol{x} - \alpha L_3 \bar{\boldsymbol{u}}_j. \tag{42}$$

Since $(\alpha \bar{L} + \beta I + \frac{1}{\tau} I)$ is positive definite, the above linear system can be solved efficiently by e.g. conjugate gradient method [57].

Finally, by exploiting the strong convexity of $\mathcal{G}_j, \forall 1 \leq j \leq K$, which is shown in the next lemma, the work in [55] suggests that we could adaptively modify σ, τ to accelerate the convergence or the primal-dual method.

Lemma 2. *The functions $\mathcal{G}_j, \forall 1 \leq j \leq K$, are strongly convex with parameter β.*

Proof For simplicity, we omit the subscript j and S_j in the following proof. First, by Eq. (30), L_S is semi-positive definite. Therefore, $(\frac{\alpha}{2}\boldsymbol{u}^\top L_S \boldsymbol{u} + \alpha \boldsymbol{u}^\top L_3 \bar{\boldsymbol{u}})$ is convex. Now the strong convexity of $\mathcal{G}$ follows from the fact that the remaining term in Eq. (35), which is $\frac{\beta}{2}\|\boldsymbol{u} - \hat{\boldsymbol{u}}\|_2^2$, is strongly convex with parameter β. □

The algorithm solving our proposed classification model (34) (i.e., model (15)) is summarized in Algorithm 2. Its convergence proof is given in Theorem 3 below. For each sub-minimization problem, the relative error between two consecutive iterations and/or a given maximum iteration number can be used as stopping criteria to terminate the algorithm. Finally, we emphasize again that our method is quite suitable for parallelism since the K sub-minimization problems are independent of each other and therefore can be computed in parallel.

Theorem 3. *Algorithm 2 converges if $\tau^{(0)}\sigma^{(0)} < \frac{1}{N^2(k-1)}$.*

Proof By Theorem 2 in [55], Algorithm 2 converges as long as $\|\mathcal{A}_S\|_2^2 < \frac{1}{\tau^{(0)}\sigma^{(0)}}$. Therefore, it suffices to find a suitable upper bound for $\|\mathcal{A}_S\|_2$. By our implementation in Eq. (32) and since the weight functions Eq. (1)–(3) take values in $[-1, 1]$, each entry in $\mathcal{A}_S$ is between $[-1, 1]$. Therefore, the 1-norm and ∞-norm of $\mathcal{A}_S$ can be easily estimated as

$$\|\mathcal{A}_S\|_1 = \max_{1 \leq j \leq N-N_T} \sum_{i=1}^{N(k-1)} |(\mathcal{A}_S)_{ij}| \leq N(k-1)$$

and

$$\|\mathcal{A}_S\|_\infty = \max_{1 \leq i \leq N(k-1)} \sum_{j=1}^{N-N_T} |(\mathcal{A}_S)_{ij}| \leq N - N_T.$$

Algorithm 2 Algorithm solving the proposed model (34) (i.e., model (15))

Initialization: $\tilde{\boldsymbol{x}}^{(0)} \in \mathbb{R}^{N\times(k-1)}$, $\boldsymbol{x}^{(0)}, \boldsymbol{z}^{(0)} \in \mathbb{R}^{N-N_T}$, $\theta \in [0,1], \tau^{(0)}, \sigma^{(0)} > 0$.
Output: $\{\boldsymbol{u}_{S_j}\}_{j=1}^{K}$.

For $j = 1, \cdots, K$ (parallelism strategy can be applied)
  For $l = 0, 1, \ldots,$ until the stopping criterion reached
    Let $\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{x}}^{(l)} + \sigma^{(l)}\mathcal{A}_S\boldsymbol{z}^{(l)}$, and compute $\tilde{\boldsymbol{x}}^{(l+1)} = (I + \sigma^{(l)}\partial\mathcal{F}^*)^{-1}(\tilde{\boldsymbol{x}})$ by Eq. (39);
    Let $\boldsymbol{x} = \boldsymbol{x}^{(l)} - \tau^{(l)}\mathcal{A}_S^*\tilde{\boldsymbol{x}}^{(l+1)}$, and compute $\boldsymbol{x}^{(l+1)} = (I + \tau^{(l)}\partial\mathcal{G})^{-1}(\boldsymbol{x})$ by Eq. (41);
    Let $\theta^{(l)} = 1/\sqrt{1+\beta\tau^{(l)}}$, and set $\tau^{(l+1)} = \theta^{(l)}\tau^{(l)}, \sigma^{(l+1)} = \sigma^{(l)}/\theta^{(l)}$;
    Compute $\boldsymbol{z}^{(l+1)} = \boldsymbol{x}^{(l+1)} + \theta(\boldsymbol{x}^{(l+1)} - \boldsymbol{x}^{(l)})$;
  Endfor
  Set $\boldsymbol{u}_{S_j} = \boldsymbol{x}^{(l+1)}$.
Endfor

We then have

$$\|\mathcal{A}_S\|_2 \le \sqrt{\|\mathcal{A}_S\|_1\|\mathcal{A}_S\|_\infty} \le N\sqrt{k-1}.$$

Therefore, we conclude that the algorithm converges as long as we choose $\tau^{(0)}, \sigma^{(0)} > 0$, such that $\tau^{(0)}\sigma^{(0)} < \frac{1}{N^2(k-1)}$. □

The convergence of Algorithm 2 proved in Theorem 3 ensures the convergence of *step one* of our proposed method (i.e., Algorithm 1). After the binary partition *step two* at the l-th iteration of Algorithm 1, we have $\hat{U} = U^{(l+1)}$ and $\beta = 2\beta$. The increasing regularization parameter β will lead to the dominance of the first term of our model (15), which yields $\|U - \hat{U}\| \to 0$ when l becomes large; in particular, this will finally lead to the satisfaction of the stopping criterion $\|U^{(l)} - U^{(l+1)}\|$ of Algorithm 1. Extensive experiments in Section 5 will show that our Algorithm 1 converges very quickly, e.g. generally no more than ten iterations (i.e., $l \le 10$); see Fig. 4 for the convergence history.

5 Numerical results

In this section, we evaluate the performance of our proposed method on four benchmark data sets—including Three Moon, COIL, Opt-Digits and MINST—for semi-supervised learning. Three Moon is a synthetic data set which has been used frequently e.g. in [2, 7, 12]. The COIL, Opt-Digits, and MNIST data sets can be found in the supplementary material of [58], the UCI machine learning repository[1], and the MNIST Database of Handwritten Digits[2], respectively.

The basic properties of these test data sets are shown in Table 1. It indicates that the number of classes in these data sets ranges from small to large (i.e., 3 to 10), which is analogous to the dimensions and number of points. The

[1] http://archive.ics.uci.edu/ml/datasets.html
[2] http://yann.lecun.com/exdb/mnist/

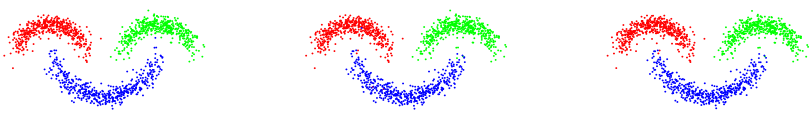
individual points in these data sets may have no texture/feature information like those in THREE MOON or may be images with low resolution like those in COIL, OPT-DIGITS and MNIST. In particular, the number of labeled points (see details below) may be very small, e.g. less than 1% of the given data set, and significantly unbalanced across the classes.

Table 1 Basic properties of the test benchmark data sets. "Dim." means dimension, i.e., the length of every vector representing individual points in the given data sets.

Data set	No. of classes	Dim.	No. of points
THREE MOON	3	100	1500
COIL	6	241	1500
OPT-DIGITS	10	64	5620
MNIST	10	784	70000

To implement our method, k-NN graphs are constructed for the test data sets, using the randomized kd-tree [59] to find the nearest neighbors with Euclidean distance as the metric. The radial basis function (1) is used to compute the weight matrix W, except for the MNIST data set where the Zelnic-Manor and Perona weight function (2) is used with the eight closest neighbors. The training samples T—samples with labels known—are selected randomly from each test data set. The classification accuracy is defined as the percentage of correctly labeled data points.

Unless otherwise specified, the regularization parameter α is fixed to 1 and the regularization parameter β is initially fixed to 0.01. Indeed, this combination of α, β provides good results for almost all data sets; moreover, the results are robust for $\alpha \in [0.5, 2]$ and for $\beta \in [0.001, 0.1]$. The choice of initial β needs fine-tuning for the COIL data set, i.e., a much smaller initial β is required to achieve reasonable accuracy. We comment that the accuracy of the proposed method can be improved further after fine-tuning the values of α and β for individual test data sets. The ways of selecting the optimized parameters are, however, beyond the scope of this work and will be conducted in future investigation. All the codes were implemented in MATLAB 2017a and run on a MacBook with 2.8 GHz processor and 16 GB RAM.



(a) Ground truth (b) TVRF (c) Our method

Fig. 1 Three-class classification for the THREE MOON synthetic data. (a): Ground truth; (b) and (c): results of method TVRF [2] and our proposed method, respectively.

5.1 Methods comparison

As mentioned in previous sections, we use the SVM method [51] to generate initializations for our proposed method. If it is not proper for a data set (e.g. very slow due to the large size of the data set), we could just use an initialization generated by assigning clustering labels randomly.

The SVM is a technique aiming to find the best hyperplane that separates data points of one class from the others. In practice, data may not be separable by a hyperplane. In that case, soft margin is used so that the hyperplane would separate many data points if not all. It is also common to kernelize data points, and then find a separating hyperplane in the transformed space. The SVM method used in our experiments is trained with a linear kernel.

The properties shown in Table 1, including the individual points with no clear texture/feature information and lack of labeled points, justify the necessity and importance of the methods based on variational models like this work, which can exploit the structure of the whole data set except for the individual points, against another type of methods based on deep learning which generally require the individual points to have rich texture/feature information and quite a large number of training points/samples. In such sense, the methods based on deep learning are not the main focus of this paper and will not be included for comparison here.

We compare our proposed method with the state-of-the-art methods proposed recently, e.g. CVM [1], GL [7], MBO [7], TVRF [2], LapRF [2], LapRLS [60], MP [60], and SQ-Loss-I [58]. The code TVRF was provided by the authors and the parameters used in it were chosen by trial and error to give the best results. The classification accuracies of methods GL, MBO, LapRF, LapRLS, MP and SQ-Loss-I were taken from [1, 2], in which methods CVM and TVRF were shown to be superior in most of the cases.

5.2 Three Moon data

The synthetic Three Moon data used here is constructed by following the way performed in [1, 2] exactly. We briefly repeat the procedure as follows. First, generate three half circles in $\mathbb{R}^2$—two half top unit circles and one half bottom circle with radius of 1.5 which are centered at (0, 0), (3, 0) and (1.5, 0.4), respectively. Then 500 points are uniformly sampled from each half circle and embedded into $\mathbb{R}^{100}$ by appending zeros to the remaining dimensions. Finally, an i.i.d. Gaussian noise with standard deviation 0.14 is added to each dimension of the data. An illustration of the first two dimensions of the Three Moon data is shown in Fig. 1 (a) where different colors are applied on each half circle. This is a three-class classification problem with the goal of classifying each half circle using a small number of supervised points from each class. This classification problem is challenging due to the noise and the high dimensionality of all the points with high similarity in $\mathbb{R}^{98}$.

A k-NN graph with $k = 10$ is built for this data set, parameter $\xi = 3$ is used in the Gaussian weight function, and the distance metric chosen is Euclidean

Table 2 Accuracy comparison for the Three Moon synthetic data set, with uniformly selected training points.

Method	Accuracy(%)
CVM	98.7
GL	98.4
MBO	99.1
TVRF	98.6
LapRF	98.4
Proposed	**99.4**

Table 3 Accuracy comparison for the Three Moon synthetic data set, with non-uniformly selected training points.

Method	Accuracy(%)
TVRF	97.8
Proposed	**99.3**

metric for $\mathbb{R}^{100}$. We first test the methods using uniformly distributed supervised points, where a total number of 75 points is sampled uniformly from this data set as training points.

The accuracies of method TVRF and ours are obtained by running the methods ten times with randomly selected labeled samples, and taking the average of the accuracies. The accuracies of method CVM are obtained from the original paper [1]. The accuracy comparison is reported in Table 2, showing that our proposed method gives the highest accuracy; see also Fig. 1 for visual validation of the results between methods of TVRF and ours. The average number of iterations taken for our proposed method is 3.8. Fig. 4 (a) gives the convergence history and partition accuracy of our proposed method corresponding to iteration steps, which clearly shows the accuracy increment during iterations (note that the accuracy at iteration 0 is the result of the initialization which is obtained by the SVM method). Table 7 reports the comparison in terms of computation time, indicating the superior performance of our proposed method in computation speed.

In the following, as a showcase, we test the methods using *non-uniformly* distributed supervised points, which is used to investigate the robustness of these methods on training points. In this case for the 75 training points, as an example, we respectively pick 5 points from the left and the bottom half circles, and pick the rest 65 points from the right half circle. This sampling is illustrated in Fig. 2.

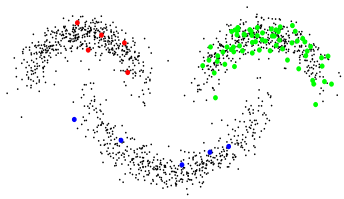


Fig. 2 Unbalanced sampling from the Three Moon data, where sampled points are highlighted with their corresponding labels.

The accuracies of TVRF and our method are shown in Table 3, from which we see clearly that our method gives much higher accuracy. The standard

deviation of the accuracy for our method is 0.11%. In particular, compared to the results in Table 2 using training points selected uniformly, the accuracy of TVRF decreases by 0.8%, whereas we observe only a very small decrease (i.e., 0.1%) in our proposed method. This shows the robustness of our method with respect to the way that training points are selected. Note that in the case of training points chosen non-uniformly, the initialization obtained by SVM is poor, because of which more iterations are needed to converge for our method—average 12.0 iterations in 10 trials versus 3.3 iterations needed for the case of training points selected uniformly.

5.3 COIL data

The benchmark COIL data comes from the Columbia object image library. It contains a set of color images of 100 different objects. These images, with size of 128×128 each, are taken from different angles in steps of 5 degrees, i.e., 72 ($= 360/5$) images for each object. In the following, without loss of generality, we also call an image a point for ease of reference. The test data set here is constructed the same way as depicted in e.g. [1, 2] and is briefly described as follows. First, the red channel of each image is down-sampled to 16×16 pixels by averaging over blocks of 8×8 pixels. Then, 24 out of the 100 objects are randomly selected, which amounts to 1728 ($= 24 \times 360/5$) images. After that, these 24 objects are partitioned into six classes with four objects—288 images ($= 4 \times 72$)—in each class. Finally, after discarding 38 images randomly from each class, a data set of 1500 images where 250 images in each of the six classes are constructed. To construct a graph, each image, which is a vector with length of 241 after randomly masking (i.e., removing) 15 pixels from the original 256 ($= 16 \times 16$) pixels (see [58, Algorithm 21.1]), is treated as a node on the graph.

For accuracy test, a k-NN graph with $k = 4$ is built for this data set, parameter $\xi = 250$ is used in the Gaussian weight function, and the distance metric chosen is Euclidean metric for $\mathbb{R}^{241}$. The initial β is chosen as 10^{-5}. The training points, amount to 10% of the points, are selected randomly from the data set. Again, we run the test methods 10 times and compare the average accuracy. The resulting accuracy listed in Table 4 shows that our method outperforms other methods. The standard deviation of the accuracy for our method is 0.84%. Moreover, the average number of iterations of our method is 12.2. Fig. 4 (b) gives the convergence history of our proposed method in partition accuracy corresponding to iterations, which again shows an increasing trend in accuracy.

5.4 MNIST data

The MNIST data set consists of 70,000 images of handwritten digits 0–9, where each image has a size of 28×28. Fig. 3 shows some images of the ten digits from the data set. Each image is a node on a constructed graph. The objective is to classify the data set into 10 disjoint classes corresponding to different

Table 4 Accuracy comparison for the COIL data set, with uniformly selected training points.

Method	Accuracy(%)
CVM	93.3
TVRF	92.5
LapRF	87.7
GL	91.2
MBO	91.5
Proposed	**94.0**

Table 5 Accuracy comparison for the MINST data set, with uniformly selected training points.

Method	Accuracy(%)
CVM	**97.7**
TVRF	96.9
LapRF	96.9
GL	96.8
MBO	96.9
Proposed	97.4

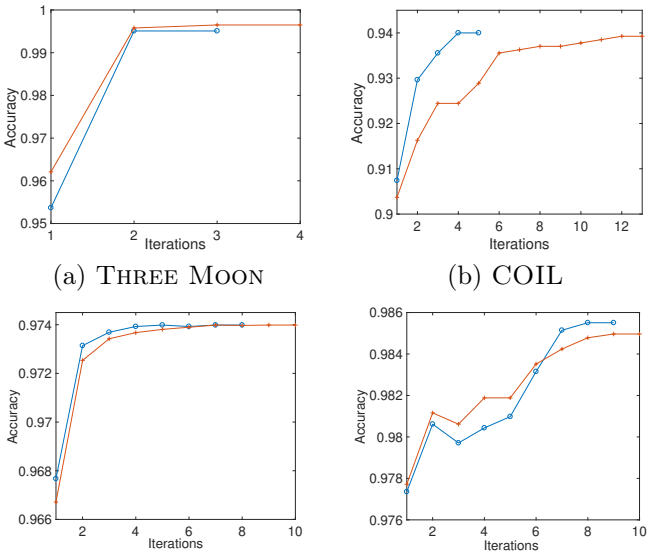
digits. For accuracy test, a k-NN graph with $k = 8$ is built for this data set, and Zelnik-Manor and Perona weight function in Eq. (2) is used to compute the weight matrix. The training 2500 (i.e., 3.57%) points (images) are selected randomly from the total 70,000 points.

The experimental results of the test methods are obtained by running them 10 times with randomly selected training set with a fixed number of points 2500, and the average accuracy is computed for comparison. The accuracy of the test results is shown in Table 5, indicating that our method is comparable to or better than the state-of-the-art methods compared here. The standard deviation of the accuracy for our method is 0.03%. Table 7 shows the computation time comparison, from which we again see that our method is very competitive in computation speed. The convergence history of our proposed method in partition accuracy corresponding to iterations is given in Fig. 4 (c), which also demonstrates a clear increasing trend in accuracy.

Fig. 3 Examples of digits 0–9 from the MNIST data set.

Table 6 Accuracy comparison for the Opt-Digits data set, with uniformly selected training points.

Sample rate	0.89% (50)	1.78% (100)	2.67% (150)
k-NN	85.5	92.0	93.8
SGT	91.4	97.4	97.4
LapRLS	92.3	97.6	97.3
SQ-Loss-I	95.9	97.3	97.7
MP	94.7	97.0	97.1
TVRF	95.9	98.3	98.2
LapRF	94.1	97.7	98.1
Proposed	**97.0**	**98.4**	**98.5**



(a) Three Moon (b) COIL

(c) MINST (d) Opt-Digits

Fig. 4 Accuracy convergence history of our proposed method corresponding to iteration steps for all the test data sets. The training samples are uniformly selected in each class. Blue and orange curves correspond to cases with the least and the largest number of iterations among the 10 trials, respectively.

5.5 Opt-Digits data

The Opt-Digits data set is constructed as follows. It contains 5620 bitmaps of handwritten digits (i.e., 0–9). Each bitmap has the size of 32×32 and is divided into non-overlapping blocks of 4×4, and then the number of "on" pixels is counted in each block. Therefore, each bitmap corresponds to a matrix of 8×8 where each element is an integer in $[0, 16]$. The classification problem is to partition the data set into 10 classes.

For accuracy test, a k-NN graph with $k = 8$ is built for this data set, parameter $\xi = 30$ is used in the Gaussian weight function, and the distance metric chosen is Euclidean metric for $\mathbb{R}^{64}$. For the experiments on this data set, we generate three training sets respectively with the number of points 50, 100 and 150, which are all selected randomly. All the methods are run 10 times for each training set and the average accuracy is used for comparison. The quantitative results in accuracy are listed in Table 6, from which we see that our proposed method is consistently better than the state-of-the-art methods compared for all the cases. The standard deviations of the accuracy for our method are $1.25\%, 0.53\%$ and 0.28% corresponding to 50, 100 and 150 number of training points, respectively. We also observe the improvement of the accuracy of these methods w.r.t. the increasing number of points in the training set. Finally, we show the convergence history of our proposed method

in partition accuracy corresponding to iterations using the training set with 150 points in Fig. 4 (d), which again clearly shows an increasing trend in accuracy.

Table 7 Computation time comparison in seconds. The value in the brackets for our method represents the average number of iterations of the 10 trials. More computation time of the related methods can be found in [2], which indicates that the TVRF method is quite efficient among the methods compared, e.g., it is at least 10 times faster than the multi-class MBO [7]. (For the Opt-Digits data, we select 100 sample points.)

Method	Three Moon	COIL	MINST	Opt-Digits
TVRF	0.71	0.65	66.00	3.42
Proposed	**0.30** (3.3)	0.76 (11.7)	82.04 (9.4)	4.45 (9.3)

5.6 One-class classification

Apart from the aforementioned multi-class classification problem, our proposed method can also be naturally extended to tackle the one-class classification problem, also known as unary classification [52, 61]. The goal of one-class classification is to distinguish one specific class from the others by learning primarily from the specific class in the data set. We regard the specific class (the class of interest) as the true data, while the others as outliers. The goal then is to discriminate between the true data and outliers in the given data set. It is natural to treat the true data and outliers as two classes, where the main challenge now is the highly uneven sampling of these two classes.

We test the performance of our proposed method on all of the above four data sets following the same parameter settings. Two types of tests are conducted for each data set, i.e., the ratios of 2:1 and 1:1 in the specific class and the outliers regarding the number of samples labeled uniformly in each class are applied. Table 8 summarizes the results in terms of classification accuracy of our proposed method (the accuracy of the TVRF method is withdrawn due to its inferior and unstable performance for unbalanced data set shown in Section 5.2). The accuracy after each iteration versus the iteration number for the four test data sets is given in Figure 5, which repeatedly shows an increasing trend in accuracy. It is evident that our proposed method consistently performs excellently in this problem even if the two classes—the specific class and the outliers—are extremely uneven, demonstrating the versatility and robustness of our proposed method in classification.

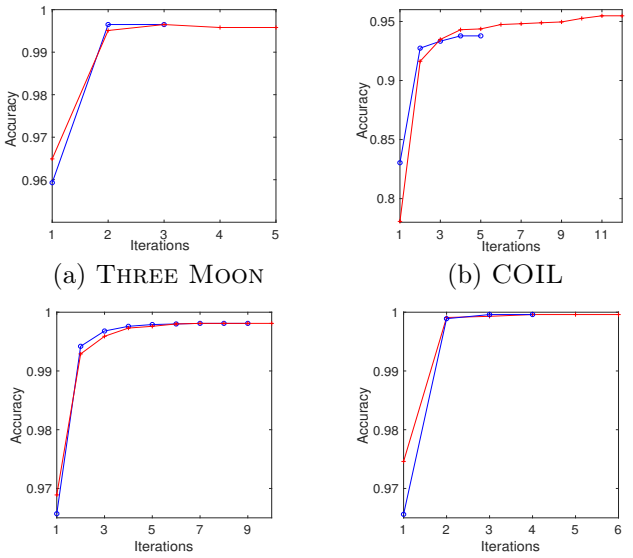
5.7 Further discussion

The above experimental results on the benchmark data sets in terms of classification accuracy, shown in Tables 2–6, indicate that our proposed method outperforms the state-of-the-art methods for high-dimensional data and point clouds classification.

Compared to the start-of-the-art variational classification models proposed e.g. in [1, 2], in addition to the data fidelity term and ℓ_1 term (e.g. TV), our

Table 8 One-class classification results of our proposed method. # true samples and # outlier samples represent the number of samples with labels in the specific class and the outliers, respectively.

Data set	# true samples : # outlier samples	Accuracy (%)
Three Moon	50 : 25 (= 2:1)	99.58
	38 : 37 (≈ 1:1)	99.62
COIL	100 : 50 (= 2:1)	91.30
	75 : 75 (= 1:1)	94.42
MNIST	1667 : 833 (≈ 2:1)	99.80
	1250 : 1250 (= 1:1)	99.81
Opt-Digits	67 : 33 (≈ 2:1)	99.97
	50 : 50 (= 1:1)	99.97



(a) Three Moon (b) COIL

(c) MINST (d) Opt-Digits

Fig. 5 Accuracy convergence history for one-class classification using our proposed method corresponding to iteration steps for all the test data sets. The training samples are uniformly selected in each class. Blue and orange curves correspond to cases with the least and the largest number of iterations among the 10 trials, respectively.

proposed model (15) contains an additional ℓ_2 term on the labeling functions which is used to smooth the classification results so as to reduce the non-smooth artifact (the so-called staircase artifact in images) introduced by the ℓ_1 term. This is one reason that our method can generally achieve better results. Moreover, the warm initialization used in our method can also play a role in improving the classification quality. Apart from generating the initialization manually, any classification methods can practically be used to generate the initialization. Starting from the initialization, our proposed method can then

be applied to achieve a better classification result by improving the accuracy iteratively. Theoretically speaking, the poorer the quality of the initialization, the more iterations are needed for our method. Nevertheless, we found that even for poor initializations (e.g. the ones generated randomly), 20 iterations are already enough to achieve competitive results. Generally, no more than 15 iterations are needed when using an initialization computed by standard classification methods (e.g. SVM).

Another distinction of our proposed model compared to the variational classification models in e.g. [1, 2] is that there are no constraints on these labeling functions in our objective functional. In other words, in each iteration, we just need to find the minimizer of the objective functional corresponding to each labeling function, but these minimizers do not need to satisfy the constraint that their summation equal to 1. Therefore, the computation speed for every single iteration is improved in our method compared to other methods which have constraints. We emphasize again that, since minimizing each sub-problem with respect to each labeling function is irrelevant to minimizing the sub-problems with respect to other labeling functions, parallelism techniques can be used straightforwardly to further improve the computation performance of our algorithm. Theoretically, we just require $1/K$ of the computation time needed for the non-parallelism scheme. This will be extremely important for large data sets. From Table 7, we see that, for all the computation time of our method, when considering the effect of parallel computing, our method should be able to outperform the state-of-the-art methods by a large margin.

The efficiency, versatility and robustness of our proposed method have also been validated in the one-class classification problem. It is indeed that our proposed method can be used to deal with different types of data sets which have e.g. extremely small number of labeled samples where individual samples have little to none texture/feature information (e.g. the samples in the Three Moon data set which only contain the coordinate information). These are the scenarios that the methods based on deep learning generally struggle. Therefore, our proposed method in this sense can complement deep learning methods in classification rather than be mutually exclusive. In particular, it would be of great interest in the future to further investigate the integration of the variational methods including ours with deep learning methods, e.g., using variational methods to classify features extracted by deep learning methods, involving graph Laplacian in deep learning frameworks [62, 63], etc.

6 Conclusions

In this paper, an efficient and versatile multi-class semi-supervised method based on variational models is proposed for classifying high-dimensional data or unstructured point clouds. The method is inspired by the SaT strategy which has been shown very effective for segmentation problems such as gray or color images corrupted by different degradations. Starting with a proper

initialization which can be obtained by using any standard classification algorithm (e.g. SVM) or constructed by users, the first step of our method is to solve a convex variational model without constraint. Most importantly, our proposed model is a lot easier to solve than the state-of-the-art variational models (e.g. [1, 2]) for the point clouds classification problem since they all need no vacuum and overlap constraint (11) on the labeling functions in the unit simplex, which could make their models to be non-convex. The second step of our method is to find a binary partition via thresholding the smoothed result obtained from the first step. We proved that our proposed model has a unique solution and the derived primal-dual algorithm converges.

We tested our proposed method on four benchmark data sets and compared with the state-of-the-art methods. We also investigated the influence of the training sets selected uniformly and non-uniformly. For our method, different ways of generating initializations were implemented and validated. The performance of the proposed method on the one-class classification problem was also validated except for the multi-class problem. On the whole, the experimental results demonstrated that our method is superior in terms of classification accuracy and computation speed when parallel computing is considered. Our method is therefore an efficient and versatile classification method for data sets like high-dimensional data or unstructured point clouds.

Acknowledgments. This work of R. Chan is partially supported by HKRGC Grants No. CityU12500915, CityU14306316, HKRGC CRF Grant C1007-15G, and HKRGC AoE Grant AoE/M-05/12. This work of T. Zeng is partially supported by the National Natural Science Foundation of China under Grant 11671002, CUHK start-up and CUHK DAG 4053296, 4053342. We thank Prof. Xue-Cheng Tai, Dr Ke Yin, Dr Egil Bae and Prof. Ekaterina Merkurjev for providing the codes of their methods [1, 2].

Data availability. The data sets generated during the current study are available from the corresponding authors on reasonable request.

Declarations

Conflict of interest. The authors have no relevant financial or non-financial interests to disclose.

References

[1] Bae, E., Merkurjev, E.: Convex variational methods on graphs for multiclass segmentation of high-dimensional data and point clouds. Journal of Mathematical Imaging and Vision **58**(3), 468–493 (2017). https://doi.org/10.1007/s10851-017-0713-9

[2] Yin, K., Tai, X.-C.: An effective region force for some variational models for learning and clustering. Journal of Scientific Computing **74**, 175–196 (2018)

[3] Merkurjev, E., Bertozzi, A., Yan, X., Lerman, K.: Modified Cheeger and ratio cut methods using the Ginzburga-Landau functional for classification of high-dimensional data. Inverse Problems **33**(7), 074003 (2017)

[4] Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(8), 888–905 (2000)

[5] Lee, J., Cai, X., Lellmann, J., Dalponte, M., *et al.*: Individual tree species classification from airborne multisensor imagery using robust PCA. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing **9**(6), 2554–2567 (2016)

[6] Osting, B., White, C., Oudet, E.: Minimal Dirichlet energy partitions for graphs. SIAM Journal on Imaging Sciences **36**(4), 1635–1651 (2014)

[7] Garcia-Cardona, C., Merkurjev, E., Bertozzi, A.L., Flenner, A., Percus, A.G.: Multiclass data segmentation using diffuse interface methods on graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(8), 1600–1613 (2014). https://doi.org/10.1109/TPAMI.2014.2300478

[8] Merkurjev, E., Kostic, T., Bertozzi, A.L.: An MBO scheme on graphs for classification and image processing. SIAM Journal on Imaging Sciences **6**(4), 1903–1930 (2013)

[9] Elmoataz, A., Lezoray, O., Bougleux, S.: Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing. IEEE Transactions on Image Processing **17**, 1047–1060 (2008)

[10] Merkurjev, E., Bae, E., Bertozzi, A.L., Tai, X.-C.: Global binary optimization on graphs for classification of high-dimensional data. Journal of Mathematical Imaging and Vision **52**(3), 414–435 (2015)

[11] Bertozzi, A.L., Flenner, A.: Diffuse interface models on graphs for classification of high dimensional data. Multiscale Modeling and Simulation **10**(3), 1090–1118 (2012)

[12] Lézoray, O., Elmoataz, A., Ta, V.T.: Nonlocal PDEs on graphs for active contours models with applications to image segmentation and data clustering. IEEE International Conference on Acoustics, Speech and Signal Processing, 873–876 (2012)

[13] Merriman, B., Ruuth, S.J.: Diffusion generated motion of curves on surfaces. Journal of Computational Physics **225**(2), 2267–2282 (2007). https://doi.org/10.1016/j.jcp.2007.03.034

[14] Yu, S.X., Shi, J.: Multiclass spectral clustering. In: Proceedings Ninth IEEE International Conference on Computer Vision, pp. 313–3191 (2003). https://doi.org/10.1109/ICCV.2003.1238361

[15] Hein, M., Setzer, S.: Beyond spectral clustering - tight relaxations of balanced graph cuts. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 24, pp. 2366–2374 (2011)

[16] Bresson, X., Tai, X.-C., Chan, T.F., Szlam, A.: Multi-class transductive learning based on l1 relaxations of cheeger cut and mumford-shah-potts model. Journal of Mathematical Imaging and Vision **49**(1), 191–201 (2014). https://doi.org/10.1007/s10851-013-0452-5

[17] Cai, X., Chan, R.H., Zeng, T.: A two-stage image segmentation method using a convex variant of the Mumford-Shah model and thresholding. SIAM Journal on Imaging Sciences **6**(1), 368–390 (2013)

[18] Boykov, Y., Funka-Lea, G.: Graph cuts and efficient N-D image segmentation. International Journal of Computer Vision **70**(2), 109–131 (2006)

[19] Mumford, D., Shah, J.: Optimal approximations by piecewise smooth functions and associated variational problems. Communications on Pure and Applied Mathematics **42**(5), 577–685 (1989)

[20] Cremers, D., Rousson, M., Deriche, R.: A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. International Journal of Computer Vision **72**(2), 195–215 (2007)

[21] Cai, X.: Variational image segmentation model coupled with image restoration achievements. Pattern Recognition **48**, 2029–2042 (2015)

[22] Cai, X., Chan, R.H., Nikolova, M., Zeng, T.: A three-stage approach for segmenting degraded color images: Smoothing, lifting and thresholding (SLaT). Journal of Scientific Computing **72**(3), 1313–1332 (2017). https://doi.org/10.1007/s10915-017-0402-2

[23] Dong, B., Chien, A., Shen, Z.: Frame based segmentation for medical images. Communications in Mathematical Sciences **9**, 551–559 (2010). https://doi.org/10.4310/CMS.2011.v9.n2.a10

[24] Bar, L., Chan, T.F., Chung, G., Jung, M., Kiryati, N., Mohieddine, R., Sochen, N., Vese, L.A.: In: Scherzer, O. (ed.) Mumford and Shah model and its applications to image segmentation and image restoration, pp. 1095–1157. Springer, New York, NY (2011). https://doi.org/10.1007/978-0-387-92920-0_25. https://doi.org/10.1007/978-0-387-92920-0_25

[25] Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J., Osher, S.: Fast global minimization of the active contour/snake model. Journal of Mathematical Imaging and Vision **28**(2), 151–167 (2007)

[26] Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Transactions on Image Processing **10**(2), 266–277 (2001)

[27] Li, F., Ng, M., Zeng, T., Shen, C.: A multiphase image segmentation method based on fuzzy region competition. SIAM Journal on Imaging Sciences **3**(2), 277–299 (2010)

[28] Vese, L., Chan, T.F.: A multiphase level set framework for image segmentation using the Mumford and Shah model. International Journal of Computer Vision **50**(3), 271–293 (2002)

[29] Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena **60**(1), 259–268 (1992)

[30] Chambolle, A., Novaga, M., Cremers, D., Pock, T.: An introduction to total variation for image analysis. In: in Theoretical Foundations and Numerical Methods for Sparse Recovery, De Gruyter (2010)

[31] Cai, X., Chan, R.H., Schönlieb, C.-B., Steidl, G., Zeng, T.: Linkage between piecewise constant Mumford-Shah model and ROF model and its virtue in image segmentation. arXiv:1807.10194 (2018)

[32] Cai, X., Steidl, G.: Multiclass segmentation by iterated ROF thresholding. In: Heyden, A., Kahl, F., Olsson, C., Oskarsson, M., Tai, X.-C. (eds.) Energy Minimization Methods in Computer Vision and Pattern Recognition, pp. 237–250. Springer, Berlin, Heidelberg (2013)

[33] Chan, R.H., Yang, H., Zeng, T.: A two-stage image segmentation method for blurry images with Poisson or multiplicative Gamma noise. SIAM Journal on Imaging Sciences **7**(1), 98–127 (2014)

[34] Chan, T.F., Esedoglu, S., Nikolova, M.: Algorithms for finding global minimizers of image segmentation and denoising models. SIAM Journal on Applied Mathematics **66**(5), 1632–1648 (2006)

[35] He, Y., Hussaini, M.Y., Ma, J., Shafei, B., Steidl, G.: A new Fuzzy C-means method with total variation regularization for image segmentation of images with noisy and incomplete data. Pattern Recognition **45**, 3463–3471 (2012)

[36] Brown, E., Chan, T., Bresson, X.: Completely convex formulation of the Chan-Vese image segmentation model. International Journal of Computer

Vision **98**, 103–121 (2012)

[37] Lellmann, J., Schnörr, C.: Continuous multiclass labeling approaches and algorithms. SIAM Journal on Imaging Sciences **44**(4), 1049–1096 (2011)

[38] Pock, T., Chambolle, A., Cremers, D., Bischof, H.: A convex relaxation approach for computing minimal partitions. IEEE Conference on Computer Vision and Pattern Recognition, 810–817 (2009)

[39] Pock, T., Cremers, D., Bischof, H., Chambolle, A.: An algorithm for minimizing the mumford-shah functional. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 1133–1140 (2009)

[40] Yuan, J., Bae, E., Tai, X.-C., Boykov, Y.: A continuous max-flow approach to potts model. In: European Conference on Computer Vision, pp. 379–392 (2010)

[41] Bezdek, J.C., Ehrlich, R., Full, W.: FCM: The Fuzzy C-means clustering algorithm. Computers & Geosciences **10**(2-3), 191–203 (1984). https://doi.org/10.1016/0098-3004(84)90020-7

[42] Cai, X., Fitschen, J., Nikolova, M., Steidl, G., Storath, M.: Disparity and optical flow partitioning using extended Potts priors. Information and Inference: A Journal of the IMA **4**(1), 43–62 (2014)

[43] Bauer, B., Cai, X., Peth, S., Schladitz, K., Steidl, G.: Variational-based segmentation of bio-pores in tomographic images. Computers & Geosciences **98**, 1–8 (2017)

[44] Zhang, Y., Matuszewski, B., Shark, L., Moore, C.: Medical image segmentation using new hybrid level-set method. In: 2008 Fifth International Conference BioMedical Visualization: Information Visualization in Medical and Biomedical Informatics, pp. 71–76 (2008)

[45] Cai, X., Chan, R.H., Morigi, S., Sgallari, F.: Framelet-based algorithm for segmentation of tubular structures. In: Bruckstein, A.M., ter Haar Romeny, B.M., Bronstein, A.M., Bronstein, M.M. (eds.) Scale Space and Variational Methods in Computer Vision, pp. 411–422. Springer, Berlin, Heidelberg (2012)

[46] Cai, X., Chan, R.H., Morigi, S., Sgallari, F.: Vessel segmentation in medical imaging using a tight-frame–based algorithm. SIAM Journal on Imaging Sciences **6**(1), 464–486 (2013)

[47] Burnet, N., Scaife, J., Romanchikova, M., Thomas, S., et al.: Applying physical science techniques and CERN technology to an unsolved problem in radiation treatment for cancer: the multidisciplinary 'VoxTox' research

programme. CERN ideaSquare journal of experimental innovation **1**(1) (2017)

[48] Scaife, J., Harrison, K., Drew, A., *et al.*: Accuracy of manual and automated rectal contours using helical tomotherapy image guidance scans during prostate radiotherapy. Journal of Clinical Oncology **33**(7_suppl), 94 (2015)

[49] Cai, X., Schönlieb, C.-B., Lee, J., *et al.*: Automatic contouring of soft organs for image-guided prostate radiotherapy. Radiotherapy and Oncology **119**, 895–896 (2016)

[50] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(7), 881–892 (2002)

[51] Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3), 273–297 (1995)

[52] Khan, S., Madden, M.: One-class classification: taxonomy of study and review of techniques. The Knowledge Engineering Review **9**(3), 345–374 (2014)

[53] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York, NY, USA (2004)

[54] Goldstein, T., Osher, S.: The split Bregman method for l1-regularized problems. SIAM Journal on Imaging Sciences **2**(2), 323–343 (2009)

[55] Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. Journal of Mathematical Imaging and Vision **40**(1), 120–145 (2011)

[56] Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning **3**(1), 1–122 (2011)

[57] Chan, R.H., Ng, M.: Conjugate gradient method for Toeplitz systems. SIAM Review **38**, 427–482 (1996)

[58] Chapelle, O., Schlkopf, B., Zien, A.: Semi-Supervised Learning, 1st edn. The MIT Press, Cambridge, Massachusetts (2006)

[59] Silpa-Anan, C., Hartley, R.: Optimised KD-trees for fast image descriptor

matching. IEEE Conference on Computer Vision and Pattern Recognition, 1–8 (2008)

[60] Subramanya, A., Bilmes, J.: Semi-supervised learning with measure propagation. Journal of Machine Learning Research **12**, 3311–3370 (2011)

[61] Moya, M., Hush, D.: Network constraints and multi-objective optimization for one-class classification. Neural Networks **9**(3), 463–474 (1996)

[62] Wang, B., Luo, X., Li, Z., Zhu, W., Shi, Z., Osher, S.: Deep neural nets with interpolating function as output activation. In: Advances in Neural Information Processing Systems 32, (2018)

[63] Wang, B., Osher, S.: Graph interpolating activation improves both natural and robust accuracies in data-efficient deep learning. European Journal of Applied Mathematics **32**(3), 540–569 (2021)