Fick's Law Assisted Propagation for Semisupervised Learning

Chen Gong, Dacheng Tao, *Fellow, IEEE*, Keren Fu, and Jie Yang

***Abstract*— How to propagate the label information from labeled examples to unlabeled examples is a critical problem for graph-based semisupervised learning. Many label propagation algorithms have been developed in recent years and have obtained promising performance on various applications. However, the eigenvalues of iteration matrices in these algorithms are usually distributed irregularly, which slow down the convergence rate and impair the learning performance. This paper proposes a novel label propagation method called Fick's law assisted propagation (FLAP). Unlike the existing algorithms that are directly derived from statistical learning, FLAP is deduced on the basis of the theory of Fick's First Law of Diffusion, which is widely known as the fundamental theory in fluid-spreading. We prove that FLAP will converge with linear rate and show that FLAP makes eigenvalues of the iteration matrix distributed regularly. Comprehensive experimental evaluations on synthetic and practical datasets reveal that FLAP obtains encouraging results in terms of both accuracy and efficiency.**

***Index Terms*— Convergence rate, Fick's law of diffusion, label propagation, semisupervised learning (SSL).**

I. Introduction

In MANY practical tasks such as object recognition, multimedia retrieval and text categorization, the labeled examples are usually inadequate or expensive to obtain, yet a large amount of unlabeled examples are available. Though these massive unlabeled examples do not have labels, they may offer the prior of data distribution, which is beneficial to accurate classification when labeled examples are scarce. Semisupervised learning (SSL) [1] is one of the learning strategies that aim to exploit these abundant unlabeled examples for boosting learning performance. SSL has been intensively investigated by many researchers from different perspectives, and in the following we will review some typical approaches that are related to our work.

Manuscript received March 13, 2014; revised August 25, 2014; accepted November 23, 2014. Date of publication December 18, 2014; date of current version August 17, 2015. This work was supported in part by the National Natural Science Foundation of China under Grant 61273258, in part by the 973 Plan, China, under Grant 2015CB856004, and in part by the Australian Research Council under Projects FT-130101457, DP-140102164, and LP-140100569.

C. Gong is with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200240, China, and also with the Centre for Quantum Computation & Information Systems, Faculty of Engineering and IT, University of Technology, Sydney, 235 Jones Street, Ultimo, NSW 2007, Australia (e-mail: goodgongchen@sjtu.edu.cn).

D. Tao is with the Centre for Quantum Computation & Information Systems, Faculty of Engineering and IT, University of Technology, Sydney, 235 Jones Street, Ultimo, NSW 2007, Australia (e-mail: dacheng.tao@uts.edu.au).

K. Fu and J. Yang are with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: krsuper@sjtu.edu.cn; jieyang@sjtu.edu.cn).

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/TNNLS.2014.2376963

A. Related Work

It is well known that SSL algorithms should be conducted under correct assumptions, otherwise the unlabeled examples are likely to impair the performance significantly [1]. There are two basic assumptions commonly used by graph-based SSL, i.e., cluster assumption and manifold assumption. Most popular SSL algorithms are developed based on one of these two assumptions.

1) Cluster Assumption: Cluster assumption assumes that the classes are well-separated, such that the decision boundary falls into the low density area in the feature space. Joachims [2] proposed transductive support vector machines (TSVMs) by adapting traditional support vector machines (SVMs) to the transductive learning setting. In contrast to the traditional SVM, it is unknown whether an unlabeled example is on the right or wrong side of the decision boundary, so TSVM incorporated the *hat loss* rather than the *hinge loss* commonly adopted by SVMs. Similar to this idea, Fung and Mangasarian [3] proposed a concave semisupervised SVM in which they replaced L_2-regularization with the L_1-regularization. Wang *et al.* [4] developed semisupervised classification based on class membership by introducing the membership vector. However, these methods are nonprobabilistic, namely the label posterior probability cannot be obtained when making classification. Besides, the optimizations in these approaches are usually nonconcave and are difficult to solve.

2) Manifold Assumption: Manifold assumption explores the geometry of the data distribution by postulating that its support has the geometric structure of a Riemannian manifold. Most graph-based SSL approaches fall into this type. Graph-based SSL regards labeled and unlabeled examples as vertices on a graph and uses edges to describe the pairwise similarity between them. The label information can then be propagated from labeled examples to unlabeled examples through the edges. Graph-based approaches have attracted intensive attention in recent years due to their promising performance and ease of implementation. The manifold assumption requires that the labels should vary smoothly on the graph. That is, the two examples connected by a strong edge tend to share similar labels. Therefore, graph-based methods often formulate SSL as optimization problems to effectively penalize the drastic variations of labels along the manifold. Graph cut algorithms [5] conduct SSL by finding

a graph partition to classify all the examples. Zhu *et al.* [6] and Zhou and Bousquet [7] exploited the traditional Graph Laplacian and normalized Graph Laplacian, respectively, to describe the smoothness of labels on the graph.

However, above methods simply deal with SSL in the traditional $\mathcal{L}^2$ space, so they can only obtain the discrete results on every example rather than the explicit formulation of decision boundary in the whole feature space. Therefore, they lack the ability to handle the unseen data. To address this problem, Belkin *et al.* [8] proposed data-dependent manifold regularization in the continues a reproducing kernel Hilbert space by adopting the extended representer theorem. This method was extended in [9] for scalable manifold regularization. The idea of manifold regularization was successfully applied to feature selection [10] and dimension reduction [11].

Different from above methods that only utilize pairwise information of examples for transduction, Wu and Schölkopf [12] developed the local learning regularizer to predict each example's label from those of its neighbors. Besides, Wang *et al.* [13] developed linear neighborhood propagation (LNP) by assuming that every example on the graph can be optimally reconstructed by its neighbors. These algorithms usually assume that the neighbors are deterministic to the label of a test example. However, this assumption does not always hold for various data distributions, so above local methods may obtain unsatisfactory results sometimes.

B. Motivation

The target of this paper is to design a new graph-based SSL algorithm based on the manifold assumption. In contrast to existing graph-based SSL algorithms that are derived from the perspective of statistical learning, this paper explains label propagation by Fick's First Law of Diffusion in fluid mechanics and presents a new SSL scheme, Fick's law assisted propagation (FLAP). In particular, FLAP simulates the diffusion of fluid for label propagation, thus the labeled examples can be regarded as the diffusive source with a high concentration of label information. When the diffusion process starts, the flux of label information will be transferred from the labeled examples to the remaining unlabeled examples. When the diffusion process is completed, all the examples on the graph will receive a certain concentration of label information, providing the foundation for final classification. Note that FLAP is more lifelike because it is straightforwardly derived from statistical physics. As a result, when and how much label information is received or transferred by an example, or where these labels should be propagated to, are directly governed by the well-known Fick's law, which is better than decided via some heuristic and ad hoc requirements or criteria exploited in conventional machine learning algorithms.

As a kind of iteration-based algorithm, it is proven that FLAP can converge more quickly than other iterative methods by analyzing the relationship between the convergence rate and the eigenvalues of the iteration matrix. We show that eigenvalues of the iteration matrix in FLAP are close to 1, while those in other methods may scatter in a wide range. This difference makes FLAP is superior to other iterative methods in terms of convergence speed. We conduct the experiments on several computer vision and pattern recognition repositories, including handwritten digit recognition, face recognition and teapot image classification. Thorough empirical studies show FLAP obtains promising performance by comparing with Minimum Cut (MinCut) [5], Harmonic Functions (HF) [6], Local and Global Consistency (LGC) [7], Linear Neighbourhood Propagation (LNP) [13], and Nonparametric Transforms of Kernels (NTK) [14].

II. Model Description

Fick's First Law of Diffusion governs mass transfer through diffusive means and has been widely used to understand the diffusion in solids, liquids, and gases. It postulates that the flux diffuses from regions of high concentration to regions of low concentration, with a magnitude that is proportional to the concentration gradient. Along one diffusion direction, the law is

$$J = -\gamma \frac{\partial \widetilde{f}}{\partial d} \tag{1}$$

where γ is the diffusion coefficient, d is the diffusion distance, $\widetilde{f}$ is the concentration that evaluates the density of molecules of fluid, and J is the diffusion flux that measures the quantity of molecules flowing through the unit area per unit time.

Given a set of labeled examples $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and a set of unlabeled examples $U = \{(\mathbf{x}_i)\}_{i=l+1}^{l+u}$, where $\mathbf{x}_i \in R^m$ for $1 \le i \le n$, $(n = l + u)$ are sampled from an unknown marginal distribution and the labels y_i for $1 \le i \le l$ take values from $\{-1, 1\}$, a semisupervised learning algorithm aims to propagate the label information from L to U.

It is thus natural to draw a parallel between this label propagation in the dataset $L \bigcup U$ and the molecule diffusion in the fluid. In particular, the label information propagating from L to U can be compared to the molecule diffusing from high concentration regions to low concentration regions; each labeled example can be compared to a high concentration region, and each unlabeled example can be compared to a low concentration region. By treating γ as the *propagation coefficient*, d as the *propagation distance*, and $\widetilde{f}$ as the *label information* (denoted as f to avoid notation confusion), the following explains the process of label propagation:

$$J = -\gamma \frac{f_j^{(t)} - f_i^{(t)}}{d_{ij}}. \tag{2}$$

Equation (2) informs us that $\mathbf{x}_i$ propagates its label information to $\mathbf{x}_j$ at the diffusion distance $d_{ij} = 1/\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/(2\sigma^2))$, and their soft labels at time t are $f_i^{(t)}$ and $f_j^{(t)}$, respectively. Here soft label means that f takes a value from a real range $[-1, 1]$.

Fig. 1 shows the propagation process from a labeled example $\mathbf{x}_i$ to an unlabeled example $\mathbf{x}_j$, where each example is modeled by a cube. The volume of both cubes is equal to V, and the area of their interface is A. Therefore, after a short time Δt from t, the amount of label information

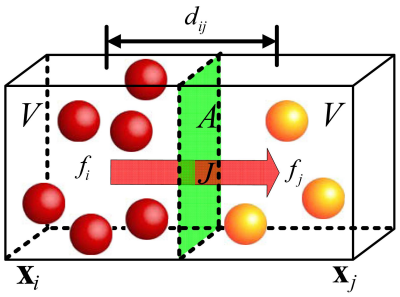


Fig. 1. Parallel between fluid diffusion and label propagation. The left cube with more balls is compared to the example with more label information. The right cube with fewer balls is compared to the example with less label information. The red arrow indicates the diffusion direction.

(i.e., the number of molecules) received by $\mathbf{x}_j$ satisfies the following:

$$\frac{f_j^{(t+\Delta t)} - f_j^{(t)}}{\Delta t} V = JA \tag{3}$$

where the variable J can be exactly expressed by Fick's First Law of Diffusion (2).

Because the label f_j varies in a discrete manner with respect to the iteration time t, Δt in (3) can be simply set to 1. Note that $V = d_{ij} A$ and substituting (2) into (3), we have

$$f_j^{(t+1)} = f_j^{(t)} - \gamma \frac{f_j^{(t)} - f_i^{(t)}}{d_{ij}^2}. \tag{4}$$

By taking the initial state of $\mathbf{x}_j$ into account, (4) is modified to

$$f_j^{(t+1)} = \alpha \left(f_j^{(t)} - \gamma \frac{f_j^{(t)} - f_i^{(t)}}{d_{ij}^2} \right) + (1-\alpha) y_j \tag{5}$$

where $y_j = f_j^{(0)}$ and it takes a value of 1, -1 or 0, if $\mathbf{x}_j$ is a positive, negative or unlabeled example, respectively. $\alpha \in (0, 1)$ is the trade off between the received information from $\mathbf{x}_i$ and the initial state of $\mathbf{x}_j$. Equation (5) models the label propagation process between two examples. However, in practice, one example receives the label information from all the other examples in the dataset. Therefore, if $J_{k \to j}$ is used to represent the *propagation flux* from $\mathbf{x}_k$ to $\mathbf{x}_j$, then the following equation holds:

$$\left(f_j^{(t+1)} - f_j^{(t)}\right) V = \sum_{k=1}^{n} J_{k \to j} A_k. \tag{6}$$

We assume that $\forall k \in \{1, 2, \ldots, n\}$, $J_{k \to j} = J$ and $A_k = A$. Then similar to the derivation of (5), after substituting Fick's First Law of Diffusion (2) into (6) and including y_j, the propagation process in the whole dataset is given by

$$f_j^{(t+1)} = \alpha \left(f_j^{(t)} - \sum_{k=1}^{n} \gamma \frac{f_j^{(t)} - f_k^{(t)}}{d_{kj}^2} \right) + (1-\alpha) y_j. \tag{7}$$

Equation (7) explains the propagation process between one example and the other examples in the dataset, and indicates that the received label information of an unlabeled example can be understood as an integration of the information emitted by the other examples.

In semisupervised learning, we apply Fick's First Law of Diffusion to both unlabeled and labeled examples. Slightly different from the original diffusion law that imposes the fluid transmitting from regions of higher concentration to those of lower concentration, FLAP simply allows the label information exchanging between arbitrary two examples regardless of their current label values. Thus, the labels of examples can be changed during the propagation process. Fortunately, Theorem 1 suggests that the labels of the labeled examples will remain virtually unchanged after the propagation process. Thus, the original labels of labeled examples, i.e., 1 or -1, will not convert from one to another.

Theorem 1: The labels of the labeled example will not change after the iteration process.

The proof of Theorem 1 is provided in Appendix A.

According to Fick's First Law of Diffusion, we update the labels of all examples simultaneously by denoting the label vector $\mathbf{f}^{(t)} = (f_1^{(t)} \; f_2^{(t)} \; \cdots \; f_n^{(t)})^T$, and the initial state vector $\mathbf{y} = (y_1 \; y_2 \; \cdots \; y_n)^T$, and then at time t, we have

$$\mathbf{f}^{(t+1)} = \alpha \mathbf{P} \mathbf{f}^{(t)} + (1-\alpha) \mathbf{y} \tag{8}$$

where the iteration matrix $\mathbf{P}$ is defined by

$$\mathbf{P} = \begin{pmatrix} 1-\gamma \sum\limits_{k=1,k\neq 1}^{n} d_{1k}^{-2} & \gamma d_{12}^{-2} & \cdots & \gamma d_{1n}^{-2} \\ \gamma d_{21}^{-2} & 1-\gamma \sum\limits_{k=1,k\neq 2}^{n} d_{2k}^{-2} \cdots & & \gamma d_{2n}^{-2} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma d_{n1}^{-2} & \gamma d_{n2}^{-2} & \cdots 1-\gamma \sum\limits_{k=1,k\neq n}^{n} & d_{nk}^{-2} \end{pmatrix}. \tag{9}$$

To guarantee that $\mathbf{P}$ is a nonnegative stochastic matrix and the summation of elements in every row is 1, we set $0 < \gamma < (\max_j \sum_{k=1,k\neq j}^{n} d_{jk}^{-2})^{-1}$. Equation (8) reveals that the soft label of an example is the linear combination of its initial state and the labels of all the examples including itself. We conduct (8) iteratively until convergence

$$\|\mathbf{f}^{(t+1)} - \mathbf{f}^{(t)}\|_2 < \varepsilon \tag{10}$$

where ε is a predefined small positive number. The converged vector $\mathbf{f}^*$ can then be obtained after the iteration process. Given a hard threshold 0, $\mathbf{x}_j$ is determined as positive if $f_j^* > 0$ (f_j^* is the j-th element in the vector $\mathbf{f}^*$) and negative otherwise.

Though (8) is derived for binary classification, it can be straightforwardly extended to multiclass classification by replacing the label vector $\mathbf{f}$ and the initial state vector $\mathbf{y}$ with the label matrix $\mathbf{F}$ and the initial state matrix $\mathbf{Y}$, respectively

$$\mathbf{F}^{(t+1)} = \alpha \mathbf{P} \mathbf{F}^{(t)} + (1-\alpha) \mathbf{Y} \tag{11}$$

where matrices $\mathbf{F}$ and $\mathbf{Y}$ are of size $n \times C$, and C denotes the total number of categories. We follow the conventional notation that $\mathbf{Y}_{jc'} = 1$ if $\mathbf{x}_j$ is labeled as $y_j = c'$, and $\mathbf{Y}_{jc'} = 0$

otherwise. If $\mathbf{x}_j$ is unlabeled, then $\mathbf{Y}_{jc'} = 0$ for $1 \le c' \le C$. Finally, $\mathbf{x}_j$ belongs to the class $c_j = \arg\max_{1\le c'\le C} \mathbf{F}_{jc'}$. The stopping criterion for propagation is modified accordingly

$$\|\mathbf{F}^{(t+1)} - \mathbf{F}^{(t)}\|_F < \varepsilon \tag{12}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Because (11) is a propagation model which simply transmits the label information according to the similarity between pairs of examples, so it can handle any kinds of data distributions. Moreover, FLAP is a graph-based algorithm, thus it can effectively exploit the manifold structure hidden in the dataset [1], [15], which explains why it always obtains encouraging classification results.

FLAP (11) converges at linear rate. This is shown in Theorem 2. The result of Theorem 2 is directly applicable to (8).

Theorem 2: The sequence $\{\mathbf{F}^{(t)}\}$, $t = 1, 2, \ldots$ generated by (11) eventually converges to

$$\mathbf{F}^* = \lim_{t\to\infty} \mathbf{F}^{(t)} = (1-\alpha)(\mathbf{I} - \alpha\mathbf{P})^{-1}\mathbf{Y} \tag{13}$$

linearly, i.e.

$$\lim_{t\to\infty} \frac{\|\mathbf{F}^{(t+1)} - \mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)} - \mathbf{F}^*\|_F} = \theta < 1 \tag{14}$$

where θ denotes the convergence rate.

Proof: Without loss of generality, the convergence of FLAP is studied for multiclass classification. By iteratively using (11), $\mathbf{F}^{(t)}$ is given by

$$\mathbf{F}^{(t)} = (\alpha\mathbf{P})^t\mathbf{Y} + (1-\alpha)\sum_{i=0}^{t-1}(\alpha\mathbf{P})^i\mathbf{Y}. \tag{15}$$

Note that $\mathbf{P}$ is a stochastic matrix ($\mathbf{P}_{ij} > 0$ and $\sum_j \mathbf{P}_{ij} = 1$), then according to the *Perron–Frobenius Theorem* [16], the spectral radius of $\mathbf{P}$ satisfies $\rho(\mathbf{P}) \le 1$, and thus we have

$$\lim_{t\to\infty}(\alpha\mathbf{P})^t = \mathbf{0}, \quad \lim_{t\to\infty}\sum_{i=0}^{t-1}(\alpha\mathbf{P})^i = (\mathbf{I} - \alpha\mathbf{P})^{-1}.$$

Therefore, the sequence $\{\mathbf{F}^{(t)}\}$ will finally converge to (13).

To prove that FLAP converges at linear rate, we need to demonstrate (14) holds. Considering that

$$\mathbf{F}^{(t+1)} = (\alpha\mathbf{P})^{t+1}\mathbf{Y} + (1-\alpha)\sum_{i=0}^{t}(\alpha\mathbf{P})^i\mathbf{Y} \tag{16}$$

we have (17) by plugging (13), (15), and (16) into (14)

$$\frac{\|\mathbf{F}^{(t+1)} - \mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)} - \mathbf{F}^*\|_F} = \frac{\left\|\left[(\alpha\mathbf{P})^{t+1} + (1-\alpha)\sum_{i=0}^{t}(\alpha\mathbf{P})^i - (1-\alpha)(\mathbf{I}-\alpha\mathbf{P})^{-1}\right]\mathbf{Y}\right\|_F}{\left\|\left[(\alpha\mathbf{P})^{t} + (1-\alpha)\sum_{i=0}^{t-1}(\alpha\mathbf{P})^i - (1-\alpha)(\mathbf{I}-\alpha\mathbf{P})^{-1}\right]\mathbf{Y}\right\|_F}. \tag{17}$$

Since $\mathbf{P}$ is symmetric, it can be decomposed into

$$\mathbf{P} = \mathbf{U}\Lambda\mathbf{U}^T \tag{18}$$

where $\mathbf{U}$ is an unitary matrix and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix containing eigenvalues of $\mathbf{P}$. According to (18) and the *Woodbury matrix identity* [17], we have

$$(\mathbf{I} - \alpha\mathbf{P})^{-1} = \mathbf{I} + \alpha\mathbf{U}(\Lambda^{-1} - \alpha\mathbf{I})^{-1}\mathbf{U}^T. \tag{19}$$

By substituting (18) and (19) into (17), we obtain

$$\begin{aligned}\frac{\|\mathbf{F}^{(t+1)} - \mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)} - \mathbf{F}^*\|_F} &= \frac{\left\|\left\{\mathbf{U}(\alpha\Lambda)^{t+1}\mathbf{U}^T + (1-\alpha)\sum_{i=0}^{t}\mathbf{U}(\alpha\Lambda)^i\mathbf{U}^T - \mathbf{F}^*\right\}\mathbf{Y}\right\|_F}{\left\|\left\{\mathbf{U}(\alpha\Lambda)^{t}\mathbf{U}^T + (1-\alpha)\sum_{i=0}^{t-1}\mathbf{U}(\alpha\Lambda)^i\mathbf{U}^T - \mathbf{F}^*\right\}\mathbf{Y}\right\|_F} \\ &= \frac{\|\{\mathbf{U}(\alpha^{t+1}\Lambda^{t+1} + \mathbf{M})\mathbf{U}^T\}\mathbf{Y}\|_F}{\|\{\mathbf{U}(\alpha^{t+1}\Lambda^{t} + \mathbf{M})\mathbf{U}^T\}\mathbf{Y}\|_F}\end{aligned} \tag{20}$$

where

$$\mathbf{M} = (1-\alpha)\left[\sum_{i=1}^{t}(\alpha\Lambda)^i - \alpha(\Lambda^{-1} - \alpha\mathbf{I})^{-1}\right].$$

From (20) we can easily observe that the only difference between the denominator and numerator is that one more diagonal matrix Λ is multiplied to the first term in the bracket of numerator. Thus (20) can be rewritten as

$$\frac{\|\mathbf{F}^{(t+1)} - \mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)} - \mathbf{F}^*\|_F} = \frac{\|\{\mathbf{U}\alpha^{t+2}\mathrm{diag}(\kappa_1\ \kappa_2\ \cdots\ \kappa_n)\mathbf{U}^T\}\mathbf{Y}\|_F}{\|\{\mathbf{U}\alpha^{t+1}\mathrm{diag}(\pi_1\ \pi_2\ \cdots\ \pi_n)\mathbf{U}^T\}\mathbf{Y}\|_F} \tag{21}$$

in which $\kappa_i = \lambda_i^{t+1}(1-\lambda_i)/1-\alpha\lambda_i$ and $\pi_i = \lambda_i^t(1-\lambda_i)/1-\alpha\lambda_i$. If we denote

$$\mathbf{U}\mathbf{J}_A\mathbf{U}^T \triangleq \mathbf{A}, \quad \mathbf{U}\mathbf{J}_B\mathbf{U}^T \triangleq \mathbf{B}$$

where $\mathbf{J}_A = \alpha^{t+2}\mathrm{diag}(\kappa_1\ \kappa_2\ \cdots\ \kappa_n)$ and $\mathbf{J}_B = \alpha^{t+1}\mathrm{diag}(\pi_1\ \pi_2\ \cdots\ \pi_n)$ that contain the eigenvalues of $\mathbf{A}$ and $\mathbf{B}$, respectively, then (21) is simplified as

$$\frac{\|\mathbf{F}^{(t+1)} - \mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)} - \mathbf{F}^*\|_F} = \frac{\|\mathbf{A}\mathbf{Y}\|_F}{\|\mathbf{B}\mathbf{Y}\|_F}. \tag{22}$$

Moreover, since $\mathbf{P}$ is a stochastic matrix with all its eigenvalues $\lambda_i \in [-1, 1]$ (in fact, 1 is the single eigenvalue of $\mathbf{P}$ according to the *Perron–Frobenius Theorem* [16]) and $\alpha \in (0, 1)$, so all the eigenvalues of $\mathbf{A}$ and $\mathbf{B}$ are nonnegative, and the eigenvalues of $\mathbf{A}$ are smaller than those of $\mathbf{B}$ in the corresponding position expressed in $\mathbf{J}_A$ and $\mathbf{J}_B$. Therefore

$$\frac{\|\mathbf{A}\mathbf{Y}\|_F}{\|\mathbf{B}\mathbf{Y}\|_F} = \frac{\left\|\mathbf{U}\mathbf{J}_A\mathbf{U}^T\mathbf{Y}\right\|_F}{\left\|\mathbf{U}\mathbf{J}_B\mathbf{U}^T\mathbf{Y}\right\|_F} = \sqrt{\frac{\mathrm{tr}\left(\mathbf{Y}^T\mathbf{U}\mathbf{J}_A{}^2\mathbf{U}^T\mathbf{Y}\right)}{\mathrm{tr}\left(\mathbf{Y}^T\mathbf{U}\mathbf{J}_B{}^2\mathbf{U}^T\mathbf{Y}\right)}} \tag{23}$$

where $\mathrm{tr}(\cdot)$ is the symbol of trace. Let $\mathbf{Q} = \mathbf{U}\mathbf{J}\mathbf{U}^T$ where $\mathbf{J} = \mathbf{J}_B{}^2 - \mathbf{J}_A{}^2 \ge 0$, then we obtain the margin between the denominator and numerator in (23)

$$\begin{aligned}&\mathrm{tr}\left(\mathbf{Y}^T\mathbf{U}\mathbf{J}_B{}^2\mathbf{U}^T\mathbf{Y}\right) - \mathrm{tr}\left(\mathbf{Y}^T\mathbf{U}\mathbf{J}_A{}^2\mathbf{U}^T\mathbf{Y}\right) \\ &\qquad = \mathrm{tr}\left[\mathbf{Y}^T\mathbf{U}\left(\mathbf{J}_B{}^2 - \mathbf{J}_A{}^2\right)\mathbf{U}^T\mathbf{Y}\right] = \mathrm{tr}\left(\mathbf{Y}^T\mathbf{Q}\mathbf{Y}\right).\end{aligned} \tag{24}$$

Note that $\mathbf{Q} = \mathbf{U}\sqrt{\mathbf{J}}(\mathbf{U}\sqrt{\mathbf{J}})^T$ and $\mathbf{U}\sqrt{\mathbf{J}}$ are invertible ($\det(\mathbf{U}\sqrt{\mathbf{J}}) = \det(\mathbf{U})\cdot\det(\sqrt{\mathbf{J}}) \ne 0$), so $\mathbf{Q}$ is a positive

TABLE I
FLAP VERSUS POPULAR SSL ALGORITHMS

Algorithm	Regularization Framework
Mincut [5]	$\min_{\mathbf{f}:f_{\{j,k\}}\in\{-1,1\}} \quad Q(\mathbf{f})=\sum_{k=1}^{n}\sum_{j=1}^{n}\omega_{kj}(f_k-f_j)^2+\theta\sum_{k=1}^{n}(f_k-y_k)^2$
HF [6]	$\min_{\mathbf{f}:f_{\{j,k\}}\in\mathbb{R}} \quad Q(\mathbf{f})=\sum_{k=1}^{n}\sum_{j=k}^{n}\omega_{kj}(f_k-f_j)^2+\theta\sum_{k=1}^{n}(f_k-y_k)^2$
LNP [13]	$\min_{\mathbf{f}:f_{\{j,k\}}\in\mathbb{R}} \quad Q(\mathbf{f})=\sum_{k=1}^{n}\sum_{j:\mathbf{x}_j\in N(\mathbf{x}_k)}\omega_{kj}(f_k-f_j)^2+\theta\sum_{k=1}^{n}(f_k-y_k)^2$
LGC [7]	$\min_{\mathbf{f}:f_{\{j,k\}}\in\mathbb{R}} \quad Q(\mathbf{f})=\frac{1}{2}\left[\sum_{k=1}^{n}\sum_{j=1}^{n}\omega_{kj}\left(\frac{1}{\sqrt{D_{kk}}}f_k-\frac{1}{\sqrt{D_{jj}}}f_j\right)^2+\theta\sum_{k=1}^{n}(f_k-y_k)^2\right]$
FLAP	$\min_{\mathbf{f}:f_{\{j,k\}}\in\mathbb{R}} \quad Q(\mathbf{f})=\frac{1}{2}\left[\sum_{k=1}^{n}\sum_{j=1}^{n}p_{kj}(f_k-f_j)^2+\theta\sum_{k=1}^{n}(f_k-y_k)^2\right]$

definite matrix. Furthermore, if $\mathbf{Y}$ is partitioned by columns as $\mathbf{Y}=(\mathbf{Y}_1\ \mathbf{Y}_2\ \ldots\ \mathbf{Y}_n)$, then

$$\mathrm{tr}(\mathbf{Y}^T\mathbf{Q}\mathbf{Y})=\mathrm{tr}\left(\sum_{i=1}^{n}Y_i^T\mathbf{Q}Y_i\right). \tag{25}$$

Because $\mathbf{Q}$ is positive definite and $Y_1, Y_2, \ldots, Y_n$ are nonzero vectors (this is guaranteed, because each class should has at least one labeled example), the result of (25) is definitely above zero. Therefore, the denominator of (23) is larger than the numerator, which indicates that

$$0<\frac{\|\mathbf{F}^{(t+1)}-\mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)}-\mathbf{F}^*\|_F}<1. \tag{26}$$

According to (26), we can conclude

$$\lim_{t\to\infty}\frac{\|\mathbf{F}^{(t+1)}-\mathbf{F}^*\|_F}{\|\mathbf{F}^{(t)}-\mathbf{F}^*\|_F}=\theta<1. \tag{27}$$

Thus, the linear convergence rate of FLAP for multiclass classification is proved.

III. INTERPRETATION AND CONNECTIONS

A. Regularization Networks

To straightforwardly compare FLAP with other SSL algorithms and show its superiority, we reformulate FLAP in the context of the classical regularization theory

$$\min_{\mathbf{f}\in\mathbb{R}^n}Q(\mathbf{f})=\frac{1}{2}\left[\sum_{k=1}^{n}\sum_{j=1}^{n}p_{kj}\left(f_k-f_j\right)^2+\tau\sum_{k=1}^{n}(f_k-y_k)^2\right] \tag{28}$$

where p_{kj} is the (k, j)th element in the matrix $\mathbf{P}$. The first term in the right-hand side of (28) forms a specific prior knowledge and enforces the labels in $\mathbf{f}$ varying smoothly. It indicates that the soft labels of similar examples should not differ too much from one another. The second fitting term means that the decided labels should be consistent with the examples' original states well.[1] The regularization parameter $\tau>0$ controls the trade off between smoothness term and fitting term.

[1]Similar to LGC [7], LNP [13] and GTAM [18], we penalize the deviations of the labels of all the examples from their initial states. This formulation prefers the consistency for labeled data, and the compromise in assigning the labels to the unlabeled data [13].

It is straightforward to show that the optimal solution of (28) equals the iterative result of (8). The derivative of $Q(\mathbf{f})$ with respect to $\mathbf{f}$ is

$$\partial Q/\partial\mathbf{f}=2(\mathbf{I}-\mathbf{P})\mathbf{f}+\tau(\mathbf{f}-\mathbf{y}). \tag{29}$$

Set the right-hand side of (29) to 0 and let $\alpha=2/2+\tau$, $\beta=\tau/2+\tau$ so that $\alpha+\beta=1$, then (29) is reformulated as

$$\mathbf{f}-\alpha\mathbf{P}\mathbf{f}-\beta\mathbf{y}=\mathbf{0} \tag{30}$$

which leads to the same closed-form solution as (13)

$$\mathbf{f}=\beta(\mathbf{I}-\alpha\mathbf{P})^{-1}\mathbf{y}. \tag{31}$$

Theorem 3: The regularization form (28) is a convex optimization problem.

Proof: By denoting

$$d_k=\sum_{j=1,j\neq k}^{n}p_{kj}$$

and calculating the Hessian matrix $\mathbf{H}_0$ of (28), we have

$$\mathbf{H}_0=\begin{pmatrix}2d_1+\tau & -2p_{12} & \cdots & -2p_{1n}\\ -2p_{21} & 2d_2+\tau & \cdots & -2p_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ -2p_{n1} & -2p_{n2} & \cdots & 2d_n+\tau\end{pmatrix}. \tag{32}$$

From *Gerschgorin circle theorem* [16], we know that all the eigenvalues of $\mathbf{H}_0$ belong to $[\tau, 4\max_{1\le k\le n}(d_k)+\tau]$, which reveals that $\mathbf{H}_0$ is positive definite. Therefore, the optimization problem (28) is convex and the convergent result is globally optimal.

Theorem 3 also implies that the convergent point of FLAP (31) corresponds to the global optimal solution. Given (28), Table I compares FLAP with representative SSL algorithms in a straightforward way, in which $N(\mathbf{x}_k)$ denotes the neighbors of the example $\mathbf{x}_k$ and $\omega_{kj}=\exp(-\|\mathbf{x}_k-\mathbf{x}_j\|^2/(2\sigma^2))$. It can be observed that the main difference between these methods is the definition of smoothness. FLAP uses the matrix $\mathbf{P}$ to describe the smoothness, while Mincut, HF, LNP, and LGC adopt the (normalized) graph Laplacian to define the smoothness. This difference leads to the faster convergence rate achieved by FLAP.

The regularization framework of FLAP is also much related to the semisupervised formulation of binary kernel spectral clustering (BKSC) [19]. Suppose the adjacency matrix of a graph is $\mathbf{W}$, and the diagonal matrix $\mathbf{D}$ is defined by $\mathbf{D}_{kk} = \sum_j \mathbf{W}_{kj}$, $\forall k = 1, 2, \ldots, n$, then BKSC is

$$\min_{\omega, \mathbf{b}} \ \frac{1}{2}\left[\omega^T \omega - \gamma_1 \mathbf{f}^T \mathbf{D}^{-1} \mathbf{f} + \gamma_2 \sum_{i=1}^{l} (f_i - y_i)^2\right]. \quad (33)$$

In (33), $\mathbf{f} = \Theta\omega + \mathbf{b}\mathbf{e}_l$, where $\mathbf{e}_l$ is an all-one l-dimensional column vector, and Θ is the kernelized data matrix with each row representing an example. ω and $\mathbf{b}$ are unknown vectors to be optimized, and γ_1, γ_2 are nonnegative parameters balancing the three terms. BKSC (33) differs from FLAP (28) regarding the prior knowledge. FLAP prefers the solution to be smooth, while BKSC emphasizes the examples corresponding to large $\mathbf{D}_{kk}$.

B. Markov Random Fields

Markov random fields (MRFs) are a set of random variables having the Markov property described by an undirected graph. The first-order intrinsic Gaussian MRFs (FO-IGMRF) is an MRF in which the joint distribution is Gaussian and the precision matrix $\mathbf{Q}$ is rank reduced and meanwhile satisfies $\mathbf{Qe} = \mathbf{0}$ ($\mathbf{e}$ is a vector of all ones) [20]. Similar to other propagation algorithms, e.g., [13], FLAP can also be cast into the framework of FO-IGMRF. Defining the increment along the edge in a graph as

$$\Delta d_{kj} = f_k - f_j. \quad (34)$$

Suppose Δd_{kj} fits the Gaussian distribution with $\Delta d_{kj} \sim N(0, d_{kj}^2/\gamma)$, then we have

$$p(\Delta d_{kj}) \propto \exp\left[-\frac{\gamma}{2d_{kj}^2}(f_k - f_j)^2\right]. \quad (35)$$

Assuming that all the increments along the edges are conditionally independent, then the joint probability over $\mathbf{f}$ is calculated as the product of the probabilities over all the increments

$$p(\mathbf{f}) \propto \prod p(\Delta d_{kj}) = \exp\left[-\frac{1}{2}\gamma \sum \frac{1}{d_{kj}^2}(f_k - f_j)^2\right]. \quad (36)$$

Suppose the adjacency matrix $\mathbf{W}$ of a graph is

$$\mathbf{W}_{kj} = \begin{cases} \gamma\, d_{kj}^{-2}, & k \neq j \\ 0, & k = j \end{cases} \quad (37)$$

then (36) is reformulated as

$$p(\mathbf{f}) \propto \exp\left[-\frac{1}{2}\gamma\, \mathbf{f}^T(\mathbf{D} - \mathbf{W})\mathbf{f}\right] = \exp\left[-\frac{1}{2}\gamma\, \mathbf{f}^T \mathbf{L}\mathbf{f}\right] \quad (38)$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian. Since $\mathbf{Le} = \mathbf{0}$, $\mathbf{L}$ is exactly the precision matrix $\mathbf{Q}$ in FO-IGMRF.

C. Graph Kernels

Because of the popularity of graph-based methods for data analyses, many diffusion kernels on graph nodes have been developed, such as exponential diffusion kernel [21] and Von Neumann diffusion kernel [22]. This section examines the relationship between the proposed FLAP and these popular diffusion kernels.

If we ignore the manually incorporated initial state term y_i in (7), and only focus on the propagation process itself, (7) reduces to

$$f_j^{(t+1)} = f_j^{(t)} + \gamma \sum_{k=1}^{n} d_{kj}^{-2} f_k^{(t)} - \gamma \sum_{k=1}^{n} d_{kj}^{-2} f_j^{(t)}. \quad (39)$$

By regarding f_j as a variable w.r.t. the time t, we have

$$\begin{aligned} \frac{df_j}{dt} &= \gamma \sum_{k=1}^{n}\left[(1-\delta_{kj}) d_{kj}^{-2} f_k - \delta_{kj} \sum_{i=1}^{n} d_{ij}^{-2} f_j\right] \\ &= \gamma \sum_{k=1}^{n}\left[(1-\delta_{kj}) d_{kj}^{-2} - \delta_{kj} \sum_{i=1}^{n} d_{ij}^{-2}\right] f_k \end{aligned} \quad (40)$$

where δ_{kj} is the Kronecker delta with $\delta_{kj} = 1$ if $k = j$, and 0 otherwise. Therefore, we can write the variables f_j $(j = 1, \ldots, n)$ as (40) into a compact matrix form

$$\frac{d\mathbf{f}}{dt} = (\mathbf{P} - \mathbf{I})\mathbf{f} \quad (41)$$

where $\mathbf{P}$ is the iteration matrix defined by (9), and $\mathbf{f} = (f_1, \ldots, f_n)^T$ is an n-dimensional variable vector of time t. By denoting $\mathbf{H} = \mathbf{P} - \mathbf{I}$, the above differential equation leads to the solution $\mathbf{f} = \exp(\mathbf{H}t)\mathbf{f}^{(0)}$, which results in the graph kernel related to FLAP

$$\mathbf{K}_{\text{FLAP}} = \exp(\tilde{\sigma}\mathbf{H}) = \sum_{i=0}^{\infty} \frac{\tilde{\sigma}^i \mathbf{H}^i}{i!} \quad (42)$$

where $\tilde{\sigma}$ is a real parameter. Since $\mathbf{H}$ is a positive semidefinite matrix, $\mathbf{K}_{\text{FLAP}}$ can be regarded as an exponential diffusion kernel according to [21].

FLAP is also related to the von Neumann diffusion kernel, which has the formation of

$$\mathbf{K}_{\text{VND}} = (\mathbf{I} - \tilde{\alpha}\mathbf{A})^{-1} = \sum_{i=0}^{\infty} \tilde{\alpha}^i \mathbf{A}^i$$

with $0 < \tilde{\alpha} < \rho(\mathbf{A})^{-1}$ [22]. For FLAP, since $\mathbf{P}$'s spectral radius $\rho(\mathbf{P}) = 1$, and $\alpha \in (0, 1)$ as explained in Section II, the term $(\mathbf{I} - \alpha\mathbf{P})^{-1}$ in (13) is actually a Von Neumann diffusion kernel and encodes the similarities of pairs of examples.

IV. Experimental Results

In this section, FLAP will be evaluated on typical synthetic and vision datasets. Several popular graph-based SSL algorithms serve as baselines for comparison, including HF [6], LGC [7], LNP [13], NTK [14], and MinCut [5]. We focus on two issues of these algorithms: one is the classification accuracy on unlabeled examples given very few labeled examples for every dataset, and the other is the efficiency such as CPU seconds and iteration times. In all the experiments

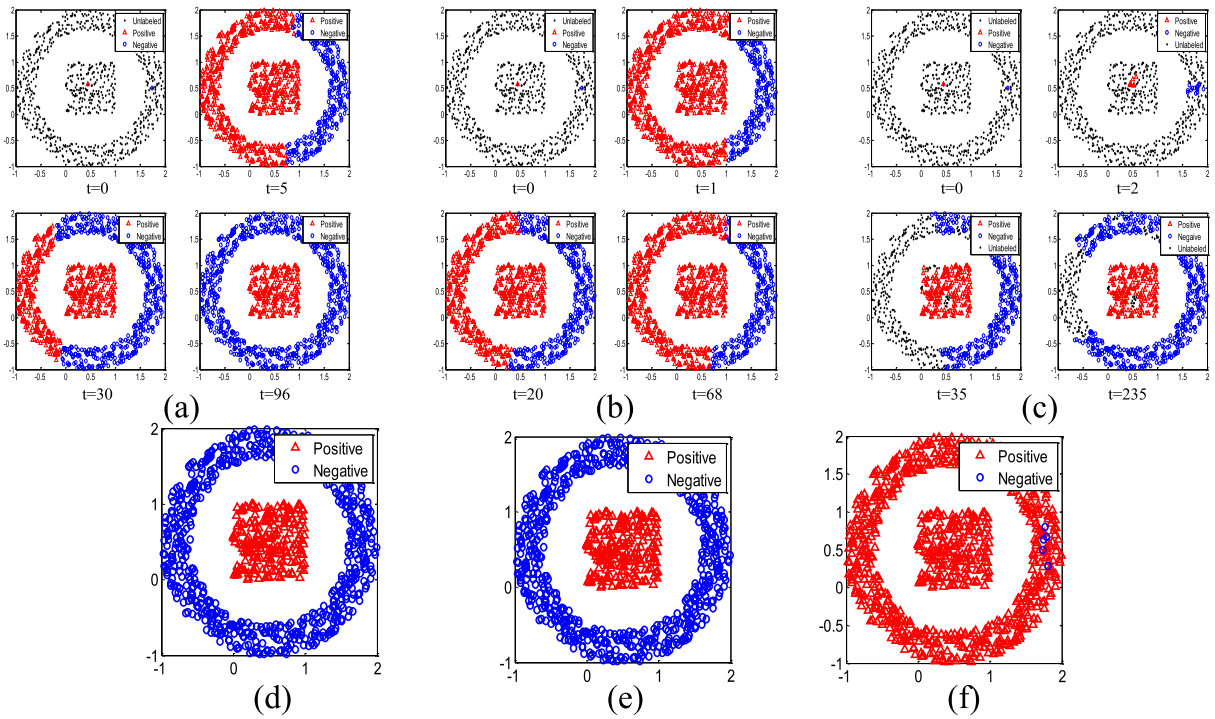


Fig. 2. Propagation results on *Square&Ring* dataset. (a)–(c) Propagation processes of FLAP, LGC, and LNP. (d)–(f) Classification results brought by MinCut, HF, and NTK.

below, the parameters of FLAP are set to $\alpha = 0.99$, and $\gamma = \eta(\max_j \sum_{k=1,k\neq j}^{n} d_{jk}^{-2})^{-1}$ where η is chosen within $[0, 1]$. In HF, LGC and LNP, the parameter α is also set to 0.99. For fair comparison, we construct the identical K Nearest Neighborhood (K-NN) graph for all the algorithms, and the σ in the expression of the diffusion distance has been also respectively adjusted to achieve the best performance for different datasets.

A. Synthetic Data

Square&Ring and *DoubleMoon* datasets are used to visualize the propagation processes of three iterative methods, including LGC, LNP and the proposed FLAP. The results of the three noniterative methods, i.e., MinCut, HF, and NTK, are also reported.

The *Square&Ring* dataset contains a square and a ring, both of which are centered at (0.5, 0.5). The radius of the outer ring is 1.3, and the length of each side of the inner square is 1 (Fig. 2). In the *DoubleMoon* dataset, 500 examples are equally divided into two moons that are centered at $(0, 0)$ and $(6, 0)$, respectively. The width of each moon is set to 6 to make the moons fatter. Compared with [7] and [23] that use the *DoubleMoon* dataset for illustration, we reduce the interclass distance to increase the classification difficulty (Fig. 3). Note that there is only one labeled example in each class for both datasets (see $t = 0$ in every subplot).

For implementing FLAP on the *Square&Ring* dataset, we set $K = 5$, $\sigma = 0.2$, and $\eta = 0.95$. Fig. 2 reveals that LGC wrongly propagates positive labels to the outer ring. Part of the unlabeled examples cannot receive label information through LNP until convergence. Most negative examples are classified as positive by NTK. By contrast, FLAP, MinCut, and HF are able to correctly classify all the examples. The parameter settings of FLAP on *DoubleMoon* are $K = 5$, $\sigma = 0.2$, and $\eta = 0.6$. The results are displayed in Fig. 3. It can be observed that LNP propagates the positive label to the upper moon by mistake in about $t = 27$. Mincut and NTK also fail to obtain the perfect results. Comparatively, LGC, HF, and FLAP perform better than the above methods. For quantitative comparison, we also report the classification accuracies of these methods on *Square&Ring* and *DoubleMoon* in Table II.

Moreover, to validate the efficiency of FLAP, we record both iteration times and CPU seconds of the three iterative methods. In this paper, all the algorithms are conducted on a work station with 2.40GHz Intel Xeon CPU and 24G memory. Table II suggests FLAP achieves comparable time costs with LGC on the *Square&Ring* dataset. However, the convergent result of LGC is incorrect shown by Fig. 2, and this can be the reason why it converges quickly in this dataset. The convergence curves are also shown in Fig. 4 and demonstrate that FLAP converges very quickly on both datasets.

We further use the *DoubleMoon* dataset to test the ability of FLAP for handling the problem of imbalanced data. The number of negative examples is fixed to 250, while we randomly sample 125, 50, 25, and 10 positive examples from the bottom moon, so the relative ratio of positive examples to negative examples are 1:2, 1:5, 1:10, and 1:25, respectively. The initially labeled examples for this experiment are identical to Fig. 3, and the propagation results are presented in Fig. 5. It can be observed that FLAP can successfully classify all

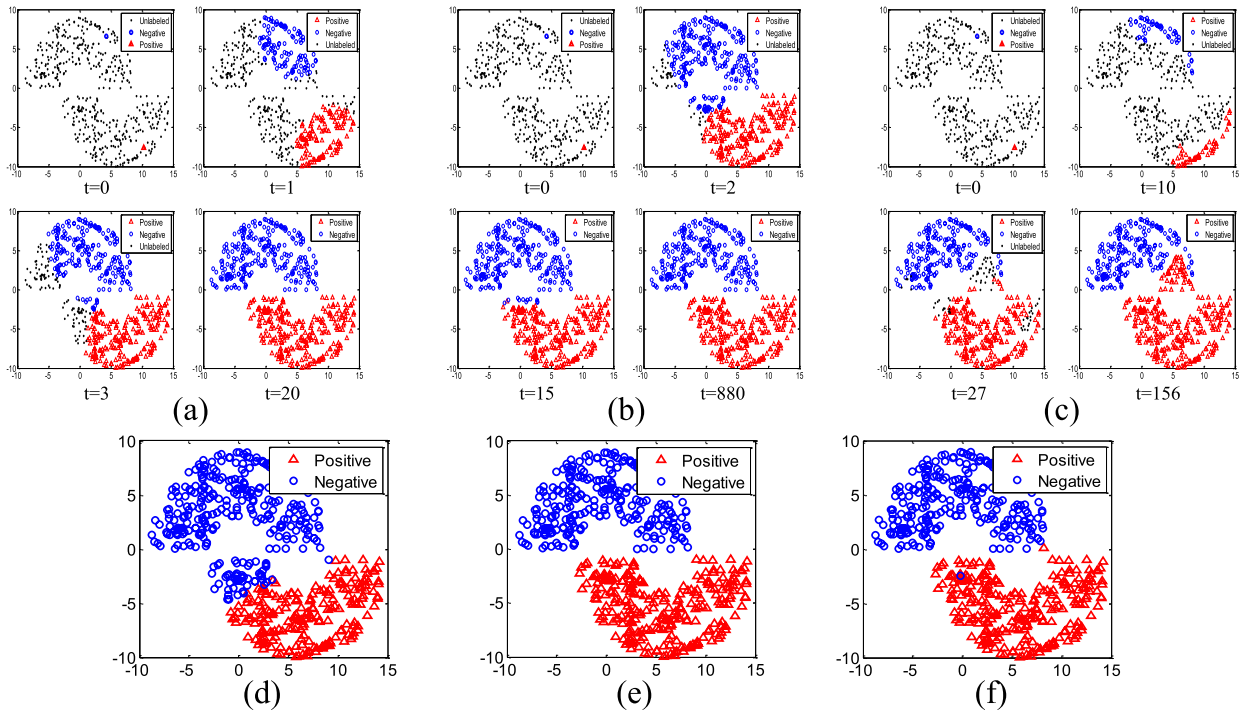


Fig. 3. Propagation results on *DoubleMoon* dataset. (a)–(c) Propagation processes of FLAP, LGC, and LNP. (d)–(f) Classification results brought by MinCut, HF, and NTK.

TABLE II
PERFORMANCES OF ALL THE METHODS ON TWO SYNTHETIC DATASETS. EACH RECORD FOLLOWS THE FORMAT ITERATION TIME/CPU SECONDS/ACCURACY

Type	Algorithms	*Square&Ring*	*DoubleMoon*
Iterative	LGC [7]	68/0.436/54.2%	880/4.488/100.0%
	LNP [13]	235/1.685/79.3%	156/1.044/91.8%
	FLAP	114/0.658/100.0%	20/0.338/100.0%
Non-iterative	MinCut [5]	-/-/100.0%	-/-/89.6%
	HF [6]	-/-/100.0%	-/-/100.0%
	NTK [14]	-/-/24.7%	-/-/99.6%

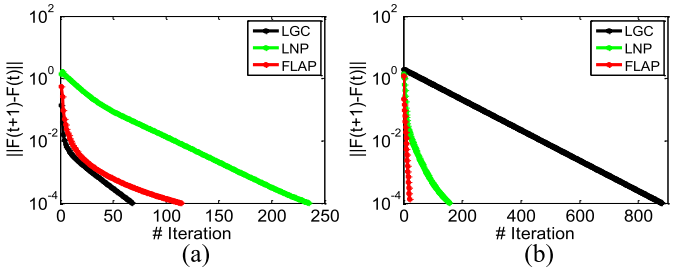


Fig. 4. Comparison of convergence curves. (a) Result on *Square&Ring*. (b) Result on *DoubleMoon*.

the examples except 1:25 situation. In Fig. 5(d), two positive examples are mistakenly classified into the negative class. This is mainly because of the following reasons:

1) The positive examples are so scarce that they fail to represent the underlying manifold of the real distribution of the positive examples (i.e., the moon shaped by positive points).

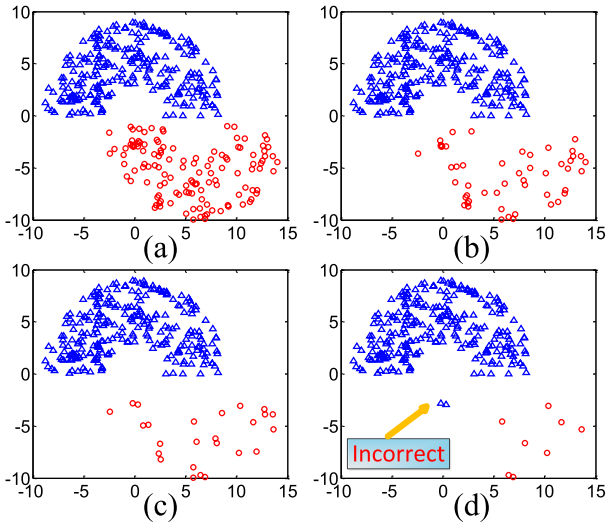


Fig. 5. Classification outputs on the imbalanced *DoubleMoon*. (a)–(d) Results of 1:2, 1:5, 1:10, and 1:25 situations.

2) The two mistakenly classified examples are closer to the upper moon, so the negative labels can be more conveniently propagated to them than the positive labels.

This experiment well demonstrates that FLAP is insensitive to the problem of imbalanced data.

B. Real Benchmarks Data

In this section, the performances of FLAP, LGC, LNP, HF, NTK, and MinCut are evaluated by testing on the two public datasets *United States Postal Service (USPS)* and *Brain Computer Interface (BCI)* in [15].

TABLE III
EXPERIMENTAL RESULTS ON THE BENCHMARK DATASETS FOR THE VARIETY OF SSL ALGORITHMS. [THE VALUES IN THE TABLE REPRESENT ACCURACY (%)]

Datasets	*USPS*[15]		*BCI*[15]	
$l(\#Labeled\ Examples)$	10	100	10	100
LGC [7]	85.90	94.96	51.40	59.85
LNP [13]	77.75	79.98	50.85	56.73
HF [6]	86.39	93.64	49.64	53.78
NTK [24]	79.57	87.65	49.29	48.97
MinCut [5]	80.21	94.21	51.88	66.35
FLAP	88.31	94.24	53.40	66.90

The *USPS* dataset contains ten digits, each with 150 images. We combine digits two and five as the positive class, and form all the others as the negative class. Therefore, the two classes are imbalanced and the relative size of the two classes is 1:4. The *BCI* dataset is used to study the brain-computer interface. It contains 400 imagined movements (examples) with 200 using the left hand (negative class) and 200 using the right hand (positive class).

In each dataset, we implement all the algorithms under $l = 10$ and $l = 100$, and the final results are averaged over 12 independent runs with different partitions of labeled and unlabeled subsets. We built 9-NN and 8-NN graphs with $\sigma = 0.2$ for *USPS* and *BCI*, respectively, and η in FLAP is set to 0.1 and 0.01 for these two datasets. The number of principal components in NTK is set to $m = 100$ to achieve the optimal performance. Table III shows the accuracies of different algorithms, in which the highest and the second highest records are marked with red and blue color, respectively. We observe that compared with all the baselines, the proposed FLAP obtains encouraging performance generally. An exceptional case is that FLAP performs slightly worse than LGC on *USPS* when $l = 100$. Specifically, the experimental results on *USPS* also demonstrate that FLAP is not sensitive to the problem of the imbalanced data.

C. UCI Data

We adopt four UCI Machine Learning Repository datasets,[2] *Iris*, *Wine*, *BreastCancer*, and *National Classification of Economic Activities-9 (CNAE-9)* to compare FLAP with the other graph-based algorithms. The classification accuracy and iteration times under different sizes of the labeled set l, in particular, are evaluated. Algorithms are implemented 30 times independently under each l with randomly selected examples, and the reported accuracy and iteration times are calculated as the mean value of the outputs of these runs. Note that at least one labeled example is guaranteed in each class when the labeled sets are generated. Five state-of-the-art graph-based SSL algorithms are adopted for baselines, including LGC, LNP, MinCut, HF, and NTK.

We built 10-NN, 6-NN, 7-NN, and 10-NN graphs for *Iris*, *Wine*, *BreastCancer*, and *CNAE-9*, respectively.

[2]http://archive.ics.uci.edu/ml/.

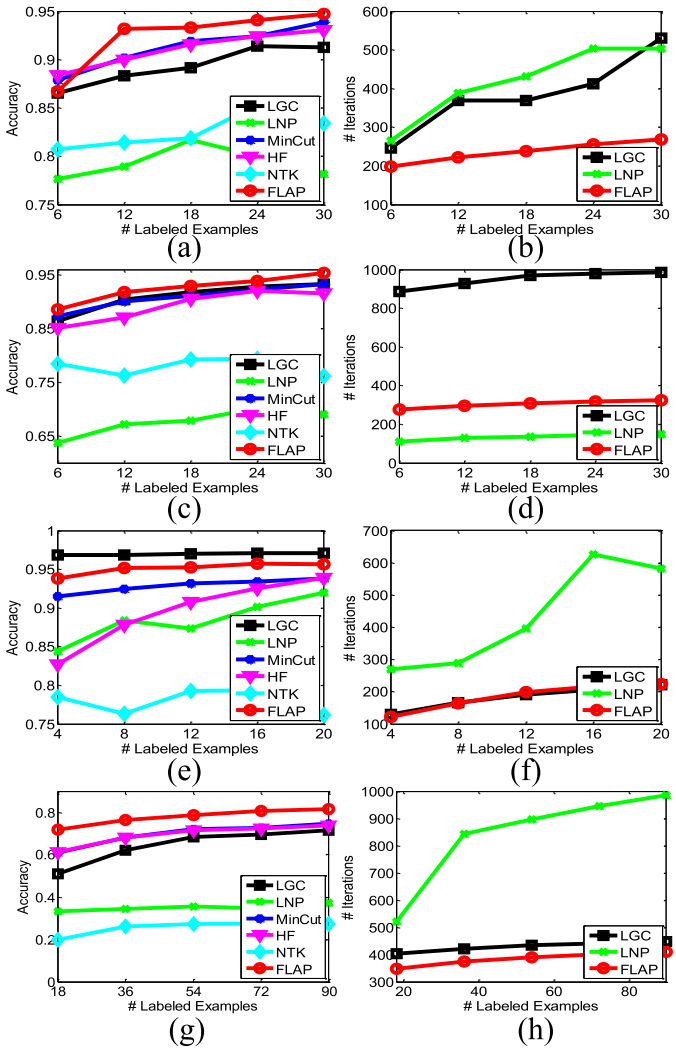


Fig. 6. Comparison of accuracy and iteration times. (a) and (b) *Iris*. (c) and (d) *Wine*. (e) and (f) *BreastCancer*. (g) and (h) *CNAE-9*.

Other parameters of FLAP are $\sigma = 0.2$, $\eta = 0.1$ for *Iris*, $\sigma = 0.5$, $\eta = 0.01$ for *Wine*, $\sigma = 0.5$, $\eta = 0.001$ for *BreastCancer*, and $\sigma = 1$, $\eta = 0.1$ for *CNAE-9*. The m in NTK is set to 50 for all the four UCI datasets. Fig. 6(a), (c), (e), and (g) demonstrates that FLAP is generally able to reach the highest accuracy among the comparators when l changes from small to large.

Moreover, Fig. 6(b), (d), (f), and (h) presents the iteration times of all the iterative methods (LGC, LNP and FLAP) on four UCI datasets, respectively. Other baselines including HF, MinCut and NTK are not compared here because they are not iteration-based. It can be observed that FLAP requires the least iteration times in most cases. An exceptional case is that LNP is more efficient than FLAP on the *Wine* dataset. However, the accuracy of LNP is not as high as that of FLAP shown by Fig. 6(c), so the results obtained by FLAP are encouraging.

D. Handwritten Digit Recognition

Handwritten digit recognition is a branch of optical character recognition (OCR). We compare FLAP with baselines

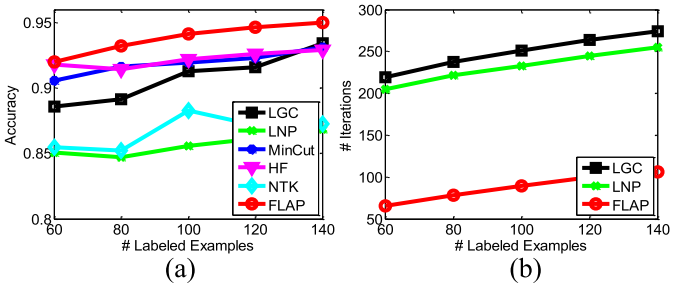


Fig. 7. Comparison of accuracy and iteration times on digit recognition dataset. (a) and (b) Curves of accuracy and iteration times with the growing of the labeled examples.

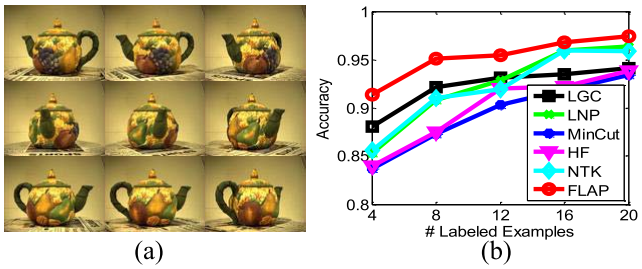


Fig. 8. Experiment on *Teapot* dataset. (a) Some typical images. (b) Accuracy curve for comparison.

on the *Optical Recognition of Handwritten Digits* Dataset,[3] (abbreviated as *Digits*) in which numbers $0 \sim 9$ are considered as different classes. This dataset contains 5620 digital images and the resolution of each image is 8×8. The pixel-wise feature is described by a 64-D vector with elements representing the gray values. Labeled examples are randomly selected from the whole dataset. We built a 10-NN graph for all the methods, and choose $\sigma = 7$ and $\eta = 0.01$ for FLAP.

When l varies from small to large, the accuracy and iteration times of the methods are plotted in Fig. 7(a) and (b), respectively. This figure indicates that FLAP achieves the highest accuracy with the least iteration times.

E. Teapot Image Classification

The *Teapot* dataset [24] contains 365 images of a teapot, and the angle of the spout in every image is different [Fig. 8(a)]. The goal is to determine whether the orientation of the spout is right or left. The resolution of each image is 12×16, and hence the pixel-wise feature adopted is a 192-dimensional vector.

A 10-NN graph with $\sigma = 30$ was established for every algorithm. η for FLAP was set to 0.01 to get the optimal performance. Fig. 8(b) shows the accuracy curve of several algorithms when the size of the labeled data varies from 4 to 20, which reveals that FLAP performs better than other algorithms.

[3]http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits.

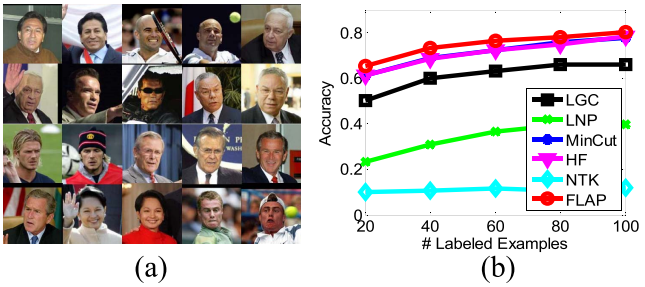


Fig. 9. Experimental results on *LFW* face dataset. (a) Some representative images. (b) Comparison of the recognition accuracy.

F. Face Recognition

We tested the ability of FLAP on face recognition by using the challenging dataset Labeled Face in the Wild (*LFW*).[4] The face images in this dataset are directly collected under natural scenes, so the facial expressions, observation angle, illumination conditions and background setting are not intentionally controlled for recognition. The faces in all the images are detected by the well-known Viola–Jones detector [25].

In this experiment, we used a subset of *LFW* by choosing the persons who have more than 30 face images. We chose the images of Toledo, Sharon, Schwarzenegger, Powell, Rumsfeld, Bush, Arroyo, Agassi, Beckham, and Hewitt for recognition, which leads to totally 392 examples belonging to 10 people in the subset [Fig. 9(a) for some examples]. We adopted the 73-D feature developed in [26], which describes some biometrics traits such as gender, race, age, and hair color. A 10-NN graph with $\sigma = 1$ was built on the entire dataset, and η was set to 0.5. The recognition rates of algorithms are compared in Fig. 9(b). We observe that HF and MinCut perform comparably to FLAP, while NTK and LNP are inferior to FLAP notably. In general, FLAP achieves very satisfactory results and outperforms other methods with l changing from 20 to 100.

G. Statistical Significance

Above experiments have empirically shown the superiority of FLAP to other baselines in terms of classification accuracy. In this section we use the 5×2 cross-validation F-test (5×2 cv F-test) [27] to show the statistical significance of FLAP over the other methods. The F-statistics value produced by the 5×2 cv F-test is to identify whether two algorithms achieve the same performance on a certain dataset. The null hypothesis is that they do obtain the same accuracy, and we reject this hypothesis with 95% confidence if the F-statistics value is greater than 4.74. For conducting the 5×2 cv F-test, five replications of twofold cross-validation are performed, and every datasets are equally and randomly split into two folds in each replication; however, the splits in the five replications were identical for all the algorithms. To compare the transductive accuracy of all the methods, in each replication we randomly sample l labeled examples in one fold, and evaluate the accuracy of classifiers on the examples of the other fold. Given $m_i^{(j)}$ as the difference of error rates generated by

[4]http://vis-www.cs.umass.edu/lfw/.

TABLE IV
F-STATISTICS VALUES OF BASELINES VERSUS FLAP ON THREE UCI DATASETS. (THE RECORDS SMALLER THAN 4.74 ARE MARKED IN RED, WHICH MEAN THAT 1) THE NULL HYPOTHESIS IS ACCEPTED AND 2) THE CORRESPONDING BASELINE ALGORITHM PERFORMS COMPARABLY TO FLAP)

	l	LGC	LNP	MinCut	HF	NTK
Iris	6	1.28	3.84	1.25	1.84	8.23
	12	20.71	12.93	5.69	6.76	12.46
	18	18.05	5.02	5.06	4.80	9.76
	24	13.77	7.17	2.14	4.60	13.08
	30	14.42	11.49	2.57	2.92	10.91
Wine	6	4.99	41.65	3.56	9.21	39.99
	12	3.29	16.81	3.13	5.58	208.63
	18	1.52	55.17	2.93	2.61	192.36
	24	1.51	24.65	2.74	2.77	222.16
	30	5.55	26.33	8.50	7.23	864.96
BreastCancer	4	2.64	14.39	2.08	8.13	37.77
	8	4.21	13.88	9.58	22.92	701.85
	12	6.86	27.37	6.80	18.29	939.93
	16	7.89	8.22	78.31	27.53	8132.42
	20	5.25	7.00	3.78	6.86	2747.30
CNAE-9	18	44.70	98.95	100.32	13.59	1434.33
	36	8.27	102.22	42.59	61.96	10596.37
	54	20.88	54.82	21.22	24.10	2107.28
	72	14.29	94.07	58.05	28.98	7535.61
	90	10.14	103.75	68.64	32.27	3852.48
Digits	60	101.51	14.53	7.14	6.12	153.55
	80	78.99	41.57	13.05	4.06	2211.52
	100	63.08	21.67	8.27	7.34	503.88
	120	73.60	118.29	7.13	7.68	438.75
	140	24.31	19.33	8.36	8.52	672.71
Teapot	4	2.94	7.39	16.51	8.41	9.54
	8	3.89	4.49	9.07	10.76	10.37
	12	4.23	3.81	18.65	31.21	12.98
	16	10.89	1.61	7.82	7.05	2.21
	20	8.40	2.42	7.66	5.91	3.48
LFW	20	17.41	267.00	3.20	7.07	1252.94
	40	33.76	410.25	8.99	12.16	5420.41
	60	28.69	129.32	6.08	11.84	3327.88
	80	18.23	106.75	19.82	7.34	4424.79
	100	24.00	113.43	4.02	12.09	2139.00

two algorithms on fold j ($j = 1, 2$) of replication i ($i = 1, 2, \ldots, 5$), then the mean error rate and the variance of replication i are $\bar{m}_i = (m_i^{(1)} + m_i^{(2)})/2$ and $s_i^2 = (m_i^{(1)} - \bar{m}_i)^2 + (m_i^{(2)} - \bar{m}_i)^2$, respectively. Therefore, according to [27], the F-statistics value $F = (\sum_{i=1}^{5}\sum_{j=1}^{2}(m_i^{(j)})^2)/2\sum_{i=1}^{5} s_i^2$ obeys the F-distribution with 10 and 5 degrees of freedom.

The statistical significance of FLAP on all the datasets is validated by Table IV, in which the F-statistics values of baselines versus FLAP are listed. Table IV suggests that the hypothesis is rejected in most cases, which statistically verifies that FLAP outperforms the baseline algorithms. Besides, we observe that the null hypothesis is often accepted by MinCut on *Wine*, and LNP on *Teapot*, which suggest that they achieve comparable performances with FLAP on the corresponding datasets. Figs. 6(c) and 8(b) also show this point.

H. Computational Cost

To further evaluate the computational efficiency of baselines and the proposed FLAP, this section compares the CPU seconds of all the algorithms on all the adopted vision datasets. The results are presented in Table V. It can be observed that HF is the most efficient algorithm among the noniterative methods, and the proposed FLAP generally needs the least CPU time compared with other iteration-based approaches. By properly setting the parameter η, FLAP can perfectly control the distribution of eigenvalues of its iteration matrix, so it achieves the fastest convergence speed among the compared iterative algorithms. The detailed reason will be explained in the next section. Besides, we note that the noniterative methods are more efficient than the iterative algorithms when the dataset is small (e.g., *Iris* and *Wine* datasets). However, when the database contains a large number of examples, e.g., *Digits* with 5620 examples, the proposed iterative FLAP begins to show its strength in terms of efficiency. Compared with noniterative methods (MinCut, HF, and NTK) that take more than 30 seconds to complete one implementation, FLAP requires only less than 20 CPU seconds. The reason is that noniterative methods usually need to invert a large

TABLE V
CPU TIME (IN SECONDS) OF VARIOUS METHODS. (FOR EACH l, THE SMALLEST RECORD AMONG ITERATIVE METHODS IS MARKED IN RED, WHILE THE SMALLEST RECORD AMONG NONITERATIVE METHODS IS HIGHLIGHTED IN BLUE)

		Non-iterative methods			Iterative methods		
	l	MinCut	HF	NTK	LGC	LNP	FLAP
Iris	6	0.006	0.002	0.324	0.023	0.025	0.021
	12	0.008	0.002	0.356	0.034	0.040	0.024
	18	0.009	0.002	0.356	0.034	0.046	0.026
	24	0.008	0.002	0.412	0.038	0.062	0.029
	30	0.008	0.002	0.416	0.048	0.056	0.029
Wine	6	0.012	0.004	0.395	0.102	0.019	0.038
	12	0.015	0.002	0.313	0.105	0.016	0.040
	18	0.010	0.003	0.377	0.110	0.020	0.043
	24	0.023	0.002	0.386	0.111	0.018	0.045
	30	0.017	0.002	0.426	0.111	0.031	0.045
BreastCancer	4	0.122	0.059	0.422	0.254	0.746	0.226
	8	0.111	0.078	0.507	0.330	0.827	0.310
	12	0.109	0.061	0.493	0.376	1.156	0.373
	16	0.122	0.057	0.544	0.401	1.771	0.405
	20	0.123	0.057	0.542	0.422	1.768	0.420
CNAE-9	18	0.352	0.151	1.207	3.877	3.904	2.967
	36	0.380	0.147	1.689	4.015	6.148	3.146
	54	0.381	0.147	2.752	4.156	6.622	3.235
	72	0.379	0.149	1.717	4.209	6.844	3.369
	90	0.318	0.150	2.112	4.218	7.120	3.484
Digits	60	40.659	33.811	35.057	41.903	38.486	12.124
	80	39.154	34.344	34.268	45.747	41.473	14.490
	100	40.027	34.086	38.052	48.374	43.471	16.658
	120	40.006	34.117	37.615	50.508	46.011	18.518
	140	42.180	33.902	40.585	52.173	47.958	19.757
Teapot	4	0.026	0.011	0.422	0.088	0.481	0.010
	8	0.027	0.014	0.473	0.106	0.477	0.031
	12	0.027	0.013	0.450	0.118	0.488	0.043
	16	0.029	0.012	0.461	0.115	0.493	0.049
	20	0.028	0.012	0.468	0.124	0.506	0.058
LFW	20	0.050	0.012	0.993	0.295	0.180	0.169
	40	0.036	0.015	1.110	0.213	0.202	0.209
	60	0.060	0.014	1.751	0.219	0.219	0.213
	80	0.036	0.012	2.780	0.228	0.239	0.245
	100	0.038	0.015	4.261	0.235	0.252	0.269

matrix when massive examples exist, while FLAP successfully avoids this inversion by conducting in an iterative way, so FLAP is more suitable for dealing with relatively large data collections.

I. Parametric Settings

Two parameters, K and η, are critical for FLAP to obtain satisfying results. K controls the sparsity of an established graph, and η plays an important role in balancing the accuracy and efficiency.

1) Choosing η: The impacts of η for FLAPs performance are twofold. Intuitively, from (15) we observe that a diagonally dominant iteration matrix (such as $\mathbf{P}$ in FLAP) leads to higher convergence rate. In other words, a small η helps to reduce the iteration times. The following analyses will elaborate this point strictly.

Lemma 4: Suppose $\mathbf{G}_{n\times n}$ is the iteration matrix of an iterative algorithm, e.g., FLAP defined in (11), of which the eigenvalues are $\xi_1, \xi_2, \ldots, \xi_n$, and then the upper bound of the iteration times $t_{\max}$ satisfies

$$\sum_{i=1}^{n} (\alpha\xi_i)^{2t_{\max}}(\xi_i - 1)^2 = \varepsilon^2/(l\alpha^2) \tag{43}$$

where ε, l and α are the same as those defined above.

Lemma 4 is proven in Appendix B. According to Lemma 4, we investigate what requirement of the eigenvalues of the iteration matrix should meet to obtain the minimum $t_{\max}$. In fact, this question is equal to the following optimization problem:

$$\begin{aligned} \min \quad & t_{\max} \\ \text{s.t.} \quad & \sum_{i=1}^{n} (\alpha\xi_i)^{2t_{\max}}(\xi_i - 1)^2 = \tfrac{\varepsilon^2}{l\alpha^2} \end{aligned} \tag{44}$$

which can be rewritten as

$$\min \quad t_{\max} + \rho_0 \left(\sum_{i=1}^{n} (\alpha\xi_i)^{2t_{\max}}(\xi_i - 1)^2 - \frac{\varepsilon^2}{l\alpha^2} \right) \tag{45}$$

where ρ_0 is the Lagrange multiplier. If we denote

$$F(\xi_1, \ldots, \xi_n, \rho_0) = t_{\max} + \rho_0 \left(\sum_{i=1}^{n} (\alpha\xi_i)^{2t_{\max}}(\xi_i - 1)^2 - \frac{\varepsilon^2}{l\alpha^2} \right) \tag{46}$$

then solving (44) equals to finding the solution of

$$\begin{cases} \frac{\partial F}{\partial \rho_0} = \sum_{i=1}^{n} (\alpha\xi_i)^{2t_{\max}}(\xi_i - 1)^2 - \frac{\varepsilon^2}{l\alpha^2} = 0 \\ \frac{\partial F}{\partial \lambda_1} = 4\rho_0 t_{\max}\alpha^{2t_{\max}}\xi_1^{2t_{\max}-1}(\xi_1 - 1) = 0 \\ \quad \vdots \\ \frac{\partial F}{\partial \lambda_n} = 4\rho_0 t_{\max}\alpha^{2t_{\max}}\xi_n^{2t_{\max}-1}(\xi_n - 1) = 0. \end{cases} \tag{47}$$

It can be concluded that (47) is maximally satisfied if ξ_i equals to 0 or 1 (note that $\varepsilon^2/l\alpha^2 \approx 0$). Fortunately, Theorem 5 guarantees that all the eigenvalues of FLAP's iteration matrix $\mathbf{P}$ are almost equivalent to 1 by choosing a small η.

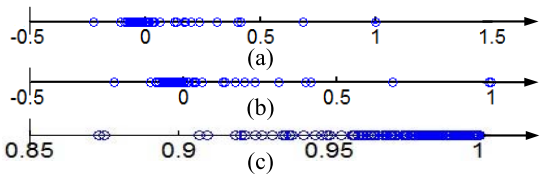


Fig. 10. Distribution of eigenvalues on *Iris*. (a) LNP. (b) LGC. (c) FLAP. Note that the ranges of the three x-axes are different.

Theorem 5: Let λ_i $(1 \le i \le n)$ be the eigenvalues of FLAP's iteration matrix, then $1-2\eta \le \lambda_i \le 1$ where $\eta = \gamma \max_j \sum_{k=1,k\neq j}^{n} d_{kj}^{-2}$ as defined in the beginning of Section IV.

Proof: To prove Theorem 5, we need a lemma from [28].

Lemma 6 [28]: Suppose $\mathbf{A} = (a_{ij})$ is a stochastic matrix, $q = \min\{a_{ii}, i \in N\}$, then all the eigenvalues of $\mathbf{A}$ satisfy $|\lambda_i - q| \le 1 - q$.

Now we begin to prove the Theorem 5. Since the iteration matrix $\mathbf{P}$ of FLAP is a symmetric stochastic matrix, all its eigenvalues are in the real range $[-1, 1]$, so according to Lemma 6, q in our case satisfies

$$\begin{aligned} q &= 1 - \gamma \max_j \sum_{k=1,k\neq j}^{n} d_{kj}^{-2} \\ &= 1 - \eta \left(\max_j \sum_{k=1,k\neq j}^{n} d_{kj}^{-2} \right)^{-1} \max_j \sum_{k=1,k\neq j}^{n} d_{kj}^{-2} = 1 - \eta. \end{aligned}$$

By substituting the expression of q into Lemma 6, we complete the proof.

Above analyses explain why we set the parameter η (or γ) to a relatively small positive number (e.g., $\eta = 0.1$ in *Iris*, $\eta = 0.01$ in *Wine*, and $\eta = 0.001$ in *BreastCancer*, etc.). By choosing a small η, $\mathbf{P}$ can be diagonally dominant and its eigenvalues $\lambda_i (1 \le i \le n)$ will be distributed very close to 1. Fig. 10 shows that the eigenvalues of the FLAP iteration matrix are only slight smaller than 1. In contrast, the eigenvalues of the iteration matrices in LNP and LGC are widely scattered in the range $[-1, 1]$. Therefore, FLAP is able to converge relatively more quickly than LGC and LNP as Table V shows.

On the other hand, if η is set to an extremely small value, then $\mathbf{P}$ will be very close to an identity matrix. Equation (13) reveals that under this situation, the convergence result is almost the same as the initial state $\mathbf{Y}$, which means the label information cannot be thoroughly propagated from initially labeled examples to other unlabeled ones. Therefore, an extremely small η will damage the classification accuracy significantly.

In short, a small η leads to higher convergence rate, but decreases the classification accuracy; a large η can boost the accuracy at the cost of high computational time. Therefore, η should be chosen by trading off the accuracy and efficiency. By fixing $l = 12$ on the *Iris* dataset, we plot the accuracy and iteration times of FLAP with respect to the variations of η in Fig. 11(a) and (b). These two figures also indicate that both the accuracy and iteration times will rise by gradually increasing η. In the *Iris* dataset, we set η to 0.1 because the accuracy is relatively high and the iteration times are also acceptable under this setting.

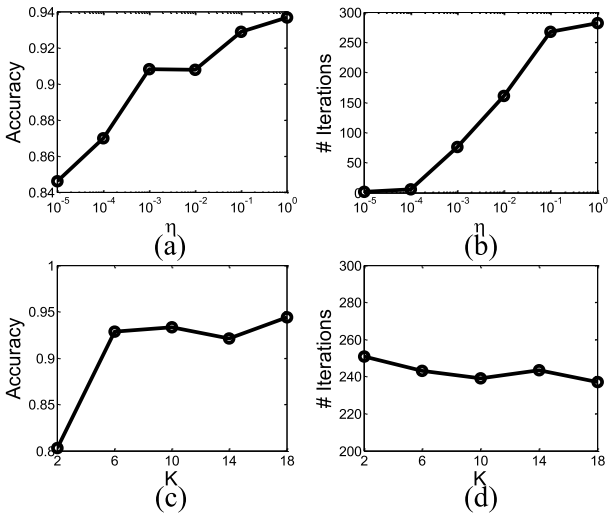


Fig. 11. Impact of parametric settings on accuracy and iteration times. (a) and (b) Investigation of η. (c) and (d) Evaluation of K.

2) Choosing K: A suitable graph is very important for improving the performance. As mentioned above, K is a critical parameter determining the number of neighborhoods and the sparsity of an established graph. This section studies how K influences the classification accuracy and iteration times. In Fig. 11(c) and (d), we fix $\eta = 0.1$ and change the value of K from small to large. It can be observed that if the graph is too sparse (e.g., $K = 2$), FLAP will not achieve satisfying performance. However, when K is larger than a certain value (e.g., $K = 6$), FLAP functions properly and produces encouraging results. Besides, Fig. 11(d) reveals that the choice of K will not influence the iteration times significantly. Therefore, both the accuracy and iteration times are not sensitive to the choice of K if K is not too small. In other words, this parameter can be easily tuned.

J. Summary of Experiments

In general, all the compared methods are able to obtain promising performance on various datasets. Specifically, statistical tests reveal that FLAP achieves higher classification accuracy than the other baselines in most cases (Table IV). Among the iterative baselines including LGC and LNP, LGC obtains the similar accuracy to FLAP. However, it usually requires more iteration times than FLAP due to the irregular distribution of eigenvalues of its iteration matrix (Figs. 6 and 7). Noniterative methods including HF, MinCut, and NTK are usually more efficient than FLAP on small-size datasets, but become very inefficient if the size of dataset is relatively large (see Table V). MinCut and HF perform comparably to FLAP (Figs. 6–9) because all of them share the same innovation, i.e., exploring the manifold embedded in datasets.

V. Conclusion

We presented the FLAP algorithm to propagate labels for SSL by adopting Fick's First Law of Diffusion in fluid mechanics. We showed FLAP can also be derived in the context of the traditional regularization theory, which not only relates the FLAP algorithm to existing algorithms but also implies that the convergent point of FLAP is globally optimal. It was also demonstrated that the parameter η played an important role in determining classification performance and iteration times. Comprehensive experimental results suggest that FLAP obtains competitive performance when compared with state-of-the-art SSL algorithms.

Similar to most label propagation algorithms, it is not computationally tractable to straightforwardly apply FLAP to big data problems. This is because the complexity of graph construction is $O(n^3)$. However, recent development of hypergraph techniques, such as [24] and [29], can be employed to address this problem.

Appendix A
Proof of Theorem 1

Proof: According to Theorem 2, the sequence $\left\{\mathbf{F}^{(t)}\right\}$ converges to

$$\mathbf{F}^* = (1-\alpha)(\mathbf{I}-\alpha\mathbf{P})^{-1}\mathbf{Y}. \tag{48}$$

By plugging (19) into (48), $\mathbf{F}^*$ can be represented as

$$\begin{aligned}\mathbf{F}^* &= (1-\alpha)[\mathbf{I}+\alpha\mathbf{U}(\Lambda^{-1}-\alpha\mathbf{I})^{-1}\mathbf{U}^T]\mathbf{Y} \\ &= \mathbf{U}\text{diag}\left(\tfrac{1-\alpha}{1-\alpha\lambda_1}\ \tfrac{1-\alpha}{1-\alpha\lambda_2}\ \cdots\ \tfrac{1-\alpha}{1-\alpha\lambda_n}\right)\mathbf{U}^T\mathbf{Y}.\end{aligned} \tag{49}$$

According to Theorem 5 and Lemma 6, all the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ are in the range $[1-2\eta, 1]$ where η is set to a very small number as suggested. Therefore, if we denote $\Omega = \text{diag}\ (1-\alpha/1-\alpha\lambda_1 1-\alpha/1-\alpha\lambda_2\ \cdots\ 1-\alpha/1-\alpha\lambda_n)^{1/2}$, then the upper bound of the relative error between Ω and $\mathbf{I}$ is

$$\begin{aligned}\frac{\|\Omega-\mathbf{I}\|_F}{\|\Omega\|_F} &= \sqrt{\sum_{i=1}^{n}\left[\frac{(\lambda_i-1)\alpha}{1-\alpha\lambda_i}\right]^2 \Big/ \sum_{i=1}^{n}\left(\frac{1-\alpha}{1-\alpha\lambda_i}\right)^2} \\ &\le \frac{2\eta\alpha}{1-\alpha}.\end{aligned} \tag{50}$$

Note that the value of the right hand in above inequality is very small because of the small η, so (50) reveals that Ω is very close to the identity matrix $\mathbf{I}$, hence $\mathbf{F}^*$ in (49) can be reformulated as

$$\mathbf{F}^* \approx \mathbf{U}\mathbf{I}\mathbf{U}^T\mathbf{Y} = \mathbf{Y} \tag{51}$$

which indicates that $\mathbf{F}^*$ is very similar to the initial state $\mathbf{Y}$. Therefore, the labels of labeled examples are considered as unchanged.

Appendix B
Proof of Lemma 4

Proof: Suppose the general iterative expression of graph-based SSL is

$$\mathbf{F}^{(t+1)} = \alpha\mathbf{G}\mathbf{F}^{(t)} + (1-\alpha)\mathbf{Y} \tag{52}$$

where $\mathbf{G}$ is the iteration matrix, then according to the stopping criterion

$$\|\mathbf{F}^{(t+1)} - \mathbf{F}^{(t)}\|_F < \varepsilon \tag{53}$$

we have the following inequality:

$$\begin{aligned}\|\mathbf{F}^{(t+1)}-\mathbf{F}^{(t)}\|_F &= \alpha^{t+1}\|(\mathbf{G}-\mathbf{I})\mathbf{G}^t\mathbf{Y}\|_F \\ &\le \alpha^{t+1}\|(\mathbf{G}-\mathbf{I})\mathbf{G}^t\|_F\|\mathbf{Y}\|_F \le \varepsilon \quad (54)\end{aligned}$$

where $\|\mathbf{Y}\|_F = \sqrt{l}$. Similarly, $\mathbf{G}$ can also be decomposed into $\mathbf{G} = \mathbf{V}\mathbf{D}\mathbf{V}^T$, where $\mathbf{V}$ is an unitary matrix and $\mathbf{D} = \mathrm{diag}(\xi_1, \xi_2, \ldots, \xi_n)$ is a diagonal matrix containing n eigenvalues of $\mathbf{G}$. If $\mathbf{V}$ is partitioned by columns as $\mathbf{V} = (\mathbf{V}_1 \ \mathbf{V}_2 \cdots \mathbf{V}_n)$, then $\mathbf{G} = \sum_{i=1}^n \xi_i \mathbf{V}_i \mathbf{V}_i^T$. Moreover, considering that the identity matrix $\mathbf{I} = \mathbf{V}\mathbf{V}^T = \sum_{i=1}^n \mathbf{V}_i \mathbf{V}_i^T$, we have

$$\begin{aligned}\|(\mathbf{G}-\mathbf{I})\mathbf{G}^t\|_F &= \left\|\left(\sum_{i=1}^n \xi_i \mathbf{V}_i\mathbf{V}_i^T - \mathbf{I}\right)\left(\sum_{i=1}^n \lambda_i^t \mathbf{V}_i\mathbf{V}_i^T\right)\right\|_F \\ &= \sqrt{\sum_{i=1}^n \xi_i^{2t}(\xi_i-1)^2 \mathrm{tr}(\mathbf{V}_i\mathbf{V}_i^T)}. \quad (55)\end{aligned}$$

Because $\|\mathbf{V}_i\|_2 = 1$, so $\mathrm{tr}(\mathbf{V}_i\mathbf{V}_i^T) = 1$ for $1 \le i \le n$. Hence (55) can be further simplified as

$$\left\|(\mathbf{G}-\mathbf{I})\mathbf{G}^t\right\|_F = \sqrt{\sum_{i=1}^n \xi_i^{2t}(\xi_i-1)^2}. \quad (56)$$

Finally, by substituting (56) into (54) we have

$$\sum_{i=1}^n (\alpha\xi_i)^{2t_{\max}}(\xi_i-1)^2 = \frac{\varepsilon^2}{l\alpha^2}. \quad (57)$$

This reveals the relationship between the upper bound of iteration times $t_{\max}$ and the eigenvalues of the iteration matrix $\mathbf{G}$.

REFERENCES

[1] X. Zhu and B. Goldberg, *Introduction to Semi-Supervised Learning*. San Rafael, CA, USA: Morgan & Claypool, 2009.

[2] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proc. Int. Conf. Mach. Learn.*, vol. 99. Jun. 1999, pp. 200–209.

[3] G. Fung and O. L. Mangasarian, "Semi-semi-superyised support vector machines for unlabeled data classification," *Optim. Methods Softw.*, vol. 15, no. 1, pp. 29–44, Oct. 2001.

[4] Y. Wang, S. Chen, and Z.-H. Zhou, "New semi-supervised classification method based on modified cluster assumption," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 689–702, May 2012.

[5] T. Joachims, "Transductive learning via spectral graph partitioning," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 290–297.

[6] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. Int. Conf. Mach. Learn.*, Washington, DC, USA, Aug. 2003, pp. 912–919.

[7] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2003, pp. 321–328.

[8] 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, Nov. 2006.

[9] L. Chen, I. W. Tsang, and D. Xu, "Laplacian embedded regression for scalable manifold regularization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 6, pp. 902–915, Jun. 2012.

[10] Z. Xu, I. King, M. R.-T. Lyu, and R. Jin, "Discriminative semi-supervised feature selection via manifold regularization," *IEEE Trans. Neural Netw.*, vol. 21, no. 7, pp. 1033–1047, Jul. 2010.

[11] Y. Huang, D. Xu, and F. Nie, "Semi-supervised dimension reduction using trace ratio criterion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 3, pp. 519–526, Mar. 2012.

[12] M. Wu and B. Schölkopf, "Transductive classification via local learning regularization," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2007, pp. 628–635.

[13] J. Wang, F. Wang, C. Zhang, H. C. Shen, and L. Quan, "Linear neighborhood propagation and its applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 9, pp. 1600–1615, Sep. 2009.

[14] X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty, "Nonparametric transforms of graph kernels for semi-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2005, pp. 1641–1648.

[15] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*, vol. 2. Cambridge, MA, USA: MIT Press, 2006.

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, vol. 3. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1996.

[17] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*. Philadelphia, PA, USA: SIAM, 1996.

[18] J. Wang, T. Jebara, and S.-F. Chang, "Graph transduction via alternating minimization," in *Proc. Int. Conf. Mach. Learn.*, Helsinki, Finland, 2008, pp. 1144–1151.

[19] C. Alzate and J. Suykens, "A semi-supervised formulation to binary kernel spectral clustering," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Brisbane, Australia, Jun. 2012, pp. 1–8.

[20] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*. London, U.K.: Chapman & Hall, 2005.

[21] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *Proc. Int. Conf. Mach. Learn.*, vol. 2. Jul. 2002, pp. 315–322.

[22] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[23] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," in *Proc. Int. Conf. Mach. Learn.*, Pittsburgh, PA, USA, 2006, pp. 321–328.

[24] X. Zhu and J. Lafferty, "Harmonic mixtures: Combining mixture models and graph-based methods for inductive and scalable semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, Bonn, Germany, Aug. 2005, pp. 1052–1059.

[25] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Dec. 2001, pp. 1511–1518.

[26] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Sep./Oct. 2009, pp. 365–372.

[27] E. Alpaydin, "Combined 5 × 2 cv F test for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 11, no. 8, pp. 1885–1892, Nov. 1999.

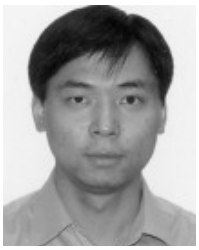
[28] T. Huang and C. Yang, *The Analysis and Applications of Special Matrix*. China: Science Press, 2007.

[29] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, Haifa, Israel, 2010, pp. 679–686.

Chen Gong received the bachelor's degree from the East China University of Science and Technology, Shanghai, China, in 2010. He is currently pursuing the dual Ph.D. degree with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai, and the Centre for Quantum Computation & Intelligent Systems, University of Technology, Sydney, Australia, under the supervision of Prof. J. Yang and Prof. D. Tao.

His current research interests include machine learning, data mining, and learning-based vision problems.

Dr. Gong has authored 21 technical papers at prominent journals and conferences, such as the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON CYBERNETICS, the Association for the Advancement of Artificial Intelligence, and the IEEE International Conference on Multimedia and Expo.



Dacheng Tao (M'07–SM'12–F'15) is Professor of Computer Science with the Centre for Quantum Computation & Intelligent Systems, and the Faculty of Engineering and IT in the University of Technology, Sydney. He mainly applies statistics and mathematics for data analytics and his research interests spread across computer vision, computational neuroscience, data mining, geoinformatics, image processing, machine learning, medical informatics, multimedia, neural networks and video surveillance.

His research results have expounded in one monograph and over 200 publications at prestigious journals and prominent conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE TRANSACTIONS ON MEDICAL IMAGING, the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, TRANSACTIONS ON CYBERNETICS, the *Journal of Machine Learning Research*, the *International Journal of Computer Vision*, the Neural Information Processing Systems Conference, the International Conference on Machine Learning, the IEEE Conference on Computer Vision and Pattern Recognition, the International Conference on Computer Vision, the European Conference on Computer Vision, the International Conference on Artificial Intelligence and Statistics, the IEEE International Conference on Data Mining, and the ACM SIGKDD and Multimedia conferences, with several best paper awards, such as the best theory/algorithm paper runner up award in IEEE ICDM'07, the best student paper award in IEEE ICDM'13, and the 2014 ICDM 10 Year Highest-Impact Paper Award.

Keren Fu received the B.Sc. degree in automation from the Huazhong University of Science and Technology, Wuhan, China, in 2011. He is currently pursuing the Ph.D. degree with the Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai, China, under the supervision of Prof. J. Yang.

His current research interests include object detection, saliency detection, visual tracking, and machine learning.

Jie Yang received the Ph.D. degree from the Department of Computer Science, University of Hamburg, Hamburg, Germany, in 1994.

He is currently a Professor with the Institute of Image Processing and Pattern recognition, Shanghai Jiao Tong University, Shanghai, China. He has led many research projects (e.g., National Science Foundation, 863 National High Technology Plan), published one book in Germany, and authored over 200 journal papers. His current research interests include object detection and recognition, data fusion and data mining, and medical image processing.