MATHEMATICAL ENGINEERING TECHNICAL REPORTS

Electric Network Classifiers for Semi-Supervised Learning on Graphs

Hiroshi HIRAI, Kazuo MUROTA, and
Masaki RIKITOKU

METR 2005–24 August 2005

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.i.u-tokyo.ac.jp/mi/mi-e.htm

Electric Network Classifiers for Semi-Supervised Learning on Graphs

Hiroshi HIRAI
Research Institute for Mathematical Sciences,
Kyoto University, Kyoto 606-8502, Japan
hirai@kurims.kyoto-u.ac.jp

Kazuo MUROTA
Department of Mathematical Informatics,
Graduate School of Information Science and Technology,
University of Tokyo, Tokyo 113-8656, Japan
murota@mist.i.u-tokyo.ac.jp

Masaki RIKITOKU
Justsystem Corporation Innovative Technology R&D Dept.
Aoyama bldg. 1-2-3, Kita-Aoyama, Tokyo 107-8640, Japan
Masaki_Rikitoku@justsystem.co.jp

August 2005

Abstract

We propose a new classifier, named *electric network classifiers*, for semi-supervised learning on graphs. Our classifier is based on nonlinear electric network theory and classifies data set with respect to the sign of electric potential. Close relationships to C-SVM and graph kernel methods are revealed. Unlike other graph kernel methods, our classifier does not require heavy kernel computations and obtain the potential directly using efficient network flow algorithms. Furthermore, with flexibility of its formulation, our classifier can incorporate various edge characteristics; influence of edge direction, unsymmetric dependence and so on. Therefore, our classifier has the potential to tackle large complex real world problems. Experimental results show that the performance is fairly good compared with the diffusion kernel and other standard methods.

1 Introduction

We consider semi-supervised classification problems on graphs, in which some vertices of the graph are labeled as positive or negative, and others are unlabeled. The task is to classify the unlabeled data. Such problems arise in biological networks [14] and text classification [6]. One possible approach to this problem is SVM and other kernel-based methods [11]. The central issue in kernel-based methods is how to construct or learn a kernel from a given graph. The *diffusion kernel* [7] is such a graph kernel constructed from the graph Laplacian. Beyond the diffusion kernel, several learning kernel algorithms have been proposed (see [12, 6, 15, 13]). However, to construct kernel matrix using graph Laplacian is a very heavy computational task; it requires a large amount of memory for the kernel matrix, diagonalizations, and optimizations on the matrix space.

Here we introduce a new binary classifier, named *electric network classifier*, for semi-supervised learning on graphs, based on nonlinear electric network theory. Our approach constructs a kernel only implicitly and classifies unlabeled data directly using electric potential. In so doing, we can avoid heavy kernel computations and obtain the potential using fast network flow algorithms. Furthermore, our classifier can incorporate the influence of edge direction (unilateral or unsymmetric dependence) and other edge characteristics unlike other graph kernels considered so far. Thus our classifiers have the potential to tackle large complex real-world problems. Experimental results show that the performance is fairly good compared with diffusion and linear kernels. Our classifiers can be understood as a kind of discrete version of *coulomb classifiers* introduced by Hochreiter, Mozer, and Obermayer [4] that relies on an analogy with electrostatics. They can also be regarded as a nonlinear extension of semi-supervised learning on graph based on Gaussian random field model proposed by Zhu, Ghahramani, and Lafferty [15]. Therefore, a random walk interpretation is also possible; see [1] for the relationship between electric networks and random walks.

This paper is organized as follows. In Section 2, before introducing our classifier, we discuss a general framework for semi-supervised learning using *monotropic programming.* This framework is very flexible and clarifies the mathematical structure of our classifier. Then we introduce electric network classifiers as its special case. In Section 3, we show experimental results.

2 Electric Network Classifiers

In this section, we introduce electric network classifiers and investigate their mathematical properties, with emphasis on its connection to the standard C-SVM framework of [11]. First, we propose a general framework for semi-supervised learning using *monotropic programming* of R.T. Rockafellar [10] and discuss its relationship to kernel methods. Next, we introduce electric network classifiers as its special case.

2.1 Monotropic Programming Framework for Semi-Supervised Learning

Let V be an input data space, $U \subseteq V$ a training data set, and $y : U \to \{-1, 1\}$ its label. To design a classifier, we assume an auxiliary space E together with a linear map $A : \mathbf{R}^E \to \mathbf{R}^V$, or a matrix called *structure matrix*, which represents a discrete structure of V. In the canonical case of a directed graph, V is the vertex set, E is the edge set, and A is the incidence matrix. More generally, in the case of simplicial complex, we can choose V to be $n-1$ dimensional faces, E to be n dimensional faces and A to be the boundary operator.

The proposed classifier constructs *discriminant potential* $p^* : V \to \mathbf{R}$ and classify the data set according to the sign of p^* as

$$\begin{cases} p_i^* \geq 0 & \Rightarrow \quad \text{the label of } i \text{ is } +1, \\ p_i^* < 0 & \Rightarrow \quad \text{the label of } i \text{ is } -1. \end{cases} \tag{2.1}$$

This potential p^* is constructed from the optimal solution of a monotropic programming problem [10], which consists of the following dual pair of convex optimization problems.

$$[\mathrm{P}] \qquad \begin{cases} \displaystyle\min_{\xi \in \mathbf{R}^E, u \in \mathbf{R}^U} & \displaystyle\sum_{e \in E} f_e(\xi_e) + \sum_{j \in U} g_j(u_j) \\ \text{s.t.} & (A\xi)_i = \begin{cases} 0 & \text{if } i \in V \setminus U, \\ u_i & \text{if } i \in U, \end{cases} \end{cases} \tag{2.2}$$

$$[\mathrm{D}] \qquad \begin{cases} \displaystyle\min_{\eta \in \mathbf{R}^E, p \in \mathbf{R}^V} & \displaystyle\sum_{e \in E} f_e^*(\eta_e) + \sum_{j \in U} g_j^*(-p_j) \\ \text{s.t.} & \eta = A^\top p, \end{cases} \tag{2.3}$$

where $f_e, g_j : \mathbf{R} \to \mathbf{R} \cup \{+\infty\}$ are convex functions and $f_e^*, g_j^* : \mathbf{R} \to \mathbf{R} \cup \{+\infty\}$ are the *Legendre transforms* of f_e and g_j defined, respectively, as

$$f_e^*(\eta_e) = \sup_{\xi_e \in \mathbf{R}} \{\eta_e \xi_e - f_e(\xi_e)\}, \quad g_j^*(q_j) = \sup_{u_j \in \mathbf{R}} \{q_j u_j - g_j(u_j)\}. \tag{2.4}$$

In this optimization problem [P], the convex functions g_j and the variables u_j play the role of teaching signals by the training set. In particular, we choose g_j^* as a kind of a penalty function like

$$g_j^*(-p_j) \quad \begin{cases} = 0 & \text{if } 1 - y_j p_j \leq 0, \\ > 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

The convex functions f_e, f_e^* play the role of the regularization. The canonical choice of f_e, f_e^* is the following squared-norm type function

$$f_e(\xi_e) = r_e \xi_e^2/2, \quad f_e^*(\eta_e) = \eta_e^2/2r_e \quad (e \in E), \tag{2.6}$$

where r_e is a positive parameter. On the basis of an optimal solution (η^*, p^*) to [D], we classify data set V according to the sign of p^* as (2.1). We call this p^* an *optimal discriminant potential.*

The relationship between our approach and kernel methods is revealed in the special case of f_e given by (2.6). Let $A^+ : \mathbf{R}^V \to \mathbf{R}^E$ be a *reflexive minimum-norm generalized inverse* of A with respect to the squared norm $\sum_{e \in E} f_e$ in (2.6), i.e., A^+ satisfies

$$AA^+A = A, \quad A^+AA^+ = A^+ \tag{2.7}$$

and that for any $y \in \operatorname{Im} A$, A^+y is the minimum norm point of $\{x \in \mathbf{R}^V \mid Ax = y\}$; see [8] for generalized inverses. From A^+, we define a positive semidefinite kernel $K : V \times V \to \mathbf{R}$ as

$$K(i,j) = ((A^+)^\top R A^+)_{ij} \quad (i, j \in V), \tag{2.8}$$

where R is the diagonal matrix whose diagonal entries are $\{r_e\}_{e \in E}$, and let D be a matrix satisfying $\operatorname{Im} A = \operatorname{Ker} D$. Then, we have the following.

Theorem 2.1. *The problem* [P] *with* f_e *of* (2.6) *is equivalent to*

$$[\mathrm{P}'] \qquad \min_{u\in\mathbf{R}^U} \quad \frac{1}{2}\sum_{i,j\in U} K(i,j)u_i u_j + \sum_{j\in U} g_j(u_j) \tag{2.9}$$

$$\text{s.t.} \quad \sum_{j\in U} D_{kj}u_j = 0 \ (\forall k : \textit{row index of } D). \tag{2.10}$$

Let u^* *be an optimal solution to* [P′] *and* μ *an optimal Lagrange multiplier of the equality constraints* (2.10)*. Then an optimal discriminant potential* p^* *is given as*

$$p_i^* = \sum_{j\in U} K(i,j)u_j^* + (D^\top \mu)_i \quad (i \in V). \tag{2.11}$$

Recall C-SVM classifier [11], which is obtained by solving the following optimization problem

$$[\text{C-SVM}]: \min_{\alpha\in\mathbf{R}^U} \quad \frac{1}{2}\sum_{i,j\in U} \alpha_i\alpha_j y_i y_j K(i,j) - \sum_{i\in U}\alpha_i$$

$$\text{s.t.} \quad \sum_{i\in U} y_i\alpha_i = 0, \ 0 \le \alpha_i \le C \quad (i \in U),$$

where C is a penalty parameter that is a positive real number or $+\infty$. Let α^* be an optimal solution of [C-SVM] and b^* an optimal Lagrange multiplier of the equality constraint. Then SVM decision function $f : V \to \mathbf{R}$ is given as

$$f(i) = \sum_{j\in U} y_j\alpha_j^* K(j,i) + b^* \quad (i \in V). \tag{2.12}$$

The relationship to C-SVM framework is summarized as follows.

Corollary 2.2. *If* $\mathrm{Im}\, A = \mathrm{Ker}\, \mathbf{1}$ *and*

$$g_j(u_j) = \begin{cases} -y_j u_j & \text{if } 0 \le y_j u_j \le C, \\ +\infty & \text{otherwise,} \end{cases} \tag{2.13}$$

$$g_j^*(q_j) = C\max(0, 1 + y_j q_j) \tag{2.14}$$

for $j \in U$ *with some positive parameter* C*, then the problem* [P′] *coincides with C-SVM and the optimal discriminant potential* p^* *defined as* (2.11) *coincides with SVM decision function.*

Remark 2.3. In semi-supervised learning, we may assume that data set V and structure matrix A are stacked in a computer memory. Hence the computation of kernel matrix K is not necessary since we can obtain potential p^* by solving the dual problem [D] directly. This p^* generally does not coincide with (2.11) if the optimal potential is not unique.

2.2 Electric Network Classifiers

We introduce *electric network classifiers* on graphs as a special case of the monotropic programming framework. Let $G = (V, E)$ be a directed graph, $U \subseteq V$ a training set, and $y : U \to \{-1, 1\}$ its label. We treat the vertex set

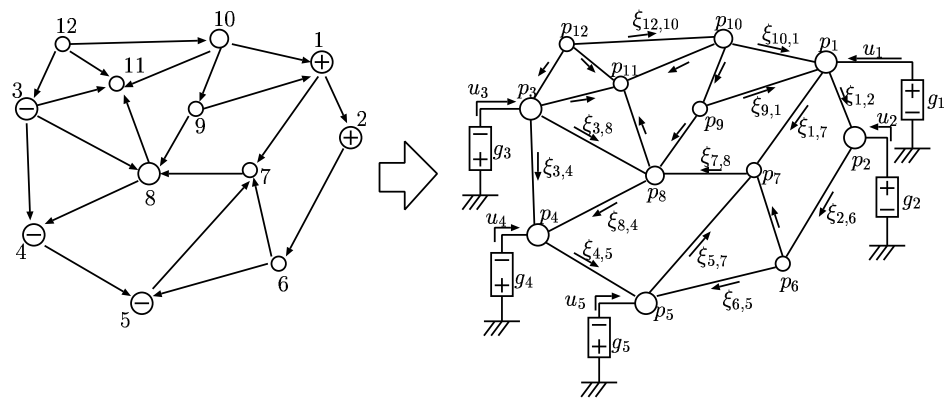

Figure 1: Physical interpretation of electric network classifiers

V as the data space, the edge set E as the auxiliary space, and the incidence matrix

$$A(v,e) = \begin{cases} 1 & \text{if } e \text{ leaves } v, \\ -1 & \text{if } e \text{ enters } v, \\ 0 & \text{otherwise,} \end{cases} \quad (v \in V,\ e \in E) \tag{2.15}$$

as the structure matrix. In this setting, we consider the optimization problems [P] and [D] with some convex functions $\{f_e\}_{e\in E}$ and $\{g_j\}_{j\in U}$. This problem is exactly the same as the *nonlinear network flow problem* [5, 10]. Then the following physical interpretation is valid; see also Figure 1.

$\xi \in \mathbf{R}^E$	: currents on edges
$u \in \mathbf{R}^U$	: currents flowing into labeled vertices from the earth
f_e, g_j	: current energy on edges
$\eta \in \mathbf{R}^E$	: potential differences on edges
$p \in \mathbf{R}^V$	: potential on vertices
f_e^*, g_j^*	: potential energy on edges

Each potential on vertices is normalized so that the earth has zero potential. We call this classifier an *electric network classifier.* With general convex functions on the edges, the electric network classifier can incorporate various types of edge characteristics; influence of edge direction, unsymmetric dependence, and so on.

When the electric network consists exclusively of Ohmic resistors, we have

$$f_e(\xi_e) = r_e \xi_e^2/2 \quad (e \in E), \tag{2.16}$$

where r_e denotes the resistances. With the choice of g_j given in (2.13) our electric network classifier coincides with C-SVM using kernel (2.8), where the graph (V,E) is assumed to be connected. Furthermore, this kernel admits an intuitive interpretation, as follows.

Theorem 2.4 ([3]). *For f_e in* (2.16)*, kernel K in* (2.8) *can be taken as*

$$K(i,j) = \{d(i,i_0) + d(j,i_0) - d(i,j)\}/2 \quad (i,j \in V), \tag{2.17}$$

where $d : V \times V \to \mathbf{R}$ is the electric distance defined as

$$d(i,j) = \textit{the electric resistance between } i \textit{ and } j \quad (i,j \in V) \tag{2.18}$$

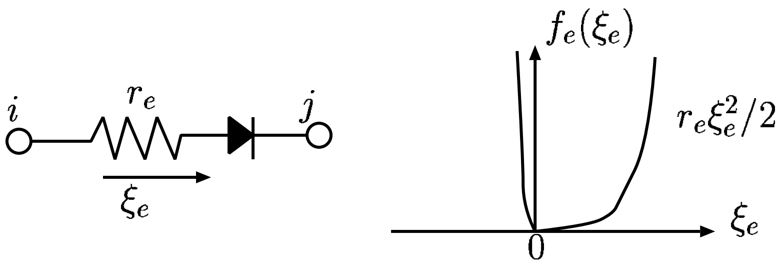

Figure 2: A series connection of a diode and a resistor

and $i_0 \in V$ is an arbitrarily fixed root vertex.

This kernel K is called the *electric network kernel* and its explicit formulas for some classes of graphs are known [3]. This electric network classifier with Ohmic resistors, however, does not make use of the direction of edges, since $f_e(\xi_e) = f_e(-\xi_e)$. To express the link structure of the Web or the citation graph of papers, for example, it is necessary to consider the influence of the edge direction. For this, we introduce unsymmetric electric resistors as follows. Set the current energy f_e to

$$f_e(\xi_e) = \begin{cases} r_e^+\xi_e^2/2 & \text{if } \xi_e \geq 0 \\ r_e^-\xi_e^2/2 & \text{if } \xi_e < 0 \end{cases} \tag{2.19}$$

for each edge $e \in E$, where r_e^+ and r_e^- are electric resistances (> 0) of positive and negative directions, respectively. With this approach, electric network classifiers with general convex functions can incorporate the influence of the edge direction. In particular, taking sufficiently large r_e^-, we can represent a series connection of a diode and a resistor as Figure 2. Furthermore, C-SVM interpretation is also possible.

Theorem 2.5. *Consider the problems* [P] *and* [D] *with f_e as* (2.19) *for each edge $e \in E$ and some convex functions $\{g_j\}_{j\in U}$. Let (ξ^*, u^*) and (η^*, p^*) be optimal solutions to* [P] *and* [D]*, respectively. Consider the modified problems* [P*] *and* [D*] *with $\tilde{f}_e$ defined as*

$$\tilde{f}_e(\xi_e) = \hat{r}_e\xi_e^2/2 \quad \textit{with} \quad \hat{r}_e = \begin{cases} r_e^+ & \textit{if } \xi_e^* \geq 0 \\ r_e^- & \textit{if } \xi_e^* < 0 \end{cases} \tag{2.20}$$

for each edge $e \in E$ and the same $\{g_j\}_{j\in U}$. Then (ξ^*, u^*) and (η^*, p^*) are also optimal to [P*] *and* [D*]*. In particular, if we choose g_j as* (2.13)*, u^* is an optimal solution to the C-SVM problem with some electric network kernel.*

2.3 Proofs

We use some basic notation and properties from convex analysis [9]. First, we note that (ξ^*, u^*) and (η^*, p^*) are optimal to the monotropic programming problems [P] [D] if and only if they are feasible and satisfy

$$f_e(\xi_e^*) + f_e^*(\eta_e^*) = \xi_e^*\eta_e^*, \quad (e \in E), \tag{2.21}$$

$$g_j(u_j^*) + g_j^*(-p_j^*) = -u_j^*p_j^* \quad (j \in U) \tag{2.22}$$

(see [10, Chapter 8] [5, Chapter IV] for optimality conditions for nonlinear network flow problems and also see [10, Chapter 11] for generall monotropic programming). Second, for a convex function $f : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$, $x^* \in \mathbf{R}^n$ is a minimizer of f if and only if it satisfies $0 \in \partial f(x^*)$, where $\partial f(x^*) := \{p \in \mathbf{R}^n \mid f(x) - f(x^*) \geq p^\top (x - x^*)\}$ is called the *subdifferential* of f at x^* (see [9, Section 72] for optimality conditions of convex functions using subdifferential).

Proof of Theorem 2.1. The problem [P] with f_e defined as (2.6) is rewritten as

$$\min_{u \in \mathbf{R}^U} \quad \min_{\xi} \left\{ \xi^\top R \xi / 2 \;\middle|\; A\xi = \begin{pmatrix} 0 \\ u \end{pmatrix} \right\} + \sum_{j \in U} g_j(u_j) \quad \text{s.t.} \quad D \begin{pmatrix} 0 \\ u \end{pmatrix} = 0. \tag{2.23}$$

Hence, using a reflexive minimum-norm generalized inverse A^+, the inner optimizer ξ^* is given as

$$\xi^* = A^+ \begin{pmatrix} 0 \\ u \end{pmatrix}. \tag{2.24}$$

Then, the inner optimal value is given by $1/2 \sum_{i,j \in U} ((A^+)^\top R A^+)_{ij} u_i u_j$. Thus, we obtain the first statement of Theorem 2.1. Next, we show that p^* defined as (2.11) and $\eta^* := A^\top p^*$ are optimal to [D]. Let u^* be an optimal solution of [P$'$] and μ an optimal Lagrange multiplier of Lagrange function of [P$'$]

$$\frac{1}{2} \sum_{i,j \in U} K(i,j) u_i u_j + \sum_{j \in U} g_j(u_j) + \mu^\top D \begin{pmatrix} 0 \\ u \end{pmatrix}. \tag{2.25}$$

Then, the subdifferential of the Lagrange function (2.25) at (u^*, μ) contains zero (see [9, Theorem 28.3]). From this, we have

$$\partial g_i(u_i^*) \ni -\sum_{j \in U} K(j,i) u_j^* - (D^\top \mu)_i = -p_i^* \quad (i \in U). \tag{2.26}$$

Hence, $-p_i^* \in \partial g_i(u_i^*)$ implies (2.22) (see [9, Theorem 23.5]). On the other hand, we have

$$\begin{aligned} \eta^* &= A^\top p^* = A^\top \left\{ (A^+)^\top R A^+ \begin{pmatrix} 0 \\ u^* \end{pmatrix} + D^\top \mu \right\} \\ &= R A^+ A A^+ \begin{pmatrix} 0 \\ u^* \end{pmatrix} = R A^+ \begin{pmatrix} 0 \\ u^* \end{pmatrix} = R \xi^*, \end{aligned}$$

where the third equality follows from $RA^+A = (A^+A)^\top R$ and $\operatorname{Im} A = \operatorname{Ker} D$, and the fourth follows from reflexivity of A^+. From $\eta_e^* = r_e \xi_e^*$, we obtain (2.21). □

Proof of Corollary 2.2. From transformation $\alpha_i = y_i u_i$ for $i \in U$, we obtain the standard C-SVM formulation. □

Proof of Theorem 2.5. It is easy to check that in the modified problem, (ξ^*, u^*) and (η^*, p^*) satisfy the optimality conditions (2.21) and (2.22). □

3 Experimental Results

We use the nonlinear cost network solver code `asspg` by Guerriero and Tseng [2], available at http://www.math.washington.edu/˜tseng/. Since this program uses a primal-dual type algorithm, we can obtain an optimal potential from this program.

We use 20 newsgroups corpus for the performance evaluations. These are available at http://www.cs.umass.edu/˜mccallum/. Each document of the 20 newsgroups is processed into the bag of words representation by Mallet tool kit. We select three binary problems,

(1) rec.auto vs. rec.motorcycles,
(2) soc.religion.christian vs. alt.atheism, and
(3) comp.sys.ibm.pc.hardware vs. comp.sys.mac.hardware.

Graph structures are constructed as follows. We connect each document to its 5-nearest neighbors, where the distance on documents is measured by the cosine similarity. We use this distance as edge weight. Resulting graph sizes are (1) 1995 vertices and 17963 edges (2) 1996 vertices and 19960 edges (3) 1993 vertices and 19930 edges. For electric network classifier, we take f_e as (2.16), r_e as edge weight, and g_j as (2.13). For comparison, we use C-SVM with linear and diffusion kernel which are implemented to LIBSVM package which is available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm/. For the diffusion kernel, we use weighted Laplacian, i.e., $L(i,j) = -1/w(i,j)(i \neq j)$ and $L(i,i) = \sum_j 1/w(i,j)$, where w is the edge weight. Then the diffusion kernel is defined by $(\exp(-\beta L))(i,j)$ for the diffusion parameter $\beta > 0$. The C-SVM parameter C is selected as $C = 5$. The diffusion parameter β is selected as (1) $\beta = 0.2$, (2) $\beta = 0.2$ and (3) $\beta = 0.3$ by preliminary experiments. A half of whole documents are randomly selected as unlabeled test data. The rest are used for training data set consisting of labeled and unlabeled data. Experiments are carried out, by varying the ratio of labeled data. This procedure is repeated for 10 times. Averages of accuracy are reported in Figure 3.

Results show that the performance of our electric network classifier is fairly good, compared with C-SVM with linear and diffusion kernel. In particular, in the range of small ratio of labeled data, our classifier shows good performance. This implies effectiveness of semi-supervised learning. Furthermore we emphasize that learning time of our classifier is very short compared with diffusion kernel, since diagonalization for computing diffusion kernel matrix is quite heavy. Indeed, average learning times of our classifier using `asspg` for data sets (1), (2), and (3) are 0.97 (s), 1.02 (s), and 1.27 (s), respectively. On the other hand, average computational times for the construction of diffusion kernel matrix $\exp(-\beta L)$ through diagonalizations for (1), (2), and (3) are 92.4 (s), 91.4 (s), and 92.5 (s), respectively. This experiment was done by Athron 64 2.2GHz CPU machine with 2GB memory, and matrix diagonalizations for diffusion kernel were done by Matlab. This indicates that our classifier has the scalability for large problems.

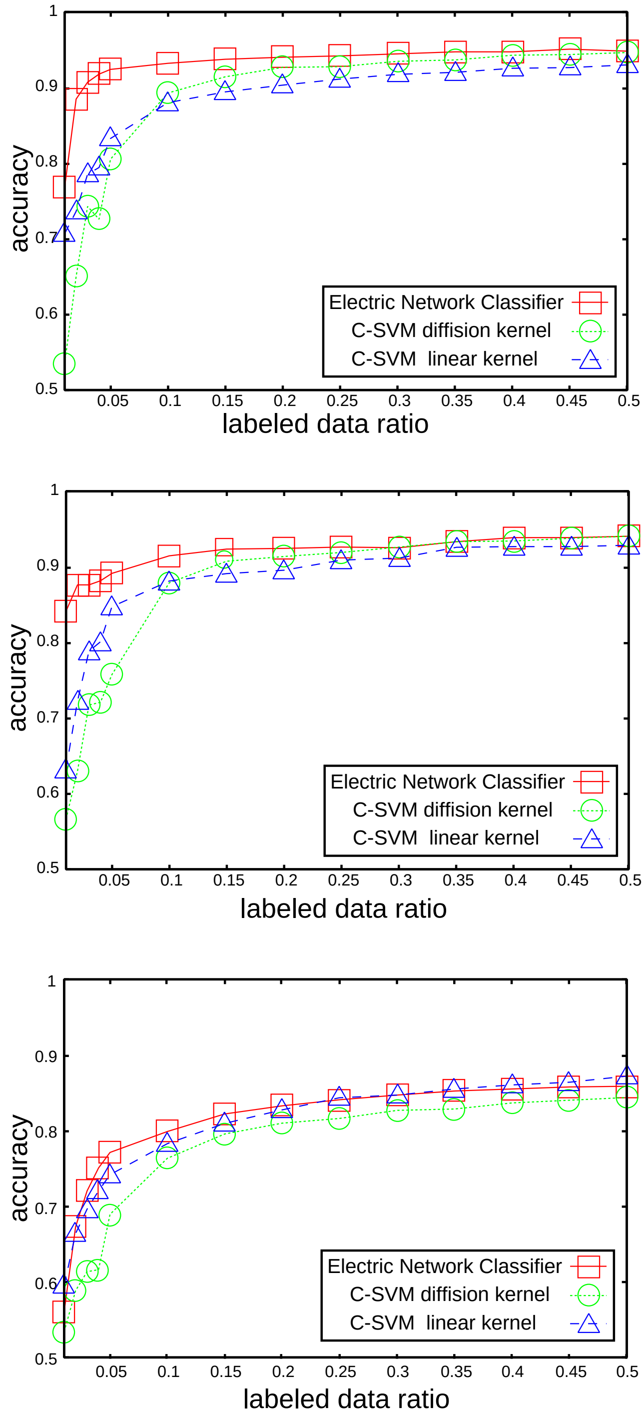

Figure 3: Accuracy for each classifier on three data sets (1) (up), (2) (middle) and (3) (down).

Acknowledgements

This work is supported by the 21st Century COE Program on Information Science and Technology Strategic Core, and by a Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] P. G. Doyle and J. L. Snell: *Random Walks and Electric Networks*, The Mathematical Association of America, 1984.

[2] F. Guerriero and P. Tseng: Implementation and test of auction methods for solving generalized network flow problems with separable convex cost. *Journal of Optimization Theory and Applications* **115** (2002), 113–144.

[3] H. Hirai, K. Murota, and M. Rikitoku: SVM kernel by electric network, *Pacific Journal of Optimization*, to appear.

[4] S. Hochreiter, M. C. Mozer, and K. Obermayer: Coulomb classifiers: generalizing support vector machines via an analogy to electrostatic systems, *Neural Information Processing Systems* **15** (2003), 545–552.

[5] M. Iri: *Network Flow, Transportation and Scheduling*, Academic Press, New York, 1969.

[6] J. Kandola, J. Shawe-Taylor, and N. Cristianini: Learning Semantic Similarity, *Neural Information Processing Systems* **15** (2003), 657–664 .

[7] R. I. Kondor and J. Lafferty: Diffusion kernels on graphs and other discrete structures, in: C. Sammut and A. Hoffmann, eds., *Proceedings of the 19th International Conference on Machine Learning,* Morgan Kaufmann, 2002, 315–322.

[8] C. R. Rao and S. K. Mitra: *Generalized Inverse of Matrices and Its Applications*, Wiley, New York, 1971.

[9] R. T. Rockafellar: *Convex Analysis*, Princeton Univ. Press, Princeton, N.J., 1970.

[10] R. T. Rockafellar: *Network Flows and Monotropic Optimization*, Wiley, New York, 1984.

[11] B. Schölkopf and A. J. Smola: *Learning with Kernels,* MIT Press, 2002.

[12] A. J. Smola and R. Kondor: Kernels and regularization on graphs, in: B. Schölkopf and M. Warmuth, eds., *Proceedings of the Annual Conference on Computational Learning Theory,* Lecture Notes in Computer Science **2777**, Springer, 2003.

[13] K. Tsuda and W.S. Noble: Learning kernels from biological networks by maximizing entropy. *Bioinformatics* **20** (2004), 326–333.

[14] J.-P. Vert and M. Kanehisa: Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA, *Neural Information Processing Systems* **15** (2003), 1425–1432.

[15] X. Zhu, Z. Ghahramani, and J. Lafferty: Semi-supervised learning using Gaussian fields and harmonic functions, *Proceedings of the Twentieth International Conference of Machine Learning*, (2003), 912–919.