# Neighbor Search with Global Geometry: A Minimax Message Passing Algorithm

**Kye-Hyeon Kim** FENRIR@POSTECH.AC.KR
**Seungjin Choi** SEUNGJIN@POSTECH.AC.KR
Department of Computer Science, Pohang University of Science and Technology, San 31 Hyoja-dong, Nam-gu, Pohang 790-784, Korea

## Abstract

Neighbor search is a fundamental task in machine learning, especially in classification and retrieval. Efficient nearest neighbor search methods have been widely studied, with their emphasis on data structures but most of them did not consider the underlying global geometry of a data set. Recent graph-based semi-supervised learning methods capture the global geometry, but suffer from scalability and parameter tuning problems. In this paper we present a (nearest) neighbor search method where the underlying global geometry is incorporated and the parameter tuning is not required. To this end, we introduce *deterministic walks* as a deterministic counterpart of Markov random walks, leading us to use the minimax distance as a global dissimilarity measure. Then we develop a message passing algorithm for efficient minimax distance calculation, which scales linearly in both time and space. Empirical study reveals the useful behavior of the method in image retrieval and semi-supervised learning.

## 1. Introduction

The $k$-nearest neighbor ($k$-NN) algorithm is a simple but widely-used method for classification and retrieval. The task is to find $k$ nearest neighbors (in a database) of a query using (Euclidean) distance measure. In the case of retrieval, such neighbors themselves are the solution. For classification, the class label of the query is assigned as a label corresponding to the majority in $k$ neighbors found. In most cases, Euclidean distance is used, which does not reflect the underlying global geometry (data manifold) of the database.

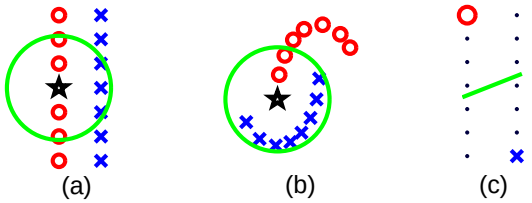


*Figure 1.* Two different classes are distinguished by (red) circles and (blue) crosses. In (a) and (b), black star-shape object is a query. Nearest neighbors are the points within big (green) circles. Some of them are irrelevant data points which should not be selected for successful retrieval. In (c), black dots are unlabeled points. $k$-NN yields an erroneous decision boundary shown as the (green) line, unless unlabeled data points are considered in learning.

Fig. 1 illustrates some drawbacks of the $k$-nearest neighbor algorithm. Nearest neighbors of a query located near the boundary between two different clusters include irrelevant data points that degrade the performance of classification as well as retrieval. Fig. 1 (c) emphasizes the role of unlabeled data points, which is not taken into account in the $k$-NN algorithm. Simple examples in Fig. 1 emphasize the underlying global geometry of a data set as well as the role of unlabeled data points, which should be considered in neighbor search for semi-supervised learning and retrieval.

Various methods for metric learning have been proposed for capturing the underlying global geometry. Domeniconi et al. (2002) proposed locally linear metric learning for $k$-NN, but it cannot utilize unlabeled data points. Xing et al. (2003) proposed Mahalanobis metric learning with pairwise constraints. This method works well with partially labeled data sets, but cannot handle the nonlinear manifold structure.

Appearing in *Proceedings of the 24$^{th}$ International Conference on Machine Learning*, Corvallis, OR, 2007. 

Chang and Yeung (2004) proposed locally linear metric learning with pairwise constraints. While it works with partially labeled data and takes a nonlinear manifold structure into account, its time complexity is very high, due to a quadratic optimization involved in every iteration of the algorithm.

Manifold learning (Tenenbaum et al., 2000; Roweis & Saul, 2000; Belkin & Niyogi, 2003) is a method of finding meaningful low-dimensional structure embedded in their high-dimensional observations. These methods capture the underlying global geometry, but their use is limited in an embedding problem. The manifold regularization (Belkin & Niyogi, 2004) initiated various graph-based semi-supervised learning methods (Zhou et al., 2004; Zhu et al., 2003; Wang & Zhang, 2006). These methods handle partially labeled classification as well as the global geometry. However, their scalability is rather poor, since the computation requires $\mathcal{O}(N^3)$ in time and $O(N^2)$ in space, where $N$ is the number of elements in a database. In addition, they involve kernel parameter tuning for a proper weighted adjacency matrix. The tolerable performance is guaranteed only in a very narrow range of the values of the parameters (Wang & Zhang, 2006).

In this paper, we present an efficient neighbor search method that is useful in semi-supervised learning and retrieval. The proposed method exploits the underlying global geometry of a partially labeled data set, and improves the scalability compared to existing graph-based semi-supervised learning methods. We introduce *deterministic walks* that can be viewed as a deterministic counterpart of Markov random walks. In this framework, we can consider the minimax distance as a global dissimilarity measure, which is easily computed on minimum spanning trees. We then develop a message passing algorithm that scales with $\mathcal{O}(N)$ in time and in space. In addition, our method is free from kernel parameters whose proper values should be determined in most of graph-based semi-supervised learning methods.

## 2. Markov Random Walks and Semi-Supervised Learning

We review graph-based semi-supervised learning (SSL) in the perspective of neighborhood selection. SSL is often illustrated in the framework of Markov random walks (Szummer & Jaakkola, 2002; Zhou & Schölkopf, 2004), the key idea of which is in the transitive closure of closeness relationship: *"My neighbor's neighbor can be also my neighbor."*

Given a data set $\mathcal{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$, the base similarity $w_{ij}$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, is computed as

$$w_{ij} = \exp\left\{-\frac{1}{2\sigma^2}||\boldsymbol{x}_i - \boldsymbol{x}_j||^2\right\}, \tag{1}$$

where $\sigma > 0$ is a kernel width parameter. The transition matrix $\boldsymbol{P}$ is then constructed by normalizing the base similarity, i.e., $[\boldsymbol{P}]_{ij} = p_{ij} = w_{ij}/\sum_k w_{ik}$.

Markov random walks implement the transitive relationship in this way: An element $\boldsymbol{x}_j$ is selected as a neighbor of $\boldsymbol{x}_i$ with the probability $\alpha^t$, if a random walk sequence starting at $\boldsymbol{x}_i$ visits the place $\boldsymbol{x}_j$ at time $t$. The discounting factor $0 < \alpha < 1$ makes $\boldsymbol{x}_j$ having a less opportunity for being selected if the relationship is longer. Then, the selection probability is the sum of the probabilities of selecting at time $t = 1, 2, 3, \cdots$.

It is well known that the $t$th power of $\boldsymbol{P}$ contains $t$-step transition probabilities, where $[\boldsymbol{P}^t]_{ij}$ represents the probability that a random walk starting at $\boldsymbol{x}_i$ visits $\boldsymbol{x}_j$ at the $t$th transition. Thus, the *neighborhood selection probability* is given by[1]

$$\widehat{\boldsymbol{P}} = (1-\alpha)(\boldsymbol{I} - \alpha\boldsymbol{P})^{-1}. \tag{2}$$

Clearly, $[\widehat{\boldsymbol{P}}]_{ij}$ can be viewed as the global similarity measure between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. There are several variants of Eq. (2) in semi-supervised learning (Zhou et al., 2004; Zhu et al., 2003).

One problem of SSL is scalability: Computing Eq. (2) takes $\mathcal{O}(N^3)$ time, and storing $\widehat{\boldsymbol{P}}$ or $\boldsymbol{P}$ takes $\mathcal{O}(N^2)$ space. For classification, one can solve a linear system $(\boldsymbol{I} - \alpha\boldsymbol{P})\boldsymbol{f} = \boldsymbol{y}$ rather than Eq. (2) so that reduce the time cost to $\mathcal{O}(N^2)$ for a sparse $\boldsymbol{P}$ (e.g. $k$ neighborhood graph).[2] However, it is still not practical for large scale problems.

Garcke and Griebel (2005) proposed a grid-based approximation technique which takes $\mathcal{O}(N)$ time for a sparse $\boldsymbol{P}$. However, it can handle only a low dimensional data set since the number of required grid points are exponential with respect to the data dimension.

Delalleau et al. (2005) proposed an approximation by constructing a smaller graph from a subset of a data set. It takes $\mathcal{O}(M^2N)$ time and $\mathcal{O}(M^2)$ space, where $M$ is the size of the subset. While efficient, it can be used only for classification. Zhu and Lafferty (2005) proposed a similar approach that reduces the original graph into a much smaller backbone graph.

[1] Note that $(\boldsymbol{I} - \alpha\boldsymbol{P})^{-1} = \boldsymbol{I} + \alpha\boldsymbol{P} + \alpha^2\boldsymbol{P}^2 + \cdots$. Note also that $(1-\alpha)$ makes the row sum of $\widehat{\boldsymbol{P}}$ be 1.

[2] Recently, Spielman and Teng (2004) proposed an approximation algorithm to solve a sparse linear system in $\mathcal{O}(N \log^c N)$ time for a constant $c$.

## 3. Deterministic Walks

In this section, we introduce a deterministic counterpart of Markov random walks, which is referred to as *deterministic walks* where the global similarity (2) is replaced by the *minimax distance* that can be easily computed on minimum spanning trees. The determinism is illustrated in Sec. 3.1, followed by the minimax distance in Sec. 3.2.

### 3.1. From randomness to determinism

We first slightly modify the transitive closeness relationship mentioned in Sec. 2 as:

*"My neighbor's neighbor* ***is*** *also my neighbor."*

The word 'can be', reflecting randomness, is replaced by the word 'is', meaning determinism. The behavior of a random walk is characterized by a transition matrix $\boldsymbol{P}$. In contrast, a *deterministic walk* is characterized by an adjacency matrix with a given $\epsilon$:

$$[\boldsymbol{A}_\epsilon]_{ij} = \begin{cases} 1 & \text{if } d_{ij} \leq \epsilon, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $d_{ij}$ is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. In deterministic walks, any two elements are neighbors with the probability 1, if there is a path (i.e. transitive relationship) between them for a given $\boldsymbol{A}_\epsilon$.

We give an illustrative example. Let a movement from $\boldsymbol{x}_i$ to $\boldsymbol{x}_j$ be allowed only if $d_{ij} \leq \epsilon$ for a threshold $\epsilon > 0$. Then $\boldsymbol{x}_i$ and $\boldsymbol{x}_k$ are neighbors by transitive relationship, if there are one or more sequences of the allowed movements between them. For instance, consider a sequence $\mathcal{P} = (\boldsymbol{x}^0, \boldsymbol{x}^1, \cdots, \boldsymbol{x}^t)$ where $\boldsymbol{x}^p$ represents a point visited at the $p$th movement in $\mathcal{P}$. If every Euclidean distance $d_{p,p+1}$ associated with a hop between $\boldsymbol{x}^p$ and $\boldsymbol{x}^{p+1}$ is less than $\epsilon$ for $p = 0, \ldots, t-1$, then $\boldsymbol{x}^0$ and $\boldsymbol{x}^t$ are neighbors of each other. Moreover, all the points $\boldsymbol{x}^p$ on such a path are also neighbors by the transitive closeness relationship.

There exist many possible sequences of points between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. For example, $\mathcal{P}_{ij}^1 = (\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\mathcal{P}_{ij}^2 = (\boldsymbol{x}_i, \boldsymbol{x}_1, \boldsymbol{x}_5, \boldsymbol{x}_j)$, $\mathcal{P}_{ij}^3 = (\boldsymbol{x}_i, \boldsymbol{x}_4, \boldsymbol{x}_j)$, and so on. However, most of them may include forbidden movements. Let a sequence be referred to be *valid* if all internal hops on the sequence do not exceed $\epsilon$. More precisely, $\mathcal{P}_{ij}^k$ (where the superscript $k$ is an index to enumerate all possible sequences between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$) is valid if $d_{p,p+1} \leq \epsilon$ for all hops $(\boldsymbol{x}^p, \boldsymbol{x}^{p+1}) \subset \mathcal{P}_{ij}^k$. It means that the sequence $\mathcal{P}_{ij}^k$ can be actually generated by a deterministic walk under the adjacency matrix $\boldsymbol{A}_\epsilon$, so that $\boldsymbol{x}_i$, $\boldsymbol{x}_j$, and all the points in $\mathcal{P}_{ij}^k$ are neighboring points by transitive closeness relationship.

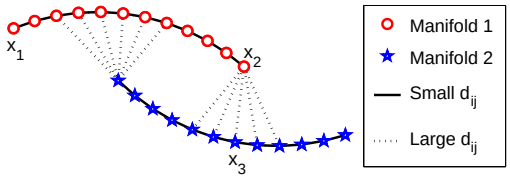


*Figure 2.* Without considering the global geometry, $\boldsymbol{x}_2$ prefer to choose $\boldsymbol{x}_3$ to $\boldsymbol{x}_1$ as its neighbor. In contrast, our deterministic walk select $\boldsymbol{x}_1$ as a neighbor of $\boldsymbol{x}_2$, given a proper value of $\epsilon$.

A proper value of $\epsilon$ prevents a deterministic walk from going out to the other manifolds. In Fig. 2, there is a sequence of the walks from $\boldsymbol{x}_2$ to $\boldsymbol{x}_1$ (i.e. moving in the same manifold) where the distance of each movement is sufficiently small. On the other hand, moving from $\boldsymbol{x}_2$ to $\boldsymbol{x}_3$ (i.e. moving from one manifold to another) requires at least one "great leap". Thus a deterministic walk under $\boldsymbol{A}_\epsilon$, where $\epsilon$ is less than such great leaps, visits only the elements in the same manifold since all possible $\mathcal{P}_{2,3}^k$ are invalid under the value of $\epsilon$.

### 3.2. Minimax distance

Denote by $d_{\mathcal{P}_{ij}}^k$ the maximum hop distance in $\mathcal{P}_{ij}^k$, i.e., $d_{\mathcal{P}_{ij}}^k = \max_{(\boldsymbol{x}^p, \boldsymbol{x}^{p+1}) \subset \mathcal{P}_{ij}^k} d_{p,p+1}$. Then, $\mathcal{P}_{ij}^k$ is valid if $d_{\mathcal{P}_{ij}}^k \leq \epsilon$. Two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are neighbors if there exists at least one valid path between them. In other words, they are neighbors if $\min_k d_{\mathcal{P}_{ij}}^k \leq \epsilon$. We define the *minimax distance* $\epsilon_{ij}$ for a pair $\boldsymbol{x}_i, \boldsymbol{x}_j$ as

$$\epsilon_{ij} = \min_k d_{\mathcal{P}_{ij}}^k = \min_{\mathcal{P}_{ij}^k} \left\{ \max_{(\boldsymbol{x}^p, \boldsymbol{x}^{p+1}) \subset \mathcal{P}_{ij}^k} d_{p,p+1} \right\}, \tag{4}$$

which is the minimum value that guarantees at least one valid path between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Corresponding path is referred to as *minimax path*.

As shown in Fig. 2, the minimax distance $\epsilon_{ij}$ tends to be small if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ lie on the same manifold. Thus, the minimax distance can be viewed as a dissimilarity measure which incorporates the underlying global geometry of a data set.

The computation of $\epsilon_{ij}$ seems to be intractable, since there exists many possible sequences of nodes between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. However, it is well known that a minimum spanning tree (MST) is also a minimax spanning tree (Carroll, 1995). Every path in an MST is also a minimax path, suggesting that $\epsilon_{ij}$ is easily computed in the MST:

$$\epsilon_{ij} = \max_{(\boldsymbol{x}^p, \boldsymbol{x}^{p+1}) \subset \mathcal{P}_{ij}} d_{p,p+1}. \tag{5}$$

## 4. Message Passing

We are given a database $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ and a query $\boldsymbol{x}_*$. The goal is to compute minimax distances $\{\epsilon_{*i}\}$ for $i = 1, \dots, N$. A naive way is to construct the MST of $\mathcal{X} \cup \{\boldsymbol{x}_*\}$ to calculate $\epsilon_{*i}$ by (5). It requires $\mathcal{O}(N^2)$ complexity in time in constructing the MST. Here we present an efficient way to compute $\epsilon_{*i}$, which scales with $\mathcal{O}(N)$ in time.

### 4.1. The minimax message passing algorithm

It is desirable to use the MST of $\mathcal{X}$ (that is already found using the database), instead of constructing a new MST of $\mathcal{X} \cup \{\boldsymbol{x}_*\}$. Note that in a sequence $(\boldsymbol{x}_*, \cdots, \boldsymbol{x}_i)$, paths that should be considered to calculate $\epsilon_{*i}$, belong to one of the following $N$ different paths: $(\boldsymbol{x}_*, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_i)$, $(\boldsymbol{x}_*, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_i)$, $\dots$, $(\boldsymbol{x}_*, \boldsymbol{x}_N, \cdots, \boldsymbol{x}_i)$. Note also that a path $(\boldsymbol{x}_*, \boldsymbol{x}_j, \cdots, \boldsymbol{x}_i)$ includes a subsequence $(\boldsymbol{x}_j, \cdots, \boldsymbol{x}_i)$ where $\epsilon_{ji}$ is easily computed on the MST of $\mathcal{X}$. Therefore, we compute $\epsilon_{*i}$ as

$$\begin{aligned} \epsilon_{*i} &= \min_j \left(\max(d_{*j}, \epsilon_{ji})\right) \\ &= \min\left(d_{*i}, \min_{j \neq i} \left(\max(d_{*j}, \epsilon_{ji})\right)\right), \end{aligned} \quad (6)$$

where $d_{*i}$ is the Euclidean distance between $\boldsymbol{x}_*$ and $\boldsymbol{x}_i$. Note that $\max(d_{*i}, \epsilon_{ii})$ is simply $d_{*i}$.

Computing Eq. (6) takes $\mathcal{O}(N)$ time for each $i$. To be more efficient, we present a message passing algorithm which is similar to the *sum-product algorithm* (Kschischang et al., 2001). Let $T_{\mathcal{X}}$ be the MST of the given database $\mathcal{X}$. Denote by $\mathcal{N}_i$ a set of indices such that $j \in \mathcal{N}_i$ if there is an edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in $T_{\mathcal{X}}$. We also define $\mathcal{N}_i^{\setminus j} = \mathcal{N}_i - \{j\}$. Then $\epsilon_{*i}$ is efficiently computed on $T_{\mathcal{X}}$ by the following *minimax message passing algorithm*:

$$m_{ij} = \max\left(d_{ij}, \min\left(d_{*i}, \min_{k \in \mathcal{N}_i^{\setminus j}} m_{ki}\right)\right), \quad (7)$$

$$\epsilon_{*i} = \min\left(d_{*i}, \min_{k \in \mathcal{N}_i} m_{ki}\right). \quad (8)$$

During the process of message passing, each node is visited two times: At the first visit to node $i$, all messages $m_{ki}$ are given from node $k \in \mathcal{N}_i$ except for one node referred to node $j$. Thus only $m_{ij}$ can be computed at this time. It is called *forward passing*. At the second visit, $m_{ji}$ is also given so that all $m_{ik}$ can be computed. It is called *backward passing*. Also, $\epsilon_{*i}$ can be computed at the second visit.

The algorithm needs the MST and the proper order of message passing in terms of $\mathcal{I}$ and $\mathcal{J}$, where $\mathcal{I}_t$ denote an index of a node to be visited at $t$th iteration and $\mathcal{J}_i$ denote an index of the target node of forward passing at node $i$. Table 1 summarizes the preprocessing. Although it takes $\mathcal{O}(N^2)$ time due to constructing $T_{\mathcal{X}}$, no more computation is needed for queries.

*Table 1.* Minimax message passing: Pre-processing

| |
|---|
| Input: $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ |
| Compute $d_{ij}$ for all $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X}$<br>Construct $T_{\mathcal{X}}$ so that obtain $\{\mathcal{N}_i\}_{i=1}^N$<br>Discard $d_{ij}$, $\forall j \notin \mathcal{N}_i$<br>Find the order of message passing $\mathcal{I}$ and $\mathcal{J}$ |
| Output: $\mathcal{N}, \mathcal{I}, \mathcal{J}$, $\{d_{ij} \mid j \in \mathcal{N}_i\}_{i=1}^N$ |

*Table 2.* Minimax message passing: The main algorithm

| |
|---|
| Input: $\mathcal{N}, \mathcal{I}, \mathcal{J}$, $\{d_{ij} \mid j \in \mathcal{N}_i\}_{i=1}^N$, $\{d_{*i}\}_{i=1}^N$ |
| Initialize $\epsilon_{*i} \leftarrow d_{*i}$, $\forall i$<br>For $t = 1, \cdots, 2N$, repeat:<br>$i \leftarrow \mathcal{I}_t$; $j \leftarrow \mathcal{J}_i$<br>If $i$ wasn't visited before (i.e. *Forward passing*),<br>$\ell_i \leftarrow \arg\min_{k \in \mathcal{N}_i^{\setminus j}} m_{ki}$ $(\mathcal{O}(\lvert\mathcal{N}_i\rvert)$ *time*$)$<br>$m_{ij} \leftarrow \max(d_{ij}, \min(d_{*i}, m_{\ell_i i}))$<br>Otherwise (i.e. *Backward passing*),<br>$\epsilon_{*i} \leftarrow \min(d_{*i}, m_{\ell_i i}, m_{ji})$<br>$m_{ik} \leftarrow \max(d_{ik}, \epsilon_{*i})$, $\forall k \in \mathcal{N}_i^{\setminus j}$ $(\mathcal{O}(\lvert\mathcal{N}_i\rvert)$ *time*$)$<br>If $m_{\ell_i i} < m_{ji}$,<br>Compute $m_{i\ell_i}$ as in Eq. (7) $(\mathcal{O}(\lvert\mathcal{N}_i\rvert)$ *time*$)$ |
| Output: $\{\epsilon_{*i}\}_{i=1}^N$ |

Table 2 summarizes the main algorithm. Note that $\min(m_{\ell_i i}, m_{ji}) = \min_{p \in \mathcal{N}_i^{\setminus k}} m_{pi}$ for all $k \in \mathcal{N}_i^{\setminus j}$ since $m_{ki} \geq m_{\ell_i i}$. Thus $m_{ik} = \max(d_{ik}, \epsilon_{*i})$. Note also that $m_{ji} = \min_{p \in \mathcal{N}_i^{\setminus \ell_i}} m_{pi}$ (thus $m_{i\ell_i} = \max(d_{i\ell_i}, \epsilon_{*i})$) if $m_{\ell_i i} \geq m_{ji}$, since $m_{ki} \geq m_{\ell_i i} \geq m_{ji}$ for all $k$.

For a given query $\boldsymbol{x}_*$ and the Euclidean distances $\{d_{*i}\}_{i=1}^N$, The algorithm computes all minimax distances $\{\epsilon_{*i}\}_{i=1}^N$ in $\mathcal{O}(N)$ time: Each node performs one forward and one backward passing, and each passing takes $\mathcal{O}(\lvert\mathcal{N}_i\rvert)$ time. Since $\sum_{i=1}^N \lvert\mathcal{N}_i\rvert$ is two times the number of edges, i.e., $2(N-1)$, all $N$ forward and $N$ backward passings take $\mathcal{O}(N)$ time.

The algorithm needs $\mathcal{O}(N)$ space to keep $\mathcal{N}$ ($2(N-1)$ space), $\mathcal{I}$ ($2N$ space), $\mathcal{J}$ ($N$ space), and $\{d_{ij} \mid j \in \mathcal{N}_i\}_{i=1}^N$ ($N-1$ space).

Note that the above costs are only for incorporating the minimax distance into $k$-NN. If the database $\{\boldsymbol{x}_i\}_{i=1}^N$ and the computation of $\{d_{*i}\}_{i=1}^N$ are counted, both time and space complexity is precisely $\mathcal{O}(Nd)$, where $d$ is the dimensionality of a data set.

### 4.2. Parameter-free

Most of graph-based semi-supervised learning methods suffer from a problem of tuning kernel parameters such as $\sigma$ in Eq. (1). The reasonable performance is guaranteed only in a very narrow range of $\sigma$. The more severe problem is that learning a proper value of the parameter with a small number of labeled points is intractable. Wang and Zhang (2006) presented this problem with experiments, and proposed a quadratic programming algorithm for computing $w_{ij}$ without kernel parameters. It remedies the problem, but needs more computations.

Our method does not have such a parameter-tuning problem, since the minimax distance is invariant by any non-decreasing mapping. More specifically, in $k$-NN, $\boldsymbol{x}_j$ is preferred to be a neighbor of $\boldsymbol{x}_i$ rather than $\boldsymbol{x}_k$ iff $d_{ij} \leq d_{ik}$, where the inequality is invariant to any nondecreasing mapping, e.g., $f(d_{ij}) \leq f(d_{ik})$ for $f(x) = \exp(x^2/\sigma^2)$ and $\sigma > 0$. Our method is also invariant, because $\epsilon_{ij}$ is obtained from the Euclidean distances by applying only *max* and *min* operations.

Of course, if a parameter influences $d_{ij}$ itself, e.g., $\boldsymbol{\Sigma}$ in $d_{ij} = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Sigma} (\boldsymbol{x}_i - \boldsymbol{x}_j)$, our method and $k$-NN are no longer free from the parameter. Such parameter, however, should be treated as a distance metric.

Note that $\epsilon$ in Eq. (3) is not a parameter. In Sec. 3.2, we assume that the value of $\epsilon$ can be determined differently for each pair of data points, and then derive that its tight bound $\epsilon_{ij}$ can be used as the global dissimilarity. That is, $\epsilon$ is not a parameter chosen by hand, but a solution of deterministic walks.

### 4.3. Augmented measure for retrieval

For retrieval, elements in a database should be ordered with respect to the (dis)similarity to the query. Using our method, however, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ have an equal dissimilarity if $\epsilon_{*i} = \epsilon_{*j}$, although $d_{*i}$ is far greater than $d_{*j}$. To remove such ambiguity, a small portion of the value of the Euclidean distance can be added to the minimax distance:

$$\epsilon^{\lambda}_{*i} = \epsilon_{*i} + \lambda d_{*i}, \tag{9}$$

where $\lambda$ is a small positive constant, e.g., $\lambda = 0.01$. However, the measure can be incorrect if a manifold of a class is highly nonlinear (e.g. S-shape curve) and data points are (almost) uniformly distributed on the manifold.

### 4.4. Multiple trees

If there are outliers between different manifolds, moving from one manifold to another is easier. More specifically, $\epsilon_{ij}$ is smaller for a pair $\boldsymbol{x}_i, \boldsymbol{x}_j$ that belong to the different manifolds each other, so that irrelevant elements can be selected as neighbors.

To make it harder such movement, there are two extreme ways that make it *impossible* to move to the different manifold, by constructing multiple trees where each tree is disconnected with the other trees. One way is constructing the MST from $k$-nearest neighborhood graphs rather than the complete graph. Wang and Zhang (2006) presented that the global geometry can be captured better by neighborhood graphs than the complete graph if a proper value of $k$ is chosen. Another way is constructing the MST with a constraint. Using labeled points in a database, one can construct multiple MSTs as each tree representing each class, by the constraint that any two labeled points that belong to the different classes each other cannot be in the same tree. Clearly, both ways can be used together.

## 5. Numerical Experiments

We compared our minimax algorithm (MM) with $k$-NN classifier for $k = 1$, and *harmonic Gaussian* (HG) that is a variant of graph-based semi-supervised learning (Zhu et al., 2003). The parameter $\sigma$ in Eq. (1) was chosen as 0.0003 for ORL, 0.00015 for Yale-B, and 1.25 for USPS. All the others followed as Zhu et al. (2003) did.

For MM, we divided a data set into a set of queries $\mathcal{Q}$ and a partially labeled database $\mathcal{X}$. Then we constructed multiple MSTs of $\mathcal{X}$ described in Sec. 4.4: (1) The MST is from the 10 nearest neighbor graph; (2) Every MST does not contain two or more labeled points belonging to the different classes. Once the MST of $\mathcal{X}$ is constructed, $\epsilon_{*i}$ is computed for each query. The combined measure was used as in Eq. (9) with $\lambda = 0.01$. For retrieval, $k$ nearest points were retrieved w.r.t. $\epsilon_{*i}$. For classification, the nearest labeled point (i.e. $k = 1$) was retrieved.

### 5.1. Synthetic data

We designed a synthetic data set to illustrate the effect of the global geometry. It contains $N$ unlabeled data points and 20 labeled points generated from four Gaussians, as Fig. 3 illustrates. The data set has nonlinear geometry, so that is intractable for linear metric learning techniques.

We compared MM with $k$-NN. 1000 query points from the Gaussians were used for classification. Table 3 summarizes the result. $N$ denotes the number of unlabeled points. The number of labeled points was 20 for all experiments. MM definitely outperformed $k$-NN.

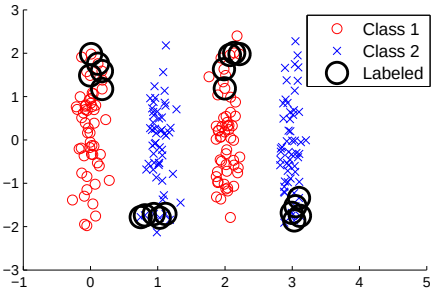


*Figure 3.* Two classes are on four Gaussian manifolds alternately, and 5 labeled points denoted by big black circles are near the boundary of each Gaussian.

We also measured the running time of our method, programmed in C. All experiments were done on a 3.0GHz Pentium 4 CPU with 512KB of cache and 1.5GB of RAM. Table 3 shows that our method actually runs in linear time so that can handle a large scale data set. The regression equation was of the form $\text{Time} = 0.0044219N - 13.2225$. Since the cache hit ratio drops as the spatial locality decreasing for a large $N$, there is little additional gap as $N$ is increased, e.g., 5 times 20000 is 100000 but $5\times 77\text{ms} < 416\text{ms}$.

### 5.2. Image classification and retrieval

We applied our method to three real image datasets, for classification and retrieval:

**Yale-B (10 classes)** : It contains 5760 images of 10 people under 9 poses and 64 illumination conditions. We used all 64 illuminations of the frontal poses of 10 people. Those 640 images were cropped to $161 \times 161$ (i.e. 25921-dimension) where the center of a face is in the center of its cropped image. Each $j$th component of an image $\boldsymbol{x}_i = [x_{i1}, \cdots, x_{i,25921}]$ was normalized as $(x_{ij} - \frac{1}{N}\sum_{k=1}^{N} x_{kj})/256/25921$.

**ORL (40 classes)** : It contains $112\times 92$ face images of 40 people (10 images/person). We used all 400 images and divided each pixel by 2637824 ($= 112 \times 92 \times 256$). Using ORL, we show that our method can handle a data set having many classes.

**USPS (4 classes)** : It contains $16 \times 16$ handwritten images of 10 digits. We used 3874 images of digits '1' (1269 images), '2' (929), '3' (824), and '4' (852). Since images of digit '1' obtain too small Euclidean distances compared with that of the others, we normalized the distance as $d_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / \sum_{k=1}^{N} \|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2$ for MM. It is similar to normalizing the base similarity as $w_{ij}/\sum_{k=1}^{N} w_{ik}$ in the graph-based semi-supervised learning methods. Using USPS, we show that our method can effectively handle a dataset containing many unlabeled points but a few labeled ones.

*Table 3.* The computation time for our minimax algorithm (MM) and classification accuracies for MM and $k$-NN on the synthetic data set.

| $N$ | Time (MM) | Acc (MM) | Acc ($k$-NN) |
|---|---|---|---|
| 10000 | 35ms/query | 100% | 59.1% |
| 20000 | 77ms/query | 100% | 55.0% |
| 40000 | 165ms/query | 100% | 57.5% |
| 100000 | 416ms/query | 100% | 58.8% |
| 200000 | 877ms/query | 100% | 56.2% |

All results were averaged over 100 independent trials. For each trial, labeled points were randomly chosen but there is at least one labeled point for each class. Two sets $\mathcal{X}$ and $\mathcal{Q}$ were also randomly divided for MM.

Fig. 4 shows the classification results. Our method outperformed $k$-NN, and was comparable to HG. Fig. 5 shows the retrieval results. We evaluated the performance in terms of the *precision*, i.e., [The number of correctly retrieved points for the query]/$k$, where $k$ is the number of whole retrieved points. MM can effectively utilize a few labeled points in $\mathcal{X}$.

The right panel of Fig. 4 shows an interesting result that the accuracy on test data hardly depends on the number of training points. For graph-based semi-supervised learning, it is well known that there are heuristics for predicting out-of-sample data in linear time, such as k-NN majority voting or linear combination of training data points, but the accuracy on test data decreases remarkably when the number of training points isn't large enough (see the experiments in (Wang & Zhang, 2006)) since the solutions are approximations, not equal to the exact transductive solution. On the other hand, our method obtains an exact solution in terms of the minimax distance measure, in linear time.

## 6. Related Work

Minimax distance was already studied (Carroll, 1995; Gower & Ross, 1969), while it was mainly used for clustering. The novelty of our work includes: (1) the minimax message passing algorithm which scales linearly in both time and space, in order to compute the minimax distance between a query (out-of-sample) and a data point in database (in-sample); (2) investigation of minimax distance in semi-supervised learning and data retrieval; (3) an alternative view of minimax distance where the deterministic counterpart of Markov random walks was exploited.

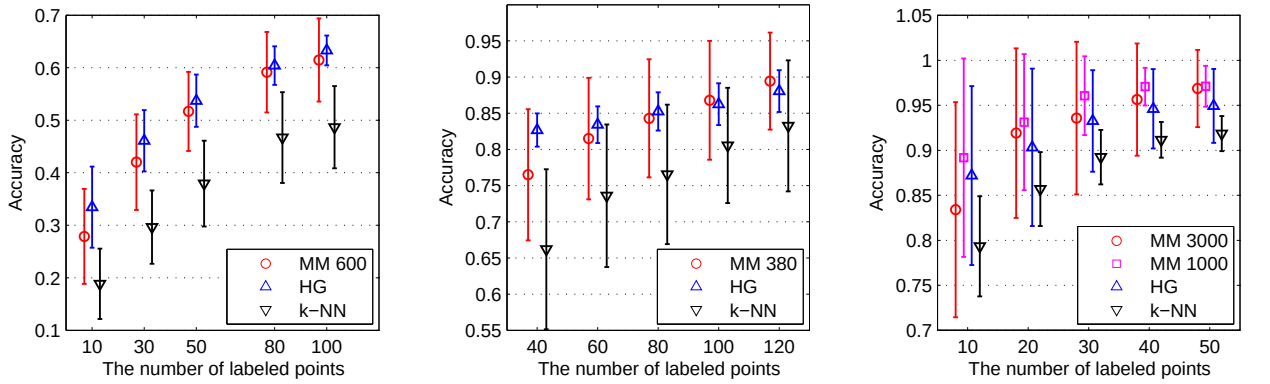


*Figure 4.* Classification accuracies on partially labeled data sets. The vertical lines indicate the standard deviation and the numbers in the legend denote the size of the database for MM. (Left) Yale-B, (Middle) ORL, (Right) USPS dataset.

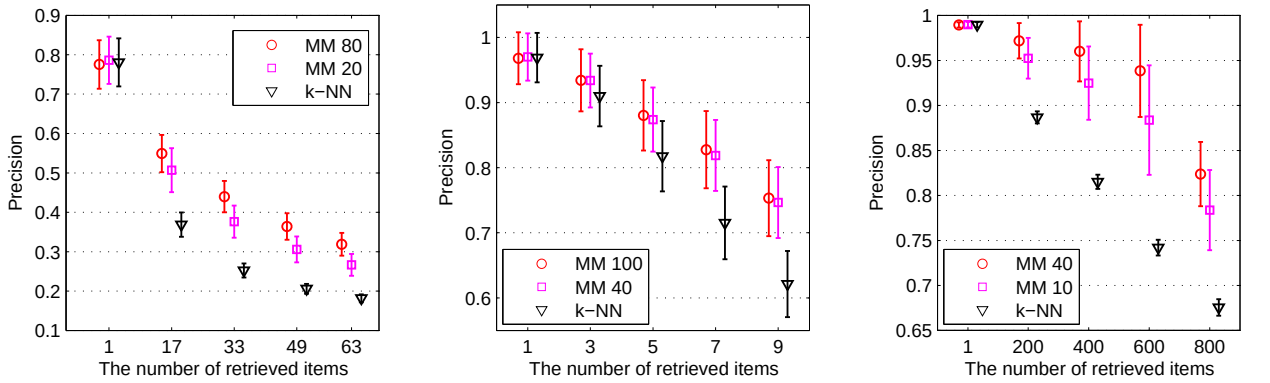


*Figure 5.* Precisions on retrieval from partially labeled databases. The numbers in the legend denote the number of labeled images in the database. (Left) Yale-B dataset. For each trial, we randomly divided whole images into 600 images for a database and 40 images for queries. (Middle) ORL dataset. 380 images for a database and 20 images for queries. (Right) USPS dataset. 3000 images for a database and 874 images for queries.

There exist fast NN search methods (see (Shakhnarovich et al., 2006) and references therein), with their focus on efficient data structures. However, the classification or retrieval performance of those methods was not improved, compared to the standard $k$-NN, since their dissimilarity measures are (approximated) Euclidean distances which do not reflect the underlying global geometry of data. Thus our scalability issue should be discussed only within methods that take the global geometry into account.

## 7. Conclusions

We have presented a new neighbor search method where the underlying global geometry of a data set was exploited through minimax paths. We have introduced deterministic walks as a deterministic counterpart of Markov random walks, leading us to consider the minimax distance as a global dissimilarity measure. For efficient calculation of minimax distances between a query and data points in the database, we have developed the minimax message passing algorithm which scales with $\mathcal{O}(N)$ complexity in time and in space. The method was applied to the problems of semi-supervised classification and retrieval, showing its useful behavior as an alternative to existing graph-based semi-supervised learning methods and $k$-NN.

However, our method may be more vulnerable to outliers, compared to random walks-based methods. For example, the Euclidean commute-time distance (ECTD) between nodes on a graph takes all existing paths between nodes into account such that two nodes are considered similar if there are many short paths between them (Fouss et al., 2005). Thus, ECTD was known to be more robust to outliers, compared to the geodesic distance (shortest path). More work will be done in the aspect of robustness, since the minimax distance used in our method solely depends on a single path (minimax path) just like the geodesic distance.

## Acknowledgments

This work was supported by Korea MCIE under Brain Neuroinformatics Program and KOSEF Basic Research Program (grant R01-2006-000-11142-0).

## References

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*, 1373–1396.

Belkin, M., & Niyogi, P. (2004). Semi-supervised learning on Riemannian manifolds. *Machine Learning*, *56*, 209–239.

Carroll, J. D. (1995). Minimax length links of a dissimilarity matrix and minimum spanning trees. *Psychometrika*, *60*, 371–374.

Chang, H., & Yeung, D. Y. (2004). Locally linear metric adaptation for semi-supervised clustering. *Proceedings of International Conference on Machine Learning* (pp. 153–160).

Delalleau, O., Bengio, Y., & Roux, N. L. (2005). Efficient non-parametric function induction in semi-supervised learning. *Proceedings of International Workshop on Artificial Intelligence and Statistics* (pp. 96–103).

Domeniconi, C., Peng, J., & Gunopulos, D. (2002). Locally adaptive metric nearest-neighbor classificaton. *IEEE Trans. Pattern Analysis and Machine Intelligence*, *24*, 1281–1285.

Fouss, F., Pirotte, A., & Saerens, M. (2005). A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation. *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 550–556).

Garcke, J., & Griebel, M. (2005). Semi-supervised learning with sparse grids. *Proceedings of ICML, Workshop on Learning with Partially Classified Training Data* (pp. 19–28).

Gower, J. C., & Ross, G. J. S. (1969). Minimum spanning tree and single-linkage cluster analysis. *Applied Statistics*, *18*, 54–64.

Kschischang, F. R., Frey, B. J., & Loeliger, H. A. (2001). Factor graphs and the sum-product algorithm. *IEEE Trans. Information Theory*, *47*, 498–519.

Roweis, S. T., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Shakhnarovich, G., Darrell, T., & Indyk, P. (2006). *Nearest-neighbor methods in learning and vision: Theory and practice*. MIT Press.

Spielman, D. A., & Teng, S. H. (2004). Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. *Proceedings of Annual ACM Symposium on Theory of Computing* (pp. 81–90).

Szummer, M., & Jaakkola, T. (2002). Partially labeled classification with Markov random walks. *Advances in Neural Information Processing Systems*. MIT Press.

Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*, 2319–2323.

Wang, F., & Zhang, C. (2006). Label propagation through linear neighborhoods. *Proceedings of International Conference on Machine Learning*. Pittsburgh, PA.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russel, S. (2003). Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems*. MIT Press.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Schölkopf, B. (2004). Learning with local and global consistency. *Advances in Neural Information Processing Systems* (pp. 321–328). MIT Press.

Zhou, D., & Schölkopf, B. (2004). Learning from labeled and unlabeled data using random walks. *Proceedings of the 26th DAGM Pattern Recognition Symposium* (pp. 237–244).

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. *Proceedings of International Conference on Machine Learning* (pp. 912–919).

Zhu, X., & Lafferty, J. (2005). Harmonic mixtures: Combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. *Proceedings of International Conference on Machine Learning* (pp. 1052–1059).