International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 1 (2018) pp. 598-605

Feature Transformation by Feedforward Neural Network for Semisupervised Classification

Gouchol Pok

Division of Computer and Information Technology Education,
Pai Chai University, Daejeon, Republic of Korea.
ORCID: 0000-0002-0499-1863.

Abstract
Semi-supervised learning is based on the cluster and manifold assumption. However, it is difficult to verify if the assumptions hold with a limited number of data samples.We present a homothetic transformation-based method that maps the input data to a more compact form in such a way that the two assumptions are strengthened. Using the benchmark data, we have empirically shown this transform guarantees high performance of the semi-supervised learning.

Keywords: Semi-supervised Learning, Cluster Validation, Homothetic Transformation, Feedforward Neural Networks, Sammon Mapping

INTRODUCTION

With large volume of unlabeled data available, semi-supervised learning (SSL) becomes an attractive approach for improving generalization and classification performance of supervised learners [1, 2]. In addition to labeled data, SSL utilizes unlabeled data which requires less effort and gives higher performance gain. SSL is based on the cluster and manifold assumption. Cluster assumption states that data belonging to the same class are located closely so that cluster boundary passes through low density region [3, 4]. Manifold assumption is based on the fact that data lie on a low dimensional manifold of the input space [5, 6]. Some authors argued that the manifold assumption alone is not sufficient to reduce the error bound of supervised learning [7, 8], and the manifold-based semi-supervised approaches performs better than supervised approaches only for certain learning problems with relatively few unlabeled samples [6]. On the other hand, cluster assumption plays a key role in successful semi-supervised learning, and many inductive learning algorithms are developed on the basis of cluster assumption [9, 10]. However, it is difficult to verify if the cluster assumption holds with a limited number of samples [7].

In this study, we present a homothetic transform-based SSL method which transforms the input data space to a low-dimensional space in which the input data are represented as more compact distribution than in the original input space. As a consequence of the transformation, the input data can be mapped to the one in which the compact cluster property is secured. Even with the data which already satisfies one or both of the two assumptions, the homothetic transformation strengthens the assumptions, and hence subsequent semi-supervised learning performs better.

As shown in Figure 1, homothetic transformation relates two similar geometric shapes in such a way that it preserves the angles and orientations of the shapes with respect to a homothetic center.

In the figure, suppose that the distribution range of a certain class data is represented as figure P.

Then, with an appropriately computed homothetic center O, a large area of P is mapped to a smaller area of Q. When the transformation is applied to each class by taking its centroid of the points belonging to the class as a homothetic center, the data samples with high similarity tend to form high density regions while decision hyperplane tend to pass low density regions between different classes.

We used the Multilayer Feedforward Network (MLFN) to implement the homothetic transformation.

MLFN has been widely used for many applications and demonstrated its strength in classification and regression [11]. In spite of its strength in machine learning applications, MLFN has rarely been used for the semi-supervised learning except a number of studies [12-14]. One of the most interesting aspects of MLFN related with this study is that the neurons in the hidden layer encode the information of the input data in such a way that the hyperplanes formed by the connection weights determine decision boundaries between different classes. Specifically, we first use the Sammon mapping to compute an optimal homothetic center for each class, and train an MLFN with the targets using the centers. Once train is done, the hidden neurons take the values inside of Q for the corresponding input values of P in Figure 1. Using the cluster validating method, we will empirically show that this kind of transformation is realized by an MLFN. Other advantageous point of using MLFN is that we can also achieve dimension-reduction of high-dimensional input data.

This article is organized as follows: We present the background knowledge and methodologies in Section 2. The results from the experiments relating cluster validating measures and the semi-supervised algorithms will be presented in Section 3. Then conclusion follows.

METHODS

Homothetic transformation

Homothetic transformation (HT) is an affine space transformation [15], $f: P \rightarrow O + \lambda \cdot \overline{PO}$, where P is a point that is projected to a homothetic center O along a line segment $\overline{PO}$ connecting P and O with a nonzero scale factor λ. Given a

value of λ, HT maps a point P to another point $Q = O + \lambda \cdot \overline{PO}$ in such a way that the line segment $\overline{QO}$ is on the same line as $\overline{PO}$ as shown in Fig. 1.

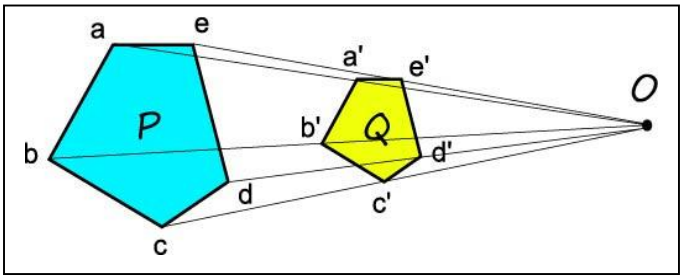

Figure 1. Homothetic transformation preserves the angles and orientations of two similar geometric shapes with respect to a homothetic center O.

In this paper, we just borrow the fundamental concept of homothetic transformation, instead of realizing its rigorous mathematical definition, and call our implementation *homothetic-like transformation* (HLT). Most notable difference of HLT from the conventional HT is that point P, Q, and O can be of different dimension for HLT, and hence the property of being on the same line in Fig. 1 does not necessarily hold anymore.

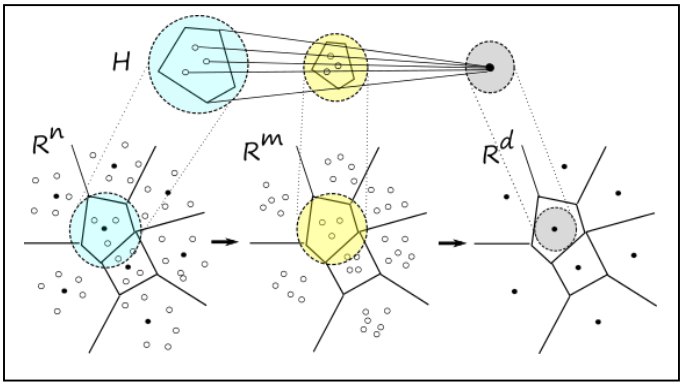

Figure 2. The proposed HLT is compared with with a homothetic transform H.

The processing of information by HLT can be more easily illustrated by Fig. 2. Suppose we deal with the input data in R^n, which are depicted as white circles in the figure. The Voronoi diagram on the left shows clusters for the data of different classes. In each cluster, the black circle represents the centroid of the data (points represented as while circles) in the cluster. The centroid plays as a role of homothetic center for the HLT. The key idea of this scheme is that the input data in a cluster are mapped to their centroid by an HLT, and a more compact form of the cluster is obtained, which is depicted in the middle part of the figure. We implemented the HLT using the information encoding of the MLFN, the details of which are described in the next section.

Homothetic transformation by a neural network

Multilayer Feedforward Neural Networks (MLFN) has been extensively used as a classifier or a predictor for a wide range of applications. An MLFN with a number of hidden layers can be trained as a universal approximator that is capable of approximating any measurable function to any desirable degree of accuracy [16]. MLFN has a typical structure consisting of input layer, one or more hidden layers, and output layer, as depicted on the left of Fig. 3.

The activation level y_j of each neuron in the hidden layer, before applying a squashing function, is computed as the sum of the inputs x_i multiplied by weightsw_{ji} connecting x_i and y_j,

$$y_j = \sum_{i=1}^{n} w_{ji}\, x_i + \theta_j \quad (1)$$

Eq. (2) is of a linear form so it determines a hyperplane in the input space. Actually, a neuron in the first hidden layer computes a linear separation of the input space into two regions along a certain orientation determined by weight coefficients w_{ji} [17]. Combining these hyperplanes appropriately, an ANN can compute the cluster boundaries in Fig. 2.

To determine the boundary of a cluster using straight lines, one needs at least three lines as shown in Fig. 4. Suppose three lines l_1, l_2, and l_3 in the figure are determined by three hidden layer neurons y_1, y_2, and y_3, respectively. These lines form a cluster boundary of the center region separating from other regions. For the clarity of separations, regions are represented as three bits associated with three lines, in which bit of 1 represents the positive region and bit of 0 for the negative region according to the following decision,

$$y_j = \sum_{i=1}^{n} w_{ji}\, x_i > 0 \quad (2)$$

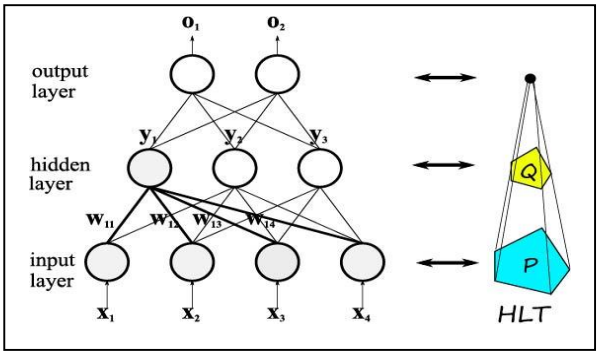

Figure 3. Three layers of MLFN correspond to three components of HLT.

The boundaries of other clusters are also determined in the same way. Recall that the entire view for the association of an ANN with an HLT is illustrated in Fig. 3. Although three lines are sufficient to determine a cluster boundary, many hidden neurons may be involved in determining a cluster boundary depending on a number of pertinent parameters according to the choice of training settings.

Referring to Figure 2 and 3 together, one can see that the dimensions of the spaces, that is *n*, *m*, and *d* are determined by the number of neurons in the input, hidden, and output layers, respectively. The dimension of the input space, *n*, is fixed because it depends on the dimension of the input vectors. The number of hidden neurons, *m*, is a user-defined parameter that should match the information complexity inherent in the input data. Although there have been many theoretical studies on estimating the number of hidden neurons [18-20], it is sufficient that the number is empirically determined with a guide of proven heuristics. More interesting part is that *m* is much less than *n*, so that we can take advantage of the dimension reduction effect, especially when the inputs are of high dimension which is common for biological and text data. Lastly, concerning the hidden neurons, if the second hidden layer exist, the neurons on the second hidden layer combine the activations of the first hidden layer neurons to produce curved regions. Therefore, more complex form of clusters can be easily processed by the ANN with two or more hidden layers.

Now we consider how to determine the number of neurons in the output layer, which corresponds to the dimension *d* of the homothetic center. There are two choices for the value of *d*. Firstly, by focusing on the process of HLT, the centroid of the input data in a cluster can be taken as the homothetic center which is shown as a black circle on the left of Fig. 2. In this case, the dimension of the input is equal to that of the output, or $n = d$. This choice is not reasonable, however, for high dimensional data due to several reasons such as potential performance degradation and unnecessarily high cost in training an MLFN. Secondly, by taking the conventional classifier model of ANN as shown in Figure 3, *d* can be set to the number of different classes or clusters. In this case, *d* will take a value of a very small number regardless of the dimension of the input data.

Suppose we are dealing with data of three different classes. Then, *d* is set to three and output neuron patterns 100_2, 010_2, and 001_2 can discriminate the classes. This target-assigning method is the most common setting for classification tasks using ANNs. One serious drawback of applying this method to HLT implementation is that the bit pattern scheme above does not reflect the distances between different classes. On the left of Fig. 2, the distances between any pair of black circles are different while their bit-pattern targets are all the same because they long to the same class. In order to address the problems incurred by the two approaches above, we apply the Sammon mapping to the cluster centers (black circles in Fig. 2), details of which will be discussed below.

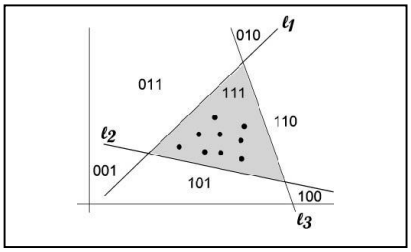

Figure 4. Three decision lines form a triangular class boundary.

Sammon mapping

Sammon mapping is a useful method that projects the high-dimensional input space to a lower dimensional output space where inter-point proximity in the input space is preserved in the output space [21]. The proximity-preserving property of the Sammon mapping is realized by penalizing differences of distances between points in the original space and the projected points. Let the distance between two points x_i and x_j in the input space be represented by d_{ij}, and the distance between two points x_i^* and x_j^* in the output space by d_{ij}^*. Then Sammon mapping tries to minimize the following Sammon error,

$$E = \sum_{i<j}(1/d_{ij}^*) \cdot \sum_{i<j}(d_{ij} - d_{ij}^*)^2/d_{ij}^* \quad (3)$$

E is insensitive to scaling because the relative differences are considered in the computation of the distances. In the original work [21], the minimum of *E* was calculated by gradient descent and later an iterative method was developed [22]. In this paper, the dimension of the input space is set to *n* and the dimension of the projected space, *d*, is the number of different classes. Now that we have shown how to implement an HLT using an MLFN, we will present an algorithm to improve the semi-supervised learning in the next section.

Algorithm for improved semi-supervised learning

Suppose there are *N* training samples consisting of *s* labeled samples $D_l = \{x_1^l, x_2^l, \dots, x_s^l\}$ and *t* unlabeled samples $D_u = \{x_1^u, x_2^u, \dots, x_t^u\}$. For the simplicity of representation, we may denote labeled and unlabeled samples as x^l and x^u, respectively, when the context clearly does not incur any confusion.

Let the label corresponding to a labeled sample x_i^l be y_i, so that the set of labels is $\boldsymbol{y} = (y_1, y_2, \dots, y_s)$ with *d* distinctive labels associated with *d* different classes. SSL aims to improve the classification task for the labeled examples additionally using the unlabeled data. SSL is involved with two computations: to determine the true labels for all the unlabeled samples of D_u (called *transductive learning*), or to estimate the label of a test sample unseen during training (called *inductive learning*).

Our method begins with developing an MLFN $M(\cdot)$ which has *n* input layer neurons, *m* hidden layer neurons, and *d* output layer neurons. Training of the MLFN is performed by using all the labeled examples in D_l. First, we compute the centroid of labeled examples for each *i* of *d* distinct labels

$$c_i = 1/n_i \,(\textstyle\sum_j^{n_i} x_{j1}\,, \sum_j^{n_i} x_{j2}\,, \dots, \sum_j^{n_i} x_{jn}), \quad i = 1, \dots, d, \quad (4)$$

where n_i denotes the number of labeled examples with label *i*, and the labeled examples are denoted by $\boldsymbol{x}_j = (x_{j1}, x_{j2}, \dots, x_{jn})$, $j = 1, \dots, n_i$. Let the set of centroid be $C = \{c_1, c_2, \dots, c_d\}$. Obviously, the centroids are of the same dimension as that of the inputs, that is $||c_d|| = n$, for $i=1,\dots,d$. We perform the

Sammon mapping $\boldsymbol{S}$ on the centroids C to get the dimension-reduced vectors,

$$\boldsymbol{S}: c_i \rightarrow c_i', \quad i = 1, \ldots, d. \tag{5}$$

Following the bit pattern-based class assigning scheme, it is sufficient for the reduced dimension to be equal to the number of distinct labels, that is $||c'_i|| = d$, for $i=1,\ldots,d$.

Let the dimension-reduced centers be $C' = \{c'_1, c'_2, \ldots, c'_d\}$. Now that we have trained the MLFN using the labeled examples, we proceed to determine the labels of the unlabeled examples.

In order to increase the confidence of estimated labels, we first compute two pseudo-labels and check if they are equal to each other. If they have the same label, then assign the label to the unlabeled example.

For each unlabeled example $\boldsymbol{x}_u$, we compute the nearest center and determine its pseudo-label,

$$cls_1 = f(argmin_i||x^u - c_i||) \tag{6}$$

where $f(i)$ gives the label y_i of x_i^l.

We feed the same unlabeled example $\boldsymbol{x}_u$ into the MLFN, and obtain the output $\boldsymbol{o} = M^o(\boldsymbol{x}_u)$. Then, search for the class whose (dimension-reduced) center is nearest to $\boldsymbol{o}$, and denote its pseudo-label as cls_1,

$$cls_2 = argmin_i||\mathrm{o} - c_i'|| \tag{7}$$

If two pseudo-labels cls_1 and cls_2 are equal to each other, then the unlabeled example $\boldsymbol{x}_u$ is given the label and added to the corresponding cluster. If they are different, we go one step further to compute \$k\$-nearest neighbors of $\boldsymbol{x}_u$ over the labeled examples, and determine the pseudo-label cls_1 as the majority label. Then, we again compare cls_1 and cls_2. If they are still different, then the labeling decision is postponed to the next round. After this labeling process is performed for all of the unlabeled examples, new examples sets D_l and D_u are obtained. Now we train incrementally the MLFN with these new D_l, and perform the label assigning for D_u again. This process repeats until there remains no more unlabeled example or other application-specific criteria are satisfied, for example there are only a few of the unlabeled samples are remained. The undecided examples may be discarded when the newly added examples are enough to characterize cluster partition of the problem space, or other ensemble method may be employed to refine the decision task.

EXPERIMENTS

In this section, we will evaluate the performance gain of the proposed HLT by conducting experiments using the benchmark data sets. We evaluated the performance of the HLT by conducting two different kinds of experiments. In the first experiment, we show that the HLT data (from now on, this refers to the data that is obtained by applying an HLT to the raw data) form a compact structure that is more appropriate for the semi-supervised learning compared to the raw data. To the goal, we will compute a number of measures, called *cluster validation index* (CVI), both for the raw data and the HLT data. Better values of CVIs indicate that the cluster assumption (and possibly manifold assumption also) are achieved and enhanced to more satisfactory level. In the second one, we run three semi-supervised learning algorithms both for the raw data and the HLT data, and then compare the performance of semi-supervised learning results for both data.

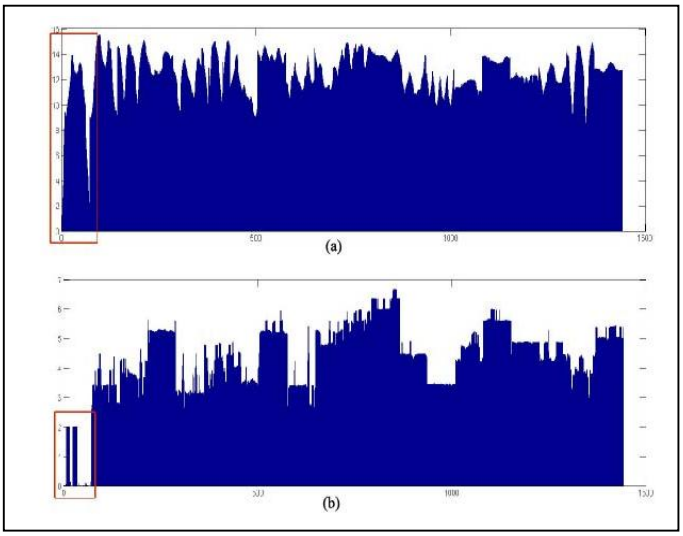

Figure 5. Distribution of distances between one sample and the remaining samples of Coil20 data set. The horizontal axis refers to the 1440 samples in the order of class labels so that first 72 samples belong to class 1, and then 72 samples to class 2, and so on up to class 20. The vertical axis refers to the distance between the vector selected from class 1 and the remaining 1439 vectors. The distances between the sample selected from class 1 and the remaining samples in the same class 1 are denoted in the red box, (a) for the raw data, and (b) for the HLT data.

The data sets used in the experiments are Coil20, Uspst, Text1 benchmark data [23]. Coil20 consists of 1440 samples of 1024 features associated with the images of 20 objects with varying angles. Uspst consists of 2007 samples of 256 features for 10 hand-written digit images. Text1 has 1946 samples of 7511 features taken from 20-newsgroup text data for binary classification. In addition, we also included two artificial data sets: G50c and G10n. G50c consists of 550 samples of 50-dimensional 2-class vectors in which 50 samples are labeled. The labels correspond to the Gaussians, and the means are located such that the Bayes error is 5% [9]. G10n also consists of 550 samples of 10-D 2-class vectors in which 50 samples are labeled. Unlike G50c this data set poses a deterministic problem where the decision function passes the centers of the Gaussians, and depends on only two of the input dimensions. It is interesting to note that for G10n, the cluster assumption does not hold [9], and see how HLT handles such kind of data.

Cluster validation

Cluster validation evaluates the goodness of clustering results. The cluster assumption expresses a condition that the decision boundaries between different classes pass through low density regions [10]. Obviously, stronger cluster assumption leads to better cluster validation, and vice versa. As for the manifold assumption, we also expect that the hidden-layer neurons of MLFN successfully captures the low-dimensional structure embedded in the high-dimensional data as remarked in previous section. Validation measures are divided into two groups, external and internal CVIs, depending on whether or not external class label information is used for cluster validation. Internal CVI only relies on information in the data, while external CVI uses external information such as class label.

Table 1. Cluster validation indices for the Coil20 data. For each m : n division of data, bold-faced numbers denote better values of the CVIs. The "raw" columns refer to the raw data U and HLT refer to the HLT data.

Data / CVI	1:5 division		2:4 division		3:3 division	
	raw	HLT	raw	HLT	raw	HLT
Rand	**0.953**	0.947	0.954	**0.963**	0.953	**0.971**
Jaccard	**0.389**	0.342	0.396	**0.477**	0.395	**0.551**
FM	**0.562**	0.509	0.569	**0.645**	0.568	**0.710**
Silhouette	0.228	**0.347**	0.228	**0.425**	0.229	**0.463**
DB*	1.546	**1.127**	1.518	**1.010**	1.474	**0.908**
Dunn	0.724	**0.932**	0.723	**0.935**	0.736	**1.058**
Homogeneity*	0.354	**0.295**	0.353	**0.254**	0.355	**0.239**
Separation	**0.637**	0.606	**0.638**	0.616	**0.639**	0.639

Data / CVI	4:2 division		5:1 division		Entire set	
	raw	HLT	raw	HLT	raw	HLT
Rand	0.951	**0.974**	0.951	**0.974**	0.952	**0.978**
Jaccard	0.382	**0.601**	0.375	**0.589**	0.385	**0.665**
FM	0.555	**0.743**	0.547	**0.740**	0.558	**0.802**
Silhouette	0.226	**0.500**	0.233	**0.513**	0.226	**0.593**
DB*	1.455	**0.834**	1.427	**0.784**	1.571	**0.696**
Dunn	0.722	**1.173**	0.706	**1.170**	0.718	**1.264**
Homogeneity*	0.360	**0.223**	0.358	**0.216**	0.347	**0.187**
Separation	**0.639**	0.635	**0.643**	0.642	**0.637**	0.626

*For the DB and Homogeneity index, smaller is better.

Table 2. p-values resulting from the t-test on the data of Table 1. Bold-faced numbers denote there are 95%-confident differences between the averaged CVIs of the two groups.

Data / CVI	1:5 division	2:4 division	3:3 division	4:2 division	5:1 division
Rand	0.0988	**0.0063**	**0.0001**	**0.0038**	**0.0001**
Jaccard	0.0536	**0.0122**	**0.0001**	**0.0107**	**0.0001**
FM	**0.0491**	**0.0124**	**0.0001**	**0.0055**	**0.0001**
Silhouette	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
DB	**0.0001**	**0.0001**	**0.0001**	**0.0001**	**0.0001**
Dunn	**0.0130**	**0.0005**	**0.0003**	**0.0001**	**0.0001**
Homogeneity	**0.0001**	**0.0001**	**0.0001**	**0.0009**	**0.0001**
Separation	**0.0265**	0.1113	0.9742	0.8130	0.9767

Because we take into account the situation that there are no class labels available as in the SSL, and hence we consider internal validation measures as well as external measures. We compute three external CVIs, namely Rand, Jaccard, and Folkes-Mallows(FM) index. Four internal CVIs include Silhouette, Davies-Bouldin(DB), Dunn, Homogeneity and Separation index.

In order to simulate the unsupervised learning, we perform the experiments as follows. We divide the input data into two groups: *m* samples used as the labeled data and *n* samples used as the unlabeled data. By varying the ratio *m:n* from *1:k* to *k:1* for a fixed *k*, we can simulate the progression steps aforementioned. In all the experiments below, we set *k* to 6.

By comparing the CVI for the raw data clusters and that for the HLT data clusters, we can check which data satisfies more strongly the cluster assumption.

Table 1 shows the CVIs for the Coil20 data set. Each value in the table shows the average CVI obtained from five runs of the proposed method. Because the Coil20 data set contains 1440 samples, 1:5 division means that the set of labeled data has 240 (=1440/6) samples and the set of unlabeled data include the remaining 1200 samples. For each *m:n* division, the better CVI value between the raw and HLT data is denoted as bold-faced. One can see that, for all the divisions, the HLT data outperforms the raw data except for a number of external CVIs in the 1:5 division, and for the Separation index.

Table 3. Cluster validation indices for the Uspst data. For each m: n division of data, bold-faced numbers denote better values of the CVIs.

Data / CVI	1:5 division		2:4 division		3:3 division	
	raw	HLT	raw	HLT	raw	HLT
Rand	0.909	**0.925**	0.911	**0.928**	0.909	**0.934**
Jaccard	0.272	**0.399**	0.281	**0.407**	0.273	**0.447**
FM	0.456	**0.584**	0.465	**0.594**	0.454	**0.634**
Silhouette	0.091	**0.418**	0.104	**0.417**	0.100	**0.467**
DB*	1.910	**1.172**	1.877	**1.173**	1.867	**1.105**
Dunn	0.372	**0.863**	0.466	**0.922**	0.433	**0.948**
Homogeneity*	0.336	**0.185**	0.335	**0.197**	0.339	**0.184**
Separation	0.522	**0.598**	0.533	**0.632**	0.532	**0.637**

Data / CVI	4:2 division		5:1 division		Entire set	
	raw	HLT	raw	HLT	raw	HLT
Rand	0.910	**0.936**	0.910	**0.937**	0.910	**0.962**
Jaccard	0.282	**0.459**	0.283	**0.463**	0.404	**0.702**
FM	0.465	**0.644**	0.462	**0.645**	0.575	**0.825**
Silhouette	0.106	**0.453**	0.120	**0.459**	0.148	**0.364**
DB*	1.869	**1.085**	1.811	**1.052**	1.945	**1.153**
Dunn	0.476	**0.953**	0.526	**0.940**	0.705	**0.939**
Homogeneity*	0.344	**0.189**	0.353	**0.197**	0.347	**0.318**
Separation	0.546	**0.644**	0.567	**0.653**	0.529	**0.670**

*For the DB and Homogeneity index, smaller is better.

The former case can be explained as follows. The HLT ***H*** is developed using a small number (240) of training samples to predict the class labels of a large number (1200) of testing samples. Expectedly, small training data do not fully capture the distribution structure of population data that is present in a large set of the testing data. Therefore, in such a case, the CVIs obtained from the raw testing data tend to be better than those obtained from the HLT testing data. As for the latter case concerning the Separation index, we need to interpret its values together with the Homogeneity index values.

In Table 1, the Homogeneity indices of the HLT data are better than those of the raw data, whereas for the Separation indices the raw data outperforms the HLT data. Notice that the Homogeneity and the Separation indices are not independent each other, but rather the trade-off condition exists between them. Therefore, the reversely alternating trend for the bold-faced numbers in the last two rows of the Table 1 is a natural result.

However, we need to examine closely the difference of two group means using the t-test. We performed two-sample t-test to see if the difference of CVIs for the HLT data and the raw data is statistically significant. The resulting p-values are illustrated in Table 2. In the table, the value of 0.0001 refers to a upper limit. All the actual values are less than the limit value. The p-values less than 5% significance level are denoted as bold-faced numbers, meaning the null hypothesis that two averages are equal is rejected. It should be noted that most p-values for the Separation index are greater than 0.05 so that the small differences of the Separation indices in the last row in Table 1 are statistically insignificant.

On the other hand, the differences of the Homogeneity indices are statistically significant because the corresponding p-values are less than 0.05. These observations indicate that, by the nature of the homothetic transformation, the input data are mapped to compact clusters (meaning lower and better Homogeneity index) and at the same time to globally shrunk clusters (meaning lower and lesser Separation index). The latter shrinking property is a natural consequence incurred in forming globally compact clusters by the homothetic transformation, but does not have any significant effect the goodness of clustering. This insignificancy is further evidenced by the superiority of the HLT data for the Silhouette index, because the index considers the homogeneity and separation at the same time.

The last two columns of Table 1 show the CVIs that are computed by considering the entire input data as labeled one. In this case also, one can see the clear superiority of the HLT data against the raw data.

The superiority of the HLT data can be confirmed by visual inspection as shown in Fig. 5.

Because the magnitudes of the bars in the red box should be smaller than the outside of the box because the inside bars are the distances between data in the same class. For the raw data, some distances between two samples in the same class 1 are larger than the distances from the samples belonging to different classes on the right as shown in Fig. 5(a). However, for the HLT data in Fig. 5(b), the magnitudes in the red box are lesser than any magnitudes on the right.

The CVIs computed for the Uspst data set are illustrated in Table 3. Generally, the CVIs for the Uspst are similar to those of the Coil20. Unlike the Coil20 data, however, the homothetic transformation of the Uspst satisfies the cluster assumption more strongly regardless of the data division condition.

This property is illustrated by the consistency that all the bold-faced numbers belong to the HLT data in Table 3. We also performed the t-test on the Uspst and observed that all the p-values are less than 0.05.

Therefore, we can see that the differences of the CVIs in Table 3 are statistically significant.

Semi-supervised learning experiments

In the experiments comparing the CVIs, we have shown the performance gains of the homothetic transformation but the ultimate objective should be evaluated in terms of semi-supervised learning.

We performed a number of transductive and semi-supervised learning methods: Manifold Regularization [23] and Low Density Regularization [9].

For all the learning methods in the experiments below, each data set has been randomly split into 40 or 50 labeled points and remaining unlabeled points. For example, Coil20 data set is divided into 40 labeled and 1440 unlabeled points, and Uspst is into 50 labeled and 1957 unlabeled points.

With this setting, we computed average errors on the 10-fold cross validation scheme for each data set.

Manifold Regularization

Sindhwani et al.[23] presented a semi-supervised learning method that can capture the underlying geometry of the input data by constructing a family of data-dependent kernels using the unlabeled data. The method deforms the original input space so that a decision surface can separate different classes easily in the deformed space. For experiments we used the software that is provided at: http://www.cs.uchicago.edu/~vikass/research.html. The implementation of this approach is achieved through two algorithms: Laplacian Support Vector Machines (LapSVM) and Laplacian Regularized Least Squares (LapRLS). As the authors remarked in the literature [23], varying values of these parameters do not have notable performance differences but less than 2% fluctuations.

We therefore fixed those values, and experimented with varying values of the degree (number of nearest neighbors) for graph laplacian computation and the width of the RBF kernel. Table 4 illustrates a number of experimental results of high performance among others. In the table, the error rates are tabulated in two big groups corresponding to the raw and the HLT input data. All the bold-faced numbers, each of which denotes the best result for each data set, belong to the group corresponding to the HLT input data. These results indicate that the HLT is effective in improving the performance of semi-supervised learning. Moreover, for all the data sets except Coil20, the differences of the error rates between two groups are great so that performance superiority of using the HLT data is clearly verified.

As mentioned earlier, unlike the other four data sets, G10n does not satisfy the cluster assumption of the data distribution. This may probably be related with poor performance for semi-supervised learning, as shown by the high error rates for the experiments using the raw data. Using the HLT data, we achieve the reduction of error rates from 18.96% to 0.22% for

the best case of LapSVM, and from 18.80% to 0.06% for the best case of LapRLS

Table 4. Error rates (%) of the experiments with Manifold regularization. NN refers to the degree or the number of nearest neighbors in the graph, and to the kernel width. Bold-faced numbers denote the best results.

Data	Input → Learning ↓	raw data			HLTped data		
Coil20	NN, σ	3, 0.5	4, 0.5	5, 0.4	30, 5.0	40, 5.0	50, 9.0
	LapSVM	2.92	3.37	4.01	1.51	1.60	**1.34**
	LapRLS	2.64	3.00	4.06	2.05	1.63	**1.21**
Uspst	NN, σ	5, 4.0	7, 4.0	9, 5.0	30, 7.0	40, 5.0	50, 6.0
	LapSVM	11.1	10.7	11.4	3.68	**3.67**	3.74
	LapRLS	11.2	10.9	11.3	3.69	**3.66**	3.72
Text1	NN, σ	30, 4.5	40, 1.0	50, 1.0	30, 3.0	40, 3.0	50, 2.5
	LapSVM	11.3	10.5	10.2	0.24	0.12	**0.01**
	LapRLS	11.4	10.4	9.81	0.78	0.35	**0.01**
G50c	NN, σ	30, 4.0	40, 18.0	50, 17.0	30, 50.0	40, 50.0	50, 60.0
	LapSVM	6.30	5.68	5.44	0.86	0.78	**0.74**
	LapRLS	5.92	5.72	5.18	0.18	0.10	**0.10**
G10n	NN, σ	10, 8.5	15, 8.0	20, 7.0	30, 10.0	40, 16.0	50, 20.0
	LapSVM	21.72	20.28	18.96	0.46	0.28	**0.22**
	LapRLS	22.45	20.66	18.80	0.36	0.14	**0.06**

Table 5. Error rates(%) of the experiments with Low Density Regularization. For each parameter setting, 10-fold cross validation experiments are conducted and the best one is kept in collection. From the collection, minimum, maximum, mean and standard deviation are then computed.

	raw data				HLTped data			
	min	max	mean	std	min	max	mean	std
Coil20	5.74	11.76	8.72	2.03	3.49	5.43	**3.93**	0.57
Uspst	15.28	17.49	15.98	0.92	4.15	5.47	**4.47**	0.40
Text1	5.61	6.31	5.96	0.27	0.01	0.01	**0.01**	0.00
G50c	4.80	5.88	5.41	0.36	0.47	0.62	**0.56**	0.12
G10n	13.68	16.96	15.17	1.15	0.13	0.16	**0.15**	0.01

Low Density Separation

Chapelle and Zien [9] proposed a semi-supervised classification algorithm called the Low Density Separation (LDS). LDS combines the advantageous points of two semi-supervised algorithms: an SVM on a graph-distance derived kernel and a TSVM trained by gradient descent. We experimented with the MATALB implementation of the LDS algorithm obtained at http://www.kyb.tuebingen.mpg.de/bs/people/chapelle/lds/. In the experiments, two main parameters, exponent γ and penalty C whose meanings are referred to the LDS algorithm [9], are involved. With varying values of the two parameters as suggested in the literature, we performed the 10-fold cross validation experiment for each pair. Among all the experimental results, only the best ones in terms of the error rates are tabulated in Table 5. In the table we first note the large difference of mean error rates between the experiments using the HLT data and the experiments using the raw data. Here again we experimentally verify the effectiveness of the homothetic transformation in semi-supervised learning.

For the Text1 data set, the experiments using the HLT data show near perfect classification though the error rates in Table 5 are given as 0.01 which denotes a conservative upper limit. The results for G10n are also notable in that the homothetic transformation successfully captures the geometric structure of the data though they do not satisfy the cluster assumption. This is indicated by the sharp contrast of mean error rates, 0.151% vs. 15.17%, resulting from the raw and the HLT data, respectively.

CONCLUSIONS

We have shown that HLT is useful for improving the performance of semi-supervised learning. Specifically, HLT strengthens cluster assumption of the input data while it also captures the embedded manifold structure inherent in the input data. HLT can be implemented using the MLFN's information encoding property in the hidden layer neurons. In addition to the usefulness in a classification task,

HLT is also useful for representing the input data as dimension-reduced features, which are highly useful for biological or text data. We have experimentally verified the superiority of HLT via experiments using the real world test data.

ACKNOWLEGEMENTS

This work was supported by the research grant of Pai Chai University in 2017, and also partially supported by the Korea National Research Foundation with the research grant of NRF-2017R1D1A2B03028954.

REFERENCES

[1] Mikhail Belkin and Partha Niyogi and Vikas Sindhwani, "Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples," *Journal of Machine Learning Research*, vol. 7, pp. 2399-2434, 2006.

[2] Xiaojin Zhu and Andrew B. Goldberg, *Introduction to Semi-Supervised Learning*, Morgan and Playpool, 2009.

[3] Philippe Rigollet, "Generalization Error Bounds in Semi-supervised Classification Under the Cluster Assumption," *Journal of Machine Learning Research*, vol. 8, pp. 1369-1392, 2007.

[4] Kausik Sinha and Mikhail Belkin, "Semi-supervised Learning Using Sparse Eigenfunction Bases," In: *Advances in Neural Information Processing Systems 22*, pp. 1687-1695, 2009.

[5] John D Lafferty and Larry Wasserman, "Statistical Analysis of Semi-supervised Regression," in *Advances in Neural Information Processing Systems 19*, pp. 801-808, 2007.

[6] Partha Niyogi, "Manifold Regularization and Semi-supervised Learning: Some Theoretical Analyses," *Journal of Machine Learning Research,* vol.14, pp. 1229-1250, 2013.

[7] Ming Ji and Tianbao Yang and Bibbin Lin and Rong Jin and Jiawei Han, "A Simple Algorithm for Semi-supervised Learning with Improved Generalization Error Bound," *Proceedings of the 29th International Conference on Machine Learning*, vol. 2, pp. 1223-1230, 2012.

[8] Boaz Nadler and Nathan Srebro and Xueyuan Zhou, "Statistical Analysis of Semi-supervised Learning: The Limit of Infinite Unlabeled Data," in *Advances in Neural Information Processing Systems 22*, pp. 1330-1338, 2009.

[9] Olivier Chapelle and Alexander Zien, "Semi-Supervised Classification by Low Density Separation," *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pp. 57-64, Jan. 2005.

[10] Pavan Kumar Mallapragada and Rong Jin and Anil K Jain and Yi Liu, "SemiBoost: Boosting for Semi-Supervised Learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2000-2014, 2009.

[11] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[12] Jonathan Malkin and Amarnag Subramanya and Jeff Bilmes, "On the Semi-supervised Learning of Multi-Layered Perceptrons," *Proceedings of 2009 Annual Conference of the International Speech Communication Association*, pp. 660-663, 2009.

[13] Dong-Hyun Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," In: *Workshop on Challenges in Representation Learning, ICML*, 2013.

[14] Frederic Ratle and Gustavo Camps-Valls and Jason Weston, "On pattern classification with Sammon's nonlinear mapping - an experimental study," *Pattern Recognition*, vol. 31, pp. 371-381, 1998.

[15] Bruce Meserve, "Homothetic transformations," In: *Fundamental Concepts of Geometry*, pp.166-169, Addison-Wesley, Cambridge, 1955.

[16] Kurt Hornik, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.

[17] David S Touretzky and Dean A Pomerleau, "What's hidden in the hidden layers," *Byte*, vol. 1, pp. 227-233, 1989.

[18] E J Teoh and K C Tan and C Xiang, "Estimating the number of hidden neurons in a feedforward network using the singular value decomposition," *IEEE Transactions on Neural Networks*, vol. 17, pp. 1623-1629, 2006.

[19] G. Huang and H. Babri, "Upper bounds on the number of hidden neurons in feedforward networks with arbitrary bounded nonlinear activation functions", *IEEE Transactions on Neural Networks*, vol. 9, pp. 224-229, 1998.

[20] M. Sartori and P. Antsaklis, "A simple method to derive bounds on the size and to train multi-layer neural networks", *IEEE Transactions on Neural Networks*, vol. 2, pp. 467-471, 1991.

[21] John W Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on Computers*, vol. 18, pp. 401-409, 1969.

[22] B Lerner and Hugo Guterman and Mayer Aladjem and Itshak Dinsteint and Yitzhak Romem, "On pattern classification with Sammon's nonlinear mapping - an experimental study," *Pattern Recognition*, vol. 31, pp. 371-381, 1998.

[23] Vikas Sindhwani and Partha Niyogi, "Beyond the point cloud: from transductive to semi-supervised learning," 3GPP Technical Specification Group Radio Access Network, *Proceedings 22nd international conference on Machine learning*, pp. 824-831, 2005.