EECS 545 Machine Learning

Recognition of handwritten digit with transformed invariant distance

- Project Final Report -

Due 12/14/07

Buyoung Yun
Christine Kim
Seungcheol Yang

Abstract

This paper presents a method to classify handwritten digit based on tangent vectors, which are the linear derivatives of transformations. The purpose of tangent vector is to find the distance between manifolds; a substitute of the classical Euclidean distance. Using the tangent vector, a satisfying performance was achieved, invariant to transformation. While implementing the classifier, we improved the thickness transformation robustness algorithm which was described in the paper. Furthermore, we devised modified tangent vectors combined with normal vectors of image manifold to overcome the high nonlinearity induced by transformation in the image space. This led to improve local invariance properties.

I. Introduction

Handwritten digit recognition is a classic topic in pattern recognition. This is, however, still a technically challenging problem considering an effective classifier since there are various individual handwritten characters. From given digit images, an effective classifier should classify it into the same category, while the difference between categories should be correctly identified.

In this paper, we established the classification function based on the distance measure between the input image and the given MNIST database [1]. For simplicity, the distance can be derived from a simple distance measure; such as Euclidean distance. However, while dealing with handwritten digit recognition, we need to store each class of digits in all possible positions, sizes, angles and thickness. This will be inefficient in practice [2].

Thus, an efficient classifier should give a distance between a given pattern and a prototype without the effect of transformation. This is possible as we deform the prototype to match to the given pattern for its best fit. At this point, we introduce a popular method satisfying the invariance requirement, which is called the tangent distance introduced by Patrice Y. Simard [3]. While applying the tangent distance, the sample data is represented as a vector. In addition, the set of pattern vectors are represented as a deformable pattern in terms of a specific transformation; the deformations can be x-translation, y-translation, rotation, thickness and so on.

Firstly, a continuous and differentiable image representation function with respect to transformation was derived, which can be founded in chapter 3.

Secondly, we focused on digit recognition classifier using tangent distance and modify it in a mathematical way to enhance thickness deformations that was considered as an ineffective transformation in the original paper [3].

Thirdly, we provided the methodology to enhance the performance of the classifier. Tangent distance method is derived from a linear approximation of the transformation using a

Taylor expansion without the second or higher order terms [4]. Therefore, this method can be locally invariant to transformation for small transformation. To extend a feasible range of transformation, we introduced the normal vector to calculate the distance measure between each image. This method can improve local invariance of given training data with respect to transformations.

Finally, we presented the enhanced tangent distance method with K-nearest neighbor (K-NN) method to classify handwritten digits. These results were shown in comparison with other algorithms appeared on the MNIST web site [1]. We also addressed relevant issues regarding on our simulation results.

II. Literature Review - Classifier for handwritten digits

Duta and Hart (1973) suggested baseline linear classifier that imposes a weight on each input pixels. The results are decided to indicate the label of input character by the highest sum. The weights can be decided because the classifier is linear. For simplicity, this method is usually presented the basic comparison for more advanced classifiers [5].

Another approach was suggested as a K-nearest neighbor classifier with Euclidean distance measure. Because this method is a memory based algorithm, no training procedure is required for constructing the classifier. However, the memory requirement and recognition time are large. This method is based on feature vectors rather than on the pixels. Also this is used for baseline comparison [6].

III. Tangent Vector Classifier

1. Tangent distance

Tangent distance aims to obtain a distance measure that is invariant with respect to specific transformation. Without changing a known priori, the distance should be allowed to identify between a pattern and a transformed prototype. For this assumption, we introduce the concept of manifold: the set of possible transformation of the pattern. Each manifold of the pattern has the property that it is invariant to pattern transformation. Therefore, the minimal distance of between two manifolds S_E and S_P can be considered as the invariant distance to transformation between E and P, which is described as $D(E, P)$.

However, it is still expensive and unreliable work to define a manifold function because the manifold is generally a high ordered nonlinear transformation. To define an attractive representation, tangent distance method provides the approach to approximate the distance between transformed patterns by linearization. First, we assumed that a manifold is a continuous function with respect to the pattern and transformation parameters. In this regards, we can approximate manifold S by Taylor expansion with respect to a particular parameter α_i

as follows:

$$s(P,\alpha) = s(P,0) + \sum_{i=1}^{L} \alpha \frac{\partial s(P,\alpha_i)}{\partial \alpha_i} + \sum_{i=1}^{L} O(\alpha_i^2) \approx P + \alpha T$$

For simplicity, we can neglect second order term and higher order terms. As a result, nonlinear manifold can be approximated as the linearization function, which is a hyperplane in pattern space. By using linearized approximation, the invariant distance between two patterns can be estimated by the distance of these linear surfaces.

This methodology can gain several advantages:

- To reduce computing cost with linearization
- To simplify to find minimum distance with quadratic programming
- Local invariance by the linearization approximation: this property is efficient when distinguishing 6 and 9 (9 can be one of transformed form of 6 by 180 degrees rotation.)

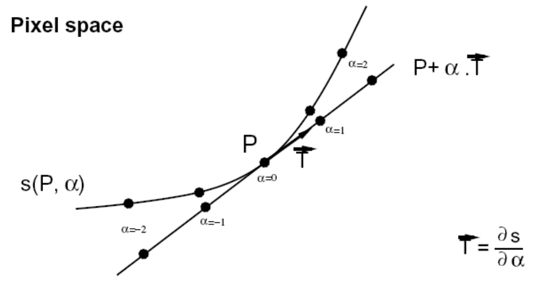

Figure 1: Representation of the effect of the rotation in pixel space [2]

Figure1 illustrates the concept on linearization of manifold with respect to a specified transformation. The curved line represents the manifold of P, which is introduced as S_P. In addition, S(P, α) is defined as the particular points of manifold, which means a pattern P transformed by parameter α. For instance, if we let α to be the quantity of x-translation, when α=1, it will denote the pattern where P is shifted by 1 by x axis. The manifold is highly nonlinear functions in pattern space. Therefore as shown in Figure 1, the manifold can be simply estimated as a linearized function of $P+\alpha\vec{T}$ by using the equation of (1), which T is called tangent vector.

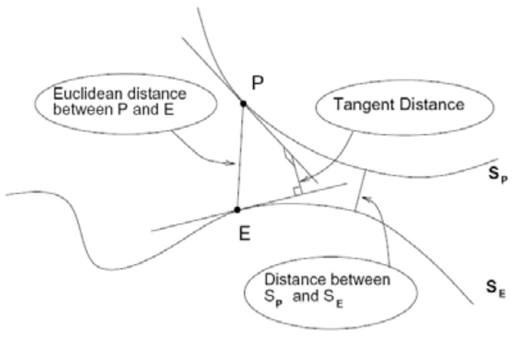
Euclidean distance between P and E
P
Tangent Distance
S_P
E
S_E
Distance between S_P and S_E

Figure 2: Illustration of Euclidean distance, tangent distance and distance of manifolds [2]

Figure 2 shows the concept of tangent distance method. Regarding given patterns P and E in pattern space, Euclidean distance is the distance between point P and E. The minimum distance between two manifolds is the minimum distance which is invariant to transformation. Last of all, tangent distance is the minimum distance between two hyper planes which are linearly approximated at P and E. The aim of tangent distance method is to access the minimum distance with reflecting the transformation of patterns. Ideally, the method can achieve good performance.

2. Tangent vectors

Tangent vectors are derivatives of manifold with respect to transformations, which can span a tangent space on all possible transformation of a pattern [4]. As mentioned in tangent distance, we should access the distance measure with respect to tangent vectors. To define tangent vector, the manifold function has assumed a continuous function. However, it is hardly differentiable because the image is practically a set of discrete data. To solve this problem, we present a differentiable interpolation functions by using convolution with a two dimensional Gaussian function. For simplicity, a set of an image can be represented as two dimensional functions.

$$P'(x,y) = \sum_{i,j} P(x,y) \cdot \delta(x-i) \cdot \delta(y-j)$$

,where $P(x,y)\ means\ a\ set\ of\ binary\ data\ of\ an\ image$

With this binary set, we can map the pattern to the differentiable function using convolution with Gaussian functions as follows:

$$f(x,y) = \sum_{i,j} P(x,y) \cdot g_\sigma(x-i, y-j)$$

The result of convolution is represented as a sum of Gaussian functions. We can explain two reasons why the Gaussian function is selected for the convolution function. First, we can control the locality of invariant with variance of Gaussian function. The linear approximation can be better when f is smoother. In general, the variance of Gaussian function is 0.9 in tangent distance method. Second, the derivative of f can be simply derived from the closed form of Gaussian function. This is because the property of convolution is as follows:

$$L_x(f) = D(P * g_\sigma) = P * \frac{\partial g_\sigma}{\partial x}$$

The function with Gaussian convolution and two tangent vectors with respect to x, y translation are shown in Fig 3.

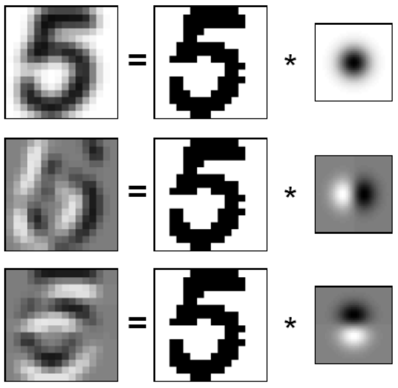

Figure 3: Graphic Illustration of computational of f and two tangent vector corresponding to x, y

3. Implementation

3.1 Linearization of Manifold

We have assumed that the pattern can be estimated a sum of two dimensional Gaussian function using interpolation, which is a continuous function. With this assumption, the pattern can be represented the transformable function in terms of two dimensional transformation. Considering that the manifold is a set of all possible transformed patterns, we can estimate the manifold with the pattern and a transformation with respect to parameter α.

First, we should consider a linear approximation of manifold using Taylor expansion. The method can be simplified to compute $P+\alpha\vec{T}$, where α is a set of parameters for transformation and $\vec{T}$ is a set of tangent vectors. As shown in Figure 1, $P+\alpha\vec{T}$ is tangent to S_P at P. This could be obtained from the equation (1) by,

$\overrightarrow{L_P} = \left.\frac{\partial s(P,\alpha)}{\partial \alpha}\right|_{\alpha=\vec{0}} = \left.\left[\frac{\partial s(P,\alpha)}{\partial \alpha_1}, \cdots, \frac{\partial s(P,\alpha)}{\partial \alpha_m}\right]\right|_{\alpha=\vec{0}}$ where $\alpha = (\alpha_1, \alpha_1, \ldots, \alpha_m)$ and m is the number of transformations we consider. Herein, we can consider $\vec{T} = \vec{L}_P$. We evaluate each terms of tangent vectors in terms of two dimensions. The manifold can be represented as follows:

$$s(f,\alpha) = f + \sum \alpha_i L_{\alpha i} + O\left(||\alpha||^2\right)(f)$$

, where f is an interpolating function of pattern P. We can consider transformations of a pattern in association with f. For example, the rotational transformation can be considered:

$$s(f(x,y),\theta) = f(x cos\theta + y sin\theta, -x sin\theta + y cos\theta)$$

If we estimate $\vec{L}_P$ with derivates of the manifold with respect to θ = 0.

$$\frac{\partial s(f,\theta)}{\partial \theta} = \left(y\frac{\partial}{\partial x} + (-x)\frac{\partial}{\partial y}\right)f$$

which corresponds to a tangent vector of rotational transformation. In this paper, we apply 7 types of transformation: consider x-translation, y-translation, rotation, scaling, parallel hyperbolic transformation, diagonal hyperbolic transformation and thickness. All transformation are represented in APPENDIX I.

3.2 Tangent distance

Let tangent distance between two patterns E and P to be D(E, P), which is defined as $D(E,P) = \min_{x\in T_E, y\in T_P} || x-y ||^2 = \min_{a_E, \alpha_P} || E + a_E L_E - P - \alpha_P L_P ||^2$ where L_E and L_P are the tangent vector of P and E respectively. In other words, we want to find the adequate points of the minimum tangent distance in Figure 2. Since these hyper planes are known, only concern is to find α_E and α_P for each hyper plane E and P. But this is simple linear quadratic programming. The optimality condition can be derived from stationary point.

$$\frac{\partial D(E,P)}{\partial \alpha_E} = 2(E - P - L_P\alpha_P + L_E\alpha_E)^T L_E = 0$$

$$\frac{\partial D(E,P)}{\partial \alpha_P} = 2(P - E - L_E\alpha_E + L_P\alpha_P)^T L_P = 0$$

The solution of this system comes out as,

$$(L_{PE}L_{EE}^{-1}L_E^T - L_P^T)(E-P) = (L_{PE}L_{EE}^{-1}L_{EP} - L_{PP})\alpha_P$$
$$(L_{EP}L_{PP}^{-1}L_P^T - L_E^T)(E-P) = (L_{EE} - L_{EP}L_{PP}^{-1}L_{PE})\alpha_E$$

where $L_{PE}=L_P^T L_E$, $L_{EE}=L_E^T L_E$ and $L_{PP}=L_P^T L_P$.

3.3 Modified thickening

The previous tangent vector for thickness is derived from divergence of each point. This aims to add pixel in the normal direction of an image:

$$L_x = ||\nabla||$$

However, this approach did not provide successful result on thickening invariance. For this reason, we introduce a new method to treat the thickness of an image. The method is to use bandwidth of Gaussian functions. Since the bandwidth in Gaussian function we used in making images as continuous surface controls interpolation between pixel points, we can use it as a thickening parameter:

$$L_T = \frac{\partial f}{\partial \sigma}$$

3.4 Modification on tangent distance with normal vectors

Before modification, we assumed that every possible transformation occurs simultaneously with small probability. This means that more than two transformations cannot deform the given images. In other words, the classifier only emphasizes the dominant transformation. Then s(P, α) can be linearized as $P + \alpha\vec{T} + \beta_i\vec{N}_i$ where i is the index corresponds to the dominant transformation such that $i \in [1, m]$ and $\vec{N}_i$ is the corresponding normal vector. Then our modified distance measure is changed as follows;

$$D_i(E,P) = \min_{a_E,\alpha_P,\beta_E^{(i)},\beta_P^{(i)}} || E + a_E L_E + \beta_E^{(i)} N_E^{(i)} - P - \alpha_P L_P - \beta_P^{(i)} N_P^{(i)} ||^2 \text{ and } i \in [1,m]$$

$$D(E,P) = \min_{i\in[1,m]} D_i(E,P)$$

We will show how much this modification improves the local invariance to transformation than the original tangent distance later.

V. Simulation Results

1. Extraction of tangent vectors from samples

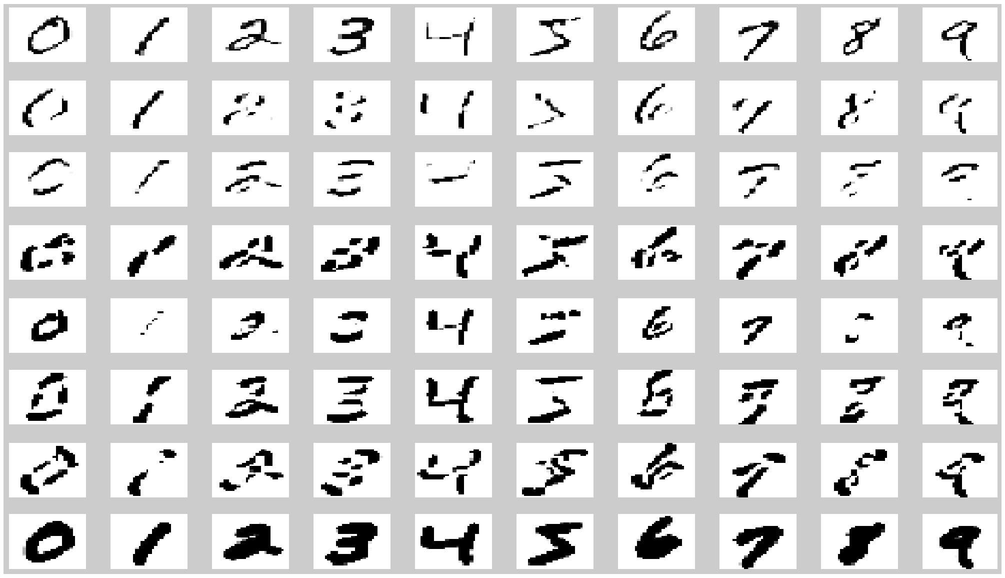

Figure 4: Tangent vector: The first row shows the original images and other rows are different tangent vectors from different transformation. Starting from the second row, each transformation is: x-translation, y-translation, rotation, scaling, parallel hyperbolic transformation, diagonal hyperbolic transformation and thickness.

2. Comparison on Euclidean distance, Tangent distance and modified tangent distance with normal vectors

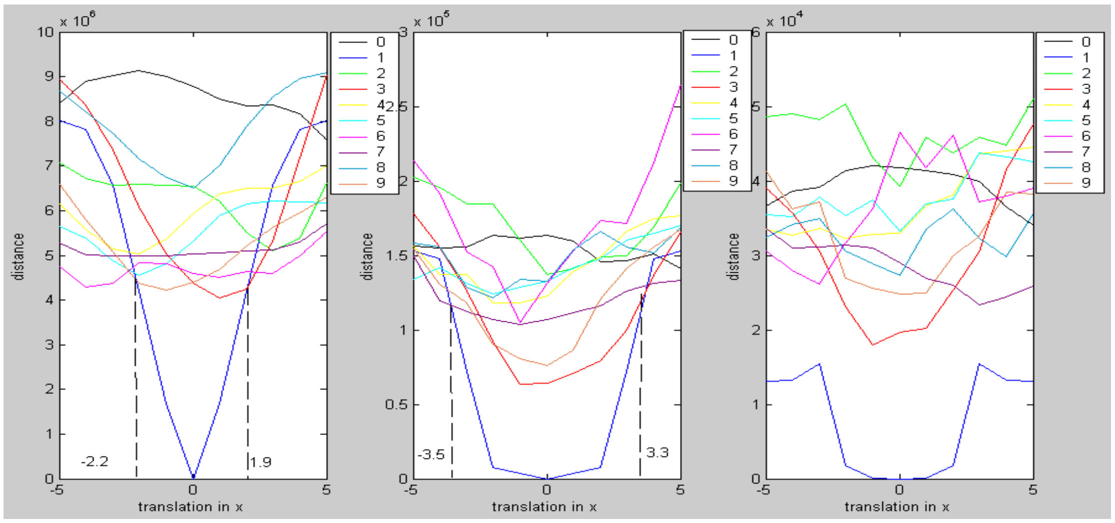

Figure 5: Distance between '1' which is transformed with x-translation and other digits. From the left to the right, Euclidean, Tangent and Modified Tangent with normal vectors are used.

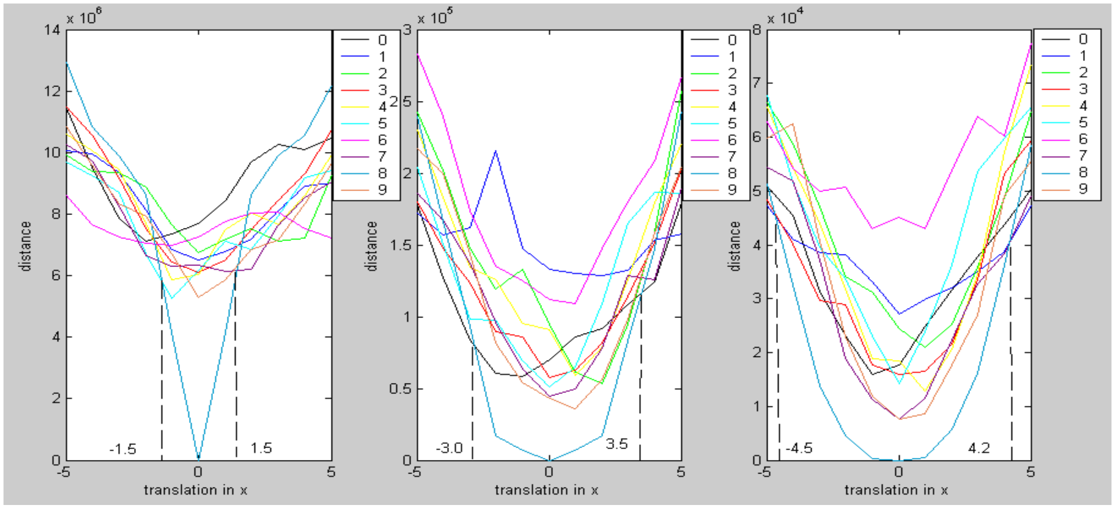

Figure 6: Distance between '8' which is transformed with both x and y translation and other digits. From the left to the right, Euclidean, Tangent and Modified Tangent with normal vectors

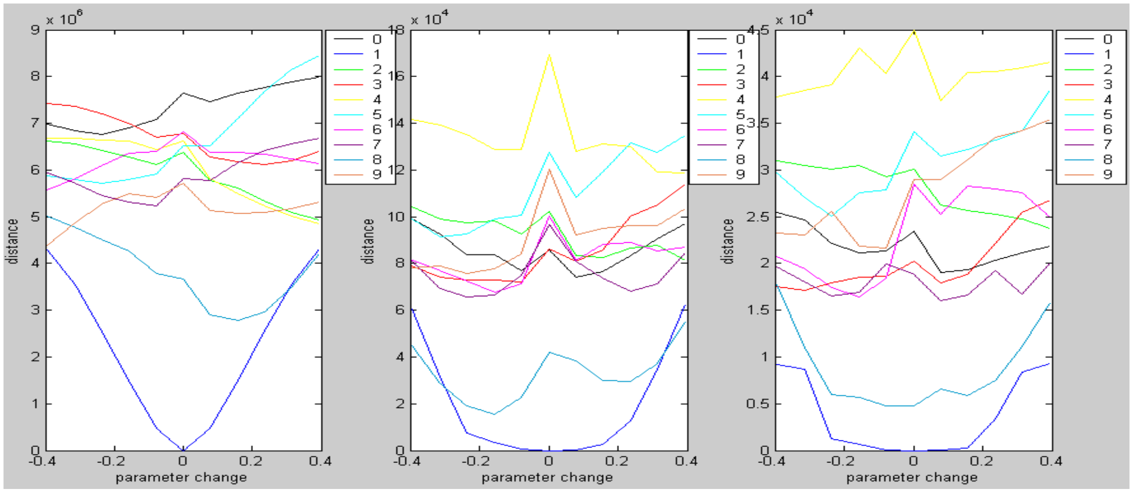

Figure 7: Distance between '1' which is transformed with rotation and other digits. From the left to the right, Euclidean, Tangent and Modified Tangent with normal vectors are used.

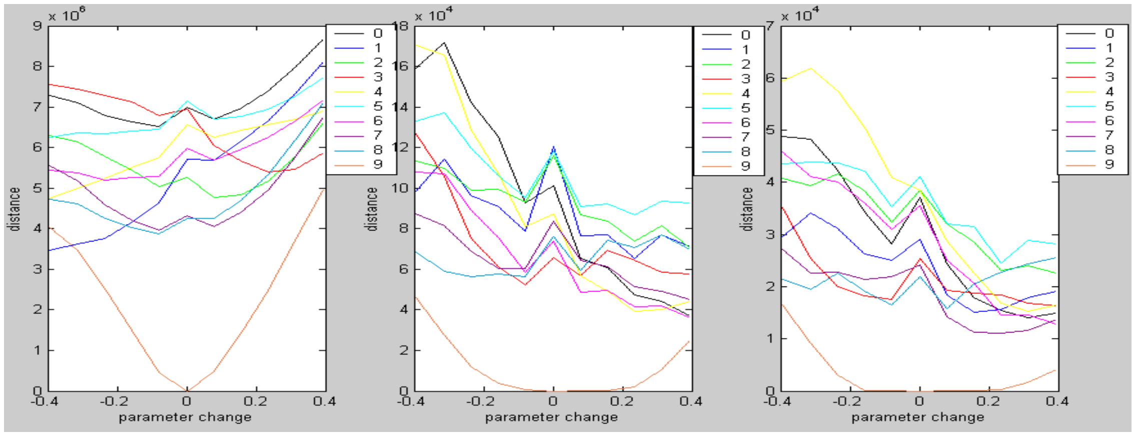

Figure 8: Distance between '9' which is transformed with scaling and other digits. From the left to the right, Euclidean, Tangent and Modified Tangent with normal vectors are used.

3. *Thickness invariant improvement*

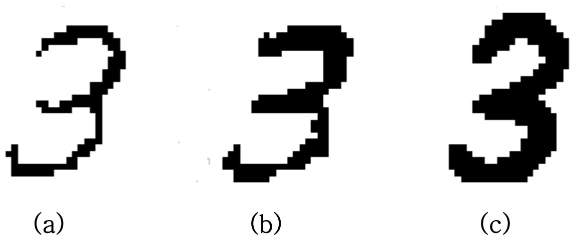

Figure 9: '3' digit with various thicknesses.

	Euclidean Distance(DE)	Tangent Distance(DT1)	New tangent Distance (DT2)	DT1/DE	DT2/DE
D|(a)-(b)|	2295.00	1251.55	309.94	0.545	0.135
D|(a)-(c)|	2686.59	1175.44	278.94	0.437	0.103
D|(b)-(c)|	3184.94	1343.98	402.14	0.42	0.126

Table 1: Comparison on various distance measures on figure 9.(a), 9.(b) and 9.(c)

4. Classification with K-NN

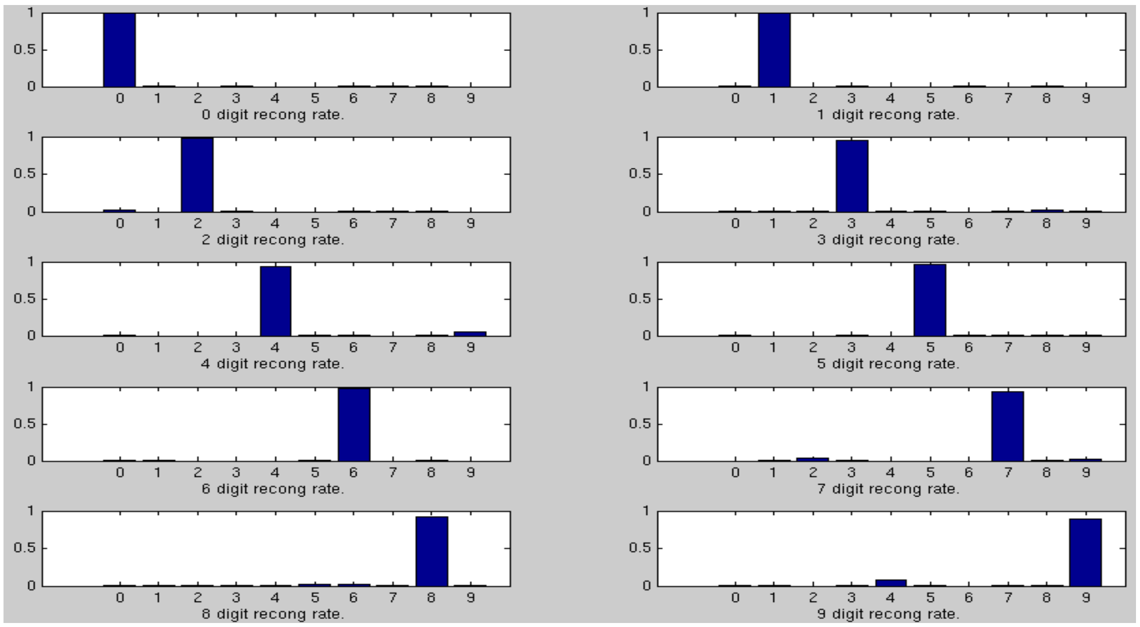

Figure 10: The simulation results of recognition rates for each digit. We used K-NN classification with K=3. The overall error rate is 4.6%

VI. Analysis

We have implemented the experiments with the datasets from MNIST [1].

In figure 4, we calculated tangent vectors for each sample digit. As we can see, since we consider seven deformations here, there are seven components in tangent vector.

From figure 5 to figure 7, we can observe the degree of robustness to each transformation. When the curve is in the most below, it means the distance between the corresponding digit and input is minimized, thus correctly classified. By these graphs, it is easy to see that the tangent distance has a larger correct-decision range than Euclidean distance. In addition, if we use our modified tangent distance combined with normal vectors, we can still improve the local invariance region as it is seen in those figures. However from figure 8, using normal vectors do not always give advantages. Especially for scaling transformation, it does not give any more robustness than conventional tangent distance. But from several experiments, we learned that considering normal vectors in distance measure does not hurt our performance.

According to the original paper [3], the author says that tangent vector for thickness was not successful. We did not use the method that the author mentioned in his paper, but use our own idea. With this modification, we can see a great improvement regarding thickness invariance in table 1.

Finally we embedded our new distance measure in K-NN classifier with K=3. As it is shown in figure 10, we obtained recognition rates on each digit and a 4.6% overall error rate. However this result is worse than the original error rate, 1.1% [3]. There is a reason about it. We added thickness invariance to our distance measure. However, when considering 4 and 9, thickness invariance property worsens the classification between them. Same situation happened in 1 and 8 as well as 1 and 9. Figure 10 shows that improving local invariance on transformation does not save us from differentiating 4 from 9.

VII. Conclusions

So far we have implemented basic algorithms of tangent distance and investigated its transformation invariant property. Furthermore, we modified the original method and advance the robustness of local invariant property. In addition, we also implemented thickness tangent vector on our own and experiment its robustness on various thickness.

However, transformation invariance does not always guarantee the perfect classification. By adding thickness robustness, we had less successful classification result, even though our experiment on robustness regarding thickness was successful as well as improving tangent distance with the help of normal vectors is successful.

Overall, we achieved 4.6% error rate and our method outperformed several linear classifications which were introduced on MNIST web site [1], but failed to achieve error rate less than 1%.

References

[1] http://yann.lecun.com/exdb/mnist/

[2] Kenneth Joseph Wilder, University of Massachusetts Amherst, 1998: Decision tree algorithms for handwritten digit recognition

[3] P. Simard, Y. LeCun, J. Denker and B. Victorri: Transformation Invariance in Pattern Recognition, Tangent Distance and Tangent Propagation, in Orr, G. and Muller K. (Eds), Neural Networks: Tricks of the trade, Springer, 1998

[4] Daniel Keysers, Wolfgan Macherey, Hermann Ney: Adaptation in Statistical Pattern Recognition Using Tangent Vectors, 2004

[5] R. O. Duta and P. E. Hart, Pattern Classification and Scene Analysis, Chapter 4, John Wiley and Sons, 1973

[6] Yann LeCun, L. D. Jackel, Leon Bottou, Corinna Cortes: Learning Algorithms for Classification: A Comparison on Handwritten Digit Recognition.

APPENDIX I

Type	Transformation	Tangent Vector
X-translation	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x+\alpha\\y\end{pmatrix}$	$L_x = \frac{\partial}{\partial x}$
Y-translation	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x\\y+\alpha\end{pmatrix}$	$L_y = \frac{\partial}{\partial y}$
Rotation	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x\ cos\alpha - y\ sin\alpha\\x\ sin\alpha + y\ cos\alpha\end{pmatrix}$	$L_R = y\frac{\partial}{\partial x} + (-x)\frac{\partial}{\partial y}$
Scaling	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x+\alpha x\\y+\alpha y\end{pmatrix}$	$L_S = x\frac{\partial}{\partial x} + y\frac{\partial}{\partial y}$
Parallel hyperbolic	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x+\alpha x\\y-\alpha y\end{pmatrix}$	$L_P = x\frac{\partial}{\partial x} - y\frac{\partial}{\partial y}$
Diagonal hyperbolic	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x+\alpha x\\y+\alpha y\end{pmatrix}$	$L_D = y\frac{\partial}{\partial x} + x\frac{\partial}{\partial y}$
Thickness	$\begin{pmatrix}x\\y\end{pmatrix} \to \begin{pmatrix}x+\alpha\gamma_x\\y+\alpha\gamma_y\end{pmatrix}$,where $(\gamma_x, \gamma_y) = \frac{\nabla f(X)}{\|\|\nabla f(X)\|\|}$	$L_x = \|\|\nabla\|\|$
Modified Thickness		$L_T = \frac{\partial f}{\partial \sigma}$

Contribution

1. Buyoung implemented the tangent distance and devised a new tangent distance with normal vectors. He also did experiments that appear on the paper.

2. Christine implemented the K-NN classifier with tangent distance. She also made program which converted test, train data sets convertible to our program.

3. Seungcheol devised a new tangent vector regarding thickness. He also surveyed several papers regarding transformation invariant classification.