LETTER

Classifying Categorical Data Based on Adoptive Hamming Distance

Jae-Sung LEE[†], ***Nonmember and*** **Dae-Won KIM**[†a)], ***Member***

SUMMARY In this paper, we improve the classification performance of categorical data using an Adoptive Hamming Distance. We defined the equivalent categorical values and showed how those categorical values were searched to adopt the distance. The effectiveness of the proposed method was demonstrated using various classification examples.

key words: *adoptive hamming distance, hamming distance*

1. Introduction

Selection of the distance measure is important in the categorical pattern classification problem: the Hamming distance, Value Difference Metric, and Class Dependent Weight Dissimilarity have been widely used distance measure [1], [2]. In this study, we extend the Hamming Distance to improve classification performance of categorical data set in terms of the accuracy. We dealt with the given data set that contains n patterns, k attributes, and was separated into W classes. The pattern X in the data set was defined as X={$x_1, \ldots, x_k$}. The Hamming distance (D_h) between X and the unseen pattern Z was defined as follows:

$$D_h(X, Z) = \sum_{m=1}^{k} I_h(x_m, z_m) \tag{1}$$

where

$$I_h(x_m, z_m) = \begin{cases} 0 & \text{if } x_m = z_m \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

x_m and z_m are the categorical attribute values in the mth attribute of patterns X and Z. Let us illustrate a data set in Table 1 composed of two categorical values and classes. Consider an unseen pattern $P_u = \{Low, Blue\}$, the distance between the training pattern P_1 and P_u was evaluated using the following equation:

$$\begin{aligned} D_h(P_1, P_u) &= I_h(High, Low) + I_h(Red, Blue) \\ &= 1 + 1 = 2 \end{aligned}$$

Similarly, $D_h(P_2, P_u) = 1$, $D_h(P_3, P_u) = 2$, and $D_h(P_4, P_u) = 1$. If we use the k-nearest neighbor classifier (k-NN, k=3) to classify P_u, the classifier finds the three nearest patterns in the given training set. The three nearest patterns of P_u were P_2, P_4, P_1(**Yes**, **No**, **Yes**) or P_2, P_4, P_3(**Yes**, **No**, **No**). Thus, the classifier could not absolutely determine the class of P_u.

Table 1 An example of categorical data set.

	Pattern	Quality	Color	Class
Training	P_1	*High*	*Red*	**Yes**
	P_2	*High*	*Blue*	**Yes**
	P_3	*High*	*Gold*	**No**
	P_4	*Low*	*Red*	**No**

The problem was that the Hamming Distance does not address the difference between categorical values according to class [2]. Now referencing the training data set in Table 1, the categorical value of pattern P_u, *Blue*, was observed only in class **Yes**. Therefore, regarding the categorical value *Red* of P_1 and the categorical value *Blue* of P_u as equivalent may imply an improved classification performance. Thus, $I_h(Red, Blue)$ should be 0 for the classification of P_u.

The equivalence of categorical values affected classification performance. Obviously, the categorical value *Blue* of P_u was not observed in class **No**, so we could not regard the categorical value *Blue* of P_u and categorical value *Red* of P_4 as equivalent. Thus, $I_h(Red, Blue) = 0$ in class **Yes**, and $I_h(Red, Blue) = 1$ in class **No**. Using the previous example, we know that finding such pairs of categorical values in a specific class is important to improve classification performance. We developed the Adoptive Hamming Distance (AHD) that adapts to the given data set to achieve this.

2. Proposed Method

2.1 Adoptive Hamming Distance

The Adoptive Hamming Distance (D_{h^*}) is defined by the following equation:

$$D_{h^*}(X, Z) = \sum_{m=1}^{k} I_{h^*}(x_m, z_m) \tag{3}$$

where

$$I_{h^*}(x_m, z_m) = \begin{cases} 0 & \text{if } x_m = z_m \\ d_c(x_m, z_m) & \text{if } x_m \neq z_m \end{cases} \tag{4}$$

The term, $c \in W$, is the class name. In the $I_{h^*}(\cdot,\cdot)$ function, $d_c(x_m, z_m) \in \{0, 1\}$ denotes the categorical value difference. If the categorical values x_m and z_m are regarded as equivalent in the given class c, then the categorical value difference $d_c(x_m, z_m)$ is set to 0.

Manuscript received June 12, 2009.

Manuscript revised September 8, 2009.

[†] The authors are with School of Computer Science and Engineering, Chung-Ang University, Seoul 156–756, Korea.

a) E-mail: dwkim@cau.ac.kr (Corresponding author)

DOI: 10.1587/transinf.E93.D.189

Table 2 An example of procedural steps.

Initial	Initialize all of variables. S_0={ 1,1,$\cdots$,1,1 }, P_0=20% accuracy S_t={ 1,1,$\cdots$,1,1 }, P_t=20% accuracy
Step 1	Change a single $d_c(\cdot,\cdot)$ in S_{t+1} to 0, and set P_{t+1}. The performance is improved 20% to 40%. Save P_{t+1} to P_0, and set S_{t+1} to S_t. (*) indicates the changed value. S_t={ 1,1,$\cdots$,1,1 }, P_t=20% accuracy S_{t+1}={ $\mathbf{0}^*$,1,$\cdots$,1,1 }, P_{t+1}=$\mathbf{40\%}^*$ accuracy S_0={ $\mathbf{0}^*$,1,$\cdots$,1,1 }, P_0=$\mathbf{40\%}^*$ accuracy
Step 2	After iterations, maximum performance of S_{t+1} has been found. Also, it is saved to S_0. S_0={ 1,$\mathbf{0}^*$,$\cdots$,1,1 }, P_0=$\mathbf{60}^*$% accuracy
Step 3	If P_0 is improved, go to Step 1. If P_0 is not improved, then stop the procedure. we get the Adoptive Hamming Distance S_0.

Algorithm 1 Proposed Method

```
$S_0 = \{d_c(x_m, z_m) \leftarrow 1 | \forall x_m, z_m, x_m \neq z_m\}$
$t \leftarrow 1$
$S_t \leftarrow S_0$
Initialize $P_0$ with the classification performance of $S_0$
Initialize $P_t$ with the classification performance of $S_t$
loop
    for each class and each categorical value do
        $S_{t+1} \leftarrow S_t$
        Change a single $d_c(\cdot,\cdot) \leftarrow 0$ in $S_{t+1}$
        Evaluate $P_{t+1}$
        if $P_{t+1} > P_t$ then
            $S_0 \leftarrow S_{t+1}$, and $P_0 \leftarrow P_{t+1}$
        end if
    end for
    if $P_0$ is not improved then
        return Adoptive Hamming Distance $S_0$
    else
        $S_{t+1} \leftarrow S_0$
    end if
    $t \leftarrow t + 1$
end loop
```

Since we observed that classification performance has improved when $d_c(\cdot,\cdot)$ changed to 0, we should find such a $d_c(\cdot,\cdot)$. For simplicity, we defined a set S that followed the definition given below:

$$S = \{d_c(x_m, z_m) | \forall (x_m, z_m), x_m \neq z_m\} \tag{5}$$

Generally $d_c(x_m, z_m) = 1$ when $x_m \neq z_m$. Therefore, all of the elements in the set S are 1 at the initial step. The set S of the data set shown in Table 1 was used in the following example.

$$\begin{array}{rl} S = & \{\ d_{Yes}(High, Low), \cdots, d_{No}(Gold, Red)\ \} \\ = & \{\qquad 1, \qquad\qquad \cdots, \qquad\quad 1 \qquad\quad \} \end{array}$$

If we regard the categorical values *Blue* and *Red* as equivalent in class **Yes**, then one of the $d_{YES}(\cdot,\cdot)$ in S changes to 0. S_t means the S of t step.

$$\begin{array}{rl} S_t = & \{\cdots, d_{Yes}(Red, Blue) = 1, \cdots\} \\ S_{t+1} = & \{\cdots, d_{Yes}(Red, Blue) = 0, \cdots\} \end{array}$$

Now we must find the circumstance in which $d_c(\cdot,\cdot)$ changes to 0 in S, so that classification performance is improved. We applied a greedy search to find such a $d_c(\cdot,\cdot)$ in S.

The detailed procedure of the proposed method is given in Algorithm 1. The classification performance P following S was evaluated at each step. If P was not improved, the loop procedure stopped and we obtained S_0, the Adoptive Hamming Distance that is finally adopted for the given data set. An example of the procedural steps is shown in Table 2, to demonstrate what was changed in S.

3. Experimental Results

We applied the nearest neighbor(NN) classifier based on the proposed AHD(NN+AHD) to widely used data sets, such as Balance, Monk2, and Tic-Tac-Toe [3], to test the effectiveness of the proposed method. We compared the performance of NN+AHD to the conventional nearest neighbor classifier, based on the Hamming Distance(NN+HD), the Value Difference Metric(NN+VDM), and the Class-Dependent Weight Dissimilarity(NN+CDW). From 10% to 50% of the original data set was held out as independent test set; remaining data was used for training each method. We iterated each method 100 times to examine the average performance of the classification tasks.

The Monk2 data set contains 601 data, where each data has 6 categorical attributes. The Balance data set contains 625 data, where each datum has 4 categorical attributes. The Tic-Tac-Toe data set contains 958 data, where each datum has 9 categorical attributes.

Figure 1 shows the classification performance of each method for Monk2 data set. The proposed NN+AHD showed superior classification performance to other conventional methods in 10% to 50% hold out conditions. Classification accuracies of NN+HD, NN+VDM, and NN+CDW were 79.5%, 89.9%, and 93.2% respectively on 10% hold out condition, while the classification performance of NN+AHD was 99.0%. Thus, we can see that the performance of NN+AHD is improved 19.5% from conventional NN+HD on 10% hold out condition. On 50% hold out condition, classification accuracies of the three conventional methods were 68.1%, 75.0%, and 87.0% respectively, while the classification performance of NN+AHD was 92.2%; NN+AHD was improved 24.1% from NN+HD. Compared to NN+CDW, the NN+AHD was improved 5.2%. Thus NN+AHD was more accurate than the other methods on changing the proportion of test set from 10% to 50%.

It is interesting to note that the AHD method shows similar effect like the feature selection method in a specific data set, such as a binary data set where the *m*th attribute of the data set has only two values. For example, we represent these two values as *Red* and *Blue*. If AHD adopts the difference of those values from 1 to 0, then attribute *m* does not contribute to the classification results; the adoption process of AHD on the binary data set plays a role of removing irrelevant attributes.

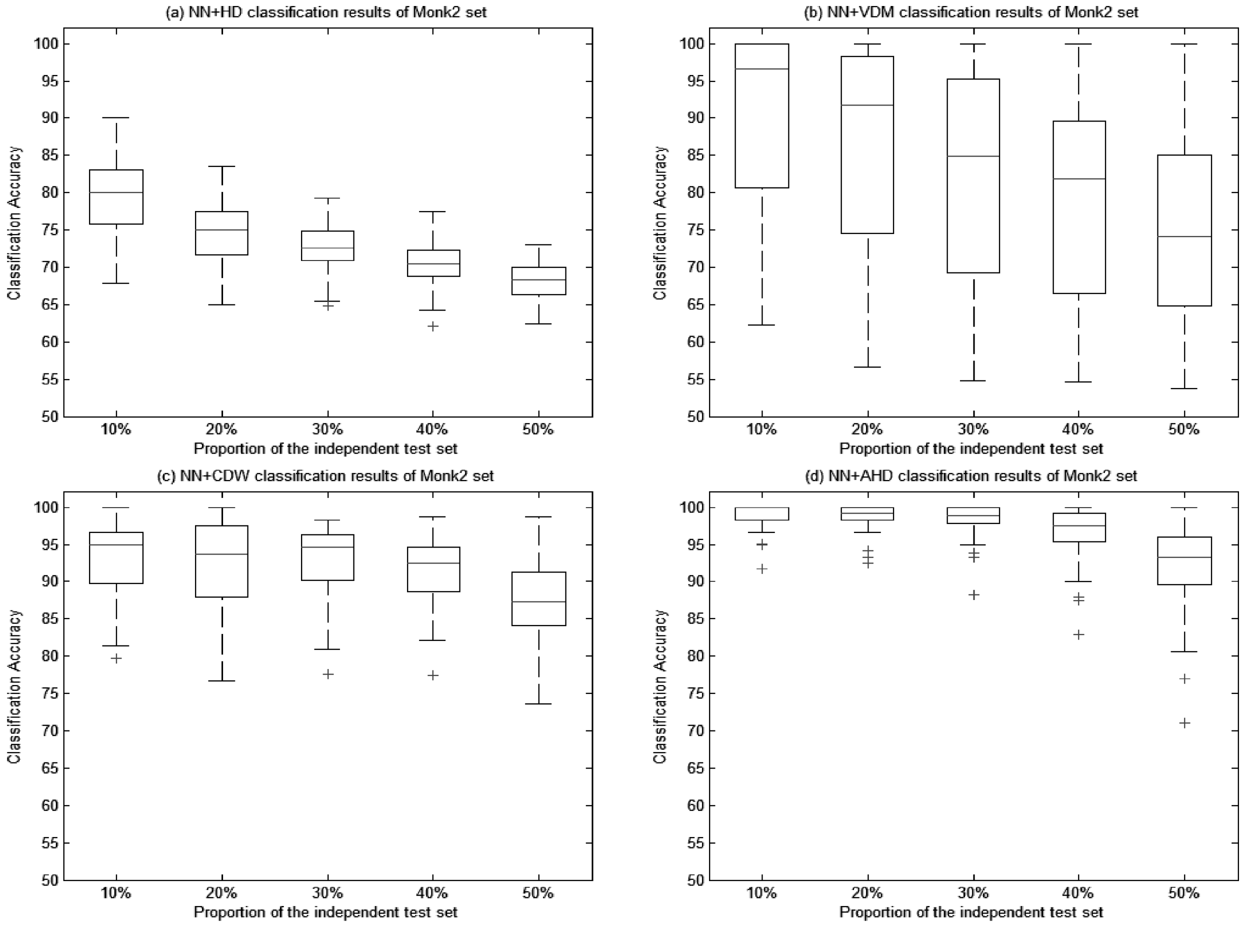

Fig. 1 Classification results of Monk2 data set using the NN+HD(a), NN+VDM(b), NN+CDW(c), and NN+AHD(d).

Table 3 The overall classification results of NN+HD, NN+VDM, NN+CDW, and NN+AHD.

Hold Out(%)	10%			30%			50%		
Data set	Balance	Monk2	Tic-Tac-Toe	Balance	Monk2	Tic-Tac-Toe	Balance	Monk2	Tic-Tac-Toe
NN+HD	86.2%	79.5%	98.7%	82.5%	72.8%	97.7%	79.9%	68.1%	95.7%
NN+VDM	84.2%	89.9%	92.1%	82.0%	82.5%	91.8%	80.1%	75.0%	90.1%
NN+CDW	79.4%	93.2%	72.2%	77.9%	93.1%	71.2%	77.3%	87.0%	69.8%
NN+ADH	**89.8%**	**99.0%**	**98.8%**	**87.2%**	**98.3%**	**98.4%**	**84.9%**	**92.2%**	**97.2%**

Table 3 shows the overall classification accuracy of the four methods in three data sets. In the Balance data set of 10% hold out condition, the classification performance of NN+HD was 86.2%, and those of NN+VDM and NN+CDW are 84.2% and 79.4% respectively, while that of NN+AHD was 89.8%. Thus the NN+AHD was 10% more accurate than NN+CDW. On the 50% hold out condition, the performance of NN+HD was 79.9%, and those of NN+VDM and NN+CDW were 80.1% and 77.3% respectively, while that of NN+AHD is 84.9%. From Table 3, we find that NN+AHD provided the best performance irrespect of the three data sets and hold out conditions.

In Fig. 2, we examined the reliability of the proposed method over the proportion of the independent test set of Tic-Tac-Toe. The classification accuracy of NN+AHD is slowly decreasing from 98.8% to 97.2% in accordance with increasing test set from 10% to 50%. In contrary, classification accuracy of NN+HD is decreasing from 98.7% to 95.7%, and NN+VDM is decreasing from 92.1% to 90.1%, and NN+CDW is decreasing from 72.2% to 68.8%. The accuracy of NN+CWD drops as twice as faster than NN+AHD with increasing test set. The accuracy standard deviation of NN+HD, NN+VDM, and NN+CDW on 10% hold out condition is ±1.08%, ±2.81%, and ±3.78% respectively, while that of NN+AHD is ±1.03%. Thus the NN+AHD is more reliable than the counterparts on all five hold out conditions.

4. Conclusion

To deal with the categorical data classification problem, HD, VDM, and CDW have been widely used. Specifically, in the VDM, the weight value is used to control the distance

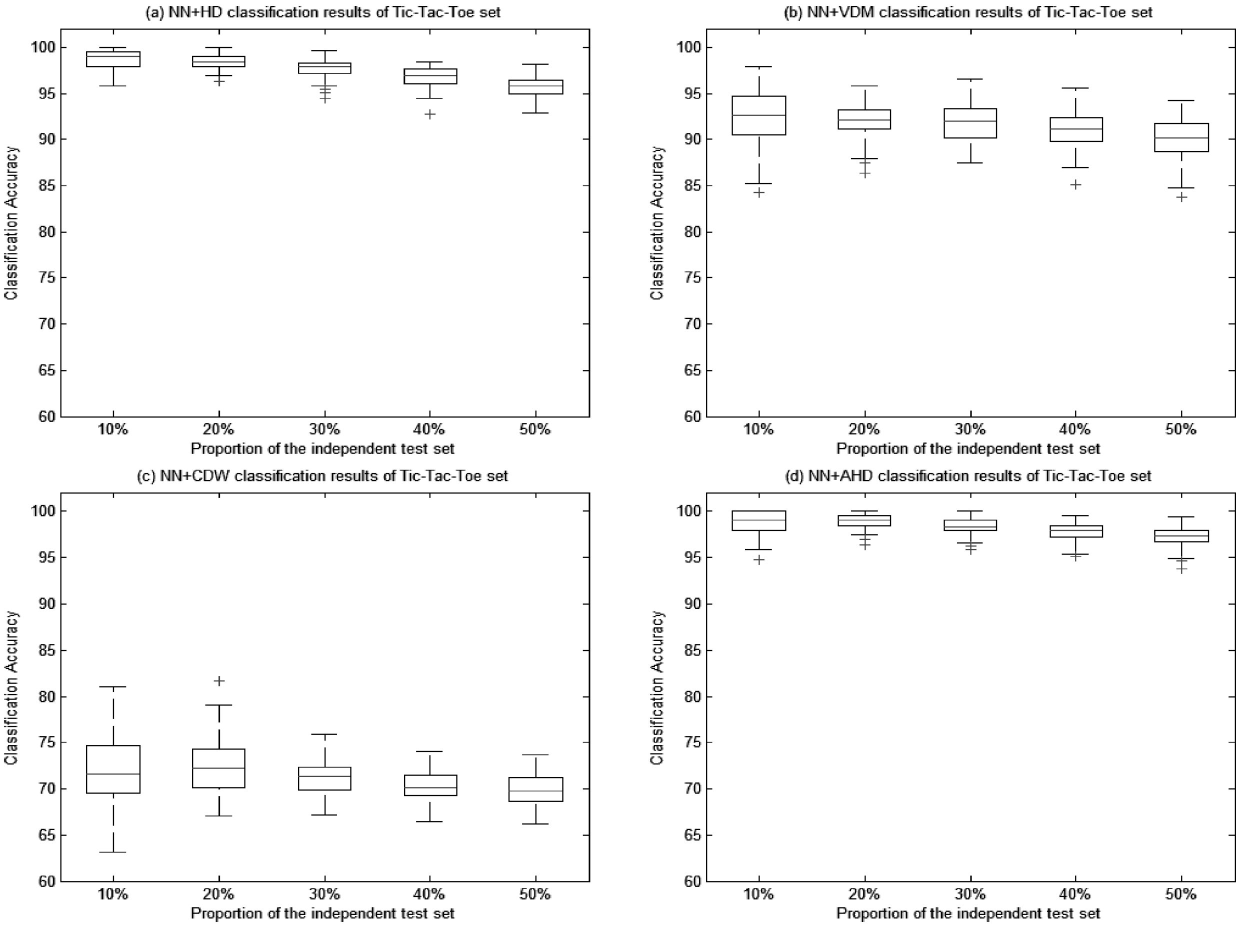

Fig. 2 Classification results of the Tic-Tac-Toe data set using NN+HD(a), NN+VDM(b), NN+CDW(c), and NN+AHD(d).

between categorical attribute values, whereas CDW assigns the weight values to attributes of each class to improve classification performance. In this letter, we presented the Adoptive Hamming Distance that adapts the distance 1 to 0 when two different categorical values are regarded as equivalent. NN+AHD showed better classification results for the given data sets compared to conventional classification algorithms, indicating the potential of the proposed approach.

Acknowledgements

This research was supported by the Chung-Ang University Research Scholarship Grants in 2009.

References

[1] D.R. Wilson and T.R. Martinez, "Improved heterogeneous distance functions," Journal of Artificial Intelligence Research, vol.6, no.1, pp.1–34, June 1997.

[2] R. Paredes and E. Vidal, "A class-dependent weighted dissimilarity measure for nearest neighbor classification problems," Pattern Recognit., vol.21, no.12, pp.1027–1036, Nov. 2000.

[3] A. Asuncion and D.J. Newman, "UCI machine learning repository [http://www.ics.uci.edu/mlearn/MLRepository.html]," irvine, CA: University of California, School of Information and Computer Science, 2007.