Multiple Classifier System for Activity Recognition

Yongkoo Han[a], Sungyoung Lee[a], Young-Koo Lee[a] and Jae Won Lee[b]

[a] *Department of Computer Engineering, Kyung Hee University*
1, Seocheon-Dong, Giheung-Gu, Yongin-si, Gyeonggi-do, 446-701, Korea
Tel: +82-31-201-3732, Fax: +82-31-202-3706, E-mail: {ykhan, sylee, yklee}@khu.ac.kr

[b] *School of Industrial Management, Korea University of Technology and Education*
307 Gajun-ri, Byungchun, Chonan Chungnam 330-708, Korea
Tel: +82-42-560-1441, E-mail: jwlee@kut.ac.kr

Abstract

Nowadays, activity recognition becomes a hot topic in context-aware computing. In activity recognition, machine learning techniques have been widely applied to learn the activity models from labeled activity samples. Most of the existing work uses only one learning method for activity learning and is focused on how to effectively utilize the labeled samples by refining the learning method. However, not much attention has been paid to the use of multiple classifiers for boosting the learning performance. In this paper, we use two methods to generate multiple classifiers. In the first method, the basic learning ***algorithms*** *for each classifier are the* ***same****, while the* ***training*** *data is* ***different*** *(ASTD). In the second method, the basic learning* ***algorithms*** *for each classifier are* ***different****, while the* ***training*** *data is the* ***same*** *(ADTS). Experimental results indicate that ADTS can effectively improve activity recognition performance, while ASTD cannot achieve any improvement of the performance. We believe that the classifiers in ADTS are more diverse than those in ASTD.*

Keywords:

Activity Recognition; Machine Learning; Multiple Classifiers

1. Introduction

Activity recognition has gained a lot of interest in recent years due to its potential and usefulness in context-aware computing such as aged care monitoring [1] and smart homes [2]. Basically, the purpose of activity recognition is to infer people's behaviors from low-level data acquired through sensors in a given setting, with which other critical decisions are made. For instance, in smart home environments for aged care monitoring [2], the system needs to automatically monitor the occupants and determine the time to raise the alarm when the occupants need assistance. The determination is based on the information provided by cameras and other pervasive sensors.

There are several ways to acquire human' activities using sensor systems. These methods include, but not limited: (1) remotely observe the scene using audio, visual, electromagnetic field, or other sensors and interpret the signal readings [3][4][5], (2) attach sensors to the body and interpret the signal readings [6][7][8], (3) attach sensors to objects and devices in the environment and interpret the sensor readings [9][10].

For different activity recognition systems, they may use various approaches mentioned above to acquire activity information. However, machine learning is always a key aspect in these systems. To automatically infer what activity is being performed, a system must have a detailed model of the activity. Currently a variety of machine learning methods have been proposed for activity recognition, such as neural networks [11], dynamic Bayesian networks [12], naïve Bayesian networks [13], hierarchical hidden semi-Markov models [14], nearest neighbors [8], decision tree [8] and so on.

However, most of the methods proposed above are used individually for activity recognition, and to achieve good recognition performance, a lot of work should be done to optimize the individual classifier. This optimization is usually complex and data-dependent. In this paper, we intend to use multiple classifiers for activity recognition. It has been proved in [15] that multiple classifiers can significantly improve the generalization ability of single learner.

The main advantage of our work is to avoid the complex optimization process in simple classifier by using multiple classifiers. We will briefly explain the basic multiple classifiers and the concept of diversity, which is the most important concept in multiple classifier systems (MCS) in Section 2. In addition, our multiple classifier construction has proved that multiple classifiers can have better performance than individual classifier. In our experiment we use two methods to generate multiple classifiers to keep diversity. In the first method ASTD, the basic learning algorithms for each learner are the same, while the training data for them is different. In the second method ADTS, the basic learning algorithms for each learner are different, while the training data is the same. The results of our experiment are shown in Section 3 and discussed in Section 4. Finally, we conclude our work in Section 5.

2. Multiple Classifier System

MCS is usually built in two steps. The first step is to generate multiple component classifiers and the second step is to combine their predictions. According to the way to generate component classifiers, current MCSs fall into two categories, i.e., algorithms that generate component classifiers in parallel [16] and algorithms that generate component classifiers in sequence [17]. In this work, we focus on the MCS in parallel because they usually require less training time compared with sequential MCSs. This is very important since in many activity recognition systems, training is required for each individual person.

2.1 Diversity in Multiple Classifier System

Each component classifier in MCS will be trained firstly. Then their output will be combined to predict the label of new examples. It is intuitively clear that an ensemble of identical classifiers will be no better than a single member thereof. If we have "the perfect classifier", then no ensemble is needed. If the ensemble members are imperfect, they should be different so that at least some of them are correct where the others are wrong.

Much work has shown that the diversity between each component classifier plays an important role in MCS. If the diversity is not enough, ensemble might not improve the generalization ability.

Let $C_i = A_i(T_i)$ denote the i_{th} component classifier. It is trained based on algorithm A_i and training data T_i. Let $C_j = A_j(T_j)$ denote the j_{th} component classifier. To achieve $C_i \neq C_j$, $A_i(T_i) \neq A_j(T_j)$, at least two methods can be used.

1) $A_i = A_j$ and $T_i \neq T_j$. We term this method ASTD (algorithms are the same, training data are different)

2) $A_i \neq A_j$ and $T_i = T_j$. We term this method ADTS (algorithms are different, training data are the same).

The details of ASTD and ADTS are given in Section 2.2 and 2.3 respectively.

2.2 ASTD

The framework of ASTD is illustrated in Fig. 1. In this example, decision tree is the basic classification method, though conceptually any classification method (e.g. Naive Bayes) can be substituted used. Each decision tree in Fig. 1 is trained using the training instances for that decision tree. As mention in Section 2.1, when the algorithms are same for each component classifier, the training data must be different in order to generate diversities between them.

For that point, the method we use is bootstrap [18]. Each classifier's training set is generated by randomly drawing, with replacement, N examples – where N is the size of the original training set; many of the original examples may be repeated in the resulting training set while others may be left out. Each individual classifier in the ensemble is generated with a different random sampling of the training set.

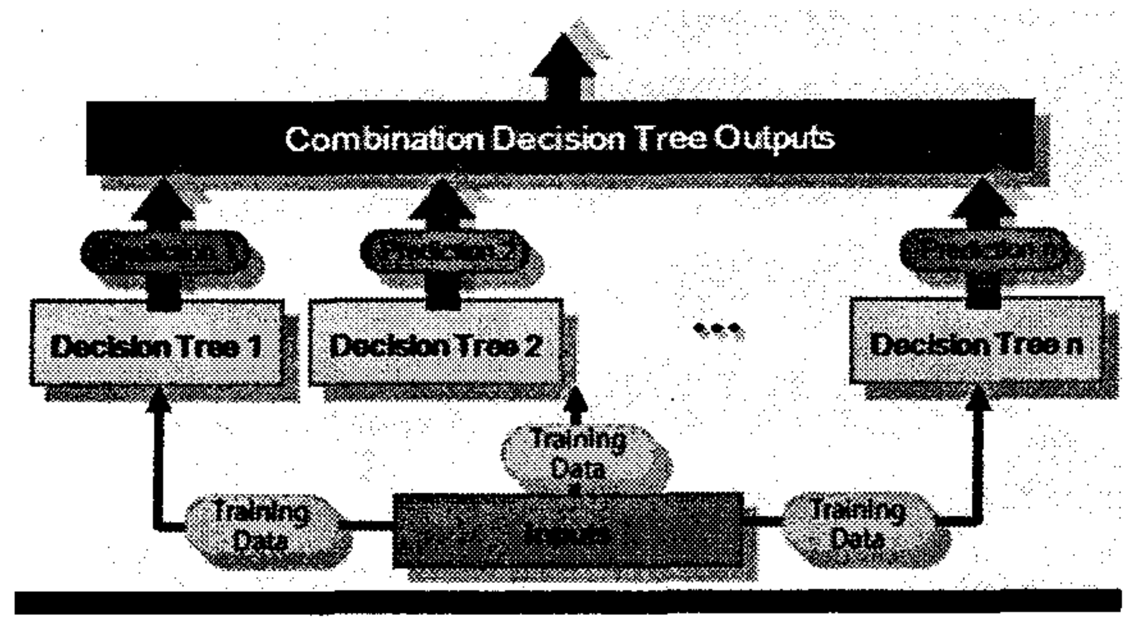


Figure 1 - Framework of ASTD

Figure 2 gives a sample of how bootstrap might work on an imaginary set of data. Since bootstrap resamples the training set with replacement, some instances are represented multiple times while others are left out. So bootstrap's training-set-1 might contain examples 3 and 7 twice, but contains neither example 4 nor 5. As a result, the classifier trained on training-set-1 might obtain a higher test-set error than the classifier using all the data. In fact, all of these four component classifiers can result in higher test-set error; however, when combined, these four classifiers can (and often do) produce test-set error lower than that of the single classifier (the diversity among these classifiers generally compensates for the increase in error rate of any individual classifier).

A sample of a single classifier on an imaginary set of data
Original training data
Training-set-1: 1, 2, 3, 4, 5, 6, 7, 8

A sample of Bootstrap on the same data
Resampled training data
Training-set-1: 2, 7, 8, 3, 7, 6, 3, 1
Training-set-2: 7, 8, 5, 6, 4, 2, 7, 1
Training-set-3: 3, 6, 2, 7, 5, 6, 2, 2
Training-set-4: 4, 5, 1, 4, 6, 4, 3, 8

Figure 2 - Example of training sets for each component classifier

2.3 ADTS

The framework of ADTS is illustrated in Fig. 3. In Fig. 3, the algorithms for each component classifier are different while training data sets are the same. In fact, to make each classifier different, the training sets are not necessarily to be same. In ADTS, we use the original training set for all the classifiers because original training set is usually better than the processed one such as bootstrap.

Compared with ASTD, ADTS is easier to implement since it is not required to generate different training sets. After each component classifier is trained, their combination will be used to predict the new unseen instances.

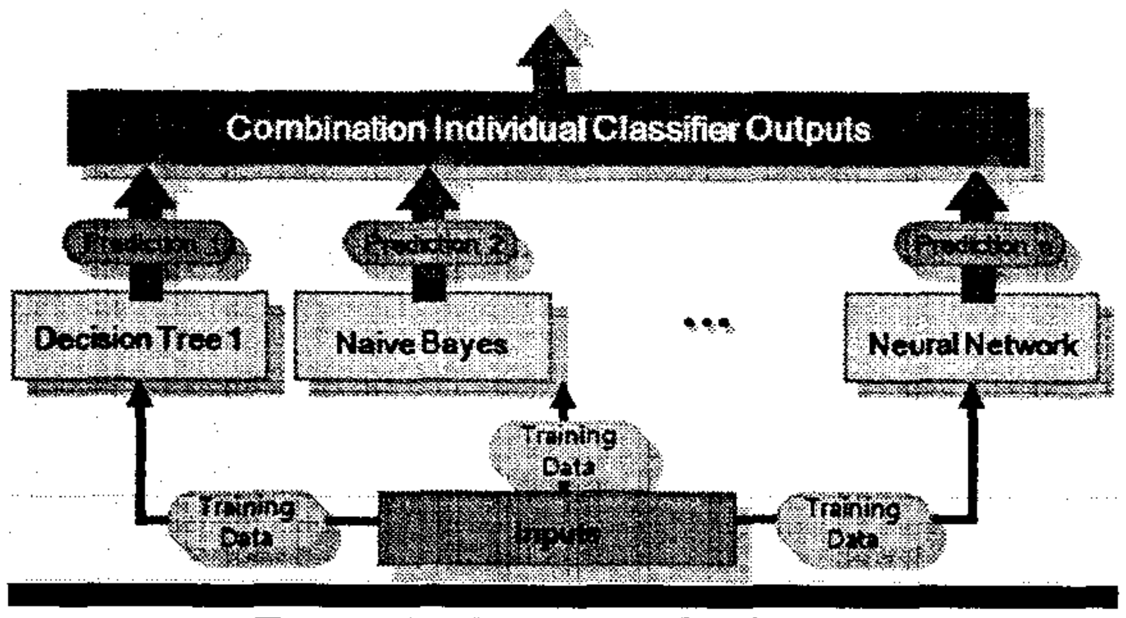


Figure 3 - Framework of ADTS

3. Experimental Result

For our experiments we use a dataset published in [19]. It consists of 10 basic activities, namely Lying, Kneeling, Sitting, Standing, Walking, Running, Climbing Stairs, Descending Stairs, Bicycling and Jumping. The activities were recorded by 40 accelerometers strapped loosely to common trousers, 20 sensors per leg, starting from the ankle to the hip.

The original data set includes 25177 data samples and 9 activities (1 activity is missing in the dataset). For each activity, we choose the first 500 samples. Hence totally 4500 data samples are used in our experiment.

For the activity data sets, 70 percent are kept as test examples while the rest are used as the pool of training examples, i.e., $L \cup U$ (labeled examples and unlabeled examples). It is noted that only L is used for training. In each pool, L and U are partitioned under different label rates including 10 percent, 20 percent, 30 percent and 40 percent. For instance, assuming a pool contains 1000 examples, when the label rate is 20 percent, 200 examples are put into L with their labels while the remaining 800 examples are put into U without their labels.

J4.8 decision trees, Naïve Bayes classifier and K-nearest neighbors (K=3) are used in the experiments. Under each label rate, ten independent runs with different random partitions of L and U are performed.

Experiment includes two parts. In the first part, different classifiers with same training data are used. In each run, the best performance among the three classifiers are shown in bold and be compared with their ensemble. In the second part, same classifiers with bagging are used. In each run, the ensemble is compared with the single classifier.

3.1 ADTS

We simulate three kinds of classifiers mentioned above. Figure 4 shows result for decision tree for various label rates. As shown in this figure, when label ratio is 10 percent, the best individual classifier's error rate is 17.4% while their ensemble's error rate is 17.0%. Totally 1.57% improvement is achieved. As shown in the Table 1, this average improvement is reported by averaging the ten run's improvement. At each run, the improvement is calculated by using ensemble to compare the best classifier. The relationship between label ratio and amount of improvement is given in the following figure. As shown in this figure, improvement is achieved under all the experimental label ratios. In addition, with the label ratio increasing, more improvement is achieved. Especially in the case when label ratio is 10%, there is only 1.57% improvement achieved. One possible reason is that when label ratio is small, there is not enough diversity between each component classifiers even though different learning algorithms are used.

Table 1 – an example experiment result (Error rates when label ratio is 30 percent)

Runs	DT	KNN	NB	Ensem.	Imp.
1	0.176	**0.157**	0.168	0.136	13.4%
2	0.153	0.139	**0.136**	0.122	10.3%
3	0.183	**0.156**	0.165	0.139	10.9%
4	0.158	**0.148**	0.152	0.119	19.6%
5	**0.156**	0.157	0.163	0.129	17.3%
6	**0.148**	0.152	**0.148**	0.115	22.3%
7	0.160	0.146	**0.137**	0.135	1.46%
8	0.160	0.156	**0.150**	0.121	19.3%
9	0.185	0.145	**0.142**	0.116	18.3%
10	0.157	**0.138**	0.152	0.114	17.4%
Ave.	0.164	0.149	0.151	0.125	15.0%

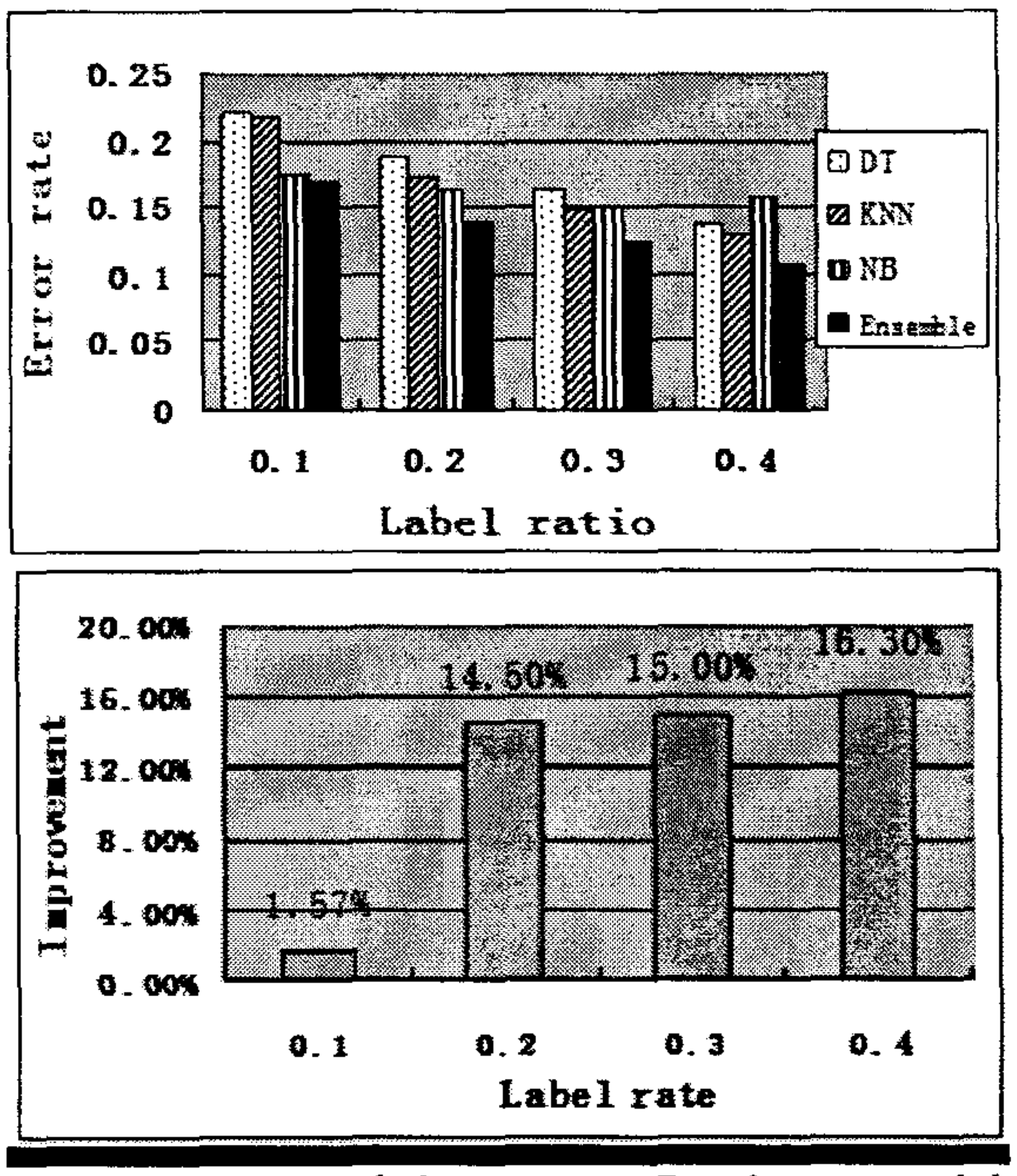
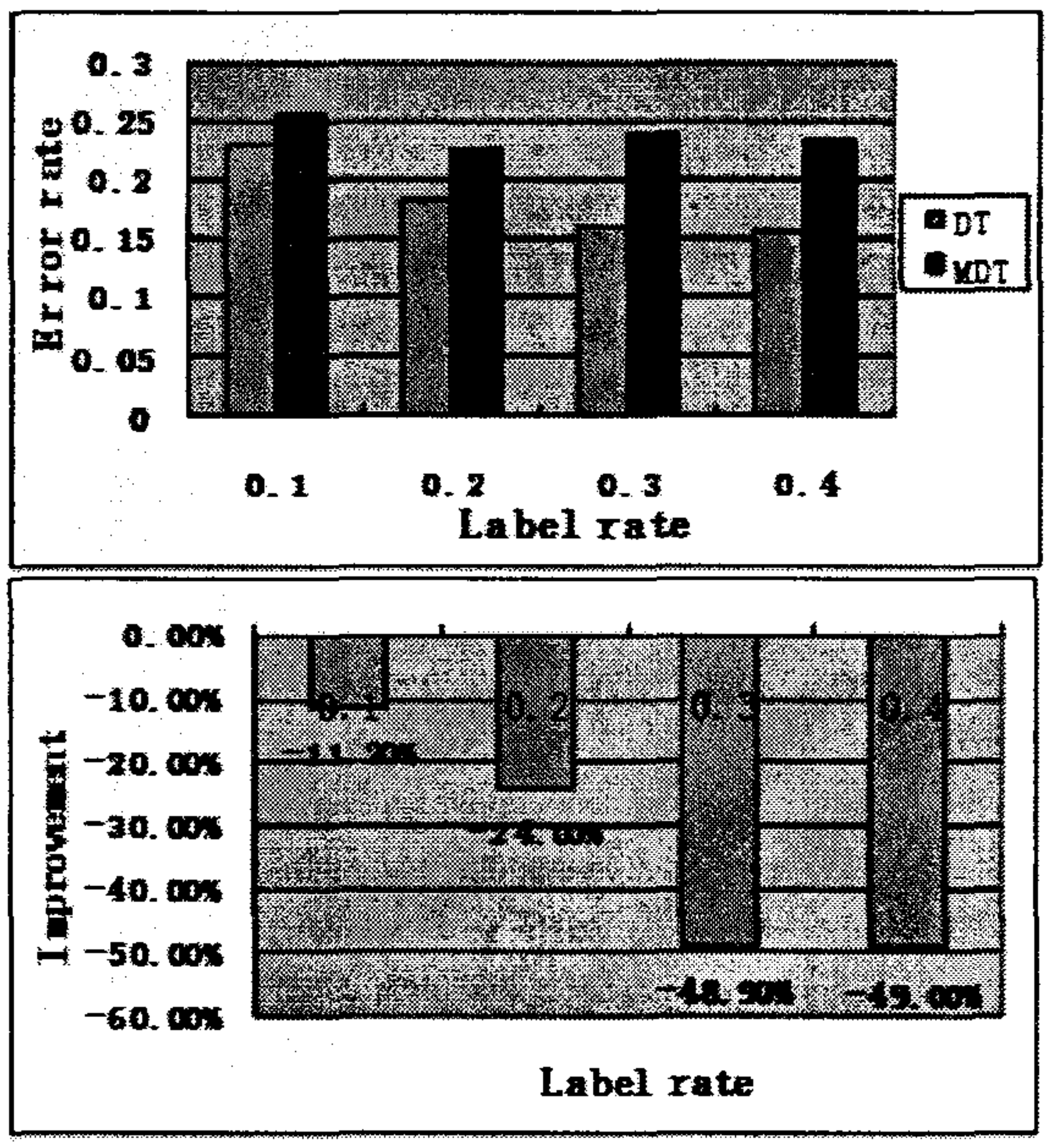


Figure 4 - ADTS result for Decision Tree for various label rates

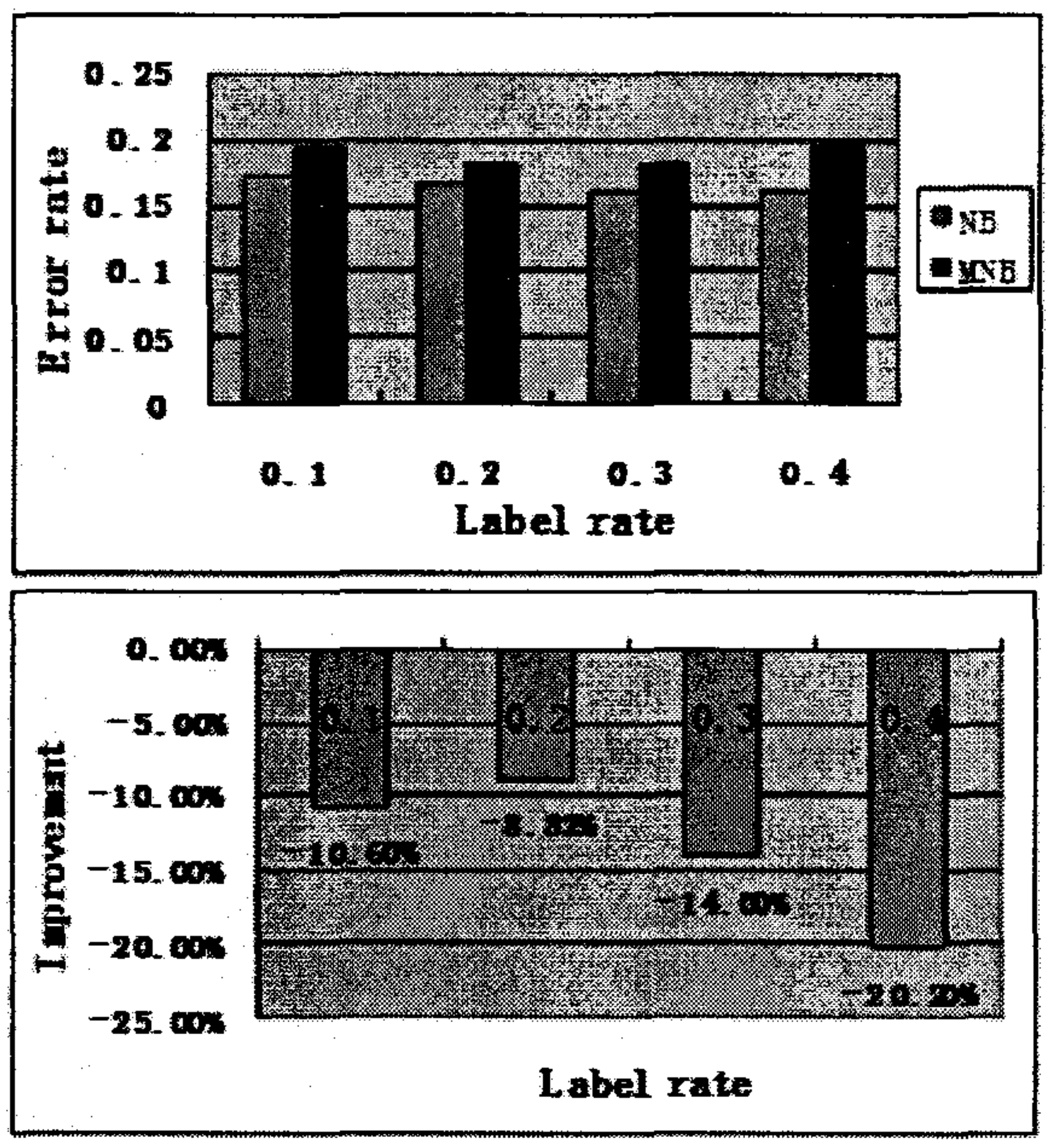
3.2 ASTD

In this part, each component classifier utilizes the same base algorithm but different training data by bootstrap.

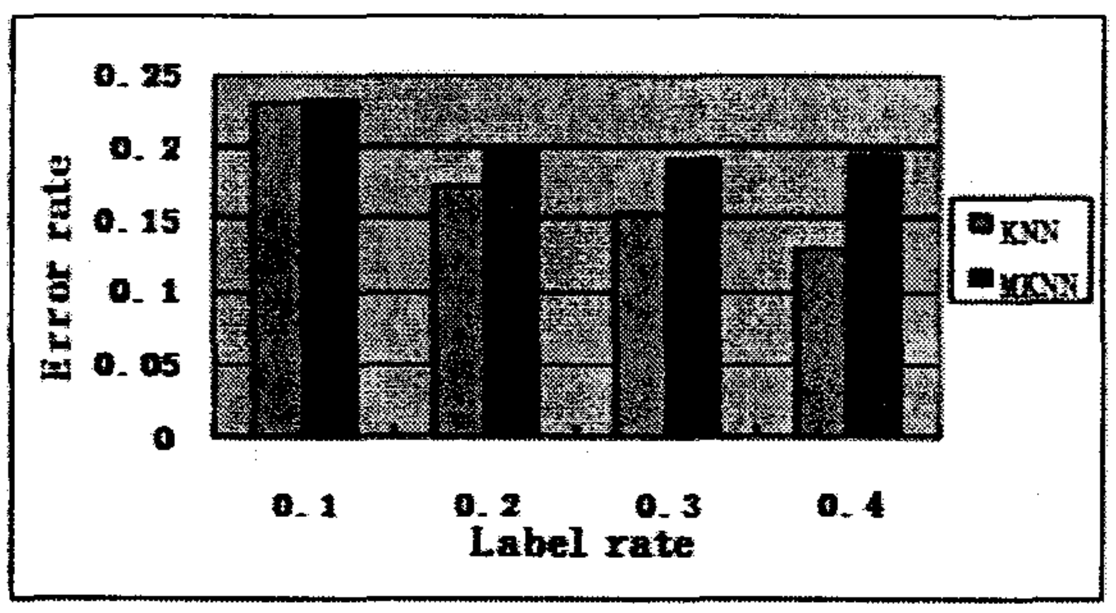
As shown in Figure 5.a, 5.b, 5.c, all ensembles of three classifiers works worse than individual classifier under all the four label ratios. Based on Figure 5, we also found that with the label ratio increasing, ensemble classifier works worse than individual classifiers. When the label ratio is 40%, the improvements are -49%, -20.20%, -51.50% for decision tree, naïve bayes and three k-nearest neighbors respectively.

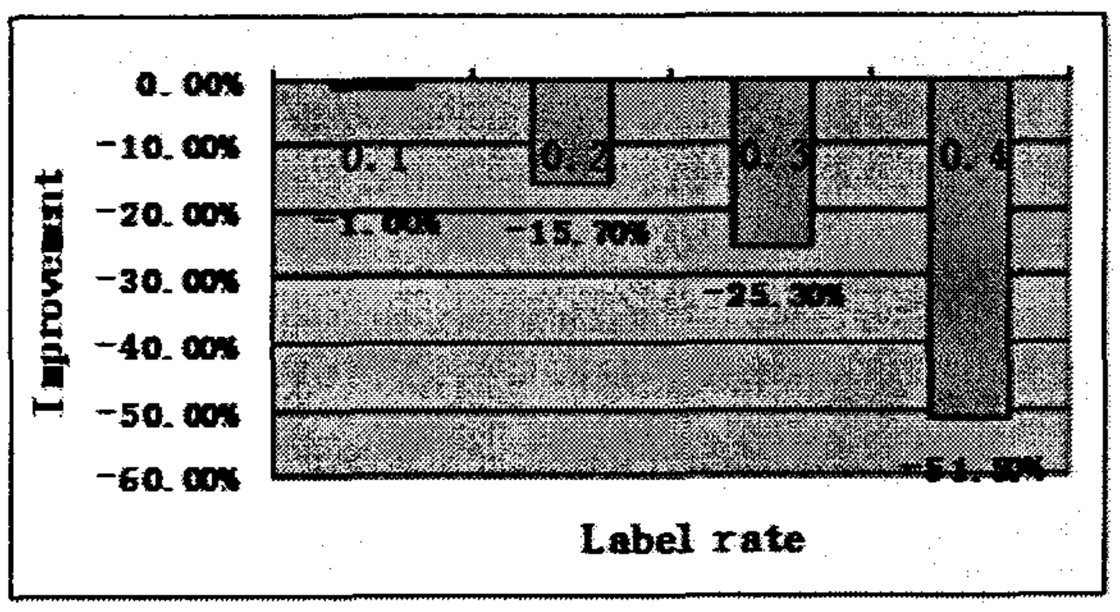


a. ASTD result for Decision Tree for various label rates



b. ASTD result for Naïve Bayesian for various label rates





c. ASTD result for KNN for various label rates

Figure 5 - ASTD result for three individual classifiers for various label rates

4. Discussions

The experimental results in Section 3 show that ADTS achieves good improvement; while ASTD fails to make any improvement compared with individual classifier.

As table 1, in each run, different individual classifier has the best performance. This result means the performance of the individual classifier depends on the training data set. However ensemble classifier has almost consistent performance in the all runs. In addition the performance of ensemble is always higher than all of the individual ones.

Most of the research work shows that the success of a classifier ensemble is that the basic classifiers perform diversely. Hence, the explanation for our result is that ADTS generates more diversity between each component classifier compared with ASTD. For ADTS, the diversity is determined by the difference between each base learning algorithm. For ASTD, the diversity is determined by the difference between each training set. Moreover, it is determined by the base learning algorithms. For example, some work has shown that ASTD does work well for stable base learning algorithms such as naïve bayes and k-nearest neighbors.

For the data, if most samples are similar enough, resampling cannot achieve enough diversity. In addition, this resampling may remove some important samples.

5. Conclusion and Future works

Activity recognition plays an important role in context-aware computing. Currently many machine learning algorithms have been proposed for activity recognition. However, most of them are required to be well optimized and devised as they are used individually.

To overcome the complex optimization for individual algorithm, we propose to combine multiple simple algorithms. Our point is although these algorithms are simple, their combination might create good result.

To generate diversity between individual classifiers, two ensemble methods are used: ASTD (algorithms are same, training data are different) and ADTS (algorithms are different, training data are same).

A set of experimental results show that ASTD works better than ADTS when using our activity data. This is because the diversity of ASTD is more than ADTS.

In the future, we will build our activity recognition system and further testify our method based on it.

Acknowledgments

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2006-C1090-0602-0002).

References

[1] V. Stanford, "Using pervasive computing to deliver elder care", *IEEE Pervasive Computing*, 2002, pp. 10-13.

[2] C.D. Kidd, R. Orr, G.D. Abowd, C.G. Atkeson and I.A. Essa, "The aware home: A living laboratory for ubiquitous computing research", *Proceedings of the Second International Workshop on Cooperative Buildings*, 1999, pp. 191-198.

[3] S.S. Intille, J. Davis, and A. Bobick, "Real-time closed-world tracking", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1997, pp. 697-703.

[4] S. Stillman, R. Tanawongsuwan, and I. Essa, "A system for tracking and recognizing multiple people with multiple cameras", *Proceedings of the Second International Conference on Audio-Vision-based Person Authentication*, 1999.

[5] I. Haritaoglu, D. Harwood, and L. Davis, "W4: Who, When, Where, What: A real time system for detecting and tracking people", *Proceedings of the Third International Conference on Automatic Face and Gesture*, Nara, 1998, pp. 222-227.

[6] M. Makikawa and H. Iizumi, "Development of an ambulatory physical activity monitoring device and its application for categorization of actions in daily life. *MEDINFO*, 1995, pp. 747–750.

[7] K. Aminian, P. Robert, E. Jequier, and Y. Schutz, "Estimation of speed and incline of walking using neural network", *IEEE Transactions on Instrumentation and Measurement*, 1995, pp. 743–746.

[8] L. Bao, "Physical activity recognition from acceleration data under seminaturalistic conditions", *M.Eng thesis*, EECS, Massachusetts Institute of Technology, 2003.

[9] G.D. Abowd, "Director of the AwareHome initiative", Georgia Insitute of Technology, November 2002.

[10] T. Barger, M. Alwan, S. Kell, B. Turner, S. Wood, and A. Naidu, "Objective remote assessment of activities of daily living: Analysis of meal preparation patterns", Poster presentation, Medical Automation Research Center, University of Virginia Health System, 2002.

[11] M. Mozer, "The neural network house: an environment that adapts to its inhabitants", *Proceedings of the AAAI Spring Symposium on Intelligent Environments*, Technical Report SS-98-02, AAAI Press, Menlo Park, CA, 1998, pp. 110–114.

[12] K. P. Murphy, "Dynamic Bayesian Networks: Representation, Inference and Learning", PhD thesis, University of California, Berkeley, 2002.

[13] S.S. Intille and A.F. Bobick, "Recognizing planned, multi-person action", *Computer Vision and Image Understanding*, 2001, pp. 414–445.

[14] H. Kautz, O. Etziono, D. Fox, and D. Weld, "Foundations of assisted cognition systems", technical report cse-02-ac-01, University of Washington, Department of Computer Science and Engineering, 2003.

[15] T. G. Dietterich, "Ensemble learning," in The Handbook of Brain Theory and Neural Networks, 2nd edition, M. A. Arbib, Ed., Cambridge, MA: MIT Press, 2002.

[16] L. Breiman, "Bagging predictors," Machine Learning, vol.24, no.2, pp.123-140, 1996.

[17] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of online learning and an application to boosting," in Proceedings of the 2nd European Conference on Computational Learning Theory, Barcelona, Spain, 1995, pp. 23-37.

[18] Efron, B., & Tibshirani, R. (1993). An Introduction to the Bootstrap. Chapman and Hall, New York.

[19] K. Van Laerhoven and H.-W. Gellersen , "Spine versus Porcupine: a Study in Distributed Wearable Activity Recognition", *Proceedings of the eighth International Symposium on Wearable Computers*, 2004, 142-149.