5-16-2003

Authoritative Citation KNN Learning with Noisy Training Datasets

Leen-Kiat Soh
University of Nebraska at Kearney, lsoh2@unl.edu

Joseph Bernadt
University of Nebraska, jbernadt@cse.unl.edu

University of Nebraska-Lincoln, Computer Science and Engineering
Technical Report # TR-UNL-CSE-2003-0003 05/16/2003

Authoritative Citation KNN Learning with Noisy Training Datasets

Leen-Kiat Soh LKSOH@CSE.UNL.EDU
Joseph Bernadt JBERNADT@CSE.UNL.EDU
Department of Computer Science and Engineering, University of Nebraska, Lincoln, NE 68588-0115 USA

Abstract

In this paper, we investigate the effectiveness of Citation K-Nearest Neighbors (KNN) learning with noisy training datasets. We devise an authority measure associated with each training instance that changes based on the outcome of Citation KNN classification. The authority is increased when a citer's classification had been right; and vice versa. We show that by modifying only these authority measures, the classification accuracy of Citation KNN improves significantly in a variety of datasets with different noise levels. We also identify the general characteristics of a dataset that affect the improvement percentages. We conclude that the new algorithm is able to regulate the roles of good and noisy training instances using a very simple authority measure to improve classification.

1. Introduction

According to (Aha, 1997), purely lazy learners typically display the following characteristics:

1. Defer: They delay the processing of their inputs until they receive requests for information,

2. Demand-Driven: They reply to information queries by combining information from their stored (e.g., training) instances, and

3. Discard: They delete the constructed query and any intermediate results.

When the training dataset is not noisy, purely lazy learners classify well and do not generally suffer from "wasting" the intermediate results. On the other hand, when the training dataset is noisy, it is possible for the query feedback to modify the training instances in some degree to improve future classification. This latter approach is akin to reinforcement learning (Sutton and Barto, 1998). Our research goal is to obtain a hybrid lazy learner that tackles noisy training datasets by incorporating performance feedback in a supervised manner.

We use K-Nearest Neighbors (KNN) as the basis of our lazy learner. Briefly, a KNN classifier labels a test instance as the most popular label among the test instance's *K* nearest neighbors. Specifically, we focus on a version of KNN called Citation KNN (Wang and Zucker, 2000). The idea of using citations and references as related documents to a given paper was outlined in (Garfield, 1979). Basically, if a research paper cites another previously published paper (as its reference), the paper is said to be related to that reference. Similarly, if a paper is cited by a subsequent article (as its citer), the paper is also said to be related to its citer. Exploiting this mutuality in KNN generally improves the classification accuracy (Wang and Zucker, 2000; Amar et al., 2001).

We propose a hybrid Citation KNN learner by adding an authority measure to each training instance. This authority value increases when the training instance helped classify a test instance correctly and decreases otherwise. We name this Authoritative Citation KNN (ACKNN). The basic idea is intuitive: A paper that has been cited more frequently and more correctly will have a higher authority. In this ACKNN design, we determine "correctness" in a supervised manner—we tell the learner whether its previous classification was correct. That triggers an adjustment process to the nearest neighbors that had helped classify the test instance in question. The citers that had been right will be given more authority; those wrong will be given less authority. If we fed in the same test instance enough times, for example, gradually, citers with low authority values would be ignored.

Our research objectives are to (1) study the correlations between the features of a dataset and the classification results of our hybrid learner, (2) examine the roles of *C* (the number of citers) and *K* (the number of nearest neighbors) in ACKNN, (3) analyze the performance of ACKNN in datasets of different noise levels, and (4) investigate how a training instance changes its authority values when the learner is reinforced positively and negatively. Our approach is single-instance learning as opposed to multiple-instance learning (Wang and Zucker, 2000; Amar et al., 2001).

In the following, we first discuss some related work to our research. In Section 3, we present the methodology of ACKNN. Then, we describe the experiments and discuss the results. Finally we conclude and outline some future work.

2. Related Work

Lazy learning, also known as memory-based learning, methods defer processing of training data until a query needs to be answered. Atkesan, Moore, and Schaal (1997) identified two forms of lazy learning. One form of lazy learning finds the nearest neighbors and selects or votes on the predictions made by each of the stored points. Another form of lazy learning, *locally weighted learning*, uses locally weighted training to average, interpolate between, extrapolate from, or otherwise combine training data. Nearest neighbor local models simply choose the closest point and use its output value. Weighted average local models average the outputs of nearby points, inversely weighted by their distance to the query point. Our ACKNN approach is a mixture of the two forms.

Aslam and Decatur (1996) studied the sample complexity of noise-tolerant learning. The standard Probably Approximately Correct (PAC) model assumes that the data presented to the learner is noise-free. Most of the standard PAC learning algorithms would fail if even a small number of the labeled examples given to the learning algorithm were "noisy" (Aslam and Decatur, 1996).

In dealing with noisy training datasets, several approaches have been proposed. Instances that are suspected to be unreliable are downweighted in (Aha and Kibler, 1990; Cost and Salzberg, 1993). Cross-validations are used to measure such reliability. Important issues in this area include the timing and the frequency of the weighting decision. Wettscherecck, Aha, and Mohri (1997) provided a review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. Methods which use performance feedback to assign weight setting demonstrated three advantages over other methods: they require less pre-processing, perform better in the presence of interacting features, and generally require less training data to learn

good settings. This provides a justification for our ACKNN design, which is also hinged upon the performance feedback of the learner.

There has been research work in multiple-instance learning where each data point is not an instance, but a bag of instances (Dietterich, Lathrop, and Lozano-Perez, 1997). Each bag assumes a label (or classification) but contains a bag of diverse instances with different characteristics. The learning process is to determine to which bag a test instance belong (by comparing the test instance to each of the instances in the bags). Maron and Lozano-Pérez (1998) described a new general framework called Diverse Density based on the assumption that the desired concept—the point of intersection of the positive bags minus the union of the negative bags—can be acquired by maximizing diverse density. Wang and Zucker (2000) further devised two general-purpose algorithms—Bayesian KNN and Citation KNN and reached the best results obtained so far by ad-hoc, multiple-instance learning algorithms on a drug discovery task. Amar et al. (2001) investigated various extensions of KNN, Citation KNN, and the diverse density algorithms for the real-valued setting and studied their performance on Boolean and real-valued data in multiple-instance learning. The performance of both the nearest neighbor and diverse density algorithms were found to be very sensitive to the number of relevant features. Our current work is based on Citation KNN, but on single-instance learning.

3. Authoritative Citation KNN Algorithm

Our Authoritative Citation KNN (ACKNN) algorithm is:

Start Algo

(1) Read in training dataset

(2) Initialize every training instance's authority

(3) Read in the query of test instance, TEI

(4) Find all C-best citers of TEI

(5) Find all K-nearest neighbors of TEI

(6) Perform ACKNN to classify TEI

(7) Feedback and adjustments

End Algo

The first step is straightforward. Each dataset is a text file and our program simply reads it in, parses each line, and stores the training instances in arrays. Each training instance's authority value is initialized to 0.5. When the system is given a test instance TEI, it performs ACKNN to classify it. By comparing the actual, true classification of TEI and the result of ACKNN, the system modifies the authority values of affected citers and neighbors. We will discuss the other steps (finding C-best citers, finding all K-nearest neighbors, perform ACKNN, and adjustments) in the following.

3.1 Finding C-Best Citers

Let $l \in L$ be a member of the set of all labels (classes) L. This step goes through all training instances in the dataset and locates the $|C_l|$ best citers for each $l \in L$ in terms of Euclidean distance,

with respect to the test instance TEI. A citer is a training instance that would cite the test instance as one of its nearest neighbors. At the end of this process, there are at most a total of $|C_l| \cdot |L|$ best citers. In a very dense and juxtaposed training dataset, $|C_l| \cdot |L|$ is large. In a very sparse and well organized (minimally interacting) one, $|C_l| \cdot |L|$ is small. By considering $|C_l| \cdot |L|$ instead of just $|C_l|$, we propagate the changes faster with respect to the number of test instances to citers. In our design, all values of $|C_l|$ are set to *C*.

3.2 Finding K-Nearest Neighbors

This step follows the traditional KNN process. The program locates the *K* nearest neighbors based on distance.

3.3 Performing ACKNN Classification

This step performs the classification of the test instance TEI. Suppose that $l \in L$, and that $c \in C_l$ is a member of the best citers of label *l* for test instance TEI, and $k \in K$ is a member of the nearest neighbors for test instance TEI. We then compute:

$$vote(l) = vote\_k(l) + vote\_c(l)$$

where $vote\_k(l) = \sum_{k \in K} f(k,l)$, where $f(k,l)$ is 0.5 when the label of *k*, $label(k)$, equals *l*; where $vote\_k(l) = \sum_{c \in C_l} Authority(c)$ where $Authority(c)$ is the authority of *c*. Thus, in the above setup, the Euclidean distance and the authority play equal parts. Given $vote(l)$ values for all $l \in L$, this step finally labels TEI with:

$$label(TEI) = \arg\max_{l} vote(l).$$

3.4 Feedback-Based Adjustments

Once the system classifies the TEI, the feedback tells the system whether it was a correct classification. Based on this feedback, the system modifies the authority values of the citers used to classify the TEI. If $c \in C_{actual}$, where C_{actual} is the set of all citers for TEI, had been correct ($label(c) = l_{correct}$), then we increase its authority, $Authority(c,l)$, by $1/|C_{actual}|$. Otherwise, we decrease its authority by the same amount. We clamp the range of authority values between 0 and 2.0.

4. Experiments and Discussions of Results

Here we discuss our experiments and analyze the results. Our experiments were designed to (1) study the correlations between the features of a dataset and the classification results of our hybrid learner, (2) examine the roles of *C* (the number of citers) and *K* (the number of nearest neighbors) in ACKNN, (3) analyze the performance of ACKNN in datasets of different noise levels, and (4) investigate how a training instance changes its authority values when the learner is reinforced positively and negatively. Our experiments on noisy training datasets used the *classification noise* model (Angluin and Laird 1988) where each training instance received by the learner is mislabeled randomly with some fixed probability less than 0.5.

4.1 Datasets

We downloaded six datasets from the UCI Machine Learning Repository (maintained by the University of California, Irvine, Dept. of Information and Computer Sciences): PIMA, ABALONE, LIVER, WINE, WPBC, and PENDIGITS, with general characteristics shown in Table 1.

Table 1. General characteristics of the datasets in our experiments.

DATA SET	TRAINING SET	TEST SET	#FEATURES	#CLASSES
PIMA	576	192	8	2
ABALONE	3133	1044	8	29
LIVER	257	88	6	2
WINE	150	27	13	3
WPBC	482	87	30	2
PENDIGITS	7494	3498	16	10

Here are some brief details on these datasets (excerpted from the README files of the datasets).

PIMA: This Pima Indians Diabetes Database contains features of patient to diagnose for diabetes, according to World Health Organization criteria.

ABALONE: This dataset contains physical measurements of abalones that together may be used to predict the age of abalones.

LIVER: This contains the records of male individuals with possible liver disorders.

WINE: This is a wine recognition data. The data was used with many others for comparing various classifiers. Using a leave-one-out technique, a 1-NN learner classified this data with 96.1% accuracy.

PENDIGITS: This is a database for Pen-Based Recognition of Handwritten Digits. There are 250 samples from 44 writers. The samples written by 30 writers are used for training cross-validation and writer dependent testing, and the digits written by the other 14 are used for writer independent testing.

WPBC: This contains the records of Wisconsin Prognostic Breast Cancer (WPBC) cases. The first 30 features were computed from a digitized image of a fine needle aspirate of a breast mass. They describe the characteristics of the cell nuclei present in the image.

When we corrupt our training datasets, we randomly (following a uniform curve) change the classification of some training instances. For a 10% noise, we change 10% of the training instances; and so on. In the following experiments, we added 10%, 20%, 30%, 40%, and 50% noise to the datasets.

In addition, we also tested on a range of C and K values. We had K = 3, 5, and 7 and C = 0, 3, 5, and 7, for a combination of 12 ACKNN learners. When C = 0, the ACKNN learner was basically a Citation KNN learner.

4.2 Authority Values

First, we look at the relationships between the authority values and the test instances. We fed each test instance into the learner. The learner then performed one iteration of ACKNN—

classifying the test instance and then adjusting the authority values of citers based on the feedback. Figure 1 shows a typical set of curves when plotting the numbers of changed authority values against the test instances or iterations. In general, learners with a higher *C* value resulted in a higher number of changed authorities per test instance. This is expected since a higher *C* means more citers are involved in the classification process.

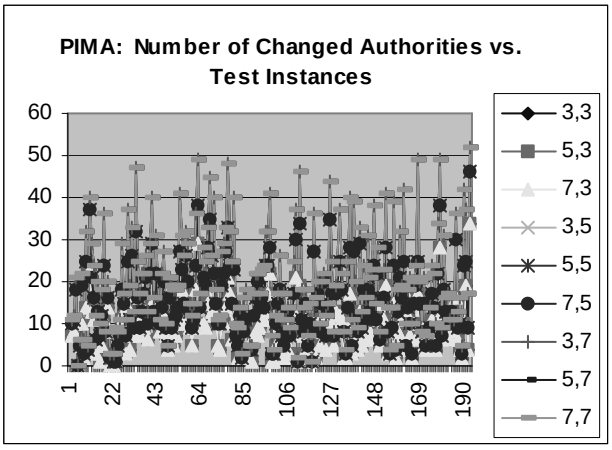

Figure 1. The numbers of changed authorities vs. test instances for the PIMA dataset. A curve of 5,3 means it is for learner with *K*=5 and *C*=3. No noise was added to the dataset.

Out of the six datasets, only one (LIVER) showed a significant trend of convergence (see Figure 2). In fact, LIVER was the only dataset that yielded a less than 1% real-time improvement in classification accuracy (Section 4.4). We performed numerous correlation tests (with the size of the training set, the size of the test set, the number of classes, the number of features, average standard deviation of the attribute values, etc.) to try to explain this observation but to no avail. When looking at the authority values, we think that this convergence was due to the clamping effect of our feedback-based adjustments (Section 3.4). Authority values reaching the maximum (or minimum) value will not be increased (or decreased) further. So we speculate that a more in-depth analysis of the feature space should give us a hint into this convergence.

The lack of convergence was due to the following factors: (1) the size of the test instances was not large enough to converge the authority values (Table 1), and (2) each test instance will bring about a change in the authority values unless the clamping effect took place.

From our experiments, we also observe that the curves for the numbers of changed authority values are very similar when the same value of *C* was used. Here is the explanation. For each test instance that we read in, every training instance that cites it can only cite it as one of two options. It can either cite it correctly or incorrectly. If it cites correctly the authority increases; if it cites incorrectly the authority decreases. Regardless, authority values will change. Now as we increase *K* but keep *C* constant, it is true that a training instance might be predicted differently than it was for the previous *K* value due to the increase in nearest neighbors that we look at. However, the exact same training instances will cite the exact same test instances for a given *C* value. So, the same number of authorities will change across different *K* values but a fixed *C* value, though for different *K* values a citer may be correct for one *K* value and incorrect for the other. But since the citer must either increase or decrease, it will still change (it may just be increased for one *K* value and decreased for the next).

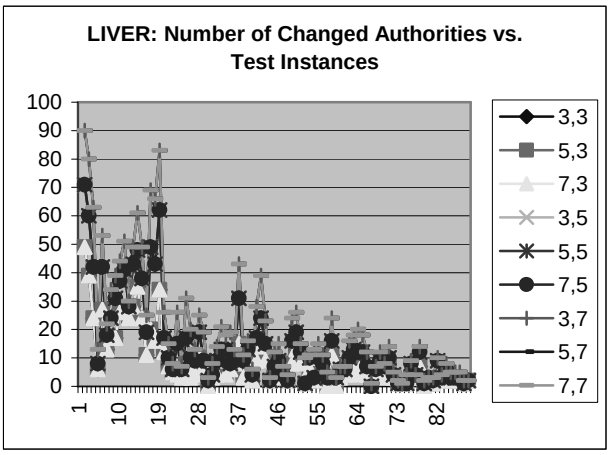

Figure 2. The numbers of changed authorities vs. test instances for the LIVER dataset. A curve of 5,3 means it is for learner with K=5 and C=3. No noise was added to the dataset.

The only time a citer's authority will not be increased further or decreased further is when it reaches a maximum or minimum authority value—these are our clamps (Section 3.4). But for a fixed C value, these will be reached at the same rate and hence the same number of authorities will change.

4.3 Good vs. Noisy Training Instances

Here we look at some typical behavior of the training instances when test instances are fed into the system. Mainly, we focus on the changes in the authority value of each instance.

Figure 3 shows the results of ACKNN on the PIMA 50%-noise data with K=3, and C=5. From the initial authority value of 0.5, the average authority value of "non-corrupted" training instances steadily increased while that of noisy training instances did the opposite. In Figure 3, there were fewer than 200 test instances with about 575 training instances, half of which were corrupted.

Figure 4 shows the results of ACKNN on the ABALONE 50%-noise data with K=3, and C=5. Figure 4 shows that the average authority value of "non-corrupted" training instances still did better than the noisy counterparts. However, the average authority value of the good training instances decreased steadily. There were about 1000 test instances with about 3000 training instances, half of which were corrupted.

We have an explanation for the discrepancy. The ABALONE dataset is inherently noisier than the PIMA dataset. This is supported by the fact that we could classify PIMA with about twice the accuracy than we could ABALONE using KNN. Thus, the reinforcement learning was not able to compensate well to promote good training instances. However, ACKNN was still able to improve the classification (Section 4.5).

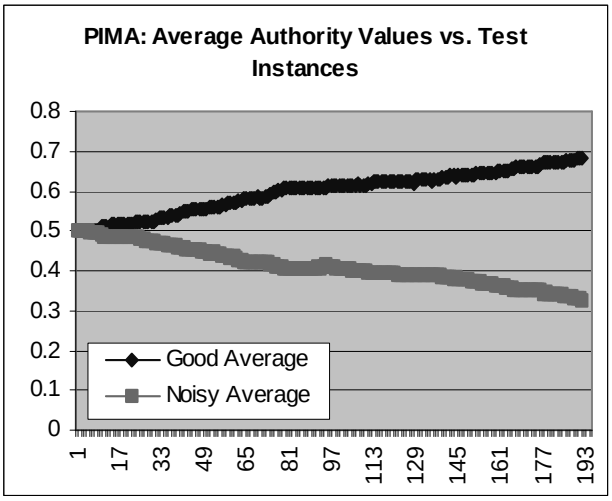

Figure 3. Average authority values vs. test instances, of good and noisy training instances for PIMA with 50%-noise, and with ACKNN of K=3, and C=5.

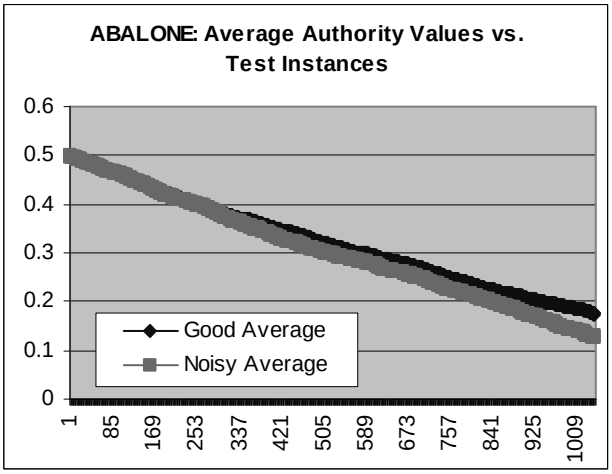

Figure 4. Average authority values vs. test instances, of good and noisy training instances for ABALONE with 50%-noise, and with ACKNN of K=3, and C=5.

We also computed the standard deviation of each training instance's authority value. Table 2 shows the values for five good training instances (G1, …, G5) and five corrupted training instances (N1, …, N5), picked at random, for PIMA and ABALONE. In fact, for all training instances, the standard deviations in authority are very similar. However, we see that the authority values of the good test instances were in general smaller that those of the corrupted test instances, for ABALONE.

Table 2. Some examples of good (G1, …, G5) and corrupted (N1, …, N5) training instances and their average and standard deviation authority values.

	G1	G2	G3	G4	G5
PIMA Average	0.72	0.70	0.85	0.52	0.66
PIMA Std. Dev.	0.13	0.16	0.15	0.04	0.06
ABALONE Ave.	0.37	0.33	0.40	0.41	0.36
ABALONE Std. Dev.	0.09	0.05	0.08	0.05	0.09
	N1	N2	N3	N4	N5
PIMA Average	0.35	0.43	0.24	0.25	0.47
PIMA Std. Dev.	0.13	0.09	0.16	0.15	0.07
ABALONE Ave.	0.26	0.3	0.19	0.25	0.27
ABALONE Std. Dev.	0.17	0.2	0.11	0.1	0.16

4.4 Run-Time Accuracy and Post-Testing Accuracy

In these experiments, we first computed the classification accuracy of Citation KNN on all test instances by turning off the authoritative module. Then we fed each test instance into the Authoritative Citation-KNN one by one, and adjusted the authority values after each instance. After feeding the last test instance, we tallied the classification accuracy. We call this *run-time* accuracy. Finally, we turned off the feedback mode of the system and fed the entire test set into the system. With this, we obtained the *post-testing* classification accuracy. We performed this test on 0%-, 10%-, 20%-, 30%-, 40%-, and 50%-noise datasets, for K = 3, 5, and 7, and C = 0, 3, 5, and 7, and all six datasets. Tables 3 and 4 show, for example, the results we collected for ABALONE (0%- and 50%-noisy, respectively). Note that there were no improvements for learners with C = 0 (authority values were not involved).

Table 3. Pre-ACKNN, Run-Time (RT), and Post-Testing classification accuracies and improvement%, for ABALONE, 0% noise. Improvement% is the ratio (Post/Pre)*100.

0 % Noise		Pre	RT	Post	%Imp.
K	C				
3	0	21.65	21.65	21.65	0
5	0	23.95	23.95	23.95	0
7	0	23.28	23.28	23.28	0
3	3	23.08	24.33	39.46	70.97
3	5	23.28	24.04	38.51	65.42
3	7	23.37	23.66	36.69	57
5	3	24.71	24.43	36.69	48.48
5	5	23.95	23.37	36.11	50.77
5	7	24.52	24.14	34.77	41.8
7	3	24.62	24.43	35.63	44.72
7	5	24.52	25.19	34.67	41.39
7	7	24.9	25	35.06	40.8

From the two tables, we see that ACKNN had a higher improvement% for noisier datasets, when K=5 or higher. The ACKNN learner had the best classification accuracy at K=3, C=3 for both noise levels. However, when we look at the Pre-ACKNN results, the best classification accuracy occurred at K=5, C=3 for 0% noise and K=3, C=0 for 50% noise. This implies that under high

noise, Citation KNN actually did not improve the classification accuracy. Moreover, the selection of the appropriate *K* and *C* values may be difficult. With the help of an authority measure, ACKNN was able to produce a more consistent classification under different levels of noise.

Table 4. Pre-ACKNN, Run-Time (RT), and Post-Testing classification accuracies and improvement%, for ABALONE, 50% noise. Improvement% is the ratio (Post/Pre)*100.

50 % Noise		Pre	RT	Post	%Imp
K	C				
3	0	17.53	17.53	17.53	0
5	0	18.87	18.87	18.87	0
7	0	17.43	17.43	17.43	0
3	3	16.95	18.01	28.26	66.73
3	5	17.05	17.72	26.25	53.96
3	7	16.09	16.48	24.04	49.41
5	3	16.48	18.3	27.2	65.05
5	5	16.57	17.91	25.77	55.52
5	7	16.09	16.86	24.33	51.21
7	3	15.61	18.49	26.72	71.17
7	5	15.9	18.1	25.67	61.45
7	7	15.8	17.34	23.47	48.54

Table 5 shows the runtime and the post-testing improvements for the six datasets. ACKNN was able to produce improvements both runtime and post-testing over the original Citation KNN classification accuracy.

Table 6 shows the runtime and the post-testing improvements from the point of view of noise levels, across the six datasets.

In general, we see that the post-testing improvement% increased as the noise level in the training instances increased. This shows that the authority value was able to counter noise to a certain degree of competency. In other words, without adding any new instances into the noise-corrupted training datasets, ACKNN is able to improve the classification accuracy by simply adjusting the authority value on each citer.

Table 5. Run-Time (RT), and Post-Testing classification improvement% for all datasets (0%-50% noise).

DATA SET	RT IMPROVEMENT%	POST-TESTING IMPROVEMENT%
PIMA	6.42	21.43
ABALONE	8.55	72.36
LIVER	0.86	23.33
WINE	7.28	19.25
WPBC	4.69	8.68
PENDIGITS	11.38	22.20
AVERAGE	6.53	27.87

Table 6. Run-Time (RT), and Post-Testing classification improvement% for all noise levels (across all datasets).

DATA SET	RT IMPROVEMENT%	POST-TESTING IMPROVEMENT%
0%	0.40	12.40
10%	-0.81	13.02
20%	2.39	20.63
30%	6.81	32.21
40%	8.16	33.48
50%	22.23	55.51
AVERAGE	6.53	27.87

We performed several correlation tests between the above tables and the general characteristics of the datasets: the size of the training set, the size of the test set, the number of classes, the number of features, average standard deviation of the attribute values, the ratio (size of the test sets)/(size of the training set) (Ratio1), the ratio (#features)/(#classes) (Ratio2), and the ratio (maximum number of instances per class)/(minimum number of instances per class) (Ratio3) at each noise level. Table 7 shows the correlation values that were above 0.75 or lower than –0.75.

In general the number of classes was a factor in the post-testing improvement% with a correlation around 0.9, and the average standard deviation of the attribute values was also a factor with a –0.9 correlation. Furthermore, the ratio (maximum number of instances per class)/(minimum number of instances per class) also played a role with a correlation of –0.9. We thus conclude that ACKNN generally performs better when the number of classes in the datasets is large, when the attribute values are clustered compactly, and when the number of instances per class is roughly the same.

Table 7. Significant correlations with respect to RT and Post-Testing improvement%.

DATA SET	RT IMPROVEMENT%	POST-TESTING IMPROVEMENT%
0%	none	Ratio3 (-0.88) #Classes (0.91) ave. std. dev. (-0.93)
10%	none	Ratio3 (-0.90) #Classes (0.89) ave. std. dev. (-0.91)
20%	Ratio3 (-0.93) #Classes (0.86)	Ratio3 (-0.92) #Classes (0.92) ave. std. dev. (-0.88)
30%	Ratio3 (-0.92) #Classes (0.90)	Ratio3 (-0.88) #Classes (0.93) ave. std. dev. (-0.90)
40%	Ratio1 (-0.97)	Ratio2 (-0.77) #Classes (0.86) ave. std. dev. (-0.92)
50%	none	Ratio1 (0.93) Ratio2 (-0.78)

During the runtime learning, the accuracy was marginally affected by Ratio3 and #Classes. The size of the training set or test set did not have a highly correlated impact on the learning behavior. The number of features did not have a significant correlation to the performance of ACKNN.

These results hint at the robustness and weakness of ACKNN. This hybrid learner's performance is influenced by three parameters (#Classes, average standard deviation of the attribute values, and Ratio3) when the noise level in the training dataset is between 0 and 30%. When the noise level is higher, the learner detracts from its normalcy.

4.5 Iterations of Test Datasets

In this set of experiments, we fed into the ACKNN learner several iterations of the same datasets to study convergence behavior. The process of each experiment was: (1) Feed the entire test dataset into the ACKNN, and obtain the Post-Testing classification accuracy, and (2) repeat step (1) with the same test dataset, until there are 8 iterations.

Figure 5 shows an example of the classification accuracy for ABALONE, for all ACKNN learners K = 3, 5, and 7 and C = 0, 3, 5, and 7. We see that the classification accuracy peaked at about the third iteration and converged after the fifth iteration. Indeed, for all datasets, the classification accuracy of the ACKNN converged before the sixth iteration, on average. If we look at each combination of K and C, we see that it took the learner longer to converge when K and C were larger: around 8 iterations for K=7 and C=7 vs. around 4.0 iterations for K=3 and C=3.

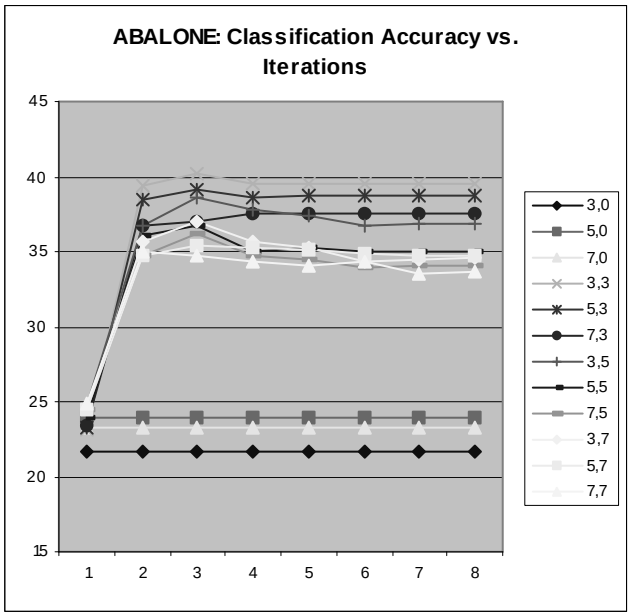

Figure 5. The classification accuracy instances for the PIMA dataset. A curve of 5,3 means it is for learner with K=5 and C=3. No noise was added to the dataset.

For these experiments, we also performed a series of correlation tests. We also added a parameter called the Best Performance (BP) value where the largest increase in classification accuracy occurred (one of the iterations), and its associated parameters best K (BK) and C (BC) values. Table 8 shows the correlation values that were significant. In general, as previously shown in Section 4.4, the three parameters (Ratio3, #Classes, and the average standard deviation of attribute values) played a role in the Best Performance value, and the best K and C values were not affected by any of the parameters studied. Further, when tallying up the average best K and C values across the noise levels, over all datasets, we observed Table 9. There was a correlation between the average BK values and the noise level (-0.75) (a smaller K is better when the noise level gets larger) but no correlation between the average BC values and the noise level (-0.32). We conclude that the appropriate C value cannot be directly determined from the parameters that we investigated.

Table 8. Significant correlations with respect to BP, BK, and BC.

DATA SET	BP	BK	BC
0%	Ratio3 (-0.93)	none	Ratio2 (-0.76)
10%	Ratio3 (-0.96) #Classes (0.85) ave. std. dev. (-0.92)	Ratio 2 (0.75) #Classes (-0.76)	none
20%	Ratio3 (-0.88) #Classes (0.89) ave. std. dev. (-0.94)	Ratio1 (-0.83)	none
30%	Ratio3 (-0.88) #Classes (0.90) ave. std. dev. (-0.91)	none	none
40%	Ratio3 (-0.92) #Classes (0.80) ave. std. dev. (-0.87)	none	none
50%	Ratio1 (-0.96)	none	none

Table 9. Average BP, BK, and BC across datasets for each noise level.

DATA SET	IMPROVEMENT% OF BP	AVERAGE BK	AVERAGE BC
0%	23.00	5.0	5.0
10%	26.06	5.0	3.8
20%	32.62	3.8	3.8
30%	43.74	2.7	4.0
40%	52.35	3.0	4.0
50%	74.67	3.7	4.3

5. Conclusions

In this paper, we have proposed a hybrid learner (combining lazy and reinforcement learning) called Authoritative Citation KNN (ACKNN) to classify noisy training datasets. We have conducted experiments on a variety of datasets with different general characteristics and noise levels. We have also tested the effectiveness of the ACKNN learner using different numbers of nearest neighbors (*K*) and citers (*C*). We have shown that, based on our results, the ACKNN is able to improve classification with noisy training data by simply changing a simple authority measure on the training instances. We have also shown that ACKNN is affected by certain parameters of the datasets through correlation tests. We have also shown that by observing the behavior of the authority values, we can judge the inherent noise level of a training dataset.

Our future work includes experiments in noisy features, multiple-instance learning, and noisy test instances. We are currently looking into incorporating the authority values as a key feature in the Euclidean distance measurement between two instances. As for an application, we aim to build a system integrating noisy, less-accurate archived data with new, more accurate data for classification.

Acknowledgement

The above research work was partially sponsored by the UCARE program at the University of Nebraska, Lincoln, NE. The authors would like to thank the UCI repository for the machine learning datasets.

References

Aha, D. W. (Ed.) (1997). *Lazy learning*. Norwell: Kluwer.

Aha, D. W. and D. Kibler (1989). Noise-tolerant instance-based learning algorithms. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (pp. 794-799). San Francisco: Morgan Kaufmann.

Amar, R. A., D. R. Dooly, S. A. Goldman, and Q. Zhang (2001). Multiple-instance learning of real-valued data. *Proceedings of the 18th Int. Conference on Machine Learning* (pp. 3-10). San Francisco: Morgan Kaufmann.

Angluin, D. and P. Laird (1988). Learning from noisy examples, *Machine Learning*, *2*, 343-370.

Aslam, J. A. and S. E. Decatur (1996). On the sample complexity of noise-tolerant learning, *Information Processing Letters*, 57, 189-195.

Atkeson, C. G., A. W. Moore, and S. Schaal (1997). Locally weighted learning, *Artificial Intelligence Review*, *11*, 11-73.

Cost, S. and S. Salzberg (1993). A weighted nearest neighbor algorithm for learning with symbolic features, *Machine Learning, 10*, 57-78.

Dietterich, T. G., R. H. Lathrop, and T. Lozano-Perez (1997). Solving the multiple-instance problem with axis-parallel rectangles, *Artificial Intelligence*, *89*, 31-71.

Garfield, E. (1979). *Citation indexing: its theory and application in science, technology, and humanities*. New York: John Wiley & Sons.

Maron, O and T. Lozano-Pérez (1998). A framework for multiple-instance learning, *Advances in neural information processing systems*, MIT Press.

Sutton, R. S. and A. C. Barto (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA, MIT Press.

Wang, J. and J.-D. Zucker (2000). Solving the multiple-instance problem: a lazy learning approach. *Proceedings of the 17th Int. Conference on Machine Learning* (pp. 1119-1125). San Francisco: Morgan Kaufmann.

Wetteschereck, D., D. W. Aha, and T. Mohri (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, *Artificial Intelligence Review*, *11*, 273-314.