An Analysis of Instance Selection for Neural Networks to Improve Training Speed

Xunhu Sun
Department of Computer Sciences
Florida Institute of Technology
Melbourne, FL 32901, USA
sunx2013@my.fit.edu

Philip K. Chan
Department of Computer Sciences
Florida Institute of Technology
Melbourne, FL 32901, USA
pkc@cs.fit.edu

***Abstract*—Training Artificial Neural Networks (ANN) is relatively slow compared to many other machine learning algorithms. In this study, we focus on instance selection to improve training speed. We first evaluate the effectiveness of instance selection algorithms for k-nearest neighbor algorithms with ANN. We then analyze factors in accuracy—distance from decision boundary, dense regions, and class distributions, and propose new instance selection algorithms. We discuss the tradeoff between accuracy and training speed, and introduce a measure for the tradeoff. Our empirical results on real data sets indicate that our proposed RDI is more effective with ANN.**

Keywords—instance selection; neural networks; training speed

I. INTRODUCTION

Training Artificial Neural Networks (ANN) is relatively slow compared to many other machine learning algorithms. Factors that contribute to the training time include the learning algorithm, stopping criterion (such as error/weight convergence and number of iterations), number of features, and number of instances. In this study, we focus on algorithms for reducing the number of instances to improve the training speed of ANN for classification tasks. To our knowledge, few studies have been conducted on instance selection for ANN.

However, to reduce storage and increase prediction speed, a number of instance selection algorithms have been proposed for k-Nearest Neighbor (kNN) algorithms. In this study, we investigate some of these algorithms and their effectiveness for ANN. We analyze some of the factors that can influence the design of instance selection algorithms for ANN.

Our contributions include: (a) analysis of the effectiveness of existing kNN instance selection algorithms for ANN, (b) interaction between instance selection and finding an effective number of iterations for ANN, (c) retention rate is an important factor in accuracy, (d) analysis of other factors--distance from decision boundaries, dense regions, and class distributions and algorithms based on those factors, (e) AIR Ratio for analyzing the tradeoff between accuracy and retention (training speed), and (f) our proposed RDI is more effective for ANN.

We first discuss related work in Sec. II. Sec. III and IV analyze existing instance selection algorithms with ANN. Sec. V to VII discuss three factors in accuracy and propose algorithms. Sec. VIII and IX analyze training speed and its tradeoff with accuracy. We conclude in Sec. X.

II. RELATED WORK

A few studies [4,7,10] compare the effectiveness of instance selection algorithms for kNN with respective to a number of data sets. To our knowledge, Garcia et al. [4] is the most recent and comprehensive study. Many of the instance selection algorithms consider the notion of border and interior instances with respect to the decision boundary between different classes. Garcia et al. [4] identify 3 types of selection algorithms: condensation, edition, and hybrid. Condensation methods (e.g. FCNN [1]) aim at retaining border points because interior points are less effective in determining the decision boundary. Generally, though condensation methods maintain accuracy over the training set, they might degrade generalization accuracy over the test set. Since border points are generally fewer than interior points, condensation methods can have a low retention rate. Edition methods (e.g. ENN [9]) aim at removing border points and points that are noisy. Consequently, the decision boundary is smoother, which can improve generalization accuracy over the test set. However, the retention rate is generally higher than condensation methods. Hybrid methods (e.g. HMNEI [5]) try to find a subset of points to maintain or improve generalization accuracy and may remove border and/or interior points.

El Hindi & Al-Akhras [3] propose using instance selection methods to smooth decision boundaries and reduce overfitting for ANN. They study seven instance selection methods, such as ENN [9] and DROP3 [10], and found ENN to be effective in smoothing decision boundaries and reducing noise. However, their study does not include an analysis of the overhead for instance selection and overall training time. Also, it does not analyze the relationship between accuracy and retention rate.

III. KNN INSTANCE SELECTION ALGORITHMS WITH ANN

The first question we would like to answer is how effective existing instance selection algorithms for kNN can be applied to ANN. In choosing instance selection algorithms for this study, we use two criteria: (1) algorithms that are relatively fast because ANN is relatively computationally expensive and (2) algorithms that are more effective according to the previous surveys. We choose FCCN [1] and HMNEI [5] based on Garcia et al. [4], SPOCNN & RPOCNN [8] according to Olvera-Lopez et al. [7], and DROP3 based on Wilson and Martinez [10]. We also include ENN [9] that is effective in smoothing decision boundaries [3] and Random (which randomly selects 50% of the instances) as a baseline.

A. Experimental Data and Procedures

We use seven data sets, summarized in Table I, from the UCI Machine Learning Repository [2]. The attribute and output values are normalized for instance selection and ANN.

TABLE I. SUMMARY OF DATA SETS

	Bupa	*Haber Man*	*Heart*	*Iono sphere*	*Iris*	*WDBC*	*Wine*
#Attributes	6	3	13	34	4	30	13
#Classes	2	2	2	2	3	2	3
#Instances	345	306	270	351	150	569	177

TABLE II. ACCURACY (%) -- INSTANCE SELECTION BEFORE CV

Algorithm	**Accuracy (%)**							
	Bupa	***Haber Man***	***Heart***	***Iono sphere***	***Iris***	***WDBC***	***Wine***	***Average***
FDS	71.5	74.5	83.2	89.7	96.3	97.3	98.0	87.2
Random	68.7	73.7	82.1	**85.4**	95.9	96.4	96.4	**85.5**
DROP3	67.4	73.4	78.4	60.7	93.3	91.2	89.7	79.2
ENN	66.5	**75.5**	**82.5**	83.8	**96.7**	**96.5**	**96.5**	85.4
FCNN	68.9	73.2	78.5	43.7	90.1	95.6	86.2	76.6
HMNEI	63.6	73.9	80.7	82.8	**96.7**	95.5	92.5	83.7
RPOCNN	69.8	72.8	80.2	67.8	78.9	92.8	92.4	79.2
SPOCNN	**73.3**	74.6	82.2	68.8	91.7	95.9	96.1	83.2

For each data set, we randomly choose 2/3 for training and 1/3 for testing. Instance selection is then performed on the training set. Using the selected training set, we perform 3-fold cross validation to determine number of iterations for the final training (seeking the iteration up to 10,000 with the highest validation accuracy). We use Backpropagation [7] with stochastic gradient descent for training the ANN. We vary the number of sigmoid units (3, 5 and 10) in the hidden layer. The number of sigmoid output units is the number of classes (but only one output unit for 2-class problems). The learning and momentum rates are both 0.1.

Our evaluation criteria are accuracy of ANN on the test set and retention rate (percentage of instances retained by the instance selection algorithm). For each criterion, we report the average of 30 runs (10 repetitions * 3 hidden-layer settings).

B. *Results and Analysis*

Table II shows the accuracy of the different algorithms. FDS (Full Data Set) does not have instance selection. Unexpectedly, Random, our baseline algorithm, has the highest average accuracy. ENN has a similar accuracy and the highest accuracy in five of the seven data sets. ENN and Random are quite comparable in terms of accuracy. We defer the discussion on retention rate to the next section.

IV. EFFECTIVE NUMBER OF ITERATIONS FOR ANN

Selecting instances before training ANN seems natural, however, the selection procedure influences how ANN finds an effective number of iterations. Consider a dataset (D) is first randomly separated into a training set (R) and a test set (T). During the 3-fold cross validation (CV) for finding an effective number of iterations, R is randomly separated into a training set (S) and a validation set (V). That is, D, R, T, S, and V have the same distribution. However, an algorithm that selects instances based on some heuristics can produce an output data set with a distribution that is different from the distribution of the input data set. In our case, the distribution of D, R, and T are the same and unchanged, but instance selection can influence the distributions of S and V. We consider two scenarios.

The first scenario performs instance selection *before* CV. That is, instance selection is performed before ANN training (Sec. III). Given R as input, instance selection generates R', which has a distribution different from R. During CV, R' is randomly separated into S' and V' -- R', S' and V' have the same distribution. That is, instance selection influences the distribution of both S' and V'. However, the distributions of V' and T are different, which is undesirable because V' is used to approximate T.

TABLE III. ACCURACY AND RETENTION (%) -- INSTANCE SELECTION INSIDE CV

Algorithm	**Accuracy (%)**							
	Bupa	***Haber Man***	***Heart***	***Iono sphere***	***Iris***	***WDBC***	***Wine***	***Average***
FDS	71.5	74.5	83.2	89.7	96.3	97.3	98.0	87.2
Random	68.1	74.2	**82.5**	**85.4***	94.1	97.1*	97.1*	**85.5**
DROP3	64.8	72.7	77.9	73.4	93.9	95.0	90.9	81.2
ENN	67.2	73.9	82.3	84.4	**95.9**	**97.3**	**97.5**	**85.5**
FCNN	70.4	**75.1**	81.7	74.4	93.9	96.5	94.2	83.7
HMNEI	59.5	73.9	81.2	82.1	92.4	96.2	95.5	83.0
RPOCNN	67.9	73.0	80.9	83.0	92.9	94.0	94.5	83.7
SPOCNN	**70.6**	73.4	81.0	83.2	91.4	96.1	95.5	84.5
	Retention Rate (% of Full data set)							
DROP3	30.9	19.1	16.7	11.1	11.1	4.5	11.0	14.9
ENN	61.7	69.6	78.3	85.9	96.0	96.3	95.8	83.4
FCNN	56.1	52.0	40.0	21.8	17.2	13.5	14.4	30.7
HMNEI	22.6	25.5	31.1	25.6	29.3	37.2	24.6	28.0
RPOCNN	50.0	40.2	34.4	18.8	13.1	12.1	15.3	26.3
SPOCNN	63.0	52.5	42.8	25.6	13.1	15.3	16.9	32.8

a. * indicates Random has significant higher accuracy than at least 3 algorithms based on a t-test with 95% confidence.

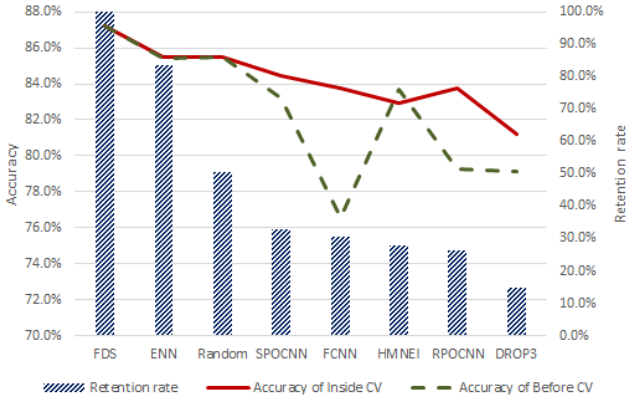

Figure 1. Retention and accuracy of instance selection inside/before CV.

The second scenario performs instance selection *inside* CV. R is randomly separated into S and V -- R, S, and V have the same distribution. Given S as input, instance selection generates S', which has a distribution different from S. That is, instance selection only influences the distribution of S, but not V. Consequently, the distribution of V remains unchanged and is the same as the distribution of T, which is desirable.

We repeat the experiments in Sec. III, but perform instance selection inside CV (instead of before CV). Table III summarizes the accuracy on the test set and instance retention rate. We observe that ENN and Random are more accurate than the other instance selection algorithms. Interestingly, compared with other algorithms, ENN seems to be more effective with easier data sets (accuracy in the 90s for FDS) and less effective with less easy data sets (accuracy in the 70s for FDS). Though ENN and Random have the same accuracy on average, Random retains only 50% of the instances, while ENN retains 83.4%. Since ENN only removes border and noisy instances to smooth the decision boundary, it does not remove a lot of instances generally.

To compare instance selection inside CV with before CV (Sec. III), we plot their accuracy in Fig. 1 (algorithms are arranged on the x-axis in descending order of retention rate). Instance selection inside CV is mostly similar or more accurate than before CV since the validation set is more representative of the test set. The only exception is for HMNEI.

Though ENN and Random are more accurate than the other algorithms, they retain more instances. In Fig. 1 we observe correlation between accuracy and retention rate. Generally, a higher retention rate yields a higher accuracy. This is expected as more instances provide more information for the learning algorithm. More importantly, the drop in accuracy is much less than the drop in retention, indicating that the algorithms are selecting more important instances. For example, DROP3 has the lowest accuracy of 81.2%, which is a drop of 6% from 87.2% (FDS), but it only keeps 14.9% of the instances, which is a drop of 85.1%. That is, though there is a tradeoff between accuracy and retention (hence ANN training speed), the relatively small drop in accuracy might be appropriate if the ANN training speed is prohibitively slow, particularly in the era of big data. *We defer our discussion on the tradeoff between accuracy and retention to Sec. IX.*

V. Distance from the Decision Boundary

Selecting border and/or interior instances is frequently considered in different instance selection algorithms for kNN. kNN does not explicitly identify the decision boundary; however, ANN does, via the weights between nodes in the different layers. We would like to study the effect of selecting instances with different distances from the decision boundary for ANN. We first create two synthetic data sets: (a) 2x2 Checkerboard and (b) Nested square (Fig. 2). In both data sets, all the instances are within the unit square and the total area representing each class is the same. We divide each class into 3 regions with respect to distance from the decision boundary: border, middle, and far.

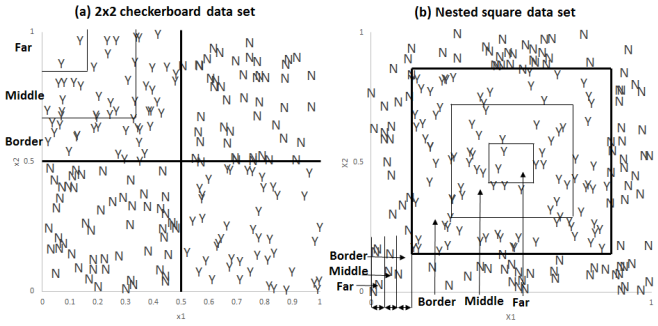

Figure 2. Synthetic data: 2x2 checkerboard and nested square.

TABLE IV. Accuracy for Distance from Boundary (Synthetic Data)

Data set	Accuracy (%)				
	All regions	***Border***	***Middle***	***Far***	***Border+ Middle***
Checker board	93.0	**93.6**	85.1	81.7	92.0
Nested square	**83.0**	80.5	80.7	72.9	**83.0**

TABLE V. Accuracy and Retention for Distance from Boundary (Real Data)

Algorithm	Accuracy (%)							
	Bupa	***Haber Man***	***Heart***	***Iono sphere***	***Iris***	***WDBC***	***Wine***	***Average***
FDS	71.5	74.5	83.2	89.7	96.3	97.3	98.0	87.2
Random	68.1	74.2	82.5	85.4	94.1	97.1	97.1	85.5
RFI	**71.8***	**74.7**	81.6	**86.1**	**96.1***	**97.2**	**98.0**	**86.5**
RFI+ENN	67.4	74.0	**82.8**	81.7	95.2	97.1	95.9	84.9
	Retention Rate (% of Full data set)							
RFI	86.5	87.3	81.7	82.5	76.8	87.3	83.9	83.7
RFI+ENN	50.4	58.3	58.9	72.6	71.7	83.9	79.7	67.9

a. * indicates RFI has significant higher accuracy than Random based on a t-test with 95% confidence.

We create three different training sets for the border, middle, and far regions by randomly select 80 points in each of the three regions. In addition, we create another training set with all regions, by selecting all 80 points from all three regions. Note that all four training sets have the same number of instances. We train the ANN with the same parameters as in previous experiments. The test set has instances from all regions. We repeat the procedure 20 times. Table IV shows the average ANN accuracy over 20 runs. As expected, we observe that ANN has a lower accuracy with training instances from the far region than those from the border, middle, and all regions. The border region is more accurate than the middle region in 2x2 Checkerboard. However, the border and middle regions in the nested square seem to be similar in accuracy. Also, all regions are the most accurate in Nested square. Since the far region is consistently the least accurate in both data sets, we create a "border+middle" training set that excludes the far region. The "border+middle" region is the most accurate in the Nested square. Overall, the far region seems to be significantly less effective and removing instances from the far region could be beneficial in real data sets.

A. RFI algorithm and real data sets

Since instances in the far region seem to be significantly less effective, we propose the RFI (Remove Far Instances) algorithm. Let enemy-distance of an instance x to be the distance of x from its nearest neighbor that is in a different class. That is, enemy-distance estimates distance from the decision boundary. RFI calculates the enemy-distance of each instance, and the average and standard deviation of the enemy-distance of all instances. RFI removes instances that are farther than the average plus standard deviation. The time complexity of RFI is $O(n^3)$, n is the number of instances. Since ENN can remove noisy and border instances to smooth the decision boundary, RFI+ENN combines RFI and ENN.

Table V shows that RFI is more accurate than Random and RFI+ENN in 6 of the 7 data sets. However, RFI retains the most instances with 83.7% of the instances.

VI. Dense Regions

Among the regions of a class, some regions might be denser than other regions. We hypothesize that we can represent the denser regions with fewer instances without significantly affecting the effectiveness of ANN (in determining the decision boundary). To test our hypothesis, we first create a synthetic data set with dense regions in 2x2 Checkerboard (Fig. 3a). In each quadrant, we randomly select 40 instances and add 20 more in the center to create a denser region. The full data set (FDS) has (40+20)*4=240 instances. We create two training sets, each with 160 instances. The first set (-dense) has the initial 40 instances in each quadrant. The second set (+dense) has dense regions by random sampling from FDS. We repeat the procedure 20 times and report the average accuracy on the test set (with the same distribution as FDS but different instances from FDS) in Table VI.

Unexpectedly, -dense is significantly more accurate than +dense -- we were expecting similar accuracy. Also surprisingly, -dense is more accurate than FDS, even though FDS has more data. We examined the 20 runs of +dense and found 4 out of the 20 runs have lower than 90% in accuracy. In our experience with the 2x2 Checkboard, ANN can hit a local minimum and yield lower than 90% accuracy. Fig. 4 shows the predictions and decision boundary of the test set in one of the less accurate runs. The decision boundary has two roughly diagonal boundaries (dotted lines) instead of the two expected perpendicular boundaries. After reaching the local minimum, ANN cannot reduce error by making small changes to the boundaries (ANN weights). Therefore, our results indicate that removing some of the instances from the denser regions can reduce the local minimum problem and yield higher accuracy.

A. RDI algorithm and real data sets

We propose RDI (Remove Dense Instances) to identity and remove some of the instance in denser regions. Let k-distance be the distance of an instance from its k-th nearest neighbor of the same class. We consider an instance is in a denser region if its k-distance is lower than the average k-distance. We iteratively remove the instance with the lowest k-distance. After an instance is removed, the region has become less dense so we update the k-distance of instances that had the removed instance as one of its k nearest neighbors. The update also prevents all instances in the initial denser regions to be removed. The time complexity of RDI is $O(c*n^3)$, n is the average number of instances in each class, c is the number of classes. More concretely, the pseudocode for RDI is:

1. For each class c
   a. For each instance x in class c, calculate k-distance
   b. Calculate the average k-distance, which is the dense threshold for class c
   c. While the lowest k-distance is less than dense threshold
      i. Remove the instance (y) with the lowest k-distance
      ii. Update the k-distance of instances that had y as one of its k nearest neighbors

To visualize the effectiveness of RDI (k=3), we provide RDI with the data in Fig. 3a and RDI removes some instances in denser regions and generates the output in Fig. 3b. Also, in Table VI, RDI (automated) and -dense (manual) yield similar accuracy.

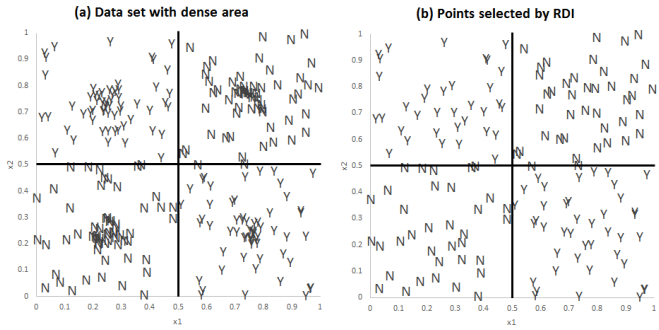

Figure 3. 2x2 Checkboard with dense regions and output of RDI.

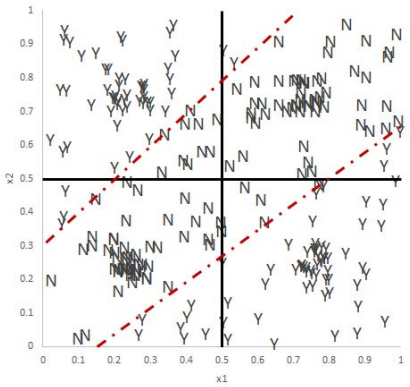

Figure 4. Local minimum with dense regions—dotted lines represent the learned decision boundaries.

TABLE VI. Accuracy with Dense Regions (Synthetic Data)

Result	*FDS*	*+dense*	*-dense*	*RDI*
Accuracy (%)	95.6	93.6	**97.8**	97.6
# of instances	240	160	160	151

TABLE VII. ACCURACY AND RETENTION FOR DENSE REGIONS (REAL DATA)

Algorithm	Accuracy (%)							
	Bupa	*Haber Man*	*Heart*	*Iono sphere*	*Iris*	*WDBC*	*Wine*	*Average*
FDS	71.5	74.5	83.2	89.7	96.3	97.3	98.0	87.2
Random	68.1	**74.2**	82.5	85.4	94.1	**97.1**	97.1	85.5
RDI	**68.6**	74.1	**83.4**	**87.6***	**97.8***	96.5	**98.0**	**86.6**
RDI+ENN	65.5	73.6	82.3	82.4	96.0	96.4	97.3	84.8
	Retention Rate (% of Full data set)							
RDI	55.7	64.2	65.6	65.4	66.7	58.8	67.8	63.4
RDI+ENN	34.3	45.1	50.0	55.6	63.6	55.9	66.1	53.0

a. * indicates RDI has significant higher accuracy than Random based on a t-test with 95% confidence.

Using the same experimental procedures for real data sets as in previous sections, we report accuracy and retention rate in Table VII. Since ENN can remove noisy and border instances to smooth the decision boundary, RDI+ENN combines RDI and ENN. Table VII indicates that RDI (k=3) is the most accurate. RDI also has the highest retention rate of 63.4%.

VII. CLASS DISTRIBUTION

Machine learning algorithms usually assume the training set has the same distribution as the test set. However, instance selection algorithms generally do not aim at maintaining the data distribution, so the training and test sets generally have different distributions. We hypothesize that difference in class distribution between the training and test set can adversely affect the accuracy on the test set. We create three different training sets for 2x2 Checkerboard with different class ratios: 1:1, 2:1, and 5:1. Fig. 5a illustrates the training set with the 1:1 ratio and Fig. 5b with the 5:1 ratio. The 3 training sets have the same number of instances (400) and the test set has a class ratio of 1:1. Table VIII displays average accuracy on the test set over 20 runs. Larger difference in class distribution between the training and test sets yields lower accuracy. Therefore, an instance selection algorithm might benefit from maintaining the class distribution.

ENN tries to remove border and noisy instances; hence, the data distribution is tilted away from the border. To maintain the data distribution, we could replace the removed instances with their nearest neighbors of the same class. Requiring the replacement instance to be of the same class maintains the class distribution. Based on these ideas, we propose the RanENN algorithm. RanENN first randomly marks 50% of the instances to be retained. It then performs ENN on the original data and marks instances to be removed. Finally, if an instance is marked to be retained but is also marked to be removed by ENN, RanENN finds a replacement that is the nearest neighbor of the instance and has the same class. The time complexity of RanENN is $O(n^2)$, n is the number of instances. In Sec. VIII, Table X shows the average RanENN selection time is less than 2% of ANN training time.

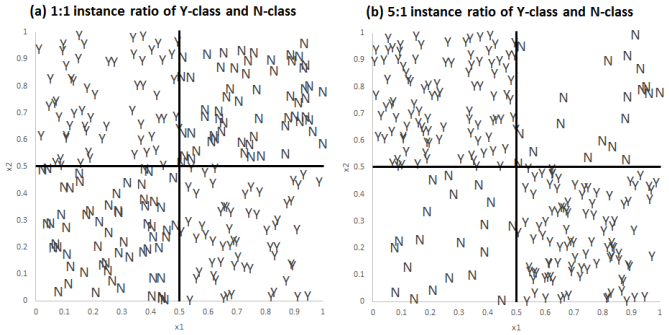

Figure 5. Synthetic data: Different class distributions.

TABLE VIII. ACCURACY FOR DIFFERENT CLASS DISTRIBUTIONS (SYNTHETIC DATA)

Class Ratio	**1:1**	**2:1**	**5:1**
Accuracy (%)	**96.3**	94.7	89.8

TABLE IX. ACCURACY FOR DATA DISTRIBUTION (REAL DATA)

Algorithm	Accuracy (%)								*AvgRet (%)*
	Bupa	*Haber Man*	*Heart*	*Iono sphere*	*Iris*	*WDBC*	*Wine*	*Avg*	
FDS	71.5	74.5	83.2	89.7	96.3	97.3	98.0	87.2	100
Random	68.1	74.2	**82.5**	**85.4**	94.1	97.1	97.1	85.5	50.4
RanENN	**68.6**	**75.1**	82.4	83.7	**96.7***	96.8	96.3	**85.7**	49.5
ENN	67.2	73.9	82.3	84.4	95.9	**97.3**	**97.5**	85.5	83.4

a. * RanENN has significant higher accuracy than Random based on a t-test with 95% confidence.

Using the same experimental procedures as before, Table IX shows the average accuracy of RanENN, which is slightly higher than Random and ENN. However, RanENN retains only 49.5% compared to 83.4% for ENN. This indicates a possibility that we can improve instance selection algorithms with some consideration on maintaining the class distribution.

VIII. INSTANCE SELECTION AND TRAINING TIME

To understand the impact of instance selection on ANN training time, we measure the total CPU time used in instance selection and ANN training, and the fraction of the total used for instance selection. CPU time is measured on a server that has 4 Xeon X7460 (6 cores each) CPUSs @ 2.66GHz with 32GB of memory. In Table X, we report the total training time as a percentage of time used by FDS (ie, no instance selection). We also report instance selection time as a percentage of total training time. On average, as expected, Random is the fastest and uses 49% of the time (ie, twice as fast as no instance selection). All instance selection algorithms use less than 2% of the total training time. The results indicate that the overhead of instance selection is very small and can improve the total training speed. To understand the relationship between retention rate and total training time, we plot the retention rate (in descending order) and the training time and in Fig. 6. As expected, training time is correlated with retention rate.

TABLE X. TRAINING AND INSTANCE SELECTION TIME

Algorithm	Total training time as a percentage of time used by FDS							
	Bupa	*Haber Man*	*Heart*	*Iono sphere*	*Iris*	*WDBC*	*Wine*	*Average*
FDS	100	100	100	100	100	100	100	100
Random	50.7	50.7	50.3	45.9	51.7	47.5	46.4	49.0
RanENN	47.2	41.6	49.4	46.8	51.9	51.8	47.8	48.1
ENN	56.2	63.5	79.4	77.2	97.0	101.6	95.0	81.4
RDI	56.3	67.7	64.8	86.4	70.0	78.9	70.6	70.7
RFI	87.0	86.5	84.9	83.2	86.5	85.3	88.0	85.9
	Instance selection time as a percentage of total training time							
Random	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1	< 0.1
RanENN	1.3	0.9	2.0	2.2	0.3	3.9	0.8	1.6
ENN	1.1	0.5	1.3	1.3	0.1	2.0	0.4	1.0
RDI	0.5	0.3	0.7	1.0	0.1	2.0	0.2	0.7
RFI	0.7	0.4	1.3	1.2	0.2	2.6	0.4	1.0

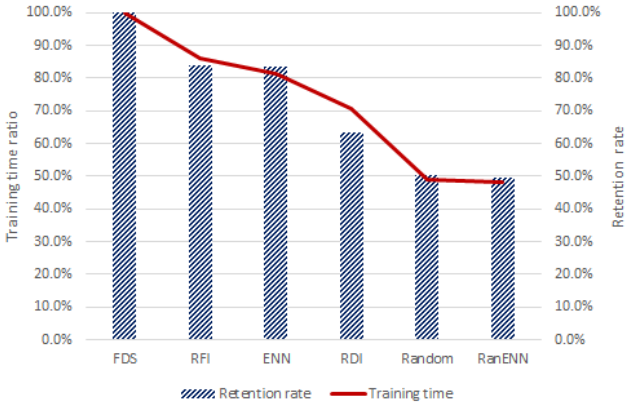

Figure 6. Training time and retention rate.

IX. TRADEOFF BETWEEN ACCURACY AND TRAINING SPEED

In Sec. IV we observe that accuracy is correlated with retention rate and in Sec. VIII we observe that training time is correlated with retention rate. That is, there is a tradeoff between accuracy and training speed. In practice, one might want to understand how much accuracy is reduced when number of instances is reduced. That is, if an instance selection algorithm gains training speed, how much it loses in accuracy. Hence, we propose the Accuracy Instance Reduction Ratio (AIR Ratio): [accuracy(algorithm) – accuracy(FDS)] / [1 – retention(algorithm)]. A negative AIR Ratio indicates a loss in accuracy for each percent of instance reduction (or in uncommon situations, a positive AIR Ratio means an increase in accuracy). Table XI orders existing and proposed instance selection algorithms in descending order of average AIR Ratio. Our results indicate that RDI (Sec. VI) is the most effective in selecting instances for ANN in 4 of the 7 data sets and on average.

TABLE XI. ACCURACY INSTANCE REDUCTION (AIR) RATIO.

Algorithm	AIR ratio * 100							
	Bupa	*Haber Man*	*Heart*	*Iono sphere*	*Iris*	*WDBC*	*Wine*	*Average*
RDI	-6.67	-1.00	**0.75**	**-6.05**	**4.76**	-1.79	**-0.18**	**-1.45**
RanENN	-5.90	1.17	-1.41	-11.96	0.92	-0.95	-3.50	-3.09
Random	-6.90	-0.52	-1.26	-8.60	-4.45	-0.35	-1.84	-3.42
RFI	**1.96**	**2.07**	-8.79	-20.28	-0.83	-0.84	-0.35	-3.86
SPOCNN	-2.43	-2.28	-3.76	-8.68	-5.59	-1.33	-3.07	-3.88
FCNN	-2.45	1.23	-2.35	-19.53	-2.93	-0.91	-4.44	-4.48
RPOCNN	-7.24	-2.52	-3.40	-8.23	-3.82	-3.71	-4.21	-4.73
HMNEI	-15.58	-0.74	-2.90	-10.13	-5.55	-1.65	-3.39	-5.71
DROP3	-9.62	-2.18	-6.32	-18.33	-2.63	-2.42	-7.99	-7.07
ENN	-11.30	-1.81	-3.71	-36.84	-9.03	**-0.05**	-12.78	-10.79

X. CONCLUDING REMARKS

We evaluated the effectiveness of existing kNN instance selection algorithms with ANN and found ENN and Random to be more effective. Our analysis indicates that retention rate is correlated with accuracy. To maintain the distribution of the validation and test sets, we select instances inside CV instead of before CV. Distance from decision boundary, dense regions, and data distribution affect accuracy--we propose RFI, RDI, and RanENN to select instance effectively. Instance selection improves training speed, which is correlated with retention rate. We propose AIR Ratio to measure the tradeoff between accuracy and instance reduction (hence speed). Our empirical results indicate that RDI is more effective for ANN.

REFERENCES

[1] F. Angiulli, "Fast nearest neighbor condensation for large data sets classification," IEEE Trans. Knowledge and Data Eng., vol. 19, no. 11, pp. 1450-1464, Nov. 2007.

[2] K. Bache, and M. Lichman. (2013). UCI Machine Learning Repository [Online]. Available: http://archive.ics.uci.edu/ml

[3] K. el Hindi, and M. AL-Akhras, "Smoothing decision boundaries to avoid overfitting in neural network training," Neural Network World, vol. 21, no. 4, pp. 311-325, 2011.

[4] S. García, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 3, pp. 417–435, Mar. 2012.

[5] E. Marchiori, "Hit miss networks with applications to instance selection," J. Machine Learning Research, vol. 9, pp. 997-1017, 2008.

[6] T. Mitchell, Machine Learning. ch. 4, Boston: McGraw-Hill, 1997.

[7] J. A. Olvera-López, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, and J. Kittler, "A review of instance selection methods," Artificial Intelligence Review, vol. 34, no. 2, pp. 133-143, May 2010.

[8] T. Raicharoen, and C. Lursinsap, "A divide-and-conquer approach to the pairwise opposite class-nearest neighbor (POC-NN) algorithm," Pattern recognition letters, vol. 26, no. 10, pp. 1554-1567, 2005.

[9] D.L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," IEEE Trans. Systems, Man, and Cybernetics, vol. 2, no. 3, pp. 408-421, July 1972.

[10] D. R. Wilson, and T. R. Martinez, "Reduction techniques for instance-based learning algorithms," Machine learning, vol. 38, no. 3, pp. 257-286, Mar. 2000.