IPRed: Instance Reduction Algorithm Based on the Percentile of the Partitions

Turki Turki[1,2]
[1]Computer Science Department
King Abdulaziz University
P.O. Box 80221, Jeddah 21589, Saudi Arabia
tturki@kau.edu.sa

Zhi Wei[2]
[2]Department of Computer Science
New Jersey Institute of Technology
University Heights, Newark, NJ 07102
{ttt2,zhiwei}@njit.edu

Abstract

Instance reduction methods are popular methods that reduce the size of the datasets to possibly improve the classification accuracy. We present a method that reduces the size of the dataset based on the percentile of the dataset partitions which we call IPRed. We evaluate our and other popular instance reduction methods from a classification perspective by 1-nearest neighbor algorithm on many real datasets. Our experimental evaluation on the datasets shows that our method yields the minimum average error with statistical significance.

Keywords: instance reduction, nearest neighbor, classification, statistical significance.

Introduction

Instance reduction methods are popular methods used in machine learning and data mining that reduce the size of the datasets to possibly improve the classification accuracy of the classification algorithms(Wilson and Martinez 2000). Among many methods the condensed nearest neighbor(CNN)(Hart 1968) is a popular algorithm. CNN outputs a reduced training set Z from the original training set S where Z classifies all instances in S correctly using 1-nearest neighbor (1NN)(Alpaydin 1997) . For given validation instance $v = (x^{'}, y^{'})$, prediction is given by(Wu et al. 2008)

$$y^{'} = \arg\max_{l^{'} \in L} I(l^{'} = y_1) \quad (1)$$

where $l^{'}$ is the label, the set of all labels is denoted by L, y_1 is the label for the closest instance x obtained by selecting the label for the instance $x \in Z$ that gives the minimum distance from $x^{'}$, $I(.)$ is an indicator function that outputs 1 if its argument is true and 0 otherwise, and $y^{'} \in L$ is the predicted label for the validation instance for which $I(y^{'} = y_1)$ attains its maximum. Reduced nearest neighbor(RNN)(Gates 1972) is a modification to improve CNN which includes additional step that is removing instances from reduced training set Z that do not cause any misclassification on the training set S(Bhatia and others 2010). Generalized condensed nearest neighbor rule(GCNN) is another variant of CNN which iteratively constructs reduced training set Z from training set S by selecting instances from S according to criterion(Chou, Kuo, and Chang 2006). All the previously mentioned algorithms significantly reduce the storage requirement for the training set used by the classification algorithms(Wilson and Martinez 2000; Chou, Kuo, and Chang 2006). However, the classification accuracy degrades most of the time on the validation set compared to the classification accuracy of the classification algorithms using the original training set(Chou, Kuo, and Chang 2006; Wilson and Martinez 2000).

In this paper we consider approach that partitions the dataset into training partitions S_k for $k = 1...9$ and a validation partition V to compute the sum of minimum distances between the validation partition V and each training partition S_k. We then construct reduced training set Z by selecting the training partitions that are less than the p^{th} percentile of all the training partitions for specific p value. The goal is to select the training partitions that contain similar instances to the validation instances and remove those with dissimilar instances. The idea behind the strategy is that instances dissimilar with validation instances will have little, if not adverse, effect on predicting validation instances, and thus can be removed with little, if not positive, effect on classification. An example in real life is that a professor gives students a take-home exam (i.e. validation set without labels) and students will look for chapters (i.e. training partitions) of the book (i.e. training set) which contain similar information (i.e. instances of the training partitions) to the exam questions (i.e. instances of the validation set) and skip irrelevant chapters. For given reduced training set Z the prediction is given by using 1NN(Equation 1). Compared to other popular instance reduction methods our experimental evaluation on 30 datasets shows that our method IPRed yields the minimum average error with statistical significance.

The rest of this paper is organized as follows. In section II we review related work. In section III we present our method IPRed. Following that we present experimental evaluation and discussion before concluding.

Related Work

Condensed Nearest Neighbor Rule

The CNN outlined in Algorithm 1 works as follows(Alpaydin 1997). CNN recieves training set S and empty subset Z(lines 1-2). At each iteration of the for loop

of lines 6-13 instance x is selected randomly from the training set S(line 7). Lines 8-9 perform 1NN classification as follows. In line 8 the instance z_c is obtained by selecting the instance z_j in the stored subset Z that gives the minimum distance from x. Lines 9-10 store the instance x in the subset Z if it is incorrectly classified. The do-while loop of lines 4-14 terminates when the instances in the training set S are correctly classified by the stored subset Z with 1NN. Line 15 returns the reduced training set Z. For given Z the prediction on the validation set V is given by 1NN(Equation 1).

Algorithm 1 Condensed Nearest Neighbor algorithm (CNN)

```
1:  CNN(S,Z)
2:      Z ← ∅
3:      Pre_add ← 1
4:      do
5:          add ← 0
6:          for all instances in training set S do
7:              Randomly select x from S
8:              Find z_c ∈ Z such that D(x, z_c) = min_j D(x, z_j)
9:              if label(x) ≠ label(z_c) then
10:                 Z ← Z ∪ x
11:                 add ← 1
12:             end if
13:         end for
14:     while (Pre_add == add)
15:     return (Z)
```

Reduced Nearest Neighbor Rule

The reduced nearest neighbor(RNN)(Gates 1972; Wilson and Martinez 2000) is a modification to CNN which starts with training set S and reduced training set Z where all instances are copied from the training set S to the reduced training set Z. The RNN algorithm iteratively removes each instance from Z if the removal does not lead to any misclassification of the other instances in S using the remaining instances in Z. Prediction is performed on the validation set using Z.

Generalized Condensed Nearest Neighbor Rule

The generalized condensed nearest neighbor(GCNN)(Chou, Kuo, and Chang 2006; Olvera-López et al. 2010) sequentially adds instances from the training set S to the reduced training set Z when the instances are not absorbed by Z. Instance $x \in S$ is absorbed when $|x - a| - |x - b| > \delta$ where $a \in Z$ is the nearest instance to x with the same class label(i.e. label(x) = label(a)), $b \in Z$ is the nearest instance to x with different class label(i.e. label(x) $\neq$ label(b)), and δ is the threshold. Prediction is made on the validation set using Z.

IPRed

As shown in Algorithm 2 the DP algorithm receives as an input a dataset $D \in R^{k \times d+1}$ of size k where the label associated to the instance $x_k \in R^d$ is y_k(line 1). In line 2 we divide the dataset size k by 10 and take the floor of the result. Lines 3-4 initialize the variables. The for loop of lines 5-18 creates 9 training partitions P_k for $k = 1...9$ by taking subsamples without replacement from dataset D where the size of each P_k is the same. The for loop of lines 20-28 creates the validation partition V_1 by taking the remaining subsamples without replacement from the dataset D. Line 29 assigns the 9 training partitions to S. In line 30 we assign the validation partition V_1 to V. In line 31 our algorithm outlined in Algorithm 3 is called by taking partitions in S and V as inputs. IPRed algorithm outputs a training set Z of reduced size to guide the 1NN algorithm for better classification performance on the validation set V. The IPRed algorithm which is outlined in Algorithm 3 works as follows. Line 2 stores the size of the validation partition V_1. The for loop of lines 3-12 iterates 9 times to store the sum of the minimum euclidean distances(line 11) between the partitions as follows. At each iteration of the for loop of lines 5-10 we store the computed euclidean distances(line 7) between the nth instance in $V_1[n, j]$ and each instance i in the kth partition $P_k[i, j]$ for $i = 1...p\_size$ and $j = 1...d$(for loop of lines 6-8).

Algorithm 2 Dataset Partitioning algorithm (DP)

```
1:  DP(D = {(x_1, y_1), ..., (x_k, y_k)})
2:      size ← ⌊k/10⌋
3:      b ← 1
4:      c ← size
5:      for k = 1 to 9 do
6:          q ← 1
7:          r ← 1
8:          for i = b to size do
9:              for j = 1 to d do
10:                 P_k[q, r] ← x[i, j]
11:                 r ← r + 1
12:             end for
13:             P_k[q, r] ← y[i]
14:             q ← q + 1
15:         end for
16:         b ← b + c
17:         size ← size + c
18:     end for
19:     q ← 1
20:     for i = b to k do
21:         r ← 1
22:         for j = 1 to d do
23:             V_1[q, r] ← x[i, j]
24:             r ← r + 1
25:         end for
26:         V_1[q, r] ← y[i]
27:         q ← q + 1
28:     end for
29:     S ← {P_1, P_2, ..., P_9}
30:     V ← {V_1}
31:     Z ← IPRed(S, V)
```

We then store the minimum distance(line 9). After the for loop of lines 5-10 terminates, we store the sum of the min-

imum distances between the kth partition P_k and V_1(line 11). Thus, the for loop of lines 3-12 iterates 9 times to store the results for the 9 training partitions(line 11). In line 13 we use the percentile as a robust measurement of the location of the data to store the 75^{th} percentile of all training partitions. Line 14 initializes j to 1. The for loop of lines 15-20 selects the ith training partition that is less than 75^{th} percentile of all training partitions for $i = 1...9$. We store the index of the ith training partition that satisfies the condition(line 16) in $Partitions\_index$(line 17). After the for loop of lines 22-31 terminates we return the reduced training set Z(line 32) which contains the selected training partitions in $Partitions\_index$. Prediction on the validation set V is given by 1NN(Equation 1) using Z.

Algorithm 3 IPRed algorithm

1: **IPRED**($S = \{P_1, P_2, ..., P_9\}$, $V = \{V_1\}$)
2: $v\_size \leftarrow V_1.size$
3: **for** k = 1 **to** 9 **do**
4: $p\_size \leftarrow P_k.size$
5: **for** n = 1 **to** v_size **do**
6: **for** i = 1 **to** p_size **do**
7: $distances[i] \leftarrow \sqrt{\sum_{j=1}^{d}(P_k[i,j] - V_1[n,j])^2}$
8: **end for**
9: $nearest\_instances[n] \leftarrow Min(distances)$
10: **end for**
11: $Partitions[k] \leftarrow \sum_{t=1}^{v\_size} nearest\_instances[t]$
12: **end for**
13: $Q4 \leftarrow Percentile(Partitions, 75\%)$
14: $j \leftarrow 1$
15: **for** i = 1 **to** $length(Partitions)$ **do**
16: **if** $Partitions[i] < Q4$ **then**
17: $Partitions\_index[j] \leftarrow i$
18: $j \leftarrow j + 1$
19: **end if**
20: **end for**
21: $n \leftarrow 1$
22: **for** m = 1 **to** $length(Partitions\_index)$ **do**
23: $k \leftarrow Partitions\_index[m]$
24: $p\_size \leftarrow P_k.size$
25: **for** i = 1 **to** p_size **do**
26: **for** j = 1 **to** $d + 1$ **do**
27: $Z[n,j] \leftarrow P_k[i,j]$
28: **end for**
29: $n \leftarrow n + 1$
30: **end for**
31: **end for**
32: **return** (Z)

Experimental Evaluation

We experimentally evaluate the performance of our instance reduction algorithm and compare it against others from a classification perspective(Japkowicz and Shah 2011) by 1-nearest neighbor(1NN) algorithm on 30 real datasets shown in Table 1. This section describes the datasets, experimental methodology, then presents the experimental results.

Code	Dataset	Classes	Dimensions	Instances
1	Soy Bean	4	35	47
2	Colon Cancer	2	2000	62
3	Leukemia	2	7129	72
4	Breast Tissue	6	9	106
5	Appendicitis	2	7	106
6	Iris	3	4	150
7	Hayes-Roth	3	4	160
8	Global Cancer Map	14	16063	190
9	Glass Identification	6	10	214
10	Heart	2	13	270
11	Haberman	2	3	306
12	Ecoli	8	6	336
13	Liver Disorders	2	6	345
14	Bci	2	117	400
15	Saheart	2	9	462
16	Musk	2	166	476
17	Led	10	7	500
18	Climate	2	18	540
19	Breast Cancer	2	30	569
20	Digits	2	63	762
21	Diabetes	2	8	768
22	Vowel	11	13	990
23	Statlog German Credit Card	2	24	1000
24	Banknote Authentication	2	4	1372
25	Contraceptive	3	9	1473
26	Wine Quality Red	11	11	1599
27	Ozone	2	72	1847
28	Steel Plates Faults	7	27	1941
29	Insurance Company COIL 2000	2	85	5822
30	Magic Gamma Telescope	2	10	19020

Table 1: Datasets from the UCI(A. Asuncion 2007), KEEL(Alcalá et al. 2010), and Bioinformatics repositories(Jesús S. Aguilar-Ruiz) that we used in our experimental evaluation

Datasets

We use 30 real datasets in our experiments for classification. The information on datasets is tabulated in Table 1 which are ordered in terms of increasing number of instances. The datasets are obtained from three different sources. The appendicitis and saheart are obtained from KEEL-dataset repository(Alcalá et al. 2010). The colon cancer, leukemia, and global cancer map are obtained from BioInformatics Research Group-dataset repository (Jesús S. Aguilar-Ruiz). All the remaining datasets are from the UCI machine learning repository(A. Asuncion 2007).

Experimental Methodology

We evaluate four classification algorithms: IPRed+1NN, CNN+1NN, RNN+1NN, and 1NN where the four algorithms are 1-nearest neighbor applied to IPRed, 1-nearest neighbor applied to condensed nearest neighbor, 1-nearest neighbor applied to reduced nearest neighbor, and 1-nearest neighbor respectively. We use the 10-fold cross-validation on each dataset ensuring the same splits for each algorithm. For each instance reduction method we reduce the size k on the original training set from $S \in R^{k \times d+1}$ to $Z \in R^{k' \times d+1}$ where $k' < k$. We then apply 1NN for the given Z on the validation set. Recall that 1NN is given by $y' = \underset{r}{\operatorname{argmax}}\ I(r = y_1)$(Equation 1). We wrote our code in R.

Code	Dataset	IPRed+1NN	%	CNN+1NN	%	RNN+1NN	%	1NN	%
1	Soy Bean	2.5	64.4681	5	15.5319	5	14.6809	2.5	100
2	Colon Cancer	15.417	60.6452	18.75	35.4839	18.333	26.9355	15.417	100
3	Leukemia	**10**	60	16.8254	28.8889	20.794	24.5833	15.397	100
4	Breast Tissue	**35.625**	59.434	39.375	54.717	36.75	49.6226	41.75	100
5	Appendicitis	**18.125**	61.6981	21.125	33.3019	23.5	26.6038	18.5	100
6	Iris	4	60	6	12.1333	8	9.8667	4	100
7	Hayes-Roth	29.375	60	30	48.875	30.625	44.9375	29.375	100
8	Global Cancer Map	**38.947**	60	40	49.7368	41.579	44.7895	39.474	100
9	Glass Identification	**0.8**	60.5607	0.8762	7.2897	0.952	4.8131	0.876	100
10	Heart	**39.259**	60	41.1111	52.8519	41.852	47.4444	42.222	100
11	Haberman	29.444	59.4118	33.5556	47.4837	37.444	45.1961	**29.111**	100
12	Ecoli	**22.867**	60.1786	25.8042	37.5	27.063	33.869	23.823	100
13	Liver Disorders	**34.472**	60	41.9306	52.9275	39.578	47.3333	36.674	100
14	Bci	44	60	45.75	55	**42.5**	49.275	45.5	100
15	Saheart	**42.88**	59.9134	43.5507	53.5498	44.629	46.9048	43.551	100
16	Musk	15.905	60.1261	**13.9904**	27.563	15.054	21.9748	15.66841	100
17	Led	28.2	59	29	42.34	**26.8**	41.64	28.6	100
18	Climate	11.111	60	16.6667	24.1667	18.704	19.7963	11.111	100
19	Breast Cancer	9.316	60.1582	9.8764	14.9736	9.698	11.9332	**9.008**	100
20	Digits	**0.128**	60.0787	1.046	3.9895	1.704	3.1102	0.256	100
21	Diabetes	32.832	59.8958	34.8872	46.0026	35.207	39.3229	**32.531**	100
22	Vowel	36.263	60	37.7778	19.0202	40.101	17.5758	36.263	100
23	Statlog German Credit Card	34.5	60	34.6	49.38	36.7	47.39	**33.4**	100
24	Banknote Authentication	0.073	60.0437	0.219	1.7274	0.219	1.5306	0.073	100
25	Contraceptive	**51.942**	59.9593	53.9116	67.3999	54.314	67.1079	51.943	100
26	Wine Quality Red	**54.552**	60	56.7105	54.7154	57.697	49.7311	55.37	100
27	Ozone	11.323	60.0758	15.9808	22.1765	18.195	17.8073	**11.162**	100
28	Steel Plates Faults	**61.619**	60	62.443	65.4044	62.907	61.6383	61.722	100
29	Insurance Company COIL 2000	10.821	59.9966	13.1751	23.0831	13.243	22.6984	**10.564**	100
30	Magic Gamma Telescope	**14.732**	60	27.6656	35.469	28.465	30.643	24.022	100
	Average	24.701	60.1881	27.253	36.0894	27.92	32.3585	25.662	100

Table 2: Average cross-validation error and average storage percentage % of the algorithms on each of the 30 real datasets from the UCI(A. Asuncion 2007), KEEL(Alcalá et al. 2010), and Bioinformatics repositories(Jesús S. Aguilar-Ruiz). The algorithm with the minimum error is shown in bold.

	CNN+1NN	RNN+1NN	1NN
IPRed+1NN	0.000007	0.00001	0.02088
CNN+1NN		0.0226	0.00018
RNN+1NN			0.00044

Table 3: P-values of Wilcox rank test(two-tailed test) between all pairs of algorithms.

Experimental Results on Thirty Datasets

For each training-validation split in our classifications tasks we measure the error on the validation split as the number of instances incorrectly predicted divided by the number of instances. We then take the average result of 10 folds to be average cross-validation error. For each dataset we measure the storage percentage as the number of instances in the reduced training set divided by the number of instances in the training set and take the average result of 10 trials in the cross-validation to be average storage percentage. In Table 2 we tabulate average cross-validation error and average storage percentage % results on each dataset.

Over the 30 datasets IPRed+1NN yields the minimum average error of 24.701% and has the minimum error in 14 out of 30 datasets. The second best is 1NN that gives an average error of 25.662% and has the minimum error in 6 out of 30 datasets. The third best is CNN+1NN that gives higher average error of 27.253% and has the minimum error in 1 out of 30 datasets. RNN+1NN gives an average error of 27.92% and has the minimum error in 2 out of 30 datasets. RNN+1NN has the best average storage requirements of 32.3585%. The second best is CNN+1NN that gives average storage requirements of 36.0894%. The third best is IPRed which gives average storage requirements of 60.1881%. 1NN uses the original training set. Thus, 1NN gives average storage requirements of 100%.

To verify that differences in classification accuracy on the datasets are statistically significant we use Wilcoxon rank test(Japkowicz and Shah 2011; Kanji 2006) which is a standard test to measure the statistical significance between two methods in many datasets. It shows that one method is considered statistically significant than the other method if it outperforms the other method in many datasets. The p-values in Table 3 show that our method IPRed+1NN outper-

forms the other methods on the 30 datasets with statistical significance.

Discussion

IPRed+1NN divides the training set into 9 partitions each of the same size. For each training partition obtained from the training set we store the sum of the minimum distances between the training partition and the validation partition. We then construct the reduced dataset from the selected training partitions that are less than 75^{th} percentile of all the training partitions. Finally, we measure the performance of 1NN on the validation set using the given reduced training set. This approach outperforms CNN+1NN and RNN+1NN in terms of accuracy(results shown in Table 2 and Table 3).

In this study we chose 1NN as a classification algorithm due to its popularity and its efficiency for CNN and RNN algorithms. Other classifiers such as support vector machine(SVM) can be used which may outperform 1NN. However, optimizing SVM by performing cross-validation to select the best parameters increases the runtime. IPRed+1NN gives higher average storage percentage than CNN+1NN and RNN+1NN on the datasets except for steel plates faults and contraceptive datasets(results shown in Table 3). Thus, It is the slowest algorithm in the conducted experiments but still computationally tractable for the large real datasets. However, It gives better classification accuracy than the other methods most of the time. We used the standard package for CNN+1NN and RNN+1NN in R(Ripley, Venables, and Ripley 2013).

Conclusion

We present instance reduction algorithm based on the percentile of the partitions that outputs dataset of reduced size used by 1-nearest neighbor algorithm for classification. Our algorithm leads to better classification performance than the other popular methods by obtaining the minimum average error with statistical significance on many real datasets.

References

A. Asuncion, D. N. 2007. UCI machine learning repository.

Alcalá, J.; Fernández, A.; Luengo, J.; Derrac, J.; García, S.; Sánchez, L.; and Herrera, F. 2010. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing* 17:255–287.

Alpaydin, E. 1997. Voting over multiple condensed nearest neighbors. *Artificial Intelligence Review* 11(1-5):115–132.

Bhatia, N., et al. 2010. Survey of nearest neighbor techniques. *arXiv preprint arXiv:1007.0085*.

Chou, C.-H.; Kuo, B.-H.; and Chang, F. 2006. The generalized condensed nearest neighbor rule as a data reduction method. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 2, 556–559. IEEE.

Gates, G. 1972. The reduced nearest neighbor rule (corresp.). *Information Theory, IEEE Transactions on* 18(3):431–433.

Hart, P. 1968. The condensed nearest neighbor rule (corresp.). *Information Theory, IEEE Transactions on* 14(3):515–516.

Japkowicz, N., and Shah, M. 2011. *Evaluating Learning Algorithms*. Cambridge University Press.

Jesús S. Aguilar-Ruiz. Dataset repository. available at www.upo.es/eps/aguilar/datasets.html.

Kanji, G. K. 2006. *100 statistical tests*. Sage.

Olvera-López, J. A.; Carrasco-Ochoa, J. A.; Martínez-Trinidad, J. F.; and Kittler, J. 2010. A review of instance selection methods. *Artificial Intelligence Review* 34(2):133–143.

Ripley, B.; Venables, W.; and Ripley, M. B. 2013. Package class.

Wilson, D. R., and Martinez, T. R. 2000. Reduction techniques for instance-based learning algorithms. *Machine learning* 38(3):257–286.

Wu, X.; Kumar, V.; Quinlan, J. R.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A.; Liu, B.; Philip, S. Y.; et al. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14(1):1–37.