Restricted Sequential Floating Search Applied to Object Selection

J. Arturo Olvera-López, J. Francisco Martínez-Trinidad, and J. Ariel Carrasco-Ochoa

Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro No. 1, Sta. María Tonantzintla, Puebla, CP: 72840, Mexico
{aolvera, fmartine, ariel}@ccc.inaoep.mx

Abstract. The object selection is an important task for instance-based classifiers since through this process the size of a training set could be reduced and then the runtimes in both classification and training steps would be reduced. Several methods for object selection have been proposed but some methods discard relevant objects for the classification step. In this paper, we propose an object selection method which is based on the idea of sequential floating search. This method reconsiders the inclusion of relevant objects previously discarded. Some experimental results obtained by our method are shown and compared against some other object selection methods.

1 Introduction

In supervised classification, a training or sample set (denoted in this paper as T) containing objects (previously assessed) described by a set of values (features) is used for classifying new objects. Commonly T contains objects with non relevant information for classifiers, therefore it is necessary to apply an object selection method over T in order to detect and retain those relevant objects for classification.

Object selection is important for instance-based classifiers because for this kind of classifiers the runtime in training and classification steps depends on the size of the training set. Thus, through the object selection, the runtimes in both training and classification steps could be reduced since these steps are applied over an object subset S ($S \subset T$) instead of using the whole set T.

Sequential search is a method used for finding a sub-optimal solution of a selection problem. This kind of search for selecting consists in evaluating at each step the relevance of each possibility in the partial solution set. This search can be done in the forward or backward direction, the forward search starts with an empty solution set and at each step it evaluates all options and includes the best one. The backward search starts with the whole set and at each step it excludes the worst element. These sequential methods analyze at each step all possibilities for including/excluding one of them but they cannot exclude/include solutions previously included/excluded, this is possible in the sequential floating methods [5] which include/exclude solutions (previously excluded/included) after each inclusion/exclusion.

P. Perner (Ed.): MLDM 2007, LNAI 4571, pp. 694–702, 2007.

Sequential search has been used for the feature selection problem [5, 6] and extended for the object selection problem in [3].

In this paper, we propose a sequential method for object selection. Our method re-considers the inclusion to *S* of relevant objects previously discarded in the selection process, so that *S* would include those objects that contribute for improving the quality in *S*.

This paper has been structured as follows: in section 2 we describe some relevant object selection methods. In section 3 we introduce our object selection method, in section 4 we report comparative results obtained by our method and other object selection methods. Finally, in section 5 conclusions and future work are given.

2 Related Work

One of the first proposed methods for object selection is the *ENN* (*Edited Nearest Neighbor)* [1]. This method is commonly used as noise filter because it deletes noisy objects, that is, objects with a different class in a neighborhood. The *ENN* rule consists in discarding from *T* those objects that do not belong to their *k* nearest neighbors' class.

In [2] the *DROP* (*Decremental Reduction Optimization Procedure*) methods were proposed. The selection criterion in *DROP* methods is based on the concept of *associate*. The *associates* of an object *O* are those objects such that *O* is one of their *k* nearest neighbors. *DROP1* starts with *S*=*T* and discards the object *O* if its associates in *S* can be classified correctly without *O*. *DROP2* considers the effect of the removal of an object on *T, DROP2* discards *O* if its associates in *T* can be classified correctly without *O*. *DROP3* and *DROP4* apply a noise filter (similar to *ENN*) before starting the selection process. Finally, *DROP5* modifies *DROP2* so that the selection process starts with the nearest enemies (nearest objects with different class).

The sequential search has been used for selecting objects. In [3] the *BSE* (*Backward Sequential Edition*) method was proposed. *BSE* applies the backward sequential search to the object selection problem. This method sequentially analyzes the relevance of each object in the partial object subset and at each step *BSE* discards the object that its deletion maximizes the classification accuracy. This selection process is repeated until the accuracy decreases. *BSE* is an expensive method since at each step it analyzes the impact of excluding each object in the sample.

In [4] the edition schemes *ENN+BSE* and *DROP+BSE* were proposed. These schemes apply a pre-processing step before the selection process using *BSE* so that *BSE* is used over previously reduced object sets. *ENN* and *DROP3,…,DROP5* methods are used by *ENN+BSE* and *DROP+BSE* respectively in the pre-processing step.

3 Proposed Method

Our object selection method is based on the idea of the Sequential Floating Selection (*SFS*) [5], which reconsiders the inclusion/exclusion (in the partial subset) of objects previously discarded/included. *SFS* consists in applying conditional inclusion/exclusion steps after each exclusion/inclusion in the set. This kind of search (as sequential search) can be done in the backward and forward directions.

The backward *SFS* consists in applying after each exclusion step a number of inclusion steps as long as the classification results are better than the previously evaluated ones. The forward *SFS* is the counterpart of backward *SFS*. These floating searches are very expensive therefore we propose an object selection method based on the backward *SFS* but in a restricted way.

Our method named *Restricted Floating Object Selection* (*RFOS*) applies an exclusion process followed by the conditional inclusion of discarded objects. The *RFOS* method is shown in figure 1.

```
RFOS (Training sample T)
Let S= subset obtained after applying ENN or DROPs over T
Best_val =Classif(S)
Repeat                         //exclusion process
        Worst=null
        For each object O in S
          S'=S-{O}
          Eval = Classif(S')
          If Eval ≥ Best_val
              Worst=O
              Best_val=Eval
        If Worst ≠ null
            S=S-{Worst}
Until Worst==null or |S|==1
D=T-S
For each object O_i in D    //conditional inclusion
   S'' = S ∪ {O_i}
   Eval = Classif(S'')
   If Eval >Best_val
      Best_val = Eval
      S=S ∪ {O_i}
Return S
```

Fig. 1. *RFOS* method for object selection

RFOS starts applying a pre-processing step followed by the exclusion process and finally the conditional inclusion is applied over the object set previously selected (S, $S \subset T$). The exclusion step sequentially discards objects in the partial set. This step analyzes the classification contribution of each object and at each step it excludes the object (*Worst*) with the smallest contribution for the subset quality, in terms of the accuracy of a classifier, which is calculated by the *Classif* function.

The selection process in *RFOS* consists in analyzing (conditional inclusion) the objects discarded from T (objects in the set D=T-S) for including in S those objects that their inclusion improves the classification, that is, an object $O \in D$ is included in S only if the classification obtained using $S \cup \{O\}$ is better than the obtained using S.

To know whether the classification after the inclusion is better or not, *RFOS* uses a classifier (*Classif* function in figure 1) to evaluate the quality of the sets.

In this work we use *ENN* or *DROP* methods for the pre-processing step but any other object selection method can be used for that step.

The *RFOS* is a restricted floating search method because first it applies only an exclusion process followed by the conditional inclusion. This restricted floating method can be done in the inverse direction (*RFOS-Inv*), that is, first applying an inclusion process followed by the conditional exclusion. The *RFOS-Inv* method is shown in figure 2.

```
RFOS-Inv (Training sample T)
 Let S= subset obtained after applying ENN or DROPs over T
 Best_val =Classif(S)
 D=T-S
 For each object O in D    // inclusion process
    S' = S ∪ {O}
    Eval = Classif(S')
    If Eval >Best_val
         Best_val = Eval
         S=S ∪ {O}
 Best_val =Classif(S)   //conditional exclusion
 Repeat
         Worst=null
         For each object O in S
            S''=S-{O}
         Eval = Classif(S'')
            If Eval ≥ Best_val
                 Worst=O
                 Best_val=Eval
         If Worst ≠ null
              S=S-{Worst}
 Until Worst==null or |S|==1
 Return S
```

Fig. 2. *RFOS-Inv* method for object selection

4 Experimental Results

In this section, we show the results obtained by *RFOS* and *RFOS-Inv* over nine datasets obtained from the UCI repository [7] and compare them against *ENN+BSE* and *DROP+BSE* methods.

In all the tables shown in this section, for each method, we show the classification accuracy (*Acc.*) and the percentage of the original training set that was retained by each method (*Str.*), that is $100|S|/|T|$. In addition, we show the classification obtained using the original training set (*Orig.*) and the average results over the nine datasets at the bottom. Also we show the average accuracy difference (*Average diff*) with respect to the original accuracy. In all the experiments 10 fold cross validation was used.

The results obtained by *ENN+BSE* and *DROP+BSE* methods over the datasets are shown in table 1. In table 2 we report the results obtained by *RFOS* using *ENN* and *DROP* methods in the pre-processing step. In table 2, *RFOS(ENN)* is the *RFOS* method using *ENN* for the pre-processing step and by analogy for *RFOS(DROP3),…,* *RFOS(DROP5)*, the *DROP3,…DROP5* methods were respectively used. Table 3 shows the results obtained applying *RFOS-Inv* method. In tables 1-3 we used as distance function the *Heterogeneous Value Difference Metric* (*HVDM*) [2].

Table 1. Classification (*Acc.*) and retention (*Str.*) results obtained by: original sample (*Orig.*), *ENN+BSE* and *DROP3+BSE…DROP5+BSE* methods

Dataset	Orig.		ENN+BSE		DROP3+BSE		DROP4+BSE		DROP5+BSE	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Bridges	37.91	100	30.27	51.27	35.45	8.42	36.72	12.02	35.81	14.79
Glass	71.42	100	69.41	21.81	59.78	14.95	59.78	17.18	54.24	15.21
Iris	93.33	100	93.00	8.00	88.00	6.42	88.00	6.64	89.33	6.39
Liver	65.22	100	57.67	26.69	59.77	10.91	61.21	12.36	54.95	11.75
Sonar	86.19	100	71.19	27.24	81.42	12.60	84.83	14.79	84.30	15.17
Tae	51.08	100	46.66	43.85	47.70	14.93	50.00	18.17	46.66	20.08
Thyroid	95.45	100	93.09	5.63	91.19	4.28	91.16	4.39	88.29	3.51
Wine	94.44	100	92.74	8.17	96.07	5.05	96.07	5.05	96.07	4.43
Zoo	91.33	100	91.11	12.59	77.77	11.72	77.77	11.97	83.33	7.76
Average	**76.26**	**100**	**71.68**	**22.81**	**70.79**	**9.92**	**71.73**	**11.40**	**70.33**	**11.01**
Average diff			**-4.58**		**-5.47**		**-4.54**		**-5.93**	

Table 2. Classification (*Acc.*) and retention (*Str.*) results obtained by *RFOS*

Dataset	Orig.		RFOS(ENN)		RFOS(DROP3)		RFOS(DROP4)		RFOS(DROP5)	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Bridges	37.91	100	32.00	58.33	36.45	15.61	35.45	18.29	35.81	24.33
Glass	71.42	100	69.43	29.34	64.48	25.75	65.41	27.46	67.74	26.11
Iris	93.33	100	93.33	10.07	93.00	9.92	93.33	10.29	93.33	10.00
Liver	65.22	100	59.98	33.68	61.70	16.94	65.00	18.39	60.03	19.64
Sonar	86.19	100	72.57	32.27	84.64	21.58	83.52	20.88	83.73	22.59
Tae	51.08	100	50.70	48.88	47.70	25.45	50.00	27.29	53.33	31.34
Thyroid	95.45	100	94.04	7.02	93.98	6.25	94.45	6.77	90.47	5.94
Wine	94.44	100	93.63	10.23	94.44	8.17	94.44	8.17	93.85	8.30
Zoo	91.33	100	91.33	71.14	91.33	14.81	91.33	14.69	91.11	14.93
Average	**76.26**	**100**	**73.00**	**33.44**	**74.19**	**16.05**	**74.77**	**16.91**	**74.38**	**18.13**
Average diff			**-3.26**		**-2.07**		**-1.49**		**-1.89**	

Table 3. Classification (*Acc.*) and retention (*Str.*) results obtained by *RFOS-Inv*

Dataset	Orig.		RFOS-Inv(ENN)		RFOS-Inv(DROP3)		RFOS-Inv(DROP4)		RFOS-Inv(DROP5)	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Bridges	37.91	100	30.54	25.06	35.54	13.23	35.09	16.12	35.54	18.08
Glass	71.42	100	58.35	19.47	43.50	23.36	54.95	20.77	55.49	16.45
Iris	93.33	100	80.66	5.33	92.66	7.18	92.00	7.25	86.00	6.29
Liver	65.22	100	58.84	21.80	59.75	19.25	61.20	20.58	60.30	19.54
Sonar	86.19	100	71.14	16.77	68.26	18.58	70.66	21.20	68.76	22.49
Tae	51.08	100	50.97	14.27	46.66	20.27	45.54	31.86	54.20	30.17
Thyroid	95.45	100	88.83	3.82	94.55	4.85	93.03	5.27	86.96	4.18
Wine	94.44	100	90.00	4.36	88.23	5.18	89.44	5.36	91.04	4.43
Zoo	91.33	100	91.11	15.92	90.00	13.58	78.88	13.45	80.00	14.19
Average	**76.26**	**100**	**68.94**	**14.09**	**68.79**	**13.94**	**68.98**	**15.76**	**68.70**	**15.09**
Average diff			**-7.33**		**-7.47**		**-7.29**		**-7.56**	

The runtimes of the experiments reported in tables 1-3 are shown in table 4. Based on the average results, we can observe that because of the inclusion/exclusion steps in *RFOS* and *RFOS-Inv*, their runtimes are higher than the *ENN+BSE* and *DROPs+BSE*.

Table 4. Runtimes (in seconds) spent by the methods shown in tables 1-3

Dataset	ENN+BSE	DROP3+BSE	DROP4+BSE	DROP5+BSE	RFOS (ENN)	RFOS (DROP3)	RFOS (DROP4)	RFOS (DROP5)	RFOS-Inv (ENN)	RFOS-Inv (DROP3)	RFOS-Inv (DROP4)	RFOS-Inv (DROP5)
Bridges	595.3	8.3	13.5	8.6	608.4	31.4	37.1	30.9	379.8	35.4	69.8	38.7
Glass	540.0	14.7	28.0	14.2	545.3	27.0	35.8	25.7	215.6	59.8	46.1	33.7
Iris	420.1	4.5	3.8	2.1	426.9	9.9	9.1	6.3	482.1	7.5	7.9	8.0
Liver	1203.8	68.3	48.9	63.8	1214.0	95.8	74.0	91.5	1211.9	87.6	124.9	120.6
Sonar	1496.8	64.1	65.5	60.3	1509.6	109.6	85.3	83.6	1512.0	142.1	140.7	183.7
Tae	49.6	7.9	13.4	12.8	50.3	11.4	16.5	17.9	55.7	15.3	22.3	26.5
Thyroid	1381.6	2.8	2.2	2.5	1393.1	12.7	11.3	12.8	1140.2	11.4	12.9	12.2
Wine	960.4	5.0	5.3	3.8	969.7	13.7	14.1	12.4	905.7	12.6	14.1	13.5
Zoo	1380.6	6.3	6.5	6.0	1402.9	18.9	15.5	20.4	1020.6	13.1	16.3	17.3
Average	**892.02**	**20.21**	**20.79**	**19.34**	**902.24**	**36.71**	**33.19**	**33.50**	**769.29**	**42.76**	**50.56**	**50.47**

The classifier used in the results shown in tables 1-3 was *k-NN* (*k*=3). The average results reported in tables 1-3 are depicted in figure 3, which shows a scatter graphic of retention (vertical axis) versus accuracy (horizontal axis). On this graphic, the most located at right the best classification accuracy and the most located at bottom the best retention percentage.

Based on results shown in tables 1-3 and figure 3, we can observe that *RFOS* outperformed to *RFOS-Inv* because this method discards relevant objects in the final exclusion step. In addition, the accuracy obtained by *RFOS* is better than the obtained by *ENN+BSE* and *DROP+BSE* schemes; this is because *RFOS* includes relevant objects discarded in the exclusion steps. As a consequence of the final inclusion step, the object sets obtained by *RFOS* are slightly bigger than those obtained by *RFOS-Inv, ENN+BSE* and *DROP+BSE*. In this experiment the best accuracy was obtained by *RFOS(DROP4)* in the average case (figure 3).

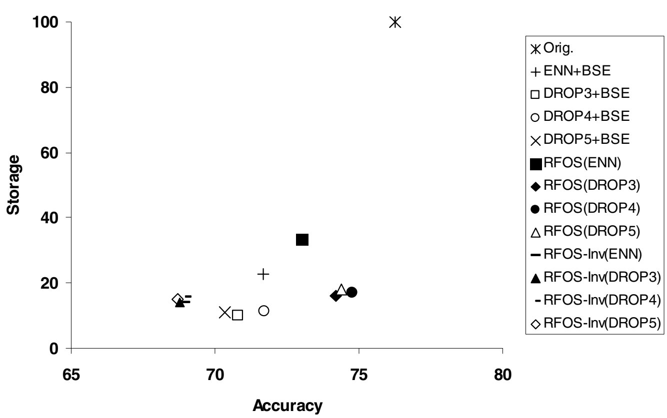

Fig. 3. Scatter graphic from results obtained in tables 1, 2 and 3

In the above results the classifier used was *k-NN*, but it is important to know the performance of the proposed object selection methods using other classifiers. Therefore we applied *RFOS* (the best restricted floating method in above experiments) and *ENN+BSE, DROP+BSE* using *LWR (Locally Weighted Regression)* and *SVM (Support Vector Machines)* classifiers during the selection process (notice that *RFOS, ENN+BSE* and *DROP+BSE* allow us to use any classifier different from *k-NN* in the selection process). In this experiment we have tested only numeric datasets because the classifiers are restricted to this kind of data.

Table 5. Classification (*Acc.*) and retention (*Str.*) results obtained by: original sample (*Orig.*), *ENN+BSE* and *DROP3+BSE…DROP5+BSE* methods using *LWR*

Dataset	Orig.		ENN+BSE		DROP3+BSE		DROP4+BSE		DROP5+BSE	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Glass	57.85	100	56.84	50.26	50.71	20.83	55.18	25.54	53.72	21.97
Iris	98.00	100	96.66	20.74	88.00	10.88	88.66	11.18	88.66	8.14
Liver	70.12	100	66.33	31.51	70.99	17.13	68.08	19.00	68.68	16.58
Sonar	64.40	100	65.36	73.29	63.98	21.37	69.26	28.26	63.88	25.21
Thyroid	91.16	100	57.84	51.06	86.10	19.22	87.03	23.66	89.78	18.04
Wine	92.15	100	88.88	57.50	90.96	14.10	88.20	14.10	88.28	9.36
Average	**78.95**	**100**	**71.99**	**47.39**	**75.12**	**17.26**	**76.07**	**20.29**	**75.50**	**16.55**
Average diff			**-6.96**		**-3.82**		**-2.88**		**-3.45**	

Table 6. Classification (*Acc.*) and retention (*Str.*) results obtained by *RFOS* using *LWR*

Dataset	Orig.		RFOS(ENN)		RFOS(DROP3)		RFOS(DROP4)		RFOS(DROP5)	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Glass	57.85	100	57.79	52.18	53.30	25.13	58.33	26.26	54.54	27.09
Iris	98.00	100	97.33	22.00	96.00	13.40	95.33	13.77	95.33	10.00
Liver	70.12	100	66.34	37.61	73.31	18.64	71.27	21.22	71.30	18.77
Sonar	64.40	100	66.81	74.36	71.00	31.94	65.35	25.96	68.78	30.12
Thyroid	91.16	100	58.66	51.99	91.21	22.58	91.62	25.93	91.19	19.90
Wine	92.15	100	90.62	58.75	90.98	16.22	90.58	16.22	90.73	16.15
Average	**78.95**	**100**	**72.93**	**49.48**	**79.30**	**21.32**	**78.75**	**21.56**	**78.65**	**20.34**
			-6.02		**0.35**		**-0.20**		**-0.30**	

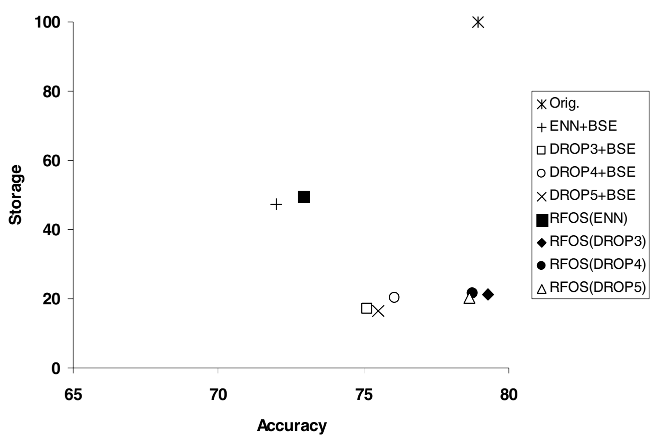

Fig. 4. Scatter graphic from results obtained using *LWR*

In tables 5 and 6 we show the accuracy and retention results obtained using the *LWR* classifier and the average scatter graphic from these results is depicted in figure 4.

Based on tables 5 and 6 we can observe that in all cases *RFOS* outperformed *ENN+BSE* and *DROP+BSE* methods. Figure 4 shows that in the average case, the best method using *LWR* was *RFOS(DROP3)* and the accuracy obtained by the other *RFOS(DROP)* methods was slightly lower than the obtained by the original set.

Also the *SVM* classifier was used for testing *RFOS, ENN+BSE* and *DROP+BSE* methods. These results are shown in tables 7-8 and the average results are depicted in figure 5.

Table 7. Classification (*Acc.*) and retention (*Str.*) results obtained by: original sample (*Orig.*), *ENN+BSE* and *DROP3+BSE…DROP5+BSE* methods using *SVM*

Dataset	Orig.		ENN+BSE		DROP3+BSE		DROP4+BSE		DROP5+BSE	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Glass	65.34	100	66.82	40.82	61.31	17.29	64.87	23.73	61.90	15.42
Iris	96.00	100	96.00	8.89	93.33	3.70	94.00	4.07	94.67	3.33
Liver	69.91	100	69.88	35.31	62.07	17.61	65.84	14.90	63.80	20.32
Sonar	79.38	100	78.60	58.07	72.83	13.68	74.48	14.42	71.57	15.12
Thyroid	72.61	100	72.61	7.23	68.34	3.20	68.20	3.36	67.27	3.31
Wine	97.18	100	96.63	21.68	93.89	3.62	94.97	3.87	92.09	2.75
Average	**80.07**	**100**	**80.09**	**28.67**	**75.30**	**9.85**	**77.06**	**10.73**	**75.22**	**10.04**
Average diff			**0.02**		**-4.78**		**-3.01**		**-4.85**	

Table 8. Classification (*Acc.*) and retention (*Str.*) results obtained by *RFOS* using *SVM*

Dataset	Orig.		RFOS(ENN)		RFOS(DROP3)		RFOS(DROP4)		RFOS(DROP5)	
	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.	Acc.	Str.
Glass	65.34	100	69.18	43.24	62.26	20.16	64.95	25.50	63.87	20.14
Iris	96.00	100	96.00	9.78	93.33	4.14	94.00	4.14	94.67	3.70
Liver	69.91	100	69.83	47.68	62.95	20.52	67.03	19.62	67.83	21.71
Sonar	79.38	100	78.90	58.19	74.42	15.48	74.48	14.90	73.16	16.86
Thyroid	72.61	100	72.61	8.16	69.07	5.42	69.23	3.77	69.59	5.78
Wine	97.18	100	96.75	22.75	95.55	5.80	95.55	5.86	92.64	4.80
Average	**80.07**	**100**	**80.55**	**31.63**	**76.26**	**11.92**	**77.54**	**12.30**	**76.96**	**12.17**
Average diff			**0.47**		**-3.81**		**-2.53**		**-3.11**	

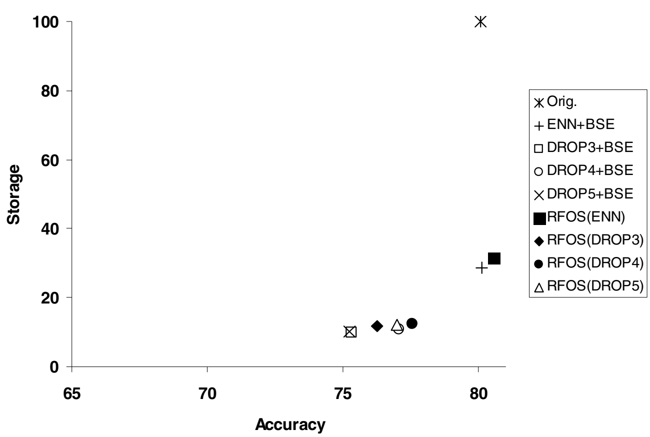

Fig. 5. Scatter graphic from results obtained using *SVM*

According to the results reported in tables 7 and 8, we can notice that using the *SVM* classifier, in all the experiments, again *RFOS* outperformed the *ENN+BSE* and *DROP+BSE* schemes. Figure 5 shows that the best accuracy results using this classifier were obtained by *RFOS(ENN).*

Based on the results shown in this section, we can observe that the proposed method obtain smaller subsets (with respect to the original size set) without a significantly accuracy reduction. The main benefit of using the subsets obtained is the reduction in training and classification stages for instance-based classifiers.

5 Conclusions

Object selection is an important task for instance-based classifiers since through this process the training set is reduced and also the runtimes in classification and training steps.

Several object selection methods which sequentially discard objects have been proposed, for example, the edition schemes *ENN+BSE* and *DROP+BSE*. It is possible that during the selection process, these methods remove relevant objects for the classification accuracy. In this work, we proposed the *RFOS* method which is an object selection method that includes those relevant objects discarded by the edition schemes.

The experiments show that *RFOS* outperforms *RFOS-Inv, ENN+BSE* and *DROP+BSE* (not only for *k-NN* but also for *LWR* and *SVM*)*,* that is*,* the inclusion of some discarded objects helps to improve the classification.

RFOS is a restricted floating sequential method because it applies only an exclusion process followed by the conditional inclusion, therefore, as future work we will adapt a full floating sequential search for solving the object selection problem.

References

1. Wilson, D.L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Transactions on Systems, Man, and Cybernetics 2(3), 408–421 (1972)
2. Randall, W.D., Martínez, T.R.: Reduction Techniques for Instance-Based Learning Algorithms. Machine Learning 38, 257–286 (2000)
3. Olvera-López, J.A., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F.: Sequential Search for Decremental Edition. In: Gallagher, M., Hogan, J.P., Maire, F. (eds.) IDEAL 2005. LNCS, vol. 3578, pp. 280–285. Springer, Heidelberg (2005)
4. Olvera-López, J.A., Martínez-Trinidad, J.F., Carrasco-Ochoa, J.A.: Edition Schemes based on BSE. In: Sanfeliu, A., Cortés, M.L. (eds.) CIARP 2005. LNCS, vol. 3773, pp. 360–367. Springer, Heidelberg (2005)
5. Pudil, P., Ferri, F.J., Novovičová, J., Kittler, J.: Floating Search Methods for Feature Selection with Nonmonotonic Criterion Functions. In: Proceedings of the 12th International Conference on Pattern Recognition, pp. 279–283. IEEE Computer Society Press, Los Alamitos (1994)
6. Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence 97, 245–271 (1997)
7. Blake, C., Keogh, E., Merz, C.J.: UCI repository of machine learning databases. In: Department of Information and Computer Science, University of California, Irvine, CA (1998) http://www.ics.uci.edu/ mlearn/MLRepository.html