Adapting noise filters for ranking

Adapting Noise Filters for Ranking

Ana Carolina Lorena
Universidade Federal de São Paulo
ICT-UNIFESP, São José dos Campos
Email: aclorena@unifesp.br

Luís Paulo Faina Garcia
Universidade de São Paulo
ICMC-USP, São Carlos
Email: lpgarcia@icmc.usp.br

André C. P. L. F. de Carvalho
Universidade de São Paulo
ICMC-USP, São Carlos
Email: andre@icmc.usp.br

***Abstract*—Noise filtering can be considered an important pre-processing step in the data mining process, making data more reliable for pattern extraction. An interesting aspect for increasing data understanding would be to rank the potential noisy cases, in order to evidence the most unreliable instances to be further examined. Since the majority of the filters from the literature were designed only for hard classification, distinguishing whether an example is noisy or not, in this paper we adapt the output of some state of the art noise filters for ranking the cases identified as suspicious. We also present new evaluation measures for the noise rankers designed, which take into account the ordering of the detected noisy cases.**

Keywords*—*Noisy data; Label Noise; Pre-processing.

I. Introduction

Noise filtering is a pre-processing step that seeks to identify and remove unsafe examples from a dataset. Thereby, Machine Learning (ML) models induced from a filtered dataset can be more accurate and less complex to learn [1]. There are many noise filtering techniques in the literature [2]. They employ different approaches for identifying the suspicious cases. For instance, while some filters regard on differences in the predictions made by distinct models induced from data [3], [4], others seek if simpler concepts can be extracted by the removal of some examples [5], [6].

In practice, it would be interesting not only to identify, but also to rank the potential noisy cases, evidencing the most unreliable instances. These examples could then be further examined by a domain expert. This knowledge can also support the development of new noise tolerant ML algorithms [2]. Despite this, the majority of the noise filters only point examples as noisy or not. A notable recent exception is [6], where noisy cases from a medical dataset are ordered.

In this paper we deal with the slightly modified problem of noise ranking, where the examples of the dataset must be ordered according to their unreliability. Therefore, safe examples, that is, those that are core for knowledge discovery, will be ideally positioned in the bottom of the ordered lists. On the other hand, unreliable cases, which do not properly represent the data patterns, will be top ranked. For evaluating the efficacy of the noise rankers, we also present new evaluation measures which take into account the orderings produced.

II. Noise Filtering

When dealing with classification problems, ML algorithms are fed with a dataset containing n pairs $(\vec{x}_i, y_i)$. Each $\vec{x}_i$ is an example described by m predictive features, while y_i corresponds to the expected label or class of $\vec{x}_i$. Using this information, the learning algorithm induces a classification model able to predict the label of new examples. Nonetheless, real data is usually defective and contains noise, which for classification datasets can be found in the predictive features and/or the labels [7]. Noise in predictive features is usually consequence of incorrect, absent or unknown values. Label noise can be caused by errors or subjectivity in the data labeling process. Since most **of** the existent ML algorithms minimize a cost function based on training data misclassification, the reliability of the class information has a major impact on the classifiers performance. For this reason, in this paper we address label noise only.

Learning in the presence of label noise can be accomplished by different approaches [2]. One of them is to modify the learning algorithm to make it more robust against noise. This is performed when Decision Trees (DT) are pruned during learning, for instance [8]. Other work try to obtain noise-tolerant classifiers by learning a label noise model jointly to the classification model [9]. In this case some information must be available about the label noise or its effects [2], [10]. The learning algorithm can also be modified to embed a data cleansing step [11]. Finally, filters can be applied in a pre-processing step. Noise filters scan the training dataset for unreliable data [4], [6] and usually remove them afterwards. In this process, each training example is either regarded as potential noise or not.

One common approach for label noise identification and filtering is to compare the predictions of multiple classifiers on the training data [3], [4]. If distinct classifiers disagree on their predictions for an instance, then it is probably incorrect. In [3] a majority voting of the ten fold cross validation predictions made by k-nearest neighbor (k-NN) [12], DT [8] and Support Vector Machines (SVM) [13] classifiers is employed in noise identification. Since the set of classifiers in this ensemble for noise identification is fixed, we will refer to this technique as Static Ensemble Filter (SEF). On the other hand, in [4] the set of classifiers is dynamically adapted to each dataset. The Dynamic Ensemble Filter (DEF) chooses a set of classifiers with best ten fold cross validation predictive performance on training data to be combined in noise identification. Afterwards, a majority voting of the predictions is used to assess whether an example is noisy, as in [3]. Another ensemble for noise filtering is the High Agreement Random Forest Filter (HARF) [6], which employs a Random Forest (RF) in noise identification. For such, it considers the rate of disagreement in the predictions made by the individual trees using ten fold cross validation. If this rate is high for a given example, it is probably noisy.

978-1-5090-0016-6/15 $31.00 © 2015 IEEE
DOI 10.1109/BRACIS.2015.58

In [5] an algorithm named Saturation Filter (SF) is proposed. It uses the first order language representation of a dataset and then exhaustively looks for examples that reduce a value named Complexity of the Least Correct Hypothesis (CLCH) associated with data. At each step, one potential noisy example is chosen and the algorithm verifies if the CLCH value can be reduced by removing this particular example. This procedure is iterated until no example is signalized as noisy or until a stop criterion is reached. Sluban et al. [6] proposed a modification to reduce the computational cost of SF, named Pruned SF (PruneSF), which uses a DT to estimate the CLCH value. An initial step considers as noisy all examples misclassified by a pruned DT. They are then removed from the dataset and the iterations of the SF step are performed. Within them, the CLCH value of an example is estimated by the size difference between DTs induced when the example is employed for learning or disregarded.

Some popular noise filtering algorithms are based on the hypothesis of similarity between examples from the same class and employ the k-NN algorithm [14], [15]. They consider an example safer if it is close to other examples from its class. Otherwise, it is either incorrectly labelled or in the decision border. Examples in the frontier of the classes can be also considered unsafe, since small perturbations in a borderline example can move it to a wrong class. Therefore, distance-based filters usually remove both noisy and borderline examples, increasing the margin of separation between classes. The *All-k-NN* (AENN) algorithm is a representative of this category [14], which applies the k-NN classifier for several increasing values of k [15]. At each iteration, examples that have the majority of their neighbors from other classes are signalized as noisy.

In this study, we will adapt the previous five filters, that are well known filters with different biases in noise identification, for noise ranking.

III. Noise Ranking

When filters are employed for noise detection, a hard decision is obtained of whether an example is noisy. Rankers can provide a soft decision instead. In noise ranking, the objective is to order a dataset according to the reliability of its examples. This reliability can be estimated by different strategies, as in noise filtering. An example that contains core knowledge for pattern discovery should be evaluated as highly reliable, while those examples that do not follow the general patterns of the dataset should be considered unsafe. Obtaining such ordering of the examples can be considered interesting for various reasons. One of them is to evidence the most problematic examples in a dataset. These instances can then be further examined by a domain specialist and improve data understanding. A noticeable relate work is [6], where an ensemble of noise detection algorithms named NoiseRank was applied to a medical coronary heart disease dataset. Interestingly, the top-ranked instances were either incorrectly diagnosed patients or worth noting outlier cases. NoiseRank takes into account the agreement level of different filters in pointing an example as noisy. In this paper we employ a different approach and adapt the output of each individual filter for ranking. Knowing which are the most problematic examples can also support the development of new noise tolerant ML techniques. In [16], for example, an estimate of instance hardness is used to adapt the training algorithm of an Artificial Neural Network, so that hard instances have a lower weight on the back-propagation error function update. The same authors consider noisy instances as hard and design a new filter based on their instance hardness measure. This measure considers an instance hard if it is misclassified by a diverse set of classification algorithms. This is also the assumption of most ensemble-based filters in noise identification. In this paper we adapt the output of the noise filters described in Section II for noise ranking. All of them are adapted to provide an estimate of the probability of an example being noisy. This probability can also be regarded as the unreliability level ("noise level") of the example. The probability values obtained are then employed in the ranking process, such that top-ranked instances will be those most unreliable and probably noisy.

For the ensemble based techniques SEF and DEF, we estimate such probability by the percentage of disagreement between the predictions of the classifiers combined, so that examples misclassified by more classifiers will be considered unsafer and will be top-ranked. For HARF the noise level of an example is given by the percentage of base trees that disagree on their predictions for that particular instance. In the case of PruneSF, we have two steps. Firstly all examples pruned by the initial DT induced are equally ranked first, that is, they are assigned a probability of 1 of being noisy. Next, the examples are ranked according to their CLCH values, that give the confidence estimate. The CLCH values are also normalized to give a probability estimate. Since AENN runs a k-NN algorithm for various k values, we estimate the reliability level of an example as the percentage of times it is signalized as noisy.

IV. Experiments

This section presents the experiments performed in this study to assess the performance of the noise rankers.

A. Datasets

Table I describes the 53 datasets from the UCI repository [17] employed in the experiments. It summarizes the main characteristics of these datasets: number of examples (#EX), number of features (#FT), number of classes (#CL) and percentage of the examples in the majority class (%MC).

Noisy versions of the datasets from Table I were created by using the *uniform random addition method*. This systematic model of noise imputation ensures that each example has the same probability of having its label exchanged by another label [18]. For each dataset, noise was added at rates of 5%, 10%, 20% and 40%. Since the selection of the examples to be corrupted is random, 10 different noisy versions of the datasets were generated, for each noise rate.

B. Parameters

The filters from Section II and the rankers from Section III were employed in the identification of the noisy examples. Both SEF and DEF employ a majority voting of three base classifiers for noise identification and ranking. The classifiers combined by SEF are 3-NN, DT and SVM (with linear Kernel), as suggested by [3]. DEF chooses the set of

TABLE I. SUMMARY OF DATASETS CHARACTERISTICS

Dataset	#EX	#FT	#CL	%MC
abalone	4153	9	19	16
accute-inflammations	120	8	2	58
appendicitis	106	8	2	80
australian	690	15	2	56
backache	180	32	2	86
balance	625	5	3	46
banana	5300	3	2	55
blood-transfusion-service	748	5	2	76
breast-tissue	106	10	6	21
bupa	345	7	2	58
car	1728	7	4	70
cardiotocography	2126	21	10	27
cmc	1473	10	3	43
collins	485	22	13	16
connectionist-mines-vs-rocks	208	61	2	53
crabs	200	6	2	50
expgen	207	80	5	58
flags	178	29	5	34
flare	1066	12	6	31
glass	205	10	5	37
habermans-survival	306	4	2	74
hayes-roth	160	5	3	41
ionosphere	351	34	2	64
iris	150	5	3	33
kr-vs-kp	3196	37	2	52
led7digit	500	8	10	11
leukemia-haslinger	100	51	2	51
molecular-promoters	106	58	2	50
monk1	556	7	2	50
monk2	601	7	2	66
movement-libras	360	91	15	07
newthyroid	215	6	3	70
page-blocks	5473	11	5	90
parkinsons	195	23	2	75
phoneme	5404	6	2	71
pima	768	9	2	65
ringnorm	7400	21	2	50
saheart	462	10	2	65
segmentation	2310	19	7	14
spectf	349	45	2	73
statlog-german	1000	21	2	70
statlog-heart	270	14	2	56
tae	151	6	3	34
tic-tac-toe	958	10	2	65
titanic	2201	4	2	68
vehicle	846	19	4	26
vowel	990	11	11	09
waveform-5000	5000	41	3	34
wdbc	569	31	2	63
wine	178	14	3	40
wine-quality	6492	12	6	44
yeast	1479	9	9	31
zoo	84	17	4	49

classifiers to be combined among: 3-NN, DT, SVM with Radial Basis Kernel function, linear SVM, Random Forest (RF) and Naive Bayes (NB). These classifiers were chosen because they represent different learning bias. The HARF filter considers an example as noisy if it is incorrectly classified by at least 70% of the RF base trees. In the ranking version, there is not such parameter. PruneSF uses the C4.5 [8] DT training algorithm for estimating the CLCH values. Finally, k-NN is run with k ranging from $k = 1$ to $k = 9$.

C. Ranking Evaluation

In order to properly evaluate the performance of noise filters in noise detection, it is necessary to know in advance which are the noisy instances. Using this knowledge, Sluban et al. [6] proposed a methodology to evaluate the efficacy of the noise filters. In this methodology, the well-known precision, recall and F-score metrics are used to assess the filters performance. Precision is the percentage of noisy cases correctly identified among those examples identified as noisy by the filter. Recall is the percentage of noisy cases correctly identified among the noisy cases present in the dataset. The F-score (F_1) metric combines precision and recall by a harmonic mean. Precision, recall and F_1 range from 0 to 1 and higher values indicate a better performance in noise detection by a filter.

The previous measures are based on the hard decision of classifying an example as noisy. For rankers, where a soft decision is obtained, other strategies should be employed instead. They should take into account the ordering produced, such that better values are obtained if noisy instances are top-ranked, while clean examples are bottom-ranked.

A simple adaptation of the previous evaluation measures is the application of a threshold to the number of top-ranked examples that will be regarded as noisy [19]. Afterwards, the precision, recall and F_1 values are recorded. These measures are named here $prec@N$, $rec@N$ and $F_1@N$, where N is the number of top-ranked examples that are considered noisy [19], [20]. For setting the N value to be employed, which is the number of top-ranked examples to be considered noisy, we use the same approach as [19]. N is set as the known number of noisy cases introduced in each corrupted dataset. In this case, we have $prec@N = rec@N = F1@N$, since a noisy example misclassified will be replaced by a clean example, increasing both false positive and false negative rates by one unit.

Based on an evaluation measure proposed for feature ranking in [21], we present next an evaluation measure named *Noise Ranking Area Under the ROC curve* (NR-AUC), which is independent of a particular threshold value. Given an ordering of the examples, first a ROC-type graph is built, which considers the true positive (TP, the number of correctly identified noisy cases) and false positive (FP, the number of cases incorrectly predicted as noisy) rates. Next, the area under the plotted curve is calculated. NR-AUC values range from 0 to 1, where higher values indicate a better performance, while values close to 0.5 are associated to a random noise identification performance. As an example, consider a problem where there are five known noisy cases and 15 clean examples. A given noise ranker produces the ordering: $n_1, n_2, c_1, c_2, n_3, n_4, c_3, c_4, n_5, c_5, ..., c_{15}$, where n stands for a noisy example and c for a clean example. It is possible to observe that the third example in the list is clean but it is between examples that are top-ranked as noisy. The adapted ROC graph obtained for this example is shown in Figure 1. Each time a noisy case is observed, a TP value is accounted and the curve grows one unit at the y axis (TP). When a clean example is found, the curve grows one unit at the x axis, corresponding to FP. NR-AUC can then be calculated as the number of unit squares bellow the curve, normalized by the total number of squares ($67/(5*15) = 0.8933$ for the example in Figure 1).

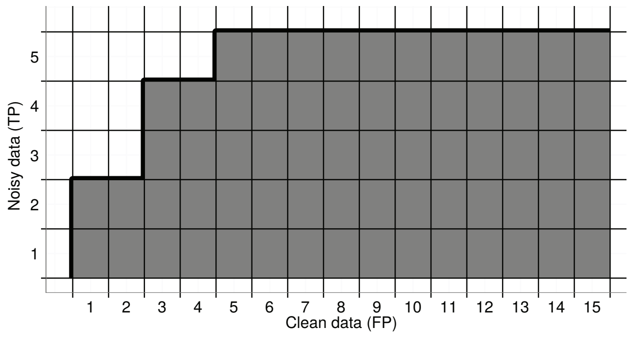

Fig. 1. Example of NR-AUC calculation

D. Experimental Results

1) Performance of Filters and Rankers: Figure 2 shows a heatmap of the F_1 and $prec@N$ $(= F_1@N)$ values obtained by the filters and rankers, respectively, in the noise identification task. These are the average for all noise levels. They are separated according to the noise filtering/ranking technique employed. Higher values are colored in red and lower values are colored in blue. It must be pointed out that, in order to calculate the $prec@N$ value, all rankers are informed on the number of top-ranked examples to be considered noisy. On the other hand, the filters can predict a smaller or a larger number of examples as noisy. This can either help or harm the rankers. For instance, if all noisy examples are poorly ranked after the N-th position, the $prec@N$ value will be null. Since the filters make predictions despite of the N value, their precision can be higher in this situation. Therefore, although there are differences of concept in the calculations of F_1 and $prec@N$, we make a direct comparison of these measures next. The objective is to observe whether the adaptations of the filter algorithms for ranking are adequate.

The results vary depending on the noise filter-ranker pair compared. For SEF and HARF, ranking in general improves the results. The improvements are notable specially for the HARF technique. For DEF and AENN, the results are mixed and there are some few worsening results. For PruneSF, ranking was in general not beneficial. In datasets *cardiotocography*, *collins*, *glass*, *led7digit*, *movement-libras*, *vowel* and *zoo*, for the majority of the techniques, ranking consistently allowed a significant improvement in noise identification. The improvements on the SEF results are noticeable, since it employs only three base classifiers for noise identification and ranking. If more classifiers are used in the ensemble, more smooth noise reliability levels can be obtained and the results can be improved further. In order to better assess the differences of performance between filters and rankers, we performed a Wilcoxon signed-ranks test between each filter and its ranking counterpart [22]. At 95% of confidence level, there are differences between HARF and PruneSF filter-ranker pairs. For HARF, the ranking version performed better, while for PruneSF, the original filter was the best. This proves the general suitability of our adaptations for ranking, although alterations might be necessary in the case of PruneSF.

2) NR-AUC: Figure 3 shows the NR-AUC values obtained by the ranking techniques for each noise level. This measure allows a ranking analysis independent of one specific threshold. In all cases, the ranking performance degrades for higher levels of label noise. Therefore, ranking results were hightly affected by the noise level present in the datasets. There are cases (datasets *abalone*, *bupa*, *cmc*, *connectionist-mines-vs-rocks*, *crabs*, *habermans-survival*, *leukemia-haslinger*, *molecular-promoters*, *pima*, *ringnorm*, *saheart*, *statlog-german*, *statlog-heart* and *tae*) where the NR-AUC performance degrades to around 0.5 for 40% of noise level, which indicates a random performance in noise prediction. Figure 3 also evidences that HARF was in general the best ranking technique, while AENN and PruneSF achieved worst performance.

V. Conclusion

The ranking of noisy examples can improve the identification of noisy examples in a dataset. This paper adapted the outputs of some popular noise filtering techniques from the literature to provide a ranking of the noisy cases identified. Thereby, all techniques output, for a given example, an estimate of its probability of being noisy. The experimental results show that noise identification can be improved by ranking the noisy examples. Future work will investigate strategies to define the threshold value used to label an instance as noisy. To adapt the PruneSF technique to take more advantage of the ranking is another research direction. Other ranking performance measures should also be investigated. And the results of the techniques can be joined to provide more robustness to the ranking produced.

Acknowledgments

The authors would like to thank the Brazilian research agencies FAPESP and CNPq for their support.

References

[1] G. L. Libralon, A. C. P. L. F. Carvalho, and A. C. Lorena, "Pre-processing for noise detection in gene expression classification data," *JBCS*, vol. 15, no. 1, pp. 3–11, 2009.

[2] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE Trans on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2014.

[3] C. E. Brodley and M. A. Friedl, "Identifying and eliminating mislabeled training instances," in *AAAI/IAAI*, 1996, pp. 799–805.

[4] L. P. F. Garcia, A. C. Lorena, and A. C. P. L. F. Carvalho, "A study on class noise detection and elimination," in *IEEE Proc 2012 Braz. Symp. Neural Networks*, 2012, pp. 13–18.

[5] D. Gamberger, N. Lavrac, and C. Groselj, "Experiments with noise filtering in a medical domain," in *ICML*, 1999, pp. 143–151.

[6] B. Sluban, D. Gamberger, and N. Lavrac, "Ensemble-based noise detection: noise ranking and visual performance evaluation," *DMKD*, vol. 28, no. 2, pp. 265–303, 2014.

[7] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *AIR*, vol. 22, no. 3, pp. 177–210, 2004.

[8] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

[9] E. Eskin, "Detecting errors within a corpus using anomaly detection," in *Proc. 1st North American Chapter Association for Computational Linguistics Conf.*, 2000, pp. 148–153.

[10] L. P. Garcia, A. C. de Carvalho, and A. C. Lorena, "Effect of label noise in the complexity of classification problems," *Neurocomputing*, vol. 160, pp. 108–119, 2015.

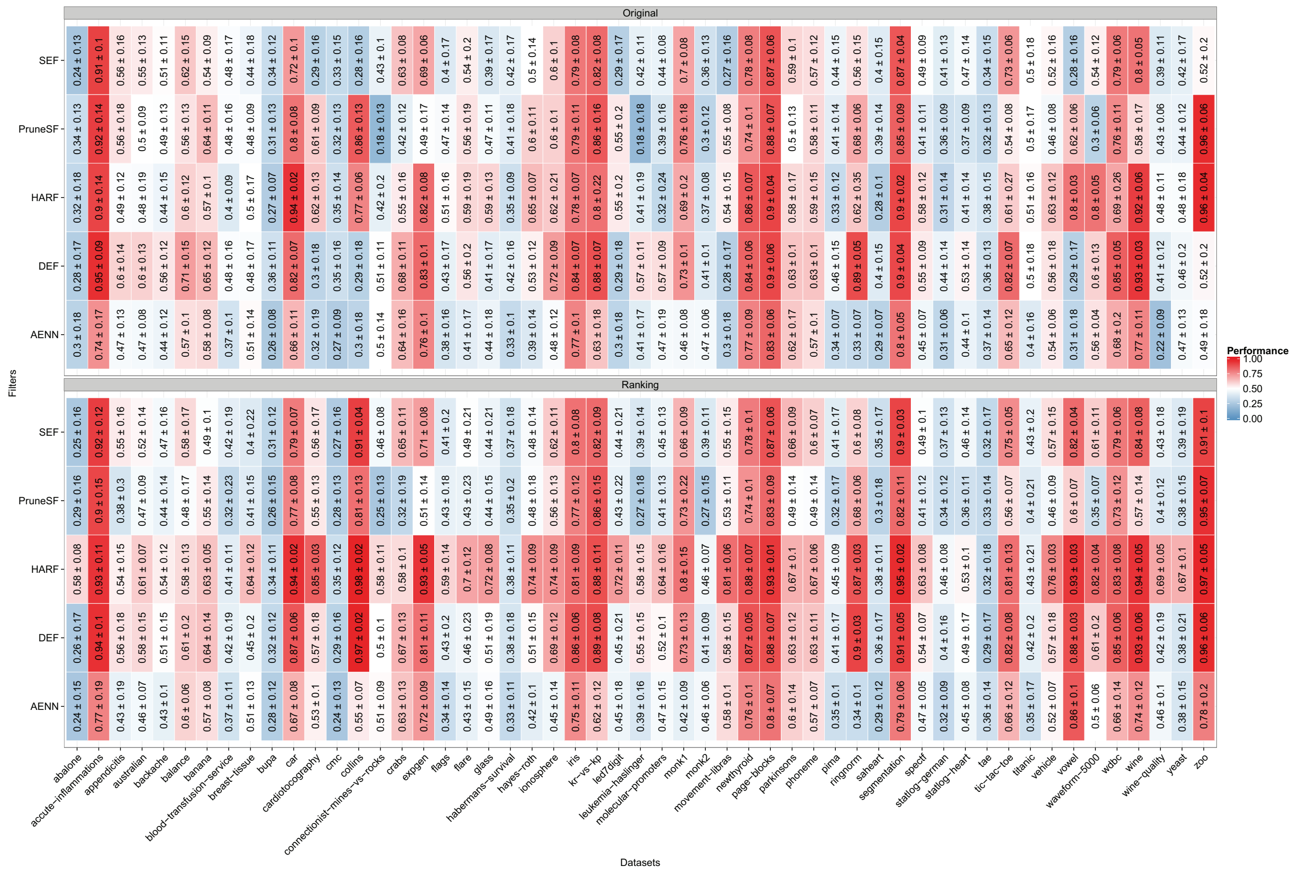

Fig. 2. F$_1$ and precision of filters and rankers, respectively, for each dataset.

[11] A. Ganapathiraju and J. Picone, "Support vector machines for automatic data cleanup," in *INTERSPEECH*, 2000, pp. 210–213.

[12] T. M. Mitchell, *Machine Learning*, 1st ed., ser. McGraw Hill series in computer science. McGraw-Hill, 1997.

[13] V. N. Vapnik, *The nature of Statistical learning theory*. Springer-Verlag New York, Inc., 1995.

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

[15] I. Tomek, "An experiment with the edited nearest-neighbor rule," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-6, no. 6, pp. 448–452, 1976.

[16] M. R. Smith, T. Martinez, and C. Giraud-Carrier, "An instance level analysis of data complexity," *Machine learning*, vol. 95, no. 2, pp. 225–256, 2014.

[17] K. Bache and M. Lichman, "UCI machine learning repository," 2013, http://archive.ics.uci.edu/ml.

[18] C.-M. Teng, "Correcting noisy data," in *ICML*, 1999, pp. 239–248.

[19] E. Schubert, R. Wojdanowski, A. Zimek, and H.-P. Kriegel, "On evaluation of outlier rankings and outlier scores," Anaheim, CA, 2012, pp. 1047–1058.

[20] N. Craswell, "Precision at n," in *Encyclopedia of Database Systems*. Springer US, 2009, pp. 2127–2128.

[21] N. Spolaor, E. A. Cherman, M. C. Monard, and H. D. Lee, "Relieff for multi-label feature selection," in *IEEE Proc. Brazilian Conf. Intelligent Systems*. IEEE, 2013, pp. 6–11.

[22] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *JMLR*, vol. 7, pp. 1–30, 2006.

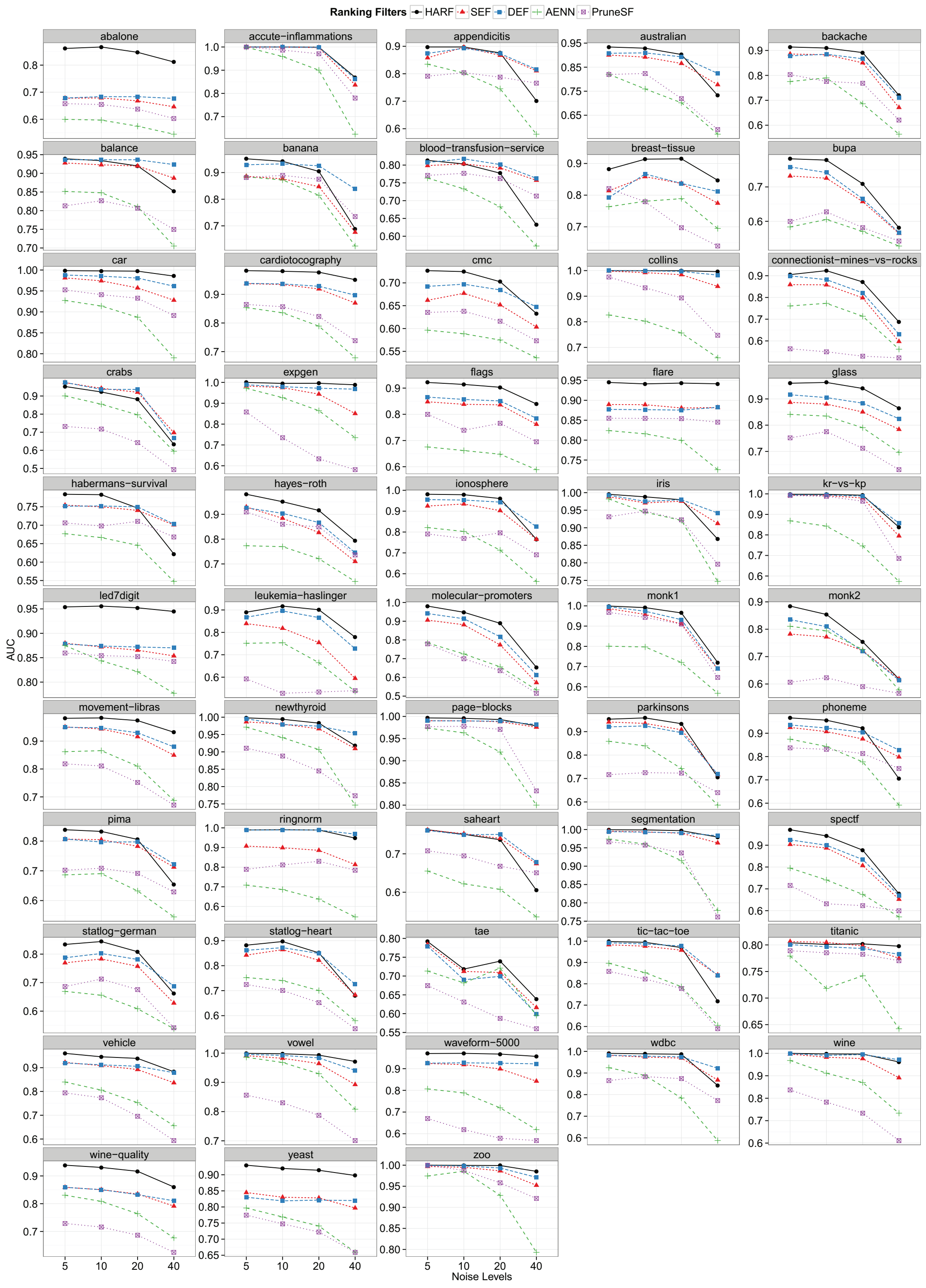

Fig. 3. NR-AUC performance in ranking for different noise levels.