Project Report

Self-labeled techniques for semi-supervised learning

Mark Laane

Abstract. The aim of this project is to reimplement a selection of semi-supervised learning techniques surveyed by Isaac Triguero et al. in paper [1] and to independently reproduce the reported results. For this, two self-labeled techniques were chosen and implemented in Python programming language: Standard self-training and Tri-Training. Two well-known UCI datasets were chosen for testing the implementations: abalone and dermatology. The transductive and inductive classification accuracy of algorithms was measured on the datasets. The classification results achieved in this project are much lower than the results achieved by Triguero et al. indicating an inferior implementation of the algorithms.

1 Background

Semi-Supervised learning is a learning paradigm that joins unsupervised and supervised learning paradigms. It can be useful on data that has small amount of labeled data and a large amount of unlabeled data. In supervised learning only labeled data can be used, but in semi-supervised learning the unlabeled data is leveraged to provide better models. It is useful in situations where there is large amount of unlabeled data and labeling all samples is unviable. [1]

Self-labeled techniques are a subset of semi-supervised classification techniques that grow the labeled dataset by iteratively labeling some samples from the unlabeled dataset. [1]

2 Base classifiers

Some Self-labeled techniques build on supervised learning classifiers by using them as base classifiers. The base classifiers are used on labeled data to predict the labels of the unlabeled samples. By incorporating those predictions to the labeled dataset, the size of the labeled dataset is made larger and base classifier is retrained with this larger dataset. [1]

In the paper, 4 base classifiers were used with self-labeled methods: K-nearest neighbor (KNN), C4.5, Naive Bayes (NB) and Support vector machines (SVM). They all are classic and well known classifiers. In this project KNN and NB was used and the C4.5 decision tree classifier was substituted with CART as it is offered in Scikit-learn library and is similar to C4.5. [2]

3 Implemented algorithms

In this project two self-labeled techniques were implemented: Standard self-training and Tri-Training. Per classification proposed by Isaac Triguero et al. both are single-view, single learning and incremental learning algorithms. This means that the feature space does not have to be split into independent views (single-view), the labeled dataset is only incrementally enlarged from unlabeled dataset (incremental) and identical learners or base classifiers are used (single-learning). The algorithms differ in the number of classifiers used: Self training is a Single-classifier algorithm and uses only one instance of base classifier. Tri-Training is a multi-classifier and uses three instances of a given base classifier.

The implementation of Standard Self-Training is based on description of the algorithm in paper [3]. Training a Standard Self-Training classifier is an iterative process - The base classifier is trained with initial labeled samples. Then it is used for labelling the unlabeled samples and the classifier is retrained with the most confident predictions added to the labeled set. The training and labeling process is repeated until the classifier output stabilizes.

The implementation of Tri-Training is based on description of the algorithm in paper [4]. In Tri-Training, three base classifiers are trained on randomly subsampled sets of the labeled data. Then each of them will be iteratively trained on labeled data gained from two other base classifiers. The prediction is made by using majority voting on three base classifiers.

4 Experiments

10-fold cross-validation procedure was used to validate the classification algorithms. The 10% in each split was kept for testing and the rest used for training. To simulate dataset with unlabeled samples, each training dataset was further split in two: unlabeled and labeled samples. From unlabeled samples, the labels were removed. Four labelling-ratios were tested: 10%, 20%, 30% and 40%

Two accuracy scores were measured in the experiments: accuracy of transductive learning and accuracy of inductive learning. Transductive accuracy describes the classifier's ability to correctly predict the labels of unlabeled samples used during the training process. Inductive accuracy describes the classifiers ability to correctly predict labels of unseen data – test data that is withhold and not been used for training.

5 Results

Each self-labeled algorithm was tested with each base classifier the validation was performed on each dataset. The results can be seen in the table in Appendix 1.

As expected, raising labelling rate also raises the transductive and test accuracy. But there is an anomalous result with unknown source, where of Tri-Training used with Naïve Bayers has the highest accuracy on 10% labelling ratio.

The classification accuracy results outputted by the implemented algorithms are lower than the results achieved by Triguero et al. indicating a poor performance of the implementations. Also, high standard deviation indicates poor performance, as the results seems to strongly depend on the fold that the data is being tested on.

On dermatology dataset, both Self-Training with CART and Tri-Training with CART achieved best results reaching towards 90% of accuracy.

6 Conclusions

In this project two self-labeled techniques - Standard self-training and Tri-Training were implemented and tested on two datasets: abalone and dermatology. The classification accuracy of the implemented algorithms is lower than the results achieved by Triguero et al. This could be explained by poor implementation. Therefore, further work should ensue to locate problematic parts of the implementations and fi them to improve prediction accuracy.

References

[1] I. Triguero, S. García and F. Herrera, "Self-labeled techniques for semi-supervised learning," *Knowledge and Information Systems,* vol. 42, no. 2, pp. 245--284, 2015.

[2] "Scikit-learn Decision Trees," [Online]. Available: http://scikit-learn.org/stable/modules/tree.html#tree-algorithms-id3-c4-5-c5-0-and-cart.

[3] Y. D, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proceedings of the 33rd annual meeting of the association for computational linguistics*, 1995.

[4] L. M. Zhou ZH, "Tri-training: exploiting unlabeled data using three classifiers," *IEEE,* 2005).

Appendix 1

	classifier	dataset	labe-ling_rate	test_mean	test_std	trans_mean	trans_std
0	TriTraining (KNN)	dermatology	0.1	0.488889	0.060131	0.447080	0.036302
1	TriTraining (KNN)	dermatology	0.2	0.650375	0.070127	0.670488	0.028531
2	TriTraining (KNN)	dermatology	0.3	0.735210	0.083414	0.742178	0.021418
3	TriTraining (KNN)	dermatology	0.4	0.784309	0.085764	0.800408	0.028411
4	TriTraining (KNN)	abalone	0.1	0.184359	0.103033	0.185527	0.062670
5	TriTraining (KNN)	abalone	0.2	0.173551	0.101490	0.143877	0.094572
6	TriTraining (KNN)	abalone	0.3	0.155177	0.112807	0.203701	0.009568
7	TriTraining (KNN)	abalone	0.4	0.203248	0.050367	0.143520	0.094432
8	TriTraining (NB)	dermatology	0.1	0.304505	0.200854	0.303841	0.175848
9	TriTraining (NB)	dermatology	0.2	0.133559	0.102494	0.131341	0.088101
10	TriTraining (NB)	dermatology	0.3	0.190991	0.094325	0.225660	0.052463
11	TriTraining (NB)	dermatology	0.4	0.152102	0.139108	0.168364	0.123478
12	TriTraining (NB)	abalone	0.1	0.073991	0.039526	0.043179	0.043297
13	TriTraining (NB)	abalone	0.2	0.083090	0.059929	0.051639	0.051677
14	TriTraining (NB)	abalone	0.3	0.100075	0.047361	0.053660	0.053850
15	TriTraining (NB)	abalone	0.4	0.104854	0.037573	0.071561	0.046911
16	TriTraining (CART)	dermatology	0.1	0.821922	0.090365	0.800315	0.055557
17	TriTraining (CART)	dermatology	0.2	0.879730	0.084876	0.908905	0.026136
18	TriTraining (CART)	dermatology	0.3	0.888063	0.084047	0.899744	0.027704
19	TriTraining (CART)	dermatology	0.4	0.909835	0.063917	0.935179	0.022320
20	TriTraining (CART)	abalone	0.1	0.189849	0.056379	0.179912	0.060278
21	TriTraining (CART)	abalone	0.2	0.201570	0.057807	0.177374	0.059797
22	TriTraining (CART)	abalone	0.3	0.172337	0.074017	0.197811	0.013502
23	TriTraining (CART)	abalone	0.4	0.164440	0.068343	0.198289	0.013589
24	Self-Training (KNN)	dermatology	0.1	0.494670	0.053112	0.453488	0.036719
25	Self-Training (KNN)	dermatology	0.2	0.678003	0.086590	0.673888	0.027315

	classifier	dataset	labe-ling_rate	test_mean	test_std	trans_mean	trans_std
26	Self-Training (KNN)	dermatology	0.3	0.724324	0.072453	0.732635	0.025003
27	Self-Training (KNN)	dermatology	0.4	0.765165	0.073864	0.796865	0.032399
28	Self-Training (KNN)	abalone	0.1	0.215711	0.063134	0.217480	0.009506
29	Self-Training (KNN)	abalone	0.2	0.210680	0.063786	0.216107	0.008401
30	Self-Training (KNN)	abalone	0.3	0.209482	0.056126	0.213241	0.009116
31	Self-Training (KNN)	abalone	0.4	0.202054	0.058924	0.205958	0.010749
32	Self-Training (NB)	dermatology	0.1	0.285060	0.081652	0.260744	0.054078
33	Self-Training (NB)	dermatology	0.2	0.196396	0.069305	0.194363	0.012955
34	Self-Training (NB)	dermatology	0.3	0.196396	0.069305	0.215726	0.040900
35	Self-Training (NB)	dermatology	0.4	0.196396	0.069305	0.223930	0.035261
36	Self-Training (NB)	abalone	0.1	0.080430	0.027688	0.076111	0.017747
37	Self-Training (NB)	abalone	0.2	0.085939	0.034251	0.082600	0.011560
38	Self-Training (NB)	abalone	0.3	0.070136	0.027342	0.076694	0.016192
39	Self-Training (NB)	abalone	0.4	0.067502	0.027958	0.067753	0.013520
40	Self-Training (CART)	dermatology	0.1	0.734234	0.084154	0.776031	0.084967
41	Self-Training (CART)	dermatology	0.2	0.882658	0.093092	0.901316	0.033661
42	Self-Training (CART)	dermatology	0.3	0.887913	0.076882	0.902362	0.023257
43	Self-Training (CART)	dermatology	0.4	0.909910	0.062707	0.932149	0.023461
44	Self-Training (CART)	abalone	0.1	0.177636	0.048120	0.199894	0.008900
45	Self-Training (CART)	abalone	0.2	0.188643	0.052186	0.198384	0.010321
46	Self-Training (CART)	abalone	0.3	0.197740	0.042104	0.202030	0.014942
47	Self-Training (CART)	abalone	0.4	0.200624	0.049635	0.196249	0.009393