DCSO: Dynamic Combination of Detector Scores for Outlier Ensembles

Yue Zhao
Department of Computer Science
University of Toronto

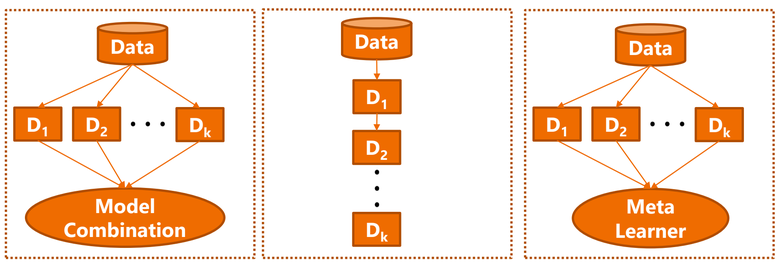
Maciej K. Hryniewicki
Data Assurance & Analytics
PricewaterhouseCoopers

UNIVERSITY OF TORONTO

1. Outlier Ensembles

Definition:

Outlier ensembles **combine** the results (scores) of either **independent** or **dependent** outlier detectors.



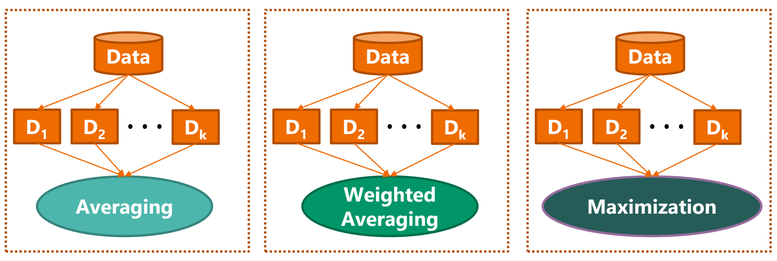
Parallel Learning Sequential Learning Stacking

Advantages:

- **Improved stability**: robust to uncertainties in complex data, e.g. high-dimensional data
- **Enhanced detection quality**: capable of leveraging the advantages of underlying models
- **More confident while using the model**

Challenges:

The ground truth (label), whether a data object is abnormal, is always **absent**.



Examples of Parallel Detector Combination

Limitations in Detector Combination:

- **Static process**: the process to measure detector competency is missing
- **Global assumption**: the importance of the data locality is underestimated
- **Limited interpretability**: the explicability of is undermined during combination

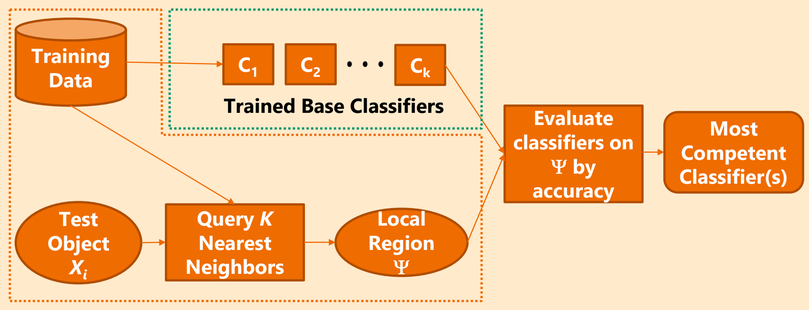
2. Proposal

Design an ***unsupervised*** combination framework to ***select performing detectors*** with a focus ***on the local region***, for improved performance and interpretability.

DCSO: **D**ynamic **C**ombination of Detector **S**cores for **O**utlier Ensembles

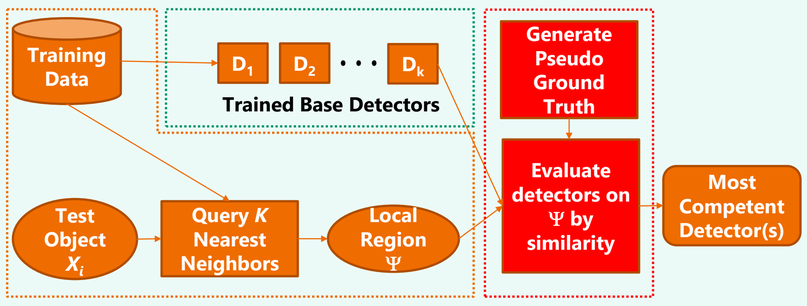
3. From Dynamic Classifier Selection (DCS) to DCSO

DCS is a well-established ensemble framework, which **selects the best classifier** for each test instance **on the fly** by **evaluating base classifiers' competency on the local region** of the test instance.



DCS (Supervised Classification):

- The ground truth **exists**
- Evaluate by **accuracy**



DCSO (Unsupervised Outlier Mining):

- The ground truth is **absent**
- Imbalanced data: # outliers << # inliers
- ➢ Generate **pseudo ground truth** for evaluation
- ➢ Evaluate the detector competency by its similarity to the pseudo ground truth

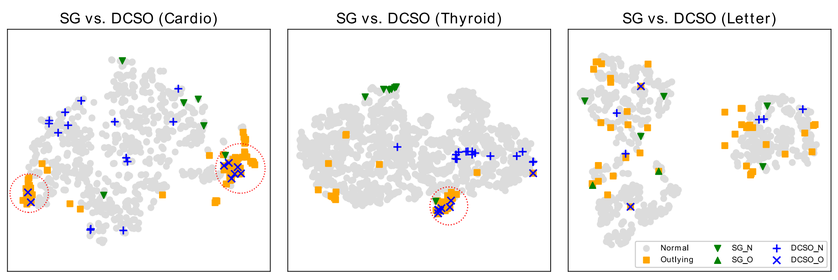
4. Results & Conclusions

DCSO is compared with 6 baseline algorithms on 10 benchmark datasets. All datasets are randomly split at 60% for training and 40% for testing. The average scores of 20 independent trials are used for evaluation.

Table 1. P@m performances (average of 20 independent trials, highest score highlighted in bold, lowest score marked with *)

Dataset	SG_A	SG_M	SG_WA	SG_THRESH	SG_AOM	SG_MOA	DCSO_A	DCSO_M	DCSO_MOA	DCSO_AOM
Pima	0.5100	0.4683	0.5127	0.4933	0.4957	0.5039	**0.5175**	0.4576	0.5083	0.4576*
Vowels	0.3074	0.3250	0.3029*	0.3074	0.3302	0.3185	**0.3682**	0.3044	0.3395	0.3161
Letter	0.2508	0.3547	0.2469	0.2508	0.2950	0.2699	0.2426*	**0.3795**	0.2862	0.3785
Cardio	0.3601	0.3733	0.3624	0.3728	0.4233	0.4104	0.3553	0.3676	**0.4453**	0.3201*
Thyroid	0.3936	0.2589	0.4061	0.3968	0.3731	0.3896	**0.4182**	0.2080*	0.3730	0.2449
Satellite	0.4301*	0.4500	0.4306	0.4466	0.4480	0.4414	0.4400	0.4427	**0.4509**	0.4398
Pendigits	0.0733	0.0590	0.0709	0.0700	0.0637	0.0617	0.0749	0.0595	**0.0811**	0.0560*
Annthyroid	0.2943	0.2951	0.2975	0.2997	**0.3215**	0.3103	0.3065	0.2904*	0.3075	0.3046
Mnist	0.3936	0.3737	0.3944	0.3956	0.3966	0.3976	0.3973	0.3541	**0.4123**	0.3520*
Shuttle	0.1508	0.1484	0.1434	0.1582	0.1591	0.1600	0.1589	0.1389*	**0.1604**	0.1393

Visualization by TSNE



Conclusions

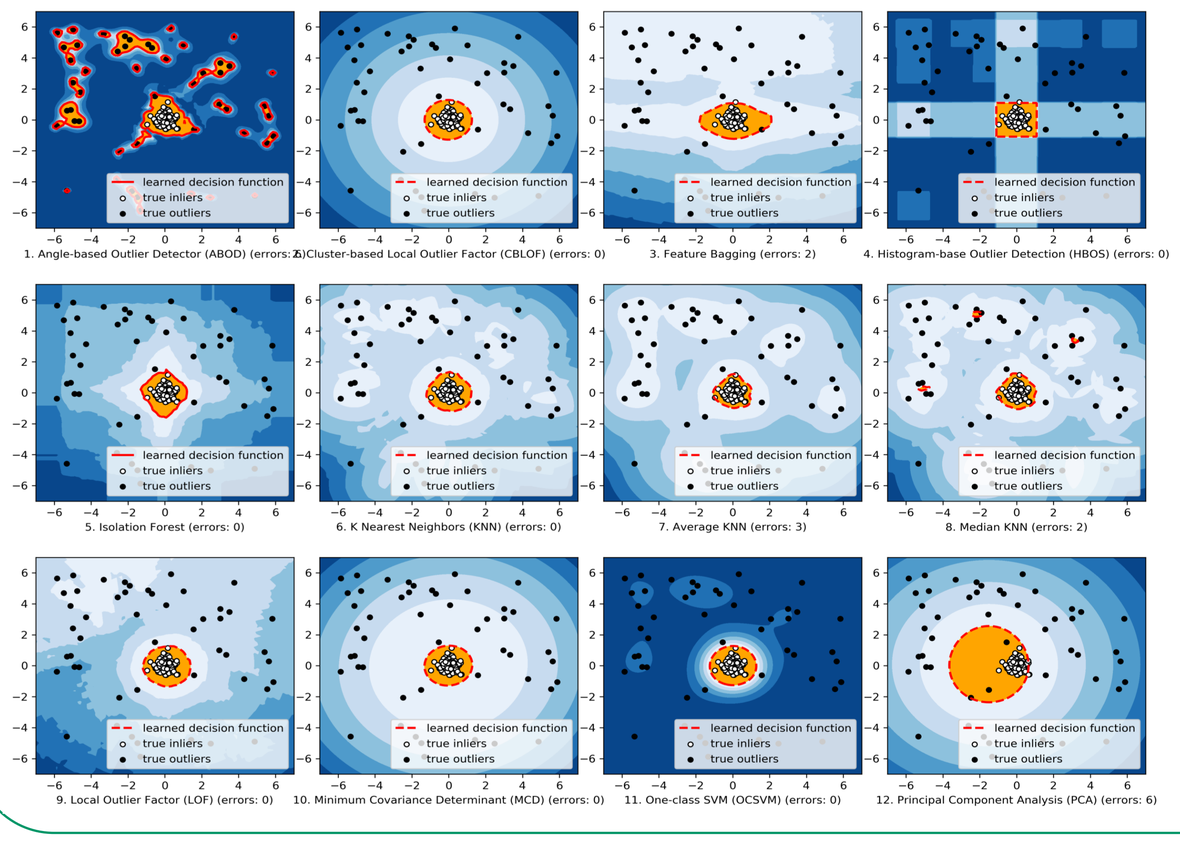
- Outperform on 8 out of 10 datasets with improved detection quality
- Easy to use and robust to underlying assumptions
- Better interpretability to show how the prediction is made individually
- Work best when data forms local clusters

Codes and figures are openly shared at https://github.com/yzhao062/DCSO

5. Python Outlier Detection Toolbox (PyOD)

https://github.com/yzhao062/Pyod (Google: "Python" + "Outlier Detection")

PyOD is a comprehensive Python toolkit to identify outlying objects in multivariate data with both unsupervised and supervised approaches. The toolkit has been successfully used in various academic researches and commercial products.



Highlights:

- **Unified and consistent APIs** across various detection algorithms for easy use
- **Compatibility with both Python 2 and 3**
- **Advanced functions**, e.g., outlier ensemble frameworks
- **Detailed API Reference**, interactive examples in Jupyter Notebooks

Implemented Models:

- PCA: Principal Component Analysis
- MCD: Minimum Covariance Determinant
- One-Class Support Vector Machines
- LOF: Local Outlier Factor
- CBLOF: Clustering-Based Local Outlier Factor
- HBOS: Histogram-based Outlier Score
- kNN: k Nearest Neighbors
- Average kNN
- Median kNN Outlier
- ABOD: Angle-Based Outlier Detection
- FastABOD: Fast Angle-Based Outlier Detection using approximation
- Isolation Forest
- Feature Bagging
- Utility functions, such as scoring metrics Precision @ Rank m