UNIVERSITY OF TORONTO

Northeastern Toronto

LSCP: Locally Selective Combination in Parallel Outlier Ensembles

Yue Zhao, Zain Nasrullah
Department of Computer Science
University of Toronto

Maciej K. Hryniewicki

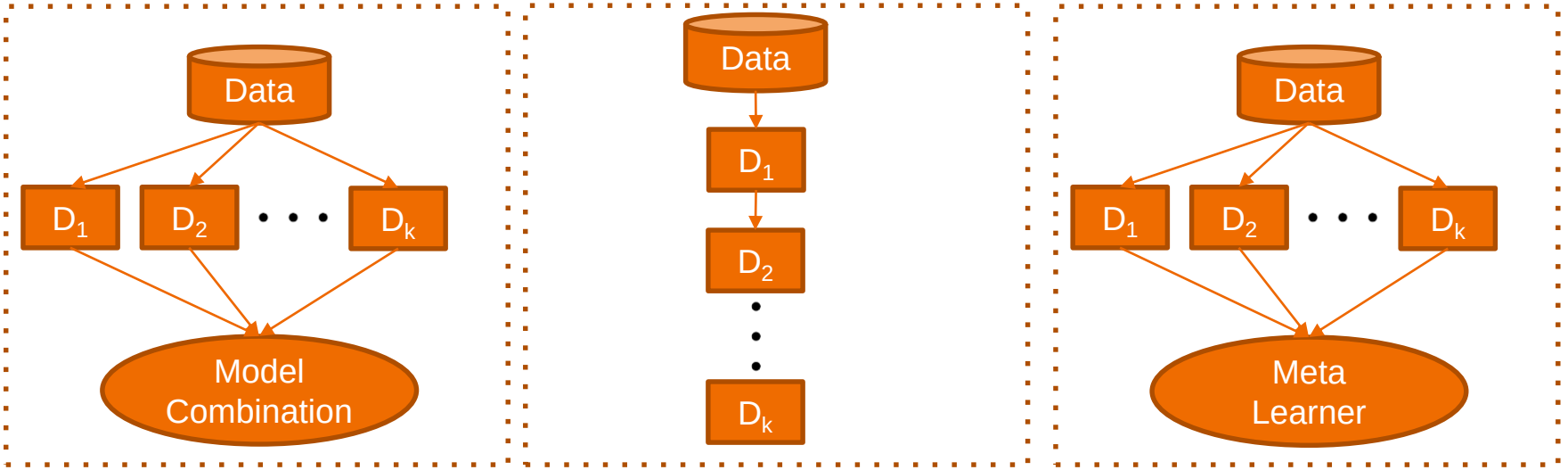
Data Analytics & Assurance

Zheng Li
Toronto Campus
Northeastern University




Outlier Ensembles

Outlier ensembles are designed to **combine** the results (scores) of either **independent** or **dependent** outlier detectors for better performance [1].



Parallel Learning (Bagging [2, 3])

Sequential Learning (Boosting [4, 5])

Stacking [6,7]

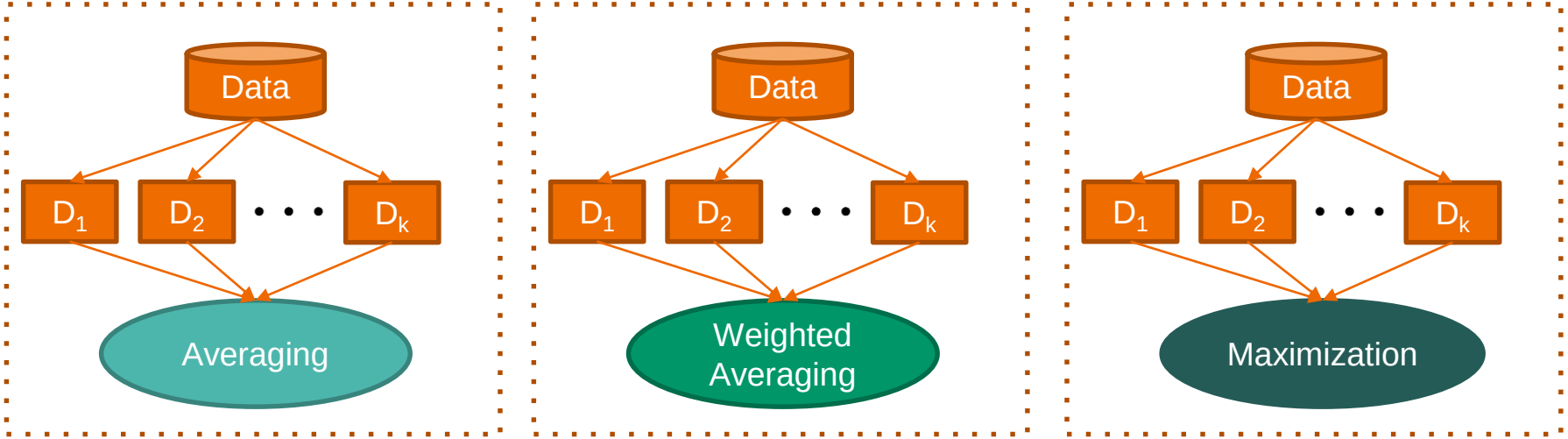
Merits of Outlier Ensembles

The ground truth (label), whether a data object is abnormal, is often **absent** in outlier detection.

- **Improved stability:** robust to uncertainties in complex data, e.g., high-dimensional data
- **Enhanced detection quality:** capable of leveraging the strength of underlying models
- **Confidence**: practitioners usually feel more **confident** to use an ensemble framework with a group of base detectors, than a single model.

Parallel Combination Models

Due to their unsupervised nature, **most of outlier ensemble combination frameworks are parallel learning**.



Examples of Parallel Detector Combination

Limitations in Parallel Outlier Score Combination

- **Generic process**: all based detectors are considered for a new test object, even the underperforming ones. **The selection process is absent**.
- **Global assumption**: the importance of the **data locality is underestimated**, if not ignored, in the combination process.

Generic & **Global** (**GG**) methods combine **all** base models generically on the **global** scale with all data objects considered, leading to mediocre performance.

Research Objective

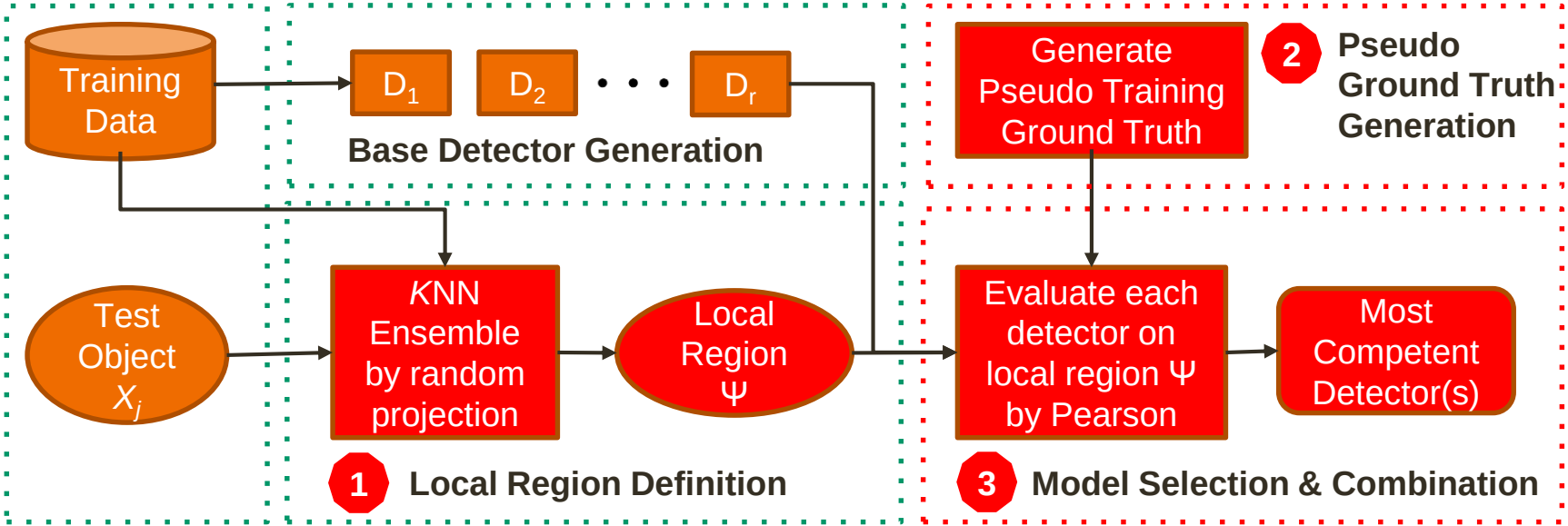
Design an ***unsupervised*** combination framework to ***select performing detectors*** by emphasizing ***data locality***, for each test instance.

For each test object, best base detector(s) can be different.

LSCP: **L**ocally **S**elective **C**ombination in **P**arallel Outlier Ensembles

LSCP Flowchart

LSCP first generates a set of base detectors. For each test object X_j, LSCP (i) **defines the local region** $\Psi(X_j)$; (ii) creates **pseudo ground truth** on $\Psi(X_j)$ and (iii) evaluates, selects, and **combines most competent detector(s)**.



P1: Local Region Definition

The **local region of an test instance X_j** is defined by *k*NN ensemble (consensus of *k* nearest neighbors of X_j in *t* random selected subspaces)

1. generate *t* subspaces by randomly selecting $\left[\frac{d}{2}, d\right]$ features
2. Find X_j's *k* nearest neighbors in each of these *t* subspaces
3. the local region is defined as $\psi_j = \left\{ x_i \mid x_i \in X_{train}, x_i \in kNN_{ens}^{j} \right\}$

P2: Pseudo Ground Truth Generation

Two simple approaches are taken to generate the pseudo ground truth for X_{train} with detectors $D_1, D_2, \ldots, D_r$

1. *target_A*: averages base detector scores on training samples
2. *target_M*: maximum scores across detectors on training samples

Note: it is the combination of training scores, i.e. $D_j(X_{train})$, not of test scores $D_j(X_{test})$.

P3: Model Competency Evaluation

The i_{th} detector performance is evaluated as the **Pearson correlation** between the output of $D_i(\Psi_j)$ and **the pseudo ground truth** $target^{\Psi_j}$ on the **local region** Ψ_j defined by test object X_j. $competency(D_i) = \rho(D_{i(\Psi_j)}, target^{\Psi_j})$

Notably, competent base detectors are assumed to have higher Pearson correlation scores.

LSCP Variants

Original (select one detector as output):

LSCP_A: select one base detector with the highest Pearson score to *target_A*

LSCP_M: select one base detector with the highest Pearson score to *target_M*

Second phase combination (select *s* base detectors):

LSCP_AOM: average *s* base detectors with highest Pearson scores to *target_M*

LSCP_MOA: report maximum of *s* base detectors with highest scores to *target_A*

Experiment Design

Dataset	Pts	Dim	Outliers	%Outlier
Annthyroid	7200	6	534	7.41
Arrhythmia	452	274	66	14.60
Breastw	683	9	239	34.99
Cardio	1831	21	176	9.61
Letter	1600	32	100	6.25
MNIST	7603	100	700	9.21
Musk	3062	166	97	3.17
PageBlocks	5393	10	510	9.46
Pendigits	6870	16	156	2.27
Pima	768	8	268	34.90
Satellite	6435	36	2036	31.64
Satimage-2	5803	36	71	1.22
Shuttle	49097	9	3511	7.15
SpameSpace	4207	57	1679	39.91
Stamps	340	9	31	9.12
Thyroid	3772	6	93	2.47
Vertebral	240	6	30	12.50
Vowels	1456	12	50	3.43
WBC	378	30	21	5.56
Wilt	4819	5	257	5.33

- Tested on 20 outlier benchmark datasets
- Each dataset is split to 60% for training and 40% for testing
- Compared with 7 widely used detector combination methods, such as averaging, average-of-maximum, and feature bagging*
- Used a pool of 50 LOF base detectors
- The average of 30 independent trials is reported and analyzed

Results & Discussions – Overall Performance

Table 2: ROC-AUC scores (average of 30 independent trials, highest score highlighted in bold)

Dataset	LSCP_A	LSCP_MOA	LSCP_M	LSCP_AOM	GG_A	GG_MOA	GG_M	GG_AOM	GG_WA	GG_TH	GG_FB
Annthyroid	0.7548	0.7590	0.7849	0.7520	0.7642	0.7660	0.7769	0.7730	0.7632	0.7552	**0.7854**
Arrhythmia	0.7746	0.7715	0.7729	**0.7763**	0.7758	0.7749	0.7656	0.7690	0.7758	0.7313	0.7709
Breastw	0.6553	0.7044	0.7236	**0.7845**	0.7362	0.7140	0.6590	0.6838	0.7453	0.6285	0.3935
Cardio	0.8691	0.8908	0.8491	**0.9013**	0.8770	0.8865	0.8798	0.8903	0.8782	0.8830	0.8422
Letter	0.7818	0.7954	0.8361	0.7867	0.7925	0.8031	**0.8434**	0.8300	0.7908	0.8001	0.7640
MNIST	0.8576	0.8623	0.7812	**0.8633**	0.8557	0.8588	0.8349	0.8553	0.8563	0.8272	0.8468
Musk	0.9950	0.9970	0.9931	**0.9981**	0.9937	0.9960	0.9960	0.9970	0.9953	0.9958	0.7344
PageBlocks	0.9349	0.9343	0.8687	**0.9488**	0.9443	0.9440	0.9240	0.9371	0.9453	0.9418	0.9284
Pendigits	0.8238	0.8656	0.7238	**0.8744**	0.8378	0.8509	0.8488	0.8622	0.8425	0.8548	0.8034
Pima	0.7059	0.6991	0.6640	**0.7061**	0.7030	0.7003	0.6730	0.6856	0.7037	0.6349	0.6989
Satellite	0.5814	0.6106	0.6006	0.6015	0.5881	0.5992	**0.6258**	0.6220	0.5876	0.6101	0.5818
Satimage-2	0.9852	0.9931	0.9878	**0.9935**	0.9872	0.9907	0.9909	0.9925	0.9880	0.9881	0.9181
Shuttle	0.5392	0.5551	0.5373	0.5514	0.5439	0.5504	**0.5612**	0.5602	0.5413	0.5561	0.3702
SpamSpace	0.3792	0.4594	0.4305	**0.4744**	0.4487	0.4377	0.4060	0.4128	0.4580	0.4104	0.3312
Stamps	0.8888	0.8719	0.8525	**0.8985**	0.8946	0.8927	0.8559	0.8763	0.8953	0.8904	0.8715
Thyroid	0.9579	0.9624	0.9413	**0.9700**	0.9656	0.9647	0.9385	0.9510	0.9665	0.9644	0.8510
Vertebral	0.3324	0.3662	**0.4306**	0.3478	0.3433	0.3467	0.3662	0.3614	0.3442	0.3678	0.3385
Vowels	0.9276	0.9185	0.9238	0.9199	0.9265	0.9275	**0.9313**	0.9271	0.9261	0.9299	0.9148
WBC	0.9379	0.9344	0.9242	**0.9451**	0.9421	0.9409	0.9321	0.9367	0.9420	0.9314	0.9407
Wilt	0.5275	0.5517	**0.6550**	0.4286	0.5101	0.5358	0.6384	0.6056	0.5037	0.5586	0.5868

- LSCP frameworks outperform on 15 out of 20 datasets for ***ROC_AUC***
- LSCP_AOM performs best on 13 out of 20 datasets

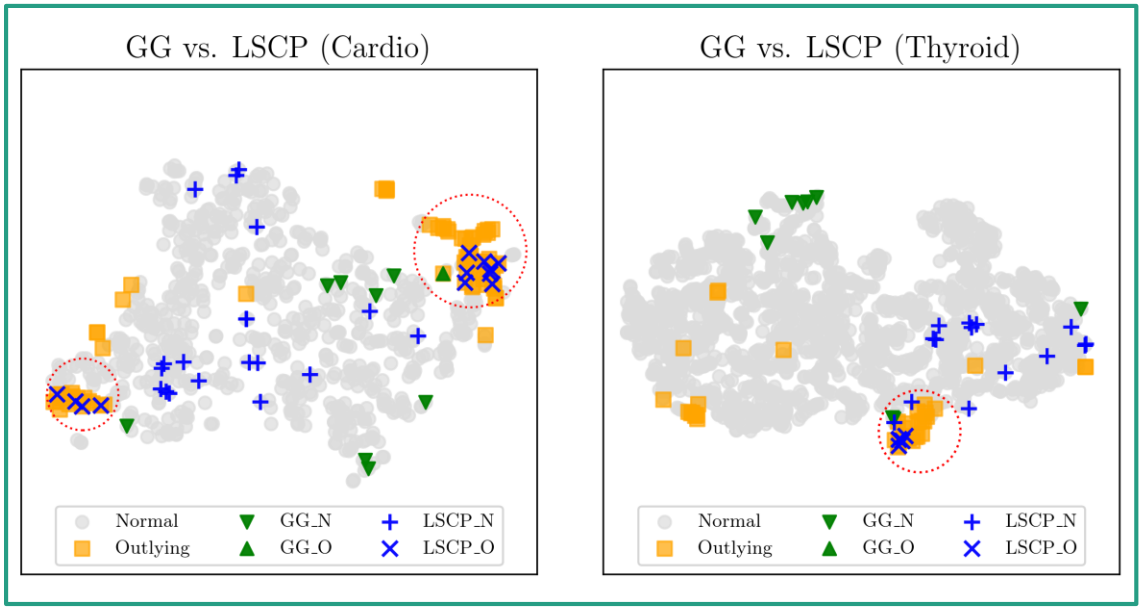
Results & Discussions – Overall Performance

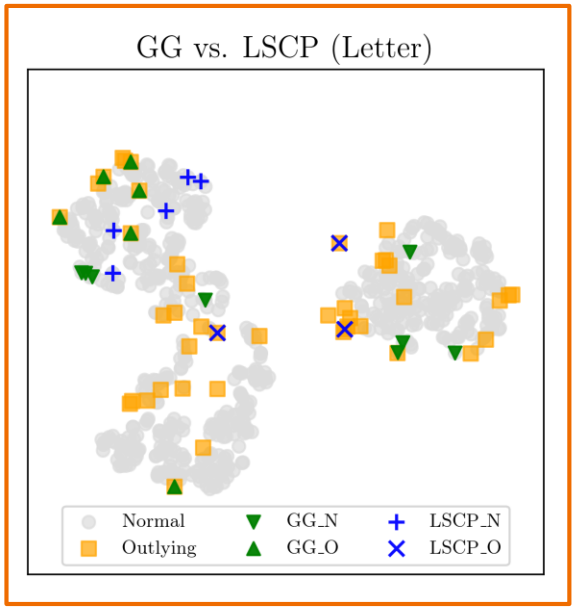
Table 3: mAP scores (average of 30 independent trials, highest score highlighted in bold)

Dataset	LSCP_A	LSCP_MOA	LSCP_M	LSCP_AOM	GG_A	GG_MOA	GG_M	GG_AOM	GG_WA	GG_TH	GG_FB
Annthyroid	0.2283	0.2375	0.2349	0.2453	0.2301	0.2395	0.2413	**0.2516**	0.2306	0.2277	0.1864
Arrhythmia	0.3780	0.3744	0.3790	**0.3796**	0.3766	0.3769	0.3690	0.3722	0.3766	0.3468	0.3707
Breastw	0.4334	0.4766	0.4728	**0.5655**	0.4995	0.4849	0.4249	0.4577	0.5085	0.4366	0.2854
Cardio	0.3375	0.3960	0.3197	**0.4117**	0.3516	0.3708	0.3666	0.3864	0.3535	0.3629	0.3643
Letter	0.2302	0.2396	**0.3346**	0.2407	0.2388	0.2473	0.3160	0.2867	0.2372	0.2416	0.2193
MNIST	0.3933	0.3974	0.3353	**0.3979**	0.3911	0.3941	0.3701	0.3896	0.3918	0.3836	0.3928
Musk	0.8478	0.8773	0.8433	**0.9240**	0.8245	0.8718	0.8479	0.8806	0.8608	0.8629	0.5806
PageBlocks	0.5805	0.5707	0.4684	**0.6360**	0.6043	0.6016	0.5297	0.5733	0.6077	0.6064	0.6094
Pendigits	0.0709	0.0893	0.0625	**0.0944**	0.0777	0.0823	0.0834	0.0895	0.0780	0.0832	0.0834
Pima	0.5092	0.5045	0.4716	**0.5142**	0.5089	0.5054	0.4813	0.4920	0.5095	0.4599	0.5094
Satellite	0.4077	0.4268	0.4223	0.4196	0.4047	0.4139	**0.4385**	0.4352	0.4047	0.4031	0.4049
Satimage-2	0.3477	0.6248	0.3994	**0.6249**	0.3959	0.5089	0.5344	0.5922	0.4159	0.4114	0.4851
Shuttle	0.1228	0.1296	0.1167	**0.1330**	0.1297	0.1316	0.1239	0.1294	0.1293	0.1316	0.0549
SpamSpace	0.3326	0.3615	0.3592	**0.3665**	0.3572	0.3521	0.3379	0.3413	0.3612	0.3601	0.3079
Stamps	0.3596	0.3310	0.3193	**0.3779**	0.3694	0.3660	0.3144	0.3387	0.3706	0.3638	0.3535
Thyroid	0.3544	0.3955	0.2638	**0.4651**	0.4045	0.4123	0.2850	0.3488	0.4130	0.4071	0.1186
Vertebral	0.0948	0.1020	**0.1230**	0.0988	0.0971	0.0975	0.1029	0.1000	0.0972	0.1067	0.0965
Vowels	**0.3913**	0.3678	0.3482	0.3539	0.3783	0.3790	0.3760	0.3732	0.3784	0.3783	0.3340
WBC	0.6033	0.5983	0.5472	**0.6131**	0.6097	0.6069	0.5579	0.5925	0.6105	0.6045	0.5933
Wilt	0.0518	0.0557	**0.0770**	0.0423	0.0493	0.0523	0.0715	0.0633	0.0486	0.0537	0.0591

- LSCP frameworks outperform on 18 out of 20 datasets for ***mAP*** (mean average precision)
- LSCP_AOM performs best on 14 out of 20 datasets

Results & Discussions – When does LSCP Work





Visualization by t-distributed stochastic neighbor embedding (t-SNE)

LSCP works well when data forms local patterns.

Conclusion

LSCP is an outlier ensemble framework to **select the top-performing base detectors** for **each test instance** relative to its **local region.**

Among all four LSCP variants, **LSCP_AOM** demonstrates the best performance.

Future Directions:

1. Incorporate more sophisticated pseudo ground truth generation methods
2. Design more efficient and robust local region definition approaches
3. Test and extend LSCP framework with a group of heterogeneous detectors

Model Reproducibility

LSCP's code, experiment results, and figures are openly shared:

- https://github.com/yzhao062/LSCP

Production level implementation is available at **Python Outlier Detection Toolbox (PyOD)**, which can be invoked as "pyod.models.lscp":

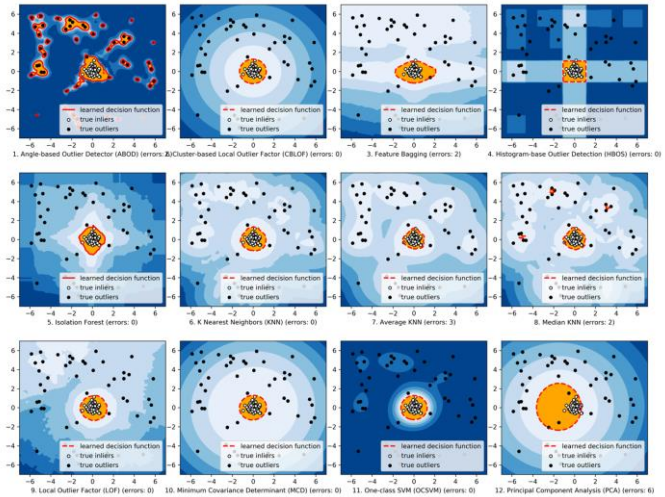
- LSCP examples: https://github.com/yzhao062/pyod/blob/master/examples/lscp_example.py
- API reference: https://pyod.readthedocs.io/en/latest/pyod.models.html#module-pyod.models.lscp

PyOD is for Everyone – Have Your Algorithms In!

PyOD has become the most popular Python Outlier Detection Toolkit:

- Downloads > 50,000 times
- GitHub stars > 1,800; forks > 350
- Featured by various tech blogs, e.g., KDnuggets
- Paper accepted by *Journal of Machine Learning Research (JMLR)* – appear soon

Interested in **having your algorithms included in PyOD** to be used by practitioners around the world? Let's connect ☺ (**Poster 86**)



https://github.com/yzhao062/pyod

Google "Python + Outlier + Detection"

LSCP: Locally Selective Combination in Parallel Outlier Ensembles Scores for Outlier Ensembles

https://github.com/yzhao062/LSCP

PyOD: Python Outlier Detection Toolbox

https://github.com/yzhao062/pyod

Yue Zhao, Zain Nasrullah
Department of Computer Science
University of Toronto

Maciej K. Hryniewicki

Data Analytics & Assurance

Zheng Li
Toronto Campus
Northeastern University

UNIVERSITY OF TORONTO

Northeastern Toronto

Reference

[1] Aggarwal, C.C. 2013. Outlier ensembles: position paper. *ACM SIGKDD Explorations*. 14, 2 (2013), 49–58.

[2] Lazarevic, A. and Kumar, V. 2005. Feature bagging for outlier detection. *ACM SIGKDD*. (2005), 157.

[3] Liu, F.T., Ting, K.M. and Zhou, Z.H. 2008. Isolation forest. *ICDM*. (2008), 413–422.

[4] Rayana, S. and Akoglu, L. 2016. Less is More: Building Selective Anomaly Ensembles. *TKDD*. 10, 4 (2016), 1–33.

[5] Rayana, S., Zhong, W. and Akoglu, L. 2017. Sequential ensemble learning for outlier detection: A bias-variance perspective. *ICDM*. (2017), 1167–1172.

[6] Micenková, B., McWilliams, B. and Assent, I. 2015. Learning Representations for Outlier Detection on a Budget. arXiv Preprint arXiv:1507.08104.

[7] Zhao, Y. and Hryniewicki, M.K. 2018. XGBOD: Improving Supervised Outlier Detection with Unsupervised Representation Learning. *IJCNN*. (2018).