

A DEFINITION OF OOD DETECTION AND ANOMALY DETECTION

The anomaly detection task aims to construct a classifier capable of detecting OOD abnormal instances (Golan & El-Yaniv, 2018). This is the same as the goal of OOD detection, which is to view the OOD detection and anomaly detection tasks in the same way. Because, since OOD detection is the task of finding data points that follow a distribution rather than an in-distribution, it differs from anomaly detection, which does not have the assumption that abnormal data points follow a distribution, but adding this assumption does not change the goal of the anomaly detection task, so we view the two tasks as the same.

B MODEL DETAILS AND HYPERPARAMETER SEARCHING SPACE

Anomaly detection in tabular data has garnered significant attention due to its relevance in various real-world applications, such as fraud detection, medical diagnosis, etc. (Yin et al., 2024). Anomaly detection tasks often suffer from imbalance problems; hence several unsupervised methods have been worked because they have the advantage of not having from this problem. The learning method of anomaly detection varies depending on the presence of a label; however, this section only introduces unsupervised methods that learn without having information about the label. Because existing approaches can be broadly categorized into shallow models and deep models, it is divided into two categories and explained in the corresponding sections.

Shallow Models Principal Component Analysis (PCA, Wold et al. (1987)) is a dimension reduction method that projects data to a lower dimension than the original using Singular Value Decomposition (SVD). This algorithm can also be utilized as an anomaly detection method by using the reconstruction error when the latent vector projected in low dimensions is reconstructed to the original data as an anomaly score (Shyu et al., 2003). K-nearest neighbor (k -NN, (Ramaswamy et al., 2000)) can measure the anomaly score of input data using the distance to the k th nearest neighbor for the training data. The Local Outlier Factor (LOF, Breunig et al. (2000)) is a method that adopts neighbors to determine anomaly scores from a local perspective. Isolation forest (IF, (Liu et al., 2008)) uses the concept that normal data is easier to isolate than outliers, one-class SVM(OCSVM, (Schölkopf et al., 1999)) determines the decision boundary by determining a support vector that can sufficiently explain the given training data well, and determines that data that exist outside this boundary are outliers. COPOD (Li et al., 2020) and ECOD (Li et al., 2022) are similar anomaly detection algorithms that utilizing the Empirical Cumulative Distribution Function (ECDF), and measure anomaly scores using the extremeness of input data under ECDF.

Deep Models DAGMM (Zong et al., 2018) utilizes a deep autoencoder to generate a low-dimensional representation and reconstruction error for each input data point, which is further fed into a Gaussian Mixture Model (GMM). DeepSVDD (Ruff et al., 2018) trains to bring the input data representation closer to the predefined center using a neural network, sets the boundary of the hypersphere and determines that data existing outside this are outliers. GOAD (Bergman & Hoshen, 2020) is a model that generalizes the transformation-based self-supervised method used in the image domain and applies it to the tabular domain. NeuTraLAD (Qiu et al., 2021) and ICL (Shenkar & Wolf, 2022) applied the contrastive learning method, one of the self-supervised methods, to the tabular domain to improve performance. DO2HSC (Zhang et al., 2024) improves the limitations of DeepSVDD with a hypersphere assumption through orthogonal projection and double hypersphere decision boundary. MCM (Yin et al., 2024) performs anomaly detection by extending the masking self-supervised method adopted in the NLP or image domain to the tabular domain.

Additionally, we report details of the 14 models we implemented and the hyperparameter search space for each model. Because it would be impractical to perform a hyperparameter search for every hyperparameter that exists per model, we have selected the hyperparameters that we believe are important per model. For each hyperparameter searching space, we set commonly used hyperparameter values, ran experiments 10 times on 47 tabular datasets provided by ADBench for all combinations of hyperparameters, and selected the hyperparameter combination with the highest average AUROC for all datasets as a representative hyperparameter that demonstrates the performance of the model. In Table 3, we record the hyperparameter search space used in our experiments, and in Table 4, we record the optimal hyperparameter combination finally selected from the hyperparam-

eter search space. In the description of the deep model where the implementation code resides, we only specify the hyperparameters we modified.

PCA We implemented PCA via the PyOD library, and chose `n_component_ratio` as the hyperparameter to explore. `n_component_ratio` is a hyperparameter for what percentage of the dimensionality of the original input data should be reduced.

LOF We implemented LOF via the PyOD library and chose `N` as the hyperparameter to explore. `N` is a hyperparameter for how many neighbors to consider.

IF We implemented IF via the PyOD library and chose `N` as the hyperparameter to explore. `N` is the hyperparameter for how many trees to ensemble.

OCSVM We have implemented OCSVM via the PyOD library and have chosen `kernel` as the hyperparameter to explore. `kernel` is the hyperparameter for which function to choose as the kernel function of OCSVM. OCSVM specially set only one hyperparameter as the searching space, because we excluded the Radial Basis Function (RBF) kernel, which is a popular kernel function for OCSVM, because it takes a very long time to run and we could not verify the experimental results.

k -NN We implemented k -NN via the PyOD library and chose `K`, `method` as the hyperparameter to explore. `K` is the hyperparameter for how many neighbors to consider, and `method` is the hyperparameter for how to calculate the final anomaly score for the `K` neighbors.

COPOD We implemented COPOD via the PyOD library and did not set a hyperparameter searching space because the model is hyperparameter-free.

ECOD We implemented ECOD via the PyOD library and did not set a hyperparameter searching space because the model is hyperparameter-free like COPOD.

DAGMM We ran DAGMM experiments based on the model implemented in <https://github.com/mperezcarrasco/PyTorch-DAGMM>. We set the batch size to 512, the epoch to 200. We chose `n_gmm`, `λ_{energy}` , and `λ_{cov}` as hyperparameters to explore. `n_gmm` is the hyperparameter for how many Gaussian mixture components to assume, `λ_{energy}` is the weight of the sample energy term in the loss function, and `λ_{cov}` is the weight of the regularization term on the covariance to avoid the singularity problem. Additionally, we set the hyperparameters that we thought were important in DAGMM as follows, but got nearly the same performance from all combinations.

DeepSVDD We implemented DeepSVDD ourselves using A as a reference. First, we pretrain using Autoencoder for 100 epochs and define the average value of the latent vector obtained by encoding the train data using the corresponding encoder part as the center. Then, we implemented DeepSVDD by training the encoder for 200 epochs with the loss function of the MSE (Mean Squared Error) of the center. The encoder consists of 3 layers with the same hidden size, the batch size is 512, the learning rate is $1e-3$, the optimizer is AdamW (Loshchilov, 2017), activation function to ReLU, the weight decay is $1e-4$, and the learning rate scheduler is CosineAnnealingWarmRestarts (Loshchilov & Hutter, 2016). We set learning rate and latent dimension as the hyperparameters to search for (latent dimension and hidden size are the same).

GOAD We ran GOAD experiments based on the model implemented in <https://github.com/lironber/GOAD>. We set the batch size to 512, the epoch to 200. We chose `n_rot`, `d_latent`, and `d_out` as hyperparameters to explore. `n_rot` is a hyperparameter for how many transformations to perform, and `d_latent` and `d_out` represent the latent dimension and the dimension after transforming the input data, respectively.

NeuTraLAD We ran NeuTraLAD experiments based on the model implemented in <https://github.com/boschresearch/NeuTraL-AD>. We set the batch size to 512, the epoch to 200, number of transformation to 11. The hidden and latent dimensions were selected using the automatic dimension selection implemented in the implementation code. We chose `n_encoder`, `n_translayer`, and `trans_methods` hyperparameters to explore. `n_encoder` represents the number of encoder layers, and `n_translayer` represents the number of layers in the block that performs the transformation. Finally, `trans_method` is a hyperparameter for how we want to perform the transformation.

ICL We ran ICL experiments based on author’s official supplementary at https://openreview.net/forum?id=_hszZbt46bT. We set the batch size 512. We chose `τ` and `d_latent` as hyperparameters to explore. Since earlystopping is implemented in the implementation

code, epoch used 2000 as it was specified in the implementation code. τ represents the softmax temperature, d_{latent} represents the latent dimension.

MCM We ran MCM experiments based on author’s official supplementary at <https://openreview.net/forum?id=1NZJyEDxy4>. According to the paper, the authors tuned only λ and learning rate per dataset, so we set these two hyperparameters to explore. λ represent the weight of mask diversity loss.

NF-SLT We implemented NICE based on <https://github.com/DakshIdnani/pytorch-nice>. The coupling layer block was set to 10, the epoch was set to 200, and the weight decay was set to $1e-4$. The optimizer was AdamW, and the learning rate scheduler was CosineAnnealingWarmRestarts. The prior distribution of the latent vector was set to $\mathcal{N}(0, I_d)$, and then trained to minimize the negative log-likelihood over the training latent vector. Additionally, due to the implementation structure of the coupling layer, it cannot be implemented when an odd-dimensional vector is input, which was solved by simply adding a single zero padding. We chose learning rate, d_{latent} , and n_{layer} as hyperparameters to explore. d_{latent} represents the latent dimension of the coupling layer block, and n_{layer} represents the number of layers in the coupling layer block.

Table 3: Hyperparameter search space each models. η is learning rate.

Model	Hyperparameter Searching Space
PCA	$n_component_ratio \in \{0.75, 0.8, 0.85, 0.9, 0.95\}$
LOF	$N \in \{1, 3, 5, 10, 20\}$
IF	$N \in \{30, 50, 100, 200, 400\}$
OCSVM	$kernel \in \{“linear”\}$
k -NN	$K \in \{1, 3, 5, 10, 20\}$ $method \in \{“mean”, “largest”, “median”\}$
COPOD	None
ECOD	None
DAGMM	$n_{gmm} \in \{3, 4, 5\}$ $\lambda_{energy} \in \{0.1, 0.5, 1\}$ $\lambda_{cov} \in \{5e-3, 1e-2, 5e-2\}$
DeepSVDD	$\eta \in \{1e-4, 5e-4, 1e-3\}$ $d_{latent} \in \{64, 128, 256\}$
GOAD	$n_{rot} \in \{16, 32, 64\}$ $d_{latent} \in \{8, 16, 32\}$ $d_{out} \in \{4, 8, 16\}$
NeuTraLAD	$n_{enclayer} \in \{5, 6, 7\}$ $n_{translayer} \in \{2, 3, 4\}$ $trans_method \in \{“mul”, “residual”\}$
ICL	$\tau \in \{1e-2, 1e-1, 1\}$ $d_{latent} \in \{100, 200, 300\}$
MCM	$\lambda \in \{1e-2, 1e-1, 1, 10\}$ $\eta \in \{1e-4, 5e-4, 1e-3, 5e-3, 1e-2\}$
NF-SLT	$\eta \in \{1e-4, 5e-3, 1e-3\}$ $d_{latent} \in \{64, 128, 256\}$ $n_{layer} \in \{2, 3, 4\}$

C PERFORMANCE COMPARISON NF-SLT BETWEEN NICE AND REALNVP

In this section, we set the flow model used for NF-SLT to RealNVP instead of NICE, and then compared the performance with NICE. RealNVP’s implementation method replaced by the additive coupling layer used in NICE with an affine coupling layer. We set the hyperparameters of RealNVP to be the same as those of NICE, which recorded the performance in Table 1, except for the learning rate. We set the learning rate to $1e-3$, $5e-3$, and $1e-4$, and selected $5e-3$, the learning rate that recorded the highest AUROC performance, to record the performance in Table 5. Table 5 shows that

Table 4: Optimal hyperparameter combination for each model

Model	Optimal Hyperparameter
PCA	n_component_ratio = 0.8
LOF	$N = 20$
IF	$N = 400$
OCSVM	kernel = "linear"
k -NN	$K = 3$ method = "mean"
COPOD	None
ECOD	None
DAGMM	$n_{gmm} = 3$ $\lambda_{energy} = 0.1$ $\lambda_{cov} = 5e-3$
DeepSVDD	$\eta = 1e-4$ $d_{latent} = 256$
GOAD	$n_{rot} = 32$ $d_{latent} = 32$ $d_{out} = 32$
NeuTraLAD	$n_{enclayer} = 5$ $n_{translayer} = 3$ trans_method = 'mul'
ICL	$\tau = 1e-1$ $d_{latent} = 300$
MCM	$\lambda = 0.1$ $\eta = 5e-3$
NF-SLT	$\eta = 1e-3$ $d_{latent} = 256$ $n_{layer} = 2$

Table 5: Performance comparison between NICE and RealNVP

Model	AUROC	AUPRC	Avg. Rank	Top2 Cum. Ratio	Fail Ratio
NICE	0.8575	0.6398	4.00	0.34	0.06
RealNVP	0.8480	0.6385	4.51	0.30	0.09

RealNVP underperformed on all metrics compared to NICE. From this, we can conclude that affine coupling layers, which have higher expressive power than additive coupling layers, are not effective in tabular anomaly detection.

D EXPERIMENT RESULT EACH DATASET

In this section, we reported the experimental results for all datasets. Table 6 records the AUROC for the shallow model and Table 7 records the AUROC for the deep model. Table 8 shows the AUPRC for the shallow model and Table 9 shows the AUPRC for the deep model. Unlike previous experiments in tabular anomaly detection research, we used an extended dataset and eliminated the bias in dataset selection to create a fairer experimental environment.

E TYPICALITY TEST PERFORMANCE

Anomaly detection using the typicality test is a methodology that started with is normal data having a small region but located in a typical set with high probability (Nalisnick et al., 2019). To explain the methodology of the typicality test, detection is performed based on whether a given $\mathbf{x} \in \mathbb{R}^d$ is included in the (ϵ, N) -typical set $\mathcal{A}_\epsilon^N[p(\mathbf{x})]$ ($p(\mathbf{x})$ is the generative model trained with $\mathbf{x}_{train} \in$

Table 6: AUROC performance of shallow models

Dataset	PCA	LOF	IF	OCSVM	k-NN	COPOD	ECOD
ALOI	0.5492	0.6971	0.5418	0.4962	0.6325	0.5153	0.5304
Cardiotocography	0.8248	0.7207	0.8150	0.2651	0.8034	0.6625	0.7856
Hepatitis	0.8145	0.8249	0.7753	0.4095	0.8066	0.7986	0.7473
InternetAds	0.7012	0.8276	0.4242	0.6245	0.8446	0.6764	0.6770
Ionosphere	0.9037	0.9550	0.9141	0.4561	0.9718	0.7975	0.7328
Lymphography	0.9923	0.9843	0.9972	0.1239	0.9862	0.9986	0.9974
PageBlocks	0.9287	0.9667	0.9284	0.1494	0.9663	0.8768	0.9149
Pima	0.7051	0.7083	0.7337	0.4126	0.7417	0.6564	0.5965
SpamBase	0.8072	0.6613	0.8406	0.2516	0.7870	0.6896	0.6578
Stamps	0.9240	0.9022	0.9312	0.1478	0.9301	0.9290	0.8717
WBC	0.9932	0.9737	0.9968	0.0051	0.9902	0.9959	0.9959
WDBC	0.9955	0.9935	0.9951	0.1461	0.9909	0.9957	0.9748
WPBC	0.4838	0.5642	0.5136	0.4148	0.5402	0.5116	0.4679
Waveform	0.6460	0.7558	0.7354	0.3471	0.7449	0.7312	0.6019
Wilt	0.2911	0.8988	0.4776	0.0433	0.8281	0.3440	0.3935
annthyroid	0.7892	0.8794	0.9141	0.2925	0.9550	0.7756	0.7888
backdoor	0.6406	0.5726	0.7592	0.2412	0.9366	0.7891	0.8459
breastw	0.9877	0.9561	0.9946	0.0079	0.9908	0.9938	0.9904
campaign	0.7678	0.6713	0.7400	0.3162	0.8214	0.7828	0.7695
cardio	0.9648	0.9088	0.9508	0.2997	0.9296	0.9216	0.9348
celeba	0.7997	0.4437	0.7177	0.4077	0.6240	0.7505	0.7569
census	0.7073	0.5437	0.6270	0.2802	0.6722	0.6741	0.6596
cover	0.9574	0.9899	0.8596	0.3076	0.9888	0.8840	0.9203
donors	0.8880	0.9824	0.8992	0.1238	0.9972	0.8151	0.8886
fault	0.5456	0.6535	0.6593	0.5179	0.8138	0.4557	0.4693
fraud	0.9536	0.8706	0.9502	0.0712	0.9543	0.9475	0.9496
glass	0.6674	0.7700	0.8041	0.7108	0.8942	0.7605	0.7136
http	0.9994	0.9405	0.9927	0.2020	0.9996	0.9916	0.9786
landsat	0.4452	0.7469	0.6124	0.5537	0.7824	0.4197	0.3670
letter	0.5363	0.8344	0.6419	0.5501	0.8709	0.5592	0.5718
magicgamma	0.7026	0.8277	0.7749	0.1857	0.8407	0.6814	0.6387
mammography	0.8973	0.8508	0.8806	0.3428	0.8603	0.9056	0.9064
mnist	0.9060	0.7707	0.8680	0.2851	0.8789	0.7763	0.7484
musk	1.0000	1.0000	0.9817	0.8144	0.9991	0.9458	0.9555
optdigits	0.5740	0.9183	0.8436	0.2278	0.9188	0.6819	0.6037
pendigits	0.9500	0.9926	0.9706	0.3294	0.9992	0.9051	0.9274
satellite	0.6761	0.8473	0.8031	0.1780	0.8820	0.6339	0.5836
satimage-2	0.9758	0.9958	0.9935	0.4117	0.9985	0.9745	0.9649
shuttle	0.9931	0.9979	0.9966	0.0131	0.9976	0.9945	0.9929
skin	0.6027	0.9228	0.8899	0.5647	0.9985	0.4702	0.4881
smtp	0.8659	0.8541	0.9074	0.7149	0.9278	0.9117	0.8801
speech	0.4687	0.4952	0.4655	0.4725	0.6065	0.4901	0.4687
thyroid	0.9811	0.9344	0.9894	0.3637	0.9883	0.9402	0.9777
vertebral	0.4334	0.4128	0.4399	0.6151	0.4697	0.3515	0.4337
vowels	0.6664	0.9455	0.7857	0.6229	0.9831	0.4955	0.5919
wine	0.9332	0.9570	0.9175	0.1942	0.9698	0.8725	0.7415
yeast	0.4214	0.4731	0.4171	0.6465	0.4679	0.3828	0.4447
Avg AUROC	0.7715	0.8169	0.8014	0.3438	0.8634	0.7471	0.7425
Avg.Rank	7.13	6.09	6.34	12.75	3.28	8.38	8.74
Top1 Ratio	0.09	0.09	0.09	0.02	0.26	0.02	0.02
Top2 Ratio	0.06	0.06	0.09	0.02	0.17	0.08	0.06
Top1,2 Cumulative Ratio	0.15	0.15	0.17	0.04	0.43	0.10	0.09
Bad Ratio	0.45	0.23	0.23	0.96	0.00	0.57	0.68

Table 7: AUROC performance of deep models

Dataset	DAGMM	DeepSVDD	GOAD	NeuTralAD	ICL	MCM	NF-SLT
ALOI	0.5024	0.5653	0.5319	0.5700	0.5892	0.4831	0.5479
Cardiotocography	0.6067	0.6759	0.3689	0.7626	0.6221	0.5713	0.7558
Hepatitis	0.6167	0.6844	0.6054	0.6516	0.6070	0.7985	0.7475
InternetAds	0.5590	0.6013	0.7356	0.7112	0.8815	0.7973	0.8746
Ionosphere	0.6580	0.9597	0.9459	0.9708	0.9556	0.9574	0.9581
Lymphography	0.8643	0.9756	0.9847	0.9779	0.9624	0.9934	0.9746
PageBlocks	0.7966	0.8866	0.9345	0.9803	0.9691	0.8828	0.9656
Pima	0.5759	0.5899	0.4432	0.5414	0.6547	0.6891	0.7219
SpamBase	0.5564	0.5074	0.3366	0.6406	0.8037	0.7379	0.7724
Stamps	0.7417	0.8361	0.6627	0.8836	0.8122	0.8720	0.9327
WBC	0.8804	0.9873	0.5576	0.8944	0.9696	0.9254	0.9726
WDBC	0.8121	0.9932	0.7814	0.9970	0.9856	0.9680	0.9785
WPBC	0.4724	0.4579	0.3565	0.4216	0.4328	0.5068	0.5051
Waveform	0.5135	0.6609	0.4180	0.8186	0.6172	0.6641	0.7357
Wilt	0.4790	0.6008	0.7298	0.7527	0.8043	0.8655	0.9066
annthyroid	0.8361	0.7689	0.5206	0.6685	0.8732	0.8640	0.9181
backdoor	0.4535	0.9473	0.2805	0.9456	0.9767	0.9222	0.9343
breastw	0.8740	0.9825	0.8402	0.9505	0.9835	0.9900	0.9842
campaign	0.5948	0.7929	0.2837	0.7814	0.6933	0.7987	0.8059
cardio	0.6091	0.9139	0.5964	0.7367	0.8456	0.8093	0.9174
celeba	0.5308	0.3604	0.5204	0.6888	0.4595	0.6400	0.7340
census	0.4619	0.6421	0.4140	0.4991	0.3917	0.7478	0.7186
cover	0.7412	0.6819	0.2571	0.9096	0.9859	0.8949	0.9658
donors	0.6243	0.9970	0.4472	0.9985	0.9630	0.9987	0.9990
fault	0.5491	0.7146	0.7028	0.7680	0.7991	0.5955	0.7518
fraud	0.8141	0.9517	0.4119	0.9448	0.9524	0.8488	0.9564
glass	0.5561	0.4046	0.4495	0.8305	0.9027	0.8783	0.8867
http	0.9966	0.9996	0.7812	0.9380	0.9999	0.9925	1.0000
landsat	0.4874	0.6759	0.6427	0.7582	0.7075	0.5724	0.6543
letter	0.5177	0.7565	0.6528	0.8227	0.9312	0.3674	0.9258
magicgamma	0.6099	0.7267	0.5677	0.7317	0.7453	0.8143	0.8863
mammography	0.5903	0.7263	0.8477	0.6377	0.8108	0.8321	0.8761
mnist	0.6168	0.5940	0.7840	0.9700	0.8931	0.8631	0.9015
musk	0.7128	0.9868	0.9999	1.0000	1.0000	0.9972	1.0000
optdigits	0.4249	0.7386	0.2742	0.8566	0.8192	0.9279	0.9205
pendigits	0.6830	0.9510	0.2513	0.9701	0.9619	0.9913	0.9930
satellite	0.6913	0.8275	0.7789	0.8186	0.8190	0.8008	0.8276
satimage-2	0.9234	0.9971	0.9960	0.9987	0.9974	0.9913	0.9966
shuttle	0.6404	0.9980	0.3801	0.9994	0.9998	0.9979	0.9984
skin	0.7063	0.7270	0.8856	0.9128	0.8954	0.8175	0.9675
smtp	0.7976	0.8855	0.6588	0.8583	0.8805	0.9268	0.9201
speech	0.5008	0.4688	0.5138	0.5123	0.5888	0.4110	0.5795
thyroid	0.9408	0.9256	0.4299	0.9619	0.9653	0.9584	0.9840
vertebral	0.4778	0.4794	0.5692	0.6381	0.5621	0.3544	0.5483
vowels	0.5295	0.9377	0.9316	0.7861	0.9914	0.6873	0.9852
wine	0.7212	0.9533	0.9430	0.9680	0.9575	0.4924	0.9497
yeast	0.5442	0.6315	0.5979	0.5438	0.5574	0.4654	0.4652
Avg AUROC	0.6467	0.7687	0.6086	0.8081	0.8208	0.7864	0.8575
Avg.Rank	11.04	7.68	10.26	6.09	5.70	7.38	4.00
Top1 Ratio	0.00	0.00	0.00	0.15	0.15	0.04	0.15
Top2 Ratio	0.00	0.04	0.00	0.09	0.09	0.04	0.19
Top1,2 Cumulative Ratio	0.00	0.04	0.00	0.23	0.23	0.09	0.34
Bad Ratio	0.87	0.40	0.62	0.30	0.26	0.36	0.06

Table 8: AUPRC performance of shallow models

Dataset	PCA	LOF	IF	OCSVM	k-NN	COPOD	ECOD
ALOI	0.0711	0.1334	0.0649	0.0585	0.0916	0.0607	0.0637
Cardiotocography	0.7324	0.6321	0.7085	0.2563	0.6736	0.5516	0.6598
Hepatitis	0.6398	0.6612	0.5448	0.2655	0.6157	0.5783	0.4707
InternetAds	0.5689	0.7937	0.2782	0.4000	0.8079	0.6206	0.6216
Ionosphere	0.9223	0.9609	0.9252	0.6409	0.9778	0.7988	0.7743
Lymphography	0.8923	0.8121	0.9741	0.0862	0.8491	0.9830	0.9717
PageBlocks	0.7340	0.8497	0.6965	0.1519	0.8662	0.5254	0.6602
Pima	0.6961	0.6776	0.7268	0.4841	0.7297	0.6852	0.6323
SpamBase	0.8427	0.7184	0.8696	0.4306	0.8180	0.7039	0.6819
Stamps	0.5743	0.5235	0.5807	0.1022	0.5988	0.5632	0.4716
WBC	0.9427	0.7288	0.9719	0.0484	0.9231	0.9556	0.9556
WDBC	0.9110	0.8804	0.9002	0.0549	0.8349	0.9283	0.6830
WPBC	0.3681	0.4122	0.3839	0.3487	0.3915	0.3742	0.3492
Waveform	0.0861	0.2901	0.1132	0.0422	0.2694	0.1047	0.0771
Wilt	0.0665	0.4182	0.0876	0.0542	0.2451	0.0708	0.0799
annthyroid	0.5127	0.5591	0.6231	0.0987	0.7559	0.2944	0.4058
backdoor	0.0783	0.0989	0.0942	0.0336	0.3812	0.1275	0.1677
breastw	0.9832	0.9181	0.9944	0.3230	0.9903	0.9936	0.9906
campaign	0.4878	0.3429	0.4645	0.1416	0.5408	0.5141	0.4994
cardio	0.8454	0.6653	0.7977	0.1197	0.7082	0.7103	0.7070
celeba	0.2083	0.0358	0.1306	0.0334	0.0658	0.1656	0.1697
census	0.2012	0.1174	0.1403	0.0755	0.1628	0.1610	0.1547
cover	0.1925	0.8391	0.0827	0.0123	0.8108	0.1214	0.1865
donors	0.3618	0.7986	0.4125	0.0632	0.9352	0.3353	0.4142
fault	0.5726	0.6218	0.6542	0.5543	0.7874	0.4751	0.4906
fraud	0.2661	0.0287	0.2209	0.0018	0.2671	0.3669	0.3339
glass	0.1791	0.2206	0.2177	0.3098	0.3189	0.2001	0.2535
http	0.9156	0.1117	0.5033	0.2016	0.9988	0.4636	0.2534
landsat	0.3169	0.6913	0.4300	0.4218	0.6333	0.2977	0.2799
letter	0.1466	0.4482	0.1661	0.1939	0.4601	0.1284	0.1432
magicgamma	0.7480	0.8612	0.8051	0.3593	0.8675	0.7225	0.6807
mammography	0.4513	0.3563	0.3854	0.0381	0.3817	0.5459	0.5516
mnist	0.6753	0.3479	0.5455	0.1107	0.6702	0.3560	0.3045
musk	1.0000	0.9995	0.7037	0.8249	0.9760	0.4869	0.6140
optdigits	0.0591	0.3212	0.1714	0.0337	0.2789	0.0807	0.0650
pendigits	0.4162	0.8520	0.5394	0.0380	0.9755	0.2930	0.3924
satellite	0.7711	0.8815	0.8455	0.3226	0.8991	0.6957	0.6596
satimage-2	0.8823	0.9305	0.9371	0.0500	0.9771	0.8349	0.7419
shuttle	0.9615	0.9784	0.9863	0.0712	0.9525	0.9773	0.9474
skin	0.3585	0.7164	0.6368	0.4248	0.9963	0.2958	0.3030
smtp	0.4636	0.4138	0.0088	0.4954	0.4962	0.0081	0.5760
speech	0.0361	0.0440	0.0389	0.0335	0.0620	0.0377	0.0385
thyroid	0.7791	0.6215	0.8265	0.1665	0.8375	0.3077	0.6362
vertebral	0.1922	0.2003	0.1971	0.3704	0.2202	0.1680	0.2015
vowels	0.1949	0.6011	0.2576	0.2249	0.8471	0.0660	0.1421
wine	0.7044	0.7307	0.6605	0.1007	0.8156	0.5629	0.3364
yeast	0.4707	0.5043	0.4787	0.6672	0.5031	0.4696	0.4998
Avg AUROC	0.5209	0.5606	0.5060	0.2200	0.6439	0.4419	0.4530
Avg.Rank	7.38	6.28	6.77	12.19	3.81	8.85	8.83
Top1 Ratio	0.06	0.17	0.06	0.04	0.23	0.02	0.02
Top2 Ratio	0.04	0.04	0.11	0.00	0.15	0.09	0.06
Top1,2 Cumulative Ratio	0.11	0.21	0.17	0.04	0.38	0.11	0.09
Bad Ratio	0.43	0.30	0.38	0.85	0.04	0.60	0.62

Table 9: AUPRC performance of deep models

Dataset	DAGMM	DeepSVDD	GOAD	NeuTralAD	ICL	MCM	NF-SLT
ALOI	0.0589	0.1186	0.0749	0.0937	0.1058	0.0578	0.0831
Cardiotocography	0.4520	0.6307	0.3829	0.6679	0.5864	0.5245	0.6742
Hepatitis	0.4622	0.5382	0.4831	0.3848	0.3773	0.5708	0.5310
InternetAds	0.3792	0.4077	0.5967	0.4142	0.8345	0.7551	0.8399
Ionosphere	0.7007	0.9702	0.9615	0.9761	0.9673	0.9705	0.9676
Lymphography	0.5683	0.7607	0.8225	0.7839	0.7166	0.9187	0.7691
PageBlocks	0.5661	0.7479	0.8474	0.9072	0.8890	0.6263	0.8644
Pima	0.5831	0.6349	0.5237	0.5611	0.6805	0.6922	0.7117
SpamBase	0.6143	0.6307	0.5289	0.6962	0.8289	0.7449	0.8063
Stamps	0.3846	0.4953	0.3601	0.5896	0.4683	0.4703	0.6273
WBC	0.6306	0.8947	0.1882	0.5681	0.7894	0.7778	0.7923
WDBC	0.3081	0.8617	0.4889	0.9632	0.8215	0.7424	0.6770
WPBC	0.3795	0.3655	0.3177	0.3349	0.3449	0.4092	0.3727
Waveform	0.0607	0.1041	0.0460	0.2014	0.1301	0.1196	0.1743
Wilt	0.1019	0.1150	0.2689	0.2147	0.3039	0.3129	0.3903
annthyroid	0.5478	0.4005	0.3897	0.3503	0.5520	0.5679	0.6227
backdoor	0.0461	0.6203	0.0527	0.9027	0.8968	0.3615	0.5688
breastw	0.8930	0.9821	0.8605	0.9442	0.9790	0.9887	0.9797
campaign	0.2771	0.5130	0.1368	0.5227	0.4218	0.5070	0.5213
cardio	0.2816	0.7979	0.4940	0.4121	0.6640	0.6778	0.7557
celeba	0.0548	0.0317	0.0426	0.0887	0.0400	0.0663	0.0955
census	0.1036	0.1589	0.0909	0.1163	0.0898	0.2249	0.2084
cover	0.0998	0.0483	0.0120	0.4044	0.7338	0.4159	0.7204
donors	0.1729	0.9754	0.0994	0.9853	0.7310	0.9783	0.9879
fault	0.5743	0.7123	0.7061	0.7222	0.7884	0.6307	0.7412
fraud	0.0515	0.7615	0.1315	0.6194	0.6445	0.6662	0.7082
glass	0.1362	0.1954	0.1929	0.2957	0.4677	0.4202	0.3655
http	0.7535	0.9956	0.3368	0.8768	0.9894	0.5124	0.9917
landsat	0.3407	0.5269	0.3988	0.5465	0.6609	0.3967	0.4795
letter	0.1360	0.2934	0.2592	0.3964	0.6395	0.0893	0.6673
magicgamma	0.6468	0.7545	0.6289	0.7665	0.8042	0.8549	0.9094
mammography	0.0888	0.0964	0.3818	0.1382	0.3841	0.3291	0.3885
mnist	0.2500	0.2424	0.5855	0.8695	0.6798	0.5856	0.7080
musk	0.3674	0.7550	0.9980	1.0000	1.0000	0.9918	1.0000
optdigits	0.0485	0.1621	0.0360	0.2009	0.1563	0.2755	0.2725
pendigits	0.1507	0.3583	0.0283	0.5856	0.6152	0.8148	0.8073
satellite	0.7369	0.8576	0.7973	0.8667	0.8632	0.8446	0.8658
satimage-2	0.4356	0.9706	0.9641	0.9735	0.9586	0.8117	0.8984
shuttle	0.2613	0.9818	0.2937	0.9944	0.9972	0.9608	0.9695
skin	0.5066	0.4875	0.6741	0.7835	0.6725	0.5557	0.9226
smtp	0.1050	0.6110	0.2985	0.4615	0.5455	0.3936	0.4675
speech	0.0409	0.0312	0.0395	0.0362	0.0507	0.0316	0.0529
thyroid	0.6545	0.6129	0.2536	0.6164	0.5694	0.6664	0.7108
vertebral	0.2437	0.2224	0.2879	0.3645	0.3090	0.1731	0.2617
vowels	0.0798	0.5534	0.6197	0.1764	0.9254	0.1425	0.8755
wine	0.4028	0.7368	0.7695	0.8384	0.7650	0.1736	0.7601
yeast	0.5627	0.5992	0.5863	0.5513	0.5573	0.4973	0.5052
Avg AUROC	0.3468	0.5388	0.4114	0.5694	0.6170	0.5383	0.6398
Avg.Rank	10.94	7.19	9.64	5.98	5.70	7.19	4.19
Top1 Ratio	0.00	0.04	0.00	0.13	0.09	0.02	0.13
Top2 Ratio	0.00	0.09	0.00	0.13	0.11	0.04	0.15
Top1,2 Cumulative Ratio	0.00	0.13	0.00	0.26	0.19	0.06	0.28
Bad Ratio	0.85	0.36	0.62	0.32	0.23	0.36	0.04

Table 10: Performance comparison between SLT and the typicality test

Model	AUROC	AUPRC	Avg. Rank	Top2 Cum. Ratio	Fail Ratio
SLT	0.8575	0.6398	4.00	0.34	0.06
Typicality Test	0.8184	0.6270	5.87	0.23	0.28

\mathcal{D}_{train}). Therefore, The method for calculating anomaly score s is as shown in Equation 8 (Zhang et al., 2021).

$$s = \left| -\log p(\mathbf{x}) - \hat{H}_p \right| \text{ s.t. } \hat{H}_p = -\frac{1}{|\mathcal{D}_{train}|} \sum_{\mathbf{x} \in \mathcal{D}_{train}} \log p(\mathbf{x}) \quad (8)$$

To demonstrate the superiority of the simple likelihood test in the tabular domain, where counter-intuitive phenomenon rarely occur, we trained the normalizing flow model using the final selected hyperparameter combination in NF-SLT in Table 1. We recorded the results of the typicality test using Equation 8 instead of the simple likelihood test in Table 10.

According to the results in Table 10, the typicality test confirms that all performance indicators are lower than those of the simple likelihood tests. In almost all datasets, there were little changes in the performance or the performance deteriorated significantly; however, in the "yeast" dataset, which was a dataset where the simple likelihood test failed to detect, the AUROC improved by as much as 8%. Indeed, this is not a dataset where a counterintuitive phenomenon occurred; however, we confirmed that there are cases where the typicality test works much significant than the simple likelihood.

F LIKELIHOOD HISTOGRAM IN DIMENSIONALITY EXPERIMENT

Here, we present a histogram of the log-likelihood of normal and abnormal data along the dimensions from each experiment corresponding to Figure 1. The title of each figure indicates the dimension, the x-axis indicates the log-likelihood, and the y-axis indicates the number of data corresponding to the bin. In addition, the orange and blue histograms represent abnormal data and normal data, respectively.

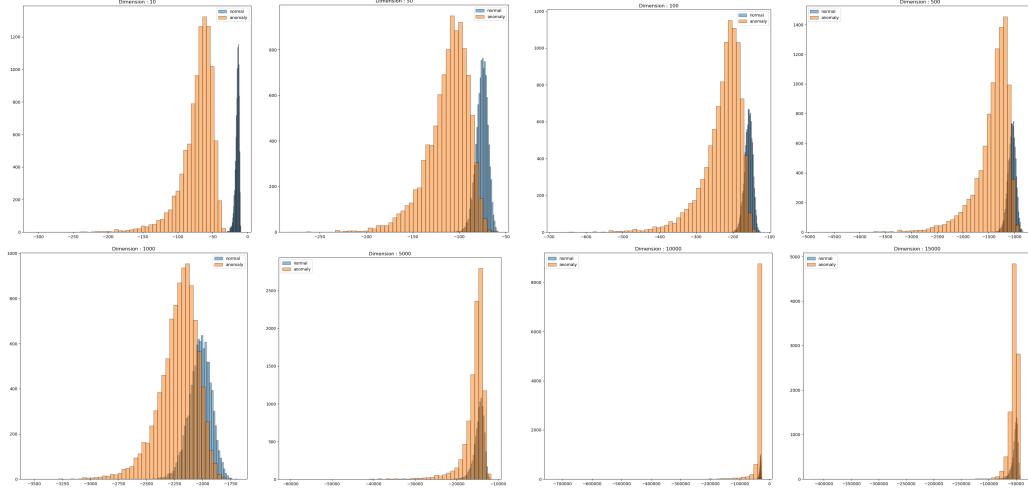


Figure 2: Histogram of log-likelihood values for the 1st subfigure in Figure 1

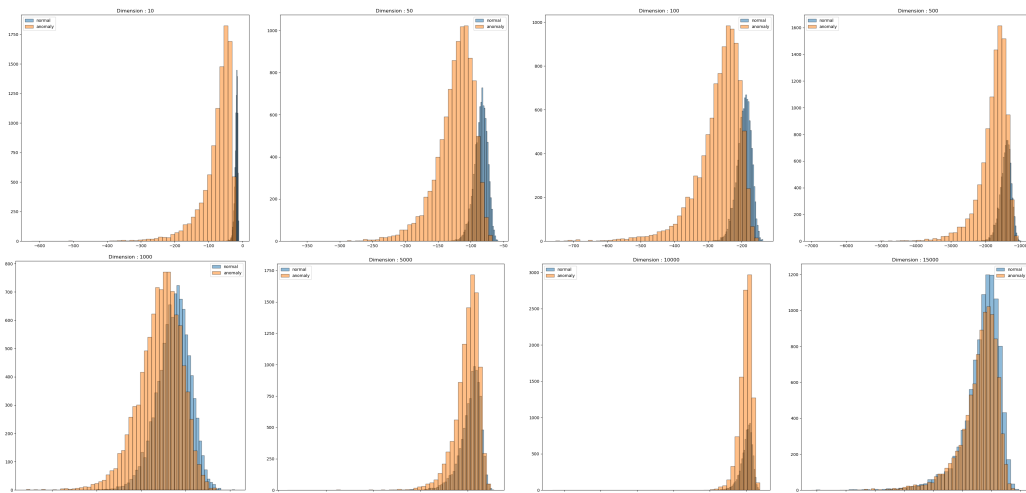


Figure 3: Histogram of log-likelihood values for the 2nd subfigure in Figure 1

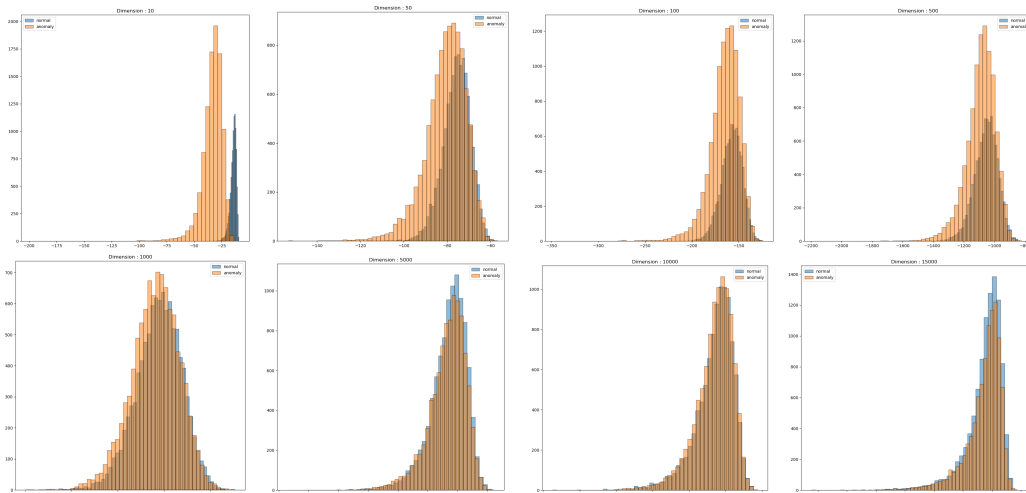


Figure 4: Histogram of log-likelihood values for the 3rd subfigure in Figure 1

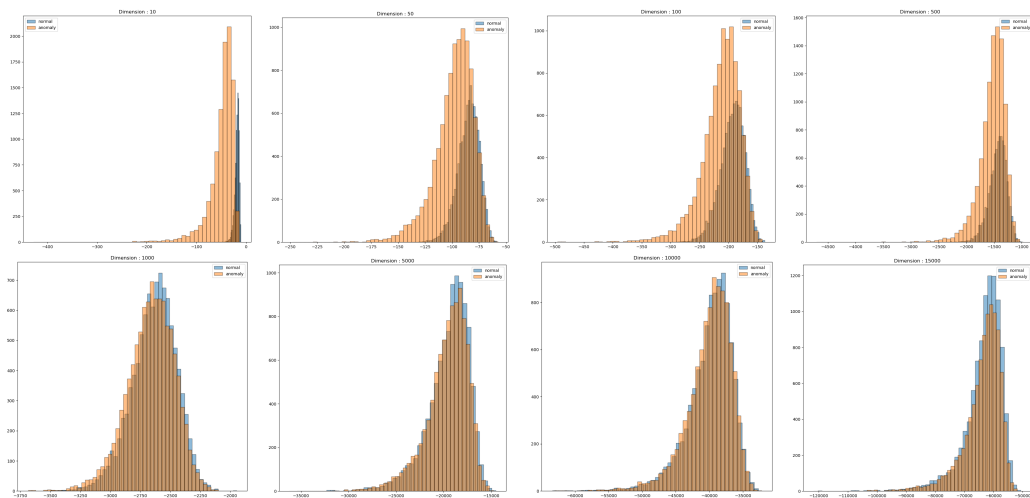


Figure 5: Histogram of log-likelihood values for the 4th subfigure in Figure 1