A. Appendix

B. Additional Experimental Results

B.1. Datasets

We conduct our method evaluations on a variety of image ND datasets, incorporating mainstream datasets like CIFAR-10, CIFAR-100, MNIST, EMNIST-Letters, Fashion-MNIST, SVHN, along with industrial datasets such as MVTecAD, FGVC-Aircraft, and medical datasets like Head CT - hemorrhage, and ISIC 2018. For FGVC-Aircraft, Due to high class similarity in FGVC, we randomly pick ten classes from the entire dataset, ensuring no shared Manufacturer. The selected classes are [91, 96, 59, 19, 37, 45, 90, 68, 74, 89]. Also for ISIC2018, It is a skin disease dataset, accessible as task 3 of the ISIC2018 challenge. It consists of seven classes. NV (nevus) is considered the normal class, and the remaining classes are regarded as anomalies.

Corrupted datasets: In this context, we have evaluated diverse image corruption datasets, encompassing well-known datasets such as CIFAR-10-C, CIFAR-100-C, MNIST-C, EMNIST-Letters-C, FMNIST-C, and SVHN-C. Each of these datasets exhibits various types of corruption, including but not limited to brightness, contrast, impulse noise, rotation, saturation, shot noise, and more. While benchmark datasets were available for CIFAR-10-C, CIFAR-100-C, MNIST-C, and FMNIST-C, for SVHN-C and EMNIST-Letters-C, we did not have pre-existing corrupted datasets. Therefore, we manually created datasets for some corruption types in these cases. Specifically, for SVHN-C, the corruption types include contrast, Gaussian blur, Gaussian noise, glass blur, impulse noise, shot noise, and speckle noise. In the case of EMNIST-Letters-C, the corruption types include brightness, contrast, glass blur, impulse noise, motion blur, rotation, saturation, scale, shear, shot noise, and general noise.

B.2. ND Experiments Details

The extensive results obtained from our experiments on Universal Novelty Detection (UNODE), conducted using standard datasets in a one-class setting, can be found in tables 7 and 8. These tables present a thorough breakdown of AUROC scores for each class. Furthermore, detailed outcomes of our UNODE experiments on corrupted datasets, also in a one-class setting, are available in tables 9 and 10, providing a comprehensive breakdown of per-class AUROC scores over various types of corruption (e.g., fog, scale, snow, shot noise, impulse_noise) .

B.3. Multi-class Setting Details

For unlabelled settings, we train our method on CIFAR-10 and CIFAR-100 as inlier datasets and evaluate that with CIFAR-10, CIFAR-100, SVHN, MNIST, FashionMNIST and ImageNet30 as outlier datasets. We show that we handle high variation setups as a general method that is robust to the diversity of inlier and outlier datasets.

In labeled settings, we explore the labeled version of the previously described setup. Specifically, we assume that each in-distribution sample includes distinctive label information. In the training procedure, we replace a binary classifier with an n-class classifier, where n represents the number of distinct in-distribution dataset classes. We employ cross-entropy as before during training. In the evaluation phase, we substitute the binary out-of-distribution (OOD) score component of the anomaly score with the maximum softmax probability, samples with lower maximum softmax probabilities are more likely to be out-of-distribution. so specially we use 1 minus the maximum softmax probability(1-MSP) as a part of score.

C. Implementation Details

C.1. Implementation Details

Model details: We employ WideResNet-50-2 as the foundational encoder network(f_{θ}), accompanied by a projection head (g_{ϕ}) comprising a 2-layer multi-layer perceptron with a 128-dimensional embedding dimensionas as well as a separate linear classification layer. As a foundational encoder network, we used both pre-trained(train on ImageNet-21k and then fine-tuning on ImageNet-1k) and from scratch versions of a WideResNet-50-2 model. We found our method performs well with either type of encoder initialization. This indicates our approach can effectively generate embeddings and predictions regardless of whether the base encoder is pre-trained or learns representations from scratch.

Training Details: Our model train for 1000 epochs using a LARS optimizer incorporating a weight decay of 1e-6 and a momentum of 0.9. To schedule the learning rate, we adopt a linear warmup for the initial 10 epochs, gradually increasing the learning rate to 1.0. Subsequently, we employ a cosine decay schedule without a restart.

Anomaly score detail: The anomaly score is composed of two parts that need to be combined: a similarity score and a binary out-of-distribution (OOD) score. In order to sum these two scores, they need to first be transformed to match the same scale. To accomplish this, we use a weighted sum to integrate the two scores. The weighting coefficient λ (balancing term) is applied to balance and rescale the binary OOD score. The calculation of λ involves two steps. First, we normalize the binary OOD scores by dividing each binary OOD score by the mean binary OOD score calculated over the entire training data set. Doing this normalization adjusts the scale of the OOD scores to match the distribution seen in the training data. The second step is to then match the scale of the normalized OOD scores to the scale of the similarity scores. This is achieved by multiplying each normalized

Table 7. Details of per-class AUROC scores for Universal Novelty Detection (UNODE) across CIFAR10, MNIST, Fashion-MNIST, SVHN, FGVC and CIFAR100 datasets.

(a) CIFAR-10

Method	Model	Classes										
		0	1	2	3	4	5	6	7	8	9	Average
UNODE	From Scratch	95.7	99.3	91.4	87.5	94.3	94.2	97.6	98.1	98.3	97.5	95.4
	Pre-trained	97.0	98.8	96.0	92.4	96.5	94.7	98.5	98.6	98.6	97.8	96.9

(b) MNIST

Method	Model	Classes										
		0	1	2	3	4	5	6	7	8	9	Average
UNODE	From Scratch	99.0	85.6	99.2	98.1	98.1	98.3	99.4	98.4	98.8	99.4	97.4
	Pre-trained	99.3	98.6	99.1	97.9	99.2	99.0	99.5	99.7	98.4	99.3	99.0

(c) FashionMNIST

Method	Model	Classes										
		0	1	2	3	4	5	6	7	8	9	Average
UNODE	From Scratch	91.0	99.3	94.4	89.1	90.1	96.6	85.1	97.9	98.9	99.2	94.2
	Pre-trained	90.7	99.5	89.7	92.4	92.0	95.6	79.1	98.7	98.2	98.2	93.4

(d) SVHN

Method	Model	Classes										
		0	1	2	3	4	5	6	7	8	9	Average
UNODE	From Scratch	97.4	95.1	96.6	94.5	97.9	96.9	95.3	97.7	96.4	95.6	96.3
	Pre-trained	92.5	81.4	92.5	89.9	94.5	92.6	92.9	93.3	88.5	92.4	91.0

(e) FGVC

Method	Model	Classes										
		0	1	2	3	4	5	6	7	8	9	Average
UNODE	From Scratch	78.1	75.6	55.2	85.4	88.3	84.3	82.9	92.0	86.6	73.5	80.2
	Pre-trained	75.0	84.0	61.5	86.9	90.4	81.4	91.7	93.3	87.3	74.9	82.9

(f) CIFAR-100

Method	Model	Classes																				
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Average
UNODE	From Scratch	92.5	93.2	97.5	95.0	97.2	94.0	96.8	92.7	93.9	97.6	97.0	93.8	93.4	89.2	96.1	85.3	91.0	99.3	96.8	94.9	94.4
	Pre-trained	92.0	93.0	96.9	92.2	97.1	91.7	95.7	92.0	94.2	96.3	97.5	92.4	92.9	88.6	93.9	85.5	91.8	98.9	94.7	95.3	93.6

Table 8. Details of per-class AUROC scores for Universal Novelty Detection (UNODE) across MVTecAD and EMNIST-Letters datasets.

(a) MVTecAD

Method	Model	Classes															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Average
UNODE	From Scratch	99.2	99.7	92.6	92.6	99.6	99.8	91.5	99.8	92.7	90.4	97.2	81.8	88.9	100.0	88.9	94.3
	Pre-trained	98.1	100.0	94.9	96.0	99.8	99.9	83.9	99.8	95.8	96.7	95.9	87.0	98.5	100.0	87.1	95.6

(b) EMNIST-Letters

Method	Model	Classes																									
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	Average
UNODE	From Scratch	98.4	98.5	99.5	99.4	99.4	98.7	98.0	98.4	96.3	98.3	98.4	95.8	99.6	97.3	99.0	99.4	98.6	96.2	99.5	98.4	98.5	99.1	99.2	99.7	96.9	98.9	98.4
	Pre-trained	97.9	97.2	98.4	99.0	99.3	97.7	97.0	97.5	97.8	96.1	99.1	98.4	98.0	96.2	98.7	98.8	97.8	98.0	99.2	97.9	98.2	98.8	97.6	98.7	97.1	99.4	98.1

binary OOD score by the mean of the norms of all the embeddings from the training data. By matching the scales in this way through the two computation steps of normalization and rescaling, the binary OOD score becomes directly compatible and summable with the similarity score. The end result is that applying the weighting coefficient λ(balancing term) to properly transform the binary OOD score allows it to be combined with the similarity score into a final, unified anomaly score with common scaling and significance. The two components are calibrated to contribute equally based on their training data distributions.

$$O_{\text{ours}}(\mathbf{x}; \mathcal{D}_{\text{in}}^{\text{train}}) = O_{\text{sim}}(\mathbf{x}; \mathcal{D}_{\text{in}}^{\text{train}}) + \lambda \cdot O_{\text{bin-OOD}}(\mathbf{x}) \tag{16}$$

Table 9. Details of per-class AUROC (%) scores for Our Universal Novelty Detection (UNODE) across corrupted datasets such as MNIST-C, CIFAR-10-C and CIFAR-100-C with various types of corruption (e.g., fog, scale, snow, shot noise, impulse_noise) are presented.

(a) MNIST-C

Normal Class	Corruption Type															
	Normal	brightness	canny_edges	dotted_line	fog	glass_blur	impulse_noise	motion_blur	rotate	scale	sheer	shot_noise	spatter	stripe	translate	zigzag	Average
0	99.3	93.4	93.2	98.8	64.1	77.3	95.4	79.0	98.3	99.5	98.3	96.3	91.8	96.4	99.3	97.4	91.9
1	99.7	95.9	99.8	94.6	70.4	98.3	96.9	98.0	97.8	99.4	98.0	99.1	94.6	98.4	99.5	88.4	95.3
2	96.5	94.5	84.4	94.3	69.6	72.7	69.3	88.1	91.9	95.0	93.5	92.5	88.3	83.5	96.4	86.4	86.7
3	97.1	97.2	90.3	97.0	78.1	65.0	83.0	88.4	87.0	92.0	96.9	94.6	92.3	91.4	96.7	89.4	89.3
4	98.9	97.6	74.3	97.3	78.8	71.1	83.1	84.1	94.2	98.0	96.0	95.6	92.7	91.6	98.7	91.8	89.7
5	96.9	94.6	87.4	94.5	72.4	84.3	81.5	91.9	93.6	95.8	96.9	94.7	90.3	70.8	97.1	88.4	88.9
6	98.2	95.1	83.7	97.4	70.0	82.8	89.5	91.0	96.2	97.6	96.7	95.5	94.8	89.2	98.1	90.7	91.2
7	99.0	95.9	94.6	94.1	65.5	82.5	76.2	90.8	93.0	98.2	97.5	95.8	90.0	85	98.4	92.1	90.0
8	97.6	97.3	32.7	96.1	78.6	60.2	81.0	82.4	94.2	95.7	98.0	92.6	91.1	88.8	96.4	89.2	85.0
9	98.2	95.0	71.0	96.7	63.5	71.4	73.5	87.0	94.3	94.6	95.8	96.2	92.0	84.3	97.5	88.4	86.7
Average	98.1	95.7	81.1	96.1	71.1	76.6	82.9	88.1	94.1	96.6	96.8	95.3	91.8	87.9	97.8	90.2	89.5

(b) CIFAR-10-C

Normal Class	Corruption Type																	
	identity	brightness	contrast	defocus_blur	elastic_transform	fog	frost	gaussian_blur	impulse_noise	jpeg_compression	motion_blur	pixelate	saturate	shot_noise	snow	spatter	speckle_noise	zoom_blur	Average
0	97.0	94.4	84.9	92.4	91.8	85.5	88.7	91.8	70.4	93.7	92.1	86.9	94.0	75.9	92.9	82.0	68.0	94.1	87.5
1	98.8	98.1	77.6	97.5	95.8	95.0	94.8	97.0	78.8	97.4	96.7	92.7	97.9	93.6	97.2	92.7	90.6	98.0	93.9
2	96.0	93.8	70.6	88.9	89.8	83.4	87.8	87.7	69.6	91.9	88.1	82.6	93.6	83.3	90.9	79.3	81.7	92.1	86.1
3	92.4	89.1	76.1	84.6	86.4	79.7	81.9	83.6	62.1	87.7	84.0	78.1	89.5	73.7	86.9	80.1	72.5	87.9	82.0
4	96.5	94.8	83.8	89.8	92.5	85.5	91.3	88.7	81.1	92.7	89.4	89.6	94.4	89.7	92.2	84.7	88.9	93.0	89.9
5	94.7	91.7	76.9	87.2	89.3	84.9	86.2	86.2	66.0	91.9	86.9	79.1	91.5	77.4	91.8	77.3	73.5	90.7	84.6
6	98.5	97.7	83.4	95.4	95.7	89.1	96.6	94.7	88.3	96.3	94.1	93.5	97.2	94.9	96.1	94.6	94.7	96.9	82.0
7	98.6	97.6	85.8	95.6	95.3	89.2	93.6	94.9	76.1	96.8	94.2	90.7	97.6	91.1	96.8	91.9	88.5	97.0	92.8
8	98.6	98.2	90.0	96.5	95.8	91.3	91.4	96.0	79.4	97.7	96.3	94.7	97.8	81.7	94.8	90.0	76.0	97.5	92.4
9	97.8	96.3	77.1	94.9	93.7	92.5	94.3	94.6	83.7	95.9	94.4	88.6	96.2	91.7	95.1	90.6	89.8	96.2	92.4
Average	96.9	95.2	80.6	92.3	92.6	87.6	90.7	91.5	75.5	94.2	91.6	87.7	95.0	85.3	93.5	86.3	82.4	94.3	89.2

(c) CIFAR-100-C

Normal Class	Corruption Type																	
	identity	brightness	contrast	defocus_blur	elastic_transform	fog	frost	gaussian_blur	impulse_noise	jpeg_compression	motion_blur	pixelate	saturate	shot_noise	snow	spatter	speckle_noise	zoom_blur	Average
0	92.5	94.4	84.9	92.4	91.8	85.5	88.7	91.8	70.4	93.7	92.1	86.9	94.0	75.9	92.9	82.0	68.0	94.1	86.7
1	93.2	92.0	77.5	89.0	88.5	77.5	85.2	88.1	67.6	90.3	87.9	80.6	91.3	77.9	86.0	76.2	75.2	91.4	83.1
2	97.5	93.8	70.6	88.9	89.8	83.4	87.8	87.7	69.6	91.9	88.1	82.6	93.6	83.3	90.9	79.3	81.7	92.1	85.1
3	95.0	89.1	76.1	84.6	86.4	79.7	81.9	83.6	62.1	87.7	84.0	78.1	89.5	73.7	86.9	80.1	72.5	87.9	80.9
4	97.2	94.8	83.8	89.8	92.5	85.5	91.3	88.7	81.1	92.7	89.4	89.6	94.4	89.7	92.2	84.7	88.9	93.0	89.2
5	94.0	91.7	76.9	87.2	89.3	84.9	86.2	86.2	66.0	91.9	86.9	79.1	91.5	77.4	91.8	77.3	73.5	90.7	83.5
6	96.8	97.7	83.4	95.4	95.7	89.1	96.6	94.7	88.3	96.3	94.1	93.5	97.2	94.9	96.1	94.6	94.7	96.9	93.9
7	92.7	89.1	67.7	88.1	87.0	78.0	85.8	87.4	70.4	87.5	87.9	78.4	86.8	80.6	87.8	79.2	79.6	90.4	82.7
8	93.9	98.2	90.0	96.5	95.8	91.3	91.4	96.0	79.4	97.7	96.3	94.7	97.8	81.7	94.8	90.0	76.0	97.5	91.7
9	97.6	96.3	77.1	94.9	93.7	92.5	94.3	94.6	83.7	95.9	94.4	88.6	96.2	91.7	95.1	90.6	89.8	96.2	91.8
10	97.0	97.1	93.1	96.2	96.2	89.7	92.3	95.9	68.1	96.3	96.5	93.2	94.8	74.1	92.8	64.5	72.4	96.7	88.3
11	93.8	90.1	71.2	89.6	87.0	79.0	79.0	88.8	50.2	87.8	88.0	74.7	90.6	69.8	86.5	81.6	70.3	91.0	80.3
12	93.4	90.8	70.5	89.9	88.4	76.7	85.0	89.0	60.4	89.9	88.2	74.5	90.2	78.0	87.6	80.3	74.7	92.5	82.7
13	89.2	85.9	69.9	85.3	81.8	73.8	79.9	84.1	64.7	84.5	81.7	75.5	85.7	78.4	82.7	73.5	75.4	87.2	79.4
14	96.1	88.7	68.1	68.1	86.8	78.9	83.7	87.9	56.7	86.4	85.5	70.2	89.2	73.5	87.5	73.4	69.5	91.3	79.2
15	85.3	84.3	63.5	85.3	81.1	70.1	78.8	84.3	59.9	83.1	81.8	70.7	83.0	69.3	80.1	73.0	68.5	87.0	76.7
16	91.0	89.8	73.4	87.6	87.0	80.2	82.4	86.5	58.6	88.5	84.7	75.8	89.0	67.5	84.6	77.5	64.7	90.7	80.5
17	99.3	98.3	89.9	97.8	97.7	92.8	97.3	97.4	77.9	97.7	96.9	93.2	98.1	94.2	95.5	91.5	93.7	98.6	94.6
18	96.8	92.3	63.2	94.2	86.6	84.2	85.3	93.6	69.9	88.3	88.8	73.9	91.9	78.9	87.3	84.8	77.2	95.1	84.5
19	94.9	92.5	71.4	93.4	89.5	81.0	86.8	92.8	61.6	90.7	88.7	82.0	91.5	77.7	88.0	80.0	70.6	94.4	84.3
Average	94.4	92.3	76.1	89.7	89.6	82.7	87.0	90.0	68.3	90.9	89.1	81.8	91.8	79.4	89.3	80.7	76.8	92.7	85.2

C.2. Augmentation Details

In the development of the AutoAugOOD pipeline, a diverse array of augmentation techniques has been employed, specifically Rotation, Permute, Gaussian Noise, CutOut, CutPaste, Sobel, Blur, and MixUp. These methods are meticulously applied to each dataset variant, resulting in a range of transformed datasets. Following the transformation process, embeddings for each augmented dataset are extracted using CLIP ResNet 50. The next critical step involves the application of tsne_1, implemented using the sklearn library, on the concatenated embeddings from all dataset variants. This procedure yields one-dimensional (1D) numerical representations.

Subsequently, these 1D representations of the augmented datasets, juxtaposed with those of the original dataset, form the basis for calculating a KL divergence score, also computed using the sklearn library. This score effectively quantifies the divergence between the distribution of augmented datasets and the original dataset. Finally, to translate these KL divergence scores into a more interpretable format, a softmax function is utilized. This conversion yields a probability score for each augmentation type.

It is crucial to emphasize that all these processes — from data augmentation to the calculation of probability scores using softmax — are conducted prior to the training stage of the model.

C.3. Leveraging Pre-trained Models for OOD Detection

Numerous methods employ large-scale pre-trained models, such as MSAD, for outlier detection, and their performance heavily depends on the rich features learned by their backbone, such as ViT. However, these pipelines do not function

Table 10. Details of per-class AUROC (%) scores for Our Universal Novelty Detection (UNODE) across corrupted datasets such as EMNIST-Letters-C with various types of corruption (e.g., fog, scale, snow, shot noise, impulse_noise) are presented.

(a) EMNIST-Letters-C

Normal Class	Corruption Type										
	brightness	contrast	glass_blur	impulse_noise	motion_blur	rotate	saturate	scale	sheer	shot_noise	Average
1	96.1	77.7	39.9	54.2	83.3	92.9	96.2	97.9	92.3	55.4	78.6
2	96.6	81.8	48.1	70.4	76.7	95.8	97.1	98.0	91.9	70.0	82.6
3	98.2	68.9	53.3	38.6	93.7	97.0	99.1	99.0	96.3	53.0	79.7
4	97.6	85.9	44.6	54.9	77.1	97.5	98.7	99.3	97.2	76.9	83.0
5	98.7	89.0	35.6	58.6	86.8	96.6	98.1	99.2	94.8	47.9	80.5
6	97.4	92.7	66.7	61.7	80.7	89.0	98.5	97.9	87.6	83.7	85.6
7	92.3	72.8	44.7	66.0	67.0	93.2	94.3	96.5	90.0	65.2	78.2
8	93.8	83.4	58.4	56.9	87.8	94.5	97.4	97.7	94.6	82.8	84.7
9	87.4	89.4	89.0	39.2	93.5	95.9	96.7	96.7	96.0	80.7	86.5
10	85.2	90.3	70.2	57.5	83.4	91.4	96.0	96.5	91.3	89.1	85.1
11	97.7	87.3	51.1	63.2	88.1	94.8	99.1	98.7	94.8	78.5	85.3
12	83.2	94.3	91.6	59.9	90.1	95.9	94.4	97.2	93.5	88.5	88.9
13	97.5	82.3	29.4	62.8	86.6	94.7	98	97.8	94.7	81.9	82.6
14	95.3	75.0	37.9	42.1	87.0	92.8	96.9	96.2	93.1	51.3	76.8
15	99.2	78.0	57.8	44.1	93.6	98.2	98.5	99.1	97.7	36.6	80.3
16	98.3	91.9	72.5	58.4	91.0	96.7	98.8	98.9	95.2	79.9	88.2
17	94.3	78.5	54.0	69.9	75.1	94.6	96.9	97.2	93.5	71.7	82.6
18	94.3	75.7	60.6	32.2	90.5	93.3	95.5	97.1	89.4	51.9	78.1
19	98.9	79.3	38.9	51.5	92.5	95.9	99.0	98.8	92.2	52.8	80.0
20	93.9	91.2	73.1	52.5	85.7	89.8	98.9	98.1	90.5	88.7	86.2
21	98.2	87.5	52.8	48.5	90.3	95.5	98.5	98.8	94.9	59.1	82.4
22	97.7	85.1	75.9	45.2	90.2	96.5	98.8	99.1	95.1	66.9	85.1
23	98.1	94.2	33.8	73.9	85.0	95.1	98.5	97.5	96.9	83.6	85.7
24	98.0	94.9	53.6	63.7	88.4	90.2	98.9	98.9	95.1	70.4	85.2
25	88.5	85.3	77.6	64.1	86.4	93.7	96.8	97.6	92.7	86.1	86.9
26	97.7	93.7	47.1	69.3	76.6	92.4	99.3	99.1	88.7	82.3	84.6
Average	95.2	84.9	56.1	56.1	85.7	94.4	97.7	98	93.5	70.6	83.2

effectively when the pre-trained models are replaced with models trained from scratch. Interestingly, our proposed method achieves significant performance improvements with both pre-trained and from-scratch models.

D. Ablation Study: In-Depth Analysis and Results

D.1. Evaluation of AutoAugOOD Replacement

Methodology:

- We replaced AutoAugOOD with a set of common fixed hard augmentations, including rotation (as per [49]), and cut-and-paste techniques [25].
- AutoAugment [9], a method designed for classification improvement, was also tested.
- Additionally, we incorporated FakeD [28] into our experiments for a comprehensive comparison.

Results:

- Performance metrics (accuracy, F1-score, etc.) for each replacement were recorded.
- A comparative analysis was conducted to assess the impact of each augmentation technique on the model's ability to detect novel instances.

Interpretation:

- The results, detailed in Table 4, highlight the effectiveness of AutoAugOOD over traditional augmentation methods.
- The comparison with AutoAugment and FakeD provides insights into the adaptability and robustness of our method in diverse scenarios.

D.2. Analysis of Training Objectives

Methodology:

- We dissected our training objective, isolating the contrastive and classification losses.
- Each component was individually omitted or modified to observe its impact on the overall performance.

Results:

- Table 5 presents the performance variations under different training objective configurations.
- Metrics such as loss convergence rate and classification accuracy were primarily focused on.

Interpretation:

- This analysis elucidates the contribution of each component in our training objective, emphasizing the synergy between contrastive and classification losses for optimal performance.

D.3. Evaluation of Novelty Score Components

Methodology:

- We examined the individual effects of each component in our novelty score, namely the binary score (14) and similarity score (13).
- Variations were introduced in these components to assess their individual and combined impact.

Results:

- Table 6 showcases how each component influences the

model's ability to differentiate between novel and familiar instances.
- The effectiveness of each scoring method was quantified and compared.

Interpretation:
- This segment of the study provides a deeper understanding of how each scoring component contributes to the overall effectiveness of our novelty detection approach.

D.4. Why Our Augmentation Pipeline (AutoAugOOD) is the Best Fit for Near-OOD Generation

In numerous studies, including those focusing on adversarial robustness, the utility of leveraging an additional dataset to enhance model performance has been substantiated, provided certain conditions are met. A crucial criterion in this context is the fidelity, diversity, and relevance of this supplementary data to the inlier training set. Intriguingly, AutoAugOOD excels in generating diverse data through a variety of transformations while concurrently ensuring the crafted data exhibit high Out-Of-Distribution (OOD) characteristics. This approach is particularly synergistic with the principles of contrastive learning. In this domain, several studies have demonstrated that the inclusion of diverse negative pairs is instrumental in fostering the development of more effective representations. By aligning with these principles, AutoAugOOD not only introduces diversity but also maintains a high degree of relevance to the original data distribution, thereby making it an optimal choice for Near-OOD generation.

E. Additional Methodological Details

OOD samples can be broadly classified into two categories: pixel-level and semantic-level. In pixel-level OOD detection, the distinction between In-Distribution (ID) and OOD samples lies in their local appearance, despite them being semantically similar. For example, a broken glass, in contrast to an intact one, can be identified as an OOD sample due to its altered local appearance, even though both are semantically related to the concept of 'glass'. On the other hand, semantic-level OOD samples exhibit differences at a conceptual or semantic level. An illustrative case is the categorization of a cat as an OOD sample in a dataset where the ID semantics are centered around dogs, signifying a divergence in underlying concepts.

Interestingly, AutoAugOOD demonstrates the capability to generate a wide array of OOD samples, encompassing both pixel-level and semantic-level variations. This is achieved through the application of diverse, potentially detrimental transformations. For instance, the 'cutpaste' transformation is adept at creating samples with textural defects, thereby contributing to pixel-level OOD detection. Conversely, transformations such as rotation are instrumental in generating semantic-level OOD samples, as they alter the conceptual understanding of the image. This versatility of AutoAugOOD in crafting various types of OOD samples underscores its utility in enhancing robustness against a broad spectrum of OOD scenarios.

E.1. Detailed Baselines

In this paper, we have conducted a comprehensive comparison of our proposed method with several state-of-the-art (SOTA) techniques, including SSD [40], ReCoNTRAST, FiTYMI , and FastFlow. In the subsequent sections, we will briefly outline each of these methods to provide a clearer understanding of their functionalities and how they compare with our approach:

The paper "SSD: A Unified Framework for Self-Supervised Outlier Detection" proposes SSD, an outlier detection method that uses only unlabeled in-distribution data. It employs self-supervised representation learning followed by a Mahalanobis distance-based detection in the feature space. The paper demonstrates that SSD outperforms most existing detectors based on unlabeled data and can even match or exceed the performance of supervised training-based detectors. The framework also includes extensions for few-shot outlier detection and the incorporation of training data labels when available.

The approach "Supervised SimCLR for Outlier Detection" does not appear in the search results. However, there is a related method called SSD (Self-Supervised Outlier Detection) that uses self-supervised representation learning followed by a Mahalanobis distance-based detection in the feature space. SSD outperforms most existing detectors based on unlabeled data and can even match or exceed the performance of supervised training-based detectors.

The paper "ReContrast: Domain-Specific Anomaly Detection via Contrastive Reconstruction" introduces a novel unsupervised anomaly detection (UAD) method called ReContrast. This method optimizes the entire network to reduce biases towards pre-trained image domains and aligns the network with the target domain. It combines the principles of contrastive learning and feature reconstruction to prevent training instability, pattern collapse, and identical shortcut. The paper demonstrates the effectiveness of ReContrast through extensive experiments across industrial defect detection benchmarks and medical image UAD tasks, where it outperforms current state-of-the-art methods.

The paper "FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows" introduces FastFlow, a method for unsupervised anomaly detection and localization. FastFlow uses 2D normalizing flows as a probability distribution estimator, transforming input visual features into a tractable distribution during the training phase. This approach can be used with any deep feature extractors like ResNet and vision transformer. The paper shows that FastFlow surpasses previous state-of-the-art methods in

terms of accuracy and inference efficiency, achieving 99.4% AUC in anomaly detection.

The paper "Towards Total Recall in Industrial Anomaly Detection" introduces PatchCore, an algorithm for cold-start anomaly detection that leverages knowledge of only nominal (non-defective) examples. PatchCore uses a maximally representative memory bank of nominal patch-features, offering competitive inference times while achieving state-of-the-art performance for both detection and localization. On the MVTec AD benchmark, PatchCore achieves an image-level anomaly detection AUROC score of up to 99.6%, significantly reducing the error compared to the next best competitor.

The paper "Fake It Till You Make It: Towards Accurate Near-Distribution Novelty Detection" addresses image-based novelty detection, particularly in the "near-distribution" setting where differences between normal and anomalous samples are subtle. The authors demonstrate that existing methods experience up to a 20% decrease in performance in this setting. They propose leveraging a score-based generative model to produce synthetic near-distribution anomalous data, which is then used to fine-tune a model for distinguishing such data from normal samples. The method is evaluated quantitatively and qualitatively, showing significant improvements over existing models and consistently reducing the performance gap between near-distribution and standard novelty detection. The approach is assessed across diverse applications such as medical images, object classification, and quality control, demonstrating its effectiveness.

F. Additional Experimetns

Here we have replaced our UNODE default backbone with different architectures. Specifically, UNODE incorporates two different components, including AutoAugOOD and the detector backbone. Here, we examined the sensitivity of UNODE to different architectures, highlighting our method's superior performance across various architectures. Moreover, we considered extra datasets, including Weather [13], Birds [44], and ImageNet30 [19]. The results are presented in Table 11.

Table 11. The table presents the performance of our method using different backbones compared to two recent SOTA methods [ID 1,2]. [ID 3,4] correspond to the results mentioned in the main paper. The default detection backbone mentioned in the paper is Wide-ResNet and AugOOD backbone is a pre-trained ResNet. The experiments in this table demonstrate our performance using different detection backbones (IDs 3-6) and various AutoAug backbones with different pre-training types, including Supervised (Sup), Contrastive (Con), Supervised Contrastive (Con-Sup). Star* indicate that the model is pre-trained.

Exp. ID	Methods	Backbones		Datasets						Mean
		Detector	AugOOD	MNIST	CIFAR10	MVTECAD	Birds	Weather	ImageNet	
1	MSAD	Res152*	N/A	96.0	97.2	87.2	96.7	92.4	96.9	94.4
2	FITYMI	ViT-B_16*	N/A	75.2	99.1	86.4	98.5	97.0	97.5	92.2
3	UNODE	Wide-Res50-2	Res18-Sup*	97.4	95.4	95.3	94.8	95.1	94.6	95.4
4	UNODE	Wide-Res50-2*	Res18-Sup*	99.0	96.9	95.8	96.0	94.7	97.3	96.6
5	UNODE	R50+ViT-B_16*	Res18-Sup*	98.7	97.8	95.3	94.5	95.8	96.3	96.4
6	UNODE	Res18	Res18-Sup*	98.3	95.0	94.8	93.5	94.8	93.8	95.0
7	UNODE	Wide-Res50-2	Res18-Con*	97.3	95.6	94.4	95.1	95.2	93.8	95.2
8	UNODE	Wide-Res50-2	Res18-SupCon*	97.1	95.0	96.2	94.0	94.6	95.1	95.3
9	UNODE	Wide-Res50-2	ViT-B_16-Sup*	96.8	95.7	95.3	94.4	93.5	94.7	95.1
10	UNODE	Wide-Res50-2	VGG19-Sup*	96.4	95.2	95.0	94.5	93.8	94.1	94.8