

DAA: Amplifying Unknown Discrepancy for Test-Time Discovery

(Appendix)

A Comparison of TTD task and other Category Discovery task

We compare TTD with related settings. Out-of-Distribution (OOD) [7] detection neither discovers novel classes nor adapts during inference. TTA [1] handles distribution shifts via self-supervised learning but assumes all test samples belong to known classes. NCD [4] and GCD [14] both aim to identify unknown classes at test time under an offline inference paradigm through clustering the entire test set. OCD [3] resembles TTD in identifying known and unknown classes, but does not incorporate test-time learning from novel classes.

Test-Time Discovery (TTD) is a challenging task that focuses on class shifts rather than domain shifts during test time. It requires the model to not only discover new classes but also classify them accurately while maintaining robust performance on previously seen classes. This dual requirement is particularly demanding due to several intrinsic complexities: the intricate nature of class discovery, the scarcity of labeled data for new classes, and the often ambiguous boundaries between classes. The key challenges in TTD can be summarized as follows:

- (1) Distinguishing between the discovery of new classes and the identification of already discovered ones.
- (2) Learning and adapting to new classes with limited sample sizes during testing phase.
- (3) Avoiding catastrophic forgetting, where the process of learning new classes can inadvertently degrade the model’s performance on previously learned classes.

Table 1: Comparison between different category discovery settings.

Type	Train	Test	Discovery	Test-time Learning
OOD	Known classes	Shift Known classes	N/A	N/A
TTA	Known classes	Shift Known classes	N/A	Shift Known classes
NCD	Known classes	Unknown classes	Post	N/A
GCD	Known classes	Known classes + Unknown classes	Post	N/A
OCD	Known classes	Known classes + Unknown classes	Real-time	N/A
TTD	Known classes	Known classes + Unknown classes	Real-time	Unknown classes

B Details of datasets

We conduct our experiments using three widely recognized benchmark datasets: CIFAR100 (C100) [5], Caltech-UCSD Birds-200-2011 (CUB) [15], and Tiny ImageNet [6]. Each of these datasets is systematically partitioned into known and unknown classes. The model undergoes training on the known training set and is subsequently evaluated on a mixed set containing both known and unknown classes. Since the primary objective of these datasets is to facilitate new class discovery, we follow the HM [8] and use the transformed versions as CIFAR100D, CUB-200D, and Tiny-ImageNetD to reflect this adaptation.

The dataset partitioning follows the scheme outlined in Table 2. Specifically, during the training phase, we divide the training set into known and unknown classes based on their class index order. For instance, in CIFAR100, the first 70 classes are designated as known, while the remaining 30 classes are treated as unknown. The supervised training process is then conducted using only the known classes within the labeled training set. More precisely, CIFAR100D consists of classes 0–69 (70 known classes in total), CUB-200D includes classes 0–139 (140 known classes in total), and Tiny-ImageNetD comprises classes 0–139 (140 known classes in total), all of which are utilized for training.

During the test phase, the model is evaluated on the entire unlabeled test set, which includes samples from all categories, enabling new class discovery and classification. While the category labels remain

structured according to the original known-unknown splits (e.g., 70+30 for CIFAR100D and 140+60 for CUB-200D and Tiny-ImageNetD), these labels are only used for metric evaluation and are not provided to the model during inference. This setup ensures a realistic scenario for open-world learning, where the model must autonomously identify and categorize previously unseen classes.

Table 2: Statistic of the used datasets.

Dataset	Labeled	CIFAR100D			CUB-200D			Tiny-ImagenetD		
		Known	Unknown	No. of samples	Known	Unknown	No. of samples	Known	Unknown	No. of samples
TrainSet	✓	70	0	42000	140	0	4195	140	0	70000
TestSet		70	30	10000	140	60	5794	140	60	10000

C Metric Definition

The evaluation process is structured into two distinct parts: one focusing on known classes and the other on unknown classes. To ensure a comprehensive assessment, we follow HM [8] and employ both real-time evaluation and post-evaluation strategies. For test-time evaluation, real-time performance is a critical factor. Thus, we compute and report real-time scores for all evaluation metrics as the model processes each test sample. This approach provides immediate insights into the model’s performance and enables dynamic tracking of classification accuracy and discovery efficiency. Alongside these real-time scores, we present the final accumulated values, which represent the overall average performance across the entire test set. In addition, recognizing that traditional novel class discovery (NCD) methods typically rely on post-evaluation, we also incorporate this approach for comparative analysis. In post-evaluation, all test samples are revalidated collectively after the entire test phase is complete. This post-hoc evaluation allows for a more refined assessment by leveraging the full distribution of test samples, potentially improving class assignment and clustering accuracy. By providing both real-time and post-evaluation scores, we ensure a thorough and balanced evaluation of the model’s effectiveness in handling both known and unknown classes.

C.1 Metrics for known classes

For the evaluation of known classes, we employ two key metrics to comprehensively assess the model’s performance: Known Accuracy (KA) and Known Forgetting (KF).

(1) *Known Accuracy (KA)*. KA measures the traditional classification accuracy of the model on known classes, reflecting its ability to correctly recognize and classify samples that were part of the training set. This metric serves as a standard benchmark for evaluating the retention of previously learned knowledge:

$$\text{KA} = \mathbb{E}_{c \in \mathcal{Y}_{\text{known}}} \frac{1}{|\mathcal{D}_c^{\text{test}}|} \sum_{x \in \mathcal{D}_c^{\text{test}}} \mathbf{1}(\hat{y}(x) = c), \quad (1)$$

where $\mathcal{Y}_{\text{known}}$ is set of predefined known classes, $\mathcal{D}_c^{\text{test}}$ is test samples with ground-truth class c , $\hat{y}(x)$ is the predicted label for sample x , $\mathbf{1}(\cdot)$ is the indicator function (1 if prediction matches true class c , 0 otherwise)

(2) *Known Forgetting (KF)*. KF, on the other hand, quantifies the degree of performance degradation on known classes over time. It captures the extent to which the model forgets previously learned information as it encounters new data, particularly when adapting to novel classes. A lower KF score indicates better knowledge retention, while a higher score suggests significant forgetting.

$$\text{KF} = \text{KA}_{\text{post}} - \text{KA}_{\text{pre}}, \quad (2)$$

where KA_{post} and KA_{pre} are the KA computed on all test data with known classes, before and after TTD.

C.2 Metrics for unknown classes

For unknown classes, since the predicted label space $\mathcal{Y}_{\text{seen}}^{\text{GT}}$ does not match the cluster label space $\mathcal{Y}_{\text{seen}}$, we propose agreement metrics to assess effectiveness. In the test set $\mathcal{D}^{\text{test}}$, a sample x has a true label $y \in \mathcal{Y}_{\text{seen}}^{\text{GT}}$ and a predicted cluster label $\hat{y}(x) \in \mathcal{Y}_{\text{seen}}$. We define the subset of $\mathcal{D}^{\text{test}}$ with true label c as $\mathcal{D}_c^{\text{test}}$, and the cluster with predicted label p as $\mathcal{C}_p^{\text{test}}$.

80 (1) *True-label Agreement ratio (TA)*. This metric measures the maximum proportion of samples from
 81 a given true class that are predicted as the same class:

$$\text{TA} = \mathbb{E}_{c \in \mathcal{Y}_{\text{seen}}^{\text{GT}}} \frac{1}{|\mathcal{D}_c^{\text{test}}|} \max_{p \in \mathcal{Y}_{\text{seen}}} \left(\sum_{x \in \mathcal{D}_c^{\text{test}}} \mathbf{1}[\hat{y}(x) = p] \right), \quad (3)$$

82 where $\mathbf{1}(\cdot)$ is the indicator function (1 if true, 0 otherwise).

83 (2) *True-label Entropy (TE)*. This metric measures the average entropy $H(\cdot)$ of the predicted labels
 84 for samples with that true class:

$$\text{TE} = \mathbb{E}_{c \in \mathcal{Y}_{\text{seen}}^{\text{GT}}} H(\{\hat{y}(x) | x \in \mathcal{D}_c^{\text{test}}\}) \quad (4)$$

85 where $H(\cdot)$ is Shannon entropy of predicted label distribution which is used to quantifies uncertainty
 86 or diversity in a distribution, $H(z_i) = -\sum_{z \in \mathcal{Z}} q(z) \log_2 q(z)$.

87 (3) *Cluster Agreement ratio (CA)*. This metric measures the maximum proportion of samples from a
 88 given predicted cluster that are with the same true label:

$$\text{CA} = \mathbb{E}_{p \in \mathcal{Y}_{\text{seen}}} \frac{1}{|\mathcal{C}_p^{\text{test}}|} \max_{c \in \mathcal{Y}_{\text{seen}}^{\text{GT}}} \left(\sum_{(x,y) \in \mathcal{C}_p^{\text{test}}} \mathbf{1}(y = c) \right). \quad (5)$$

89 (4) *Cluster Entropy (CE)*. This metric measures the average entropy of the samples that predicted the
 90 true class contained in clusters:

$$\text{CE} = \mathbb{E}_{p \in \mathcal{Y}_{\text{seen}}} H(\{y | (x, y) \in \mathcal{C}_p^{\text{test}}\}). \quad (6)$$

91 C.3 Clustering metrics

92 Traditional novel class discovery (NCD) methods typically rely on post-cluster evaluation, where
 93 the quality of the discovered clusters is assessed after the entire test set has been processed. To
 94 ensure a comprehensive comparison with existing approaches, we also report several widely used
 95 clustering evaluation metrics, including Hungarian Cluster Accuracy (HCA) [10], Adjusted Rand
 96 Index (ARI) [11], Normalized Mutual Information (NMI) [9], and V-Measure [12]. Note that these
 97 metrics are only evaluated after TTD, say post evaluation.

98 (1) *Hungarian Cluster Accuracy (HCA)*. This metric measures the clustering accuracy by computing
 99 an optimal one-to-one mapping between predicted clusters and ground-truth labels using the Hungar-
 100 ian algorithm. It provides an intuitive evaluation of how well the discovered clusters align with the
 101 actual class distributions. HCA can be computed as

$$\text{HCA} = \mathbb{E}_{(x,y) \in \mathcal{D}^{\text{test}}} (y = \text{map}(\hat{y}(x))), \quad (7)$$

102 where $\text{map}(\cdot)$ is the optimal mapping from clustering to true labels obtained based on the Hungarian
 103 algorithm

104 (2) *Adjusted Rand Index (ARI)*. ARI quantifies the similarity between the predicted clustering
 105 assignments and the ground-truth labels while adjusting for chance. It accounts for both correct
 106 pairwise clustering and misclustered pairs, offering a robust measure of clustering consistency.

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{\max(\text{RI}) - \mathbb{E}[\text{RI}]}, \quad (8)$$

107 where Rand Index (RI) = $\frac{a+b}{C_n^2}$, a is the logarithm of samples of the same class assigned to the same
 108 cluster, and b is the logarithm of samples of different classes assigned to different clusters. n is the
 109 total number of samples, combination $C_n^2 = \frac{n(n-1)}{2}$ and $\mathbb{E}[\text{RI}]$ is the expected value of RI.

110 (3) *Normalized Mutual Information (NMI)*. NMI assesses the mutual dependence between predicted
 111 and true labels by measuring the shared information between the two distributions. A higher NMI
 112 value indicates better alignment between the discovered clusters and the actual categories. The value
 113 interval of NMI is $[0,1]$, and a larger value indicates a higher degree of information sharing between
 114 the clustering results and the real labels.

$$\text{NMI}(\mathcal{U}, \mathcal{V}) = \frac{2 \cdot I(\mathcal{U}; \mathcal{V})}{H(\mathcal{U}) + H(\mathcal{V})}, \quad (9)$$

where \mathcal{U} is collection of true labels and \mathcal{V} is collection of predictions. $I(\mathcal{U}; \mathcal{V})$ is Mutual Information where $I(\mathcal{U}; \mathcal{V}) = H(\mathcal{U}) - H(\mathcal{U}|\mathcal{V})$. $H(\mathcal{U})$ is the entropy of true label, $H(\mathcal{U}) = -\sum_{c=1}^C p(c) \log p(c)$, and $H(\mathcal{V})$ is the entropy of prediction, $H(\mathcal{V}) = -\sum_{k=1}^K p(k) \log p(k)$.

(4) *V-Measure (VM)*. The VM is taken in the interval $[0,1]$, which simultaneously constrains the purity and coverage of the clusters through the harmonic mean. Both VM and NMI are symmetric metrics that support the comparison of clusters and categories at different scales.

$$\text{V-Measure} = \frac{2 \cdot h \cdot c}{h + c}, \quad (10)$$

where homogeneity $h = 1 - \frac{H(\mathcal{U}|\mathcal{V})}{H(\mathcal{U})}$, and completeness $c = 1 - \frac{H(\mathcal{V}|\mathcal{U})}{H(\mathcal{V})}$. $H(\mathcal{U}|\mathcal{V}) = -\sum_{k=1}^K \sum_{t=1}^T p(k, t) \log \frac{p(k, t)}{p(k)}$ and $H(\mathcal{V}|\mathcal{U}) = -\sum_{t=1}^T \sum_{k=1}^K p(t, k) \log \frac{p(t, k)}{p(t)}$, where $p(t) = \frac{N_t}{N}$ is the sample proportion of class t , $p(k) = \frac{N_k}{N}$ is the sample proportion of cluster k , and $p(t, k) = \frac{\text{count}_k(t)}{N}$ is the joint distribution probability.

D More TTD Comparisons

In this section, we provide a detailed comprehensive comparison of our method with several approaches on three benchmark datasets: CIFAR100D, CUB-200D, and Tiny-ImageNetD. The evaluation includes both real-time and post evaluations, where real-time evaluation reflects the accumulated performance across all test batches, and post evaluation reassesses all test samples after training the DAA, updating the memory, and prototypes. The results are summarized in Table 3.

D.1 Comparisons on TA & CA

TA and CA are critical metrics for evaluating the performance of test-time discovery methods. TA measures the overall accuracy of the model in predicting the correct class labels, while CA evaluates the model’s ability to correctly classify samples within each class. A balanced performance in both TA and CA indicates that the model not only achieves high overall accuracy but also maintains meaningful and distinct class structures. Analysis is given in the main text.

D.2 Comparisons on TE & CE

TE and CE are complementary metrics to TA and CA, respectively. Lower values of TE and CE indicate better performance. However, Our method is not training-free, which means it updates the model’s representations during test time. This adaptive updating mechanism allows our method to refine its predictions and improve performance. As a result, our method has higher TE and CE values compared to training-free methods like HM.

And the results demonstrate that our method achieves a balanced and maximized performance in both TA and CA, while also maintaining relatively low TE and CE values. This is attributed to our DAA and STMR mechanism, which allows the model to refine its representations in response to new classes without suffering from catastrophic forgetting. Unlike training-free methods that struggle to learn from new classes, our method leverages the benefits of continuous adaptation to improve performance. Compared to other model-training methods, our method shows superior robustness and adaptability, making it a more effective solution for test-time discovery tasks.

E Different discoverable class numbers

In our experiments, we set an upper limit on the number of discoverable classes to investigate its impact on the performance of test-time discovery (TTD). This is a crucial parameter, as real-world scenarios may involve a much larger or even infinite number of potential new classes. The results are summarized in Table 4, where we compare the performance of our method with the HM method under different settings. We find that increasing the number of discoverable classes generally improves TA and CA, but the effect depends on the number of known classes. For example, when the number of known classes is fixed at 70, increasing the number of discoverable unknown classes from 30 to 100 and then to 200, both TA and CA improve significantly. Specifically, for our method, TA

Table 3: More TTD comparisons on CIFAR100D, CUB-200D, and Tiny-ImageNetD (*Tiny-IND in table*). Real-time evaluation reflects the accumulated performance across all test batches, while post evaluation reassesses all test samples after training the DAA, updating the memory and prototypes. (**Bold** data is the best performance and Underline data is the second-best performance.)

Method	Real-time Evaluation					Post Evaluation						
	KA \uparrow	TA \uparrow	TE \downarrow	CA \uparrow	CE \downarrow	KA \uparrow	TA \uparrow	TE \downarrow	CA \uparrow	CE \downarrow	KF \downarrow	
CIFAR100D	Threshold	76.46 \pm 0.98	17.21 \pm 1.33	0.52 \pm 0.04	36.91 \pm 3.26	2.07 \pm 0.41	76.62 \pm 1.85	34.60 \pm 2.02	1.10 \pm 0.04	18.70 \pm 1.24	1.42 \pm 0.05	6.45 \pm 1.78
	L2P [17]	59.93 \pm 2.15	8.57 \pm 2.49	0.60 \pm 0.06	43.10 \pm 4.30	1.85 \pm 0.18	50.53 \pm 7.25	9.60 \pm 1.50	0.77 \pm 0.12	27.39 \pm 2.11	1.37 \pm 0.24	27.85 \pm 7.21
	DP [16]	66.09 \pm 1.01	8.80 \pm 1.69	0.53 \pm 0.08	48.34 \pm 6.78	1.63 \pm 0.30	56.19 \pm 2.00	8.68 \pm 2.06	0.70 \pm 0.08	28.93 \pm 2.25	1.34 \pm 0.05	29.06 \pm 1.99
	GMP [2]	72.77 \pm 1.20	7.37 \pm 0.88	0.51 \pm 0.05	42.26 \pm 4.54	1.80 \pm 0.24	67.21 \pm 2.53	13.04 \pm 2.59	1.18 \pm 0.06	27.10 \pm 1.77	1.55 \pm 0.08	17.69 \pm 2.53
	PHE [18]	68.18 \pm 1.12	16.63 \pm 1.08	0.81 \pm 0.03	35.43 \pm 1.31	1.96 \pm 0.10	68.20 \pm 1.07	12.17 \pm 1.01	1.06 \pm 0.03	21.56 \pm 1.35	1.68 \pm 0.06	2.15 \pm 0.92
	HM [8]	79.17 \pm 0.13	21.13 \pm 0.62	0.67 \pm 0.02	56.37 \pm 1.42	1.23 \pm 0.08	80.73 \pm 1.59	31.03 \pm 1.24	1.07 \pm 0.02	34.81 \pm 1.22	1.50 \pm 0.02	3.41 \pm 1.49
	Ours	80.81 \pm 0.44	31.94 \pm 0.98	0.76 \pm 0.02	45.38 \pm 0.88	1.53 \pm 0.06	80.27 \pm 0.60	35.63 \pm 1.11	1.45 \pm 0.06	37.31 \pm 1.20	1.55 \pm 0.09	4.10 \pm 0.60
CUB200D	Threshold	66.09 \pm 1.20	43.60 \pm 2.08	0.40 \pm 0.06	44.05 \pm 4.96	1.46 \pm 0.40	65.52 \pm 3.68	49.90 \pm 1.33	0.81 \pm 0.00	6.64 \pm 0.67	0.70 \pm 0.00	1.61 \pm 3.55
	L2P [17]	46.22 \pm 1.53	9.01 \pm 0.87	0.44 \pm 0.02	55.37 \pm 7.79	0.97 \pm 0.25	31.97 \pm 3.35	4.75 \pm 0.73	0.51 \pm 0.03	24.48 \pm 1.65	0.62 \pm 0.06	42.29 \pm 3.14
	DP [16]	53.69 \pm 1.24	43.68 \pm 2.20	0.40 \pm 0.06	45.68 \pm 5.88	1.50 \pm 0.36	63.37 \pm 3.27	45.94 \pm 0.91	1.69 \pm 0.02	7.92 \pm 1.10	1.10 \pm 0.03	5.85 \pm 3.11
	GMP [2]	62.97 \pm 1.33	46.44 \pm 1.87	0.59 \pm 0.03	47.99 \pm 4.34	1.49 \pm 0.13	58.11 \pm 3.00	48.02 \pm 1.20	1.53 \pm 0.01	10.31 \pm 1.45	0.90 \pm 0.00	5.46 \pm 2.77
	PHE [18]	44.66 \pm 1.03	28.64 \pm 1.43	0.63 \pm 0.02	59.17 \pm 2.88	0.93 \pm 0.03	44.63 \pm 0.95	13.49 \pm 0.43	1.28 \pm 0.04	18.73 \pm 1.39	1.56 \pm 0.12	3.96 \pm 1.11
	HM [8]	66.20 \pm 0.55	58.30 \pm 2.37	0.35 \pm 0.02	43.33 \pm 6.10	1.92 \pm 0.83	64.42 \pm 0.65	65.28 \pm 1.78	1.02 \pm 0.03	37.25 \pm 4.90	1.24 \pm 0.22	4.07 \pm 0.47
	Ours	68.09 \pm 0.33	54.49 \pm 1.03	0.39 \pm 0.01	58.34 \pm 1.35	1.08 \pm 0.05	66.26 \pm 0.39	64.13 \pm 2.23	1.12 \pm 0.08	44.65 \pm 0.88	1.29 \pm 0.13	3.60 \pm 0.39
Tiny-IND	Threshold	57.53 \pm 1.80	11.35 \pm 1.56	0.48 \pm 0.03	63.31 \pm 3.55	0.66 \pm 0.12	52.36 \pm 3.10	6.48 \pm 1.40	0.37 \pm 0.02	13.81 \pm 2.11	0.44 \pm 0.02	22.90 \pm 3.08
	L2P [17]	46.25 \pm 1.41	7.79 \pm 2.92	0.51 \pm 0.03	53.55 \pm 6.47	1.33 \pm 0.23	29.50 \pm 3.77	10.38 \pm 2.42	0.89 \pm 0.06	23.28 \pm 0.79	1.37 \pm 0.06	47.97 \pm 3.77
	DP [16]	46.51 \pm 0.58	6.41 \pm 0.93	0.51 \pm 0.03	58.27 \pm 6.10	1.15 \pm 0.21	28.53 \pm 3.33	9.63 \pm 2.10	0.85 \pm 0.02	26.80 \pm 1.38	1.33 \pm 0.02	47.57 \pm 3.32
	GMP [2]	62.47 \pm 1.40	6.25 \pm 1.72	0.45 \pm 0.02	58.02 \pm 4.29	1.08 \pm 0.14	63.95 \pm 2.04	15.64 \pm 2.63	1.30 \pm 0.03	26.31 \pm 2.33	1.54 \pm 0.03	16.86 \pm 2.04
	PHE [18]	58.39 \pm 1.29	13.84 \pm 0.90	0.48 \pm 0.02	71.45 \pm 3.80	0.40 \pm 0.03	58.39 \pm 1.14	12.10 \pm 1.05	0.70 \pm 0.02	18.64 \pm 1.42	1.24 \pm 0.01	3.47 \pm 1.32
	HM [8]	75.31 \pm 1.31	16.04 \pm 0.76	0.51 \pm 0.00	73.81 \pm 2.67	0.61 \pm 0.04	74.94 \pm 2.20	16.23 \pm 1.24	0.81 \pm 0.00	37.43 \pm 1.30	1.21 \pm 0.02	1.15 \pm 2.18
	Ours	76.38 \pm 0.82	24.10 \pm 0.80	0.54 \pm 0.00	71.70 \pm 2.02	0.73 \pm 0.05	75.50 \pm 1.96	23.17 \pm 1.35	1.14 \pm 0.01	36.41 \pm 1.12	1.44 \pm 0.01	2.39 \pm 1.56

increases from 32.40 to 36.21 and then to 41.84, while CA increases from 46.02 to 48.13 and then to 30.55 (note that CA drops slightly when the number of unknown classes becomes very large). Increasing the number of discoverable classes generally leads to higher KF values, indicating more severe forgetting. For example, when the number of discoverable classes increases from 30 to 200, the KF value for our method increases from 3.54 to 8.27.

Compared to the HM method, our method shows more balanced performance across different settings. For instance, when the number of discoverable classes is set to 100, our method achieves a TA of 36.21 and a CA of 48.13, which are significantly more balanced than those of HM (TA: 17.58, CA: 81.28). This indicates that our method is more effective in balancing the discovery of new classes and the recognition of known classes. However, when the number of discoverable classes far exceeds the true number, both methods suffer from performance degradation. For example, in the case of 70 known classes and an infinite number of discoverable unknown classes, our method achieves a TA of 31.78 and a CA of 89.58, while HM achieves a TA of 20.37 and a CA of 92.94. This suggests that while our method is more robust in general, both methods struggle when the number of discoverable classes becomes excessively large.

The results highlight the importance of appropriately setting the number of discoverable classes for optimal TTD performance. While more discoverable classes generally improve the model’s ability to recognize new patterns, they also introduce more complexity and risk of forgetting. Our method shows a more balanced performance across different settings, achieving higher TA and CA values while maintaining reasonable TE, CE, and KF values. This demonstrates the effectiveness of our approach in balancing adaptability and stability during test-time discovery.

F T-SNE visualization of the effect of DAA.

To provide a more intuitive understanding of how our method affects the feature space, we conducted T-SNE [13] visualizations of the feature embeddings before and after applying our DAA (Dynamic Adaptation and Augmentation) mechanism. The results are shown in Fig. 1.

In the pre-training phase, our method attempts to disrupt the feature representations of unknown classes to some extent. This is evident from the visualization in Fig. 1b, where the feature embeddings of unknown classes are more scattered and less well-separated from known classes. This disruption is intentional, as it helps the model to avoid overfitting to the initial feature space and encourages it to adapt more flexibly during the test phase. After test-time training with DAA, the boundaries between known and unknown classes become clearer again, as shown in Fig. 1c. This indicates that our method effectively refines the feature space during test-time training, allowing the model to better

Table 4: Comparisons of different discoverable class numbers.

Known +		Real-time Eval				Post Eval				
Unknown		TA	TE	CA	CE	TA	TE	CA	CE	KF
HM	70+30	21.11	0.66	56.87	1.27	31.03	1.07	34.81	1.50	3.47
	70+100	17.58	0.70	81.28	0.44	25.03	1.82	40.74	1.05	6.46
	70+200	19.86	0.76	85.60	0.33	26.87	2.17	42.63	0.84	7.79
	70+ ∞	20.37	0.84	92.94	0.16	22.63	2.86	47.09	0.46	10.69
	70+Human	52.10	0.48	42.96	1.68	48.27	1.19	49.44	1.24	5.81
Ours	70+30	32.40	0.74	46.02	1.52	35.97	1.45	37.55	1.53	3.54
	70+100	36.21	0.87	48.13	0.69	32.56	2.05	31.44	1.22	6.73
	70+200	41.84	0.90	30.55	0.67	31.00	2.22	27.96	0.93	8.27
	70+ ∞	31.78	0.98	89.58	0.35	22.35	2.99	49.32	0.55	11.07
	70+Human	75.22	0.35	30.70	1.97	49.30	1.35	50.61	1.36	2.68

distinguish between known and novel classes. In contrast, baseline methods that fix the model and do not change throughout the entire test phase (as shown in Fig. 1a) struggle to adapt to new classes, resulting in less clear boundaries and poorer performance.

This visualization demonstrates the effectiveness of our DAA mechanism in dynamically adapting the feature space during test-time training, leading to improved performance in recognizing both known and novel classes.

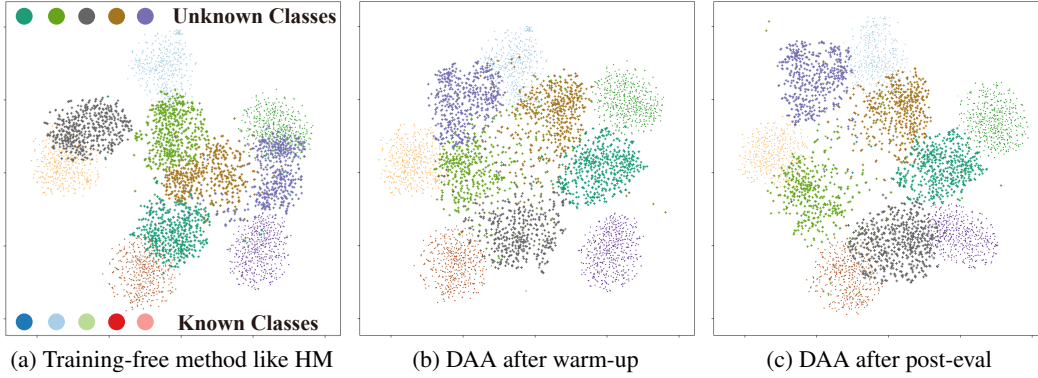


Figure 1: Comparison of unknown class label matching between predictions and ground truth across different methods (Cifar-100D 90+10)

G Hyper-parameter analysis

G.1 Frequency of STMR.

The frequency of STMR plays a crucial role in balancing the trade-off between model performance and computational efficiency. As shown in Fig. 2, we conducted experiments to investigate the impact of varying the frequency of STMR on the overall performance of our method. The results indicate that as the frequency of STMR decreases, both the TA and CA deteriorate significantly. This suggests that a lower frequency of STMR leads to insufficient updates and refinements of the model during the test phase, resulting in poorer recognition performance for both known and novel classes.

On the other hand, while increasing the frequency of STMR can improve the model’s ability to adapt to new data, it also leads to a substantial increase in computational overhead. Frequent STMR operations require more time and resources for processing each batch, which can be impractical for real-time or resource-constrained applications. Moreover, the improvement in performance brought by each additional STMR operation diminishes as the frequency increases, indicating that there is a point of diminishing returns.

Therefore, selecting an optimal frequency for STMR is essential to achieve a balance between performance and efficiency. A moderate frequency ensures that the model can effectively leverage STMR for continuous adaptation while avoiding excessive computational costs.

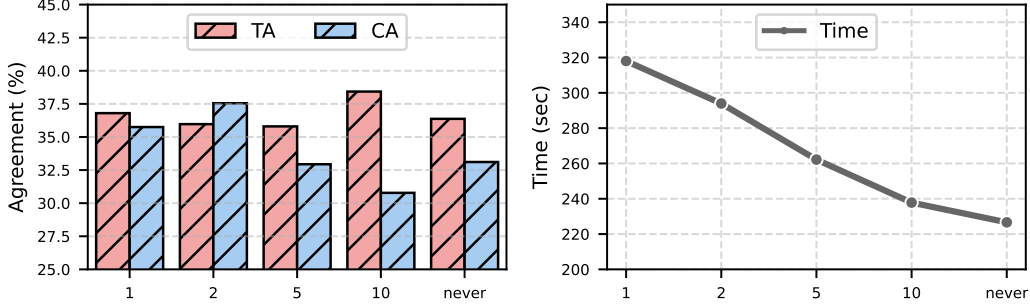


Figure 2: Trends of TA, CA and KF with different replay samples.

214 G.2 Different replay samples.

215 The number of replay samples used in the memory replay mechanism is another critical hyper-
 216 parameter that affects the performance of our method. Fig. 3 shows that the number of replay
 217 samples has a significant influence on the model’s ability to retain knowledge from previous classes
 218 while adapting to new ones. When no replay samples are used, the model exhibits the highest level
 219 of knowledge forgetting, as indicated by the largest KF value. This suggests that without replay,
 220 the model is more prone to catastrophic forgetting, where it quickly forgets previously learned
 221 information as it adapts to new data.

222 As the number of replay samples increases, the KF value decreases, indicating that the model is
 223 better able to retain knowledge from previous classes. However, this improvement comes at the
 224 cost of increased computational time, as more replay samples require additional processing during
 225 each batch. Moreover, while a moderate number of replay samples can help stabilize the model’s
 226 performance, an excessive number of replay samples can lead to diminishing returns in terms of
 227 performance gains, while further increasing the computational burden.

228 Thus, choosing an appropriate number of replay samples is crucial for achieving a balance between
 229 knowledge retention and computational efficiency. A well-tuned number of replay samples ensures
 230 that the model can effectively leverage memory replay to mitigate catastrophic forgetting while
 231 maintaining reasonable processing times.

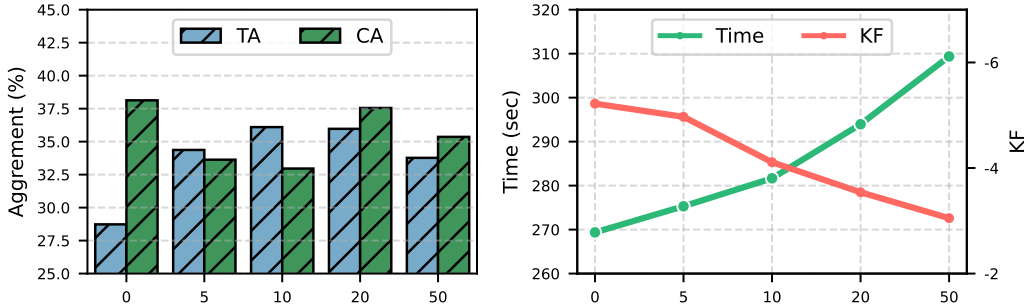


Figure 3: Trends of TA, CA and KF with different STMR frequency.

232 G.3 λ_1 and λ_2 during TTT

233 During test-time training (TTT), the overall testing phase loss for the DAA is shown below:

$$\mathcal{L}_{\text{test}}(\mathcal{B}) = \lambda_1 \cdot \mathbb{E}_{c \in \mathcal{Y}_{\text{kn}}} \left(\mathcal{L}_{\text{kn}}(\mathcal{B}, c) + \mathcal{L}_{\text{kn}}^{\text{replay}}(c) \right) + \lambda_2 \cdot \mathbb{E}_{c \in \mathcal{Y}_{\text{seen}}} \left(\mathcal{L}_{\text{un}}(\mathcal{B}, c) + \mathcal{L}_{\text{un}}^{\text{renewal}}(c) \right), \quad (11)$$

234 where λ_1 and λ_2 are hyperparameters that balance the contributions of each loss term.

235 We explored the impact of different ratios between λ_1 and λ_2 on the overall performance of our
 236 method. The results are presented in Table 5. The MSE loss is used to refine the model’s predictions
 237 for known classes, while the contrastive loss encourages the model to separate the feature embeddings

of known and unknown classes. We conducted experiments with different ratios of MSE loss to contrastive loss to determine the optimal balance between these two objectives.

When the model places a much higher emphasis on refining the predictions for known classes. This results in a relatively high TA and CA. However, the model’s ability to distinguish between known and unknown classes is somewhat limited, as indicated by the higher CE value. As the ratio decreases, the model starts to pay more attention to the contrastive loss, which helps improve the separation between known and unknown classes. This leads to a slight increase in CA and a decrease in CE, indicating better class separation. However, the overall TA and TE values are affected, suggesting that the model’s ability to accurately classify known classes is slightly compromised. When the ratio further decreases, the model crashes, as indicated by the extremely high TA and TE values and the near-zero CA and CE values. This suggests that an excessive emphasis on the contrastive loss can destabilize the model, leading to poor performance.

These results highlight the importance of carefully balancing the MSE loss and contrastive loss during TTT. An optimal ratio ensures that the model can effectively refine its predictions for known classes while also maintaining clear boundaries between known and unknown classes. This balance is crucial for achieving high accuracy and robustness in recognizing both known and novel classes.

Table 5: Comparisons of ratio between λ_1 and λ_2 during TTT.

$\lambda_1 : \lambda_2$	Real-time Eval				Post Eval				
	TA	TE	CA	CE	TA	TE	CA	CE	KF
1000:0.1	28.41	0.80	44.91	1.56	29.50	1.49	30.53	1.74	0.92
1000:0.5	29.40	0.82	45.80	1.55	33.33	1.53	34.66	1.73	1.23
1000:1	33.66	0.66	49.81	1.39	34.8	1.03	37.23	1.48	2.64
1000:2	32.40	0.74	46.02	1.52	35.97	1.45	37.55	1.53	3.54
1000:5(crashed)	78.22	0.20	19.00	0.38	100	0.00	1.00	1.99	—
500 : 1	30.20	0.84	42.64	1.64	33.30	1.48	34.37	1.71	3.76
500 : 2	34.98	0.64	49.68	1.40	37.60	1.22	32.33	1.53	4.48
500 : 3(crashed)	81.59	0.35	15.37	0.62	100	0.00	1.20	2.20	—

G.4 Comparison of different threshold Gamma

During testing phase, we base on the max similarity scores, and apply a confidence threshold γ to determine whether a test sample x belongs to a seen class or an unseen class.

$$\hat{y} = \begin{cases} \arg \max_{c \in (\mathcal{Y}_{\text{kn}} \cup \mathcal{Y}_{\text{seen}})} (P(x)), & \text{if } \max_c P(x) > \gamma, \\ \text{new unseen class}, & \text{otherwise,} \end{cases} \quad (12)$$

where prediction comparison $P(x) = \text{sim}(\text{DAA}[\mathbf{r}(x)], \mathbf{p}_c)$.

The confidence threshold γ plays a crucial role in balancing the trade-off between recognizing known classes and discovering new classes. We conducted experiments with different γ values to analyze their impact on the performance metrics, as shown in Table 6.

When γ is set to 0.5, the model is more likely to classify samples as belonging to known classes, leading to a relatively high TA of 11.85 and a CA of 44.36. However, this setting also results in a high TE of 0.83 and a CE of 1.61, indicating that many samples are misclassified as known classes when they actually belong to new classes. As γ increases to 0.7, the model becomes more selective in classifying samples as known classes, resulting in a higher TA of 32.40 and a CA of 46.02. This setting achieves a better balance between recognizing known classes and discovering new classes, with a TE of 0.74 and a CE of 1.52. This indicates that the model is more accurate in classifying known classes while also effectively identifying new classes. When γ is set to 0.9, the model becomes very conservative in classifying samples as known classes, leading to a significant drop in TA to 12.75 and a CA of 46.00. While this setting reduces TE to 0.81 and CE to 1.53, it also results in a higher KF of 10.78, indicating that the model is more prone to forgetting previously learned classes.

The choice of γ also significantly impacts the error rates. Lower γ values generally lead to higher TE and CE, as the model is more likely to misclassify new classes as known classes. Conversely, higher γ values reduce TE and CE but at the cost of lower TA and higher KF. For example, at $\gamma = 0.7$, the model achieves a good balance with a TE of 0.74 and a CE of 1.52, indicating that it effectively

distinguishes between known and new classes. And we find that higher γ values generally lead to higher KF, as the model becomes more conservative in classifying samples as known classes, leading to more forgetting. For instance, at $\gamma = 0.9$, the KF value is 10.78, indicating significant forgetting of previously learned classes.

Table 6: Comparisons of different γ during Testing phase.

γ	Real-time Eval				Post Eval				
	TA	TE	CA	CE	TA	TE	CA	CE	KF
0.5	11.85	0.83	44.36	1.61	30.03	1.39	37.14	1.67	5.89
0.6	20.04	0.81	48.07	1.56	29.83	1.46	34.25	1.78	4.43
0.7	32.40	0.74	46.02	1.52	35.97	1.45	37.55	1.53	3.54
0.8	29.90	0.84	42.78	1.65	29.60	1.55	28.65	1.82	7.96
0.9	12.75	0.81	46.00	1.53	31.10	1.73	27.64	1.72	10.78

References

- [1] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8344–8353, 2022.
- [2] Fernando Julio Cendra, Bingchen Zhao, and Kai Han. Promptccd: Learning gaussian mixture prompt pool for continual category discovery. In *European Conference on Computer Vision*, pages 188–205. Springer, 2025.
- [3] Ruoyi Du, Dongliang Chang, Kongming Liang, Timothy Hospedales, Yi-Zhe Song, and Zhanyu Ma. On-the-fly category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11691–11700, 2023.
- [4] Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8401–8409, 2019.
- [5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- [6] Yann Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.
- [7] Jiashuo Liu, Zheyang Shen, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.
- [8] Fan Lyu, Tianle Liu, Zhang Zhang, Fuyuan Hu, and Liang Wang. Test-time discovery via hashing memory. *arXiv preprint arXiv:2503.10699*, 2025.
- [9] Aaron F McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms. *arXiv preprint arXiv:1110.2515*, 2011.
- [10] Marina Meilă. Comparing clusterings by the variation of information. In *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*, pages 173–187. Springer, 2003.
- [11] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
- [12] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420, 2007.
- [13] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.
- [14] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized category discovery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7492–7501, 2022.
- [15] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

- 315 [16] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong
316 Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual
317 learning. In *European Conference on Computer Vision*, pages 631–648. Springer, 2022.
- 318 [17] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot,
319 Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF*
320 *Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022.
- 321 [18] Haiyang Zheng, Nan Pu, Wenjing Li, Nicu Sebe, and Zhun Zhong. Prototypical hash encoding for
322 on-the-fly fine-grained category discovery. *arXiv preprint arXiv:2410.19213*, 2024.