# RELATIONAL SEMANTIC GROUNDING FOR TEST-TIME DISCOVERY

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Test-Time Discovery (TTD) aims to recognize known classes and detect novel ones from a continuous unlabeled test stream after deployment. Existing TTD methods mainly rely on visual clustering or prototype comparison, and can only assign anonymous labels to discovered classes, leaving them difficult to interpret in open-world applications. In this paper, we propose SEGA, a Semantic Grounding with Attributes framework for TTD. SEGA leverages large language models to generate attribute-rich descriptions for known classes and constructs an attribute-anchored visual-semantic space during training. At test time, we introduce Relational Semantic Grounding (RSG), which grounds unknown samples with similar and dissimilar known-class anchors, prompts the LLM to generate candidate concepts, and reranks them in the learned semantic space. The selected concept is further used for confidence-guided test-time semantic stabilization. By coupling attribute-level semantic alignment with relational concept grounding, SEGA transforms anonymous test-time clusters into interpretable semantic concepts. Experiments show that SEGA improves TTD baselines with the promise of semantic grounding for open-world. Our code will be available.

## 1 INTRODUCTION

Novel Class Discovery (NCD) (Pu et al., 2023; Wu et al., 2023) aims to discover unseen categories by transferring knowledge from labeled known classes to an unlabeled data pool. However, existing NCD methods assume that the unlabeled data are available in advance, making discovery an offline process. This assumption becomes restrictive in open-world deployment, where new categories may continuously emerge in applications. Test-Time Discovery (TTD) (Lyu et al., 2025) addresses this setting by requiring the model to recognize known classes, discover novel ones, and update its category space from a continual unlabeled test stream during inference. In traditional NCD, unknown classes are discovered offline and represented by semantic-agnostic cluster IDs (such as "#1"), making immediate semantic interpretation less necessary. TTD, however, discovers novel classes during deployment, where the discovered classes should be understandable and reusable for humans or subsequent inference rather than merely separated as new clusters. Existing TTD methods (Lyu et al., 2025; Liu et al., 2025) largely inherit the NCD-style treatment, assigning semantic-agnostic IDs to unknown classes rather than grounding them as meaningful concepts, which limits their practical value (Fig. 1a). Some semantic-guided NCD methods (Wang et al., 2024; Zheng et al., 2024b) introduce textual information through open-vocabulary recognition or fixed lexicon retrieval (Fig. 1b). However, these strategies assume that candidate novel categories can be covered by a predefined vocabulary, which is misaligned with open-world TTD where possible novel classes are unknown before deployment. A simple alternative is to use vision-language models with class-name prompts such as "a photo of a [class]", but such prompts provide only coarse semantic anchors and lack fine-grained attributes for relating unknown classes to known ones. LLMs, with their rich conceptual and attribute-level knowledge, offer a more flexible source of semantic cues for open-ended discovery (Du et al., 2024; Fu et al., 2025). This suggests that attribute-rich descriptions generated by LLMs can provide a semantic basis for TTD, allowing known-class attributes to serve as transferable bridges for grounding novel classes at test time.

In this paper, we propose **Semantic Grounding with Attributes (SEGA)**, a framework that transforms TTD from anonymous clustering into semantic concept generation (Fig. 1c). During training, SEGA first uses LLM-generated attribute descriptions to enrich known classes and learns an attribute-



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

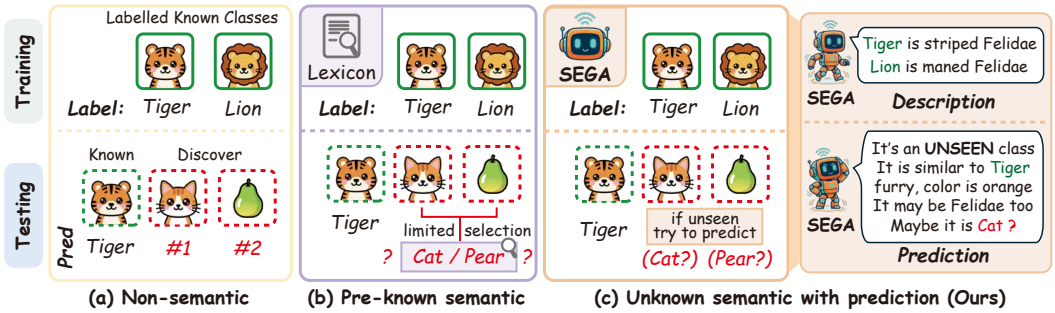



Figure 1: Comparison among different discovery method. (a) Non-semantic discovery, where unknown samples are detected or clustered online but are only assigned anonymous placeholder labels. (b) Pre-known semantic discovery handles unknown sample by matching visual features to a predefined category vocabulary, but it is unrealistic and its discovery ability is limited. (c) Our method aims to more meaningful and open-ended test-time discovery where unknown classes are grounded with visual and semantic relations to known classes, and interpreted through concept generation.

aware visual-semantic space through cross-modal alignment. At test time, when an unknown-class candidate is detected, we introduce Relational Semantic Grounding (RSG) to describe it by its relations to known-class anchors and visual attributes, and then query an LLM to generate candidate category names. Since free-form LLM generation can produce overly broad or implausible names, SEGA reranks the candidates according to their plausibility in the learned semantic space. The selected concept is further converted into a pseudo description for confidence-guided test-time training, which stabilizes subsequent discovery and reduces the effect of noisy semantic predictions. By coupling attribute-level semantic alignment, relational concept generation, and semantic stabilization, SEGA provides an interpretable TTD pipeline that grounds discovered unknown classes as meaningful concepts rather than anonymous tokens. Experiments on four TTD benchmarks show that SEGA improves discovery performance over recent TTD baselines, with up to 4.55% gain in agreement metrics. In summary, our contributions are:

(1) We identify and address the semantic grounding problem in TTD, where discovered unknown classes should be represented as meaningful concepts rather than anonymous cluster IDs, and propose SEGA as a unified framework for semantic test-time discovery.
(2) We introduce RSG, which combines LLM-generated known-class attributes, relational semantic anchors, and semantic reranking to generate reliable candidate names for unknown classes.
(3) We evaluate SEGA on four TTD benchmarks, showing improvements over prior TTD methods in discovery performance, known-class stability, and semantic quality of the generated class names.

## 2 Related Work

**From Semantics-Agnostic to Semantic-Guided Category Discovery.** (Han et al., 2020; Zhang et al., 2023; Zhong et al., 2021) Novel Class Discovery (NCD) aims to identify new categories from unlabeled data by transferring knowledge from known classes. GCD (Vaze et al., 2022) and PromptCCD (Cendra et al., 2024) replace the traditional k-means stage with contrastive learning prompting for purer prototypes. On-the-fly Category Discovery (OCD) (Du et al., 2023; Zheng et al., 2024a) pushes NCD toward real-time operation by hashing images and assigning them according to Hamming-distance. Later, some work start to explore incorporating textual information into category discovery to address the semantics gap. SNCD (Wang et al., 2024) cluster unlabeled images but also semantically match them by constructing semantic similarity matrices. Some CLIP-guided GCD works (Zheng et al., 2024b; Wang et al., 2025) leverage pre-trained vision-language alignment to guide discovery with semantic priors. And the open-vocabulary paradigm (Wu et al., 2024; Zhu & Chen, 2024) has further expanded the semantic horizon. However, these approaches operate in offline or static settings and depend on predefined or retrieved textual descriptions.

**Test-Time Discovery (TTD)** requires models to detect and learn novel classes in real-time as samples arrive sequentially. The pioneering HM (Lyu et al., 2025) proposes a training-free approach using locality-sensitive hashing for efficient sample retrieval and prototype comparison. DAA (Liu et al., 2025) introduces a lightweight trainable adapter to amplify feature-level discrepancies between known and unknown classes. While these methods achieve impressive performance in visual feature discrimination, discovered classes are assigned placeholder labels without any interpretable meaning.

**LLM for Visual Knowledge.** LLM-generated descriptions for visual recognition are widely used in different work recently (Zhang et al., 2025; Lei et al., 2025; Fu et al., 2025). Detailed visual descriptions are revealed to outperform fixed templates and can improve task like zero-shot, open-vocabulary and attribute-concept annotation. The dynamic, streaming nature of TTD with its real-time constraints and irreversible decisions remains unexplored in the context of LLM-assisted semantic discovery. Our work bridges this gap by enabling test-time LLM reasoning for semantic interpretation, transforming TTD from visual clustering to conceptual understanding.

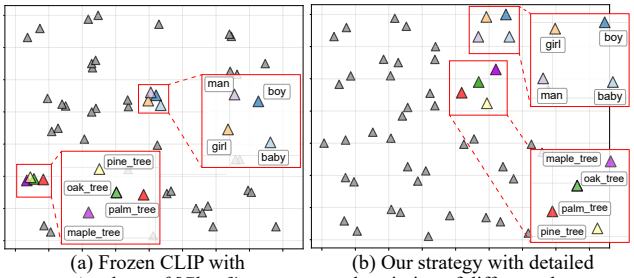
# 3 METHOD

## 3.1 TTD AND A CLIP-BASED BASELINE

TTD aims to recognize known classes and discover unknown ones in a continuous unlabeled stream after model deployment. Formally, let $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the training set, where $x_i \in \mathcal{X}$ is input image, and $y_i \in \mathcal{Y}_{\text{kn}}$ is its number label from the known class set $\mathcal{Y}_{kn}$. During testing, the model encounters a test stream $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{kn}} \cup \mathcal{D}_{\text{un}}$, where $\mathcal{D}_{\text{un}}$ may contain unknown classes that were never seen during training. TTD requires the model to satisfy two objectives simultaneously. First, *novel class discovery and learning*, where inputs $x \in \mathcal{D}_{\text{test}}^{\text{un}}$ must be identified as belonging to unseen classes and assigned new labels to distinguish them from known-class samples. Second, *unified classification*, where all inputs $x \in \mathcal{D}_{\text{test}}$ are accurately classified into either a known class $y \in \mathcal{Y}_{\text{kn}}$ or a novel class $y \in \mathcal{Y}_{\text{seen}}$, with $\mathcal{Y}_{\text{seen}}$ denoting the set of dynamically discovered novel classes.

Inspired by recent vision-language discovery methods (Zheng et al., 2024b; Wang et al., 2025), we adopt CLIP (Radford et al., 2021) as the backbone. To preserve linguistic knowledge, we freeze the pretrained text encoder and fine-tune the image encoder, with lightweight learnable projection layers attached to both modalities. The projection layers map image and text features into a shared attribute-aware space and are updated during training and test-time adaptation, enabling flexible alignment without modifying the frozen text encoder.

Standard CLIP-based methods often use class-name prompts such as *"a photo of [CLASS]"*, which provide only coarse semantic anchors. As shown in Fig. 2, such template-based text embeddings are dominated by class names and may entangle semantically related but visually distinct categories. In the following subsection, we instead enrich each known class with LLM-generated attribute descriptions, following the intuition that attributes provide transferable semantic cues across categories (Du et al., 2024; Zheng et al., 2024b). Attribute descriptions capture fine-grained cues such as shape, color, texture, and parts, helping distinguish similar classes while preserving their semantic relations.




(a) Frozen CLIP with '*a photo of [Class]*'

(b) Our strategy with detailed description of different classes

Figure 2: T-SNE visualization of the CLIP-based textual feature for different kinds of text prompt.

## 3.2 ATTRIBUTE-ANCHORED SEMANTIC SPACE CONSTRUCTION

Based on the above analysis, we first construct an attribute-anchored visual-semantic space using known-class training data to support semantic grounding at test time. For each known class $c \in \mathcal{Y}_{\text{kn}}$, we query an LLM, e.g., GPT-4o-mini, to generate an attribute-rich description $t_c$ that summarizes discriminative visual properties such as color, shape, texture, and parts[1]. This yields a set of textual anchors $\mathcal{T} = \{t_c\}_{c \in \mathcal{Y}_{\text{kn}}}$, which provide semantic references for known classes[2]. The text encoder is frozen to preserve its linguistic knowledge, while the image encoder and lightweight projection layers are trainable. Given an image $x$ and the textual anchor $t_c$ of its class, we obtain

$$\mathbf{z_v}(x) = P_v\big(E_v(x); W_v\big) \quad \mathbf{z_t}(t_c) = P_t\big(E_t(t_c); W_t\big), \tag{1}$$

where $P_v$ and $P_t$ map both modalities into a shared attribute space $\mathcal{Z}$.

[1] Such as "Describe [CLASS] in detail, including visual attributes like color, shape...".

[2] "A zebra has black and white stripes, a horse-like shape..."

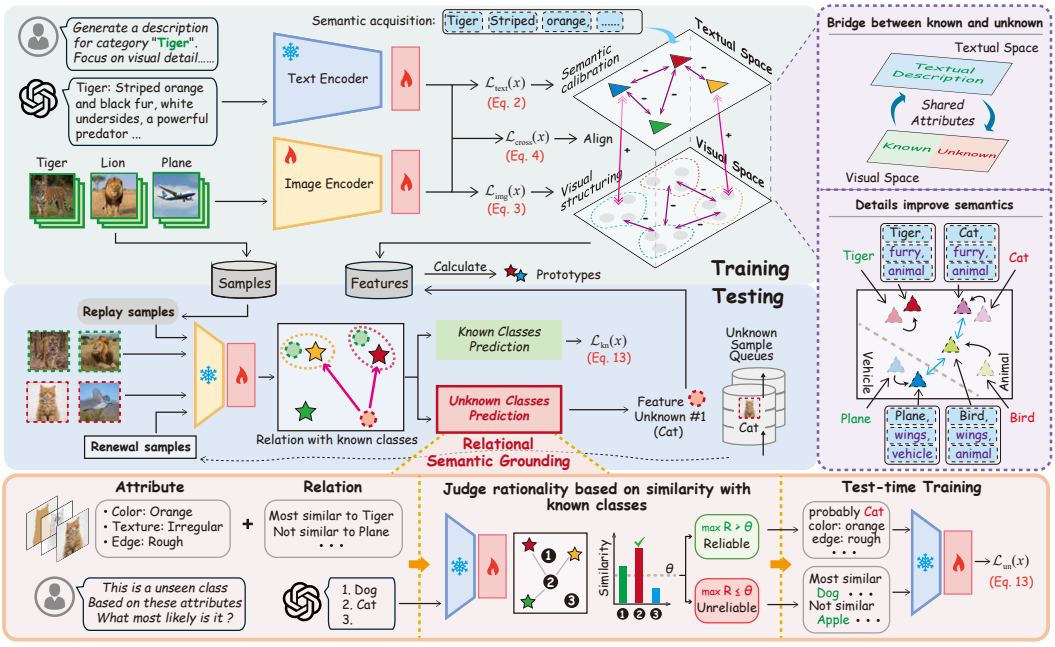



Figure 3: Method Schema. During training, LLM-generated attribute descriptions are used to construct a semantic space and guide cross-modal alignment between images and text. During testing, we discovers unknown classes, retrieves their semantic relations to known class anchors, and queries the LLM to generate candidate category names. The candidates are then grounded and filtered by the RSG module, and the selected prediction is used to guide confidence-aware test-time updates.

**Semantic anchor calibration.** The LLM-generated descriptions are not used as ordinary class labels. Actually, they act as semantic anchors $\mathbf{a}_c = \mathbf{z}_t(t_c)$ , and all known-class anchors form an anchored textual manifold $\mathcal{M}_t = \{\mathbf{a}_c \mid c \in \mathcal{Y}_{\text{kn}}\}$ that define how known classes are arranged in the attribute space. However, directly using descriptions as anchors can be unstable, since pretrained models are not inherently arranged for rich descriptions. We therefore *calibrate* these textual anchors so that descriptions of the same-class textual neighbors $\mathcal{N}_t^+(i) = \{j \in \mathcal{B} : y_j = y_i\}$ collapse into compact semantic regions, while anchors from different classes $\mathcal{N}_t^-(i) = \{j \in \mathcal{B} : y_j \neq y_i\}$ occupy separable territories. A well-calibrated anchor space therefore provides a reliable coordinate system for subsequent relational inference, rather than merely serving as a set of text classifiers.
Specifically, given a test batch $\mathcal{B}$, we ensure that the text embedding space preserves class-level discriminability with attribute-rich descriptions by

$$\mathcal{L}_{\text{text}} = -\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\log\frac{\sum_{j:y_j=y_i}\exp[\mathbf{z_t}(t_i)^\top\mathbf{z_t}(t_j)]}{\sum_{c\in\mathcal{B}}\exp[\mathbf{z_t}(t_i)^\top\mathbf{z_t}(t_c)]}. \tag{2}$$

**Visual neighborhood structuring.** While $\mathcal{M}_t$ provide the semantic coordinate system, the visual stream also needs a stable neighborhood structure before it can support discovery. Let $\mathbf{v}_i = \mathbf{z}_i(x_i)$ denote the visual feature of image $x_i$ and if visual manifolds $\mathcal{M}_v = \{\mathbf{v}_i\}$ are scattered or overly fitted to known-class decision boundaries, an unseen sample may be absorbed by a known class or isolated as noise. We therefore organize $\mathcal{M}_v$ with a dual objective by

$$\mathcal{L}_{\text{img}} = -\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\log\frac{\exp\left[\mathbf{z_i}(x_i)^\top\tilde{\mathbf{z}}_\mathbf{i}(x_i)\right]}{\sum_{c\in\mathcal{B}}\exp\left[\mathbf{z_i}(x_i)^\top\mathbf{z_i}(x_c)\right]} + \log\frac{\sum_{j:y_j=y_i}\exp\left[\mathbf{z_i}(x_i)^\top\mathbf{z_i}(x_j)\right]}{\sum_{c\in\mathcal{B}}\exp\left[\mathbf{z_i}(x_i)^\top\mathbf{z_i}(x_c)\right]}\right), \tag{3}$$

where $\tilde{\mathbf{z}}_\mathbf{i}$ denotes the feature of an augmented view of $x_i$. In this way, the visual representation is made robust to visual variations and the image encoder learns a visual manifold that is suitable for both recognizing known classes and detecting deviations caused by emerging unknown classes.

**Cross-modal coordinate binding.** The calibrated textual manifold $\mathcal{M}_t$ and the structured visual manifold $\mathcal{M}_v$ are useful only if they share the same coordinate system. To bridge this gap, we build image-anchor correspondences $\mathcal{G} = \{(\mathbf{v}_i, \mathbf{a}_{y_i})\}_{i\in\mathcal{B}}$, where each visual feature is paired with

the semantic anchor of its class. Therefore, we further introduce a cross-modal binding objective along $\mathcal{G}$ connecting each image to its corresponding attribute-rich textual anchor without which the image encoder may rely on spurious visual correlations. This binding encourages visual features to inherit the semantic directions encoded by the LLM descriptions, such as shape, texture, color, and part-level cues. Once bound, visual representations are no longer determined purely by appearance and unknown sample can then be described through its position relative to known semantic anchors because they share the same semantic coordinates in the attribute space. This binding is achieved via:

$$\mathcal{L}_{\text{cross}} = -\frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\log\frac{\exp\left[\mathbf{z_i}(x_i)^\top \mathbf{z_t}(t_i)\right]}{\sum_{c\in\mathcal{B}}\exp\left[\mathbf{z_i}(x_i)^\top \mathbf{z_t}(t_c)\right]} + \log\frac{\exp\left[\mathbf{z_t}(t_i)^\top \mathbf{z_i}(x_i)\right]}{\sum_{c\in\mathcal{B}}\exp\left[\mathbf{z_t}(t_i)^\top \mathbf{z_i}(x_c)\right]}\right). \quad (4)$$

The three objectives play complementary roles which together prepare the model for testing where known classes become stable semantic references, while unknown samples can be interpreted by their relational position in the anchored visual-semantic space. The total training loss is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{text}} + \mathcal{L}_{\text{img}} + \mathcal{L}_{\text{cross}}. \quad (5)$$

### 3.3 Relational Concept-Level Semantic Inference

When encountering the test sample $x$ with a known or unknown class, the prediction procedure involves calculating the cosine similarity $s_k = \text{sim}(\mathbf{z_i}(x), \mathbf{p}_c)$ between the image feature $\mathbf{z_i}(x)$ and the prototypes $\mathbf{p}$ of known classes to determine their classification. Based on the max similarity scores, we apply a confidence threshold $\gamma$ to determine whether a test sample $x$ belongs to a seen class or an unseen class.

$$\hat{y} = \begin{cases} \arg\max_{c\in(\mathcal{Y}_{\text{kn}}\cup\mathcal{Y}_{\text{seen}})}(s_k), & \text{if } \max_c s_k > \gamma, \\ \text{new discovered class}, & \text{otherwise.} \end{cases} \quad (6)$$

**Prompt construction and candidate generation.** If the sample is classified as an unseen class $\hat{y} \in \mathcal{Y}_{\text{unseen}}$, two sets Top-$k$ most similar classes $\mathcal{S}_{\text{sim}}$ and Top-$k$ least similar classes $\mathcal{S}_{\text{dis}}$ are extracted. These similarity patterns serve as relational anchors, characterizing the unknown class by what it resembles and what it differs from in the known semantic space.

$$\mathcal{S}_{\text{sim}} = \{(c, s_k) \mid c \in \text{top-}k \text{ of } s_k\} \qquad \mathcal{S}_{\text{dis}} = \{(c, s_k) \mid c \in \text{bottom-}k \text{ of } s_k\} \quad (7)$$

Relational similarity alone may be insufficient for fine-grained naming. To provide additional discriminative cues, we extract interpretable visual attributes $\mathcal{A}$ from the image $x$, which include color statistics (dominant hue, saturation, temperature), texture properties (roughness, regularity), lighting conditions (brightness, uniformity), and structural characteristics (edge density, composition). These attributes are computed using only low-level operations on the raw images.

$$\mathcal{A} = \Phi_{\text{attr}}(\mathbf{x}) = \{\mathcal{A}_{\text{color}}, \mathcal{A}_{\text{texture}}, \mathcal{A}_{\text{lighting}}, \mathcal{A}_{\text{structure}}\} \quad (8)$$

Then we construct a prompt $\mathcal{P} = (\mathcal{S}_{\text{sim}}, \mathcal{S}_{\text{dis}}, \mathcal{A})$ that integrates both relational and attribute information to instruct the LLM to generate $M$ candidate category names for the discovered class:

$$\mathcal{C}_{\text{LLM}} = \{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_M\} = \text{LLM}(\mathcal{P}(\mathcal{S}_{\text{sim}}, \mathcal{S}_{\text{dis}}, \mathcal{A})) \quad (9)$$

**Semantic reranking with text encoder.** Some of these candidates are plausible but potentially noisy. To ground the LLM's free-form generation in the actual visual-semantic space, we propose a semantic reranking mechanism. We encode both the LLM-generated candidates and the top-$k$ most similar known classes with our text encoder. And the relevance score $R$ for each candidate $\hat{c}$ is computed as the average similarity with the known reference classes.

$$R(\hat{c}) = \frac{1}{|\mathcal{S}_{\text{sim}}|}\sum_{c\in\mathcal{S}_{\text{sim}}} \mathbf{z_t}(\hat{c}) \cdot \mathbf{z_t}(y_c), \quad \hat{c} \in \mathcal{C}_{\text{LLM}} \quad (10)$$

The top-ranked candidate serves as the semantic label $\hat{y}_t$ for the discovered class, and this class will become a seen class $\mathcal{Y}_{\text{seen}} \leftarrow \hat{y}$, and the model will treat this sample as the first instance of this new seen class. The image feature $\mathbf{z_i}(x)$ will be used as the initial prototype for this class, denoted as $\mathbf{p}_{\hat{y}}$. Subsequent samples will be compared to all known and seen prototypes, including this new $\mathbf{p}_{\hat{y}}$.

$$\hat{y}_t = \arg\max_{\hat{c}\in\mathcal{C}_{\text{LLM}}} R(\hat{c}) \quad (11)$$

### 3.4 CONFIDENCE-GUIDED TEST-TIME SEMANTIC STABILIZATION

We acknowledge that precisely predicting the ground-truth label of a never-before-seen class is inherently challenging due to the open-world nature of the task. By introducing LLMs, we take an initial step toward addressing the lack of semantic interpretability which is a fundamental challenge intrinsic to TTD. However, we believe that while accurate label prediction is the ultimate goal for downstream targets, it is not entirely strict for auxiliary classification. What matters more is that the discovered semantic label $\hat{y}_t$ occupies a reasonable position in the text embedding space.

Therefore, we propose a reliability strategy for test-time training guided by the semantic reranking score $R(\hat{c})$. A high $R$ indicates that the top candidate $\hat{y}_t$ is semantically consistent with known reference classes, suggesting a stable prediction. Conversely, a low $R$ implies ambiguous generation. For a sample $x$ with a stable prediction, we construct an attribute-rich textual description $\tilde{t}_{\hat{y}}$ with the predicted label $\hat{y}_t$. While a sample with an uncertain prediction, we avoid using a potentially incorrect label. Instead, we construct a relational textual prompt based solely on the similarity set $\mathcal{S}_{\text{sim}}$ and dissimilarity set $\mathcal{S}_{\text{dis}}$.

$$\tilde{t}_{\hat{y}} = \begin{cases} \mathcal{P}(\hat{y}_t, \mathcal{A}, \mathcal{S}_{\text{sim}}, \mathcal{S}_{\text{dis}}), & \text{if } \max_{\hat{c}\in\mathcal{C}_{\text{LLM}}} R(\hat{c}) > \theta, \\ \mathcal{P}(\mathcal{S}_{\text{sim}}, \mathcal{S}_{\text{dis}}), & \text{otherwise.} \end{cases} \tag{12}$$

For a test batch $\mathcal{B}$, the first type of loss applies to samples with the pseudo labels:

$$\begin{aligned} \mathcal{L}_{\text{kn}}(x,t) &= \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{kn}}}\Big[\mathcal{L}_{\text{text}}(t \mid \hat{y}) + \mathcal{L}_{\text{img}}(x \mid \hat{y}) + \mathcal{L}_{\text{cross}}((x,t) \mid \hat{y})\Big], \\ \mathcal{L}_{\text{un}}(x,\tilde{t}_{\hat{y}}) &= \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{un}}}\Big[\mathcal{L}_{\text{text}}(\tilde{t}_{\hat{y}} \mid \hat{y}) + \mathcal{L}_{\text{img}}(x \mid \hat{y}) + \mathcal{L}_{\text{cross}}((x,\tilde{t}_{\hat{y}}) \mid \hat{y})\Big]. \end{aligned} \tag{13}$$

The second kind of loss follows DAA (Liu et al., 2025) to conduct replay for known classes and renewal for seen classes by retraining on samples stored in a memory bank. Since both initial predictions and early-stage classifications are inevitably error-prone, we incorporate a self-correction mechanism as DAA does to, which in the TTD setting can effectively refine subsequent classification performance. Specifically, we retrieve the samples regularly, and these samples would be reprocessed, re-evaluated, and update SEGA again. The overall testing phase loss is shown below:

$$\mathcal{L}_{\text{test}} = \lambda_1 \cdot \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{kn}}}\left[\mathcal{L}_{\text{kn}}(x,t|\hat{y}) + \mathcal{L}_{\text{kn}}^{\text{replay}}(x)\right] + \lambda_2 \cdot \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{seen}}}\left[\mathcal{L}_{\text{un}}((x,\tilde{t}_{\hat{y}})|\hat{y}) + \mathcal{L}_{\text{un}}^{\text{renew}}(x|\hat{y})\right], \tag{14}$$

where $\lambda_1$ and $\lambda_2$ are hyperparameters that balance the contributions of each loss term.

## 4 EXPERIMENT

### 4.1 EXPERIMENTAL DETAILS

**Datasets**. We conduct our experiments based on four benchmark datasets. CIFAR100 (C100) (Krizhevsky et al., 2009) is a commonly used classic dataset. Tiny ImageNet (Le & Yang, 2015) is a subset of ImageNet (Deng et al., 2009). Caltech-UCSD Birds-200-2011 (CUB) (Wah et al., 2011) and FGVC-Aircraft (Maji et al., 2013) are complex fine-grained datasets. All these datasets are split into known and unknown classes (7:3). The model is trained on the known training set and tested on the mixture of known and unknown test sets. Because the dataset is used for discovery, we follow the names of four transformed datasets from HM (Lyu et al., 2025): CIFAR100D, CUB-200D, Tiny-ImageNetD, and AircraftD. More dataset details can be seen in Appendix.

**Implementation details**. In our implementation, we build our method on CLIP (Radford et al., 2021) as the image and text encoder, which employs a ViT-B/16 backbone (Khan et al., 2022) following the pretraining procedure of GCD and TTD work. A linear projection layer is added after the encoders for each modality and only the last layers of image encoders in the backbone are fine-tuned on the known classes, using SGD optimizer and cosine decay learning rate scheduler with an initial learning rate of 0.1 and minimum learning rate of 0.0001, and weight decay of 0.00005. All input images are resized to 224 × 224 and augmented to match the pretrained backbone settings.

**Evaluation metrics**. Following (Lyu et al., 2025), we use the following metrics. For unknown classes, we use two agreement metrics: (1) *True-label Agreement ratio* (**TA**). This metric measures the maximum proportion of samples from a given true class that are predicted as the same class;

Table 1: Comparisons on four datasets. Real-time evaluation reflects the accumulated performance across all test batches, while post evaluation reassesses all test samples after testing. **Bold** and underlining denote the best and second-best results.

| | Method | Real-time Evaluation | | | Post Evaluation | | | |
|---|---|---|---|---|---|---|---|---|
| | | KA↑ | TA↑ | CA↑ | KA↑ | TA↑ | CA↑ | KF↓ |
| CIFAR100D | L2P (Wang et al., 2022b) | 59.93±2.15 | 8.57±2.49 | 43.10±4.30 | 50.53±7.25 | 9.60±1.50 | 27.39±2.11 | 27.85±7.21 |
| | DP (Wang et al., 2022a) | 66.09±1.01 | 8.80±1.69 | 48.34±6.78 | 56.19±2.00 | 8.68±2.06 | 28.93±2.25 | 29.06±1.99 |
| | GMP (Cendra et al., 2024) | 72.77±1.20 | 7.37±0.88 | 42.26±4.54 | 67.21±2.53 | 13.04±2.59 | 27.10±1.77 | 17.69±2.53 |
| | PHE (Zheng et al., 2024a) | 72.35±1.12 | 16.74±1.05 | 37.34±1.31 | 70.12±1.07 | 14.55±**1.01** | 24.65±1.35 | **2.12**±0.92 |
| | HM (Lyu et al., 2025) | 79.17±**0.13** | 21.13±**0.62** | **56.37**±1.42 | 80.73±1.59 | 31.03±1.24 | 34.81±1.22 | 3.41±1.49 |
| | DAA (Liu et al., 2025) | 80.81±0.44 | 31.94±0.98 | 45.38±**0.88** | 80.27±**0.60** | 35.63±1.11 | 37.31±**1.20** | 4.10±0.60 |
| | Ours | **83.49**±0.64 | **34.88**±0.92 | 47.94±1.04 | **82.78**±0.89 | **38.64**±1.20 | **39.15**±1.24 | 3.04±**0.54** |
| CUB200D | L2P (Wang et al., 2022b) | 46.22±1.53 | 9.01±0.87 | 55.37±7.79 | 31.97±3.35 | 4.75±0.73 | 24.48±1.65 | 42.29±3.14 |
| | DP (Wang et al., 2022a) | 53.69±1.24 | 43.68±2.20 | 45.68±5.88 | 63.37±3.27 | 45.94±0.91 | 7.92±1.10 | 5.85±3.11 |
| | GMP (Cendra et al., 2024) | 62.97±1.33 | 46.44±1.87 | 47.99±4.34 | 58.11±3.00 | 48.02±1.20 | 10.31±1.45 | 5.46±2.77 |
| | PHE (Zheng et al., 2024a) | 57.29±1.23 | 32.68±1.66 | 57.22±2.75 | 54.77±1.75 | 27.92±**0.43** | 21.36±1.36 | **2.18**±0.71 |
| | HM (Lyu et al., 2025) | 66.20±0.55 | 58.30±2.37 | 43.33±6.10 | 64.42±0.65 | **65.28**±1.78 | 37.25±4.90 | 4.07±0.47 |
| | DAA (Liu et al., 2025) | 68.09±**0.33** | 54.49±1.03 | **58.34**±1.35 | 66.26±**0.39** | 64.13±2.23 | 44.65±**0.88** | 3.60±**0.39** |
| | Ours | **68.68**±1.21 | **58.30±0.62** | 57.72±**1.01** | **67.73**±0.94 | 64.25±1.63 | **46.00**±1.22 | 2.83±0.62 |
| T-ImageNetD | L2P (Wang et al., 2022b) | 46.25±1.41 | 7.79±2.92 | 53.55±6.47 | 29.50±3.77 | 10.38±2.42 | 23.28±**0.79** | 47.97±3.77 |
| | DP (Wang et al., 2022a) | 46.51±**0.58** | 6.41±0.93 | 58.27±6.10 | 28.53±3.33 | 9.63±2.10 | 26.80±1.38 | 47.57±3.32 |
| | GMP (Cendra et al., 2024) | 62.47±1.40 | 6.25±1.72 | 58.02±4.29 | 63.95±2.04 | 15.64±2.63 | 26.31±2.33 | 16.86±2.04 |
| | PHE (Zheng et al., 2024a) | 63.28±1.29 | 13.54±1.20 | 70.33±3.80 | 58.39±**1.14** | 12.80±**1.05** | 19.42±1.45 | 1.88±**0.35** |
| | HM (Lyu et al., 2025) | 75.31±1.31 | 16.04±0.76 | **73.81**±2.67 | 74.94±2.20 | 16.23±1.24 | 37.43±1.30 | **1.15**±2.18 |
| | DAA (Liu et al., 2025) | 76.38±0.82 | 24.10±0.80 | 71.70±2.02 | 75.50±1.96 | 23.17±1.35 | 36.41±1.12 | 2.39±1.56 |
| | Ours | **77.37**±1.53 | **26.01±0.75** | 72.72±**1.74** | **76.20**±1.93 | **25.77**±1.55 | **37.69**±1.27 | 1.65±0.46 |
| AircraftD | L2P (Wang et al., 2022b) | 43.21±1.08 | 20.11±1.20 | 35.52±1.93 | 40.17±2.34 | 21.00±1.21 | 27.21±1.33 | 15.04±2.10 |
| | DP (Wang et al., 2022a) | 40.18±1.10 | 21.88±1.23 | 34.80±1.79 | 42.53±1.46 | 20.22±**0.96** | 28.15±1.74 | 12.68±1.74 |
| | GMP (Cendra et al., 2024) | 41.26±1.04 | 21.12±0.67 | 35.10±2.50 | 32.56±1.75 | 21.25±1.02 | 26.34±1.48 | 22.65±2.25 |
| | PHE (Zheng et al., 2024a) | 45.82±**1.02** | 22.56±1.05 | 41.72±2.13 | 47.28±2.88 | 25.09±1.76 | 24.38±1.67 | 7.93±1.13 |
| | HM (Lyu et al., 2025) | 50.35±1.08 | 28.01±0.85 | 40.23±2.22 | 49.77±1.94 | 29.80±1.25 | 28.74±1.53 | 5.44±1.56 |
| | DAA (Liu et al., 2025) | 49.37±1.46 | 27.79±**0.31** | 40.51±**1.05** | 48.87±1.73 | 28.37±1.69 | 28.99±1.69 | 5.61±1.04 |
| | Ours | **51.27**±1.51 | **28.53**±0.32 | **41.84**±1.38 | **50.84±1.16** | **30.26**±1.17 | **30.44±1.23** | **4.42±0.74** |

and (2) *Cluster Agreement ratio* (**CA**). This metric measures the maximum proportion of samples from a given predicted cluster that have the same true label. For known classes, we also employ two key metrics to comprehensively assess the model's performance: *Known Accuracy* (**KA**) and *Known Forgetting* (**KF**). KA measures the traditional classification accuracy of the model on known classes while KF quantifies the degree of performance degradation on known classes over time. To quantitatively assess the quality of LLM-generated unknown class names, we introduce the *Semantic Score* ($\mathbf{S}^2$). This metric evaluates how semantically close the predicted class names are to the ground-truth labels in vector space via a pretrained Sentence-BERT encoder (Reimers & Gurevych, 2019), $\mathbf{S}^2(\hat{c}, y) = \frac{f_{\text{BERT}}(\hat{c}) \cdot f_{\text{BERT}}(y)}{\|f_{\text{BERT}}(\hat{c})\|_2 \|f_{\text{BERT}}(y)\|_2}$. For more details of the metrics, see Appendix.

### 4.2 Major Comparisons

In this paper, we first compare our methods with naive thresholding-based training-free methods. When exceeding the threshold, the naive method will be considered to have discovered a new class. And we compare with some training-required methods including L2P (Wang et al., 2022b), DP (Wang et al., 2022a), and GMP (Cendra et al., 2024) as these methods update prompts like TTA and CL methods. We also compare with the recent PHE (Zheng et al., 2024a) method for OCD and HM (Lyu et al., 2025), DAA (Liu et al., 2025) for TTD.

**Comparisons on TA & CA**. Table 1 presents a comprehensive comparison of our method against various TTD approaches across four benchmarks. Training-based methods often suffer from severe performance degradation, indicating that directly updating the model on streaming unknown classes destabilizes known-class recognition. In contrast, training-free method like HM achieve better known-class stability but exhibit only moderate TA, reflecting limited discovery capability. While DAA incorporates a lightweight module that can be updated which still only based on non-semantic discovery and performance remains limited. Notably, a good TTD method must balance and improve both TA and CA; a high value in only one metric often masks poor overall discovery quality. Our proposed SEGA framework achieves the best overall performance, demonstrating that bridging visual clusters to interpretable semantic concepts via LLM-enriched text alignment improves both discovery accuracy and knowledge retention.

**Comparison on Semantic Score of predictions**. To quantitatively evaluate the semantic quality of our discovered labels, we compute the $\mathbf{S}^2$ and Figure 4 reports averaged results over multiple runs on

Table 2: Comparisons of clustering metrics of post evaluation on four datasets.

| Method | CIFAR100D | | | | CUB200D | | | | T-ImageNetD | | | | AircraftD | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM |
| HM (Lyu et al., 2025) | 63.69 | 48.35 | 73.08 | 73.08 | 50.46 | 27.81 | 70.25 | 70.25 | 56.64 | 40.35 | 72.70 | 72.70 | 33.30 | 21.33 | 46.46 | 46.46 |
| DAA (Liu et al., 2025) | 65.62 | 53.96 | 75.20 | 75.21 | 58.19 | 32.43 | 75.29 | 75.29 | 58.91 | 42.50 | 73.53 | 73.53 | 32.08 | 21.20 | 46.44 | 46.44 |
| Ours | 66.73 | 53.47 | 75.51 | 75.51 | 58.58 | 33.66 | 75.07 | 75.07 | 58.55 | 43.50 | 74.59 | 74.59 | 33.74 | 22.58 | 47.12 | 47.12 |

CIFAR100D and CUB200D. First, replacing the simple *"a photo of [CLASS]"* template with our attribute-rich LLM descriptions method consistently improves Semantic Scores across all metrics, confirming that fine-grained attribute alignment enhances semantic discriminability. Moreover, when LLM is provided with some attribute assistance, the score of the predicted results is higher, whether it is the template sentence method or our method. Notably, the maximum score among the top-5 candidates rises over Top-1, showing that the correct label often appears within the first few LLM suggestions but may not always be ranked first. However, as we enlarge the candidate pool to top-10, the maximum score only marginally increases, while the average score over top-10 drops significantly which reflects that LLMs sometimes generate implausible or hallucinated candidates when asked for too many options, and these low-quality entries pull down the average.

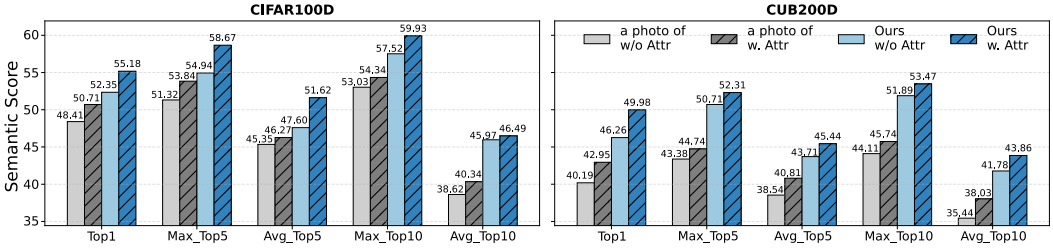



Figure 4: Comparisons of average Semantic Score ($\mathbf{S}^2$) on predictions of unknown classes.

### 4.3 Analysis on SEGA

**Examples of prediction procedure and performance**. Fig. 5 showcases representative discovery cases from our framework. When the candidate set includes the ground-truth label in some cases, the correct label is successfully selected with a high reranking score. Even when exact prediction is not achieved, the top candidate remains semantically close. And from the visualization of the text embedding space, the predicted label consistently lies closer to the ground-truth embedding than to any known-class prototype, while still maintaining a clear separation from visually similar known categories. Consequently, even a non-exact prediction provides a useful anchor for test-time training which respects both semantic relatedness and discriminability guiding the adaptation for unknown classes without causing catastrophic confusion.

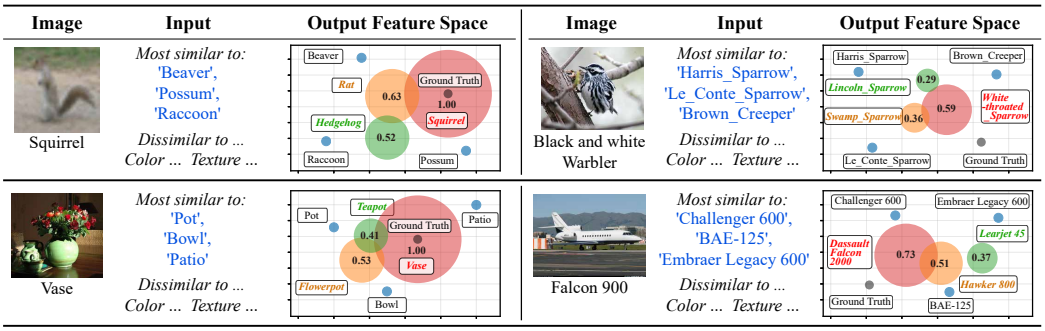



Figure 5: Some examples of our prediction procedure and textual feature distribution of output. (Size of bubble and value in bubble is $\mathbf{S}^2$ of the output.)

**Comparison of compatibility metrics.** TA and CA directly reflect online discovery quality and class consistency as the primary metrics, while we also provide some clustering metrics that traditional NCD methods use, including Hungarian Cluster Accuracy (HCA) (Meilă, 2003), Adjusted Rand Index (ARI) (Rand, 1971), Normalized Mutual Information (NMI) (McDaid et al., 2011) and V-Measure (Rosenberg & Hirschberg, 2007). The results in Table 2 demonstrate that our method also has better clustering performance on post evaluation.

**Ablation study**. As shown in Table 3, when none of these components is used which relies solely on a frozen CLIP model, the performance is poor, demonstrating that the pre-trained CLIP model struggles to handle the open-world TTD task due to its limited fine-grained semantic discriminability.

Adding only Test-time Training (T) leads to modest improvements, indicating that zero-shot test-time adaptation alone is insufficient without a well-structured semantic space. In contrast, employing only Semantic Pre-construction (C) brings a substantial performance gain, highlighting the importance of constructing an attribute-anchored semantic space before deployment. Finally, the full model combining all three components achieves the best performance, confirming that semantic pre-construction provides a solid foundation, test-time training enables dynamic adaptation, and semantic reranking effectively refines the performance.

**Reranking threshold**. Table 4 shows the impact of the confidence threshold $\theta$, which determines whether the top-ranked LLM-generated candidate is considered reliable enough to be used as a pseudo label for test-time training. Setting $\theta = 0$ which treats all predictions as reliable leads to the inclusion of many erroneous and noisy pseudo labels, which misguides subsequent test-time training and causes performance degradation. Conversely, setting $\theta = 1$ makes the method degenerate to using only relational cues similar/dissimilar known classes without explicit label and attribute assignment; while this conservative strategy avoids noise, it provides limited guidance and yields suboptimal results. The best performance is achieved at $\theta = 0.8$, where reliable predictions are adopted as pseudo labels to fully leverage semantic guidance, while unreliable ones are discarded and fall back to relation-only prompts, striking an effective balance between noise tolerance and semantic informativeness.

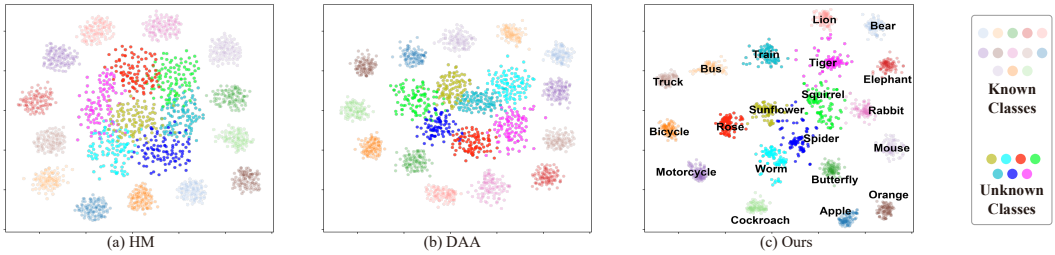
Table 3: Ablation study. C: Semantic Pre-construction, T: Test-time Training, S: Semantic Reranking, Sum: TA+CA.

| C | T | S | Real-time Eval | | | Post Eval | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | TA | CA | Sum | TA | CA | Sum | KF |
| | | | 20.57 | 31.16 | 51.73 | 23.54 | 29.26 | 52.80 | 8.81 |
| | ✓ | | 21.46 | 40.69 | 62.15 | 24.00 | 33.20 | 57.20 | 11.89 |
| ✓ | | | 25.07 | 34.59 | 59.65 | 25.27 | 31.53 | 56.80 | 6.54 |
| ✓ | ✓ | | 31.58 | 42.20 | 73.78 | 32.10 | 35.99 | 68.09 | 4.34 |
| ✓ | ✓ | ✓ | 34.74 | 47.77 | 82.51 | 38.66 | 39.21 | 77.87 | 3.28 |

Table 4: Comparisons of different reranking threshold. Sum: TA+CA.

| $\theta$ | Real-time Eval | | | Post Eval | | | |
|---|---|---|---|---|---|---|---|
| | TA | CA | Sum | TA | CA | Sum | KF |
| 0 | 30.88 | 46.59 | 77.46 | 31.40 | 37.57 | 68.96 | 5.39 |
| 0.6 | 31.43 | 47.46 | 78.89 | 33.98 | 37.35 | 71.33 | 4.32 |
| 0.7 | 32.35 | 45.50 | 77.85 | 35.27 | 35.93 | 71.19 | 4.06 |
| 0.8 | 34.74 | 47.77 | 82.51 | 38.66 | 39.21 | 77.87 | 3.28 |
| 0.9 | 33.93 | 46.60 | 79.63 | 38.25 | 38.15 | 76.40 | 3.61 |
| 1 | 33.15 | 45.90 | 79.05 | 37.63 | 37.76 | 75.39 | 3.82 |

**Post visualization using T-SNE**. In Fig. 6, we employ t-SNE (Van der Maaten & Hinton, 2008) to visualize the distribution of test samples. HM produces scattered and overlapping clusters, where known and unknown classes are poorly separated. DAA improves separation by amplifying feature discrepancies, yet the resulting clusters remain largely anonymous and categories are either mixed together or arbitrarily scattered, offering no interpretable organization. Our SEGA yields notably clearer and more structured clusters. Semantically similar classes like Bus and Truck, Apple and Orange, Rose and Sunflower are embedded close to each other, while still maintaining clear inter-class boundaries not only improving discrimination but also supporting understandable semantics.




(a) HM (b) DAA (c) Ours

Figure 6: T-SNE visualization in feature space on CIFAR100D with known and unknown classes.

## 5 Conclusion

We presented SEGA, a framework that transforms Test-Time Discovery from anonymous clustering into interpretable semantic concept generation. By leveraging LLM-generated attribute descriptions to construct an attribute-anchored visual-semantic space, and introducing Relational Semantic Grounding with confidence-guided test-time stabilization, SEGA enables unknown classes to be named and understood rather than merely separated. Experiments on four benchmarks show consistent improvements over prior TTD methods in both discovery accuracy and semantic coherence. While LLM-generated candidates may occasionally be noisy or hallucinated, the semantic anchoring in a discriminative feature space provides a practical and robust foundation for open-world discovery. Future work will focus on more principled uncertainty estimation and adaptive filtering to further suppress spurious predictions, and to more dynamic, open-ended environments.

### AI USE STATEMENT

In this work, GPT-4o-mini was used as an explicit component of SEGA: it generated attribute-rich descriptions for known categories before deployment and proposed candidate semantic names after the local model opened an unknown-class slot at test time. These outputs were treated as inputs or predictions of the evaluated system. The LLM did not determine novelty, create or maintain clusters, or replace the local classification and update procedure. Generative AI was not used to formulate or refine research hypotheses, develop the conceptual framework, design the methodology or experiments, implement the method, clean or reformat data, analyze data, or interpret results. Uses related to theoretical claims or proofs, translation, and synthetic dataset construction were not applicable. During manuscript preparation, generative AI tools were used only for language editing, including grammar correction, concision, and readability improvements. All scientific content, arguments, claims, and results were supplied and verified by the authors. The authors reviewed all AI-assisted language edits, verified the prompts and experimental pipeline, and evaluated the generated descriptions and candidate names through the quantitative and qualitative analyses reported in the paper. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

### REPRODUCIBILITY STATEMENT

We provide the implementation code as supplementary material. Section 3 specifies the model and training objectives, while Section 4 reports the implementation settings and evaluation protocol. Appendix B details the dataset splits, and the appendix reports more analyses of hyperparameters. The main comparisons in Table 1 were repeated across multiple runs and report both mean performance and standard deviation to make variability visible.

## REFERENCES

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics and the Sixth International Workshop on Semantic Evaluation*, pp. 385–393, 2012.

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*, 2022.

Fernando Julio Cendra, Bingchen Zhao, and Kai Han. Promptccd: Learning gaussian mixture prompt pool for continual category discovery. In *European Conference on Computer Vision*, 2024.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity: Multilingual and cross-lingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pp. 1–14, 2017.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2009.

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*, 2023.

Yingjun Du, Wenfang Sun, and Cees G Snoek. Ipo: Interpretable prompt optimization for vision-language models. *Advances in Neural Information Processing Systems*, 2024.

Shenghao Fu, Qize Yang, Qijie Mo, Junkai Yan, Xihan Wei, Jingke Meng, Xiaohua Xie, and Wei-Shi Zheng. Llmdet: Learning strong open-vocabulary object detectors under the supervision of large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

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*, 2019.

Kai Han, Sylvestre-Alvise Rebuffi, Sébastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. Automatically discovering and learning new visual categories with ranking statistics. In *International Conference on Learning Representations*, 2020.

Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

Ting Lei, Shaofeng Yin, Qingchao Chen, Yuxin Peng, and Yang Liu. Open-vocabulary hoi detection with interaction-aware prompt and concept calibration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Jiashuo Liu, Zheyan 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.

Tianle Liu, Fan Lyu, Chenggong Ni, Zhang Zhang, Fuyuan Hu, and Liang Wang. Daa: Amplifying unknown discrepancy for test-time discovery. *Advances in Neural Information Processing Systems*, 2025.

Fan Lyu, Tianle Liu, Zhang Zhang, Fuyuan Hu, and Liang Wang. Test-time discovery via hashing memory. *arXiv preprint arXiv:2503.10699*, 2025.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Aaron F McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms. *arXiv preprint arXiv:1110.2515*, 2011.

Marina Meilă. Comparing clusterings by the variation of information. In *Proceedings of Learning Theory and Kernel Machines*, 2003.

Nan Pu, Zhun Zhong, and Nicu Sebe. Dynamic conceptional contrastive learning for generalized category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 1971.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*, 2019.

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*, 2007.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.

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*, 2022.

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.

Enguang Wang, Zhimao Peng, Zhengyuan Xie, Fei Yang, Xialei Liu, and Ming-Ming Cheng. Get: Unlocking the multi-modal potential of clip for generalized category discovery. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Weishuai Wang, Ting Lei, Qingchao Chen, and Yang Liu. Semantic-guided novel category discovery. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022b.

Jianzong Wu, Xiangtai Li, Shilin Xu, Haobo Yuan, Henghui Ding, Yibo Yang, Xia Li, Jiangning Zhang, Yunhai Tong, Xudong Jiang, et al. Towards open vocabulary learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Yanan Wu, Zhixiang Chi, Yang Wang, and Songhe Feng. Metagcd: Learning to continually learn in generalized category discovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision*, pp. 5635–5662, 2024.

Jianyang Zhang, Qianli Luo, Guowu Yang, Wenjing Yang, Weide Liu, Guosheng Lin, and Fengmao Lv. Attribute-formed class-specific concept space: Endowing language bottleneck model with better interpretability and scalability. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Sheng Zhang, Salman Khan, Zhiqiang Shen, Muzammal Naseer, Guangyi Chen, and Fahad Shahbaz Khan. Promptcal: Contrastive affinity learning via auxiliary prompts for generalized novel category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020.

Haiyang Zheng, Nan Pu, Wenjing Li, Nicu Sebe, and Zhun Zhong. Prototypical hash encoding for on-the-fly fine-grained category discovery. *Advances in Neural Information Processing Systems*, 2024a.

Haiyang Zheng, Nan Pu, Wenjing Li, Nicu Sebe, and Zhun Zhong. Textual knowledge matters: Cross-modality co-teaching for generalized visual class discovery. In *European Conference on Computer Vision*, 2024b.

Zhun Zhong, Enrico Fini, Subhankar Roy, Zhiming Luo, Elisa Ricci, and Nicu Sebe. Neighborhood contrastive learning for novel class discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Chaoyang Zhu and Long Chen. A survey on open-vocabulary detection and segmentation: Past, present, and future. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

# APPENDIX

## A COMPARISON OF TTD TASK AND OTHER CATEGORY DISCOVERY TASK

We compare TTD with related settings. Out-of-Distribution (OOD) (Yang et al., 2024; Liu et al., 2021) detection neither discovers novel classes nor adapts during inference. TTA (Boudiaf et al., 2022) handles distribution shifts via self-supervised learning but assumes all test samples belong to known classes. NCD (Han et al., 2019) and GCD (Vaze et al., 2022) both aim to identify unknown classes at test time under an offline inference paradigm through clustering the entire test set. OCD (Du et al., 2023) 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 5: Comparison of evaluation protocols across related tasks. K: known classes, U: unknown classes.

| Task | Train | Test | Discovery | Evaluation |
|---|---|---|---|---|
| NCD | K | U | Post-hoc | Post-hoc |
| GCD | K | K + U | Post-hoc | Post-hoc |
| CGCD | K | K + $U_1$ + $U_2$ + $\cdots$ | Post-hoc | Post-hoc |
| OCD | K | K + U | Real-time | Post-hoc |
| TTA | K | Shifted K | N/A | Real-time |
| TTD | K | K + U | Real-time | Real-time |

**Setting Comparison**. In Table 5, we compare TTD with related settings, focusing on their evaluation protocols. In contrast, TTD requires models to classify known classes and initiate novel ones under sequential test-time constraints, while being evaluated in real time rather than post-hoc. *Existing NCD, GCD, CGCD, and OCD rely on post-hoc reclustering where the data for discovery and evaluation are inconsistent, which obscures the difficulties of real-time deployment.* TTD addresses this gap by jointly measuring discovery and classification during testing, thereby motivating our hash-based memory method.

**Challenges**. TTD requires real-time evaluation, making it more demanding than conventional NCD. The main challenges are threefold: *(1) distinguishing novel class discovery from the identification of already discovered ones*, *(2) coping with the scarcity of samples when a novel class first appears*, and *(3) maintaining performance on known classes while incorporating new ones to avoid forgetting (Kirkpatrick et al., 2017).*

## B DATASET DETAILS

We conduct our experiments using three widely recognized benchmark datasets: CIFAR100 (C100) (Krizhevsky et al., 2009), Caltech-UCSD Birds-200-2011 (CUB) (Wah et al., 2011), Tiny ImageNet (Le & Yang, 2015) and FGVC-Aircraft (Maji et al., 2013). 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 denote the transformed versions as CIFAR100D, CUB-200D, Tiny-ImageNetD, and AircraftD to reflect this adaptation.

Table 6: Statistic of the used datasets.

| | | **CIFAR100D** | | | **CUB-200D** | | |
|---|---|---|---|---|---|---|---|
| **Dataset** | Labeled | Known | Unknown | No. of samples | Known | Unknown | No. of samples |
| TrainSet | ✓ | 70 | 0 | 35000 | 140 | 0 | 4195 |
| TestSet | | 70 | 30 | 10000 | 140 | 60 | 5794 |
| | | **Tiny-ImagenetD** | | | **AircraftD** | | |
| **Dataset** | Labeled | Known | Unknown | No. of samples | Known | Unknown | No. of samples |
| TrainSet | ✓ | 140 | 0 | 70000 | 70 | 0 | 2331 |
| TestSet | | 140 | 60 | 10000 | 70 | 30 | 3333 |

The dataset partitioning follows the scheme outlined in Table 6. 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), Tiny-ImageNetD comprises classes 0–139 (140 known classes in total), and AircraftD includes classes 0-69 (70 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 AircraftD 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.

## 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 (Lyu et al., 2025) 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:

$$\text{KA} = \mathbb{E}_{c \in \mathcal{Y}_{\text{known}}} \frac{1}{|\mathcal{D}_c^{\text{test}}|} \sum_{x \in \mathcal{D}_c^{\text{test}}} \mathbf{1}(\hat{y}(x) = c), \tag{15}$$

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.

$$\text{KF} = \text{KA}_{\text{pre}} - \text{KA}_{\text{post}}, \tag{16}$$

where $\text{KA}_{\text{post}}$ and $\text{KA}_{\text{pre}}$ are the KA computed on all test data with known classes, before and after Testing Phase.

### 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 $\mathcal{C}_p^{\text{test}}$.

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

$$\text{TA} = \mathbb{E}_{c \in \mathcal{Y}_{\text{seen}}^{\text{GT}}} \frac{1}{|\mathcal{D}_c^{\text{test}}|} \max_{p \in \mathcal{Y}_{\text{seen}}} \left( \sum\nolimits_{x \in \mathcal{D}_c^{\text{test}}} \mathbf{1}[\hat{y}(x) = p] \right), \tag{17}$$

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

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

$$\text{CA} = \mathbb{E}_{p \in \mathcal{Y}_{\text{seen}}} \frac{1}{|\mathcal{C}_p^{\text{test}}|} \max_{c \in \mathcal{Y}_{\text{seen}}^{\text{GT}}} \left( \sum\nolimits_{(x,y) \in \mathcal{C}_p^{\text{test}}} \mathbf{1}(y = c) \right). \tag{18}$$

### C.3 Clustering metrics

Traditional novel class discovery (NCD) methods typically rely on post-cluster evaluation, where the quality of the discovered clusters is assessed after the entire test set has been processed. To ensure a comprehensive comparison with existing approaches, we also report several widely used clustering evaluation metrics, including Hungarian Cluster Accuracy (HCA) (Meilă, 2003), Adjusted Rand Index (ARI) (Rand, 1971), Normalized Mutual Information (NMI) (McDaid et al., 2011), and V-Measure (Rosenberg & Hirschberg, 2007). Note that these metrics are only evaluated after TTD, say post evaluation.

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

$$\text{HCA} = \mathbb{E}_{(x,y) \in \mathcal{D}^{\text{test}}}(y = \text{map}(\hat{y}(x))), \tag{19}$$

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

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

$$\text{ARI} = \frac{\text{RI} - \mathbb{E}[\text{RI}]}{\max(\text{RI}) - \mathbb{E}[\text{RI}]}, \tag{20}$$

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

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

$$\text{NMI}(\mathcal{U}, \mathcal{V}) = \frac{2 \cdot I(\mathcal{U}; \mathcal{V})}{H(\mathcal{U}) + H(\mathcal{V})}, \tag{21}$$

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.

$$\text{V-Measure} = \frac{2 \cdot h \cdot c}{h + c}, \tag{22}$$

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 $p(t,k) = \frac{count_k(t)}{N}$ is the joint distribution probability.

### C.4 Semantic Score($\mathbf{S}^2$)

Traditional classification metrics such as accuracy assume a fixed label or label vocabulary and require exact prediction matches. However, in the TTD setting, the model is tasked with naming previously unseen categories for which no ground-truth labels exist during training. Exact prediction of the true class name is inherently challenging and arguably an ill-posed requirement due to the open-world nature of the problem. A more reasonable evaluation paradigm is to measure whether the predicted name captures the semantic essence of the true category, even if phrased differently. Therefore, we employ a pretrained sentence embedding model to map both predictions and the ground truth into a shared semantic vector space. This approach follows established practices in semantic textual similarity (STS), where the semantic relatedness between two texts is commonly measured by comparing their sentence embeddings in a shared vector space (Agirre et al., 2012; Cer et al., 2017; Zhang et al., 2020). Sentence-BERT offers a favorable trade-off between computational efficiency and semantic fidelity, producing embeddings that correlate well with human judgments of semantic relatedness. Specifically, all texts are encoded using a Sentence-BERT (Reimers & Gurevych, 2019) model $f_{\text{BERT}}$ which produces dense vector representations (such as all-MiniLM-L6-v2). The Semantic Score is computed as the similarity between their embedding vectors:

$$\text{S}^2(\hat{c}, y) = \frac{f_{\text{BERT}}(\hat{c}) \cdot f_{\text{BERT}}(y)}{\|f_{\text{BERT}}(\hat{c})\|_2 \|f_{\text{BERT}}(y)\|_2}, \tag{23}$$

where $\hat{c}$ and $y$ represent the predicted category name and the ground-truth category name, respectively. $f_{\text{BERT}}(\cdot)$ denotes the pretrained Sentence-BERT encoder that maps a textual label into a dense semantic embedding. The score is computed as the cosine similarity between the two embeddings, with the numerator measuring their semantic alignment and the denominator normalizing their magnitudes. Thus, $\text{S}^2(\hat{c}, y)$ measures the semantic consistency between the predicted and ground-truth category names. A larger value indicates that the prediction is semantically closer to the true category, even if it is not an exact lexical match.

## D MORE EXAMPLES ON SEMANTICS

In Figure 7, we provide additional qualitative examples of our method's prediction procedure, along with the corresponding Semantic Score $\mathbf{S}^{(2)}$ for each discovered class. To further highlight the importance of semantic grounding, we implement a simple yet instructive variant of DAA: during test time, this variant is trained with textual labels and prompt LLM with the top-$k$ most similar and least similar known classes.

However, because the original DAA lacks any explicit semantic alignment mechanism, the "similar" classes it retrieves are often devoid of meaningful semantic relationships. DAA either fails to retrieve semantically relevant anchors or yields implausible candidate names. Consequently, its predictions become highly chaotic, with $\mathbf{S}^2$ scores significantly lower than those of our method. In contrast, our SEGA framework, equipped with attribute-anchored semantic space and relational semantic grounding, consistently produces semantically coherent and interpretable category names.

| Image | Input (DAA) | Output & $S^2$ (DAA) | Input (Ours) | Output & $S^2$ (Ours) |
|---|---|---|---|---|
|  Woman | *Most similar to:* 'Bridge', 'Cockroach', 'Dolphin' ... ... ... ... | 1.Whale - 0.31<br>2.Shark - 0.30<br>3.Eel - 0.21<br>4.Crocodile - 0.25<br>5.Caterpillar - 0.22 | *Most similar to:* 'Girl', 'Boy', 'Man' ... ... ... ... | 1.Woman - 1.00<br>2.Adult female - 0.66<br>3.Child - 0.33<br>4.Teenager - 0.35<br>5.Elder - 0.34 |
|  Snake | *Most similar to:* 'Mouse', 'Pear', 'Bed' ... ... ... ... | 1.Hamster - 0.34<br>2.Rat - 0.55<br>3.Gerbil - 0.32<br>4.Apple - 0.32<br>5.Orange - 0.39 | *Most similar to:* 'Lizard', 'Dinosour', 'Caterpillar' ... ... ... ... | 1.Snake - 1.00<br>2.Gecko - 0.51<br>3.Salamander - 0.53<br>4.Centipede - 0.32<br>5.Eel - 0.48 |
|  Tractor | *Most similar to:* 'Bus', 'Chair', 'Lizard' ... ... ... ... | 1.Stool - 0.28<br>2.Gecko - 0.17<br>3.Bench - 0.33<br>4.Train - 0.44<br>5.Throne - 0.30 | *Most similar to:* 'lawn_mower', 'pickup_truck', 'motorcycle' ... ... ... ... | 1.Garbage truck - 0.47<br>2.ATV - 0.63<br>3.Golf cart - 0.49<br>4.Excavator - 0.44<br>5.Bulldozer - 0.38 |
|  Trout | *Most similar to:* 'Maple_tree', 'Road', 'Clock' ... ... ... ... | 1.Oak_tree - 0.16<br>2.Elm_tree - 0.10<br>3.Street - 0.20<br>4.Sundial - 0.26<br>5.Roundabout - 0.07 | *Most similar to:* 'Aquarium_fish', 'Dolphin', 'Elephant' ... ... ... ... | 1.Whale - 0.41<br>2.Manatee - 0.41<br>3.Dugong - 0.32<br>4.Seal - 0.27<br>5.Sea lion - 0.31 |
|  Summer_ Tanager | *Most similar to:* 'Lazuli_Bunting', 'Frigatebird', 'Scarlet_Tanager' ... ... ... ... | 1.Blue_Grosbeak - 0.37<br>2.Varied_Bunting - 0.27<br>3.Blue_black_Grosbeak - 0.39<br>4.Vermilion_Flycatcher - 0.41<br>5.Rose_breasted_Grosbeak - 0.40 | *Most similar to:* 'Scarlet_Tanager', 'Vermilion_Flycatcher', 'Pied_Kingfisher' ... ... ... ... | 1.Crimson_Sunbird - 0.49<br>2.Summer_Tanager - 1.00<br>3.Northern_Cardinal - 0.41<br>4.Red_winged_Blackbird - 0.46<br>5.Belted Kingfisher - 0.31 |
|  Yellow_ Throated_Vireo | *Most similar to:* 'Great_Grey_Shrike', 'Song_Sparrow', 'Dark_eyed_Junco' ... ... ... ... | 1.House_Sparrow - 0.29<br>2.Eastern_Towhee - 0.27<br>3.Fox_Sparrow - 0.32<br>4.American_Tree_Sparrow - 0.34<br>5.Savannah_Sparrow - 0.30 | *Most similar to:* 'Western_Wood_Pewee', 'Yellow_bellied_Flycatcher', 'Olive_sided_Flycatcher' ... ... ... ... | 1.Gray_Flycatcher - 0.36<br>2.Eastern_Wood_Pewee - 0.35<br>3.Dusky_Flycatcher - 0.35<br>4.Gray_Flycatcher - 0.36<br>5.Alder_Flycatcher - 0.25 |

Figure 7: More examples of our prediction procedure and $\mathbf{S}^2$ of the output

## E MORE TTD COMPARISONS

### E.1 DETAILED COMPARISON ON CLUSTERING METRICS

We also provide a detailed post-evaluation comparison on clustering metrics of our method with several approaches on four benchmark datasets: CIFAR100D, CUB-200D, Tiny-ImageNetD, and AircraftD. Our method achieves the best performance on HCA, ARI, NMI, and VM metrics. This indicates that our method excels in hierarchical clustering, clustering quality, and matching accuracy. The results are summarized in Table 7 and Table 8 demonstrating that our method also has better clustering performance on both real-time and post evaluation.

Table 7: Detailed comparisons of clustering metrics on Post Evaluation. **Bold** and underlining denote the best and second-best results.

| | Method | HCA↑ | ARI↑ | NMI↑ | VM↑ |
|---|---|---|---|---|---|
| **CIFAR100D** | L2P (Wang et al., 2022b) | 50.33±2.08 | 37.29±1.68 | 69.94±5.04 | 69.94±5.04 |
| | DP (Wang et al., 2022a) | 55.87±1.14 | 44.39±1.57 | 72.44±0.89 | 72.44±0.89 |
| | GMP (Cendra et al., 2024) | 58.06±2.25 | 46.42±1.82 | 71.48±0.66 | 71.48±0.66 |
| | PHE (Zheng et al., 2024a) | 58.53±1.04 | 42.73±**0.99** | 70.74±1.44 | 70.74±1.44 |
| | HM (Lyu et al., 2025) | 63.69±**0.18** | 48.35±1.76 | 73.08±**0.54** | 73.08±**0.54** |
| | DAA (Liu et al., 2025) | 65.62±1.29 | **53.96**±1.73 | 75.21±1.41 | 75.21±1.41 |
| | Ours | **66.73**±1.22 | 53.79±1.38 | **85.35**±0.58 | **85.35**±0.58 |
| **CUB200D** | L2P (Wang et al., 2022b) | 41.58±1.33 | 27.18±1.65 | 71.15±1.82 | 71.15±1.82 |
| | DP (Wang et al., 2022a) | 48.36±2.38 | 24.19±1.00 | 48.60±**0.14** | 48.60±**0.14** |
| | GMP (Cendra et al., 2024) | 48.06±2.25 | 26.42±1.82 | 71.48±0.21 | 71.48±0.21 |
| | PHE (Zheng et al., 2024a) | 49.60±1.73 | 24.46±**0.17** | 65.58±1.20 | 65.58±1.20 |
| | HM (Lyu et al., 2025) | 50.46±**1.20** | 27.81±1.30 | 70.25±0.38 | 70.25±0.38 |
| | DAA (Liu et al., 2025) | 58.19±1.23 | 32.43±1.05 | **75.29**±0.71 | **75.29**±0.71 |
| | Ours | **58.58**±1.72 | **33.66**±1.06 | 75.07±0.36 | 75.07±0.36 |
| **T-ImageNetD** | L2P (Wang et al., 2022b) | 43.63±1.38 | 33.37±1.36 | 69.81±0.79 | 69.81±0.79 |
| | DP (Wang et al., 2022a) | 42.54±1.38 | 32.98±1.04 | 68.59±1.21 | 68.59±1.21 |
| | GMP (Cendra et al., 2024) | 56.18±**1.03** | 43.00±1.88 | 72.79±1.94 | 72.79±1.94 |
| | PHE (Zheng et al., 2024a) | 45.24±1.62 | 34.37±1.33 | 64.89±1.48 | 64.89±1.48 |
| | HM (Lyu et al., 2025) | 56.64±2.93 | 40.35±**0.74** | 72.70±0.99 | 72.70±0.99 |
| | DAA (Liu et al., 2025) | **58.91**±1.85 | 42.50±1.45 | 73.53±**0.12** | 73.53±**0.12** |
| | Ours | 58.55±1.09 | **43.50**±1.56 | **74.59**±0.25 | **74.59**±0.25 |
| **AircraftD** | L2P (Wang et al., 2022b) | 30.27±0.87 | 16.33±0.64 | 43.05±1.68 | 43.05±1.68 |
| | DP (Wang et al., 2022a) | 31.66±1.63 | 17.05±0.27 | 44.02±**0.35** | 44.02±**0.35** |
| | GMP (Cendra et al., 2024) | 27.74±0.82 | 17.61±**0.17** | 42.22±1.20 | 42.22±1.20 |
| | PHE (Zheng et al., 2024a) | 31.76±**0.29** | 19.76±1.36 | 44.70±1.13 | 44.70±1.13 |
| | HM (Lyu et al., 2025) | 33.30±0.73 | 21.33±1.76 | 46.46±0.94 | 46.46±0.94 |
| | DAA (Liu et al., 2025) | 32.08±1.87 | 21.20±0.96 | 46.44±0.85 | 46.44±0.85 |
| | Ours | **33.74**±1.45 | **22.58**±1.30 | **47.12**±0.36 | **47.12**±0.36 |

Table 8: Comparisons of clustering metrics on four datasets.

| Method | Real-time Evaluation | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CIFAR100D | | | | CUB200D | | | | T-ImageNetD | | | | AircraftD | | | |
| | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM | HCA | ARI | NMI | VM |
| HM (Lyu et al., 2025) | 61.49 | 43.04 | 82.78 | 82.78 | 52.84 | 34.90 | 83.51 | 83.51 | 61.76 | 38.18 | 87.24 | 87.24 | 30.02 | 20.16 | 44.25 | 44.25 |
| DAA (Liu et al., 2025) | 66.14 | 52.29 | 85.06 | 85.06 | 63.36 | 40.16 | 87.77 | 87.77 | 63.41 | 41.35 | 87.60 | 87.60 | 30.44 | 20.54 | 44.52 | 44.52 |
| Ours | 67.70 | 53.79 | 85.35 | 85.35 | 64.57 | 41.82 | 88.09 | 88.09 | 64.36 | 42.40 | 88.09 | 88.09 | 31.49 | 22.44 | 45.23 | 45.23 |

### E.2 Comparisons on different TTD method under different backbones.

To investigate the role of textual modality in Test-Time Discovery, we compare the original ViT-based HM and DAA with their CLIP-based variants (†), where the ViT visual encoder is replaced by a frozen CLIP vision-language backbone. As shown in Table 9, equipping the same TTD framework with CLIP consistently improves performance across all four datasets. For instance, on CIFAR100D, HM† (CLIP) increases TA from 21.13 to 28.50 compared to HM (ViT), while DAA† (CLIP) improves TA from 31.94 to 32.49 and CA from 45.38 to 46.97. Similar gains can be observed on CUB200D, Tiny-ImageNetD, and AircraftD, where CLIP-based variants achieve higher TA, CA, and often lower known forgetting (KF). These results confirm that pre-aligned vision-language representations provide rich semantic priors that benefit both known-class stability and novel-class discovery, even without explicit attribute enrichment. The improvement stems from the text encoder's ability to implicitly ground visual features to class-name semantics, which is absent in pure visual backbones. Our SEGA framework further exploits this textual modality through explicit attribute descriptions and relational grounding, achieving the best overall performance.

## F Running time

We evaluate the running time of the compared methods, as shown in Fig. 8. Training-based methods require significantly more time due to full backbone updates. Although updating the backbone enables adaptation to new knowledge, it incurs substantial time and memory overhead, and often leads to catastrophic forgetting. HM (Lyu et al., 2025) is training-free and thus faster than methods

Table 9: Comparisons on different TTD method under different backbones. † denotes our improved baseline that replaces the original ViT backbone with CLIP.

| | Method | Back bone | Real-time Evaluation KA↑ | TA↑ | CA↑ | Post Evaluation KA↑ | TA↑ | CA↑ | KF↓ |
|---|---|---|---|---|---|---|---|---|---|
| **CIFAR100D** | HM (Lyu et al., 2025) | ViT | 79.17±**0.13** | 21.13±**0.62** | **56.37**±1.42 | 80.73±1.59 | 31.03±1.24 | 34.81±1.22 | 3.41±1.49 |
| | HM † | CLIP | 81.42±1.67 | 28.50±1.50 | 50.79±1.14 | 81.13±1.86 | 33.57±1.91 | 35.42±**1.01** | 3.37±**0.12** |
| | DAA (Liu et al., 2025) | ViT | 80.81±0.44 | 31.94±0.98 | 45.38±**0.88** | 80.27±**0.60** | 35.63±**1.11** | 37.31±1.20 | 4.10±0.60 |
| | DAA † | CLIP | 81.11±1.01 | 32.49±1.33 | 46.97±1.74 | 81.29±1.23 | 36.29±1.57 | 37.44±1.80 | 3.21±0.21 |
| | Ours | CLIP | **83.49**±0.64 | **34.88**±0.92 | 47.94±1.04 | **82.78**±0.89 | **38.64**±1.20 | **39.15**±1.24 | **3.04**±0.54 |
| **CUB200D** | HM (Lyu et al., 2025) | ViT | 66.20±0.55 | 58.30±2.37 | 43.33±6.10 | 64.42±0.65 | **65.28**±1.78 | 37.25±4.90 | 4.07±0.47 |
| | HM † | CLIP | 67.04±1.73 | 56.09±1.12 | 47.51±1.23 | 65.85±1.75 | 60.97±1.97 | 44.79±1.31 | 3.15±**0.28** |
| | DAA (Liu et al., 2025) | ViT | 68.09±**0.33** | 54.49±1.03 | 58.34±1.35 | 66.26±**0.39** | 64.13±2.23 | 44.65±**0.88** | 3.60±0.39 |
| | DAA † | CLIP | 68.22±1.95 | 55.85±**0.59** | **58.46**±1.22 | 66.43±0.85 | 63.88±**1.12** | 45.10±1.08 | 2.96±0.36 |
| | Ours | CLIP | **68.68**±1.21 | **58.30**±0.62 | 57.72±**1.01** | **67.73**±0.94 | 64.25±1.63 | **46.00**±1.22 | **2.83**±0.62 |
| **T-ImageD** | HM (Lyu et al., 2025) | ViT | 75.31±1.31 | 16.04±0.76 | **73.81**±2.67 | 74.94±2.20 | 16.23±**1.24** | 37.43±1.30 | **1.15**±2.18 |
| | HM † | CLIP | 76.01±1.84 | 19.38±0.73 | 71.02±1.72 | 75.76±1.59 | 18.13±1.48 | 36.38±1.76 | 1.59±0.84 |
| | DAA (Liu et al., 2025) | ViT | 76.38±**0.82** | 24.10±0.80 | 71.70±2.02 | 75.50±1.96 | 23.17±1.35 | 36.41±**1.12** | 2.39±1.56 |
| | DAA † | CLIP | 76.94±1.97 | 25.91±**0.38** | 71.19±**1.51** | 75.65±**1.58** | 24.49±1.51 | 36.78±1.67 | 1.70±**0.42** |
| | Ours | CLIP | **77.37**±1.53 | **26.01**±0.75 | 72.72±1.74 | **76.20**±1.93 | **25.77**±1.55 | **37.69**±1.27 | 1.65±0.46 |
| **AircraftD** | HM (Lyu et al., 2025) | ViT | 50.35±**1.08** | 28.01±0.85 | 40.23±2.22 | 49.77±1.94 | 29.80±1.25 | 28.74±1.53 | 5.44±1.56 |
| | HM † | CLIP | 50.80±1.19 | 28.59±**0.17** | 40.17±1.26 | 50.01±1.20 | **30.80**±1.37 | 28.75±1.55 | 5.25±1.24 |
| | DAA (Liu et al., 2025) | ViT | 49.37±1.46 | 27.79±0.31 | 40.51±**1.05** | 48.87±1.73 | 28.37±1.69 | 28.99±1.69 | 5.61±1.04 |
| | DAA † | CLIP | 50.38±1.53 | **28.70**±0.78 | 40.95±1.32 | 49.69±1.31 | 29.34±1.76 | 29.56±1.87 | 5.57±0.80 |
| | Ours | CLIP | **51.27**±1.51 | 28.53±0.32 | **41.84**±1.38 | **50.84±1.16** | 30.26±**1.17** | **30.44±1.23** | **4.42±0.74** |

that update the model, but its neighbor search, hash bucketing, and graph construction still introduce noticeable latency. DAA (Liu et al., 2025) uses a lightweight adapter, making its per-step update even faster than HM's search procedure.

Compared with DAA, our method introduces additional computational overhead primarily from LLM-based candidate generation and reranking. Notably, the LLM API calls are triggered only when a sample is classified as novel. The number of such calls is bounded by the number of distinct unknown classes encountered during the entire test stream, which is typically small. Moreover, each call uses a lightweight prompt and a fast LLM (e.g., GPT-4o-mini), incurring only a few hundred milliseconds per query. As a result, the total API overhead remains not high compared to feature extraction and memory bank operations. Overall, our method runs slightly slower than DAA due to the extra attribute extraction and reranking logic, but remains significantly faster than HM.

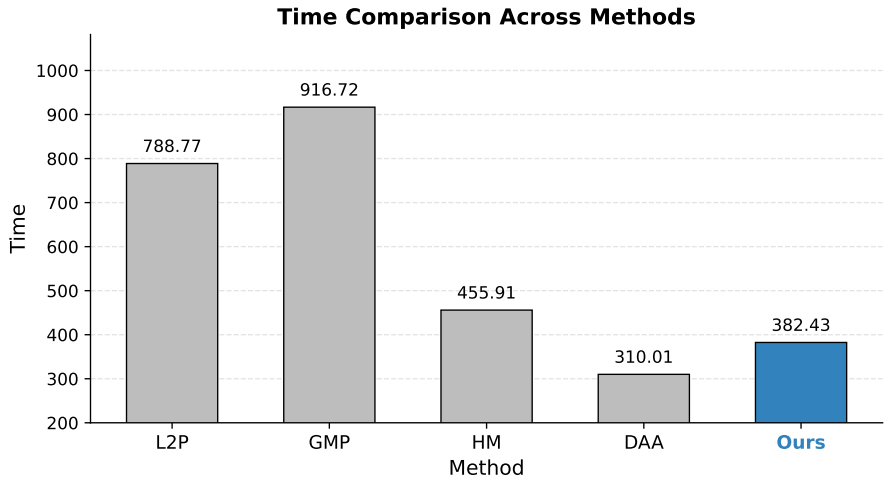



Figure 8: Comparisons of running time.

## G Different discoverable class numbers

The number of discoverable classes is a key hyperparameter in Test-Time Discovery, as it determines how many distinct unknown clusters the model is allowed to form. Table 10 reports the performance of

HM, DAA, and our SEGA framework under varying known/unknown splits as well as an unbounded setting where the model can discover an arbitrary number of new classes.

Across all settings, our method consistently outperforms both HM and DAA, demonstrating superior robustness to different discovery budgets. However, when the budget is set to an excessively large value, a clear trade-off emerges. Although real-time TA and CA continue to increase where our method reaches 34.04 TA and 85.72 CA in real-time, the post-evaluation metrics reveal a significant drop in TA (from 38.66 to 25.68) and a sharp rise in known forgetting (KF from 3.28 to 10.41). This suggests that an overly generous discovery allowance encourages the model to form many spurious or over-split clusters, which temporarily boosts online adaptability but ultimately corrupts previously acquired knowledge.

Moreover, in the infinite setting, every newly encountered sample that is classified as unknown triggers a full LLM query for candidate generation and reranking. Unlike the bounded settings where LLM calls are limited to the first few instances of each novel class, the unbounded budget leads to a potentially unlimited number of LLM invocations, significantly increasing computational overhead and latency. This practical concern further motivates choosing a reasonable discovery budget that balances semantic interpretability, clustering quality, and test-time efficiency.

Table 10: Comparisons of different discoverable class numbers.

| | Known + | **Real-time Eval** | | | **Post Eval** | | | |
|---|---|---|---|---|---|---|---|---|
| | Unknown | TA | CA | T+C | TA | CA | T+C | KF |
| **HM** | 70+30 | 21.11 | 56.87 | 77.98 | 31.03 | 34.81 | 65.84 | 3.47 |
| | 80+20 | 11.57 | 64.15 | 75.72 | 20.90 | 31.09 | 51.99 | 0.69 |
| | 90+10 | 14.92 | 52.98 | 67.90 | 21.50 | 30.61 | 52.11 | 0.44 |
| | 70+$\infty$ | 20.37 | 92.94 | 113.31 | 22.63 | 47.09 | 69.72 | 10.69 |
| **DAA** | 70+30 | 32.40 | 46.02 | 78.42 | 35.97 | 37.55 | 73.52 | 3.54 |
| | 80+20 | 36.21 | 48.13 | 84.34 | 32.56 | 31.44 | 64.00 | 1.73 |
| | 90+10 | 41.84 | 30.55 | 72.39 | 31.00 | 27.96 | 58.96 | 1.57 |
| | 70+$\infty$ | 31.78 | 89.58 | 121.36 | 22.35 | 49.32 | 71.67 | 11.07 |
| **Ours** | 70+30 | 34.74 | 47.77 | 82.51 | 38.66 | 39.21 | 77.87 | 3.28 |
| | 80+20 | 38.60 | 50.51 | 89.11 | 34.89 | 33.80 | 68.69 | 1.71 |
| | 90+10 | 42.98 | 33.72 | 76.70 | 32.24 | 30.40 | 62.63 | 1.16 |
| | 70+$\infty$ | 34.04 | 85.72 | 119.76 | 25.68 | 46.98 | 72.65 | 10.41 |

## H $\lambda_1$ AND $\lambda_2$ DURING TTT

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

$$\mathcal{L}_{\text{test}} = \lambda_1 \cdot \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{kn}}}\left[\mathcal{L}_{\text{kn}}(x,t|\hat{y}) + \mathcal{L}_{\text{kn}}^{\text{replay}}(x)\right] + \lambda_2 \cdot \mathbb{E}_{\hat{y}\in\mathcal{Y}_{\text{seen}}}\left[\mathcal{L}_{\text{un}}((x,\tilde{t}_{\hat{y}})|\hat{y}) + \mathcal{L}_{\text{un}}^{\text{renew}}(x|\hat{y})\right], \tag{24}$$

$\lambda_1 : \lambda_2$ balances known/replay and unknown/renewal losses. Larger known-class weight protects prior knowledge but limits adaptation; excessive unknown-class weight reduces stability. The grid selects $2:1$ as the best balance.

| $\lambda_1 : \lambda_2$ | Real-time TA | Real-time CA | Real-time TA+CA | Post TA | Post CA | Post TA+CA | KF |
|---|---|---|---|---|---|---|---|
| 100:1 | 31.08 | 43.74 | 74.82 | 30.83 | 35.68 | 66.50 | 1.30 |
| 10:1 | 32.40 | 46.47 | 78.87 | 36.93 | 37.59 | 74.52 | 2.37 |
| **2:1** | **34.74** | **47.77** | **82.51** | **38.66** | **39.21** | **77.87** | **3.28** |
| 1:1 | 33.70 | 47.98 | 81.68 | 36.35 | 39.07 | 75.43 | 4.29 |
| 1:2 | 31.70 | 49.34 | 81.04 | 34.76 | 40.87 | 75.64 | 4.98 |
| 1:10 | 30.23 | 48.76 | 78.99 | 31.85 | 37.12 | 68.96 | 6.76 |

# I COMPARISON OF DIFFERENT THRESHOLD $\gamma$

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

$$\hat{y} = \begin{cases} \arg\max_{c \in (\mathcal{Y}_{\text{kn}} \cup \mathcal{Y}_{\text{seen}})}(s_k), & \text{if } \max_c s_k > \gamma, \\ \text{new discovered class}, & \text{otherwise}, \end{cases} \tag{25}$$

$\gamma$ was selected through grid search in our experiment, and we analyze the impact of $\gamma$. The typical threshold of $\gamma$ is 0.7. If the threshold is too high, unknown classes are difficult to discover and will be mistakenly classified as known classes. On the contrary, if the threshold is too low, incoming samples will be erroneously classified as unknown classes. And our method is not sensitive to selection from one dataset to other datasets.

Table 11: Comparisons of different $\gamma$ during Testing phase.

| Threshold | **Real-time Eval** | | | **Post Eval** | | | |
|---|---|---|---|---|---|---|---|
| $\gamma$ | TA | CA | T+C | TA | CA | T+C | KF |
| 0.5 | 13.80 | 47.71 | 61.52 | 29.70 | 34.29 | 63.99 | 5.08 |
| 0.6 | 23.15 | 46.18 | 69.33 | 30.77 | 35.48 | 66.25 | 4.13 |
| 0.7 | 34.74 | 47.77 | 82.51 | 38.66 | 39.21 | 77.87 | 3.28 |
| 0.8 | 30.41 | 45.43 | 75.84 | 33.75 | 31.37 | 65.12 | 5.10 |
| 0.9 | 14.48 | 46.84 | 61.32 | 32.32 | 28.95 | 61.27 | 9.62 |

# J COMPARISON OF DIFFERENT THRESHOLD $k$

In our Relational Semantic Grounding (RSG), we retrieve the top-$k$ most similar known classes as semantic anchors to help characterize an unknown sample. The choice of $k$ controls how much relational context is provided to the LLM for candidate generation. As shown in Table 12, when $k$ is too small, the retrieved anchors are insufficient to capture the semantic neighborhood of the unknown class, leading to ambiguous or incomplete prompts and degraded discovery performance. Conversely, when $k$ is too large, the prompt becomes cluttered with loosely related or even misleading known classes, which confuses the LLM and causes a drop in both TA and CA. Moreover, a larger $k$ increases the number of text encoder forward passes and LLM prompt tokens, leading to higher computational overhead at test time. The best trade-off is achieved at $k = 3$, where the prompt provides enough discriminative yet compact relational information, yielding the highest agreement scores and semantic quality. Therefore, we set $k = 3$ as the default in all main experiments.

Table 12: Comparisons of different $k$ during Testing phase.

| $k$ | **Real-time Eval** | | | **Post Eval** | | | | **S**$^2$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | TA | CA | T+C | TA | CA | T+C | KF | Top1 | Max_Top5 | Avg_Top5 |
| 1 | 28.42 | 40.11 | 68.53 | 33.40 | 36.32 | 69.72 | 5.46 | 50.10 | 52.60 | 45.69 |
| 2 | 32.92 | 45.05 | 77.97 | 35.66 | 37.30 | 72.96 | 4.67 | 52.88 | 55.27 | 48.68 |
| 3 | 34.74 | 47.77 | 82.51 | 38.66 | 39.21 | 77.87 | 3.28 | 55.18 | 58.67 | 51.62 |
| 5 | 31.90 | 46.60 | 78.50 | 34.29 | 36.09 | 70.38 | 4.20 | 53.43 | 56.54 | 49.34 |
| 10 | 26.98 | 39.83 | 66.81 | 32.18 | 35.97 | 68.15 | 6.74 | 49.10 | 50.36 | 43.23 |

# K FURTHER ANALYSIS OF SEMANTIC GROUNDING IN SEGA

## K.1 SCOPE OF SEMANTIC ASSISTANCE AND CONTRIBUTION

SEGA retains the central requirement of TTD: a local model must recognize known classes, decide when an incoming sample initiates a novel class, and maintain that class in a continuous unlabeled stream. Semantic assistance is introduced only after the visual discovery rule opens a novel-class slot. RSG then translates the slot's relations to known classes and its visual attributes into an open-ended

set of candidate names, which are reranked locally in the learned visual-semantic space. Thus, the language model neither classifies individual stream images nor decides whether a novel class exists.

This division of responsibility distinguishes SEGA from both prior visual-only TTD methods and direct multimodal recognition. HM and DAA discover and update classes from visual evidence, but the resulting classes remain anonymous. SEGA adds a semantic coordinate system for known classes, grounds a discovered slot relative to these anchors, and admits the resulting semantic signal into adaptation only through the reliability gate in Sec. 3.4. The contribution is therefore the integration of open-ended semantic grounding with streaming discovery, rather than description generation, a generic adaptation loss, or an isolated LLM call. LLM-generated descriptions have also been used for zero-shot recognition and VLM adaptation (Zhang et al., 2025; Lei et al., 2025; Fu et al., 2025); those settings generally select among known target labels, whereas TTD must first establish the identity of a class whose label space is unavailable before deployment.

Unless explicitly augmented in the controlled experiments below, the baselines in Table 1 use their published visual pipelines and do not use LLM-generated known-class descriptions or online LLM/VLM calls.

### K.2 Relational evidence and deterministic visual attributes

RSG already receives high-level visual-semantic evidence through the most similar and dissimilar known-class anchors and through local reranking in the aligned embedding space. The role of $\Phi_{\text{attr}}(x)$ is complementary: it supplies concise, verbalizable appearance cues that may not be resolved by class-level relations alone. Passing a learned visual feature vector directly to a text-only LLM would not be meaningful without an additional learned image-to-text interface while replacing that interface with a VLM would instead move image recognition to an external multimodal model which is unrealistic in our setting.

We therefore use a deterministic, label-free image-processing operator rather than an additional learned semantic extractor. It summarizes HSV statistics (dominant and central color, saturation, and brightness), Gaussian-smoothed Sobel texture and edge statistics, and coarse spatial composition. These cues are inexpensive, stable across the stream, and deliberately restricted to secondary evidence; they do not replace either the visual representation or the relational anchors.

The prompt evidence has four complementary roles. Similar anchors locate a discovered class in a local semantic neighborhood, dissimilar anchors exclude incompatible regions, and attributes separate candidates that share similar neighbors but differ in appearance. The LLM converts this structured evidence into an open-ended candidate set, after which the learned semantic space performs local reranking. Low-confidence candidates are withheld from semantic feedback by the reliability gate.

Table 13 isolates these sources of evidence. Relation-only guidance already improves over DAA without generating a semantic name. LLM candidate generation adds open-ended semantics, and deterministic attributes provide a further improvement in both semantic quality and TTD agreement.

Table 13: Ablation of the evidence used for relational semantic grounding. Dashes indicate variants that do not generate candidate names and therefore have no semantic score.

| Evidence | $S^2$ Top-1 | $S^2$ Max Top-5 | $S^2$ Avg. Top-5 | Real-time TA+CA | Post TA+CA |
|---|---|---|---|---|---|
| DAA | – | – | – | 77.54 | 72.10 |
| Similar + dissimilar relations (no LLM) | – | – | – | 79.19 | 74.81 |
| Similar + dissimilar relations (LLM) | 48.18 | 51.17 | 45.56 | 80.15 | 76.22 |
| Similar + dissimilar relations + attributes (SEGA) | 55.25 | 58.57 | 51.28 | 82.55 | 77.63 |

### K.3 Language-model invocation and direct VLM recognition

Known-class descriptions are generated before deployment and cached locally. During testing, a text-only LLM query is triggered only when the local visual rule opens a novel-class slot, so the number of online queries scales with the number of discovered slots rather than the number of stream images. Candidate reranking and subsequent classification are performed by the local text and image encoders.

Direct image-based VLM naming provides a useful upper-strength comparison because it exposes each discovery event to a stronger external recognizer. It is not, however, equivalent to the intended TTD pipeline: real-time discovery through local model implementation, while querying individual images transfers recognition from the deployed model to an external API. Table 14 reports both predictive results and API cost in a single comparison.

Table 14: Comparison with direct image-based VLM naming. The lower panel reports the average cost of one text or image API query.

| Method/input | $S^2$ Top-1 | $S^2$ Top-5 | Real-time TA+CA | Post TA+CA | Image-VLM calls | API time / tokens |
|---|---|---|---|---|---|---|
| *End-to-end comparison* | | | | | | |
| Direct-VLM | 62.18 | 65.96 | 85.79 | 79.51 | 58 | 352.51 s / 157,248 |
| SEGA | 55.25 | 58.57 | 82.55 | 77.63 | 0 | 47.65 s / 3,527 |
| *Average cost per API query* | | | | | | |
| Text input | – | – | – | – | – | 0.735 s / 100 |
| Image input | – | – | – | – | – | 6.520 s / 2,637 |

Direct-VLM improves the semantic and agreement scores in this controlled comparison to some extent, but requires 58 image calls and substantially greater latency and token consumption. SEGA instead uses the LLM as a sparse semantic reference after local discovery, preserving the local discovery-and-update process while avoiding per-image multimodal inference.

### K.4 Controlled attribution of the performance gains

The main ablation in Table 3 separates semantic pre-construction, test-time training, and semantic reranking under a common pipeline; Table 4 further isolates the reliability gate. We additionally construct two families of controls to distinguish the effect of the representation from the effect of SEGA's semantic grounding procedure.

In the first family, the original HM and DAA online strategies are retained and only their ViT backbone is replaced by CLIP. In the second, the HM and DAA online policies are run on SEGA's trained visual-semantic backbone; these are SEGA-backbone variants rather than the published methods. A final control retains the SEGA pipeline but replaces LLM-enriched descriptions with the conventional prompt "a photo of [CLASS]." Table 15 consolidates all three comparisons using real-time TA+CA.

Table 15: Controlled comparison of representation, online policy, and description source. All entries report real-time TA+CA.

| Dataset | HM + CLIP | DAA + CLIP | SEGA backbone + HM | SEGA backbone + DAA | Fixed prompt | Full SEGA |
|---|---|---|---|---|---|---|
| CIFAR100D | 79.29 | 79.46 | 77.62 | 80.11 | 81.61 | 82.82 |
| CUB200D | 103.60 | 114.31 | 104.30 | 115.03 | 115.55 | 116.02 |
| Tiny-ImageNetD | 90.40 | 97.10 | 92.27 | 97.69 | 98.16 | 98.73 |
| AircraftD | 68.76 | 69.65 | 67.26 | 69.71 | 70.15 | 70.37 |

Replacing the backbone improves the visual baselines but does not close the gap to full SEGA. Likewise, applying the HM/DAA policies to the SEGA representation remains below the complete method. The fixed-prompt variant shows that the visual-semantic pipeline itself is strong, while the consistent additional gain from full SEGA measures the value of attribute-rich descriptions and relational grounding. Taken together with the component and gate ablations, these controls show that neither a CLIP backbone nor LLM enrichment alone accounts for the complete result.

### K.5 Predefined vocabularies and open-ended grounding

A fixed vocabulary provides a different semantic interface from SEGA. Direct CLIP retrieval can select a name from a predefined lexicon, while semantic-discovery methods can expand the candidate source with an external resource such as WordNet. Both alternatives expose the model to a naming space before deployment; consequently, their performance depends on its coverage and granularity. We evaluate both variants under the TTD protocol in Table 16.

Table 16: Comparison of alternative semantic interfaces for TTD. The fixed vocabulary explicitly contains the ground-truth names and therefore represents an easier label-space assumption.

| Method | Candidate source | Exact Top-1 | $S^2$ Top-1 | Real-time TA+CA | Post TA+CA |
|---|---|---|---|---|---|
| Fixed-vocabulary retrieval | Predefined lexicon with ground truth | 66.74 | 70.15 | 90.13 | 82.67 |
| Semantic discovery | External lexicon (e.g., WordNet) | 21.05 | 53.86 | 80.78 | 76.48 |
| SEGA | Open-ended RSG candidates | 22.85 | 55.66 | 82.55 | 77.63 |

The fixed vocabulary performs best, as expected, because it includes the ground-truth categories and converts open-ended naming into closed-set retrieval. The external-lexicon result is much closer to SEGA, illustrating the difficulty of choosing the correct concept from a large generic vocabulary. These methods are informative comparisons but not equivalent substitutes for TTD: in the target setting, neither the number nor the identity of novel classes is known, and the system must first discover and maintain a class before assigning it a name. SEGA therefore uses semantic information to assist that discovery-and-grounding process rather than treating a precompiled vocabulary as the target label space.

### K.6 Complementary semantic evaluation

Sentence-BERT-based $S^2$ measures graded semantic and we complement it in two more ways. First, the generated and ground-truth names are encoded with SEGA's aligned CLIP text encoder, and their Top-1 and maximum Top-5 cosine similarities are reported in the learned semantic space. Second, a blind questionnaire asks 32 participants to inspect an image or representative cluster together with anonymized generated names, and to judge both exact-or-synonym identity and whether a name is semantically useful and non-confusing.

Table 17: Complementary evaluation of generated names (%). "Fixed prompt" denotes SEGA with "a photo of [CLASS]" in place of attribute-rich descriptions.

| Method | CLIP Top-1 | CLIP Max Top-5 | Human exact/synonym | Human useful/non-confusing |
|---|---|---|---|---|
| Fixed prompt | 52.98 | 58.72 | 36.50 | 50.81 |
| Full SEGA | 60.31 | 65.89 | 43.47 | 58.81 |

Exact ground-truth naming remains intrinsically difficult because a class must be inferred from a short, evolving stream without access to a target vocabulary. The purpose of semantic grounding is therefore broader than exact string recovery: a non-exact name can still provide a useful, non-confusing anchor for discovery and adaptation when it occupies an appropriate and well-separated position in the learned space, as illustrated in Fig. 5. Full SEGA improves over the fixed-prompt control under both the model-based and human evaluations, although improving exact naming accuracy remains an important direction for future work.