ENERGY GRADIENT SPECIFICATION: PROVABLE GENERALIZATION GUARANTEES FOR LEARNWARE IDENTIFICATION AND REUSE

Anonymous authors
Paper under double-blind review

ABSTRACT

The *learnware paradigm* enables reusing well-established models on new tasks without retraining or exposing private training data, and a *learnware dock system* identifies reusable candidates through the compact *specification* stored with each model. However, existing specifications primarily capture statistical information, which may discard predictive information under tight storage budgets and make specification similarity an unreliable indicator of reuse compatibility. In an attempt to address this limitation, we propose **E**nergy **G**radient **S**pecification (EGS), which characterizes and compares well-established models and user tasks in a shared one-step energy gradient space. Specifically, EGS optimizes compact proxy sets to match their gradient responses and uses the resulting gradients as specifications. We further establish a capacity-based generalization bound with only logarithmic dependency on the learnware size, together with an excess-risk decomposition explaining why reuse accuracy rises and then saturates as the dock system expands. Our theoretical results offer new insights into understanding how the learnware size affects reuse performance. Extensive experiments on image and text benchmarks show that EGS outperforms existing distribution-based specification methods across multiple learnware reuse scenarios.

1 INTRODUCTION

Machine learning has achieved remarkable success across diverse domains, while developing a high-quality model for a new task often requires substantial labeled data, computation, and expertise. Model reuse is also challenging when training data are private, model architectures are heterogeneous, and user supervision is limited, while adaptation or continual learning may incur additional cost and catastrophic forgetting. These challenges raise a fundamental question: *How can users effectively leverage well-established models without accessing their training data or retraining them?*

The *learnware paradigm* (Zhou, 2016; Zhou & Tan, 2024) organizes well-established models as *learnwares* in a *learnware dock system*, with each learnware pairing a model with a compact specification of its reuse-relevant properties (Lei et al., 2026a) for reuse on new tasks. Given a user task, the dock system matches its specification against stored learnware specifications to identify suitable learnwares for reuse (Liu et al., 2026), including complementary reuse beyond individual learnware functionalities (Tan et al., 2024a). Since developer and user data cannot be directly shared across parties and exhaustive model evaluation is impractical, specifications serve as the key interface for learnware identification and reuse (Lei et al., 2024). Their quality therefore directly determines how effectively the dock system can characterize and compare models and tasks for reuse.

Existing learnware specifications show that compact statistical information can effectively support learnware identification and reuse (Tan et al., 2024c). Distribution-based methods such as RKME (Wu et al., 2023) compare user and learnware data distributions in a reproducing kernel Hilbert space, providing a practical basis for specification comparison (Lei et al., 2026a). However, distributional similarity does not necessarily imply reuse compatibility: *under a limited specification size, distribution matching may discard class-conditional and task-relevant predictive information*. In contrast, gradient matching characterizes how data induce local changes in a shared network by aligning their learning signals (Zhao & Bilen, 2023), thereby capturing optimization-induced

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.

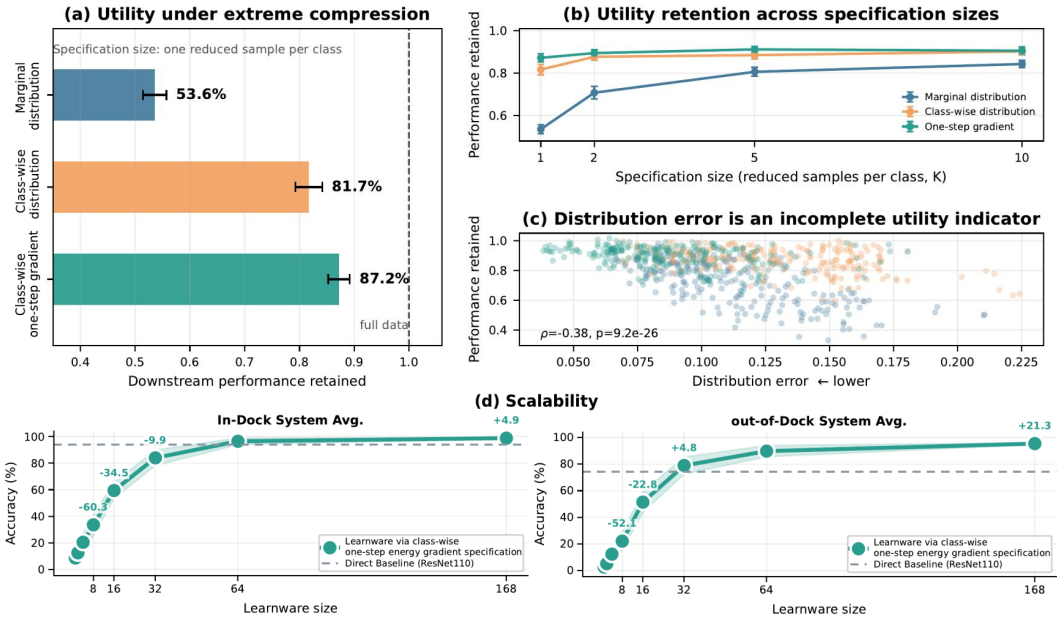


Figure 1: **Predictive utility and scalability of specifications. (a–c) Specification study.** Under matched different specification sizes, we compare marginal distribution, class-wise distribution, and class-wise one-step gradient specification. We use a linear probe to evaluate the downstream predictive utility retained by each specification. Experimental details and results are provided in Appendix A. **(a) Class-conditional information is important under extreme compression.** At specification size $K = 1$, the three methods retain 53.6%, 81.7%, and 87.2% of full-data performance, respectively. **(b) Gradient matching achieves the highest utility** across the evaluated specification sizes. **(c) Distribution error alone does not fully predict utility.** Although distribution error is negatively associated with retained utility, specifications with similar distribution errors can yield different downstream utility. **(d) Dock system scalability.** As the learnware size increases, the reuse accuracy of the dock system improves substantially before gradually saturating.

behavior beyond static distributional statistics (Zhao et al., 2021). This perspective is consistent with trajectory-based distillation (Cazenavette et al., 2022) and theoretical results showing that even one-step gradients from random initialization can encode nontrivial task-dependent representation changes (Ba et al., 2022). These observations motivate using one-step gradients to preserve predictive information that may not be fully captured by distributional discrepancy alone.

To better understand the design requirements of an effective learnware specification, we conduct a motivating study from two complementary perspectives: the information preserved under a limited specification size and the scalability of our specification as the learnware dock system expands. As shown in Fig. 1(a–c), preserving class-conditional information is crucial when the specification size is constrained, and class-wise one-step gradients retain more downstream predictive utility than marginal or class-wise distribution matching. Fig. 1(d) further examines the scaling behavior of our method, showing that reuse performance steadily improves as more learnwares are incorporated into the dock system. These observations motivate a **E**nergy **G**radient **S**pecification (EGS), which preserves task-relevant predictive information while remaining effective as the dock system scales.

Despite the empirical success of existing learnware methods (Wu et al., 2023; Liu et al., 2025b), the theoretical understanding of learnware identification and reuse remains limited (Lei et al., 2024; 2026a). From the perspective of learning theory, two aspects need to be explored: 1) whether reuse carries generalization guarantees when the dock system selects from many learnwares, and 2) whether the rise-and-saturate scaling behavior in Fig. 1(d) can be explained theoretically rather than only observed empirically. Answering these questions is essential for understanding why and when specification-based reuse is guaranteed to work. Our contributions are summarized as follows:

- We propose EGS for learnware identification and reuse, which constructs class-wise gradient responses to characterize and compare well-established models and user tasks in a shared one-step energy gradient space.

- We provide generalization guarantees for learnware identification and reuse. The cost of selecting from a growing dock system is only logarithmic in the number of candidates, and reuse performance is further shown to be governed by two decoupled factors, dock coverage and specification distortion, which account for the plateau observed in Fig. 1(d).
- Extensive experiments on image and text benchmarks demonstrate that EGS consistently outperforms existing distribution-based specification methods and exhibits robust performance across diverse learnware identification and reuse settings.

2 RELATED WORK

The learnware paradigm (Zhou, 2016) provides a systematic framework for managing and reusing well-established models, allowing users to address new tasks without training models from scratch. Central to this paradigm is the learnware specification (Liu et al., 2026), which serves as a compact and comparable representation of a learnware for identification and reuse. Existing specification methods are predominantly distribution-based, characterizing learnwares through the compact distribution or feature statistics of the tasks mastered by their associated models. Representative approaches include Reduced Kernel Mean Embedding (RKME) (Wu et al., 2023), Label-Aware Neural Embedding (LANE) (Chen et al., 2025b), and subsequent extensions that improve label awareness and discriminative alignment (Liu et al., 2025b; Chen et al., 2025a). Beyond specification design, prior work has studied heterogeneous learnware reuse (Tan et al., 2024b; 2023; 2024a; Guo et al., 2023), unseen-task adaptation (Zhang et al., 2021; Tan et al., 2026), and efficient learnware identification (Xie et al., 2023; Liu et al., 2024; 2025a), while recent studies have begun to establish theoretical foundations for learnware specifications (Lei et al., 2024; 2026a). The Beimingwu learnware dock system (Tan et al., 2024c) further provides a practical platform for learnware management, identification, and reuse. More recently, parameter-based specifications have been explored to characterize model capabilities through task-induced parameter changes in large pre-trained foundation models (Shi et al., 2026; Lei et al., 2026b). Although these methods move beyond conventional distribution-based specifications, they typically depend on large pre-trained models and task-specific adaptation, whereas our proposed EGS constructs specifications through class-wise gradient responses in a shared one-step energy gradient space without adapting a large foundation model.

3 METHODOLOGY

Problem Setup. The learnware paradigm contains two stages: the *Developer-Oriented Submitting Stage*, which constructs learnware specifications and stores corresponding learnwares, and the *User-Oriented Deploying Stage*, which identifies reusable learnwares and reuses their associated models.

1) Developer-Oriented Submitting Stage. Suppose c developers participate in the learnware dock system. Developer i holds a private training dataset $\mathcal{D}_i = \{(\boldsymbol{x}_{i,n}, y_{i,n})\}_{n=1}^{N_i}$ and a well-established model $f_{\boldsymbol{\theta}_i} : \mathcal{X} \to \mathcal{Y}$ satisfying $\mathcal{R}_i(f_{\boldsymbol{\theta}_i}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\left[\ell\left(y, f_{\boldsymbol{\theta}_i}(\boldsymbol{x})\right)\right] \leq \epsilon$, where ℓ is the task-specific loss and $\epsilon > 0$ is a small tolerance. The dock system then constructs a learnware specification Φ_i to sketch the model capability and stores the resulting learnware $\{(f_{\boldsymbol{\theta}_i}, \Phi_i)\}$ for future use.

2) User-Oriented Deploying Stage. For a user with a limited labeled task $\hat{\mathcal{D}} = \{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}})\}_{\hat{n}=1}^{\hat{N}}$, the dock system constructs a user specification $\hat{\Phi}$ and identifies suitable learnwares $\mathcal{I}$ by matching it against the stored learnware specifications. Rather than training a task-specific model from scratch, the user can directly reuse the well-established models $\{f_{\boldsymbol{\theta}_i}\}_{i\in\mathcal{I}}$ associated with the identified learnwares to address the target task.

3.1 ENERGY GRADIENT SPECIFICATION

Existing specification methods typically sketch model capabilities and user requirements into a unified distributional metric space for reuse assessment, such as the RKHS adopted by distribution-based specifications. In contrast, the proposed EGS establishes a shared one-step energy gradient space, where model capabilities and user requirements are represented by their gradient responses and directly compared to measure reuse compatibility.

Shared One-Step Energy Gradient Space. To establish a shared gradient space for characterizing and comparing well-established models and user tasks, we introduce a shared network $\phi_{\boldsymbol{\xi}} = g_{\boldsymbol{w},\boldsymbol{\eta}} \circ \psi_{\boldsymbol{\vartheta}}$, where $\psi_{\boldsymbol{\vartheta}} : \mathcal{X} \to \mathbb{R}^d$ is a frozen modality-specific feature extractor and $g_{\boldsymbol{w},\boldsymbol{\eta}}$ is a randomly parameterized prediction function. Specifically,

$$\phi_{\boldsymbol{\xi}}(\boldsymbol{x}) = \boldsymbol{\eta}^\top \sqrt{2}\cos\big(\boldsymbol{w}^\top \psi_{\boldsymbol{\vartheta}}(\boldsymbol{x}) + b\big)\,,$$

where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ denotes the randomly initialized linear energy parameters. The random Fourier feature (RFF) parameters are sampled as $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \sigma^{-2}\boldsymbol{I}_d)$ and $b \sim \mathrm{Uniform}[0, 2\pi)$, with $\boldsymbol{w}$ and b representing the random frequency and phase, respectively, and σ controlling the probing scale, set to 1 by default. The RFF mapping provides an explicit finite-dimensional approximation to a shift-invariant kernel, allowing the shared network to capture nonlinear relations in the feature space while retaining a computationally efficient parameterization (Rahimi & Recht, 2007).

This shared parameterization defines a common energy gradient coordinate system in which both model capabilities and user requirements can be represented and directly compared. Given an input–target pair $(\boldsymbol{x}, y)$, we define its one-step gradient with respect to the shared energy parameters $\boldsymbol{\eta}$ as

$$\boldsymbol{G}(\boldsymbol{x}, y; \boldsymbol{\eta}) = \nabla_{\boldsymbol{\eta}} \ell(g_{\boldsymbol{w},\boldsymbol{\eta}} \circ \psi_{\boldsymbol{\vartheta}}(\boldsymbol{x}), y), \tag{1}$$

where $\ell(\cdot, \cdot)$ denotes the task loss, such as cross-entropy for classification. Because all developer models and user tasks are probed under the same parameterization, their resulting gradients lie in a shared coordinate space and can serve as comparable representations for subsequent specification construction and reuse assessment.

Gradient Matching in the Shared Gradient Space. For a well-established model $f_{\boldsymbol{\theta}}$ trained on a private dataset $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, we characterize its capability using the one-step energy gradient in Eq. 1, with the model prediction $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ serving as the target instead of the ground-truth label. To preserve class-conditional structure, we further compute a class-wise energy gradient as

$$\boldsymbol{G}_{\mathcal{D}^k}(\boldsymbol{x}, f_{\boldsymbol{\theta}}(\boldsymbol{x}); \boldsymbol{\eta}) = \nabla_{\boldsymbol{\eta}} \frac{1}{|\mathbf{B}_{\boldsymbol{x}}^k|} \sum_{\boldsymbol{x} \in \mathbf{B}_{\boldsymbol{x}}^k} \ell(g_{\boldsymbol{w},\boldsymbol{\eta}} \circ \psi_{\boldsymbol{\vartheta}}(\boldsymbol{x}), f_{\boldsymbol{\theta}}(\boldsymbol{x})), \tag{2}$$

where $\mathbf{B}_{\boldsymbol{x}}^k \subseteq \mathcal{D}^k := \{(\boldsymbol{x}, y) \in \mathcal{D} \mid y = k\}$ is a mini-batch sampled from class k. By jointly incorporating task inputs and model-predicted labels, $\{\boldsymbol{G}_{\mathcal{D}^k}\}_{k=1}^K$ captures the model capabilities. To avoid directly exposing the private training data and their gradients, we introduce a compact proxy set $\mathcal{S} = \{(\boldsymbol{s}_m, y_{\boldsymbol{s}_m})\}_{m=1}^M$, where $M \ll N$. The proxy samples are randomly initialized as $\boldsymbol{s}_m \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and subsequently optimized to match the class-wise energy gradients of the original data in the shared gradient space. The optimization objective is defined as

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathcal{L}_{\text{EGS}}(\mathcal{D}, \mathcal{S}) = \sum_{k=1}^K \|\boldsymbol{G}_{\mathcal{D}^k} - \boldsymbol{G}_{\mathcal{S}^k}\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and

$$\boldsymbol{G}_{\mathcal{S}^k}(\boldsymbol{s}, y_{\boldsymbol{s}}; \boldsymbol{\eta}) = \nabla_{\boldsymbol{\eta}} \frac{1}{|\mathbf{B}_{\boldsymbol{s}}^k|} \sum_{(\boldsymbol{s}, y_{\boldsymbol{s}}) \in \mathbf{B}_{\boldsymbol{s}}^k} \ell(g_{\boldsymbol{w},\boldsymbol{\eta}} \circ \psi_{\boldsymbol{\vartheta}}(\boldsymbol{s}), y_{\boldsymbol{s}}) \tag{4}$$

represents the class-wise one-step energy gradient computed for the proxy set $\mathcal{S}$. By matching class-wise gradients in the shared energy-gradient space, the proxy set learns task-dependent and model-dependent learning signals. Since optimized proxy samples may differ across realizations, we characterize them through their induced energy gradients rather than the samples themselves. Accordingly, the learnware specification is defined as $\Phi^{\text{EGS}} = \{\boldsymbol{G}_{\mathcal{S}^k}\}_{k=1}^K$. Because these gradients are computed under the same shared network, they have fixed dimensionality and are directly comparable across learnwares, enabling identification and reuse without exposing the original private data. The complete procedure is provided in Algorithm 1.

3.2 Developer-Oriented Submitting Stage

For developer i, the targets in Eq. 3 are given by the predictions of the submitted well-established model $f_{\boldsymbol{\theta}_i}$ on its private training task $\mathcal{D}_i = \{(\boldsymbol{x}_{i,n}, y_{i,n})\}_{n=1}^{N_i}$. The corresponding proxy set $\mathcal{S}_i$ is optimized to reproduce these model-induced gradients in the shared one-step energy gradient space:

$$\mathcal{S}_i^* = \arg\min_{\mathcal{S}_i} \mathcal{L}_{\text{EGS}}(\mathcal{D}_i := \{(\boldsymbol{x}, f_{\boldsymbol{\theta}_i}(\boldsymbol{x}))\}, \mathcal{S}_i).$$

The resulting learnware specification is given as $\Phi_i^{\text{EGS}} = \{\boldsymbol{G}_{\mathcal{S}_i^k}\}_{k=1}^{K_i}$ and the corresponding learnware $\{(f_{\boldsymbol{\theta}_i}, \Phi_i^{\text{EGS}})\}$ is stored in the dock system for subsequent identification and reuse. Since all specifications are constructed in the shared one-step energy gradient space, they can be directly compared with user specifications to assess reuse compatibility. The complete submitting procedure is summarized in Algorithm 2.

3.3 User-Oriented Deploying Stage

Given a user task with limited labeled data $\hat{\mathcal{D}} = \{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}})\}_{\hat{n}=1}^{\hat{N}}$, the dock system optimizes a proxy set $\hat{\mathcal{S}}$ according to Eq. 3 in the shared one-step energy gradient space:

$$\hat{\mathcal{S}}^* = \arg\min_{\hat{\mathcal{S}}} \mathcal{L}_{\text{EGS}}(\hat{\mathcal{D}} := \{(\hat{\boldsymbol{x}}, \hat{y})\}, \hat{\mathcal{S}}).$$

The resulting user specification is defined as $\hat{\Phi}^{\text{EGS}} = \{\hat{\boldsymbol{G}}_{\hat{\mathcal{S}}^{\hat{k}}}\}_{\hat{k}=1}^{\hat{K}}$. While the learnware specification sketches model capability from developer inputs and model predictions, the user specification sketches task requirements from user inputs and ground-truth labels, enabling direct comparison in the shared gradient space. The dock system then compares $\hat{\Phi}^{\text{EGS}}$ with the stored specifications $\{\Phi_i^{\text{EGS}}\}_{i=1}^{c}$ to identify reusable learnwares. Specifically, the partial one-to-one correspondence matrix $\boldsymbol{E}_i$ between the user and i-th learnware specifications is defined as

$$\begin{aligned} \boldsymbol{E}_i^* &= \arg\min_{\boldsymbol{E}_i} \sum_{\hat{k}=1}^{\hat{K}} \sum_{k=1}^{K_i} e_i^{\hat{k},k} \left\| \hat{\Phi}^{\hat{k}} - \Phi_i^k \right\|_F^2, \\ \text{s.t.} \quad & \sum_{k=1}^{K_i} e_i^{\hat{k},k} \le 1, \quad \sum_{\hat{k}=1}^{\hat{K}} e_i^{\hat{k},k} \le 1 \quad \text{and} \quad \sum_{\hat{k}=1}^{\hat{K}} \sum_{k=1}^{K_i} e_i^{\hat{k},k} = \min(\hat{K}, K_i) \end{aligned} \tag{5}$$

where $\hat{\Phi}^{\hat{k}} = \hat{\boldsymbol{G}}_{\hat{\mathcal{S}}^{\hat{k}}}$ and $\Phi_i^k = \boldsymbol{G}_{\mathcal{S}_i^k}$. The cardinality constraint avoids the trivial empty matching, while the one-to-one constraints allow unmatched classes when the two label spaces differ in size. After solving the matching problem for each candidate learnware, the dock system assigns each user class to the matched learnware class with the smallest gradient distance across all candidates. This yields the final class-wise reuse relation $\mathcal{M} = \{(\hat{k}, i, k) \mid \hat{k} \in [\hat{K}], i \in [c], k \in [K_i]\}$, together with the selected learnware set $\mathcal{I} = \{i : \exists, (\hat{k}, i, k) \in \mathcal{M}\}$. The corresponding models $\{f_{\boldsymbol{\theta}_i}\}_{i \in \mathcal{I}}$ are then retrieved, and each user-class subset is routed to its assigned model for reuse. The complete deployment procedure is summarized in Algorithm 3.

4 Theoretical Analysis

We present the theoretical analysis of learnware identification and reuse under EGS. The difficulty of reuse is primarily determined by two quantities: *the number of user classes $\hat{K}$ that the routed predictor must serve simultaneously*, and *the learnware size c from which a model is selected for each user class*. We first introduce the reuse function class, the routed risk and the assumptions for learnware reuse, and derive a capacity-based generalization bound with no dependency on $\hat{K}$ in terms of class-averaged risk and only a $\sqrt{\log(c\bar{K})}$ dependency on c. Then, we decompose the excess reuse risk into a coverage term that decays geometrically with c and a specification term independent of c, which explains why the reuse accuracy rises and then saturates as the learnware size increases. The proof sketches and detailed proofs of the theoretical results are provided in Appendix C.

4.1 The Capacity Bound for Learnware Reuse

Let the dock system store c learnwares $\{(f_{\boldsymbol{\theta}_i}, \Phi_i^{\text{EGS}})\}_{i=1}^{c}$, with the i-th learnware covering K_i classes, and let $\bar{K} = \max_{i \in [c]} K_i$. The deploying stage assigns a learnware-class pair $\pi(\hat{k}) = (i_{\hat{k}}, k_{\hat{k}})$ drawn from the *candidate pool* $\mathcal{P} = \{(i, k) : i \in [c],\ k \in [K_i]\}$, of size at most $P := c\bar{K}$, to each user class $\hat{k} \in [\hat{K}]$. For a candidate pair (i, k), the frozen model $f_{\boldsymbol{\theta}_i}$ is deployed with its k-th

local class aligned to the routed user class, so each candidate contributes a *single* fixed aligned score function $h_{i,k}(\boldsymbol{x}) = \big[f_{\boldsymbol{\theta}_i}(\boldsymbol{x})\big]_k$, forming the singleton class $\mathcal{H}_{i,k} = \{h_{i,k}\}$. The *reuse hypothesis pool* induced by the dock system is $\mathcal{H} = \bigcup_{(i,k)\in\mathcal{P}} \mathcal{H}_{i,k}$, with a maximum size of P. The space of *possible* submissions is infinite, we define $\mathcal{H}^\star$ for the *global candidate pool* induced by all possible submissions, whereas the class $\mathcal{H}$ actually formed by the realized dock system is a finite subset. Crucially, $\mathcal{H}^\star$ entails no additional complexity cost: all bounds below are conditional on the realized dock system and depend on it only through P and the uniform envelope E. The routed predictor draws every coordinate from the pool $\mathcal{F}_{\mathrm{LW}} = \big\{\boldsymbol{x} \mapsto f(\boldsymbol{x}) = \big(f_1(\boldsymbol{x}), \ldots, f_{\hat{K}}(\boldsymbol{x})\big) : f_{\hat{k}} = h_{\pi(\hat{k})},\ \pi : [\hat{K}] \to \mathcal{P},\ h_{\pi(\hat{k})} \in \mathcal{H}_{\pi(\hat{k})}\big\}$. The routing π is *unconstrained*, which effectively relaxes the partial one-to-one feasibility region in Algorithm 3, so the uniform bounds over $\mathcal{F}_{\mathrm{LW}}$ remains valid for the deployed predictor, and our results provide worst-case theoretical guarantees for Algorithm 3.

We score a routed predictor with the weighted Hamming loss surrogate (Zhang et al., 2025): $\ell_{\mathrm{W}}\big(f(\boldsymbol{x}), y\big) = \frac{1}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} v_{\hat{k}}(y)\, \ell_{\hat{k}}\big(f_{\hat{k}}(\boldsymbol{x}), y\big)$, with base class losses $\ell_{\hat{k}}$ and weights $v_{\hat{k}}(y) \in [0, V]$. The *routed* choice $v_{\hat{k}}(y) = \mathbf{1}[y = \hat{k}]$ ($V = 1$) matches the deployment of Algorithm 3, wherein a user sample is scored solely by the model associated with its specific class, the $\hat{K}$ summands then have *disjoint supports*, improving the bound by a further factor $\sqrt{\hat{K}}$. The expected and empirical reuse risks are $\mathcal{R}_{\mathrm{LW}}(f) = \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\hat{\boldsymbol{X}}Y}}\big[\ell_{\mathrm{W}}(f(\boldsymbol{x}), y)\big]$ and $\hat{\mathcal{R}}_{\hat{\mathcal{D}},\mathrm{LW}}(f) = \frac{1}{\hat{N}} \sum_{\hat{n}=1}^{\hat{N}} \ell_{\mathrm{W}}\big(f(\hat{\boldsymbol{x}}_{\hat{n}}), \hat{y}_{\hat{n}}\big)$, with induced loss space $\mathcal{L}_{\mathrm{LW}} = \{\ell_{\mathrm{W}}(f(\boldsymbol{x}), y) : f \in \mathcal{F}_{\mathrm{LW}}\}$. The *class-$\hat{k}$ risk* $\mathcal{R}_{\hat{k}}(h) := \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\hat{\boldsymbol{X}}Y}}\big[\mathbf{1}[y = \hat{k}]\, \ell_{\hat{k}}\big(h(\boldsymbol{x}), y\big)\big]$ satisfies $\mathcal{R}_{\mathrm{LW}}(f) = \frac{1}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \mathcal{R}_{\hat{k}}\big(f_{\hat{k}}\big)$ for arbitrary class marginals. Let $\hat{\mathcal{D}}_{\hat{k}} = \{\hat{\boldsymbol{x}}_{\hat{n}} : \hat{y}_{\hat{n}} = \hat{k}\}$ denote the user examples of class $\hat{k}$ and $\hat{N}_{\hat{k}} = |\hat{\mathcal{D}}_{\hat{k}}|$.

Assumption 1. *(i) The reuse loss and every candidate score function are bounded: $\ell_{\mathrm{W}}(\cdot,\cdot) \le M$, and $|h(\boldsymbol{x})| \le E$ for all $h \in \mathcal{H}^\star$ and $\boldsymbol{x} \in \mathcal{X}$, where $M, E > 0$ are constants; (ii) For every user class $\hat{k} \in [\hat{K}]$ and every label y, the base loss $\ell_{\hat{k}}(\cdot, y)$ is μ-Lipschitz continuous.*

Assumption 1 is mild: E is finite since every $h \in \mathcal{H}^\star$ is an aligned score of a frozen model (a class probability or a bounded logit), and the Lipschitz condition covers the squared, logistic, squared hinge and smoothed ramp losses, etc. The boundedness on every candidate score function is required over the global pool $\mathcal{H}^\star$, so that it holds uniformly over all possible submissions and in particular over every realized dock system $\mathcal{H} \subseteq \mathcal{H}^\star$. Under the routed weights, $\ell_{\mathrm{W}} \le B/\hat{K}$ when every base loss is bounded by B, so the penalty term below can use the smaller range $M = B/\hat{K}$.

Theorem 1. *Let $\mathcal{F}_{\mathrm{LW}}$ be the learnware reuse function class and let Assumption 1 hold. Given a dock system of c learnwares with candidate pool size $P \le c\bar{K}$ and the routed weights $v_{\hat{k}}(y) = \mathbf{1}[y = \hat{k}]$ actually used by Algorithm 3. Then, for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of a user dataset $\hat{\mathcal{D}}$ of size $\hat{N}$, the following holds for every $f \in \mathcal{F}_{\mathrm{LW}}$:*

$$\mathcal{R}_{\mathrm{LW}}(f) \;\le\; \hat{\mathcal{R}}_{\hat{\mathcal{D}},\mathrm{LW}}(f) + \frac{2C_\ell\, \mu E}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \frac{\hat{N}_{\hat{k}}}{\hat{N}} \sqrt{\frac{2\log(c\bar{K})}{\hat{N}_{\hat{k}}}} + 3M\sqrt{\frac{\log\frac{2}{\delta}}{2\hat{N}}}, \tag{6}$$

where C_ℓ is the absolute constant of the scalar contraction inequality. For EGS *deployments the envelope is explicit, $E = 1$ for class-probability outputs and $E = B$ for logits clipped at level B.*

Remark 1. *Theorem 1 does not involve the $\hat{K}$ factor on the class-averaged risk scale, and the dependency on learnware size c is at most logarithmic: since every candidate class is a singleton, its empirical Rademacher complexity vanishes and the dock system cost arises solely from the selection among the $P \le c\bar{K}$ frozen candidates. This cost is determined by the number of candidates and manifesting as $\sqrt{\log(c\bar{K})}$, which explains why the cost remains manageable even in the large-scale dock system. For balanced classes $\hat{N}_{\hat{k}} = \hat{N}/\hat{K}$, the order of the bound is $\tilde{\mathcal{O}}\big(\sqrt{\log(c\bar{K})}/\sqrt{\hat{N}\hat{K}} + 1/\sqrt{\hat{N}}\big)$, the impact of class imbalance is reflected only through the $\hat{N}_{\hat{k}}$ inside the square root.*

4.2 Dock System Scaling Law: Why Performance Saturates

A generalization bound alone cannot explain Fig. 1(d), in which accuracy *rises* with the learnware size and then *saturates*, we decompose the excess reuse risk into a coverage term that decreases geometrically in c and a specification term independent of c.

We make three assumptions (formally stated as Assumptions 2-4 in Appendix C): (i) *class coverage*: a learnware ε-covers a user class if its per-class risk is within ε of Bayes, with per-class uncoverable probability $\pi_{\hat{k}}$ (a diversity floor) and per-learnware covering probability $p_{\hat{k}}$; (ii) *routing margin and specification alignment*: the population gradient distances $D_{\hat{k},(i,k)}$ are risk-aligned and separated by $\Delta(c) > 0$; and (iii) *uniform proxy accuracy*: every optimized proxy specification is within $\bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}})$ of its full-data counterpart, uniformly over all possible submissions.

Let $\hat{f}$ be the predictor returned by Algorithm 3, f^{opt} the Bayes-optimal predictor for the user task,

$$\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}}) = \frac{1}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \mathbb{E}\big\|\hat{\boldsymbol{G}}_{\hat{\mathcal{S}}^{\hat{k}}} - \hat{\boldsymbol{G}}_{\hat{\mathcal{D}}^{\hat{k}}}\big\|_F + \bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}}) \tag{7}$$

be the expected distortion from representing the user task and every learnware with M_{spec} proxy samples instead of their full data, $\hat{\boldsymbol{G}}_{\hat{\mathcal{D}}^{\hat{k}}}$ denoting the full-data analogue of $\hat{\boldsymbol{G}}_{\hat{\mathcal{S}}^{\hat{k}}}$. Since the user specification is estimated from $\hat{N}$ examples, we define $\epsilon_{\mathrm{data}}(\hat{N}) = \frac{1}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \mathbb{E}\big[\big\|\hat{\boldsymbol{G}}_{\hat{\mathcal{D}}^{\hat{k}}} - \bar{\boldsymbol{G}}^{\hat{k}}\big\|_F\big]$ as its sampling error against the population gradient of Assumption 3. Based on the bounded difference property in Hilbert spaces, the magnitude of this error is $O\big(\frac{1}{\hat{K}} \sum_{\hat{k}} 1/\sqrt{\hat{N}_{\hat{k}}}\big)$.

Theorem 2. *Let Assumptions 1-4 hold and let the weights be routed, $v_{\hat{k}}(y) = \mathbf{1}[y = \hat{k}]$. Then, in expectation over the structural types of the user classes, the independent generation of the dock system, the proxy-set optimization and the draw of the user sample,*

$$\begin{aligned} &\mathbb{E}\big[\mathcal{R}_{\mathrm{LW}}(\hat{f})\big] - \mathcal{R}_{\mathrm{LW}}\big(f^{\mathrm{opt}}\big) \\ &\leq \underbrace{\Delta_{\max}\left[\bar{\pi} + \frac{1}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} (1 - p_{\hat{k}})^c\right] + \varepsilon\big(1 - \bar{\pi}\big)}_{\mathcal{E}_{\mathrm{cov}}(c)} + \underbrace{\frac{2\Delta_{\max}}{\Delta(c)} \big(\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}}) + \epsilon_{\mathrm{data}}(\hat{N})\big)}_{\mathcal{E}_{\mathrm{spec}}(M_{\mathrm{spec}}, \hat{N})}, \end{aligned} \tag{8}$$

where $\Delta_{\max} = \max_{\hat{k}} \big[\sup_{h \in \mathcal{H}^\star} \mathcal{R}_{\hat{k}}(h) - \mathcal{R}_{\hat{k}}\big(f^{\mathrm{opt}}\big)\big]$ is the worst per-class approximation gap over the global pool $\mathcal{H}^\star$, $\bar{\pi} = \frac{1}{\hat{K}} \sum_{\hat{k}} \pi_{\hat{k}}$ and $p_{\min} = \min_{\hat{k}} p_{\hat{k}}$.

Remark 2. *The coverage term splits into a diversity floor $\Delta_{\max}\bar{\pi}$, a proximity floor $\varepsilon(1 - \bar{\pi})$ on ε-covered classes, and a dock system-dependent part (geometric in c). The specification term is independent of c by Assumption 4: a flipped ranking on class $\hat{k}$ requires realized distortion $\geq \Delta(c)/2$, with probability at most $2\big(\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}}) + \epsilon_{\mathrm{data}}(\hat{N})\big)/\Delta(c)$ on average over the classes (Appendix C.6). Regarding the scope of this identification result, it should be noted that the selector analyzed is the* unconstrained *per-class minimizer of the estimated distances, a relaxation of Algorithm 3 excluding regimes where the one-to-one constraints of Eq. 5 are active, hence, the bound serves as a worst-case theoretical guarantee for* EGS.

Corollary 3. *Under the conditions of Theorem 2, the dock system-dependent part of the excess reuse risk, $\frac{\Delta_{\max}}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} (1 - p_{\hat{k}})^c$, is non-increasing and geometric in c, sandwiched between $\frac{\Delta_{\max}}{\hat{K}}(1 - p_{\min})^c$ and $\Delta_{\max}(1 - p_{\min})^c$ ($\Delta_{\max}(1 - p)^c$ in the homogeneous case). The excess risk therefore converges to the floor $\Delta_{\max}\,\bar{\pi} + \varepsilon\big(1 - \bar{\pi}\big) + \frac{2\Delta_{\max}}{\Delta(c)} \big(\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}}) + \epsilon_{\mathrm{data}}(\hat{N})\big)$, and enlarging the dock system entails no disadvantage in the bound provided $\Delta(c)$ stays bounded away from zero.*

Remark 3. *Corollary 3 shows that enlarging the dock system monotonically improves the upper bound until the floor is approached: the reuse accuracy rises with c and then saturates instead of degrading, which is consistent with the phenomenon observed in Fig. 1(d).*

5 EXPERIMENTS

In this section, we evaluate EGS for learnware identification and reuse on image and text benchmarks. We first introduce the datasets, baselines, and experimental settings, and then report results under three reuse scenarios. We further present ablation studies and additional analyses to examine the robustness, effectiveness, and limitations of EGS. Additional experimental details, complete results, and further analyses are provided in Appendix D.

5.1 EXPERIMENT SETUP

Dataset. We evaluate EGS on twelve image and text classification benchmarks, including representative datasets such as CIFAR-100 (Krizhevsky & Hinton, 2009), Food101 (Bossard et al., 2014), DTD (Cimpoi et al., 2014), Amazon (Ni et al., 2019), and TREC (Li & Roth, 2002). These benchmarks span diverse visual and textual classification scenarios with varying semantic granularity, class scales, and task characteristics. Full dataset details are provided in Appendix D.

Compared Methods. We compare EGS with representative learnware specification baselines, including RKME (Wu et al., 2023), CRKME (Liu et al., 2025b), LANE (Chen et al., 2025b), and DALI (Chen et al., 2025a). We also report *MAX* and *Oracle* as non-deployable manual reference points based on downstream performance. Full details are provided in Appendix D.2.

Experimental Configurations. We use predefined superclass-based training/test splits, with training superclasses defining developer learnware tasks and test superclasses defining user tasks. For each dataset and scenario, we sample 100 user tasks to evaluate performance. All methods share the same feature extractor—ResNet110 for images and Sentence Transformer for text—and the same specification size. Full experimental details are provided in Appendix D.3.

5.2 PERFORMANCES

We evaluate EGS under three learnware reuse scenarios: *Beyond Original Functionality*, *Within Original Functionality*, and *Few-Shot Beyond Original Functionality*. Under *Beyond Original Functionality* (Table 1), EGS achieves the best reuse average accuracy (0.710) and rank (1.46), outperforming DALI by 7.6 percentage points and matching Oracle on 9 of 12 datasets. Consistent performance is also observed under *Within Original Functionality* (Table 4; 0.707 accuracy, rank 2.00) and *Few-Shot Beyond Original Functionality* (Table 5; 0.686 accuracy, rank 1.29), demonstrating the strong average performance and robustness of EGS across different reuse settings. Additional performance analysis is provided in Appendix D.

Table 1: Performance comparison under the *Beyond Original Functionality* setting. Results are reuse classification accuracy, reported as mean ± standard deviation over sampled user tasks. For each dataset, the best mean among non-oracle entries is highlighted in bold and underlined.

User Task	*Learnware Identification and Reuse*					*Manual References*	
Dataset	**RKME**	**CRKME**	**LANE**	**DALI**	**EGS**	**MAX**	**Oracle**
CIFAR-100	0.472 ± 0.116	0.871 ± 0.065	0.858 ± 0.077	$\underline{\mathbf{0.894 \pm 0.035}}$	$\underline{\mathbf{0.894 \pm 0.035}}$	0.500 ± 0.114	0.894 ± 0.035
Food101	0.206 ± 0.092	0.608 ± 0.191	0.381 ± 0.173	$\underline{\mathbf{0.813 \pm 0.049}}$	$\underline{\mathbf{0.813 \pm 0.049}}$	0.440 ± 0.099	0.813 ± 0.049
DTD	0.210 ± 0.103	0.103 ± 0.142	0.445 ± 0.212	$\underline{\mathbf{0.808 \pm 0.065}}$	$\underline{\mathbf{0.808 \pm 0.065}}$	0.445 ± 0.090	0.808 ± 0.065
Cars	0.059 ± 0.057	0.055 ± 0.070	0.107 ± 0.069	0.203 ± 0.090	0.199 ± 0.093	$\underline{\mathbf{0.243 \pm 0.085}}$	0.402 ± 0.086
SUN397	0.162 ± 0.074	0.130 ± 0.067	0.371 ± 0.097	$\underline{\mathbf{0.503 \pm 0.094}}$	0.502 ± 0.106	0.328 ± 0.090	0.546 ± 0.085
GTSRB	0.513 ± 0.167	0.976 ± 0.041	0.949 ± 0.067	$\underline{\mathbf{0.997 \pm 0.002}}$	$\underline{\mathbf{0.997 \pm 0.002}}$	0.629 ± 0.149	0.997 ± 0.002
20Newsgroups	0.434 ± 0.084	0.781 ± 0.099	0.809 ± 0.094	0.649 ± 0.172	$\underline{\mathbf{0.860 \pm 0.032}}$	0.501 ± 0.101	0.860 ± 0.032
Amazon	0.489 ± 0.121	$\underline{\mathbf{0.856 \pm 0.042}}$	0.843 ± 0.056	0.845 ± 0.061	$\underline{\mathbf{0.856 \pm 0.042}}$	0.489 ± 0.099	0.856 ± 0.042
Bugs	0.218 ± 0.102	0.321 ± 0.199	0.407 ± 0.197	0.249 ± 0.160	$\underline{\mathbf{0.534 \pm 0.150}}$	0.364 ± 0.135	0.535 ± 0.150
Coronavirus	0.333 ± 0.122	0.517 ± 0.095	0.437 ± 0.186	0.407 ± 0.118	$\underline{\mathbf{0.669 \pm 0.050}}$	0.433 ± 0.059	0.669 ± 0.050
TREC	0.380 ± 0.146	0.375 ± 0.262	0.639 ± 0.147	0.566 ± 0.221	$\underline{\mathbf{0.694 \pm 0.108}}$	0.503 ± 0.124	0.694 ± 0.108
WoS	0.415 ± 0.087	0.640 ± 0.077	0.664 ± 0.065	0.672 ± 0.069	$\underline{\mathbf{0.692 \pm 0.052}}$	0.440 ± 0.105	0.692 ± 0.052
Avg. Metric (↑)	0.324	0.519	0.576	0.634	0.710	0.443	0.731
Avg. Rank (↓)	5.63	3.96	3.29	2.58	1.46	4.08	–
EGS (win/tie/loss)	12/0/0	11/1/0	12/0/0	6/4/2	–	11/0/1	0/9/3

5.3 ABLATION STUDY

We conduct ablation studies to examine two key aspects of EGS: robustness to model quality and sensitivity to the proxy set size.

Model Quality. We evaluate whether EGS effectively captures model-specific predictive utility, thereby supporting robust learnware identification and reuse while automatically filtering out low-quality learnwares. For each normal learnware associated with a high-quality model, we construct a corrupted counterpart by deliberately degrading its training procedure and predictive performance, and compare system performance before and after adding these corrupted learnwares to the dock system. Tables 2, 6 and 7 report results across the three reuse settings. EGS consistently iden-

tifies compatible learnwares despite the presence of low-quality distractors, although limited user supervision makes such quality perturbations more difficult to distinguish.

Table 2: Performance of EGS and baselines under the *Beyond original functionality* setting with corrupted learnwares. For corrupted settings, the value in parentheses denotes the change relative to the corresponding normal setting (green: improvement; red: degradation).

	RKME		CRKME		LANE		DALI		EGS		Oracle
Dataset	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	
Image Benchmarks											
CIFAR-100	0.472	0.416 (-0.056)	0.871	0.828 (-0.043)	0.858	0.795 (-0.063)	0.894	0.894 (0.000)	0.894	0.894 (0.000)	0.894
Food101	0.206	0.236 (+0.030)	0.608	0.514 (-0.094)	0.381	0.361 (-0.020)	0.813	0.813 (0.000)	0.813	0.813 (0.000)	0.813
DTD	0.210	0.168 (-0.041)	0.103	0.105 (+0.002)	0.445	0.396 (-0.049)	0.808	0.808 (0.000)	0.808	0.807 (-0.001)	0.808
Cars	0.059	0.037 (-0.022)	0.055	0.018 (-0.037)	0.107	0.083 (-0.024)	0.203	0.186 (-0.017)	0.199	0.213 (+0.014)	0.402
SUN397	0.162	0.131 (-0.032)	0.130	0.081 (-0.048)	0.371	0.325 (-0.046)	0.503	0.483 (-0.019)	0.502	0.498 (-0.004)	0.546
GTSRB	0.513	0.380 (-0.133)	0.976	0.944 (-0.032)	0.949	0.890 (-0.059)	0.997	0.996 (-0.001)	0.997	0.996 (-0.001)	0.997
Text Benchmarks											
20Newsgroups	0.434	0.307 (-0.127)	0.781	0.608 (-0.172)	0.809	0.732 (-0.077)	0.649	0.860 (+0.211)	0.860	0.860 (0.000)	0.860
Amazon	0.489	0.274 (-0.215)	0.856	0.691 (-0.166)	0.843	0.805 (-0.038)	0.845	0.806 (-0.039)	0.856	0.856 (0.000)	0.856
Bugs	0.218	0.153 (-0.065)	0.321	0.229 (-0.092)	0.407	0.399 (-0.008)	0.249	0.502 (+0.252)	0.534	0.528 (-0.006)	0.535
Coronavirus	0.333	0.324 (-0.009)	0.517	0.184 (-0.333)	0.437	0.297 (-0.140)	0.407	0.317 (-0.090)	0.669	0.669 (0.000)	0.669
TREC	0.380	0.327 (-0.053)	0.375	0.202 (-0.173)	0.639	0.495 (-0.144)	0.566	0.665 (+0.099)	0.694	0.694 (0.000)	0.694
WoS	0.415	0.310 (-0.105)	0.640	0.358 (-0.282)	0.664	0.625 (-0.039)	0.672	0.692 (+0.020)	0.692	0.692 (0.000)	0.692

Proxy Sample Size. We further confirm that EGS is robust to the proxy sample set size for learnware identification and reuse. To avoid exposing private training data or their gradients, EGS optimizes a compact proxy sample set $\mathcal{S}$ and uses its class-wise gradient responses to sketch model capabilities or user task requirements in the shared one-step energy gradient space. As summarized in Figs. 2 and 3, the proxy set size has a limited influence on the characterization and comparison of EGS, resulting in consistently stable reuse compatibility.

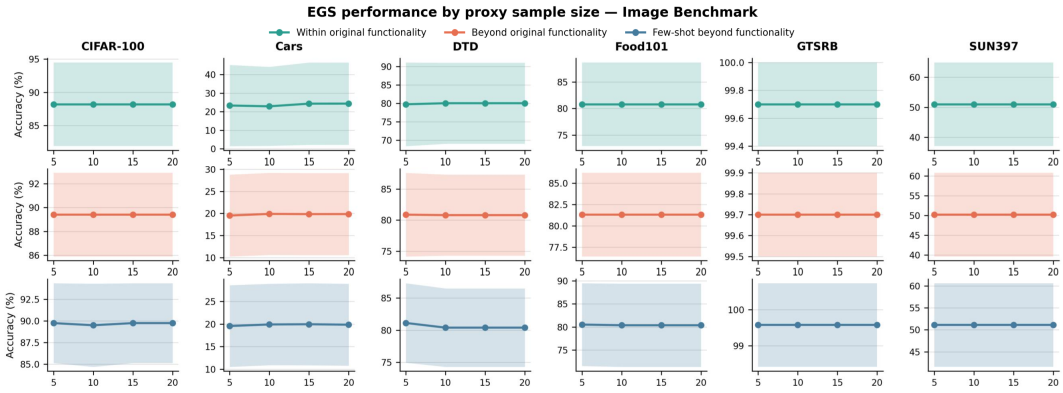


Figure 2: Effect of proxy sample size on the performance of EGS in the image benchmark.

6 CONCLUSION

In this paper, we investigate how learnware specifications can compactly characterize model capabilities under tight storage budgets while reliably reflecting reuse compatibility, and propose **E**nergy **G**radient **S**pecification (EGS) for this purpose. EGS characterizes and compares well-established models and user tasks in a shared one-step energy gradient space by optimizing compact proxy sets to align their gradient responses and using the resulting gradients as specifications. Our theoretical analysis attributes the performance gains from a growing dock system, as well as their plateau, to the interplay between dock system coverage and specification fidelity. Extensive experiments on image and text benchmarks demonstrate the effectiveness, superiority, and robustness of EGS across multiple learnware identification and reuse scenarios. In future work, we will investigate more efficient proxy optimization, formal privacy guarantees, broader task settings beyond classification, and more transferable shared representations to reduce reliance on specific feature extractors and facilitate more efficient learnware identification and reuse.

AI USE STATEMENT

Generative AI tools were used solely to aid and polish the writing of this paper. All AI-assisted text was reviewed and revised by the authors for accuracy, clarity, and consistency with the underlying research. The authors take full responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

We provide the mathematical formulation and implementation details of EGS in Section 3 and Appendix B. The assumptions, formal statements, and proofs supporting the theoretical analysis are given in Section 4 and Appendix C. The datasets, task construction, compared methods, model configurations, and evaluation protocols are documented in Section 5.1 and Appendix D. These materials specify the information needed to reproduce the reported experiments and verify the theoretical results. The implementation of EGS used in this work is provided in the **Supplementary Material** for review, and the core code will be publicly released upon publication.

REFERENCES

Jimmy Ba, Murat A. Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. In *Advances in Neural Information Processing Systems 35*, pp. 37932–37946, New Orleans, USA, 2022.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *Proceedings of 13th European Conference on Computer Vision*, pp. 446–461, Zurich, Switzerland, 2014.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A. Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10708–10717, New Orleans, USA, 2022.

Wei Chen, Jun-Xiang Mao, Xiaozheng Wang, and Min-Ling Zhang. Learnware specification via dual alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 8683–8699, Vancouver, Canada, 2025a.

Wei Chen, Jun-Xiang Mao, and Min-Ling Zhang. Learnware specification via label-aware neural embedding. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence*, pp. 15857–15865, Philadelphia, USA, 2025b.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3606–3613, Columbus, USA, 2014.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.

Lan-Zhe Guo, Zhi Zhou, Yu-Feng Li, and Zhi-Hua Zhou. Identifying useful learnwares for heterogeneous label spaces. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 12122–12131, Honolulu, USA, 2023.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Kamran Kowsari, Donald E. Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S. Gerber, and Laura E. Barnes. HDLTex: Hierarchical deep learning for text classification. In *Proceedings of the 16th IEEE International Conference on Machine Learning and Applications*, pp. 364–371, Cancun, Mexico, 2017.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 554–561, Sydney, Australia, 2013.

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

Ken Lang. NewsWeeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 331–339, San Francisco, USA, 1995.

Jae Yong Lee, Sungmin Kang, Juyeon Yoon, and Shin Yoo. The github recent bugs dataset for evaluating LLM-based debugging applications. In *Proceedings of the IEEE Conference on Software Testing, Verification and Validation*, pp. 442–444, Toronto, Canada, 2024.

Hao-Yi Lei, Zhi-Hao Tan, and Zhi-Hua Zhou. On the ability of developers' training data preservation of learnware. In *Advances in Neural Information Processing Systems 37*, pp. 36471–36513, Vancouver, Canada, 2024.

Hao-Yi Lei, Zhi-Hao Tan, and Zhi-Hua Zhou. A statistical framework for analyzing specification resistance to learnware-inversion risks. In *Proceedings of the 43rd International Conference on Machine Learning*, Seoul, South Korea, 2026a.

Hao-Yi Lei, Jin-Hui Wu, Zhi-Hao Tan, and Zhi-Hua Zhou. On the privacy-preserving capabilities of PAVE specifications in learnware. In *Proceedings of the 35th International Joint Conference on Artificial Intelligence*, pp. 4465–4473, Bremen, Germany, 2026b.

Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics*, volume 1, pp. 1–7, Taipei, Taiwan, 2002.

Hai-Tian Liu, Peng Tan, Jian-Dong Liu, Zhi-Hao Tan, and Zhi-Hua Zhou. Integrated learnware identification and reuse via reusability-aware metric learning. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Jeju, South Korea, 2026.

Jian-Dong Liu, Zhi-Hao Tan, and Zhi-Hua Zhou. Towards making learnware specification and market evolvable. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, pp. 13909–13917, Vancouver, Canada, 2024.

Jian-Dong Liu, Zhi-Hao Tan, and Zhi-Hua Zhou. Dynamic learnware filtering for efficient learnware identification and system slimming. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1811–1822, Toronto, Canada, 2025a.

Jian-Dong Liu, Zhi-Hao Tan, and Zhi-Hua Zhou. Identifying and reusing learnwares across different label spaces. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence*, pp. 5734–5742, Montreal, Canada, 2025b.

Colin McDiarmid et al. On the method of bounded differences. *Surveys in combinatorics*, 141(1): 148–188, 1989.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 188–197, Hong Kong, China, 2019.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, pp. 1177–1184, Vancouver, Canada, 2007.

Hao-Yu Shi, Zhi-Hao Tan, Zi-Chen Zhao, Yang Yu, and Zhi-Hua Zhou. A study on PAVE specification for learnware. In *The 14th International Conference on Learning Representations*, Rio de Janeiro, Brazil, 2026.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. Man vs. Computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332, 2012.

Peng Tan, Zhi-Hao Tan, Yuan Jiang, and Zhi-Hua Zhou. Handling learnwares developed from heterogeneous feature spaces without auxiliary data. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, pp. 4235–4243, Macao, China, 2023.

Peng Tan, Hai-Tian Liu, Zhi-Hao Tan, and Zhi-Hua Zhou. Handling learnwares from heterogeneous feature spaces with explicit label exploitation. In *Advances in Neural Information Processing Systems 37*, pp. 12767–12795, Vancouver, Canada, 2024a.

Peng Tan, Zhi-Hao Tan, Yuan Jiang, and Zhi-Hua Zhou. Towards enabling learnware to handle heterogeneous feature spaces. *Machine Learning*, 113(4):1839–1860, 2024b.

Peng Tan, Feifan Yang, Zhi-Hao Tan, and Zhi-Hua Zhou. Tabular learnwares can be repurposed for seemingly irrelevant new tasks. In *Proceedings of the 40th AAAI Conference on Artificial Intelligence*, pp. 25778–25786, Singapore, 2026.

Zhi-Hao Tan, Jian-Dong Liu, Xiao-Dong Bi, Peng Tan, Qin-Cheng Zheng, Hai-Tian Liu, Yi Xie, Xiao-Chuan Zou, Yang Yu, and Zhi-Hua Zhou. Beimingwu: A learnware dock system. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5773–5782, Barcelona, Spain, 2024c.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*, Online, 2020.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Xi-Zhu Wu, Wenkai Xu, Song Liu, and Zhi-Hua Zhou. Model reuse with reduced kernel mean embedding specification. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):699–710, 2023.

Jianxiong Xiao, Krista A. Ehinger, James Hays, Antonio Torralba, and Aude Oliva. SUN database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119 (1):3–22, 2016.

Yi Xie, Zhi-Hao Tan, Yuan Jiang, and Zhi-Hua Zhou. Identifying helpful learnwares without examining the whole market. In *Proceedings of the 26th European Conference on Artificial Intelligence*, volume 372, pp. 2752–2759, Kraków, Poland, 2023.

Yi-Fan Zhang, Xiao Zhang, and Min-Ling Zhang. Generalization analysis for controllable learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 77017–77043, Vancouver, Canada, 2025.

Yu-Jie Zhang, Yu-Hu Yan, Peng Zhao, and Zhi-Hua Zhou. Towards enabling learnware to handle unseen jobs. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, pp. 10964–10972, Virtual Event, 2021.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6514–6523, 2023.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *Proceedings of the 9th International Conference on Learning Representations*, Virtual Event, 2021.

Zhi-Hua Zhou. Learnware: on the future of machine learning. *Frontiers of Computer Science*, 10 (4):589–590, 2016.

Zhi-Hua Zhou and Zhi-Hao Tan. Learnware: Small models do big. *Science China Information Sciences*, 67(1):112102, 2024.

APPENDIX

A EXPERIMENTAL DETAILS AND ADDITIONAL RESULTS FOR FIG. 1

This section provides the experimental setups and additional results for Fig. 1. Fig. 1(a–c) examine how different specifications preserve task-relevant information under limited specification sizes, while Fig. 1(d) evaluates how reuse performance changes as the learnware dock system expands.

A.1 SPECIFICATION UTILITY UNDER COMPRESSION

Experimental Setup. We conduct the specification study on the 20 superclass tasks of CIFAR-100, each containing five subclasses. For each superclass, we use three random seeds, resulting in 60 task–seed runs. In each run, we sample 100 examples per class from the official CIFAR-100 test split. Of these, 70 examples per class are used to construct the specification, and the remaining 30 examples per class form a held-out evaluation set.

We compare three specification methods: marginal distribution specification matching (RKME) Wu et al. (2023), class-wise distribution specification matching (CRKME) Liu et al. (2025b), and the proposed class-wise one-step gradient matching (EGS). Let $l \in \{1, 2, 5, 10\}$ denote the number of retained samples per class. All methods use the same total representation budget of $5 \times l$ prototypes for each five-class superclass task. To quantify the predictive information retained by a specification, we train a linear probe on frozen image features using only its compact representation and evaluate the probe on the held-out examples. We report its accuracy relative to that of an identically trained probe using the complete specification-construction set. Error bars represent 95% confidence intervals over the 60 task–seed runs.

Effect of class-conditional information. Fig. 1(a) shows that class-conditional information is particularly important under extreme compression. With only one retained sample per class ($l = 1$), marginal distribution specification matching preserves 53.6% of the full-data probe performance, whereas class-wise distribution specification matching preserves 81.7%. The proposed class-wise one-step gradient specification further increases the retained utility to 87.2%.

Effect of one-step gradient matching. As shown in Fig. 1(b), the proposed class-wise one-step gradient specification attains the highest numerical mean retained utility across all evaluated specification sizes. This result suggests that class-wise one-step gradient matching preserves learning-relevant information beyond that captured by marginal or class-conditional feature statistics alone.

Relationship between distribution error and utility. Fig. 1(c) examines whether distribution-matching error serves as a reliable indicator of downstream utility. Distribution error is negatively associated with retained utility ($\rho = -0.38$, $p = 9.2 \times 10^{-26}$), but the association is only moderate. In particular, specifications with similar distribution errors can exhibit substantially different downstream performance. Thus, distribution fidelity alone does not fully characterize the learning-relevant information retained by a compact specification.

A.2 SCALABILITY OF THE LEARNWARE DOCK SYSTEM

Experimental Setup. We next examine how reuse performance scales with the coverage of the learnware dock system. Each learnware is constructed from a random five-class subset of the CIFAR-100 training split, with 200 source examples per class, and represented by class-wise one-step energy gradients using five proxy samples per class. We progressively increase the dock size from 1 to 168 learnwares and evaluate the resulting reuse performance. The specification used in this study corresponds to EGS, which is formally introduced in the methodology.

We consider two user settings: In the *in-dock system* setting, user tasks are sampled from a disjoint 10% holdout of the CIFAR-100 training split. In the *out-of-dock system* setting, user tasks are sampled exclusively from the official CIFAR-100 test split. The source data used to construct learnwares exclude all examples used by either type of user task. At each dock size, we evaluate 30 sampled five-class user tasks, obtained from 10 independently sampled tasks over three repetitions, with 50 user examples per class.

Results. Fig. 1(d) shows that reuse accuracy increases rapidly as the dock system expands and gradually approaches saturation. In the in-dock setting, accuracy increases from 8.6% with one learnware to 98.8% with 168 learnwares, surpassing the 93.9% accuracy of the native 100-way ResNet110 checkpoint once the dock system contains 64 learnwares. In the out-of-dock setting, accuracy increases from 2.5% to 95.5% and surpasses the corresponding direct-model baseline of 74.2% once the dock system contains 32 learnwares. These results are consistent with the scalability hypothesis: expanding a diverse learnware dock system increases its coverage and thereby improves the likelihood of identifying class-relevant learnwares for previously unseen user tasks.

B Algorithmic Implementation of EGS in the Learnware Paradigm

This section presents the algorithmic workflow of EGS, covering both stages of the learnware paradigm: learnware specification construction during developer-oriented submitting, and learnware identification and reuse during user-oriented deployment.

B.1 Algorithm for Constructing EGS

Algorithm 1 summarizes the procedure for constructing an EGS in the learnware paradigm. Given a task $\mathcal{D}$ and a shared network $\phi_{\boldsymbol{\xi}}$, the dock system first initializes a compact proxy set $\mathcal{S}$ with M samples (Line 1). Then, it iteratively optimizes the proxy set to match the class-wise one-step energy gradients of the original task in the shared gradient space (Lines 2-7). After T iterations, the resulting class-wise one-step energy gradients of the optimized proxy set are stored as the specification Φ^{EGS} (Line 8).

Algorithm 1 Energy Gradient Specification (EGS)

Require: Task data $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$, shared network $\phi_{\boldsymbol{\xi}} = g_{\boldsymbol{w},\boldsymbol{\eta}} \circ \psi_{\boldsymbol{\vartheta}}$, proxy sample size $M = K \times l$, and optimization iterations T
Ensure: Specification Φ^{EGS}
1: Initialize proxy set $\mathcal{S} = \{(\boldsymbol{s}_m, y_{\boldsymbol{s}_m})\}_{m=1}^{M}$ and $y_{\boldsymbol{s}_m} \in \{1, \ldots, K\}$
2: **for** $t = 1$ to T **do**
3: Randomly initialize $\boldsymbol{w}$, b, and $\boldsymbol{\eta}$
4: Compute $\{\boldsymbol{G}_{\mathcal{D}^k}\}_{k=1}^{K}$ of $\mathcal{D}$ according to Eq. 2
5: Compute $\{\boldsymbol{G}_{\mathcal{S}^k}\}_{k=1}^{K}$ of $\mathcal{S}$ according to Eq. 4
6: Update proxy set $\mathcal{S} \leftarrow \mathcal{S} - \alpha \nabla_{\boldsymbol{s}} \mathcal{L}_{\text{EGS}}$ by Eq. 3
7: **end for**
8: specification $\Phi^{\text{EGS}} = \{\boldsymbol{G}_{\mathcal{S}^k}\}_{k=1}^{K}$

B.2 Algorithm of the Developer-Oriented Submitting Stage

As shown in Algorithm 2, the developer-oriented submitting stage consists of three steps. First, the submitted well-established model is evaluated on the private developer data to obtain model-induced predictive targets (Line 1). Second, Algorithm 1 constructs the learnware specification by optimizing a compact proxy set to match the class-wise energy gradients jointly induced by the developer data and model predictions in the shared gradient space (Line 2). Finally, the resulting specification is stored together with the well-established model as a learnware for subsequent identification and reuse, without retaining the private developer data (Line 3).

Algorithm 2 Developer-oriented submission stage with EGS in the learnware paradigm

Require: Developer local training task $\mathcal{D}_i = \{(\boldsymbol{x}_{i,n}, y_{i,n})\}_{n=1}^{N_i}$ with K classes, well-established model $f_{\boldsymbol{\theta}_i}$, proxy sample size l, and optimization iterations T
Ensure: Learnware $\{(f_{\boldsymbol{\theta}_i}, \Phi_i^{\text{EGS}})\}$
1: Evaluate $f_{\boldsymbol{\theta}_i}(\boldsymbol{x})$ for each $\boldsymbol{x} \in \mathcal{D}_i$
2: $\Phi_i^{\text{EGS}} \leftarrow \text{EGS}(\{(\boldsymbol{x}_{i,n}, f_{\boldsymbol{\theta}_i}(\boldsymbol{x}_{i,n}))\}_{n=1}^{N_i}, K \times l, T)$ using Algorithm 1
3: Store learnware $\{(f_{\boldsymbol{\theta}_i}, \Phi_i^{\text{EGS}})\}$ in the learnware dock system

B.3 Algorithm of the User-Oriented Deploying Stage

As shown in Algorithm 3, the user-oriented deploying stage consists of four steps. First, the dock system constructs the user specification from the labeled user task using Algorithm 1, which optimizes a compact proxy set to match the class-wise energy gradients of the user task in the shared gradient space (Line 1). Second, for each learnware, the dock system computes the pairwise distances between the class-wise user and learnware specifications and solves the corresponding class-wise matching problem (Lines 2–5). Third, it aggregates the matched candidates across learnwares and selects, for each user class, the learnware class with the smallest specification distance, thereby constructing the class-wise correspondence $\mathcal{M}$ and the set of selected learnwares $\mathcal{I}$ (Lines 6–14). Finally, the dock system retrieves the selected models and applies the reuse operator $\mathcal{R}$ according to $\mathcal{M}$ to obtain the reused predictor $\hat{f}$ (Lines 15–17).

Algorithm 3 User-oriented deploying stage with EGS in the learnware paradigm

Require: Labeled user task $\hat{\mathcal{D}} = \{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}})\}_{\hat{n}=1}^{\hat{N}}$, proxy sample size l, iterations T, learnware dock system $\{(f_{\boldsymbol{\theta}_i}, \Phi_i^{\text{EGS}})\}_{i=1}^{c}$

Ensure: Selected learnwares $\{f_{\boldsymbol{\theta}_i}\}_{i\in\mathcal{I}}$, class-wise correspondence $\mathcal{M}$, and reused predictor $\hat{f}$

1: $\hat{\Phi}^{\text{EGS}} \leftarrow \text{EGS}(\{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}})\}_{\hat{n}=1}^{\hat{N}}, \hat{K} \times l, T)$ using Algorithm 1
2: **for** $i = 1, \ldots, c$ **do**
3: $\quad D_i^{\hat{k},k} \leftarrow \left\| \hat{\Phi}^{\hat{k}} - \Phi_i^{k} \right\|_F^2$ for all $(\hat{k}, k) \in [\hat{K}] \times [K_i]$
4: $\quad$ Obtain correspondence between user specification and learnware specification $\boldsymbol{E}_i^*$ by Eq. 5
5: **end for**
6: $\mathcal{M} \leftarrow \emptyset$
7: **for** each user class $\hat{k}$ **do**
8: $\quad \mathcal{C}^{\hat{k}} \leftarrow \{(i,k) : e_i^{\hat{k},k} = 1 \mid i \in [c],\ k \in [K_i],\ e_i^{\hat{k},k} \in E_i^*\}$
9: $\quad$ **if** $\mathcal{C}^{\hat{k}} \neq \emptyset$ **then**
10: $\quad\quad (\hat{k}, i, k) \leftarrow \arg\min_{(i,k)\in\mathcal{C}^{\hat{k}}} D_i^{\hat{k},k}$
11: $\quad\quad \mathcal{M} \leftarrow \mathcal{M} \cup \{(\hat{k}, i, k)\}$
12: $\quad$ **end if**
13: **end for**
14: $\mathcal{I} \leftarrow \{i : \exists(\hat{k}, i, k) \in \mathcal{M} \mid \hat{k} \in [\hat{K}],\ i \in [c],\ k \in [K_i]\}$
15: retrieve suitable learnware models $\{f_{\boldsymbol{\theta}_i}\}_{i\in\mathcal{I}}$
16: $\hat{f} \leftarrow \mathcal{R}(\{f_{\boldsymbol{\theta}_i}\}_{i\in\mathcal{I}}, \mathcal{M})$ ▷ *class-wise routing and reuse*
17: **return** $\{f_{\boldsymbol{\theta}_i}\}_{i\in\mathcal{I}}$, $\mathcal{M}$, and $\hat{f}$

C Theoretical Analysis

C.1 Outline

Here we give the detailed proofs of the theoretical results in Section 4. Our main proofs include:

- The definitions of the complexity measures used in the analysis (Subsection C.2).
- The bound for the reuse loss function space via McDiarmid's inequality and symmetrization (Subsection C.2).
- The additive entropy cost of enlarging the dock system (Lemma 7).
- The coordinate-wise contraction of the reuse loss (Lemma 9).
- The capacity-based generalization bound for learnware reuse with a logarithmic dependency on the learnware size (Theorem 1).
- The excess risk decomposition and the dock system scaling law (Theorem 2 and Corollary 3).

C.2 PRELIMINARIES

Complexity measures. We introduce the complexity measures used throughout the proofs. Let $\mathcal{F}$ be a class of real-valued functions mapping from $\mathcal{X}$ to $\mathbb{R}$, and let $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ be a set with n i.i.d. samples.

Definition 1 (Rademacher complexity)**.** *The empirical Rademacher complexity of $\mathcal{F}$ over $\mathcal{D}$ is*

$$\hat{\mathfrak{R}}_{\mathcal{D}}(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\epsilon}}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \epsilon_i f(\boldsymbol{x}_i)\right], \tag{9}$$

where $\epsilon_1, \ldots, \epsilon_n$ are i.i.d. Rademacher random variables. The worst-case Rademacher complexity is $\tilde{\mathfrak{R}}_n(\mathcal{F}) = \sup_{\mathcal{D}\in\mathcal{X}^n} \hat{\mathfrak{R}}_{\mathcal{D}}(\mathcal{F})$.

Bound for the reuse loss function space. According to McDiarmid's inequality (McDiarmid et al., 1989) and the symmetrization technique (e.g., Theorem 4.4 in (Mohri et al., 2018)), for any user dataset $\hat{\mathcal{D}} = \{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}}) : \hat{n} \in [\hat{N}]\}$ and $\ell_{\mathrm{W}}(\cdot,\cdot) \le M$, with probability at least $1-\delta$, the following holds for any $f \in \mathcal{F}_{\mathrm{LW}}$:

$$\mathcal{R}_{\mathrm{LW}}(f) \;\le\; \hat{\mathcal{R}}_{\hat{\mathcal{D}},\mathrm{LW}}(f) + 2\,\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}\big(\mathcal{L}_{\mathrm{LW}}\big) + 3M\sqrt{\frac{\log\frac{2}{\delta}}{2\hat{N}}}. \tag{10}$$

The bounded differences condition is satisfied since replacing one user example changes $\ell_{\mathrm{W}}(f(\hat{\boldsymbol{x}}_{\hat{n}}), \hat{y}_{\hat{n}})$ by at most M, so changing the whole sample changes the empirical risk by at most $M/\hat{N}$.

Auxiliary lemmas. We will repeatedly use the following standard results. Throughout, a random variable Z is σ-sub-Gaussian if $\mathbb{E}[\exp(\lambda(Z - \mathbb{E}Z))] \le \exp(\lambda^2\sigma^2/2)$ for all $\lambda \in \mathbb{R}$.

Lemma 4 (Maximum inequality for sub-Gaussian variables, e.g. Theorem 2.5 in (Boucheron et al., 2013))**.** *Let $Z_1, \ldots, Z_P$ be σ-sub-Gaussian random variables. Then*

$$\mathbb{E}\left[\max_{p\in[P]} Z_p\right] \;\le\; \sigma\sqrt{2\log P}. \tag{11}$$

Proof. For any $\lambda > 0$, Jensen's inequality and sub-Gaussianity give

$$\exp\!\Big(\lambda\,\mathbb{E}\Big[\max_p Z_p\Big]\Big) \;\le\; \mathbb{E}\Big[\exp\Big(\lambda \max_p Z_p\Big)\Big] \;\le\; \sum_{p=1}^{P}\mathbb{E}\big[e^{\lambda Z_p}\big] \;\le\; P\,e^{\lambda^2\sigma^2/2}.$$

Taking logarithms and dividing by λ yields $\mathbb{E}[\max_p Z_p] \le \frac{\log P}{\lambda} + \frac{\lambda\sigma^2}{2}$, which is minimized at $\lambda = \sqrt{2\log P}/\sigma$ and gives Eq. 11. □

Lemma 5 (Talagrand's lemma, Lemma 5.7 in (Mohri et al., 2018))**.** *Let $\ell : \mathbb{R} \to \mathbb{R}$ be μ-Lipschitz continuous and let $\mathcal{G}$ be a class of real-valued functions. Then, for any finite sample $\mathcal{S}$ of size n,*

$$\hat{\mathfrak{R}}_{\mathcal{S}}\big(\ell \circ \mathcal{G}\big) \;\le\; \mu\,\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G}),$$

where $\ell\circ\mathcal{G} = \{s \mapsto \ell(g(s)) : g \in \mathcal{G}\}$. If in addition ℓ is multiplied by a bounded factor $0 \le v \le V$, the right-hand side becomes $V\mu\,\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})$.

Lemma 6 (Subadditivity of the empirical Rademacher complexity, cf. Exercise 3.8 in (Mohri et al., 2018))**.** *Let $\mathcal{G}_1, \ldots, \mathcal{G}_{\hat{K}}$ be classes of real-valued functions. Then, for any finite sample $\mathcal{S}$,*

$$\hat{\mathfrak{R}}_{\mathcal{S}}\Big(\sum_{\hat{k}=1}^{\hat{K}} \mathcal{G}_{\hat{k}}\Big) \;\le\; \sum_{\hat{k}=1}^{\hat{K}} \hat{\mathfrak{R}}_{\mathcal{S}}\big(\mathcal{G}_{\hat{k}}\big),$$

where $\sum_{\hat{k}} \mathcal{G}_{\hat{k}} = \{s \mapsto \sum_{\hat{k}} g_{\hat{k}}(s) : g_{\hat{k}} \in \mathcal{G}_{\hat{k}}\}$.

Proof. Immediate from the subadditivity of the supremum: $\sup_{(g_1,\ldots,g_{\hat{K}})} \sum_i \epsilon_i \sum_{\hat{k}} g_{\hat{k}}(\boldsymbol{x}_i) = \sum_{\hat{k}} \sup_{g_{\hat{k}}} \sum_i \epsilon_i g_{\hat{k}}(\boldsymbol{x}_i)$, since the $\hat{K}$ choices are independent. □

C.3 Proof of Lemma 7

Lemma 7. *Let $\mathcal{H} = \bigcup_{(i,k)\in\mathcal{P}} \mathcal{H}_{i,k}$ with $|\mathcal{P}| \le P$, let Assumption 1 hold, and let $\hat{\mathcal{D}}$ be any user dataset of size $\hat{N}$. Then*

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}) \;\le\; E\sqrt{\frac{2\log P}{\hat{N}}} \;\le\; E\sqrt{\frac{2\log(c\bar{K})}{\hat{N}}}. \tag{12}$$

For every $\hat{k} \in [\hat{K}]$, the same bound holds on any class-conditional subset:

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}(\mathcal{H}) \;\le\; E\sqrt{\frac{2\log P}{\hat{N}_{\hat{k}}}}. \tag{13}$$

Before proving Lemma 7, we first prove the following lemma, which is slightly more general: the sample on which the complexity is measured can be any subset of the user data and the member classes are arbitrary, and Lemma 7 follows as the singleton specialization.

Lemma 8. *Let $\mathcal{H} = \bigcup_{(i,k)\in\mathcal{P}} \mathcal{H}_{i,k}$ with $|\mathcal{P}| \le P$, and suppose $|h(\boldsymbol{x})| \le E$ for every $h \in \mathcal{H}$ and every $\boldsymbol{x}$. Let $\mathcal{S}$ be any finite sample of size m. Then*

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{H}) \;\le\; \max_{(i,k)\in\mathcal{P}} \hat{\mathfrak{R}}_{\mathcal{S}}\big(\mathcal{H}_{i,k}\big) + E\sqrt{\frac{2\log P}{m}}. \tag{14}$$

Proof. Fix the sample $\mathcal{S} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_m\}$, for each candidate pair $(i,k) \in \mathcal{P}$, we denote $A_{i,k}(\boldsymbol{\epsilon}) = \sup_{h\in\mathcal{H}_{i,k}} \frac{1}{m}\sum_{t=1}^{m} \epsilon_t h(\boldsymbol{x}_t)$, so that

$$\hat{\mathfrak{R}}_{\mathcal{S}}\big(\mathcal{H}_{i,k}\big) = \mathbb{E}_{\boldsymbol{\epsilon}}\big[A_{i,k}\big]. \tag{15}$$

Step 1: The supremum over the union is the maximum over its members. For every realization of $\boldsymbol{\epsilon}$,

$$\sup_{h\in\mathcal{H}} \frac{1}{m}\sum_t \epsilon_t h(\boldsymbol{x}_t) = \sup_{(i,k)\in\mathcal{P}} \sup_{h\in\mathcal{H}_{i,k}} \frac{1}{m}\sum_t \epsilon_t h(\boldsymbol{x}_t) = \max_{(i,k)\in\mathcal{P}} A_{i,k}(\boldsymbol{\epsilon}),$$

where the first equality uses $\mathcal{H} = \bigcup_{(i,k)} \mathcal{H}_{i,k}$ and the second uses that $\mathcal{P}$ is finite, so the outer supremum is attained. Taking expectations,

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\epsilon}}\Big[\max_{(i,k)\in\mathcal{P}} A_{i,k}(\boldsymbol{\epsilon})\Big]. \tag{16}$$

Note that the finiteness of $\mathcal{P}$ is used only in this step, while the individual classes $\mathcal{H}_{i,k}$ are allowed to be arbitrary.

Step 2: Centering. Denote each $A_{i,k}$ as its mean plus a centered fluctuation and use $\max_p(a_p + b_p) \le \max_p a_p + \max_p b_p$, we have

$$\max_{(i,k)\in\mathcal{P}} A_{i,k} = \max_{(i,k)\in\mathcal{P}} \Big(\mathbb{E}\big[A_{i,k}\big] + \big(A_{i,k} - \mathbb{E}\big[A_{i,k}\big]\big)\Big) \;\le\; \max_{(i,k)\in\mathcal{P}} \mathbb{E}\big[A_{i,k}\big] + \max_{(i,k)\in\mathcal{P}} \big(A_{i,k} - \mathbb{E}\big[A_{i,k}\big]\big). \tag{17}$$

Since the first term on the right-hand side is deterministic, taking expectations of Eq. 17 and invoking Eq. 16, then

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{H}) \;\le\; \max_{(i,k)\in\mathcal{P}} \mathbb{E}_{\boldsymbol{\epsilon}}\big[A_{i,k}\big] + \mathbb{E}_{\boldsymbol{\epsilon}}\Big[\max_{(i,k)\in\mathcal{P}} \big(A_{i,k} - \mathbb{E}[A_{i,k}]\big)\Big].$$

The first term equals $\max_{(i,k)} \hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{H}_{i,k})$ by Eq. 15.

Step 3: Each $A_{i,k}$ is $E/\sqrt{m}$-sub-Gaussian. Let $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}'$ differ only in the t-th coordinate. Note that a difference of suprema is at most the supremum of the difference, we have

$$A_{i,k}(\boldsymbol{\epsilon}) - A_{i,k}(\boldsymbol{\epsilon}') \;\le\; \sup_{h\in\mathcal{H}_{i,k}} \frac{1}{m}\sum_{u=1}^{m} (\epsilon_u - \epsilon'_u)\, h(\boldsymbol{x}_u) \;=\; \sup_{h\in\mathcal{H}_{i,k}} \frac{(\epsilon_t - \epsilon'_t)\, h(\boldsymbol{x}_t)}{m} \;\le\; \frac{2E}{m},$$

with $\epsilon_t, \epsilon'_t \in \{-1, +1\}$ and $|h(\boldsymbol{x}_t)| \le E$ for every $h \in \mathcal{H}_{i,k}$, the symmetric bound follows by swapping $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}'$, and hence $\left|A_{i,k}(\boldsymbol{\epsilon}) - A_{i,k}(\boldsymbol{\epsilon}')\right| \le \frac{2E}{m}$. Applying Azuma–Hoeffding's inequality to the Doob martingale of $A_{i,k}$ with respect to the successive revealings $\epsilon_1, \dots, \epsilon_m$, whose bounded differences are all at most $2E/m$, the centered variable $A_{i,k} - \mathbb{E}[A_{i,k}]$ is sub-Gaussian with parameter

$$\sigma^2 = \frac{1}{4}\sum_{t=1}^{m}\left(\frac{2E}{m}\right)^2 = \frac{E^2}{m}, \qquad \text{i.e.} \qquad \sigma = \frac{E}{\sqrt{m}}.$$

Note that this bound is uniform over $(i,k) \in \mathcal{P}$, since it only depends on the envelope E rather than any property of the individual class.

Step 4: The maximum inequality. Applying Lemma 4 to the P centered variables $\{A_{i,k} - \mathbb{E}[A_{i,k}]\}_{(i,k)\in\mathcal{P}}$, each of which is $\frac{E}{\sqrt{m}}$-sub-Gaussian by Step 3, we have

$$\mathbb{E}_{\boldsymbol{\epsilon}}\Big[\max_{(i,k)\in\mathcal{P}}\big(A_{i,k} - \mathbb{E}[A_{i,k}]\big)\Big] \;\le\; \frac{E}{\sqrt{m}}\sqrt{2\log P} = E\sqrt{\frac{2\log P}{m}}.$$

Finally, combining Steps 2 and 4, we have Eq. 14, which completes the proof. □

Proof of Lemma 7. Eq. 12 follows from Lemma 8 with $\mathcal{S} = \hat{\mathcal{D}}$ and $m = \hat{N}$: since each candidate class $\mathcal{H}_{i,k}$ is a singleton, $\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}_{i,k}) = 0$ holds for every (i,k) and the first term on the right-hand side of Eq. 14 vanishes, while $\log P \le \log(c\bar{K})$ holds by the definition of $\mathcal{P}$. Eq. 13 then follows by applying Lemma 8 to the class-conditional subset $\mathcal{S} = \hat{\mathcal{D}}_{\hat{k}}$ of size $m = \hat{N}_{\hat{k}}$ with the same argument, which completes the proof. □

Remark 4. *The finiteness of $\mathcal{P}$ is used at exactly one place in the proof of Lemma 8, i.e., in Step 1, where the outer supremum over the union becomes a maximum over P indices. The individual classes $\mathcal{H}_{i,k}$ influence the bound only through $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{H}_{i,k})$ and the uniform envelope E, so they are allowed to be arbitrary, in particular infinite. For the deployed predictor, they are singletons and the first term of Eq. 14 vanishes, so Lemma 7 reduces to Massart's finite-class lemma. The general form of Lemma 8 is retained since it preserves the property of additive-entropy split for any future deploying stage that fits a per-candidate read-out.*

C.4 Proof of Lemma 9

Lemma 9. *Let $\mathcal{F}_{\mathrm{LW}}$ be the learnware reuse function class and let Assumption 1 hold. Then, for any user dataset $\hat{\mathcal{D}}$ of size $\hat{N}$,*

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}\big(\mathcal{L}_{\mathrm{LW}}\big) \;\le\; C_\ell\,\mu V\,\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}) \;\le\; C_\ell\,\mu V E\sqrt{\frac{2\log(c\bar{K})}{\hat{N}}}, \tag{18}$$

where C_ℓ is the absolute constant of the scalar contraction inequality. The bound involves no factor depending on the number of user classes $\hat{K}$ or on the learnware size c beyond the logarithm.

Proof. Let $\hat{\mathcal{D}} = \{(\hat{\boldsymbol{x}}_{\hat{n}}, \hat{y}_{\hat{n}})\}_{\hat{n}=1}^{\hat{N}}$. For each user class $\hat{k} \in [\hat{K}]$, we define the coordinate loss class

$$\mathcal{G}_{\hat{k}} = \Big\{(\boldsymbol{x}, y) \mapsto v_{\hat{k}}(y)\,\ell_{\hat{k}}\big(h(\boldsymbol{x}), y\big) \;:\; h \in \mathcal{H}\Big\}, \tag{19}$$

hence, by the definitions of $\mathcal{F}_{\mathrm{LW}}$ and ℓ_{W},

$$\mathcal{L}_{\mathrm{LW}} = \Big\{(\boldsymbol{x}, y) \mapsto \frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}} g_{\hat{k}}(\boldsymbol{x}, y) \;:\; g_{\hat{k}} \in \mathcal{G}_{\hat{k}}\Big\} = \frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}}\mathcal{G}_{\hat{k}}. \tag{20}$$

The equality in Eq. 20 holds since routing π is unconstrained: for any choice of $h_{\pi(1)}, \dots, h_{\pi(\hat{K})} \in \mathcal{H}$ there exists an $f \in \mathcal{F}_{\mathrm{LW}}$ realizing it, so the $\hat{K}$ summands of ℓ_{W} range over $\mathcal{G}_1, \dots, \mathcal{G}_{\hat{K}}$ independently.

Applying Lemma 6 to Eq. 20, and using the homogeneity $\hat{\mathfrak{R}}_{\mathcal{S}}(\lambda\mathcal{G}) = \lambda\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})$ for $\lambda > 0$,

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{L}_{\mathrm{LW}}) = \hat{\mathfrak{R}}_{\hat{\mathcal{D}}}\Big(\frac{1}{\hat{K}}\sum_{\hat{k}}\mathcal{G}_{\hat{k}}\Big) \;\le\; \frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}}\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{G}_{\hat{k}}). \tag{21}$$

Fix $\hat{k}$, then for every label y, the map $s \mapsto v_{\hat{k}}(y)\,\ell_{\hat{k}}(s, y)$ is $v_{\hat{k}}(y)\mu$-Lipschitz by Assumption 1, and $v_{\hat{k}}(y) \le V$ by definition of ℓ_{W}, so it is $V\mu$-Lipschitz uniformly in y. Applying Lemma 5 pointwise in y and then taking the supremum over y on the left-hand side, we have

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{G}_{\hat{k}}) \;\le\; C_\ell\, V\mu\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}), \tag{22}$$

for every $\hat{k} \in [\hat{K}]$, where C_ℓ is the absolute constant of Lemma 5. Note that the right-hand side of Eq. 22 involves the *full* pool $\mathcal{H}$ rather than a single $\mathcal{H}_{i,k}$, since the function h in Eq. 19 ranges over all of $\mathcal{H}$, and the routing is not fixed in advance but is part of the supremum that defines the complexity.

Substituting Eq. 22 into Eq. 21, we have

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{L}_{\mathrm{LW}}) \;\le\; \frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}} C_\ell\, V\mu\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}) = C_\ell\, V\mu\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{H}),$$

in which the $\hat{K}$ identical summands are exactly cancelled by the $1/\hat{K}$ prefactor of ℓ_{W}. This gives the first inequality of Eq. 18, and the second follows by substituting Lemma 7. Since neither Eq. 21 nor Eq. 22 introduces any factor depending on $\hat{K}$, and C_ℓ, μ and V are independent of c and $\hat{N}$, Eq. 18 involves no such factor, which completes the proof. □

Remark 5. *The absence of a $\hat{K}$-dependent factor in Eq. 18 comes from two properties used in the aforementioned proof: the surrogate loss ℓ_{W} is* already an average *over the $\hat{K}$ user classes, so the $1/\hat{K}$ prefactor is available for free in Step 1 and cancels the $\hat{K}$ summands produced by subadditivity in Step 3, and the routing π is unconstrained, which makes Eq. 20 an equality of classes rather than an inclusion, so that no additional combinatorial cost is paid for the $\hat{K}$-fold choice. The unconstrained routing class is a superset of the assignments realizable by the partial one-to-one matching of Eq. 5 and the per-class selection of Algorithm 3, and enlarging to the superset keeps every uniform bound valid for the deployed predictor while making the coordinates independent.*

It is worth noting that the first property is specific to the surrogate loss we adopted *rather than a general characteristic of learnware reuse: replacing the averaged loss ℓ_{W} by the unnormalized sum $\sum_{\hat{k}} v_{\hat{k}}(y)\ell_{\hat{k}}(f_{\hat{k}}(\boldsymbol{x}), y)$ leaves the proof unchanged except that Step 1 no longer supplies a $1/\hat{K}$ factor, resulting in a bound* linear *in $\hat{K}$. The averaged Hamming surrogate is the natural choice here since the reuse loss ℓ_{W} weights each user class equally, and under the routed weights $v_{\hat{k}}(y) = \mathbf{1}[y = \hat{k}]$ the averaging coincides with the disjoint-support refinement of Theorem 1.*

C.5 Proof of Theorem 1

Proof Sketch: Since every candidate class is a singleton, each candidate's empirical Rademacher average is a centered scalar average, which is $E/\sqrt{\hat{N}}$-sub-Gaussian uniformly over the pool, and according to Lemma 7 (Massart's finite-class lemma), the maximum over the P candidates incurs a cost of $E\sqrt{2\log P/\hat{N}}$, meaning the selection process introduces only a logarithmic cost with respect to c. Then, the coordinate-wise contraction of Lemma 9 yields $\hat{K}$ identical summands that the $1/\hat{K}$ prefactor of ℓ_{W} cancels exactly, while the disjoint supports of the routed weights localize each coordinate to its class-conditional subsample. Finally, combining the above results with the symmetrization bound of Eq. 10, the desired bound can be derived, with the entropy term improved by a further factor $\sqrt{\hat{K}}$.

Proof. Let $v_{\hat{k}}(y) = \mathbf{1}[y = \hat{k}]$, so that $V = 1$ and the $\hat{K}$ summands of ℓ_{W} have disjoint supports, for every $(\boldsymbol{x}, y)$ exactly one index $\hat{k} = y$ contributes. Consequently, for any $f \in \mathcal{F}_{\mathrm{LW}}$,

$$\ell_{\mathrm{W}}\big(f(\boldsymbol{x}), y\big) = \frac{1}{\hat{K}}\,\ell_y\big(f_y(\boldsymbol{x}), y\big),$$

and a user example of class $\hat{k}$ constrains only the $\hat{k}$-th coordinate.

Let $\mathcal{G}_{\hat{k}} = \{(\boldsymbol{x}, y) \mapsto \mathbf{1}[y = \hat{k}]\, \ell_{\hat{k}}(h(\boldsymbol{x}), \hat{k}) : h \in \mathcal{H}\}$ be the coordinate loss class of Eq. 19 under the routed weights. Subadditivity as in Eq. 21 gives $\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{L}_{\mathrm{LW}}) \le \frac{1}{\hat{K}} \sum_{\hat{k}} \hat{\mathfrak{R}}_{\hat{\mathcal{D}}}(\mathcal{G}_{\hat{k}})$. Since $\mathbf{1}[y = \hat{k}]$ vanishes on every user example whose label is not $\hat{k}$, the empirical process of $\mathcal{G}_{\hat{k}}$ on $\hat{\mathcal{D}}$ reduces to one on the class-conditional subset $\hat{\mathcal{D}}_{\hat{k}}$, while the normalization of Eq. 9 still taken over all $\hat{N}$ examples:

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}}\big(\mathcal{G}_{\hat{k}}\big) = \mathbb{E}_{\boldsymbol{\epsilon}}\left[\sup_{h \in \mathcal{H}} \frac{1}{\hat{N}} \sum_{\hat{n}=1}^{\hat{N}} \epsilon_{\hat{n}}\, \mathbf{1}\big[\hat{y}_{\hat{n}} = \hat{k}\big]\, \ell_{\hat{k}}\big(h(\hat{\boldsymbol{x}}_{\hat{n}}), \hat{k}\big)\right] = \frac{\hat{N}_{\hat{k}}}{\hat{N}}\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}\big(\mathcal{G}_{\hat{k}}\big), \tag{23}$$

where the second equality reindexes the sum over the $\hat{N}_{\hat{k}}$ examples of class $\hat{k}$ and factors out $\hat{N}_{\hat{k}}/\hat{N}$. Then, substituting Eq. 23 into the symmetrization bound of Eq. 10 yields

$$\mathcal{R}_{\mathrm{LW}}(f) \;\le\; \hat{\mathcal{R}}_{\hat{\mathcal{D}},\mathrm{LW}}(f) + \frac{2}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \frac{\hat{N}_{\hat{k}}}{\hat{N}}\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}\big(\mathcal{G}_{\hat{k}}\big) + 3M\sqrt{\frac{\log\frac{2}{\delta}}{2\hat{N}}}. \tag{24}$$

Then, on the sample $\hat{\mathcal{D}}_{\hat{k}}$ of size $\hat{N}_{\hat{k}}$, the map $s \mapsto \ell_{\hat{k}}(s, \hat{k})$ is μ-Lipschitz by Assumption 1, so with Lemma 5, we have $\hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}(\mathcal{G}_{\hat{k}}) \le C_\ell\, \mu\, \hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}(\mathcal{H})$, and applying Lemma 7 in the form of Eq. 13, we further have

$$\hat{\mathfrak{R}}_{\hat{\mathcal{D}}_{\hat{k}}}\big(\mathcal{G}_{\hat{k}}\big) \;\le\; C_\ell\, \mu E \sqrt{\frac{2\log(c\bar{K})}{\hat{N}_{\hat{k}}}}, \tag{25}$$

since every candidate class $\mathcal{H}_{i,k}$ is a singleton and the first term of Eq. 14 vanishes. Substituting Eq. 25 into Eq. 24, we have Eq. 6.

If $\hat{N}_{\hat{k}} = \hat{N}/\hat{K}$ holds for all $\hat{k}$, then Eq. 25 is the same for every $\hat{k}$, and Eq. 24 becomes

$$\mathcal{R}_{\mathrm{LW}}(f) \;\le\; \hat{\mathcal{R}}_{\hat{\mathcal{D}},\mathrm{LW}}(f) + \frac{2C_\ell\, \mu E}{\hat{K}} \sqrt{\frac{2\hat{K}\log(c\bar{K})}{\hat{N}}} + 3M\sqrt{\frac{\log\frac{2}{\delta}}{2\hat{N}}},$$

which is the balanced-case bound in Remark 1 up to a constant factor. Dividing $E\sqrt{2\log(c\bar{K})/\hat{N}}$ by $\hat{K}$ and reinserting $\hat{N}_{\hat{k}} = \hat{N}/\hat{K}$ inside the square root yields $E\sqrt{2\log(c\bar{K})/(\hat{N}\hat{K})}$, so the entropy term improves by exactly $\sqrt{\hat{K}}$ over the unlocalized pool bound of Lemma 7.

Finally, the EGS constants follow immediately. Under the deployed predictor of Section 4.1, each candidate class is the singleton $\mathcal{H}_{i,k} = \{h_{i,k}\}$ with $h_{i,k}(\boldsymbol{x}) = \big[f_{\boldsymbol{\theta}_i}(\boldsymbol{x})\big]_k$, and the envelope condition of Assumption 1 holds over the global pool $\mathcal{H}^\star$ with $E = 1$ for probability outputs and $E = B$ for logits clipped at level B, which completes the proof. □

C.6 PROOF OF THEOREM 2

Assumption 2. *Let $\mathcal{H}^\star$ be the global candidate pool defined in Section 4.1 (of which the dock system-induced pool $\mathcal{H}$ is a subset), a learnware ε-*covers *a user class $\hat{k}$, for a tolerance $\varepsilon \ge 0$, if $\mathcal{P}$ contains a pair (i, k) whose aligned score attains the per-class risk within ε of the Bayes risk, i.e., $\mathcal{R}_{\hat{k}}(h_{i,k}) \le \mathcal{R}_{\hat{k}}(f^{\mathrm{opt}}) + \varepsilon$ ($\varepsilon = 0$ recovers exact coverage). For every $\hat{k}$: (i) $\pi_{\hat{k}} \in [0, 1)$ is the probability that $\hat{k}$ is* structurally uncoverable*, i.e., no learnware ever ε-covers it; (ii) $p_{\hat{k}} \in (0, 1]$ is the probability that a single learnware ε-covers $\hat{k}$, conditional on coverability; (iii) structural types and learnwares are generated independently, and covering events of each class are independent across learnwares, where cross-class correlations through shared learnwares are allowed since the analysis only averages per-class expectations. We define $\bar{\pi} = \frac{1}{\hat{K}} \sum_{\hat{k}} \pi_{\hat{k}}$ and $p_{\min} = \min_{\hat{k}} p_{\hat{k}}$, and $\pi_{\hat{k}} \equiv 0$, $p_{\hat{k}} \equiv p$ recovers the homogeneous case.*

Assumption 3. *For every user class $\hat{k} \in [\hat{K}]$, let $D_{\hat{k},(i,k)} = \big\|\bar{\boldsymbol{G}}^{\hat{k}} - \boldsymbol{G}_{\mathcal{D}_i^k}\big\|_F$ be the population gradient distance to candidate $(i, k) \in \mathcal{P}$, pairing the population one-step energy gradient $\bar{\boldsymbol{G}}^{\hat{k}} =$*

$\mathbb{E}_{(\boldsymbol{x},y)\sim P_{\hat{\boldsymbol{X}}|y=\hat{k}}}\big[\nabla_{\boldsymbol{\eta}}\,\ell\big(g_{\boldsymbol{w},\boldsymbol{\eta}}\circ\psi_{\boldsymbol{\vartheta}}(\boldsymbol{x}),\hat{k}\big)\big]$ *of the user class with the developer's full-data gradient* $\boldsymbol{G}_{\mathcal{D}_i^k}$ *of Eq. 2. Since squaring is monotone, these distances induce the same ranking as the squared objective of Eq. 5. Assume the population distances are: (i)* risk-aligned*: a candidate attaining the smallest distance is per-class optimal, i.e.,* $\min_{h\in\mathcal{H}_{(i^*,k^*)}}\mathcal{R}_{\hat{k}}(h)=\min_{h\in\mathcal{H}}\mathcal{R}_{\hat{k}}(h)$ *for every* $(i^*,k^*)\in\arg\min_{(i,k)\in\mathcal{P}}D_{\hat{k},(i,k)}$*; and (ii)* separated*:* $D^{(2)}_{\hat{k}}-D^{(1)}_{\hat{k}}\geq\Delta>0$ *holds for the smallest and the second-smallest such distances over* $\mathcal{P}$*, where we write* $\Delta=\Delta(c)$ *to emphasize that the margin is taken on the realized pool of* c *learnwares and may shrink as the dock system grows.*

Assumption 4. *There exists* $\bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}})$ *such that, for every learnware that could be submitted, the optimized proxy specification is within* $\bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}})$ *of its full-data specification, i.e.,* $\big\|\boldsymbol{G}_{\mathcal{S}_i^k}-\boldsymbol{G}_{\mathcal{D}_i^k}\big\|_F\leq\bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}})$ *holds for every candidate pair* (i,k) *of every possible submission almost surely over the proxy-set optimization, and* $\bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}})\to 0$ *as* $M_{\mathrm{spec}}\to\infty$*. This uniformity is imposed over the optimization procedure rather than the models, making* $\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}})$ *independent of* c *by construction.*

Proof. Let $f^*_{\mathrm{cov}}\in\mathcal{F}_{\mathrm{LW}}$ be the population-optimal routed predictor *achievable with the learnwares actually present in the dock system*, i.e., the predictor that assigns to each user class $\hat{k}$, the candidate in $\mathcal{H}$ minimizing the class-$\hat{k}$ risk. Adding and subtracting its risk yields the exact decomposition

$$\mathcal{R}_{\mathrm{LW}}(\hat{f})-\mathcal{R}_{\mathrm{LW}}\big(f^{\mathrm{opt}}\big)\;=\;\underbrace{\Big[\mathcal{R}_{\mathrm{LW}}(f^*_{\mathrm{cov}})-\mathcal{R}_{\mathrm{LW}}\big(f^{\mathrm{opt}}\big)\Big]}_{\mathcal{E}_{\mathrm{cov}}(c)}+\underbrace{\Big[\mathcal{R}_{\mathrm{LW}}(\hat{f})-\mathcal{R}_{\mathrm{LW}}(f^*_{\mathrm{cov}})\Big]}_{\mathcal{E}_{\mathrm{spec}}(M_{\mathrm{spec}})},\tag{26}$$

where the first bracket measures how well the learnwares in the dock system can approximate the Bayes-optimal predictor, and the second measures how well the deploying stage recovers this best achievable predictor from the proxy specifications.

Step 1: The coverage term. By the definition of f^*_{cov}, the class-$\hat{k}$ contribution to $\mathcal{E}_{\mathrm{cov}}(c)$ equals the per-class approximation gap $\min_{h\in\mathcal{H}}\mathcal{R}_{\hat{k}}(h)-\mathcal{R}_{\hat{k}}\big(f^{\mathrm{opt}}\big)$, which lies in $[0,\Delta_{\max}]$. It is non-negative since f^{opt} is Bayes-optimal, and it is at most $\Delta_{\max}$ since the dock system-induced pool $\mathcal{H}$ is a subset of the global candidate pool $\mathcal{H}^\star$ over which $\Delta_{\max}$ is defined. The gap is at most ε whenever the dock system contains a learnware that ε-covers $\hat{k}$ in the sense of Assumption 2. Fix a class $\hat{k}$ and condition on its structural type. With probability $\pi_{\hat{k}}$, the class is structurally uncoverable, in which case no learnware in the dock system can ε-cover it and the contribution is at most $\Delta_{\max}$ for *every* c. With probability $1-\pi_{\hat{k}}$, the class is structurally coverable, and then by Assumption 2 each learnware ε-covers it independently with probability $p_{\hat{k}}$, so for a dock system of size c, the probability that *no* learnware ε-covers $\hat{k}$ is $(1-p_{\hat{k}})^c$; on that event the contribution is at most $\Delta_{\max}$, and on the complementary event it is at most ε. Then, taking expectation over the dock system generation and averaging over the $\hat{K}$ classes yields

$$\begin{aligned}\mathbb{E}\big[\mathcal{E}_{\mathrm{cov}}(c)\big]&\leq\frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}}\Big[\pi_{\hat{k}}\Delta_{\max}+(1-\pi_{\hat{k}})\big(\varepsilon\,\big(1-(1-p_{\hat{k}})^c\big)+\Delta_{\max}(1-p_{\hat{k}})^c\big)\Big]\\&\leq\Delta_{\max}\left[\bar{\pi}+\frac{1}{\hat{K}}\sum_{\hat{k}=1}^{\hat{K}}(1-p_{\hat{k}})^c\right]+\varepsilon\big(1-\bar{\pi}\big),\end{aligned}\tag{27}$$

where the second inequality uses $(1-\pi_{\hat{k}})\leq 1$, $(1-\pi_{\hat{k}})(1-p_{\hat{k}})^c\leq(1-p_{\hat{k}})^c$, $(1-\pi_{\hat{k}})(1-(1-p_{\hat{k}})^c)\leq(1-\pi_{\hat{k}})$ and $\frac{1}{\hat{K}}\sum_{\hat{k}}\pi_{\hat{k}}=\bar{\pi}$. Note that the first summand is a *diversity floor*, it does not depend on c and is the only part of the bound that cannot be removed by enlarging the dock system, while the third is a *proximity floor* paid on the classes that the dock system manages to ε-cover. The second summand is geometric in c and is sandwiched between $\frac{1}{\hat{K}}(1-p_{\min})^c$ and $(1-p_{\min})^c$, so the least-covered class determines its rate. When $\pi_{\hat{k}}\equiv 0$ and $p_{\hat{k}}\equiv p$, Eq. 27 reduces to $\Delta_{\max}(1-p)^c+\varepsilon\big(1-(1-p)^c\big)$.

Step 2: The specification term. We analyze the *unconstrained* per-class selector $\hat{\pi}(\hat{k}) \in \arg\min_{(i,k)\in\mathcal{P}} \hat{D}_{\hat{k},(i,k)}$ produced by the gradient matching of Eq. 5 with proxy-set estimates, which is a relaxation of the routed selection of Algorithm 3. Since no fitting operations are performed on the user sample under the deploying stage, the second bracket of Eq. 26 is exactly the class-selection error of this selector. For each user class $\hat{k}$, the error of every estimated distance relative to the population distances of Assumption 3 is at most the realized total distortion

$$\delta_{\hat{k}} \;=\; \left\|\hat{\boldsymbol{G}}_{\hat{\mathcal{S}}^{\hat{k}}} - \hat{\boldsymbol{G}}_{\hat{\mathcal{D}}^{\hat{k}}}\right\|_F + \left\|\hat{\boldsymbol{G}}_{\hat{\mathcal{D}}^{\hat{k}}} - \bar{\boldsymbol{G}}^{\hat{k}}\right\|_F + \bar{\delta}_{\mathrm{spec}}(M_{\mathrm{spec}}) \tag{28}$$

by the triangle inequality and the uniform envelope of Assumption 4. Whenever $\delta_{\hat{k}} < \Delta(c)/2$, the estimated distances move the smallest and the second-smallest population distances by less than $\Delta(c)/2$ each, so the margin of Assumption 3 keeps the minimizer of the ranking unchanged. By the alignment clause of Assumption 3, the selected candidate then attains the per-class optimum $\min_{h\in\mathcal{H}} \mathcal{R}_{\hat{k}}(h)$, which coincides with the class-$\hat{k}$ contribution of f^*_{cov}, and hence the class-$\hat{k}$ contribution to the second bracket of Eq. 26 is zero. Otherwise the contribution is at most $\Delta_{\max}$, since $\mathcal{R}_{\hat{k}}(h) - \mathcal{R}_{\hat{k}}(h') \le \Delta_{\max}$ holds for any $h, h' \in \mathcal{H}$ by the definition of $\Delta_{\max}$ and the non-negativity of the per-class gaps. Then, by Markov's inequality $\mathbb{P}\big[\delta_{\hat{k}} \ge \Delta(c)/2\big] \le \frac{2\,\mathbb{E}[\delta_{\hat{k}}]}{\Delta(c)}$, and averaging the per-class contributions over the $\hat{K}$ classes, we have

$$\begin{aligned}
\mathbb{E}\Big[\mathcal{R}_{\mathrm{LW}}(\hat{f}) - \mathcal{R}_{\mathrm{LW}}\big(f^*_{\mathrm{cov}}\big)\Big] &\le \frac{\Delta_{\max}}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} \mathbb{P}\big[\delta_{\hat{k}} \ge \Delta(c)/2\big] \\
&\le \frac{2\Delta_{\max}}{\hat{K}\Delta(c)} \sum_{\hat{k}=1}^{\hat{K}} \mathbb{E}\big[\delta_{\hat{k}}\big] = \frac{2\Delta_{\max}}{\Delta(c)} \big(\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}}) + \epsilon_{\mathrm{data}}(\hat{N})\big),
\end{aligned} \tag{29}$$

where the expectation is over the proxy-set optimization and the draw of the user sample, the first inequality uses the exact per-class decomposition of Section 4.1, and the last equality follows from the definitions of $\epsilon_{\mathrm{spec}}(M_{\mathrm{spec}})$ and $\epsilon_{\mathrm{data}}(\hat{N})$ in Section 4.2. Note that the specification bound is independent of c by the uniform envelope of Assumption 4 and vanishes as M_{spec} and $\hat{N}$ grow.

Step 3: Collecting. Finally, adding Eqs. 27 and 29 and using Eq. 26 yields Eq. 8 in expectation over the structural types, the dock system generation, the proxy-set optimization and the draw of the user sample. High-probability statements are not required here since no fitting based on the user samples occurs during the deployment phase, the user samples influence the excess risk only through the specification term, meanwhile, the generalization cost associated with evaluating an arbitrary routed predictor on the finite user samples is separately controlled by Theorem 1. In particular, on the event that the realized total distortion of Eq. 28 satisfies $\delta_{\hat{k}} < \Delta(c)/2$ for a user class $\hat{k}$, the unconstrained class-$\hat{k}$ ranking is preserved and the selected candidate attains the population-optimal per-class risk by the alignment clause of Assumption 3. The specification term captures exactly the residual selection error of the unconstrained selector, which vanishes as M_{spec} and $\hat{N}$ grow. □

C.7 Proof of Corollary 3

Proof. The dock system affects Eq. 8 only through the coverage term and the margin $\Delta(c)$, the diversity floor $\Delta_{\max}\bar{\pi}$, the proximity floor $\varepsilon(1-\bar{\pi})$ and the specification terms are independent of c, the latter benefiting from the uniform envelope of Assumption 4. The dock system-dependent part is therefore $\frac{\Delta_{\max}}{\hat{K}} \sum_{\hat{k}=1}^{\hat{K}} (1-p_{\hat{k}})^c$, which is non-increasing in c since every factor $(1-p_{\hat{k}})^c$ is non-increasing, and decreases geometrically at the rate of the least-covered class. Retaining only the summand indexed by $\hat{k}_{\min} = \arg\min_{\hat{k}} p_{\hat{k}}$ yields the lower side of the sandwich, and bounding every $(1-p_{\hat{k}}) \le (1-p_{\min})$ yields the upper side. Finally, adding the c-independent floors and specification terms yields the floor of Corollary 3, and the monotonicity conclusion holds whenever $\Delta(c)$ stays bounded away from zero. In the homogeneous case the sum has $\hat{K}$ identical terms and reduces to $\Delta_{\max}(1-p)^c$. □

D EXPERIMENT

D.1 DATASET

We conduct experiments on the twelve image and text classification benchmarks introduced in Section 5.1. The suite is selected to cover a broad range of semantic granularity, visual or linguistic variability, and task-partition structures, thereby testing whether a specification can identify reusable learnwares beyond a single data modality or a narrowly defined category.

Image Benchmarks. The image suite consists of six benchmarks covering generic object, fine-grained, texture, scene, and traffic-sign recognition.

- **CIFAR-100.** CIFAR-100 (Krizhevsky & Hinton, 2009) contains 100 low-resolution object categories organized into 20 coarse superclasses. This explicit hierarchy makes it a natural benchmark for constructing superclass-based developer and user tasks.
- **Food101.** Food101 (Bossard et al., 2014) contains 101 food categories. Its categories exhibit substantial intra-class variation and high visual similarity among related dishes, providing a challenging fine-grained recognition setting.
- **DTD.** The Describable Textures Dataset (DTD) (Cimpoi et al., 2014) contains 47 texture categories. It emphasizes local patterns, material properties, and appearance characteristics that differ from object-centric visual cues.
- **Cars.** Stanford Cars (Krause et al., 2013) is a fine-grained vehicle classification benchmark whose categories differ by manufacturer, model, and production year. It evaluates performance when semantically related classes have highly similar visual appearances.
- **SUN397.** SUN397 (Xiao et al., 2016) contains 397 indoor and outdoor scene categories with diverse layouts, background structures, and contextual cues. It provides the largest visual label space among the image benchmarks.
- **GTSRB.** The German Traffic Sign Recognition Benchmark (GTSRB) (Stallkamp et al., 2012) contains 43 traffic-sign categories collected under varying scales, viewpoints, and illumination conditions.

Text Benchmarks. The text suite consists of six benchmarks spanning topic, review, technical-domain, question, and scientific-document classification.

- **20Newsgroups.** 20Newsgroups (Lang, 1995) covers 20 discussion topics from domains including computers, science, politics, and religion. It is used to evaluate general topic classification across semantically related subject areas.
- **Amazon.** Amazon (Ni et al., 2019) consists of product reviews from multiple product categories. The corpus includes diverse writing styles and sentiment-related expressions.
- **Bugs.** Bugs (Lee et al., 2024) contains software defect and bug-report text. Its classification tasks rely on technical vocabulary and domain-specific descriptions of software behavior.
- **Coronavirus.** Coronavirus (Wang et al., 2020) contains documents related to COVID-19 and spans heterogeneous pandemic topics. It therefore presents substantial topical and linguistic variation.
- **TREC.** TREC (Li & Roth, 2002) is a question classification benchmark whose labels include semantic types such as person, location, numerical quantity, and entity.
- **WoS.** Web of Science (WoS) (Kowsari et al., 2017) contains scientific abstracts categorized by research area. The dataset combines specialized terminology with a hierarchical disciplinary structure.

D.2 COMPARED METHODS

We compare EGS with four specification-based baselines: RKME (Wu et al., 2023), CRKME (Liu et al., 2025b), LANE (Chen et al., 2025b), and DALI (Chen et al., 2025a). We also report MAX and Oracle as non-deployable manual reference methods. All methods are evaluated on exactly the same

developer and user tasks, use the same stored well-established models and shared feature extractors, and are assigned the same specification sizes. They are further evaluated with the same reuse metric. Thus, the comparison isolates the effect of the information retained by the specification and the corresponding matching rule.

- **RKME.** Reduced Kernel Mean Embedding (RKME) specification represents the marginal distribution of a task through a reduced-set kernel mean embedding in a reproducing kernel Hilbert space. It provides a compact distributional specification, but does not explicitly retain class-conditional structure or the predictive behavior of the associated model. We use a Gaussian kernel with $\gamma = 0.1$ for RKME throughout all experiments.
- **CRKME.** Class-Conditional Reduced Kernel Mean Embedding (CRKME) specification extends RKME by constructing an independent reduced kernel mean embedding for every class. This class-wise construction preserves conditional distributional information that is discarded by the marginal representation of RKME and enables class-level matching between developer and user tasks.
- **LANE.** Label-Aware Neural Embedding (LANE) specification replaces the predefined kernel space of RKME with a learned neural embedding space. It constructs compact task specifications by matching class-wise feature distributions in this space. LANE therefore serves as a representative class-aware distribution-matching baseline whose representation is learned rather than kernel-defined.
- **DALI.** Dual ALIgnment (DALI) specification combines distributional alignment with discriminative alignment. In addition to preserving information about the task distribution, it incorporates discriminative behavior from the associated model. This baseline is particularly relevant for assessing whether the proposed energy-gradient specification captures model-dependent reuse information beyond conventional distribution matching.
- **MAX and Oracle.** MAX selects the single learnware that yields the highest downstream accuracy for a user task. Oracle selects the best feasible reuse solution and may combine multiple learnwares when multi-learnware reuse is allowed. Both methods require downstream evaluation information that is unavailable during practical learnware identification; they are therefore included only to characterize the performance attainable with hindsight rather than as deployable methods.

D.3 EXPERIMENT CONFIGURATION

Dataset Splits and Task Construction. We retain the predefined training/test split supplied with each benchmark to avoid overlap between developer and user data. For each dataset, a superclass from the training partition defines one developer learnware task, and the fine-grained subclasses nested within that superclass are used as the task labels. The corresponding well-established model is trained only on data from this developer task. User tasks are sampled from superclasses in the test partition according to the three reuse scenarios defined in Section 5.2. For every dataset and scenario, we sample 100 user tasks and report the mean and standard deviation over these tasks. Table 3 details the resulting 131 developer tasks, including the retained superclass partitions. The class totals in the table count only the subclasses retained by our task construction and may therefore differ from the nominal class counts of the original datasets.

Feature Extractors and Learnware Models. For specification construction, every method uses the same pretrained *ResNet110*[1] on image tasks and the same pretrained *Sentence Transformer*[2] on text tasks. These shared extractors are frozen and are used solely to construct specifications; they are independent of the well-established models that are submitted to and stored in the learnware dock system. For standard image learnwares, we fine-tune a pretrained *ResNet110* using samples from the corresponding superclass. For standard text learnwares, we train a classifier using *TfidfVectorizer* features extracted from the corresponding task data. The specification budget is fixed to $M = 10 \times K$ for all methods, where K is the number of classes in the relevant task. This choice equalizes the

[1]Code of ResNet110: https://github.com/bearpaw/pytorch-classification/blob/master/TRAINING.md

[2]Code of Sentence Transformer: https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2/tree/main

Table 3: Developer-task partitions used to construct the learnware repository. "Tasks" and "Classes" denote the number of developer tasks and retained fine-grained classes, respectively (131 tasks in total). Each entry in the final column is formatted as *superclass* (number of retained subclasses).

Dataset	Tasks	Classes	Developer-task partition
Image Benchmarks			
CIFAR-100	20	100	aquatic_mammals (5); electrical_devices (5); fish (5); flowers (5); food-containers (5); fruit-vegetable (5); furniture (5); insects (5); invertebrates (5); large_carnivores (5); medium_sized_mammals (5); omnivores (5); outdoor_scenes (5); outdoor_things (5); people (5); reptiles (5); small_mammals (5); trees (5); vehicles_1 (5); vehicles_2 (5)
DTD	11	34	straight_line_grid (4); dot_bump (3); textile_weave (3); organic_groove_dynamic (3); layered (3); stochastic_granular (4); porous (3); damaged_irregular (3); surface_relief (3); complex_natural (3); fine_delicate (2)
Food101	24	66	cakes (3); desserts_fine (3); desserts_fried (3); desserts_cold (2); desserts_bakery (3); bread_based (3); sandwiches (4); pasta (3); rice_dishes (3); asian_noodles (3); beef (3); pork (2); chicken (2); seafood_main (3); raw_seafood (2); fried_food (3); mexican (3); middle_eastern (2); japanese (3); chinese (3); soups (2); salads (3); appetizers (3); breakfast (2)
GTSRB	4	43	speed_limits (9); prohibitions-derestrictions (8); mandatory (9); danger (17)
Cars	19	192	compact_sedans (10); mid_size_sedans (10); luxury_sedans (12); executive_sedans (18); sports_coupes (10); supercars (11); luxury_convertibles (11); compact_convertibles (15); compact_hatchbacks (14); wagons (5); compact_suvs (8); mid_size_suvs (11); full_size_suvs (15); light_pickups (10); heavy_pickups (8); cargo_vans (6); passenger_vans (6); muscle_cars (5); classic_sports (7)
SUN397	18	313	indoor_residential (33); indoor_commercial (35); indoor_office (13); indoor_education (8); indoor_medical (9); indoor_religious (10); indoor_entertainment (13); indoor_sports (12); indoor_industrial (13); outdoor_urban (16); outdoor_architecture (19); outdoor_transportation (20); outdoor_sports (20); outdoor_agriculture (15); outdoor_natural (38); outdoor_water (12); outdoor_industrial (16); transport_interior (11)
Text Benchmarks			
20Newsgroups	5	20	comp (5); misc (3); rec (4); sci (4); talk (4)
TREC	6	50	ABBR (2); ENTY (22); DESC (4); HUM (4); LOC (5); NUM (13)
Coronavirus	2	5	Extremely (2); Bool (3)
WoS	7	134	CS (17); ECE (16); Psychology (19); MAE (9); Civil (11); Medical (53); Biochemistry (9)
Bugs	10	85	ACPI (12); UNKNOW (9); Drivers (23); File-Systems (10); IO-Storage (7); Memory-Management (4); Networking (5); Other (4); Platform-Specific-Hardware (6); Management (5)
Amazon	5	25	arts-crafts-and-sewing (5); electronics (5); grocery-and-gourmet-food (5); musical-instruments (5); video-games (5)

storage budget across methods while allowing the number of specification samples to scale with task complexity.

Corrupted Learnwares. To construct corrupted learnwares, we replace the standard image model with a *CNN* and the standard text model with a *MultinomialNB* classifier. We additionally inject label noise into the training data used for these models. Consequently, corrupted learnwares differ from the standard learnware pool in both model family and label reliability. They serve as controlled unreliable candidates for evaluating whether a specification can distinguish useful learnwares from models whose capability is degraded by architectural mismatch or noisy supervision.

D.4 PERFORMANCE

We construct user tasks by sampling subclasses from the superclass partitions in Table 3. For each benchmark and scenario, we draw 100 user tasks. The scenarios differ in whether selected subclasses originate from one or multiple superclasses and in the amount of labeled user data:

- **Beyond Original Functionality.** Each user task combines subclasses drawn from multiple test superclasses. No individual learnware covers the complete class set; successful reuse therefore requires identifying multiple compatible learnwares and assigning the matched subset of user classes to each selected model. This setting evaluates both class-wise compatibility estimation and the ability to compose several learnwares into a single reuse solution.
- **Within Original Functionality.** Each user task is formed by sampling fine-grained subclasses from a single test superclass. Because the corresponding semantic scope is contained within one original functionality, one learnware is sufficient to cover all classes in the user task. This setting evaluates whether a specification can identify and retrieve the most compatible single learnware without requiring downstream trial evaluation or combining models.
- **Few-Shot Beyond Original Functionality.** This setting retains the same multi-superclass construction as *Beyond Original Functionality*, so complete coverage still requires multi-learnware selection and class-wise routing. However, only 10 labeled examples per selected subclass are retained for constructing the user specification. It evaluates whether the identification and composition procedure remains reliable when the user provides only limited labeled supervision.

Performance Analysis. Tables 1, 4, and 5 report results for *beyond original functionality*, *within original functionality*, and *few-shot beyond original functionality* reuse, respectively. EGS is the strongest deployable method on average in all three reuse settings, obtaining accuracies of 0.710, 0.707 and 0.686. Relative to the strongest automatic baseline, DALI, the corresponding margins are 7.6, 8.5 and 7.2 percentage points. The margins over CRKME are 19.1, 19.5 and 23.7 points, and those over RKME are 38.6, 33.2 and 39.4 points. Thus, class-wise distribution matching improves over marginal distribution matching, while gradient-based matching supplies a stronger signal for learnware compatibility.

The advantage is maintained across both modalities and the three reuse regimes. Under within-original-functionality reuse, EGS achieves the best automatic result on 11 of the 12 datasets (including ties), with particularly clear gains on 20Newsgroups (0.864), Coronavirus (0.716), and WoS (0.722). The remaining gap to the manual MAX reference is only 2.2 points on average (0.707 versus 0.729), indicating that the learned specification can usually recover the appropriate single learnware without downstream trial evaluation. The task becomes more demanding when the user task spans multiple original functionalities. Nevertheless, EGS remains the top automatic method with an average rank of 1.46, and matches the Oracle performance on nine datasets. Its shortfalls are concentrated on Cars and SUN397, where the Oracle has access to the downstream-optimal combination unavailable to a practical dock system.

Few-shot user supervision does not change this trend. With only ten labeled examples per selected subclass, EGS has the best average rank (1.29) and wins or ties DALI on 11 of 12 datasets. It remains close to Oracle on several heterogeneous text tasks, e.g., Bugs (0.360 versus 0.366), TREC (0.646 versus 0.647), and WoS (0.664 versus 0.664), while retaining high accuracy on CIFAR-100 (0.895) and GTSRB (0.996). The average 2.8-point gap to Oracle in this setting is expected because Oracle can directly evaluate downstream performance and therefore represents an unattainable upper bound. Overall, the results demonstrate that the proposed gradient-based specification supports accurate learnware identification and effective multi-learnware reuse even when user labels are scarce.

D.5 MODEL QUALITY

Experimental Setup. We evaluate whether EGS supports reliable learnware identification when the dock system contains low-quality models. The construction of corrupted learnwares and all shared experimental configurations follow Appendix D.3. The *Normal* setting contains only the

Table 4: Performance comparison under the *Within Original Functionality* setting. Results are reuse classification accuracy, reported as mean ± standard deviation over sampled user tasks. For each dataset, the best mean among non-oracle entries is highlighted in bold and underlined.

User Task	*Learnware Identification and Reuse*					*Manual References*	
Dataset	**RKME**	**CRKME**	**LANE**	**DALI**	**EGS**	**MAX**	**Oracle**
CIFAR-100	0.461 ± 0.170	0.850 ± 0.137	0.848 ± 0.166	$\mathbf{0.882 \pm 0.063}$	$\mathbf{0.882 \pm 0.063}$	$\mathbf{0.882 \pm 0.063}$	0.882 ± 0.063
Food101	0.229 ± 0.132	0.615 ± 0.298	0.448 ± 0.344	$\mathbf{0.808 \pm 0.079}$	$\mathbf{0.808 \pm 0.079}$	$\mathbf{0.808 \pm 0.079}$	0.808 ± 0.079
DTD	0.218 ± 0.170	0.117 ± 0.232	0.490 ± 0.331	$\mathbf{0.802 \pm 0.119}$	0.800 ± 0.110	$\mathbf{0.802 \pm 0.119}$	0.802 ± 0.119
Cars	0.092 ± 0.140	0.057 ± 0.126	0.113 ± 0.149	0.210 ± 0.195	0.229 ± 0.213	$\mathbf{0.441 \pm 0.163}$	0.441 ± 0.163
SUN397	0.166 ± 0.124	0.111 ± 0.104	0.354 ± 0.175	0.500 ± 0.150	0.510 ± 0.140	$\mathbf{0.551 \pm 0.124}$	0.551 ± 0.124
GTSRB	0.427 ± 0.231	0.971 ± 0.093	0.938 ± 0.152	$\mathbf{0.997 \pm 0.003}$	$\mathbf{0.997 \pm 0.003}$	$\mathbf{0.997 \pm 0.003}$	0.997 ± 0.003
20Newsgroups	0.500 ± 0.113	0.771 ± 0.174	0.821 ± 0.138	0.658 ± 0.278	$\mathbf{0.864 \pm 0.057}$	$\mathbf{0.864 \pm 0.057}$	0.864 ± 0.057
Amazon	0.780 ± 0.132	$\mathbf{0.856 \pm 0.077}$	0.846 ± 0.092	$\mathbf{0.856 \pm 0.077}$	$\mathbf{0.856 \pm 0.077}$	$\mathbf{0.856 \pm 0.077}$	0.856 ± 0.077
Bugs	0.259 ± 0.199	0.283 ± 0.295	0.330 ± 0.298	0.183 ± 0.247	0.480 ± 0.281	$\mathbf{0.483 \pm 0.280}$	0.483 ± 0.280
Coronavirus	0.516 ± 0.211	0.497 ± 0.246	0.508 ± 0.368	0.343 ± 0.289	$\mathbf{0.716 \pm 0.179}$	$\mathbf{0.716 \pm 0.179}$	0.716 ± 0.179
TREC	0.355 ± 0.205	0.342 ± 0.341	0.592 ± 0.222	0.528 ± 0.284	0.620 ± 0.204	$\mathbf{0.626 \pm 0.195}$	0.626 ± 0.195
WoS	0.501 ± 0.145	0.677 ± 0.139	0.690 ± 0.143	0.702 ± 0.131	$\mathbf{0.722 \pm 0.103}$	$\mathbf{0.722 \pm 0.103}$	0.722 ± 0.103
Avg. Metric (↑)	0.375	0.512	0.582	0.622	0.707	0.729	0.729
Avg. Rank (↓)	5.33	4.71	4.08	3.33	2.00	1.54	–
EGS (win/tie/loss)	12/0/0	11/1/0	12/0/0	7/4/1	–	0/7/5	0/7/5

Table 5: Performance comparison under the *Few-Shot Beyond Original Functionality* setting. Results are reuse classification accuracy, reported as mean ± standard deviation over sampled user tasks. For each dataset, the best mean among non-oracle entries is highlighted in bold and underlined.

User Task	*Learnware Identification and Reuse*					*Manual References*	
Dataset	**RKME**	**CRKME**	**LANE**	**DALI**	**EGS**	**MAX**	**Oracle**
CIFAR-100	0.473 ± 0.125	0.721 ± 0.156	0.859 ± 0.091	0.893 ± 0.049	$\mathbf{0.895 \pm 0.048}$	0.509 ± 0.125	0.895 ± 0.048
Food101	0.119 ± 0.109	0.303 ± 0.237	0.389 ± 0.187	$\mathbf{0.813 \pm 0.086}$	0.804 ± 0.090	0.454 ± 0.106	0.827 ± 0.060
DTD	0.206 ± 0.108	0.119 ± 0.133	0.472 ± 0.216	$\mathbf{0.804 \pm 0.061}$	$\mathbf{0.804 \pm 0.061}$	0.449 ± 0.105	0.804 ± 0.061
Cars	0.062 ± 0.071	0.042 ± 0.050	0.103 ± 0.071	0.192 ± 0.083	0.199 ± 0.090	$\mathbf{0.241 \pm 0.079}$	0.413 ± 0.085
SUN397	0.173 ± 0.078	0.126 ± 0.072	0.374 ± 0.096	0.508 ± 0.090	$\mathbf{0.511 \pm 0.095}$	0.353 ± 0.090	0.554 ± 0.079
GTSRB	0.352 ± 0.132	0.877 ± 0.101	0.927 ± 0.075	$\mathbf{0.996 \pm 0.007}$	$\mathbf{0.996 \pm 0.012}$	0.599 ± 0.127	0.997 ± 0.004
20Newsgroups	0.424 ± 0.112	0.728 ± 0.127	0.475 ± 0.185	0.597 ± 0.187	$\mathbf{0.842 \pm 0.061}$	0.484 ± 0.099	0.850 ± 0.045
Amazon	0.564 ± 0.119	$\mathbf{0.852 \pm 0.047}$	0.776 ± 0.098	0.827 ± 0.070	$\mathbf{0.852 \pm 0.047}$	0.478 ± 0.106	0.852 ± 0.047
Bugs	0.153 ± 0.065	0.208 ± 0.095	0.162 ± 0.106	0.173 ± 0.105	$\mathbf{0.360 \pm 0.118}$	0.242 ± 0.097	0.366 ± 0.118
Coronavirus	0.226 ± 0.121	0.464 ± 0.151	0.397 ± 0.156	0.377 ± 0.151	$\mathbf{0.660 \pm 0.112}$	0.436 ± 0.076	0.695 ± 0.067
TREC	0.357 ± 0.129	0.332 ± 0.187	0.494 ± 0.169	0.556 ± 0.156	$\mathbf{0.646 \pm 0.083}$	0.411 ± 0.097	0.647 ± 0.082
WoS	0.400 ± 0.076	0.620 ± 0.093	0.536 ± 0.112	0.633 ± 0.077	$\mathbf{0.664 \pm 0.074}$	0.407 ± 0.086	0.664 ± 0.074
Avg. Metric (↑)	0.292	0.449	0.497	0.614	0.686	0.422	0.714
Avg. Rank (↓)	5.58	4.04	3.75	2.50	1.29	3.83	–
EGS (win/tie/loss)	12/0/0	11/1/0	12/0/0	9/2/1	–	11/0/1	0/4/8

original learnwares, whereas the *Corrupt* setting augments this pool with their corrupted counterparts. Tables 2, 6, and 7 report results under the three reuse settings, where parenthesized values denote changes relative to the normal pool. Cross-dataset averages are computed from the reported accuracies by assigning equal weight to each benchmark.

Performance Analysis. Under *Beyond Original Functionality* (Table 2), EGS maintains an average accuracy of approximately 0.710 after corrupted learnwares are introduced. Its accuracy remains unchanged at the reported precision on seven benchmarks, with a maximum decrease of 0.006 on Bugs. In comparison, RKME, CRKME, and LANE exhibit average decreases of 6.90, 12.26, and 5.89 percentage points, respectively. Similar robustness is observed under *Within Original Functionality* (Table 6), where the average accuracy of EGS changes from 0.707 to 0.706. Performance again remains unchanged on seven benchmarks, and the largest decrease is 0.013 on SUN397. Among the five specification methods, EGS achieves the best accuracy, including ties, on 11 of 12 benchmarks in the former setting and all 12 benchmarks in the latter. These results demonstrate that EGS maintains reliable reuse performance in the presence of low-quality candidates across both single- and multi-learnware reuse scenarios.

The effect of model quality becomes more pronounced under *Few-Shot Beyond Original Functionality* (Table 7), where only 10 labeled examples per selected subclass are available to construct the user specification. The average accuracy of EGS decreases from approximately 0.686 to 0.674, corresponding to a reduction of 1.21 percentage points. The largest decreases occur on Food101 (0.041) and Coronavirus (0.032). Nevertheless, EGS retains the highest average accuracy among the speci-

Table 6: Performance of EGS and baselines under the *Within original functionality* setting with corrupted learnwares. For corrupted settings, the value in parentheses denotes the change relative to the corresponding normal setting (green: improvement; red: degradation).

	RKME		CRKME		LANE		DALI		EGS		Oracle
Dataset	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	
Image Benchmarks											
CIFAR-100	0.461	0.381 (-0.081)	0.850	0.846 (-0.004)	0.848	0.778 (-0.070)	0.882	0.882 (0.000)	0.882	0.882 (0.000)	0.882
Food101	0.229	0.244 (+0.015)	0.615	0.512 (-0.103)	0.448	0.393 (-0.055)	0.808	0.808 (0.000)	0.808	0.808 (0.000)	0.808
DTD	0.218	0.153 (-0.065)	0.117	0.116 (-0.001)	0.490	0.452 (-0.038)	0.802	0.798 (-0.004)	0.800	0.802 (+0.002)	0.802
Cars	0.092	0.044 (-0.048)	0.057	0.033 (-0.025)	0.113	0.080 (-0.032)	0.210	0.203 (-0.007)	0.229	0.225 (-0.004)	0.441
SUN397	0.166	0.122 (-0.045)	0.111	0.072 (-0.039)	0.354	0.319 (-0.034)	0.500	0.491 (-0.009)	0.510	0.497 (-0.013)	0.551
GTSRB	0.427	0.337 (-0.090)	0.971	0.946 (-0.024)	0.938	0.892 (-0.046)	0.997	0.997 (0.000)	0.997	0.997 (0.000)	0.997
Text Benchmarks											
20Newsgroups	0.500	0.375 (-0.125)	0.771	0.621 (-0.150)	0.821	0.778 (-0.043)	0.658	0.859 (+0.201)	0.864	0.864 (0.000)	0.864
Amazon	0.780	0.472 (-0.308)	0.856	0.714 (-0.142)	0.846	0.822 (-0.025)	0.856	0.804 (-0.053)	0.856	0.856 (0.000)	0.856
Bugs	0.259	0.209 (-0.050)	0.283	0.125 (-0.157)	0.330	0.320 (-0.010)	0.183	0.456 (+0.273)	0.480	0.475 (-0.006)	0.483
Coronavirus	0.516	0.460 (-0.056)	0.497	0.264 (-0.233)	0.508	0.316 (-0.193)	0.343	0.361 (+0.018)	0.716	0.716 (0.000)	0.716
TREC	0.355	0.304 (-0.051)	0.342	0.168 (-0.174)	0.592	0.467 (-0.126)	0.528	0.527 (0.000)	0.620	0.626 (+0.006)	0.626
WoS	0.501	0.347 (-0.154)	0.677	0.380 (-0.297)	0.690	0.668 (-0.022)	0.702	0.722 (+0.020)	0.722	0.722 (0.000)	0.722

Table 7: Performance of EGS and baselines under the *Few-Shot Beyond Original Functionality* setting with corrupted learnwares. For corrupted settings, the value in parentheses denotes the change relative to the corresponding normal setting (green: improvement; red: degradation).

	RKME		CRKME		LANE		DALI		EGS		Oracle
Dataset	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	Normal	Corrupt	
Image Benchmarks											
CIFAR-100	0.473	0.383 (-0.091)	0.721	0.692 (-0.028)	0.859	0.782 (-0.078)	0.893	0.894 (0.000)	0.895	0.894 (-0.001)	0.895
Food101	0.119	0.095 (-0.023)	0.303	0.293 (-0.010)	0.389	0.374 (-0.015)	0.813	0.764 (-0.048)	0.804	0.763 (-0.041)	0.827
DTD	0.206	0.163 (-0.043)	0.119	0.133 (+0.014)	0.472	0.424 (-0.048)	0.804	0.804 (0.000)	0.804	0.804 (0.000)	0.804
Cars	0.062	0.040 (-0.022)	0.042	0.015 (-0.027)	0.103	0.075 (-0.028)	0.192	0.170 (-0.022)	0.199	0.182 (-0.017)	0.413
SUN397	0.173	0.129 (-0.043)	0.126	0.079 (-0.047)	0.374	0.324 (-0.049)	0.508	0.492 (-0.017)	0.511	0.498 (-0.013)	0.554
GTSRB	0.352	0.288 (-0.064)	0.877	0.809 (-0.068)	0.927	0.845 (-0.082)	0.996	0.975 (-0.021)	0.996	0.984 (-0.012)	0.997
Text Benchmarks											
20Newsgroups	0.424	0.304 (-0.120)	0.728	0.564 (-0.164)	0.475	0.730 (+0.255)	0.597	0.823 (+0.226)	0.842	0.838 (-0.004)	0.850
Amazon	0.564	0.318 (-0.246)	0.852	0.548 (-0.304)	0.776	0.798 (+0.022)	0.827	0.789 (-0.037)	0.852	0.839 (-0.013)	0.852
Bugs	0.153	0.114 (-0.039)	0.208	0.144 (-0.064)	0.162	0.285 (+0.123)	0.173	0.351 (+0.178)	0.360	0.352 (-0.008)	0.366
Coronavirus	0.226	0.205 (-0.022)	0.464	0.254 (-0.210)	0.397	0.306 (-0.091)	0.377	0.342 (-0.035)	0.660	0.628 (-0.032)	0.695
TREC	0.357	0.291 (-0.066)	0.332	0.188 (-0.144)	0.494	0.484 (-0.010)	0.556	0.593 (+0.037)	0.646	0.643 (-0.003)	0.647
WoS	0.400	0.279 (-0.122)	0.620	0.382 (-0.238)	0.536	0.600 (+0.064)	0.633	0.663 (+0.031)	0.664	0.663 (-0.001)	0.664

fications and achieves the best result, including ties, on 11 of 12 benchmarks. The only exception is Food101, where DALI obtains a slightly higher accuracy (0.764 vs. 0.763). The larger degradation relative to the standard-supervision settings suggests that limited user labels make it more difficult to distinguish model-specific reuse compatibility in the presence of corrupted learnwares.

These observations are consistent with the construction of EGS: the developer-side gradients in Eq. 2 use the associated model's predictions as targets, thereby retaining predictive information that can distinguish models trained for the same developer task. The results suggest that this information contributes to robust learnware identification when model quality varies. DALI, which also incorporates model-dependent discriminative information, improves its average accuracy after pool augmentation in all three settings, although its averages remain below those of EGS. Improvements on individual benchmarks are also observed for other baselines, indicating that the addition of corrupted candidates does not uniformly degrade reuse performance. Taken together, the results support the robustness of EGS to low-quality learnwares, while identifying limited user supervision as a more challenging condition. This evidence concerns downstream reuse accuracy and does not imply complete exclusion of corrupted learnwares during identification.

D.6 PROXY SAMPLE SIZE

Experimental Setup. We examine the sensitivity of EGS to the number of proxy samples per class by varying $l \in \{5, 10, 15, 20\}$, corresponding to a total specification size of $M = K \times l$ for a K-class task. All remaining configurations follow Appendix D.3. We evaluate each proxy size under the three reuse settings on both image and text benchmarks. The image results are presented in Fig. 2, while Fig. 3 reports the corresponding text results.

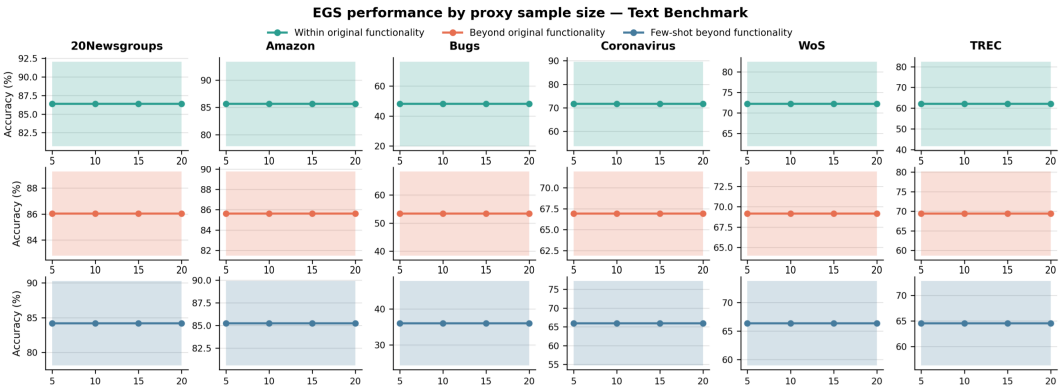


Figure 3: Effect of the number of proxy samples per class on the performance of EGS across the text benchmarks. Columns correspond to datasets and rows correspond to the three reuse settings. Curves and shaded regions denote the mean reuse accuracy and its reported standard deviation over sampled user tasks, respectively.

Performance Analysis. Across both modalities and all three reuse settings, the performance of EGS is largely insensitive to the proxy sample size. On the image benchmarks, most accuracy curves remain nearly unchanged as l increases from 5 to 20. The largest observed variation is approximately 1.46 percentage points on Cars under *Within Original Functionality*; the variations on the remaining image benchmarks are smaller, including under the more demanding few-shot setting. On the text benchmarks, the results remain unchanged at the reported precision for all four proxy sizes across every dataset and reuse setting. These results indicate that a small class-wise proxy set is sufficient to preserve the gradient information required for learnware identification and reuse. Increasing the proxy size provides no consistent performance gain, suggesting diminishing returns once the principal class-wise gradient responses have been captured. We therefore use $l = 10$ proxy samples per class as the default setting, which provides a compact specification while maintaining stable reuse performance.

D.7 Further Analysis and Limitations

In this section, we provide a statistical analysis of the performance differences among the five deployable learnware specification methods. We also discuss the limitations of our work and directions for future research.

Statistical Setup. We assess whether the performance differences among the five learnware specification methods are consistent across twelve benchmarks. For each learnware reuse setting, the mean accuracy on each of the twelve benchmarks is treated as one paired observation. We first apply the Friedman test (Friedman, 1940; Demšar, 2006) to RKME, CRKME, LANE, DALI, and EGS, using datasets as blocks, and report Kendall's W as the corresponding effect size. After rejecting the omnibus null hypothesis, we compare EGS with each baseline using an exact two-sided Wilcoxon signed-rank test (Wilcoxon, 1945). Tied pairs with zero difference are excluded from the corresponding signed-rank calculation. To control the family-wise error rate, the resulting twelve pairwise p-values—four baselines under each of the three reuse settings—are jointly adjusted using Holm's procedure (Holm, 1979). Statistical significance is determined at $\alpha = 0.05$. MAX and Oracle are excluded because they are non-deployable references based on downstream performance.

Results. The Friedman tests reject equal performance in all three reuse settings ($p < 2 \times 10^{-5}$), with Kendall's W ranging from 0.564 to 0.704, indicating that the differences among methods are consistent across benchmarks. The post-hoc comparisons further show that EGS significantly outperforms every deployable baseline after Holm correction. Against RKME, CRKME, and LANE, the adjusted p-value is 0.0059 in each setting. The comparison with the strongest baseline, DALI, also remains significant in all three settings ($p_{\text{Holm}} = 0.0410$). The median paired improvements over DALI are 0.55, 1.45, and 1.60 percentage points under *Beyond Original Functionality*, *Within Original Functionality*, and *Few-Shot Beyond Original Functionality*, respectively. Together with the corresponding win/tie/loss counts of $6/4/2$, $7/4/1$, and $9/2/1$, these results show that the aver-

age improvements of EGS are supported by a consistent advantage across datasets rather than by a small number of isolated benchmarks.

Table 8: Statistical significance across the twelve benchmarks. The Friedman test compares all five deployable specification methods within each reuse setting. Pairwise entries report Holm-adjusted p-values from exact two-sided Wilcoxon signed-rank tests comparing EGS with each baseline.

Reuse Setting	**Friedman** χ^2	**Omnibus** p	**Kendall's** W	**vs. RKME**	**vs. CRKME**	**vs. LANE**	**vs. DALI**
Beyond Original Functionality	30.21	4.43×10^{-6}	0.629	0.0059	0.0059	0.0059	0.0410
Within Original Functionality	27.07	1.92×10^{-5}	0.564	0.0059	0.0059	0.0059	0.0410
Few-Shot Beyond Original Functionality	33.81	8.17×10^{-7}	0.704	0.0059	0.0059	0.0059	0.0410

Limitations. Although EGS achieves strong and robust performance in learnware identification and reuse, its advantage is less pronounced on fine-grained image benchmarks such as Cars, where the shared *ResNet110* representation may not sufficiently capture subtle inter-class differences. The method also faces several limitations, including difficult proxy optimization in high-dimensional spaces, additional computational overhead from iterative optimization, and dependence on shared representations. Unlike PAVE (Shi et al., 2026), EGS does not rely on large-scale pre-trained foundation models or task-specific adaptation, making direct comparison under our controlled experimental protocol less informative. These limitations motivate future work on more effective proxy optimization, broader task support, and more transferable shared representations to overcome reliance on specific feature extractors.