# Learnware Specification via Label-Aware Neural Embedding

**Wei Chen**[1, 2], **Jun-Xiang Mao**[1, 2, 3], **Min-Ling Zhang**[1, 2*]

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China
[2] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
[3] Information Technology and Data Management Department of China Mobile Communications Group Zhejiang Co., Ltd
wei-chen@seu.edu.cn, maojx@seu.edu.cn, zhangml@seu.edu.cn

## Abstract

The *learnware paradigm* aims to establish a *learnware dock system* of numerous well-trained machine learning models, enabling users to reuse existing helpful models for their tasks instead of starting from scratch. Each learnware in the system is a well-established model submitted by its developer, associated with a *specification* generated by the learnware dock system. The specification characterizes the specialty of the corresponding model, enabling it to be identified accurately for new task requirements. Existing specification generation methods are mostly based on the ***R**educed **K**ernel **M**ean **E**mbedding* (RKME) technique, which uses the **M**aximum **M**ean **D**iscrepancy (MMD) in the ***R**eproducing **K**ernel **H**ilbert **S**pace* (RKHS) to seek a reduced set that characterizes the model's capabilities. However, existing RKME-based methods mainly utilize feature information to generate specifications by assuming the existence of the ground-truth labeling function, while leaving the label information, which is capable of providing rich semantic characterization, untouched. Furthermore, the quality of the generated specifications heavily relies on the choice of the kernels, which makes it prohibitive to adapt to all real-world scenarios. In this paper, to overcome the above limitations, we propose a novel specification approach named LANE, i.e., ***L**abel-**A**ware **N**eural **E**mbedding*. In LANE, the neural embedding space is utilized to replace the RKHS, effectively circumventing the step of kernel selection and thereby addressing the dependency on kernels in existing RKME-based specification methods. More importantly, LANE uses the label information as additional supervision to enhance the generation process, resulting in specifications of superior quality. Extensive experiments demonstrate the effectiveness and superiority of the proposed LANE approach in the learnware paradigm.

## Introduction

In recent years, advancements in machine learning technology and its growing real-world applications have given rise to a multitude of models (Jordan and Mitchell 2015). However, the traditional machine learning paradigm requires a step-by-step construction of the model from scratch tailored to each task. This approach demands a large amount of high-quality training data (Zhu et al. 2016), costly computational resources (Menghani 2023), and proficient training skills (Mumuni, Mumuni, and Gerrar 2024), thus making the process highly intricate and expensive (Sarker 2021). Furthermore, concerns surrounding data privacy and proprietary restrictions significantly impede users from developing and reusing models across varying tasks.

To address these challenges concurrently, the *learnware paradigm* has been introduced (Zhou 2016). This paradigm aims to establish a *learnware dock system* of numerous well-trained machine learning models, enabling users to reuse existing helpful models for their tasks instead of starting from scratch (Zhou and Tan 2024). Each learnware in the dock system is a well-established model associated with a *specification*, which characterizes the specialty of the corresponding model in the submitting stage, enabling it to be identified accurately for new task requirements in the deploying stage. To achieve this vision, the key challenge is: How to generate specifications that represent the ability and specialty of models without exposing the developers' training data, and leverage these specifications to identify and even reassemble a few helpful learnware's models to tackle new tasks?

Reviewing the entire learnware paradigm, it becomes evident that the quality of the specification is a critical factor influencing model reuse. Existing specification generation methods are mostly based on the ***R**educed **K**ernel **M**ean **E**mbedding* (RKME) technique (Wu et al. 2023; Zhou and Tan 2024), which uses the **M**aximum **M**ean **D**iscrepancy (MMD) in the ***R**eproducing **K**ernel **H**ilbert **S**pace* (RKHS) to seek a reduced set that characterizes the model's capabilities. Through the RKME technique, users can efficiently identify useful learnwares by aligning their requirement data with the generated specification, all while preserving privacy (Lei, Tan, and Zhou 2024). Recently, the first open-source learnware dock system, *Beimingwu*, has been successfully developed and released (Tan et al. 2024c).

Although existing RKME-based specification methods for learnware retrieval and model reuse have proven effective in many application scenarios (Zhang et al. 2021; Guo et al. 2023; Xie et al. 2023; Tan et al. 2024a). such methods mainly utilize feature information to generate specifications by assuming the existence of the ground-truth labeling function, while leaving the label information, which is capable of providing rich semantic characterization, untouched. Furthermore, the quality of the generated specifications heavily relies on the choice of the kernels, which makes it pro-

*Corresponding author

hibitive to adapt to all real-world scenarios.

To overcome the above limitations of existing RKME-based specification methods, we propose a novel specification approach named LANE, i.e., *learnware specification via **L**abel-**A**ware **N**eural **E**mbedding*. Specifically, in the submitting stage, we leverage both feature and label information of the training data to model intra-class feature distributions through numerous random neural networks, mapping them to the neural embedding space. Subsequently, we implement distribution matching to ensure the specification accurately reflects the inherent characteristics of the dataset. In the deploying stage, we devise a fine-grained method to identify useful learnwares based on the proposed LANE specification, which incorporates the intra-class feature information retained by the specification into the mixture weights estimation. This enables the correlation between the user requirements and the learnwares to be effectively captured in the neural embedding space. Different from existing RKME-based specification methods, in LANE, the neural embedding space is utilized to replace the RKHS, effectively circumventing the step of kernel selection and thereby addressing the dependency on kernels in existing methods. More importantly, LANE uses the label information as additional supervision to enhance the generation process, resulting in specifications of superior quality. Extensive experiments demonstrate the effectiveness and superiority of the proposed LANE approach in the learnware paradigm.

The rest of this paper is organized as follows. Section 2 briefly reviews related works. Section 3 introduces preliminaries required to understand our proposed approach. Section 4 presents the details of the proposed LANE approach. Section 5 reports the experimental results of comparative studies. Section 5 concludes the paper.

## Related Works

Learnware (Zhou 2016) introduces a novel paradigm that differs from the traditional machine learning paradigm. This paradigm enables users to identify learnwares, consisting of both a well-established model and its specification, in the learnware dock system according to their task requirements, and then reuse the models they need. The specification is a critical element of the learnware paradigm, and it characterizes the model's capabilities through semantic descriptions/labels and statistical representations (Zhou and Tan 2024). Moreover, the specification must ensure the comprehensibility of the model while safeguarding data inaccessibility. To this end, the RKME specification method was proposed to generate specifications and support the learnware paradigm (Wu et al. 2023), and it was also demonstrated in (Lei, Tan, and Zhou 2024) that RKME specification process privacy-preserving capabilities. Recently, substantial progress has been made in this direction across various machine learning scenarios. For example, in the heterogeneous feature space scenarios, Tan et al.(2024a) strengthened the characterization of model capabilities through RKME specification in a unified subspace, utilizing the conditional distributions induced by label information, and further enhanced learnware identification by additionally matching conditional distributions. Furthermore, Tan et al.(2024b) collected auxiliary data from the entire feature space, enabling the mapping of heterogeneous data into a unified subspace for reduction. Subsequently, (Tan et al. 2023) investigated the organization and utilization of heterogeneous learnware Dock system without the need to access the original or auxiliary data across feature spaces. Zhang et al.(2021) extended RKME specification to handle user tasks with unseen parts, while Guo et al.(2023) addressed the issue of heterogeneous label spaces by incorporating a linear proxy model into the RKME specification. However, existing methods require identifying useful learnwares one by one across the learnware dock system. Xie et al.(2023) utilize the similarity of specifications, represented by distances in the RKHS, to construct learnware anchors, thereby accelerating the identification efficiency. With the increasing number of learnwares available, an evolvable learnware specification with an index-based approach was proposed to address the challenge of evaluating a model's capacity to exceed its original training task, allowing for more accurate and efficient identification of useful learnwares (Liu, Tan, and Zhou 2024).

Based on the aboved research, the first learnware dock system, *Beimingwu* has been developed and released (Tan et al. 2024c). The system streamlines the entire learnware process and provides a highly scalable architecture, facilitating the implementation of future research.

These RKME-based studies generate specifications by approximating data distribution via pre-defined kernel functions, which are hard to generalize for all real-world scenarios. In this paper, we propose a novel learnware specification approach, LANE, making an attempt to incorporate label information from the training data and eliminate the dependency on kernels by introducing the neural embedding.

## Preliminaries

### Learnware Paradigm

The learnware paradigm comprises two distinct stages: a submitting stage and a deploying stage.

**The Submitting Stage.** In this stage, suppose there are $c$ developers submitting their well-established models to the learnware dock system for future use. The $i$-th developer has access to a private local dataset $\mathcal{D}_i = \{(\mathbf{x}_{i,n}, y_{i,n})\}_{n=1}^{N_i}$, representing the specific task $T_i$. Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ denote the output space. The task $T_i = (P_i, f_i)$ includes a data distribution $P_i$ of the dataset $\mathcal{D}_i$ on the input space $\mathcal{X}$ and a well-established model $f_i$: $\mathcal{X} \to \mathcal{Y}$, such that:

$$\forall i \in [c], \forall j \in [N_i], \forall (\mathbf{x}_{i,j}, y_{i,j}) \in \mathcal{D}_i, f_i(\mathbf{x}_{i,j}) = y_{i,j}, \quad (1)$$

where $[c]$ denotes the set $\{1, \cdots, c\}$.

When the learnware dock system receives an uploaded task $T = (P, f)$, it generates a specification $\boldsymbol{z}$ corresponding to the model $f$, aiming to approximate $P_{\boldsymbol{z}}$, the distribution of the specification $\boldsymbol{z}$, as closely as possible to the data distribution $P$ in order to capture the model's specialty, i.e.,

$$\min_{\boldsymbol{z}} \|\mu(P) - \mu(P_{\boldsymbol{z}})\|^2, \quad (2)$$

where $\mu(\cdot)$ is a formalized distribution function, such as the Gaussian distribution, characterizing a specific distribution

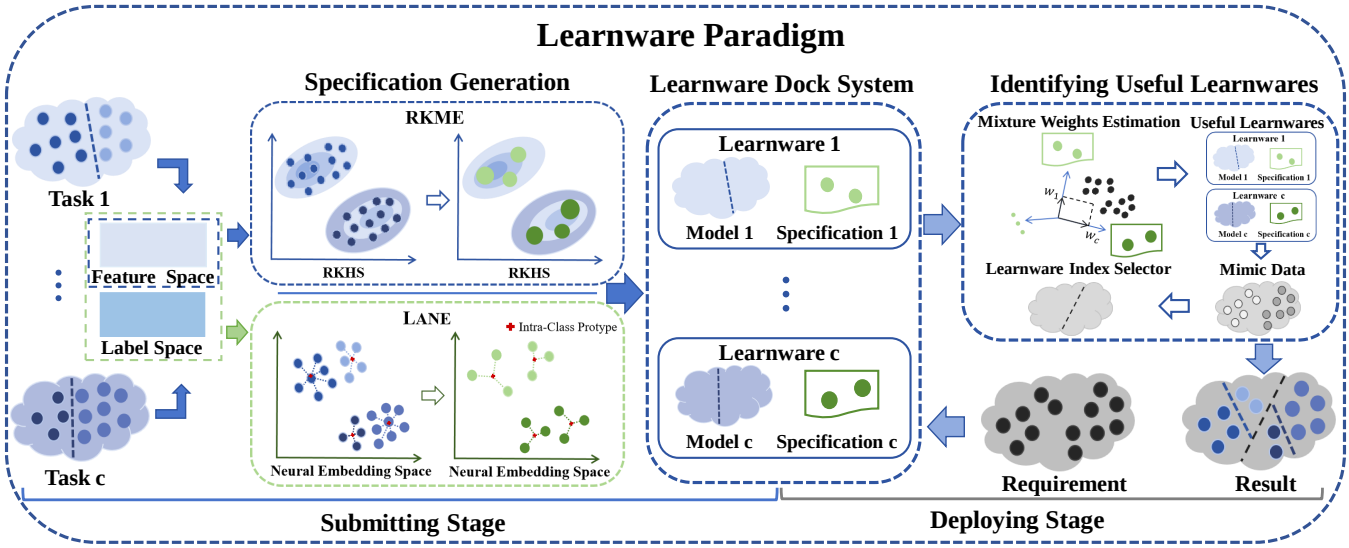



Figure 1: The learnware paradigm of LANE and RKME. Note that the process of identifying useful learnware is performed in the corresponding embedding space through the respective specification (LANE or RKME) and the given requirement. In addition, to ensure effective differentiation of data for different tasks and requirements, a global deep learning feature extractor is incorporated into the learnware paradigm, which maps all data to a unified feature space.

pattern. Subsequently, the learnware $\{f, P_{\boldsymbol{z}}\}$ is created and stored in the learnware dock system for future use.

**The Deploying Stage.** In the traditional machine learning paradigm, different task scenarios generally require different datasets. The characteristics of different datasets can be utilized to differentiate various well-established models. Therefore, the model $f$ corresponding to task $T = (P, f)$ in the learnware dock system can be reused on new task $\hat{T} = (P, \hat{f})$, which has the same data distribution as the training data corresponding to $f$ (Tan et al. 2024c), i.e.,

$$\forall i \in [c], \mathbb{E}_{\mathbf{x}\sim P_i}\left[\mathcal{L}(f(\mathbf{x}), \hat{f}(\mathbf{x}))\right] \leq \epsilon, \tag{3}$$

where $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the loss function such as mean-square error (MSE) for measuring the difference of the two models, and $\epsilon$ is a positive real number close to zero.

In this stage, the user hopes to receive some helpful models from the learnware dock system to handle her task while ensuring data security, i.e., without disclosing her raw data. These expected models will enable the user to predict her dataset $\hat{\mathcal{D}} = \{\hat{\mathbf{x}}_n\}_{n=1}^{\hat{N}}$, which contains unknown labels and is sampled from a distribution $P_{\hat{\mathbf{x}}}$. The learnware paradigm then identifies useful learnwares by calculating the correlating the requirements with existing $c$ specifications in the dock system through the mixture weights estimation method, i.e.,

$$\min_{w_1,w_2,\ldots,w_c} \left\| \mu(P_{\hat{\mathbf{x}}}) - \sum_{i=1}^{c} w_i \mu(P_{\boldsymbol{z}_i}) \right\|^2, \tag{4}$$

where $w_i$ indicates the degree of correlation between the requirement and the $i$-th specification. Thus, suitable learnwares that meet the user's requirement can be identified and then the helpful models in the dock system can be reused.

To maintain the privacy-preserving nature of the learnware paradigm, the requirement data cannot be accessed directly. Instead, as the combinatorial distribution of the useful learnware specifications closely mirrors that of the requirement dataset, the paradigm can generate a mimic dataset by sampling from these specifications. This mimic dataset allows the useful learnwares to determine which portions of the requirement they can address by training a learnware index selector, or classifier. Consequently, the paradigm enables model reuse by having the useful learnwares train on the relevant portions of the requirement, thereby generating the desired results $\hat{Y} = \{\hat{y}_n\}_{n=1}^{\hat{N}}$.

## Reduced Kernel Mean Embedding Specification

The RKME-based specification method is derived from the combination of kernel mean embedding (KME) and reduced set techniques. The core idea of RKME specification is to use the reduced set containing minor weighted samples $\{(\beta_m, \boldsymbol{z}_m)\}_{m=1}^{M}$ to approximate the empirical KME of the original dataset $\{\mathbf{x}_n\}_{n=1}^{N}$. This method is well-suited for generating learnware specifications, as it not only captures the feature distribution of the training data but also ensures privacy preservation. Specific examples are illustrated in Figure 1. In the submitting stage of learnware paradigm, Eq.(2) is reformulated by RKME specification as

$$\min_{\boldsymbol{\beta},\mathbf{Z}} \left\| \sum_{n=1}^{N} \frac{1}{N} k(\mathbf{x}_n, \cdot) - \sum_{m=1}^{M} \beta_m k(\boldsymbol{z}_m, \cdot) \right\|_{\mathcal{H}_k}^2. \tag{5}$$

Here, $M \ll N$, $k(\cdot,\cdot)$ is the pre-defined kernel function corresponding to the RKHS $\mathcal{H}_k$, $\boldsymbol{\beta} = \{\beta_m\}_{m=1}^{M}$ and $\mathbf{Z} = \{\boldsymbol{z}_m\}_{m=1}^{M}$ are combined as the reduced set $\{(\beta_m, \boldsymbol{z}_m)\}_{m=1}^{M}$. Let $\Phi(\cdot) = \sum_{m=1}^{M} \beta_m k(\boldsymbol{z}_m, \cdot)$, the learnware is denoted as

$\{f, \Phi(\cdot)\}$ in the RKME-based specification method. Then, in the deploying stage, Eq.(4) is rewritten as

$$\min_{w_1,w_2,\ldots,w_c} \left\| \sum_{n=1}^{\hat{N}} \frac{1}{\hat{N}} k(\hat{\mathbf{x}}_n, \cdot) - \sum_{i=1}^{c} w_i \Phi_i(\cdot) \right\|_{\mathcal{H}_k}^2 . \quad (6)$$

Here, $\Phi_i$ denotes the $i$-th leanware in the dock system. Eq.(6) ensures that both the requirement and existing specifications from the learnware dock system reside whithin the same RKHS space, allowing for the identification of useful learnwares. Correspondingly, leveraging the RKME specification, data that mimics the requirement distribution is synthesized in the RKHS using kernel herding techniques (Chen, Welling, and Smola 2010; Chen 2013). Finally, the learnware index selector, developed by the classifier, identifies the portion of the requirement that can be effectively addressed by the model of useful learnwares. It subsequently facilitates model reuse to generate the results in response to the requirement.

## The Proposed LANE Approach

The proposed LANE approach is illustrated in Figure.1. By considering the rich semantic characterization associated with label information, the LANE approach maps labeled data into the neural embedding space using numerous random neural networks. This ensures that the generated specification preserves the intra-class feature distribution of the task dataset. Subsequently, user requirements can utilize the LANE specification to identify useful learnwares in the neural embedding space and achieve model reuse.

### The Submitting Stage

For the learnware paradigm, the submitting stage involves the developer submitting the pre-trained model $f$ and the dataset $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ to generate the corresponding specification. The model and the specification then constitute the learnware to be submitted to the dock system.

To overcome the limitations of existing RKME-based specification methods, the LANE approach incorporates label information as supplementary guidance, allowing the specification in the neural embedding space to better approximate the intra-class feature distributions within the dataset. This ensures that the statute accurately captures both the model's specialty and inherent characteristics of the dataset. Moreover, since ground-truth intra-class feature distributions are inaccessible, we rely on empirical estimates derived from **M**aximum **M**ean **D**iscrepancy (MMD) (Gretton et al. 2012) within the neural embedding space to achieve this. Therefore, Eq.(2) can be rewritten as:

$$\min_{\mathcal{Z}} \sum_{k=1}^{K} \left\| \frac{1}{|B_k^{\mathcal{D}}|} \sum_{\mathbf{x} \in B_k^{\mathcal{D}}} \psi_\upsilon(\mathbf{x}) - \frac{1}{|B_k^{\mathcal{Z}}|} \sum_{\boldsymbol{z} \in B_k^{\mathcal{Z}}} \psi_\upsilon(\boldsymbol{z}) \right\|^2 , \quad (7)$$

where $K$ is the number of classes in the dataset $\mathcal{D}$, $B_k^{\mathcal{D}}$ represents the mini-batch of the $k$-th class in dataset $D$, and $B_k^{\mathcal{Z}}$ denotes the mini-batch of the $k$-th class in specification $\mathcal{Z}$. Here, the specification $\mathcal{Z}$ is defined as $\{(\boldsymbol{z}_m, y_{\boldsymbol{z}_m})\}_{m=1}^{M}$, and initialized by randomly sampling from the data distribution $P_{\mathbf{x}}$ of the dataset $\mathcal{D}$ while maintaining the same of classes. Subsequently, the learnware, defined as $\{f, \mathcal{Z}\}$, is created and stored in the learnware dock system for future use.

Let $\mathcal{F}$ denotes the objective function of Eq.(7), the complete submitting stage of the proposed LANE approach is then summarized in Algorithm 1. Firstly, the specification is initialized through random sampling from the developer dataset $\mathcal{D}$ (Step 1). After that, The specification is generated through an optimization procedure according to Eq.(7) (Step 2-7). Finally, the model $f$ and the generated specification $\mathcal{Z}$ are utilized to create the corresponding learnware (Step 8).

Algorithm 1: The submitting stage of our LANE approach

**Input**: Developer dataset $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ and pre-trained model $f$.
**Parameter**: Neural network $\psi_\upsilon$ parameterized with $\upsilon$, random probability distribution over parameters $P_\upsilon$, iteration $T$, and learning rate $\eta$.
**Output**: Learnware $\{f, \mathcal{Z}\}$.

1: Randomly initialize the specification $\mathcal{Z} \sim P_{\mathbf{x}}$, with each class having a size of $M$;
2: **for** $t = 1$ to $T$ **do**
3: Sample $\upsilon \sim P_\upsilon$;
4: Sample mini-batches $B_k^{\mathcal{D}} \sim \mathcal{D}$ and $B_k^{\mathcal{Z}} \sim \mathcal{Z}$ for each class;
5: Compute the objective function $\mathcal{F}$ of Eq.7;
6: Update $\mathcal{Z} \leftarrow \mathcal{Z} - \eta \nabla_{\mathcal{Z}} \mathcal{F}(\mathcal{Z})$;
7: **end for**
8: Combine the pre-trained model $f$ and the generated specification $\mathcal{Z}$ to form the learnware $\{f, \mathcal{Z}\}$;
9: **return** The corresponding learnware $\{f, \mathcal{Z}\}$.

It is also worth noting that $\psi_\upsilon$ is a strategy that uses random sampling of numerous random neural network parameters, i.e., $\upsilon \sim P_\upsilon$, where $P_\upsilon$ is the random probability distribution over parameters (Zhao and Bilen 2021). Since random neural network have been demonstrated to perform distance-preserving embedding of data, i.e., smaller distances between data of the same class and larger distances between data of the different classes (Giryes, Sapiro, and Bronstein 2016), we learn the generated specification by sampling to minimize the difference between the two feature distributions in neural network embedding spaces. In contrast to the RKME specification methods, which use a kernel function to fit the distribution in the RKHS, the proposed LANE approach employs numerous random neural networks to learn the feature distribution in the neural embedding space. This approach effectively addresses the dependency on pre-defined kernel functions and is more generalizable.

As stated in Proposition 4.3 of (Dong, Zhao, and Lyu 2022), Eq.7 guarantees that the dataset $\mathcal{D}$'s center of gravity aligns with that of the generated specification $\mathcal{Z}$. Moreover, given that the specification is initialized through random sampling from $\mathcal{D}$, i.e., $\forall(\boldsymbol{z}, y_{\boldsymbol{z}}) \in \mathcal{Z}, \boldsymbol{z} \sim P_{\mathbf{x}}$, combined with Proposition 4.3 and 4.4 in (Dong, Zhao, and Lyu 2022), we can conclude that the generated specification is capable of representing the dataset $\mathcal{D}$ with a small amount of data.

Furthermore, according to Proposition 4.8-4.10 in (Dong, Zhao, and Lyu 2022), the privacy bounds of Eq.7 are analyzed using the membership inference attacks (Kairouz, Oh, and Viswanath 2015), directly relating to the differential privacy (Dwork et al. 2006). These theoretical results clearly demonstrate the privacy-preserving nature of our proposed LANE specification approach.

## The Deploying Stage

During the submitting stage of the proposed LANE approach, the learnware dock system is enriched with a variety of leanwares. In the deploying stage, users can submit their requirement $\hat{\mathcal{D}}$ to the leanware dock system, identifying appropriate learnwares and reusing helpful models.

In order to identify appropriate learnwares in the dock system, the first step is to estimate the degree of correlation between the requirement and the existing learnware specifications. Given that the LANE specification encompasses intra-class feature distributions and is generated within the neural embedding space, we conduct mixture weights estimation in this space while concurrently considering the intra-class feature distribution information of the specification. Accordingly, Eq.(4) is reformulated as follows:

$$\min_{\mathbf{W}} \left\| \frac{1}{\hat{N}} \sum_{n=1}^{\hat{N}} \psi_v(\hat{\mathbf{x}}_n) - \sum_{i=1}^{c} \sum_{j=1}^{K_i} w_{i,j} \text{mean}(\psi_v(\mathcal{Z}_{i,j})) \right\|^2, \tag{8}$$

where

$$\text{mean}(\psi_v(\mathcal{Z}_{i,j})) = \frac{1}{|\mathcal{Z}_{i,j}|} \sum_{(\boldsymbol{z},y_{\boldsymbol{z}})\in\mathcal{Z}_{i,j}} \psi_v(\boldsymbol{z}),$$

$K_i$ is the number of classes in the $i$-th learnware specification, $\mathcal{Z}_{i,j}$ and $w_{i,j}$ denote the specification and the weight for the $j$-th class of the $i$-th learnware, respectively. Here, $\mathbf{W} = [w_{1,1}, w_{1,2}, \cdots, w_{c,K_c}]^\top \in \mathbb{R}^{N_w}$, where $N_w = \sum_{i=1}^{c}\sum_{j=1}^{K_i} 1$, is the mixture weight vector. The optimization problem in Eq.(8) can be equivalently reformulated as the following quadratic programming problem:

$$\begin{aligned} \min_{\mathbf{W}} \ & \frac{1}{2}\mathbf{W}^\top \mathbf{H} \mathbf{W} + \mathbf{C}^\top \mathbf{W} \\ \text{s.t.} \ & \mathbf{1}_{N_w}^\top \mathbf{W} = 1, \end{aligned} \tag{9}$$

where

$$\mathbf{H} = \begin{bmatrix} H_{11,11} & H_{11,12} & \dots & H_{11,cK_c} \\ H_{12,11} & H_{12,12} & \dots & H_{12,cK_c} \\ \vdots & \vdots & \ddots & \vdots \\ H_{cK_c,11} & H_{cK_c,12} & \dots & H_{cK_c,cK_c} \end{bmatrix} \in \mathbb{R}^{N_w \times N_w},$$

$$H_{ij,hl} = \text{mean}(\psi_v(\mathcal{Z}_{i,j}))^\top \text{mean}(\psi_v(\mathcal{Z}_{h,l})),$$

$$\mathbf{C} = [C_{1,1}, C_{1,2}, \cdots, C_{c,K_c}]^\top \in \mathbb{R}^{N_w},$$

$$C_{i,j} = \frac{1}{\hat{N}} \sum_{n=1}^{\hat{N}} \text{mean}(\psi_v(\mathcal{Z}_{i,j}))^\top \psi_v(\hat{\mathbf{x}}_n),$$

and $\mathbf{1}_{N_w}$ is an all 1 column vector with size $N_w$.

Then, we can obtain the mixture weights $\mathbf{W}$ by optimizing the quadratic programming problem Eq.(9) with any off-the-shelf solvers (Vandenberghe 2010). In this way, the degree of correlation between the user requirement and the $i$-th learnware can be achieved by calculating $\sum_{j=1}^{K_i} w_{i,j}$.

Moreover, to maintain the privacy-preserving nature of the learnware paradigm, we employ the intra-class information of the specification and multivariate normal distribution sampling technique (Williams and Rasmussen 2006) to generate the mimic data $S$ that approximates user requirement data, i.e., intra-class mean $\text{mean}(\cdot)$ and intra-class covariance $\mathbf{\Sigma}$. In addition, to mitigate the issue of data imbalance arising from using $w_{i,j}$ to determine the number of mimic data, the degree of correlation for each useful learnware (i.e., $\sum_{j=1}^{K_i} w_{i,j}$) is applied to ascertain the corresponding amount of mimic data. The foundation for the mimic data sampling is:

$$\{X_{i,j}^{\text{mimic}}\}_{n=1}^{N_{i,j}^{\text{mimic}}} \sim \mathcal{N}(\text{mean}(\mathcal{Z}_{i,j}), \mathbf{\Sigma}_{i,j}), \tag{10}$$

where $N_{i,j}^{\text{mimic}} = \hat{N} \cdot \sum_{j=1}^{K_i} w_{i,j}$. The obtained mimic data is utilized to train a learnware index selector $g(\cdot)$ (e.g., SVM). Using this selector, the model identifies relevant parts of the requirement, enabling it to be trained to achieve the desired result $\hat{Y}$ and facilitating model reuse.

The detailed procedure of the deploying stage based on LANE specification is shown in Algorithm 2. First of all, mixture weights estimation is performed in the neural embedding space to obtain $\mathbf{W}$ (Step 1). Then, the mimic dataset $S$ is generated (Step 2-10). Thereafter, a learnware index selector $g(\cdot)$ is trained on $S$ (Step 11). Finally, the selector $g(\cdot)$ predicts the learnware index for each data point in $\hat{\mathcal{D}}$, and the results $\hat{Y}$ are generated by using the corresponding learnware model for predictions (Steps 12-15).

Algorithm 2: The deploying stage of our LANE approach

**Input**: User requirement dataset $\hat{\mathcal{D}} = \{\hat{\mathbf{x}}_n\}_{n=1}^{\hat{N}}$ and Learnware dock system $\{f_i, \mathcal{Z}_i\}_{i=1}^{c}$.
**Output**: Prediction $\hat{Y}$

1: Optimize Eq.(9) to estimate $\mathbf{W}$;
2: Initialize the mimic dataset $S = \emptyset$;
3: **while** $|S| < c\hat{N}$ **do**
4: Sample a learnware index $i$ by weight $\sum_{j=1}^{K_i} w_{i,j}$;
5: **for** $j = 1$ to $K_i$ **do**
6: Compute $\text{mean}(\mathcal{Z}_{i,j})$ and $\mathbf{\Sigma}_{i,j}$;
7: Sample $\{X_{i,j}^{\text{mimic}}\}_{n=1}^{N_{i,j}^{\text{mimic}}}$ according to Eq.(10);
8: $S = S \cup \{X_{i,j}^{\text{mimic}}\}_{n=1}^{N_{i,j}^{\text{mimic}}}$ ;
9: **end for**
10: **end while**
11: Train a selector $g(\cdot)$ on mimic dataset $S$;
12: **for** $n = 1$ to $\hat{N}$ **do**
13: Assign $\hat{\mathbf{x}}_n$ to selector $g(\cdot)$ to get learnware index $\hat{i}$;
14: Predict $\hat{y}_n = f_{\hat{i}}(\hat{\mathbf{x}}_n)$;
15: **end for**
16: **return** $\hat{Y} = \{\hat{y}_n\}_{n=1}^{\hat{N}}$.

# Experiments

## Experimental Setup

**Datasets.** We evaluate the proposed LANE specification approach on two benchmark dataset of different data types: *CIFAR100* (Krizhevsky 2009) for image and *20newsgroups* (Joachims 1997) for text. *CIFAR100* contains 100 classes, which are grouped into 20 superclasses, while *20newsgroups* encompasses 5 superclasses, encompassing 20 classes. The superclass structure inherent in these two datasets is highly suitable for evaluating the learnware paradigm. Specifically, each superclass dataset can serve as a task during the submitting stage. In this stage, models that have been trained on these tasks, along with their specifications, are combined to create learnwares, which are then submitted to the learnware dock system. In the deploying stage, if the user's task requirement is associated with only one dataset, then this dataset will randomly select from the classes included in a single superclass. However, if the requirement is associated with multiple datasets, these datasets will be screened from various superclasses.

**Implementation Details.** To ensure the fairness of the comparative experiments and eliminate the influence of different deep learning feature extractors, both the LANE and RKME approaches in the learnware paradigm utilize *ResNet110*[1] for images and *Sentence Transformer*[2] for text as the global feature extractors. Additionally, the random neural network $\psi_v$ in the LANE approach is set to *ConvNetBN* (Rawat and Wang 2017), and all pre-trained models are obtained using the SVM method. All pre-trained models, tasks, and requirements are shared, with the number of requirements fixed at 50 for each mixed task.

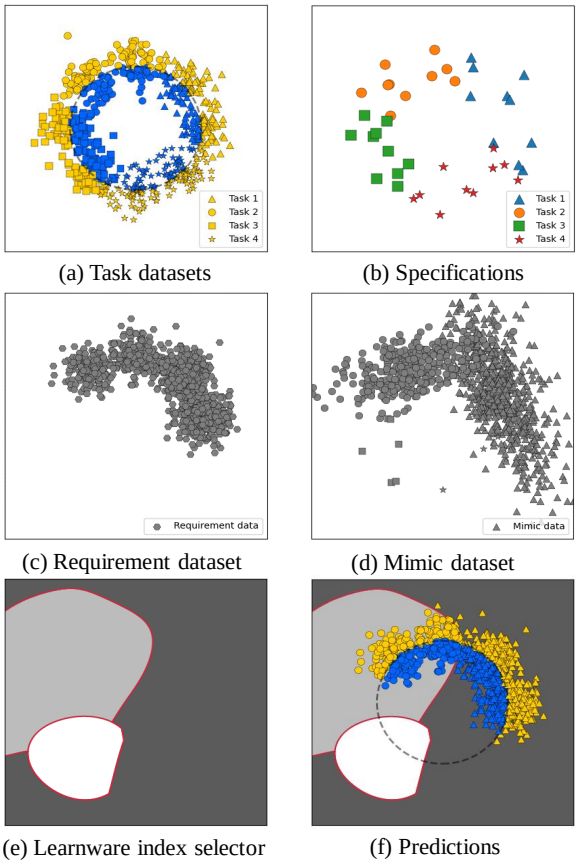
## Comparative Studies

To validate the effectiveness and superiority of the proposed LANE approach in the learnware paradigm, we conduct comparative studies between the LANE approach and three compared methods: a naive baseline, MAX, alongside a related method, HMR (Wu, Liu, and Zhou 2019), and a specification method, RKME, using the benchmark dataset.

| Mixed tasks | 1 | 2 | 5 | 10 | 20 |
|---|---|---|---|---|---|
| MAX | 51.69 | 52.50 | 52.35 | 51.15 | 52.15 |
| HRM | 67.33 | 68.50 | 67.93 | 67.32 | 67.58 |
| RKME | 84.04 | 81.75 | 77.22 | **73.02** | 66.27 |
| LANE | **84.50** | **83.40** | **78.43** | 72.51 | **68.13** |

Table 1: Results of CIFAR100 in Accuracy(%).

Both RKME and LANE follow the two-stage procedure in the learnware paradigm. In the submitting stage, specifications from the superclass dataset are integrated with pre-trained models to create learnwares, which are then submitted to a model pool, i,e., the learnware dock system. During the deploying stage, the pool will be accessible to user requirements to identify valuable learnwares. In contrast, the MAX method directly uses all pre-trained models in the model pool to select the most suitable class. The HMR method employs a communication protocol that exchanges a limited number of key samples, known as the example communication budget (Wu, Liu, and Zhou 2019), to update the models, making predictions similarly to the MAX method. It is notation that this comparison is inherently unfair, as MAX and HMR are not privacy-preserving approaches, whereas RKME and LANE achieve superior or competitive performance without exposing any raw data points.




(a) Task datasets (b) Specifications (c) Requirement dataset (d) Mimic dataset (e) Learnware index selector (f) Predictions

Figure 2: Synthetic examples visualization for learnware paradigm via LANE specification.

| Mixed tasks | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| MAX | 39.62 | 41.26 | 42.75 | 42.92 | 41.50 |
| HMR | 54.43 | 54.72 | 56.24 | 56.17 | 54.32 |
| RKME | 73.83 | 69.69 | 67.75 | **66.27** | **64.78** |
| LANE | **74.29** | **70.73** | **68.12** | 66.20 | 64.28 |

Table 2: Results of 20newsgroup in Accuracy(%).

[1]ResNet110 is trained by using the command provided at https://github.com/bearpaw/pytorch-classification/blob/master/TRAINING.md

[2]More detailed descriptions on Sentence Transformer please refer to https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2/tree/main

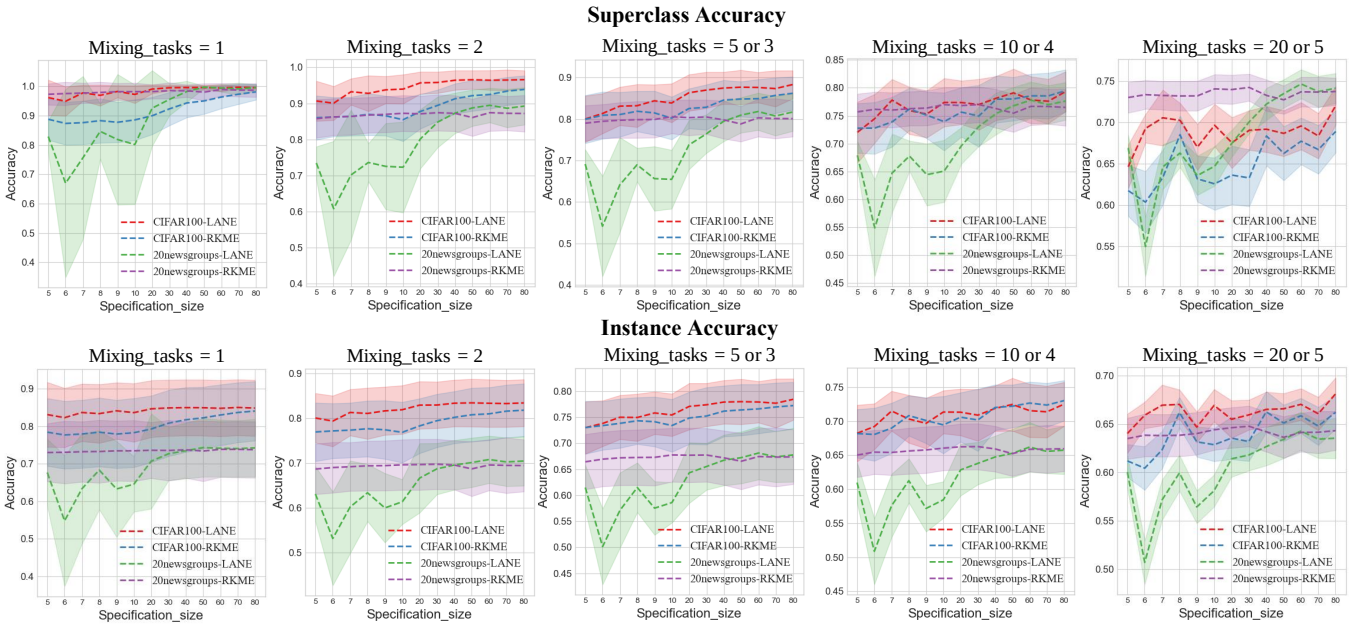



Figure 3: The impact of specification size of accuracy on superclasses and instances.

The parameter settings for the compared methods are as follows: The RKME method uses a Gaussian kernel function ($\gamma = 0.01$) and the size of the specification is set to 10; the example communication budget in the HMR method is set to 10. Empirical results related to the accuracy of model reuse on *CIFAR100* and *20newsgroups*, based on different numbers of mixed tasks, are reported in Table 1 and Table 2, respectively. The results clearly demonstrate the effectiveness and superiority of our LANE approach in model reuse.

## Further Analysis

**Visualization and Validation.** To further demonstrate the effectiveness of the proposed LANE approach, we validate the ability of model reuse in the learnware paradigm with LANE specification. Firstly, we create developer datasets for binary classification tasks derived from four synthetic datasets, as depicted in Figure 2(a). Figure 2(b) illustrates the specifications, with a size of 5 for each class, effectively preserving the intra-class feature distributions. During the deploying stage, we compose a user requirement from a mixture of two similar datasets, as detailed in Figure 2(c). The real mixture weights for the requirement datasets are $(0.3, 0.7, 0.0, 0.0)$, whereas the estimated mixture weights based on the LANE specification are approximately $(0.3021, 0.6827, 0.0083, 0.0069)$, demonstrating the effectiveness of LANE in identifying useful learnwares. Subsequently, the generated mimic dataset, presented in Figure 2(d), closely resembles the requirement dataset. Finally, the learnware index selector is trained using the mimic dataset, and the ensemble of models achieves a prediction accuracy of 98.70% on the requirement, as detailed in Figures 2(e) & 2(f). This example visually demonstrates that the proposed LANE specification effectively achieves dataset inaccessibility and the reusability of pre-trained models.

**Impact of the Specification Size.** Recalling the learnware paradigm, it is evident that the specification is its core element. This raises a natural question: *How does the size of the specification affect performance*? To this end, we investigate the performance of learnware paradigm with LANE and RKME under different specification sizes. Figure 3 shows that the LANE performs better as the specification size increases, while the RKME method exhibits no significant performance fluctuations across different sizes. Upon a detailed analysis of Eq.(7) and Eq.(10), we observe that an increase in the specification size leads to a richer retention of information from the developer dataset and a more precise calculation of intra-class covariance. Consequently, the diversity of the mimic dataset, generated through LANE-based sampling, is enhanced, leading to a better alignment with the requirement dataset. As a result, the overall performance of the learnware paradigm is improved.

## Conclusion

In this paper, we introduce a novel learnware specification approach named LANE, making the first attempt to incorporate label information from the training data into the specification generation process. This approach endows the generated specifications with superior dataset characterization capabilities compared to existing RKME-based specification methods that solely rely on feature information. Furthermore, the neural embedding space is utilized to replace the existing RKHS, effectively addressing the kernel dependency issue of current approaches. Comprehensive experiments verify the effectiveness and superiority of our proposed LANE approach. In the future, it will be interesting to investigate how to effectively mitigate the impact of specification size on model reuse and thereby enhance the robustness of the LANE-based specification learnware paradigm.

## References


Chen, Y. 2013. *Herding: Driving deterministic dynamics to learn and sample probabilistic models dissertation*. Ph.d. diss., Doctor of Philosophy, University of California, CA. USA.

Chen, Y.; Welling, M.; and Smola, A. 2010. Super-samples from kernel herding. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 109–116. Catalina Island, CA.

Dong, T.; Zhao, B.; and Lyu, L. 2022. Privacy for free: How does dataset condensation help privacy? In *Proceedings of the 39th International Conference on Machine Learning*, 5378–5396. Baltimore, Maryland.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. 2006. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*, 265–284. New York, NY.

Giryes, R.; Sapiro, G.; and Bronstein, A. M. 2016. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13): 3444–3457.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1): 723–773.

Guo, L.-Z.; Zhou, Z.; Li, Y.-F.; and Zhou, Z.-H. 2023. Identifying useful learnwares for heterogeneous label spaces. In *Proceedings of the 40th International Conference on Machine Learning*, 12122–12131. Honolulu, Hawaii.

Joachims, T. 1997. A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 9, 143–151. Nashville, Tennessee.

Jordan, M. I.; and Mitchell, T. M. 2015. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245): 255–260.

Kairouz, P.; Oh, S.; and Viswanath, P. 2015. The composition theorem for differential privacy. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, 1376–1385. Lille, France.

Krizhevsky, A. 2009. Poligon: A system for parallel problem solving. Technical Report Online, Available: https://www.cs.toronto.edu/ kriz/cifar.html.

Lei, H.-Y.; Tan, Z.-H.; and Zhou, Z.-H. 2024. On the ability of developers' training data preservation of learnware. In *Advances in Neural Information Processing Systems 37*. Vancouver, Canada.

Liu, J.-D.; Tan, Z.-H.; and Zhou, Z.-H. 2024. Towards making learnware specification and market evolvable. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 13909–13917. Vancouver, Canada.

Menghani, G. 2023. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, 55(12): 1–37.

Mumuni, A.; Mumuni, F.; and Gerrar, N. K. 2024. A survey of synthetic data augmentation methods in machine vision. *Machine Intelligence Research*, 21(5): 831–869.

Rawat, W.; and Wang, Z. 2017. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9): 2352–2449.

Sarker, I. H. 2021. Machine learning: Algorithms, real-world applications and research directions. *SN computer science*, 2(3): 160.

Tan, P.; , H.-T.; Tan, Z.-H.; and Zhou, Z.-H. 2024a. Handling learnwares from heterogeneous feature spaces with explicit label exploitation. In *Advances in Neural Information Processing Systems 37*. Vancouver, Canada.

Tan, P.; Tan, Z.-H.; Jiang, Y.; and Zhou, Z.-H. 2023. Handling learnwares developed from heterogeneous feature spaces without auxiliary data. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence*, 4235–4243. Macao, China.

Tan, P.; Tan, Z.-H.; Jiang, Y.; and Zhou, Z.-H. 2024b. Towards enabling learnware to handle heterogeneous feature spaces. *Machine Learning*, 113(3): 1839–1860.

Tan, Z.-H.; Liu, J.-D.; Bi, X.-D.; Tan, P.; Zheng, Q.-C.; Liu, H.-T.; Xie, Y.; Zou, X.-C.; Yu, Y.; and Zhou, Z.-H. 2024c. Beimingwu: A learnware dock system. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5773–5782. Barcelona, Spain.

Vandenberghe, L. 2010. The CVXOPT linear and quadratic cone program solvers. Technical Report Online, http://cvxopt.org/documentation/coneprog.pdf.

Williams, C. K.; and Rasmussen, C. E. 2006. *Gaussian processes for machine learning*. MIT press Cambridge, MA.

Wu, X.-Z.; Liu, S.; and Zhou, Z.-H. 2019. Heterogeneous model reuse via optimizing multiparty multiclass margin. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, 6840–6849. Long Beach, California.

Wu, X.-Z.; Xu, W.; Liu, S.; and Zhou, Z.-H. 2023. Model reuse with reduced kernel mean embedding specification. *IEEE Transactions on Knowledge and Data Engineering*, 35(1): 699–710.

Xie, Y.; Tan, Z.-H.; Jiang, Y.; and Zhou, Z.-H. 2023. Identifying helpful learnwares without examining the whole market. In *Proceedings of the 26th European Conference on Artificial Intelligence*, 2752–2759. Kraków, Poland.

Zhang, Y.-J.; Yan, Y.-H.; Zhao, P.; and Zhou, Z.-H. 2021. Towards enabling learnware to handle unseen jobs. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 10964–10972. Virtual Event.

Zhao, B.; and Bilen, H. 2021. Dataset condensation with differentiable siamese augmentation. In *Proceedings of the 38th International Conference on Machine Learning*, 12674–12685. Virtual Event.

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

Zhou, Z.-H.; and Tan, Z.-H. 2024. Learnware: Small models do big. *Science China Information Sciences*, 67(1): 112102.

Zhu, X.-X.; Vondrick, C.; Fowlkes, C. C.; and Ramanan, D. 2016. Do we need more training data? *International Journal of Computer Vision*, 119(1): 76–92.