Identifying Useful Learnwares for Heterogeneous Label Spaces

Lan-Zhe Guo [*1] **Zhi Zhou** [*1] **Yu-Feng Li** [1] **Zhi-Hua Zhou** [1]

Abstract

The *learnware* paradigm aims to build a *learnware market* containing numerous learnwares, each of which is a well-performing machine learning model with a corresponding *specification* to describe its functionality so that future users can identify useful models for reuse according to their own requirements. With the learnware paradigm, model developers can spontaneously submit models to the market without leaking data privacy, and users can leverage models in the market to accomplish different machine learning tasks without having to build models from scratch. Recent studies have attempted to realize the model specification through Reduced Kernel Mean Embedding (RKME). In this paper, we make an attempt to improve the effectiveness of RKME specification for heterogeneous label spaces, where the learnware market does not contain a model that has the same label space as the user's task, by considering a class-specific model specification explicitly, along with a class-wise learnware identification method. Both theoretical and empirical analyses show that our proposal can quickly and accurately find useful learnwares that satisfy users' requirements. Moreover, we find that for a specific task, reusing a small model identified via the specification performs better than directly reusing a pre-trained generic big model.

1. Introduction

Machine learning has achieved great success in various tasks and applications (Jordan & Mitchell, 2015). However, training a well-performing machine learning model from scratch is difficult since it requires a huge number of training data, particularly labeled data (Zhou, 2017); proficient training skills (LeCun et al., 2015); massive computing resources (Guo et al., 2020). Moreover, the amount of possible machine learning tasks can be unimaginably big or even infinite, it would be expensive or even impossible to train a well-performing model for every task. Data privacy concerns are also a serious issue when reusing or adapting a trained model among different users.

To deal with the above challenges simultaneously, learnware (Zhou, 2016; Zhou & Tan, 2022) provides a new paradigm that attempts to change the current style of standard machine learning to a style where the previous efforts of other users can be identified and reused, by constructing a public learnware market containing numerous learnwares. Each learnware is a well-performing machine learning model with a specification that conveys its specialty and utility. A standard learnware paradigm is conducted as follows: a developer or owner of a trained machine learning model (no matter whether the model structure or the training algorithm) can spontaneously submit the trained model into the learnware market. The learnware market assigns a specification to the model, which captures its specialty and utility and enables it to be identified for reuse, and then incorporates the model into the market. When a user is going to tackle a machine learning task, the user can submit the requirement to the learnware market, and then the market will identify useful learnware(s) by considering the model specification. It is noteworthy that the learnware market has no access to either the data of developers or users. Figure 1 presents a comparison of the standard machine learning paradigm and the learnware paradigms.

It is evident that the specification plays a pivotal role in the learnware paradigm. A model specification should satisfy two important properties: 1) the specification should accurately describe the characteristics of the model, such that given a new learning task, it is possible to identify useful learnwares based on the specification; 2) the specification must not leak its original training data, otherwise the model developers may not be able to share their models due to concerns about the leak of data privacy. A recent effort is the RKME (Reduced Kernel Mean Embedding) specification (Zhou & Tan, 2022), based on techniques of the reduced set of KME (Kernel Mean Embedding) (Muandet et al., 2017; Wu et al., 2021). The KME is a powerful technique to map a probability distribution to a point in RKHS (Reproducing Kernel Hilbert Space) without losing infor-

[*]Equal contribution [1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. Correspondence to: Yu-Feng Li <liyf@lamda.nju.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

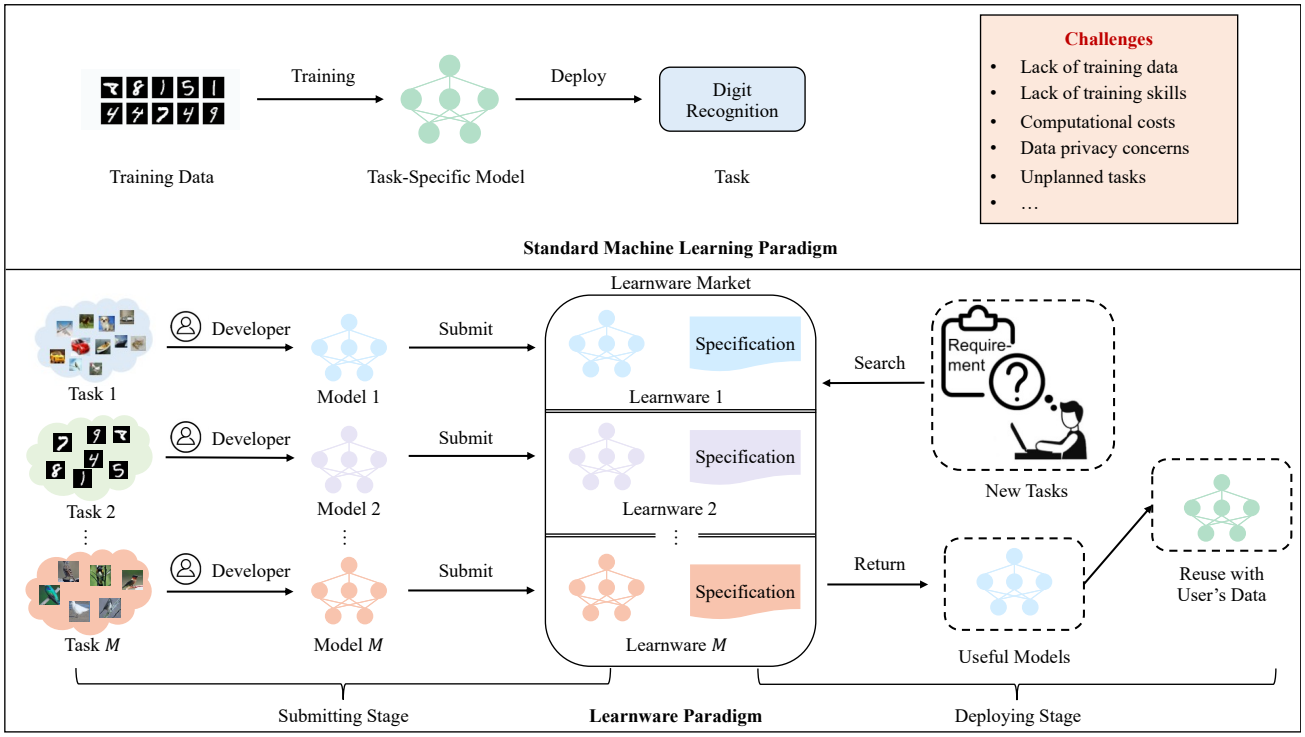

Figure 1. Comparison between the standard machine learning paradigm and the learnware paradigm.

mation (Sriperumbudur et al., 2011), and the reduced set reserves the ability with a concise representation that does not expose the original data privacy.

The RKME specification is based on the assumption that identifying useful models for a user's task can be approached by identifying models whose training data distribution is close to the user task's data distribution. Although identifying useful learnwares approached by whole data distribution similarity has achieved impressive results on various tasks (Wu et al., 2021; Zhou & Tan, 2022; Tan et al., 2022), the conditional probability distribution for each class has not been modeled explicitly, thus, the correspondence between the label space of the models in the market and the user's task can not be taken into account. This limits its effectiveness in the heterogeneous label space settings where the learnware market does not contain a model with the same label space as the user's task.

To this end, we propose a new plugin to improve the effectiveness of RKME specification for heterogeneous label spaces. Specifically, in the submitting stage, we propose to assign a class-wise specification for each submitted model by reducing the model into a linear proxy model via twice learning (Zhou & Jiang, 2004), a similar idea later called knowledge distillation (Hinton et al., 2014), to approximate the model's functionality on each class, in addition to reducing the whole training data distribution into a point in RKHS. In the deploying stage, we propose a fine-grained matching method to identify the most useful learnware in a class-wise manner, which contains two steps. In the first step, we identify multiple candidate learnwares by considering the model specifications. In the second step, we construct a bipartite graph for each candidate learnware and the user's task and then obtain the maximum matched learnware through the Hungarian algorithm (Kuhn, 1955). Theoretical analysis shows that specification similarity can accurately approximate the ground-truth model reuse performance on the user's task. Experimental results on more than 20 tasks show that our proposal not only identifies the most useful model for the user's task but also the similarity ranking of the new proposal is closely related to the ranking of ground-truth model reuse performance. Moreover, we show that for a specific task, fine-tuning a small model identified via the model specification can do better than directly fine-tuning a pre-trained generic big model.

2. Related Works

This paper is mainly related to learnware, model reuse, model selection via LLM, and reusability evaluation studies.

Learnware. Learnware (Zhou, 2016; Zhou & Tan, 2022) presents a general and realistic paradigm where numerous

models for various tasks associated with their specifications are available in the learnware market and users can search for a useful model from the market based on their requirements, reducing the number of resources required. The model specification is an essential component of the learnware paradigm, describing the functionality of each model. Recently, there have been some efforts in this direction. For example, the RKME specification (Zhou & Tan, 2022) constructs the specification space by mapping the training data of models to an element of the RKHS, helpful models can be identified by calculating the RKHS distance between RKMEs (Wu et al., 2021). When the user's task involves unseen tasks not covered by the learnware market, Zhang et al. (2021) extended the RKME specification by using the mixture proportion estimation (MPE) technique (Ramaswamy et al., 2016; Zhang et al., 2020) to identify examples from these unseen tasks while assigning the rest to proper models returned from the market. Tan et al. (2022) provides a solution for models from heterogeneous feature spaces by generating the RKME specification in a unified subspace. These studies describe the model's specialty and utility by approximating the whole training data distribution while ignoring class-wise conditional probability distributions. This paper provides a solution to improve the effectiveness of learnware paradigm for heterogeneous label spaces.

Model Reuse. Model reuse tries to adapt models from related source tasks to a new target task (Pan & Yang, 2010). Transfer learning or domain adaptation is one way of implementing model reuse (Duan et al., 2009; Du et al., 2017; Kuzborskij & Orabona, 2013). But they usually assume access to both source and target domain data simultaneously, which may not be feasible in real-world scenarios due to privacy and confidentiality concerns. There are also source-free domain adaptation methods (Chidlovskii et al., 2016; Liang et al., 2020; Shao et al., 2021; 2022), which only utilize the source-trained model and target data to adapt to the target domain. However, they require that all the source models are helpful for the target task while ignoring the challenging problem of how to identify useful models for the target task. In the learnware paradigm, there may be thousands or millions of models in the market, and only a tiny portion of models are helpful for the target task, so identifying useful models is indispensable.

Model Selection via LLMs. Recently, with the success of large language models (LLMs) (Brown et al., 2020; Touvron et al., 2023), there are some works proposed to adopt the LLM to identify useful models for a user's task based on the natural language descriptions of the model's characteristics and the user's requirements. For example, HuggingGPT (Shen et al., 2023) proposed to use ChatGPT as a machine learning model selector to identify useful models that meet users' requirements from the HuggingFace platform. In contrast to these efforts that only provide a hub or pool of pre-trained models, where the models are to be used "as-what-was-submitted", the learnware paradigm demonstrates a fundamentally different manner, where the learnware market is to enable its accommodated models to be used "beyond-what-was-submitted". In other words, the learnware market aims to enable its accommodated models to be useful in tasks that were not considered by the models' developers, where the learnware specification plays a fundamental role. Needless to say, the specification can help the learnware market accommodate the models in a highly organized way, enabling efficient identification and assembling helpful models for new tasks.

Reusability Evaluation. There are also studies focused on how to evaluate the reusability of trained models for a target task. For example, Tran et al. (2019) developed the negative conditional entropy measure between the source and target label sets to study the transferability between classification tasks. Bao et al. (2019) developed a transferability measure based on H-scores, which are derived from information-theoretic principles. Nguyen et al. (2020) developed a LEEP measure, which is the log expectation of the empirical predictor constructed by estimating the joint distribution over pre-trained labels and the target labels. You et al. (2021) proposed to estimate the maximum value of label evidence given features extracted by trained models and obtained the Logarithm of Maximum Evidence (LogME) measure. However, these measures necessitate running each candidate model on the target data, which is impractical for real-world scenarios with numerous models. On the contrary, with the design of the model specification to describe the specialty and utility of each model, we can identify useful models simply by comparing user requirements with model specifications, without running models.

3. Our Approach

In this section, we present our approach, including the problem setup, details of the specification assignment and learnware identification, and theoretical analysis.

3.1. Problem Setup

In the learnware paradigm, we assume there are M available models $\{f_m\}_{m=1}^{M}$ in the learnware market where f_m is trained on the dataset $\mathcal{D}_m = \{\mathbf{X}_m, \mathbf{Y}_m\}$. $\mathbf{X}_m \in \mathbb{R}^{N_m \times d_m}$ indicates the feature matrix, $\mathbf{Y}_m \in \mathbb{R}^{N_m \times K_m}$ indicates the label matrix. N_m, d_m, and K_m represent the number of training examples, the dimension of feature space, and the dimension of label space, respectively. When the model developer submits the model to the learnware market, the market needs to assign a specification S_m to each model. When a user wants to deploy a model to tackle his/her own task with dataset $\mathcal{D}_\mathcal{T} = \{\mathbf{X}_\mathcal{T}, \mathbf{Y}_\mathcal{T}\}$ where $\mathbf{X}_\mathcal{T} \in \mathbb{R}^{N_\mathcal{T} \times d_\mathcal{T}}$ and $\mathbf{Y}_\mathcal{T} \in \mathbb{R}^{N_\mathcal{T} \times K_\mathcal{T}}$, the learnware market generates $S_\mathcal{T}$

Algorithm 1 Specification Assignment.

Input: Dataset $\{\mathbf{X}, \mathbf{Y}\}$, model f, feature extractor $G(\cdot)$.
Output: Model specification S.

1: Project the feature into a new space with $G(\cdot)$.
2: Obtain the data reduction $\Phi = (\boldsymbol{\beta}, \mathbf{U})$ by optimizing Eq.(1).
3: Obtain the model reduction $\mathbf{W}$ by optimizing Eq.(2).
4: **Return** $S = (\Phi, \mathbf{W})$.

Algorithm 2 Identify Useful Learnwares.

Input: User's dataset $\{\mathbf{X}_\mathcal{T}, \mathbf{Y}_\mathcal{T}\}$, M trained models $\{f_m\}_{m=1}^M$, specifications $\{S_m\}_{m=1}^M$.
Output: Identified model f.

1: Obtain $\Phi_\mathcal{T}$ and $\mathbf{W}_\mathcal{T}$ for the user's data.
2: **if** homogeneous setting **then**
3: Select C candidate models that have the most similar Φ and the same label space as the user's task.
4: Calculating the similarity between $\mathbf{W}_c$ and $\mathbf{W}_\mathcal{T}$ via Eq.(5).
5: Return f with the most similar $\mathbf{W}$ to $\mathbf{W}_\mathcal{T}$.
6: **end if**
7: **if** heterogeneous setting **then**
8: Select C candidate models that have the most similar Φ as the user's task.
9: For each class k in the user's task, select a model with the most similar class-wise specification to $\mathbf{W}_{\mathcal{T},k}$, and obtain L candidate models.
10: Construct a bipartite graph for the user's task and each candidate model f_l.
11: Run the Hungarian algorithm to find the maximum matching for each model f_l.
12: Return f with the maximum matching score.
13: **end if**

as the user's requirements using $\mathcal{D}_\mathcal{T}$ and returns the model with the most similar specification to $S_\mathcal{T}$ in the learnware market. It is noteworthy that, the learnware paradigm protects *data privacy*, i.e., the learnware market only stores the model and related specification, the original training data is not accessible.

To realize the learnware paradigm, there are two important challenges:

- **In the submitting stage, how to assign specifications to describe the utility of the submitted model?**
- **In the deploying stage, how to identify useful learnwares in the market given a user's requirements?**

3.2. Assign Specifications in the Submitting Stage

In the submitting stage, the model developer submits a model to the learnware market, and the market needs to assign a proper specification to the model. The model specification should convey the model's specialty and utility while also protecting the training data privacy. Considering the fact that there will exist heterogeneous models with different feature spaces, i.e., d_m could be varied in different models. To obtain a unified feature space, we assume there is a public feature extractor $G(\cdot)$ that can map the training data of each model into a new unified representation space. The assumption is practical for real-world tasks since we can adopt a public pre-trained representation learning model as the feature extractor.

When a model f_m is submitted to the learnware market, we first generate a new feature matrix for its original training data using $G(\cdot)$ and obtain $\mathbf{Z}_m = G(\mathbf{X}_m) \in \mathbb{R}^{N_m \times d}$.

Then, we obtain a reduced training dataset by optimizing the following objective:

$$\min_{\boldsymbol{\beta_m}, \mathbf{U}_m} \left\| \frac{1}{N_m} \sum_{i=1}^{N_m} k(\mathbf{Z}_{m,i}, \cdot) - \sum_{j=1}^{V_m} \beta_{m,j} k(\mathbf{U}_{m,j}, \cdot) \right\|_\mathcal{H}^2 \quad (1)$$

where $k(\cdot, \cdot)$ is the kernel function with associated RKHS $\mathcal{H}$, $\mathbf{U}_{m,j}$ is the element in the reduced set, and $\beta_{m,j}$ is the corresponding coefficient.

The above minimization problem can be solved with the alternating optimization algorithm, as presented in Wu et al. (2021). After solving the problem, we can obtain $\Phi_m = \{(\boldsymbol{\beta}_m, \mathbf{U}_m)\}$ where $\mathbf{U}_m \in \mathbb{R}^{V_m \times d}$.

Next, we reduce the submitted model f_m into a linear proxy model by minimizing the following objective:

$$\min_{\mathbf{W}_m} \mathcal{L}_{CE}(\mathbf{Z}_m \mathbf{W}_m, \mathbf{Y}_m) + \mathcal{L}_{KL}(\mathbf{Z}_m \mathbf{W}_m, f_m(\mathbf{X}_m)) \quad (2)$$

where $\mathcal{L}_{CE}$ and $\mathcal{L}_{KL}$ indicates the cross-entropy loss and KL-divergence, respectively, and $\mathbf{W}_m \in \mathbb{R}^{d \times K_m}$ is the parameter of the reduced model.

It is noteworthy that the k-th column in the matrix $\mathbf{W}_m$ can be seen as a specification for class k, thus, $\mathbf{W}_m$ is a class-wise specification that describes the model's functionality on each class.

We treat Φ_m and normalized $\mathbf{W}_m$ as the specification of model f_m, i.e., $S_m = (\Phi_m, \mathbf{W}_m)$, and the corresponding learnware is (f_m, S_m).

Remark. It is worth noting that the model specification S_m is unrelated to the feature dimension, implying that our proposal can naturally deal with heterogeneous feature space. Moreover, the learnware market only stores the model and corresponding specification, thus, the model developer does not leak original training data.

3.3. Identify Useful Learnwares in the Deploying Stage

In the deploying stage, the user searches for useful models in the learnware market according to the model specification. In our study, we consider both homogeneous and heterogeneous label spaces to make the proposal more practical. In the homogeneous label space setting, the learnware market contains models that have the same label space as the user's task, while in the heterogeneous setting, there is not a model in the learnware market that can fit the user's task perfectly.

Similar to the specification assignment process in the submitting stage, given a small dataset $\mathcal{D}_\mathcal{T} = \{\mathbf{X}_\mathcal{T}, \mathbf{Y}_\mathcal{T}\}$ of the user's task, we first map the feature into a new representation space with $G(\cdot)$ and obtain $\mathbf{Z}_\mathcal{T} = G(\mathbf{X}_\mathcal{T}) \in \mathbb{R}^{N_\mathcal{T} \times d}$.

The data reduction of the user's task can be obtained as

$$\Phi_\mathcal{T} = \frac{1}{N_\mathcal{T}} \sum_{i=1}^{N_\mathcal{T}} k(\mathbf{Z}_{\mathcal{T},i}, \cdot) \tag{3}$$

and the model reduction can be obtained by optimizing the following objective

$$\min_{\mathbf{W}_\mathcal{T}} \mathcal{L}_{CE}(\mathbf{Z}_\mathcal{T}\mathbf{W}_\mathcal{T}, \mathbf{Y}_\mathcal{T}) \tag{4}$$

For the homogeneous label space setting, we first select all models that have the same label space as the user's task and then select C candidate models $\{f_c\}_{c=1}^C$ by comparing the RKHS distance between $\Phi_\mathcal{T}$ with Φ_m in the learnware market, which is the same as Wu et al. (2021).

Next, we compute the similarity between $\mathbf{W}_\mathcal{T}$ and $\mathbf{W}_c$

$$\text{Similarity}(\mathbf{W}_\mathcal{T}, \mathbf{W}_c) = \text{Tr}(\mathbf{W}_\mathcal{T}^\top \mathbf{W}_c) \tag{5}$$

where $\text{Tr}(\cdot)$ is the trace of a matrix.

Finally, we can return a model that has the most similar specification to the user's task.

For the heterogeneous setting, since there is no model with the same label space as the user's task, we first select C candidate learnwares according to $\Phi_\mathcal{T}$, and then identify the most useful learnware in a class-wise manner according to $\mathbf{W}_\mathcal{T}$. This class-wise identification process contains two steps. In the first step, we select potentially useful learnwares for each class. Note that the specification $\mathbf{W}_\mathcal{T}$ can be written as $[\mathbf{W}_{\mathcal{T},1}; \cdots; \mathbf{W}_{\mathcal{T},k}; \cdots; \mathbf{W}_{\mathcal{T},K_\mathcal{T}}]$ where $\mathbf{W}_{\mathcal{T},k} \in \mathbb{R}^{d \times 1}$. We denote $\mathbf{W}_{\mathcal{T},k}$ as the class-wise specification for class k in the user's task. For each model f_m in the learnware market, we also have $\mathbf{W}_{m,i}$ as the class-wise specification for the i-th class of model f_m.

Then, for each class k in the label space of the user's task, we can select a model by comparing the similarity between $\mathbf{W}_{\mathcal{T},k}$ and $\mathbf{W}_{m,i}$, the similarity score can be computed as

$$\text{Similarity}(\mathbf{W}_{\mathcal{T},k}, \mathbf{W}_{m,i}) = \mathbf{W}_{\mathcal{T},k}^\top \mathbf{W}_{m,i} \tag{6}$$

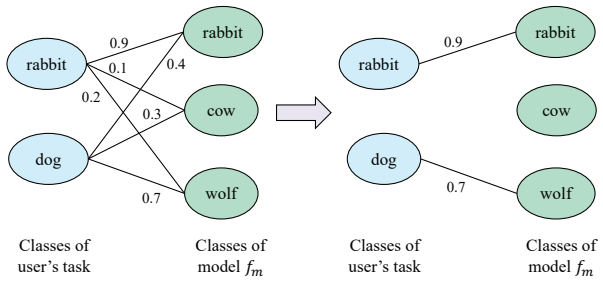

Figure 2. An example of the matching process in the heterogeneous label space setting. Assume that the label space of the user's task and the candidate model f_m are {"rabbit", "dog"} and {"rabbit", "cow", "wolf"}, respectively. We construct a bipartite graph with the class-wise similarity score as the edge weights, as shown on the left. Then, we find the maximum matching via the Hungarian algorithm and can obtain that the matching score between the user's task and f_m is 0.9+0.7=1.6.

After the first step, we can obtain L candidate models $\{f_1, \cdots, f_L\}$ and $L \leq K_\mathcal{T}$ because the same model can be selected for different classes.

In the second step, we construct a bipartite graph for the user's task and each candidate model f_l. The graph contains two node sets: the nodes of target classes and the nodes of classes in model f_l. The weights of edges in the graph are the similarity score: $\text{Similarity}(\mathbf{W}_{\mathcal{T},k}, \mathbf{W}_{l,i})$. Then we run the Hungarian algorithm (Kuhn, 1955) to find the maximum matching of each bipartite graph. We give an example of the matching process in Figure 2. Finally, we can return a model with the maximum matching score.

3.4. How the Proposal Works

Generally, we expect the selected model to achieve a good performance on the user's task. Wu et al. (2021) has provided theoretical results to rigorously justify the reusability of models by using the data reduction Φ. In this section, we show that the specification similarity between $\mathbf{W}$ also acts as a good approximation of the ground-truth model reuse performance.

Given a model f_m, we denote the prediction of model f_m on the target dataset $\mathcal{D}_\mathcal{T}$ as

$$\hat{\mathbf{Y}}_m = f_m(\mathbf{X}_\mathcal{T}) \tag{7}$$

The classification accuracy of f_m can be calculated as

$$\text{Accuracy}(f_m, \mathcal{D}_\mathcal{T}) = \text{Tr}(\hat{\mathbf{Y}}_m^\top \mathbf{Y}_\mathcal{T}) \tag{8}$$

By using the feature extractor $G(\cdot)$ and the reduced model, we have

$$\hat{\mathbf{Y}}_m \approx G(\mathbf{X}_\mathcal{T})\mathbf{W}_m, \mathbf{Y}_\mathcal{T} \approx G(\mathbf{X}_\mathcal{T})\mathbf{W}_\mathcal{T} \tag{9}$$

Then, the classification accuracy can be approximated as

$$\begin{aligned}&\text{Accuracy}(f_m, \mathcal{D}_\mathcal{T}) \\ &\approx \text{Tr}\Big(\big(G(\mathbf{X}_\mathcal{T})\mathbf{W}_m\big)^\top \big(G(\mathbf{X}_\mathcal{T})\mathbf{W}_\mathcal{T}\big)\Big) \\ &= \text{Tr}\big(\mathbf{W}_m^\top G(\mathbf{X}_\mathcal{T})^\top G(\mathbf{X}_\mathcal{T})\mathbf{W}_\mathcal{T}\big)\end{aligned} \tag{10}$$

Based on the analysis, we have the following proposition,

Proposition 3.1. *Assume that the representation matrix $G(\mathbf{X}_\mathcal{T})$ satisfies that $G(\mathbf{X}_\mathcal{T})^\top G(\mathbf{X}_\mathcal{T}) = I$, the accuracy of model f_m on the target dataset $\mathcal{D}_\mathcal{T}$ can be well approximated by the similarity between $\mathbf{W}_m$ and $\mathbf{W}_\mathcal{T}$, i.e.,*

$$Accuracy(f_m, \mathcal{D}_\mathcal{T}) = Tr(\mathbf{W}_m^\top \mathbf{W}_\mathcal{T}) \tag{11}$$

Remark. Proposition 3.1 reveals that the model specification similarity can accurately approximate the performance of reusing this model on the user's task. The results help us understand why the proposal works. The assumption is also reasonable since a good representation should overcome the feature redundancy problem (Wang et al., 2020).

4. Experiments

To demonstrate the effectiveness of the learnware paradigm and the new proposal, we conduct experiments to verify the following questions: 1) Can the most useful learnware be identified via the specification? 2) How about the correlation between the model reuse performance and specification similarity? 3) Is it better to reuse the identified model than to directly reuse a generic big model? 4) How much time can be saved compared to brute-force fine-tuning?

4.1. Experimental Setup

Datasets. We adopt the NICO (He et al., 2021) and DomainNet (Peng et al., 2019) datasets to evaluate the proposed learnware paradigm. The two datasets are both designed to help evaluate the effectiveness of model reuse algorithms. Specifically, the NICO dataset has two super-classes: animal and vehicle, with 10 classes for animals and 9 classes for vehicles. The dataset contains images with both main concepts (e.g., dog) and contexts (e.g., on grass). By selecting images with different contexts, we have 6 different domains: ["autumn", "dim", "grass", "outdoor", "rock", "water"]. The DomainNet (Peng et al., 2019) dataset is an image dataset with 345 categories of common objects. Specifically, we select 5 domains from the DomainNet dataset: ["clipart", "infograph", "painting", "quickdraw", "real"].

Learnware Market Construction. In practice, we expect model developers to spontaneously submit models to the learnware market. In our experiments, we manually divide different tasks using data from different domains and different label spaces, train models on these tasks and then construct the learnware market. Specifically, for the homogeneous label space setting, we select images from the common label spaces of the NICO and DomainNet datasets and obtain two label spaces. The *label space A* is ["flower", "horse", "cow", "rabbit", "tiger", "bird", "bus", "sailboat", "train", "helicopter"] and the *label space B* is ["butterfly", "owl", "bird", "giraffe", "frog", "squirrel", "train", "tent", "truck", "umbrella"]. Thus, we obtain 11 domains $\times$ 2 label spaces = 22 classification tasks. We treat each task as the user's task separately, leaving models trained on the other 21 tasks as source models in the learnware market. Overall, we have 22 user tasks, and for each user's task, the learnware market contains 21 trained models. For the heterogeneous label space setting, we construct another two label spaces in addition to label space A and label space B. Specifically, *label space C* is ["flower", "horse", "cow", "rabbit", "tiger"] and *label space D* is ["bird", "giraffe", "frog", "squirrel", "tiger"]. We treat images from the 11 domains and label spaces C/D as the user's tasks, leaving images from the 11 domains with label spaces A/B as the source tasks. Overall, we have 11 user tasks, and for each task, the learnware market contains 11 trained models. In the heterogeneous setting, we cannot find a model that has the same label space as the user's task in the learnware market.

Table 1. Pre@k measure in the learnware identification process. The best method is highlighted in bold.

Settings	Methods	Pre@1	Pre@2	Pre@3
Homo-direct use	RKME-basic	54.54	81.82	95.45
	Ours	**95.15**	**100.0**	**100.0**
Homo-fine-tuning	RKME-basic	40.90	68.18	81.82
	Ours	**81.82**	**90.91**	**90.91**
Hetero-fine-tuning	RKME-basic	36.36	45.45	50.00
	Ours	**59.09**	**63.63**	**68.18**

Training Details. For each source task, we train a ResNet-18 (He et al., 2016) as the trained model. We train the model for 500 epochs using the SGD optimizer. The initial learning rate is 0.1, and we adopt the cosine annealing learning rate decay strategy. The feature extractor $G(\cdot)$ in our experiments is a DenseNet201 (Huang et al., 2017) pre-trained on the ImageNet dataset. To generate the model specification, we train a linear model for 200 epochs using the SGD optimizer, and the learning rate is 0.1. For the user's tasks, we assume there are only 10 labeled examples available for each class. We adopt the small labeled dataset to help generate specifications and fine-tune the selected trained model, leaving the others as the test data.

Compared Methods. The basic implementation of the RKME specification(Wu et al., 2021) is adopted as the comparison method in our studies. Specifically, in the submitting stage, the RKME-basic specification constructs a reduced set of empirical KME as the specification, which maps

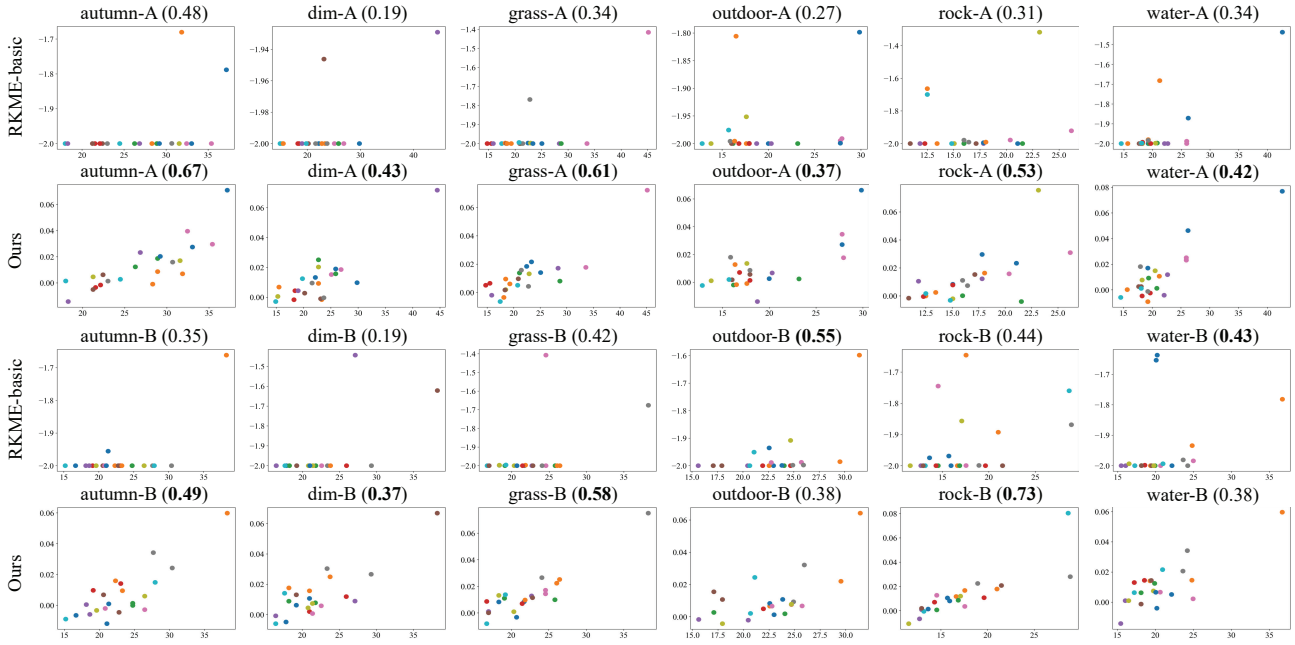

Figure 3. Correlation τ between fine-tuned performance (X-axis) and specification similarities (Y-axis) for trained model selection. The target dataset name is combined with the domains and the label spaces (e.g., autumn-A indicates examples are sampled from the domain 'autumn' and the label space is 'A'). One circle marker indicates a trained model. τ is in the bracket next to the dataset name and the best τ in each dataset is highlighted in bold.

a probability distribution to a point in RKHS and can be regarded as a representation of the distribution. In the deploying stage, the RKME-basic specification measures the RKHS distance between the mean embedding of the user's task and the reduced embedding in the learnware market. We adopt the official code in Wu et al. (2021) to implement the RKME-basic specification.

4.2. Can the most useful learnware be identified via the model specification?

The goal of the learnware paradigm is to help identify the most useful learnware from the learnware market given a user's requirement. Thus, we first pay attention to the problem that can the most useful learnware be identified based on the model specification. To verify this property, we adopt the $Pre@k$ as the evaluation measure. Specifically, given a user's task, we rank all models in the learnware market according to their specifications' similarity to the user's task. The ranking can be represented by the vector π^f, where $\pi^{f_i} < \pi^{f_j}$ if $\text{Similarity}(S_i, S_\mathcal{T}) > \text{Similarity}(S_j, S_\mathcal{T})$. Given T users' tasks $\{\mathcal{T}_1, \cdots, \mathcal{T}_T\}$ and the rank of the best model for the user's task $\mathcal{T}_t$ is $\pi_t^{f_{best}}$, the $Pre@k$ measure can be defined as:

$$Pre@k = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(\pi_t^{f_{best}} \leq k) \tag{12}$$

We report the $Pre@k$ measure on both homogeneous and heterogeneous label space settings in Table 1. Specifically, we adopt two model reuse methods for the homogeneous setting: direct use of the trained model on the user's task and fine-tuning the trained model on the user's dataset. The results show that: in the homogeneous setting, the best-performing models for direct use are always in the top 2; the model with the best fine-tuning performance is also more than 90% likely to be identified in the top-2 selected models; even in the more difficult heterogeneous setting, the best model still has a high probability of being selected. Additionally, our proposal outperforms the basic RKME specification consistently. These results demonstrate that the model specification can help identify the most useful learnwares, which is in line with our starting point.

4.3. How about the correlation between model reuse performance and specification similarity?

Except for the most useful learnware, we are also interested in the correlation between the ground-truth model reuse performance and the model specification similarity rank. Inspired by (You et al., 2021), we adopt Kendall's τ coefficient (Kendall, 1938) as the performance measure. Assume we have M trained models $\{f_1, \cdots, f_M\}$, the model reuse performance of the trained model f_i on the user's task is P_i and the similarity between the model's specification S_i

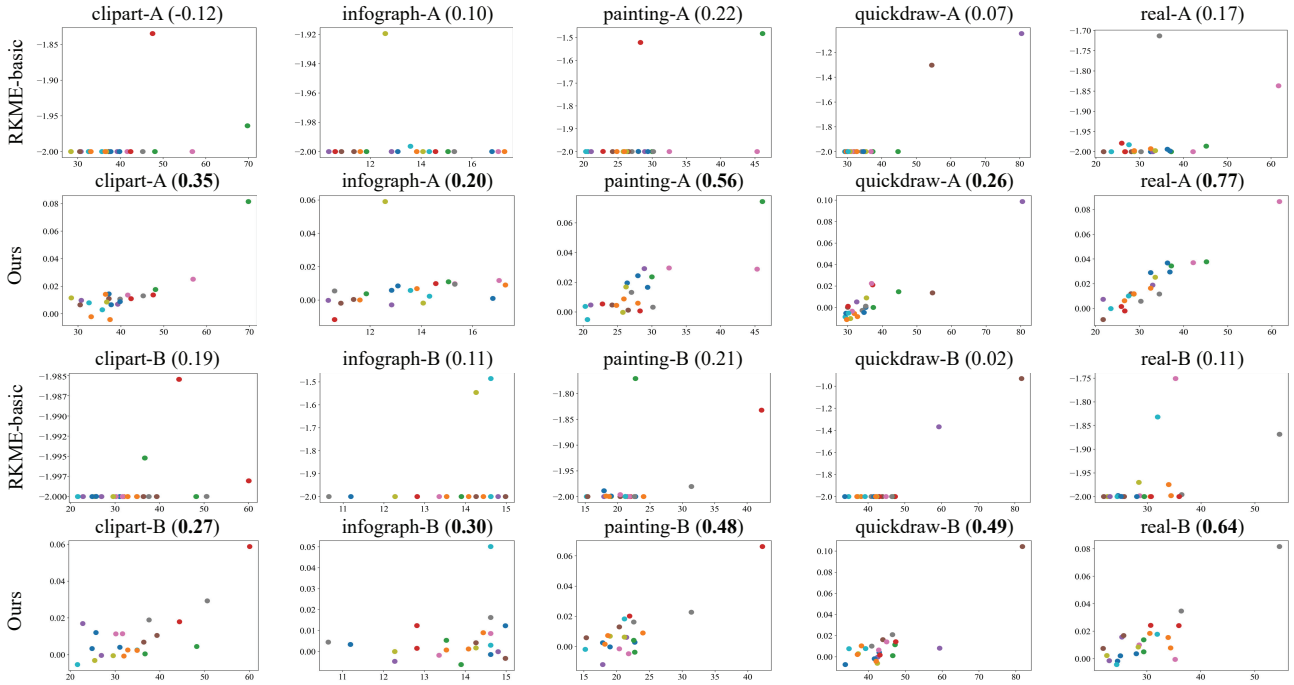

Figure 4. Correlation τ between fine-tuned performance (X-axis) and specification similarities (Y-axis) for trained models. τ is in the bracket next to the dataset name and the best τ is highlighted in bold.

and the user's task $S_{\mathcal{T}}$ is Q_i, the Kendall's τ coefficient is defined as:

$$\tau = \frac{2}{M(M-1)} \sum_{1 \le i \le j \le M} sign(P_i - P_j) sign(Q_i - Q_j) \tag{13}$$

This measure reflects the rank order consistency between the ground-truth model reuse performance and the model specification similarities.

We use the basic RKME and our proposal to compute similarities between S_i and $S_{\mathcal{T}}$ for the trained model f_i and the user's task, separately. Figure 3 and Figure 4 show the correlation τ between specification similarity and model reuse performance on 22 user tasks. We can find that our proposal outperforms the basic RKME specification on most datasets (20 datasets out of 22 datasets).

4.4. Is it better to reuse the identified model than to directly reuse a generic big model?

To generate the model specification, we adopt a generic big model DenseNet201 (Huang et al., 2017) pre-trained on the ImageNet dataset as the feature extractor. One may question about that why don't we directly fine-tune the big model on the user's task. We perform experiments on 22 user tasks to compare the performance of fine-tuning the generic big model, the model selected by the basic RKME specification, and the model selected by the new proposal. For each user's task, the models in the learnware market are ResNet18 trained on the other 21 tasks. The average classification accuracy of fine-tuning different models on 22 users' tasks is reported in Table 2. The results show that fine-tuning the model selected by both the basic RKME and the new proposal outperforms the generic big model. This demonstrates that though the big model paradigm has achieved great success, it is difficult to expect one big model to achieve consistently good performance on a wide range of tasks, whereas there are abundant task-specific small models with the development of machine learning, it is crucial to collect numerous task-specific small models and reuse them to solve different tasks.

Table 2. The average classification accuracy of fine-tuning different models on 22 users' tasks.

Methods	Big model	RKME-basic	Ours
Accuracy	32.99	33.94	**43.54**
Performance Gains		2.88%	**31.98%**

4.5. How much time can be saved compared to brute-force fine-tuning?

Given M trained models, the straightforward way to select the useful model is to fine-tune every model on the user's

task and select the best one. In our experiments, we estimate the training time for fine-tuning a ResNet-18 on a single NVIDIA 3090 GPU card to be close to 48s. In the learnware paradigm, we simply compute the similarity between model specifications and then choose the one that is most similar to the user's task. The average computation time cost to compute the similarity between Φ and $\mathbf{W}$ is nearly 0.003s and 8×10^{-6}s, respectively. Therefore, the learnware paradigm can achieve at least a thousand-fold speedup compared with brute-force fine-tuning, and the more models available, the more time can be saved.

5. Conclusions

In this paper, we study the learnware paradigm, which attempts to enable users to build machine learning models without needing to start from scratch. The key ingredient is the specification, which enables a trained model to be adequately identified for reuse according to the requirements of future users who know nothing about the model in advance. This paper proposes a new plugin to improve the effectiveness of the learnware paradigm for heterogeneous label space settings by considering data and model reduction simultaneously to obtain a class-specific specification, along with a class-wise learnware identification method. Both experimental results and theoretical analysis demonstrate the effectiveness of our proposal.

One limitation is that the identification process needs to examine the whole market. In the future, we plan to study how to identify useful learnwares more efficiently via *anchor learnware* (Zhou & Tan, 2022). Moreover, it is also interesting to study how to incorporate the LLMs to facilitate the learnware identification process.

Acknowledgements

This research was supported by the National Science Foundation of China (62250069). We thank Zhi-Hao Tan for helpful discussions.

References

Bao, Y., Li, Y., Huang, S.-L., Zhang, L., Zheng, L., Zamir, A., and Guibas, L. An information-theoretic approach to transferability in task transfer learning. In *2019 IEEE International Conference on Image Processing*, pp. 2309–2313, 2019.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pp. 1877–1901, 2020.

Chidlovskii, B., Clinchant, S., and Csurka, G. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 451–460, 2016.

Du, S. S., Koushik, J., Singh, A., and Póczos, B. Hypothesis transfer learning via transformation functions. In *Advances in Neural Information Processing Systems*, pp. 574–584, 2017.

Duan, L., Tsang, I. W., Xu, D., and Chua, T. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 289–296, 2009.

Guo, L.-Z., Zhang, Z.-Y., Jiang, Y., Li, Y.-F., and Zhou, Z.-H. Safe deep semi-supervised learning for unseen-class unlabeled data. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 3897–3906, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

He, Y., Shen, Z., and Cui, P. Towards Non-I.I.D image classification: A dataset and baselines. *Pattern Recognition*, 110:107383, 2021.

Hinton, G. E., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. In *NIPS Workshop on Deep Learning and Representation Learning*, 2014.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017.

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

Kendall, M. G. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2): 83–97, 1955.

Kuzborskij, I. and Orabona, F. Stability and hypothesis transfer learning. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 942–950, 2013.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521(7553):436–444, 2015.

Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 6028–6039, 2020.

Muandet, K., Fukumizu, K., Sriperumbudur, B., Schölkopf, B., et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10(1-2):1–141, 2017.

Nguyen, C., Hassner, T., Seeger, M. W., and Archambeau, C. LEEP: A new measure to evaluate transferability of learned representations. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 7294–7305, 2020.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (10):1345–1359, 2010.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., and Wang, B. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

Ramaswamy, H. G., Scott, C., and Tewari, A. Mixture proportion estimation via kernel embeddings of distributions. In *Proceedings of the 33nd International Conference on Machine Learning*, pp. 2052–2060, 2016.

Shao, J.-J., Cheng, Z., Li, Y.-F., and Pu, S. Towards robust model reuse in the presence of latent domains. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pp. 2957–2963, 2021.

Shao, J.-J., Guo, L.-Z., Yang, X.-W., and Li, Y.-F. LOG: Active model adaptation for label-efficient ood generalization. In *Advances in Neural Information Processing Systems*, pp. 11023–11034, 2022.

Shen, Y., Song, K., Tan, X., Li, D., Lu, W., and Zhuang, Y. HuggingGPT: Solving ai tasks with chatgpt and its friends in hugging face. *CoRR*, abs/2303.17580, 2023.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. G. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12: 2389–2410, 2011.

Tan, P., Tan, Z.-H., Jiang, Y., and Zhou, Z.-H. Towards enabling learnware to handle heterogeneous feature spaces. *Machine Learning*, pp. 1–22, 2022.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

Tran, A. T., Nguyen, C. V., and Hassner, T. Transferability and hardness of supervised classification tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1395–1405, 2019.

Wang, J., Chen, Y., Chakraborty, R., and Yu, S. X. Orthogonal convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11505–11515, 2020.

Wu, X.-Z., Xu, W., Liu, S., and Zhou, Z.-H. Model reuse with reduced kernel mean embedding specification. *IEEE Transactions on Knowledge and Data Engineering*, 35 (1):699–710, 2021.

You, K., Liu, Y., Wang, J., and Long, M. LogME: Practical assessment of pre-trained models for transfer learning. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 12133–12143, 2021.

Zhang, Y.-J., Zhao, P., Ma, L., and Zhou, Z.-H. An unbiased risk estimator for learning with augmented classes. In *Advances in Neural Information Processing Systems*, pp. 10247–10258, 2020.

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

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

Zhou, Z.-H. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53, 2017.

Zhou, Z.-H. and Jiang, Y. NeC4.5: Neural ensemble based C4.5. *IEEE Transactions on Knowledge and Data Engineering*, 16(6):770–773, 2004.

Zhou, Z.-H. and Tan, Z.-H. Learnware: Small models do big. *CoRR*, abs/2210.03647, 2022.