Front. Comput. Sci., 2025, 19(9): 199344
https://doi.org/10.1007/s11704-024-40135-0

RESEARCH ARTICLE

Identifying useful learnwares via learnable specification

Zhi-Yu SHEN[1], **Ming LI** (✉)[1,2]

1 National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
2 School of Artificial Intelligence, Nanjing University, Nanjing 210023, China

Abstract The *learnware paradigm* has been proposed as a new manner for reusing models from a market of various well-trained models, which can relieve users' burden of training a new model from scratch. A learnware consists of a well-trained model and a specification which explains the purpose or specialty of the model without revealing data. By specification matching, the market can identify the most useful learnwares for users' tasks. Prior art attempted to generate the specification by a reduced kernel mean embedding approach. However, such kind of specification is defined by some pre-designed kernel function, which lacks flexibility. In this paper, we advance a methodology for direct specification learning from data, introducing a novel neural network named *SpecNet* for this purpose. Our approach accepts unordered datasets as input and subsequently produces specification vectors in a latent space. Notably, the flexibility and efficiency of our learned specifications are underscored by their derivation from diverse tasks, rendering them particularly adept for learnware identification. Empirical studies provide validation for the efficacy of our proposed approach.

Keywords learnware, specification, model reuse

1 Introduction

Nowadays, machine learning has been widely utilized in various fields. For example, it is applied in computer vision, AI conversation, speech recognition and so on. Thus, there is a high demand for collecting data and training models in various industries. However, collecting data and training models require a lot of time and computational resources, which avoid the users building their own models. The technical barrier of machine learning also prevents many users from accomplishing their own tasks. To overcome such challenges, users would like to seek for some convenient and inexpensive pipeline to get usable models. Now that there already exists many well-trained models in numerous fields in the real world, reusing them is an excellent idea. Some works have claimed that it is very meaningful to build a model sharing market to help users benefit from existing models [1−4]. Additionally other works have proposed to adapt the existing models or data to new tasks. However, the adaptation process is not a straightforward job, requiring massive algorithmic knowledge and computational resource [5−9].

Zhou et al. [1] proposed to establish a model sharing market called *Learnware Market* which assists users in solving their tasks by leveraging existing efforts instead of starting from scratch. The basic stages of learnware market is illustrated in Fig. 1. The market maintains a repository of learnwares for users to download and reuse. A *learnware* is comprised of a well-trained model and a *specification*. The model is trained by the developer on her own dataset and the specification is assigned by the market for explaining the purpose and/or specialty of the model.

The market is responsible for a submitting stage and a deploying stage. In the submitting stage, the market requires the developers to upload their models without revealing the training data. Then, the market assigns a specification for each model and stores them in the repository. In the deploying stage, the users wish to download and reuse the suitable models for their own tasks. And then the market requires them to generate their specifications according to their task requirements and submit the specifications to the market instead of exposing their private data. The market would identify the proper models by specification matching and return them to the users.

Consequently, a foundation of the learnware proposal is the specification matching between developers and users, enabling the successful reuse of appropriate models. The specifications enable the market to know the abilities of models and the requirements of users. Hence, to realize the mechanism of learnware, a refined specification design is supposed to satisfy the property of preserving precise sketch of model abilities while not exposing training data.

The previous approach [10] utilizes reduced kernel mean embedding (RKME) to generate a reduced set of kernel mean embedding (KME) as the specification [11]. It reserves the ability with a concise representation which does not expose the original data. This approach first defines the kernel function $k(\cdot,\cdot)$ corresponding to the RKHS $\mathcal{H}$, then reconstructs the data distribution by using a reduced set by the RKHS norm. The reduced set of KME is used as the specification of the task and it has fewer elements than the original data.

Received February 2, 2024; accepted August 13, 2024

E-mail: lim@nju.edu.cn

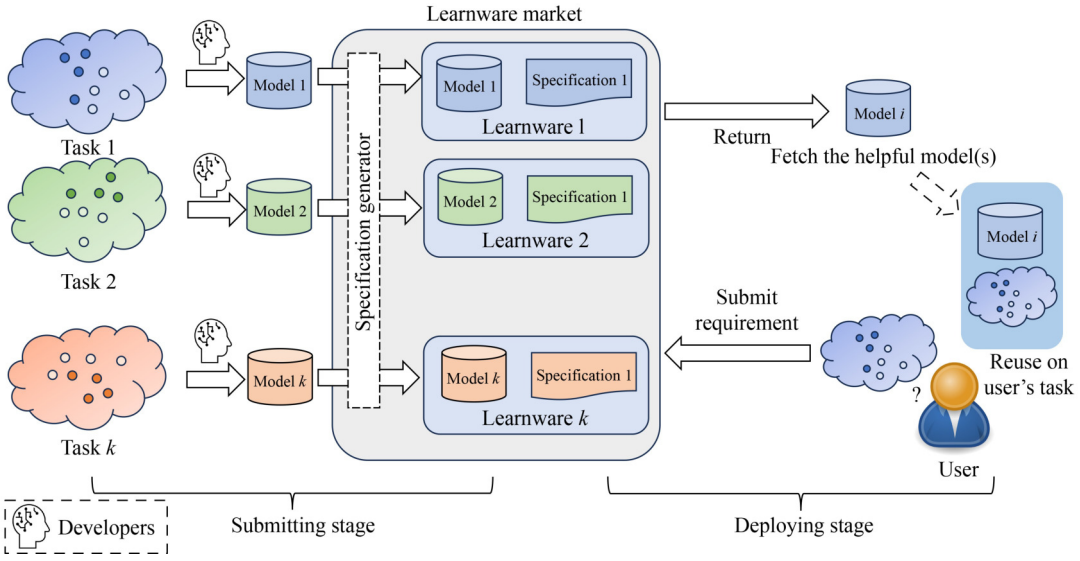

Fig. 1 The learnware market and the involved two stages

However, RKME specification may has the following limitation: The kernel function is fixed and hard to select. Once the kernel function is defined by the market, all the specifications in such RKHS $\mathcal{H}$ are secured. In other words, in such specification space, the distance metric is fixed. The market cannot adjust the distance metric according to the requirements of practical application scenario. Such limitation is just like the fixed kernel based distance metric is less flexible than the metric learning based distance metric.

For example, there are three tasks T_1, T_2, T_3 in the market (T_1 is a user's task seeking for a suitable model, T_2, T_3 are well-trained tasks with provided models M_2, M_3) and the market needs to identify a proper model reused for T_1 (illustrated in Fig. 2). In ground truth, T_2 is similar to T_1 and suitable for reusing while T_3 is an unsimilar task which is not supposed be identified. In principle, the market should assign similar specifications to T_1 and T_2 and push away T_3 in specification space. However, once the market select a fixed kernel space $\mathcal{H}$ where T_3's specification may be closer to T_1 than T_2, the distance metric cannot be adjusted, which leads to an incorrect reusable model identifying. Obviously, the flexibility of the specification is limited by the fixed kernel space. Any pre-defined RKME-based specification is unlikely to satisfy all the requirements of the market.

To the best of our knowledge, no prior work has proposed to learn the specification space. In this paper, we propose a novel neural network *SpecNet* to learn the embeddings of different task distributions in a low-dimensional latent space and the embeddings can be used as the specifications. In our approach, to distinguish whether a model is reusable, our learned metrics pulls reusable tasks closer together and pushes away tasks that cannot be reused. Therefore, the distance in such specification space could indicate the similarity or the reusability between tasks. The learning methodology employed in our approach is founded upon the representation of unordered sets, thereby ensuring that the specification preserves a succinct and accurate abstraction of the original task distribution.

The contributions of this paper are summarized as follows:

- We are the first to propose learning specifications, which makes the specification matching more flexible and adjustable in the learnware market.
- We propose a novel neural network *SpecNet* with a well-defined objective for acquiring the distance metric

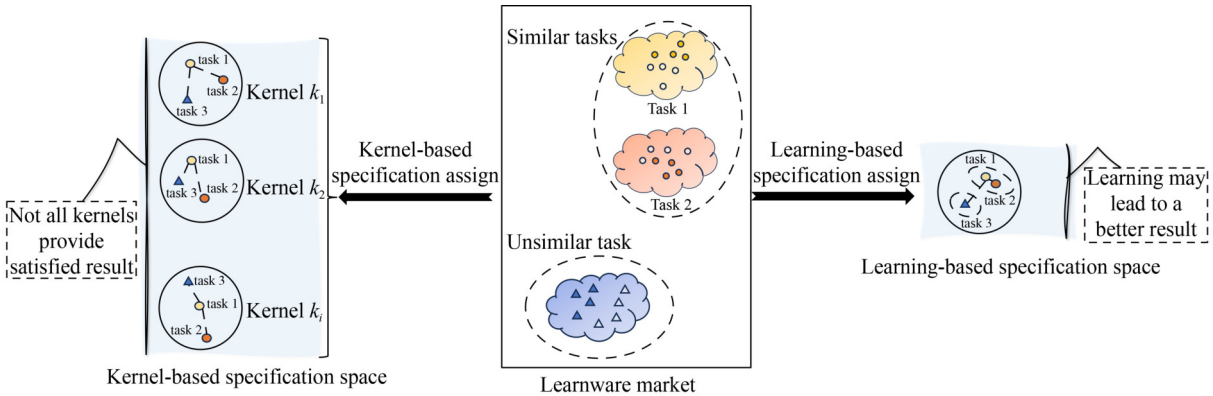

Fig. 2 Difference between our approach and kernel-based approach. In this example, the truth is that T_2 is more similar to T_1 than T_3, so model M_2 is more suitable to be reused for T_1 than M_3. However, in kernel-based specification assigning, the distance metric is fixed and not all the kernels can provide satisfied results. Our approach can help to learn a desired metric distance: $d(sp_1, sp_2) < d(sp_1, sp_3)$

between distinct task distributions. Empirical results demonstrate the efficacy of our approach in acquiring a pertinent distance metric that faithfully reflects the similarities among diverse tasks.

The subsequent sections of this paper are structured as follows. Section 2 provides a comprehensive review of related work. Section 3 introduces the specification learning by a novel neural network, referred to as *SpecNet*. Section 4 elucidates the experimental results, and Section 5 concludes the paper.

2 Related work

For the purpose of model reusability, the learnware paradigm is proposed. In this paradigm, Wu et al. [10] propose a specification to describe the model. The specification is defined as a set of features based on reduced kernel mean embedding, which can be used to identify models. In this approach, learnware can be reused by RKME matching and a task selector. RKME is a variant of Kernel Mean Embedding (KME), which is a method for mapping probability distributions to points in a Reproducing Kernel Hilbert Space (RKHS). The primary purpose of RKME is to serve as a specification for pre-trained machine learning models. This allows users to determine the applicability of a model to a new task without accessing the original training data, thus facilitating model reuse while respecting data privacy. Based on RKME, Zhang et al. [3] develops a scheme to reuse the learnwares for unseen jobs. Apart from that, Ding and Zhou [12] use outlier detectors as the specification and develop a boosting-based approach to deploy models. Tan et al. [2] proposed an approach for handling heterogeneous feature spaces by integrating subspace learning in the specification design.

Domain adaptation [13−17] and transfer learning [18−22] are related topics to learnware paradigm, which aims to adapt the related domains to the new tasks. Domain adaptation is the process of adapting or learning from a source data distribution to the target distribution [7]. Transfer learning is the process of transferring the knowledge of the problem to a related new one [23]. The key difference between our paradigm and these two topics is that our paradigm is based on the market specification matching, and the other two are based on the raw data. These approaches assume that the raw data of source domain or learned knowledge always benefits to the target problem. Some works assume that the source data is always avaliable to all target users or the source model is always tuneable for the target problem. Besides, they focus only on how to retrieve knowledge from source raw data, which does not protect data privacy. However, in learnware market scenario, most models does not benefit to the target domain. Selecting and reusing the suitable existing models is the priority of learnware market.

Set representation learning, as explored in works such as Maturana and Scherer [24], Qi et al. [25,26], is a research domain dedicated to acquiring effective representations for sets of data points. From a structural perspective, a learning task dataset manifests as an unordered set comprising vectors and their corresponding labels. While the majority of deep learning endeavors have concentrated on conventional input structures such as sequences (as observed in speech and language processing), images, and volumes (pertaining to video or 3D data), there has been a relative dearth of research in the realm of deep learning for datasets. Vinyals et al. [27] have explored the utilization of a read-process-write network featuring an attention mechanism, demonstrating its efficacy in processing unordered input sets, with a particular demonstration of the network's capacity to perform numerical sorting. However, their work, being more focused on generic sets and natural language processing applications, overlooks the nuanced role of geometry within sets. In contrast, Qi et al. [28] have proposed a neural network architecture designed to directly consume individual data points, catering to various tasks, including object classification and part segmentation.

3 Specification learning

In this section, we first present the formulation of the problem. Then, we state the architecture of *SpecNet*, our proposed neural network for specification learning. Finally, we give an analysis of several properties of the proposed approach.

3.1 Formulation

Consider a learnware market comprising a set of learnwares $\{L_i\}_{i=1}^N$, where N is the number of learnwares. Each learnware $L_i = (M_i, sp_i)$ denotes the ith learnware in the market, with M_i representing the model trained on dataset $(X^{(i)}, Y^{(i)})$ and sp_i denoting the corresponding specification. When a user seeks a model for her task, she could generate a specification sp_u based on her own data and submit it to the market. The market, in turn, identifies and returns the model $M_i, i = \arg\min_i d(sp_u, sp_i)$, that aligns most closely with the user's requirements, utilizing the distance metric $d(\cdot,\cdot)$ to measure similarity between specifications. Intuitively, the specifications of two datasets are expected to be similar if the datasets are designed for the same task, and conversely, dissimilar if the data distributions differ.

To achieve this, we propose to *learn* a specification generator $G(\cdot): \mathcal{X} \to \mathbb{R}^l$, where a dataset $X^{(i)}$ serves as input, producing a specification vector sp_i as output. In contrast to prior work [10] which assigns specification using a reduced set of KME in a fixed kernel space, our approach learns the specification generator from dataset-indicator pairs $\{(X^{(i)}, Z^{(i)})\}_{i=1}^N$. Here, $X^{(i)}$ represents a dataset, $Z^{(i)}$ is an indicator vector describing the classes appearing in the corresponding task, and N is the total number of datasets. Especially, $Z^{(i)}$ is a one-hot vector, where the jth element is 1 if the jth class is present in the task, and 0 otherwise. To uncover the inherent essence of each task and capture the similarities between different tasks, we employ a dual-objective function comprising a cross-entropy loss and a triplet ranking loss. The cross-entropy loss captures the unique characteristics of each task, utilizing label space information as the sole supervised knowledge available. Simultaneously, the triplet ranking loss measures the similarity between distinct tasks, aiding in minimizing the distance between similar tasks and maximizing the distance between dissimilar

ones. The overall objective function is formally defined as follows:

$$\min_{G,C} \mathcal{L} = \frac{1}{N}\sum_{i}^{N} \mathcal{L}_{ce}(C(G(X^{(i)})), Z^{(i)}) + \frac{\lambda}{N_{tri}} \sum_{a,p,n}^{N} \mathcal{L}_{triplet}(G(X^{(a)}), G(X^{(p)}), G(X^{(n)})), \quad (1)$$

where $\mathcal{L}_{ce}$ denotes the cross-entropy loss, $\mathcal{L}_{triplet}$ represents the triplet ranking loss, λ is a hyperparameter for balancing the two losses, N_{tri} is the number of triplets in the training set, $C(\cdot)$ is a network for predicting the indicators, and $X^{(a)}, X^{(p)}, X^{(n)}$ are the anchor, positive and negative datasets respectively.

Unlike conventional machine learning tasks that learn mapping functions from a single input vector to an output vector, $G(\cdot)$ is tasked with processing an unordered, variable-length task dataset as input and synthesizing the task specification into a single fixed-length vector. We instantiate the learning task by introducing a novel neural network *SpecNet*, designed to learn a unified specification generator $G(\cdot)$.

3.2 SpecNet

The architecture of *SpecNet* is illustrated in Fig. 3. It initially compresses the input dataset into a singular specification vector and subsequently predicts task indicators, which refines the learning of specifications that align with the distance metric between different tasks.

Compressing the entire input dataset into a singular specification vector presents challenges due to the inherent essence of the input, which constitutes an unordered set of vectors with an variable size. These characteristics introduce complexities in generating a fixed-length deep representation.

To address this issue, we devise a deep model for specification generation, which is illustrated in the upper left dashed box of Fig. 3. The model $G(\cdot)$ processes an entire

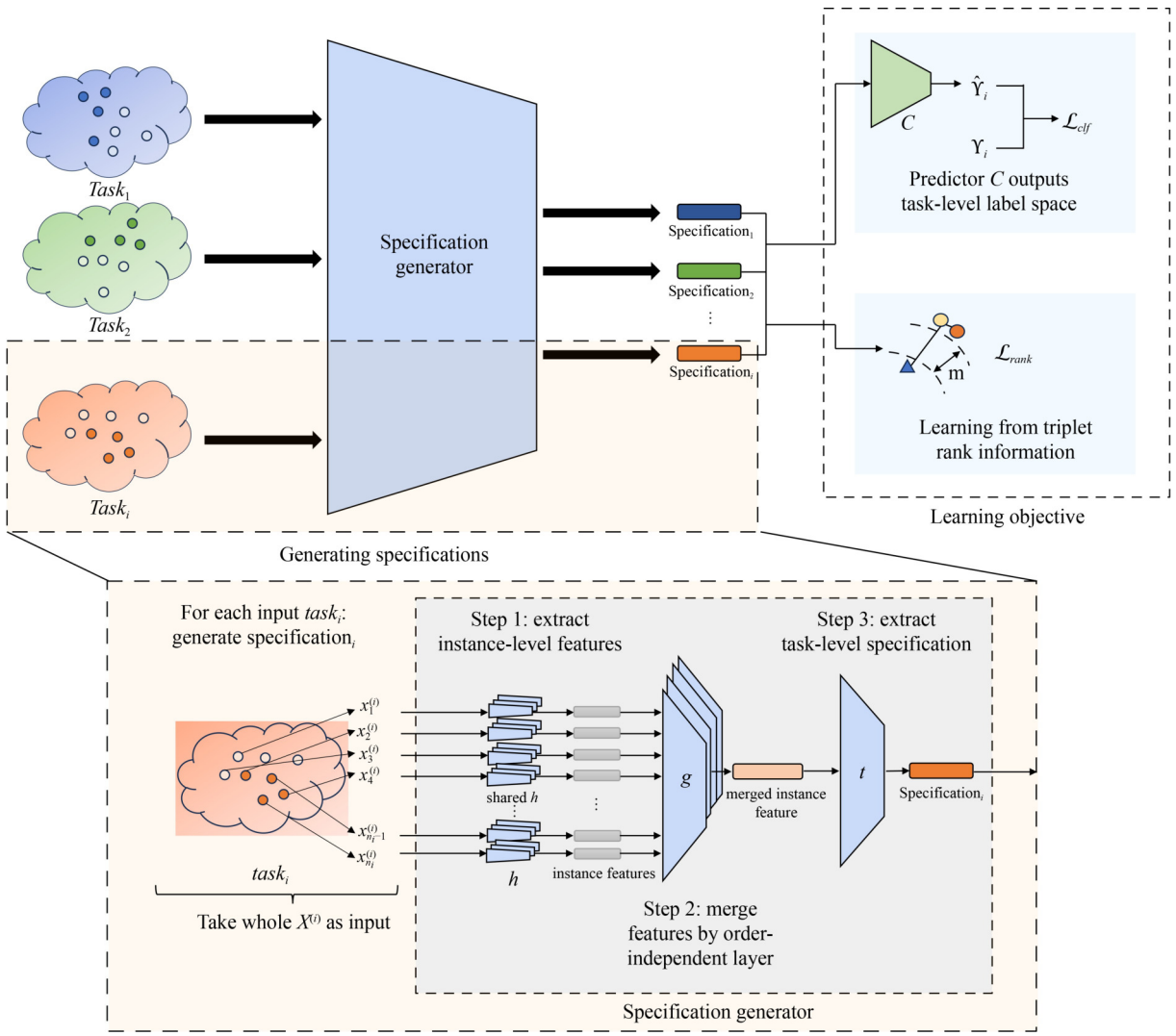

Fig. 3 The overall architecture of the proposed *SpecNet*. Firstly, the specification generator takes each whole dataset $task_i$ as an input, and outputs a vector $specification_i$. Subsequently, the generated specification vectors are inputted into the classifier for task indicator prediction. Simultaneously, the triplet ranking loss is employed to gauge the similarity between distinct tasks. In our approach, we make the specification learnable by backpropagation. The detailed structure of the specification generator is shown in the lower dashed box. Especially, an input "instance" of specification generator is an unordered set of vectors $X^{(i)} = \{x_j^{(i)}\}_{j=1}^{n_i}$

unordered dataset $X^{(i)}$ as input and produces a specification vector of dimension l. The detailed pipeline is delineated in lower dashed box in Fig. 3 as follows: (1) Extract the feature of each instance within a dataset through a shared weight feature extraction network $h(\cdot)$; (2) Merge the instance-level features into a task-level feature through the order-independent network $g(\cdot)$; (3) Extract the specification vector from the merged feature by a task-level feature extraction network $t(\cdot)$.

The capability to compress an unordered set into a vector relies on the functionality of the order-independent network. This network is designed to confer invariance of the model output to the input order, achieved through the application of a symmetric function on transformed elements within the set. Suppose $f: \mathcal{X} \to \mathbb{R}$ is a single variable continuous set function, we approximate it by applying a symmetric function on transformed elements in the set:

$$f(\{x_1, ..., x_n\}) \approx g(h(x_i), ..., h(x_n)), \tag{2}$$

where $h: \mathbb{R}^D \to \mathbb{R}^K$ and $g: \underbrace{\mathbb{R}^K \times \cdots \times \mathbb{R}^K}_{n} \to \mathbb{R}$ is a symmetric function. Therefore, we can learn a number of f's to capture different properties of the set $X^{(i)}$.

In the implementation, we use a composition of a single variable function and a max pooling function as the order-independent network $g(\cdot)$. And we use a multi-layer convolutional neural network as the feature extraction network $h(\cdot)$. We use a multi-layer perceptron as the task-level feature extraction network $t(\cdot)$. Formally, specification generator is defined as follows:

$$\begin{aligned} sp_i = G(X^{(i)}) &= t([f_1, ..., f_K]) \\ &= t\left(\begin{bmatrix} g_1(h_1(x_1^{(i)}), ..., h_1(x_{n_i}^{(i)})) \\ \vdots \\ g_K(h_K(x_1^{(i)}), ..., h_K(x_{n_i}^{(i)})) \end{bmatrix}^{\mathrm{T}}\right), \end{aligned} \tag{3}$$

where $t: \mathbb{R}^K \to \mathbb{R}^l$ is a transformation network.

In the context of specification learning, we formulate the objective function to capture the intrinsic characteristics of each task and the similarities between different tasks. The key concept is to establish task relationships through the exploitation of label space information. Specifically, for each task denoted as $(X^{(i)}, Y^{(i)})$, we assign an indicator vector $Z^{(i)}$ to signify the classes present in the task. The generator $G(\cdot)$, in conjunction with a classifier $C(\cdot)$, jointly learns the transformation $C(G(X^{(i)})) \to Z^{(i)}$ as the primary objective. The principal objective function is defined as follows:

$$\min_{G,C} \frac{1}{N} \sum_{i}^{N} \mathcal{L}_{ce}(C(G(X^{(i)})), Z^{(i)}), \tag{4}$$

where $\mathcal{L}_{ce}(\cdot, \cdot)$ is the cross-entropy loss.

To enhance the capture of task similarities, we incorporate a triplet ranking loss as an auxiliary objective function. This choice is motivated by the capability of the ranking loss to decrease the distance between similar tasks and magnify the distance between dissimilar tasks. Triplets are constructed through the random sampling of an anchor task, a positive task, and a negative task. Here, the positive task signifies a task that is akin to the anchor task, while the negative task represents a task that diverges from the anchor task. We define the similarity between tasks using the number of the same elements in the task indicator vectors, that is, by calculating the number of categories that appear directly in common between two tasks: $Z_a \cdot Z_p > Z_a \cdot Z_n$. For example, there are three tasks: task A, task B, and task C. The indicator vectors of the three tasks are $[1,1,0,0]$, $[0,1,1,0]$, and $[0,0,1,1]$, respectively. The anchor task is task A, the positive task is task B, and the negative task is task C. The primary objective is to ensure that the distance between the anchor task and the positive task is smaller than the distance between the anchor task and the negative task. The triplet ranking loss function is formally defined as follows:

$$\min_{G} \frac{\lambda}{N_{tri}} \sum_{a,p,n}^{N} \left[d(sp_a, sp_p) - d(sp_a, sp_n) + m \right]_+, \tag{5}$$

where $d(\cdot, \cdot)$ is the distance between two specifications, $[\cdot]_+ = \max(0, \cdot)$ is the ReLU function, m is a margin hyperparameter, N_{tri} is the number of triplets in the training set, sp_a, sp_p, sp_n are the anchor, positive and negative specification vectors from $X^{(a)}, X^{(p)}, X^{(n)}$ respectively.

By combining Eqs. (4) and (5), the final objective function 1 is derived. The comprehensive framework is trained end-to-end through the application of backpropagation.

The preceding discussion illustrates the resolution of the central challenge in learnware specification learning. Following the optimization of the objective function, the specification generator $G(\cdot)$ is capable of producing a specification vector sp_i corresponding to each task $(X^{(i)}, Y^{(i)})$. For the sake of simplicity, the distance metric employed in this study is defined as the Euclidean distance between these specification vectors: $d(sp_i, sp_j) = |sp_i - sp_j|_2$.

3.3 Theoretical analysis

In this section, we analyze some properties of our approach. Given two input datasets $X^{(1)}$ and $X^{(2)}$, the specification generator aims to output similar specification vectors sp_1 and sp_2 if they are accomplishing the same task. However, empirically there is often bias between $X^{(1)}$ and $X^{(2)}$ since the data variance is inevitable in the real world. Hence, how the data perturbation affects the specification is a key issue. Our approach is supposed to be robust to these small perturbation of the input dataset. Here we give a proposition to explain the robustness of our approach.

We define $f: \mathcal{X} \to \mathbb{R}$ as a continuous set function which can be decomposed as $f = g \circ h$, where h is a feature network for $x_i \in X$ and g is a symmetric function mapping the feature set to a vector. Let $u = \max_{x_i \in X} \{h(x_i)\}$ which maps a set X in $\mathcal{X}$ to a vector in $\mathbb{R}^K$ be the front-end sub-network of g. Here, $\max\{\cdot\}$ is a vector max operator that outputs the element-wise maximum. The following propositions show that small data perturbation in the input set will not significantly change the output.

Proposition 1 Given $f = g \circ h$ and $u = \max_{x_i \in X} \{h(x_i)\}$. Then,

$\forall X^{(1)}, \exists C, \mathcal{N} \subseteq \mathcal{X}, f(X^{(2)}) = f(X^{(1)})$ if $C \subseteq X^{(2)} \subseteq \mathcal{N}$;

Proof For the jth dimension of u, there exists at least one $x_j \in \mathcal{X}$ that makes $h_j(x_j) = u_j$, where h_j is the jth dimension of the output vector from h. Take the C as the union of all x_j for $j = 1, ..., K$. C can be seen as the minimal set that makes $u(X^{(1)}) = u(C)$. Therefore, $\forall X^{(1)}, \exists C \subseteq \mathcal{X}, f(X^{(2)}) = f(X^{(1)})$ if $C \subseteq X^{(2)}$.

There exists extra data perturbation which makes $h(x) \leqslant u(X^{(1)})$ at all dimensions. Adding these perturbation data to C does not change the output of u. Therefore, by adding the union of such data to C, the union can be seen as the maximal set $\mathcal{N}$ that makes $u(X^{(1)}) = u(\mathcal{N})$. Therefore, $\forall X^{(1)}, \exists \mathcal{N} \subseteq \mathcal{X}, f(X^{(2)}) = f(X^{(1)})$ if $X^{(2)} \subseteq \mathcal{N}$. □

Proposition 1 shows that $f(X^{(1)})$ and $f(X^{(2)})$ can be consistent up to a small data perturbation on condition that $X^{(2)}$ is in the range of C and $\mathcal{N}$. This proposition shows that the output of our model is robust to small perturbation of the input dataset.

Obviously, the minimal set C is a core set that makes $f(X) = f(C)$. Hence, whether our approach could capture the core set C is another key issue. Our approach is supposed to be able to capture the minimal core set C by the parameter K in Eq. (2). Here we give a proposition to explain it.

Proposition 2 Given $f = g \circ h$ and $u = \max_{x_i \in X}\{h(x_i)\}$. Then, $|C| \leqslant \mathcal{K}$.

Proof For the jth dimension of u, there exists at least one $x_j \in \mathcal{X}$ that makes $h_j(x_j) = u_j$, where h_j is the jth dimension of the output vector from h. Take the C as the union of all x_j for $j = 1, ..., K$. Hence, the number of elements in C is upper bounded by K. Therefore, $|C| \leqslant K$. □

Proposition 2 shows that the size of $|C|$ is upper bounded by the dimension number K in Eq. (2), which means that as long as the parameter K is large enough, our approach could capture the core set of the input dataset.

Propositions 1 and 2 affirm the robustness and stability of our approach. The robustness is analogously acquired through a principle akin to sparsity in machine learning models. Intuitively, our approach learns to succinctly encapsulate the essence of a task dataset through a sparse set of key data.

4 Experiments

To demonstrate that our framework could achieve a better performance than the existing specification design, we conduct experiments to verify the following questions. (1) Can the most useful learnware be identified via our specification? (2) Is our specification still effective when the market scale is increasing with new unknown classes?

Datasets We use *Tiny-ImageNet* [29] and *CIFAR-10* [30] as our benchmark datasets. For simplicity, we only simulate binary classification tasks from these datasets in the following experiments. Firstly, we simulate a pre-train task set (pre-train set) for training our *SpecNet* and then utilize the submodule $G(\cdot)$ to generate the specifications. At the submitting stage, to construct the learnware market, we simulate the developers' task set (market set) and upload a well-trained model for each task in this set. Meanwhile, the market assigns a specification for each model via the submodule $G(\cdot)$. At the deploying stage, we simulate a users' task set (user set) to evaluate the reuse performance. For each user's task, the specification is also generated by the submodule $G(\cdot)$. Hence, the user could use the specification similarity to retrieve the top-1 matched model from the learnware market. In this paper, the retrieved models are directly reused on the users' tasks without fine-tuning.

Evaluation metrics The evaluation metrics include the average accuracy of the reused models and the top-k hit rate of retrieved models. They are defined as follows:

$$\text{average reuse accuracy} = \frac{\sum_{i=1}^{N} acc(task_i)}{N}, \tag{6}$$

where $acc(task_i)$ is the accuracy of the reused model on $task_i$. N is the number of users' tasks.

$$\text{top-}k\text{ hit rate} = \frac{\sum_{i=1}^{N} hit_i}{N}, \tag{7}$$

where hit_i is 1 if the retrieved model is in the top-k best models, otherwise 0. N is the number of users' tasks. More details of the experimental setup are provided in the following subsections.

Baselines To compare the learnware retrieval and reusing performance, we compare different specification design under the same settings. We compare our approach with the following baselines:

(1) *RKME* [10]: a kernel-based specification which uses the reduced set in the RKHS to represent the data distribution. In this experiment, the feature extractor is also adopted like the original paper's setting.

(2) *Random*: a random baseline, which randomly selects a model from the market.

(3) *Best*: the performance of reusing the best model in the market from the ground truth. It is the upper bound of the reuse performance.

4.1 Effectiveness

To demonstrate that the specifications learned via our framework can help identify useful models, we conduct experiments to verify the effectiveness of our approach.

Experimental setup In this part, we assume that the pre-train set, market set and user set all share the same label space. For the *Tiny-ImageNet* dataset, we randomly select 10 classes as the label space. Then we randomly simulate the binary classification tasks in the lable space to construct the pre-train set, market set and user set. The data size of each simulated task is about 400−500. For the *CIFAR-10* dataset, we use all 10 classes as the label space. The task sets are simulated in the same way. The data size of each simulated task is about 3000−5000.

The pre-train set size in this experiment is 100. The market set size ranges from 200 to 1000, while the user set size is 500. We conduct the whole experiment 10 times and report the average results.

Results The evaluation results are presented in Tables 1, 2

Table 1 Average accuracy (%) of reused model on users' tasks (Tiny-ImageNet). The best results are in bold. The market set size is ranging from 200 to 1000

Method	Market set size				
	200	400	600	800	1000
Random	15.62	14.93	15.27	14.69	12.98
RKME	40.83	44.65	45.37	47.70	50.24
SpecNet	**73.24**	**74.37**	**75.48**	**75.63**	**76.05**
Best	77.88	79.32	79.85	80.11	80.36

Table 2 Average accuracy (%) of reused model on users' tasks (CIFAR-10). The best results are in bold. The market set size is ranging from 200 to 1000

Method	Market set size				
	200	400	600	800	1000
Random	17.19	16.11	17.10	15.59	14.08
RKME	41.23	42.43	44.74	45.47	47.96
SpecNet	**78.64**	**78.75**	**80.20**	**81.38**	**81.79**
Best	81.10	82.19	83.54	83.71	84.92

and Figs. 4, 5. It is evident that our specification design outperforms the baselines.

In Tables 1 and 2 the results indicate that our approach could achieve the highest average reuse accuracy in all settings for both benchmark datasets. For example, in Tiny-ImageNet dataset, when the market set size varies from 200 to 1000, our approach could achieve at least 73.24% accuracy, which is much higher than the baselines. The average reuse accuracy increases when the market set size increases. The reason is that when the market set size increases, the suitable models are more possible to appear in the market. The results of "Best" method is the upper bound of the reuse performance. The small gap between "Best" and our approach means that we almostly retrieve the most suitable model for the user's task. In general, our proposed framework could help to generate more accurate and efficient specifications.

Figures 4 and 5 illustrate the likelihood of users successfully retrieving the top-k best models available in the market. We can see that our approach could achieve the highest top-k hit rate in all settings for both benchmark datasets. The evaluation performance is almostly 2 times higher than the baselines. The results also showed that the top-k hit rate decreases with the

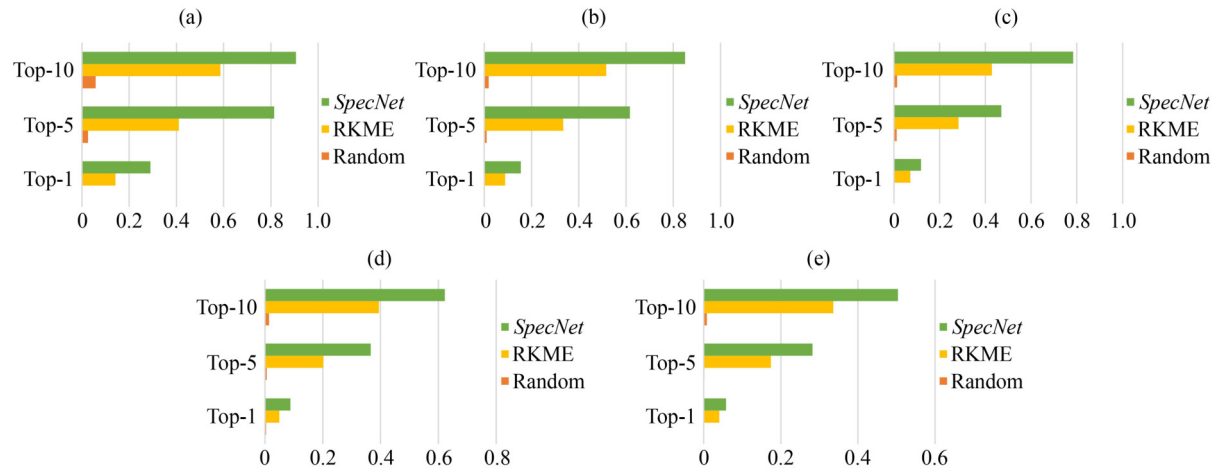

Fig. 4 Top-k hit of the retrieved models on users' tasks of Tiny-ImageNet datasets. The higher the metric value, the better the performance. It can be observed that our approach could achieve the highest top-k hit rate in all settings. (a) Top-k hit rate (market size=200); (b) Top-k hit rate (market size=400); (c) Top-k hit rate (market size=600); (d) Top-k hit rate (market size=800); (e) Top-k hit rate (market size=1000)

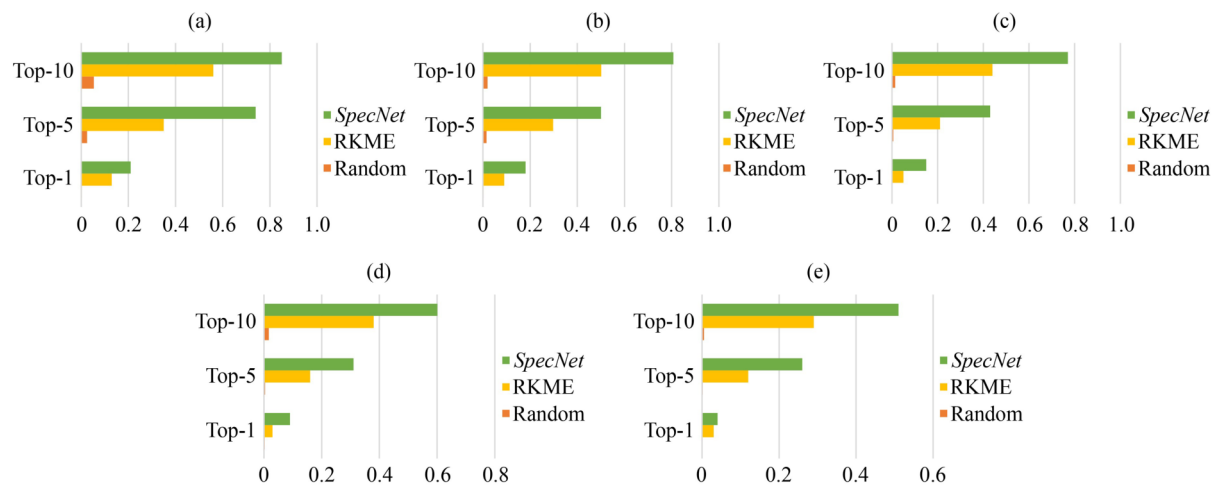

Fig. 5 Top-k hit of the retrieved models on users' tasks of CIFAR-10 datasets. The higher the metric value, the better the performance. It can be observed that our approach could achieve the highest top-k hit rate in all settings. (a) Top-k hit rate (market size=200); (b) Top-k hit rate (market size=400); (c) Top-k hit rate (market size=600); (d) Top-k hit rate (market size=800); (e)Top-k hit rate (market size=1000)

increase of market set size. This is because when the market set size increases, more suitable models are available in the market, which makes it harder to retrieve the best models. The market does not return the best top-k returned models, the users could still retrieve the well performed models from the market. In general, our approach could still achieve the highest top-k hit rate competing with the baselines.

4.2 Scalability

To answer the question that whether our approach could still achieve a good performance when there are new unknown classes in the market set, we conduct experiments to verify the scalability of our approach. Intuitively, our specification could be used as a general metric for the similar unknown task distributions. Hence, it is supposed to be efficient for the user tasks with unseen classes.

Experimental setup To verify our hypothesis, we assume that the pre-train set and the market set share different label spaces. We conduct experimental evaluation in two senarios in this part:

a) In-domain setting: The pre-train set, market set and user set are all from Tiny-ImageNet dataset. However, the label space of the pre-train set (Y_{pre}) and the label space of the market set (Y_{mar}) are disjoint.

b) Out-of-domain setting: The pre-train set is from Tiny-ImageNet dataset, while the market set and user set are from CIFAR-10 dataset. Also, the label space of the pre-train set (Y_{pre}) and the label space of the market set (Y_{mar}) are disjoint.

We compare our approach with the above baselines. The evaluation metrics are the average reuse accuracy of the retrieved models.

Results The results are shown in Table 3. We can see that our approach could also achieve the highest accuracy in both senarios. As shown in Table 3, in the in-domain setting, our approach could achieve at least 60.5% accuracy, which is much higher than the baselines. When the number of unknown classes ranges from 10 to 40, our approach could always achieve stable performance. In the out-of-domain setting, the tasks of pre-train set and the market set are quite unsimilar. The results under this setting can more clearly demonstrate what happens with our method when a user submits an unknown task. The performance is still compariable with the in-domain setting and much higher than the baselines. The evaluation results demonstrate that our approach could learn the general representation from the datasets with label set Y_{pre}, and the specification knowledge or distribution knowledge could be transfered to other tasks with unknown classes. Such phoenomenon is also appeared in the metric learning or transfer learning scenario.

Table 3 Average reuse accuracy (%) of two senarios. The label space of the pre-train set (Y_{pre}) and the label space of the market set (Y_{mar}) are disjoint. The best results are in bold

Senario	In-domain				Out-of-domain
	# of unknown classes				
method	10	20	30	40	
Random	6.9	6.1	6.6	7.0	8.2
RKME	46.4	45.7	48.5	45.3	37.6
SpecNet	**60.5**	**62.3**	**61.7**	**62.9**	**53.3**

The results indicate that our approach could benefit from the distance metric learning. The learned specification space has the advantage of generalization. The specification could still be effective for the user tasks with unseen classes. Therefore, our framework has the ability to extend the specifications to real-world and large-scale learnware markets.

5 Conclusion

In this paper, we propose to learn specifications in the learnware paradigm, which could provide more flexible representations of task distributions to identify the reusable learnwares. A novel learning framework called *SpecNet* is proposed to learn the specifications end-to-end with the information of task label space. It takes the whole task datasets as input training data, and then learns the specifications by the proposed objective function. After that, the specification generator in *SpecNet* could assign universal specifications for both developers and users. Especially, since the specifications are learned from the various market tasks rather than hard defined metric, our learned specifications are more accurate and efficient for learnware matching. The framework is able to output informative specifications for both seen and unseen tasks. Empirical studies validate the effectiveness and scalability.

SpecNet is a general framework for learning specifications. In the future, we will explore more effective and efficient methods based on it to learn the specifications. Utilizing the information from task models and data distributions simultaneous will be investigated in future work. Moreover, we will explore the ability of *SpecNet* to learn specifications for heterogeneous tasks.

Acknowledgements This research was supported by the National Natural Science Foundation of China (Grant Nos. 62076121, 61921006) and the Major Program (JD) of Hubei Province (2023BAA024).

Competing interests Ming Li is an Editorial Board member of the journal and a co-author of this article. To minimize bias, they were excluded from all editorial decision-making related to the acceptance of this article for publication. The remaining authors declare no conflict of interest.

References

1. Zhou Z H. Learnware: on the future of machine learning. Frontiers of Computer Science, 2016, 10(4): 589−590
2. Tan P, Tan Z H, Jiang Y, Zhou Z H. Towards enabling learnware to handle heterogeneous feature spaces. Machine Learning, 2024, 113(4): 1839−1860
3. Zhang Y J, Yan Y H, Zhao P, Zhou Z H. Towards enabling learnware to handle unseen jobs. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence. 2021, 10964−10972
4. Guo L Z, Zhou Z, Li Y F, Zhou Z H. Identifying useful learnwares for heterogeneous label spaces. In: Proceedings of the 40th International Conference on Machine Learning. 2023, 12122−12131
5. Hoffman J, Tzeng E, Park T, Zhu J Y, Isola P, Saenko K, Efros A, Darrell T. CyCADA: cycle-consistent adversarial domain adaptation. In: Proceedings of the 35th International Conference on Machine Learning. 2018, 1994−2003
6. Ben-David S, Blitzer J, Crammer K, Kulesza A, Pereira F, Vaughan J W. A theory of learning from different domains. Machine Learning, 2010, 79(1−2): 151−175

7. Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning. 2015, 1180−1189
8. Kumar A, Ma T, Liang P. Understanding self-training for gradual domain adaptation. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 507
9. Luo Y, Wang Z, Huang Z, Baktashmotlagh M. Progressive graph learning for open-set domain adaptation. In: Proceedings of the 37th International Conference on Machine Learning. 2020, 600
10. Wu X Z, Xu W, Liu S, Zhou Z H. Model reuse with reduced kernel mean embedding specification. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(1): 699−710
11. Smola A, Gretton A, Song L, Schölkopf B. A hilbert space embedding for distributions. In: Proceedings of the 18th International Conference on Algorithmic Learning Theory. 2007, 13−31
12. Ding Y X, Zhou Z H. Boosting-based reliable model reuse. In: Proceedings of the 12th Asian Conference on Machine Learning. 2020, 145−160
13. Ben-David S, Blitzer J, Crammer K, Pereira F. Analysis of representations for domain adaptation. In: Proceedings of the 19th International Conference on Neural Information Processing Systems. 2006, 137−144
14. Germain P, Habrard A, Laviolette F, Morvant E. A new PAC-Bayesian perspective on domain adaptation. In: Proceedings of the 33rd International Conference on Machine Learning. 2016, 859−868
15. Liu M Y, Tuzel O. Coupled generative adversarial networks. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 469−477
16. Chang W, Shi Y, Tuan H D, Wang J. Unified optimal transport framework for universal domain adaptation. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 2140
17. Zhu Y, Wu X, Qiang J, Yuan Y, Li Y. Representation learning via an integrated autoencoder for unsupervised domain adaptation. Frontiers of Computer Science, 2023, 17(5): 175334
18. Pan S J, Yang Q. A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(10): 1345−1359
19. Ding Y X, Wu X Z, Zhou K, Zhou Z H. Pre-trained model reusability evaluation for small-data transfer learning. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. 2022, 2710
20. Wang B, Mendez J A, Cai M B, Eaton E. Transfer learning via minimizing the performance gap between domains. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 955
21. Houlsby N, Giurgiu A, Jastrzebski S, Morrone B, De Laroussilhe Q, Gesmundo A, Attariyan M, Gelly S. Parameter-efficient transfer learning for NLP. In: Proceedings of the 36th International Conference on Machine Learning. 2019, 2790−2799
22. Yang Y, Guo J, Li G, Li L, Li W, Yang J. Alignment efficient image-sentence retrieval considering transferable cross-modal representation learning. Frontiers of Computer Science, 2024, 18(1): 181335
23. Wang Z, Dai Z, Póczos B, Carbonell J. Characterizing and avoiding negative transfer. In: Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, 11293−11302
24. Maturana D, Scherer S. VoxNet: a 3D convolutional neural network for real-time object recognition. In: Proceedings of 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2015, 922−928
25. Qi C R, Su H, Nießner M, Dai A, Yan M, Guibas L J. Volumetric and multi-view CNNs for object classification on 3D data. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. 2016, 5648−5656
26. Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J. 3D ShapeNets: a deep representation for volumetric shapes. In: Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. 2015, 1912−1920
27. Vinyals O, Bengio S, Kudlur M. Order matters: sequence to sequence for sets. In: Proceedings of the 4th International Conference on Learning Representations. 2016
28. Qi Charles R, Su H, Kaichun M, Guibas L J. PointNet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 77−85
29. Le Y, Yang X. Tiny ImageNet visual recognition challenge. In: Proceedings of the Stanford CS231N Convolutional Neural Networks for Visual Recognition. 2015
30. Krizhevsky A. Learning multiple layers of features from tiny images. Technical Report TR-2009. Toronto: University of Toronto, 2009

Zhi-Yu Shen received the BEng degree in computer science and technology from Nanjing University, China in 2014. Currently he is a PhD candidate in the Department of Computer Science and Technology, Nanjing University, and is a member of the LAMDA Group. His research interests mainly include machine learning and data mining, especially in domain adaptation and transfer learning.

Ming Li is currently a professor with the LAMDA group, the National Key Laboratory for Novel Software Technology, Nanjing University, China. His major research interests include machine learning and data mining, especially on software mining. He has served as the area chair of IJCAI, IEEE ICDM, etc., senior PC member of the premium conferences in artificial intelligence such as AAAI, and PC members for other premium conferences such as KDD, NeurIPS, and ICML. He is the founding chair of the International Workshop on Software Mining. He has been granted various awards including the PAKDD Early Career Award, the NSFC Excellent Youth Award, the New Century Excellent Talents program of the Education Ministry of China, etc.