A Protein-Context Enhanced Master Slave Framework for Zero-Shot Drug Target Interaction Prediction

Yuyang Xu, Jingbo Zhou, Haochao Ying*, *Member, IEEE*, Jintai Chen, Wei Chen, Danny Z. Chen, *Fellow, IEEE,* and Jian Wu *Member, IEEE*

Abstract—Drug Target Interaction (DTI) prediction plays a crucial role in in-silico drug discovery, especially for deep learning (DL) models. Along this line, existing methods usually first extract features from drugs and target proteins, and use drug-target pairs to train DL models. However, these DL-based methods essentially rely on similar structures and patterns defined by the homologous proteins from a large amount of data. When few drug-target interactions are known for a newly discovered protein and its homologous proteins, prediction performance can suffer notable reduction. In this paper, we propose a novel Protein-Context enhanced Master/Slave Framework (PCMS), for zero-shot DTI prediction. This framework facilitates the efficient discovery of ligands for newly discovered target proteins, addressing the challenge of predicting interactions without prior data. Specifically, the PCMS framework consists of two main components: a Master Learner and a Slave Learner. The Master Learner first learns the target protein context information, and then adaptively generates the corresponding parameters for the Slave Learner. The Slave Learner then perform zero-shot DTI prediction in different protein contexts. Extensive experiments verify the effectiveness of our PCMS compared to state-of-the-art methods in various metrics on two public datasets. *The Code and the processed Data will be open once the paper is accepted.*

Index Terms—Drug Target Interaction Prediction, Graph Neural Networks, Zero-shot Learning, Drug Discovery.

1 Introduction

DRUG Target Interaction (DTI) for drug discovery relies on a myriad of wet lab experiments on drug molecules and target proteins, a labor-intensive and time-consuming process [1], [2], [3], [4]. Thus, in-silico DTI prediction methods [47], [48] play an increasingly important role in pharmacy studies as the first step to quickly screen candidate molecules. Rule-based in-silico DTI methods like Docking and Pharmacophore Mapping [6], [7], [8] mimic the docking process with chemical structures or energy analysis. A key to them is to identify the core sub-structures for proteins and ligands (i.e., pockets and pharmacophores) respectively, using chemical rules and similarities among samples. But, since such rules are defined by humans with prior knowledge, these methods are often inflexible and computationally expensive [1], [2], [3], [4]. In contrast, data-driven machine learning methods can flexibly learn interaction rules by themselves and provide a significant acceleration for drug discovery.

Data-driven methods often model the DTI prediction as a binary classification problem given drug-target pairs [33], [35], [38]. Previous studies mainly focus on how to effectively learn representations of molecules and proteins, and input such representations into advanced classification models or ranking models [27], [28], [29], [34]. These representation learning methods can be categorized into a one-dimensional (1D) perspective and a three-dimensional (3D) perspective. From the 1D perspective, known methods use the molecule fingerprint [49], simplified molecular input line entry system (SMILES) [50], and the FASTA sequence of the target protein as the input features [36], [37]). From the 3D perspective, some recent studies leverage the geometric deep learning (DL) models to embed the 3D structures of the target proteins and potential candidate molecules [5], [10], [11], [23], [24], [25], [30], [38].

However, the predictions of these DL methods implicitly or explicitly exploit similar patterns or structures between the target protein and its homologous proteins in the training set. Hence traditional DL models usually require a large amount of data to train models to extract such knowledge. Yet, when given a newly discovered target protein, it may be hard to find homologous proteins with similar structures and there may be few known interactions of molecules. In this scenario, it is difficult for these data-driven methods to work well as they rely on verified knowledge. Consequently, prediction performances of these methods may become inferior and unsatisfactory. To our best knowledge,

- *Yuyang Xu and Jintai Chen are with the College of Computer Science and Technology and School of Public Health and Eye Center of the Second Affiliated Hospital, Zhejiang University, Hangzhou 310058, China. They are also with Institute of Wenzhou, Zhejiang University, Wenzhou 325036, China. E-mail: xuyuyang@zju.edu.cn; jtchen721@gmail.com.*
- *Jingbo Zhou is with the Baidu Research, Beijing 100193, China. E-mail: zhoujingbo@baidu.com.*
- *Haochao Ying is with the Department of Big Data in Health Science of the School of Public Health, and the Department of Breast Surgery and Oncology of the Second Affiliated Hospital, Zhejiang University School of Medicine, Hangzhou 310058, China. E-mail: haochaoying@zju.edu.cn.*
- *Wei Chen is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China. E-mail: chenwei@cad.zju.edu.cn.*
- *Danny Z. Chen is with the Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556, USA. E-mail: dchen@nd.edu.*
- *Jian Wu is with the School of Public Health, Second Affiliated Hospital School of Medicine and Institute of Wenzhou, Zhejiang University, Hangzhou 310058, China. E-mail: wujian2000@zju.edu.cn.*
- *Corresponding Author: Haochao Ying.*

Manuscript received April 19, 2005; revised August 26, 2015.

so far there is no known study on DTI prediction in such zero-shot learning scenarios.

To this end, we propose a novel Protein-Context enhanced Master Slave (PCMS) Framework for zero-shot drug-target interaction prediction. PCMS mainly consists of two major components: the Master Learner and the Slave Learner. The general idea is that we introduce a mechanism to use the Master Learner to generate the corresponding model parameters of the Slave Learner for conducting zero-shot DTI classification. The Master Learner takes a protein as input to learn a comprehensive protein representation, from the data perspective of hierarchical protein graph: residue FASTA sequence, and multiple sequence alignment (MSA). Meanwhile, the Slave Learner takes a drug (molecule graph) as input to perform molecule representation learning and build a classification model for DTI prediction.

Specifically, for Master Learner, we use an adaptive hierarchical GNN to model the protein context information. We construct an atom-residue heterogeneous graph for incorporating the protein level, residue level, and atom level information. Aiming to effectively utilize the protein level information, we additionally apply residue partial alignment with the aligned FASTA sequence between proteins during training. In this way, the Master Learner is able to make use of the protein structures and incorporate prior bioinformatics knowledge.

For both Master Learner and Slave Learner, we first embed the atoms of proteins or molecules with GNN, and use a graph information bottleneck approach to identify their key subgraphs (i.e., the pocket of the protein and the pharmacophore of the molecule). After identifying the pocket from the protein, Master Learner leverages the extracted protein context information (i.e. the learned pocket embedding) to generate the corresponding parameters of modules in the Slave Learner for DTI task fine-tuning. We design a meta-training method to optimize the overall framework. From the bioinformatics perspective, we mimic the process of finding the potential pharmacophore using the identified pocket structure of the target protein, thus approximating the overall interaction with low required resources (i.e., labeled data and training cost).

To summarize, our main contributions are as follows:

- We are among the first to study zero-shot DTI prediction with newly discovered pockets but only few known interactions and homologous proteins.
- We propose a novel and effective zero-shot DTI prediction framework, PCMS, based on a parameter generation/fine-tuning method.
- We conduct a suite of extensive experiments and dissections on two benchmark datasets, demonstrating the superiority of our PCMS framework over its competitors in various metrics.

2 Related Work

2.1 Data-driven Methods for DTI Prediction.

Existing methods such as machine learning and deep learning [33], [35], [38] usually treat drug-target pairs as input samples to make predictions, in which feature representation and prediction model are two decisive factors of performance. Align with the prior knowledge of chemistry and bioinformatics, several DTI methods [33], [34], [35] utilized 1D sequences (*i.e.,* SMILES and FASTA) embedded by NLP methods to initialize features, but naturally lack important 3D structure information. To alleviate it, some studies utilize CNN [23], [24], [25] to model the 3D images of protein and molecule voxel structures. Otherwise, the development of GNN methods [10], [30] made it possible to embed topological structures. Existing work on DTI prediction and a similar task of binding affinity prediction [9], [34], [38], [52] also utilized GNNs to embed chemical structures of molecules and target proteins. However, considerably large irrelevant chemical structures of proteins for DTI prediction make the learning of GNN models difficult.

After the feature representation, existing methods [36], [38] usually simply concatenated the features of molecules and target proteins as input to the predication models (e.g., random forest, support vector machine (SVM), and logistic regression (LR)). Also, several methods modeled DTI prediction as a recommendation problem [27], [28], [29], [54], [55]. However, these methods rely on the similarity between the whole target protein structures by MSA and so on [53]. When it comes to dealing with a newly discovered target protein with few known interacting drugs, it is hard for them to learn an effective generative pattern with so little data. Thus, we formulate a meta-learning problem for this DTI prediction scenario with little known DTI data.

2.2 Existing Works of Meta-Learning.

Meta-Learning, also known as few-shot learning, typically has two types of methods: gradient-based methods and metric-based methods. The former uses a meta learner to instruct the way for a basic learner to update, in order to learn a median initialization of parameters between all the tasks. For example, Finn *et al.* [21] proposed MAML, which sums up losses of all the seen tasks to balance the initialized parameters. For metric-based methods, Snell *et al.* [18] proposed a Prototype Network, which learns a prototype representation in the metric feature space for each class; during testing, the network computes the distances between the test samples and all the seen classes to make classification. Similarly, Vinyals *et al.* [19] proposed a Matching Network by matching samples of the test set with the class of few-shot labeled samples in the training set. Sung *et al.* [20] proposed a Relation Network to learn distances in the metric space as relation scores of different sample classes.

Zero-shot learning is a special case of meta-learning, in which a model utilizes task-level information to fast adapt to unseen classes. There are also works evaluating traditional Out-of-Domain methods of drug discovery problem [56]. However, applying zero-shot learning to DTI prediction is still a less-explored topic, given that newly discovered proteins may rarely have known homologous proteins and drug-target interactions. There is a similar concept used by the master-slave regularized model [26], but its objective is to directly predict parameters of logistic regression models for company revenue prediction, whose methodology and application domain are both substantially different from ours. In this paper, we propose a new zero-shot framework with parameter generation for DTI prediction problem, which even does not require support set for test tasks.

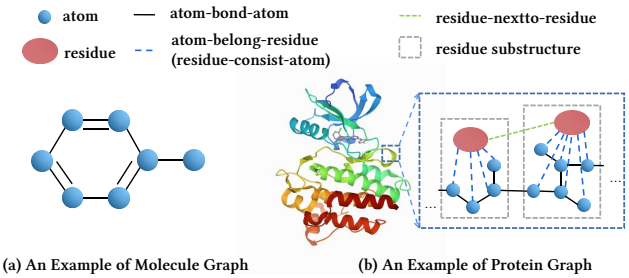


Fig. 1. Examples of a Molecule Graph and a Protein Graph.

3 PRELIMINARIES

In this section, we introduce the basic concepts and notations used throughout the paper. A summary of all the notation used can be found in Table 1 in Appendix.

As discussed in the Introduction section, existing methods implicitly or explicitly require homologous proteins for determining the similarity between the whole target protein structure. In this paper, we model DTI prediction as a meta-learning problem and propose a novel zero-shot learning framework to address the challenging situation with few known interactions and homologous proteins for a newly discovered target protein. Meta-learning utilizes similar source tasks as the training set to transfer knowledge for adaptation to unseen tasks (a.k.a., zero-shot) or tasks with few samples (a.k.a., few-shot). We begin with a formal definition of the target problem.

Definition 1 (DTI Task). We define a DTI task t_p from task set T for meta-learning as $t_p = (p, M_p)$, where p is a target protein from the target protein set P and M_p is the molecule set of its corresponding active and decoy molecules. An active molecule is presumed to be active with a target protein p (i.e., likely binding to the target protein), and a decoy is presumed to be inactive. Specifically, we define each member of M_p as (m, y), where m is one of the molecules in the active set or the decoy set corresponding to the target protein p and y is the label of the molecule m ($y = 1$ means the molecule m is an active one and $y = 0$ means m is a decoy).

For meta-learning tasks, samples of a task are usually further divided into the support set (for simulating few-shot training) and the query set (for measuring errors after training). We denote the support (molecule) set of the task as M_s and the query set of the task as M_q, i.e., $M_p = M_s \cup M_q$.

Definition 2 (Molecule Graph). We define a Molecule Graph as $G_m = \{V_m, E_m\}$, where V_m is the node set representing the atoms of a molecule and E_m is the edge set of bonds between the atoms. We denote the embedding of the i-th node as x_m^i ($x_m^i \in \mathbb{R}^{d_m}$, where d_m is the dimension of the node embeddings). Fig. 1(a) gives an example of a Molecule Graph. Though there are special chemical bonds like double bonds of benzene ring presented in Fig. 1(a), we take them as normal chemical bond edges with corresponding edge attributes in Molecule Graph.

Definition 3 (Pharmacophore). We define the "Pharmacophore" as the common key substructure of the molecules that potentially interact with the identified pocket structure of the protein, which is not exactly the same as the pharmacophore defined in Pharmaceutical Science. We aim to identify the similar key sub-structure that appears repeatedly in the active set of a specific target protein, which means the "pharmacophore" is essential for the DTI prediction of the given target protein.

Definition 4 (Protein Graph). We define a Protein Graph as $G_p = \{V_p, V_r, E_p, R_p\}$. Different from Molecule Graphs, a Protein Graph is defined as a heterogeneous graph in which V_p is the node set of atoms, V_r is the node set of amino acid residues, E_p is the set of edges between these two types of nodes and between nodes of each type, and R_p gives the relation type of each edge. The relations in R_p are divided into three types: $atom\text{-}bond\text{-}atom$, $residue\text{-}nextto\text{-}residue$, and $atom\text{-}belong\text{-}residue$ (or $residue\text{-}consist\text{-}atom$, since a protein graph is an undirected graph). We denote the embedding of the i-th protein atom as x_p^i ($x_p^i \in \mathbb{R}^{d_p}$) and the embedding of the o-th residue as x_r^o ($x_r^o \in \mathbb{R}^{d_r}$), where d_p and d_r are the dimensions of the protein atom embeddings and protein residue embeddings, respectively. Fig. 1(b) gives an example of a Protein Graph.

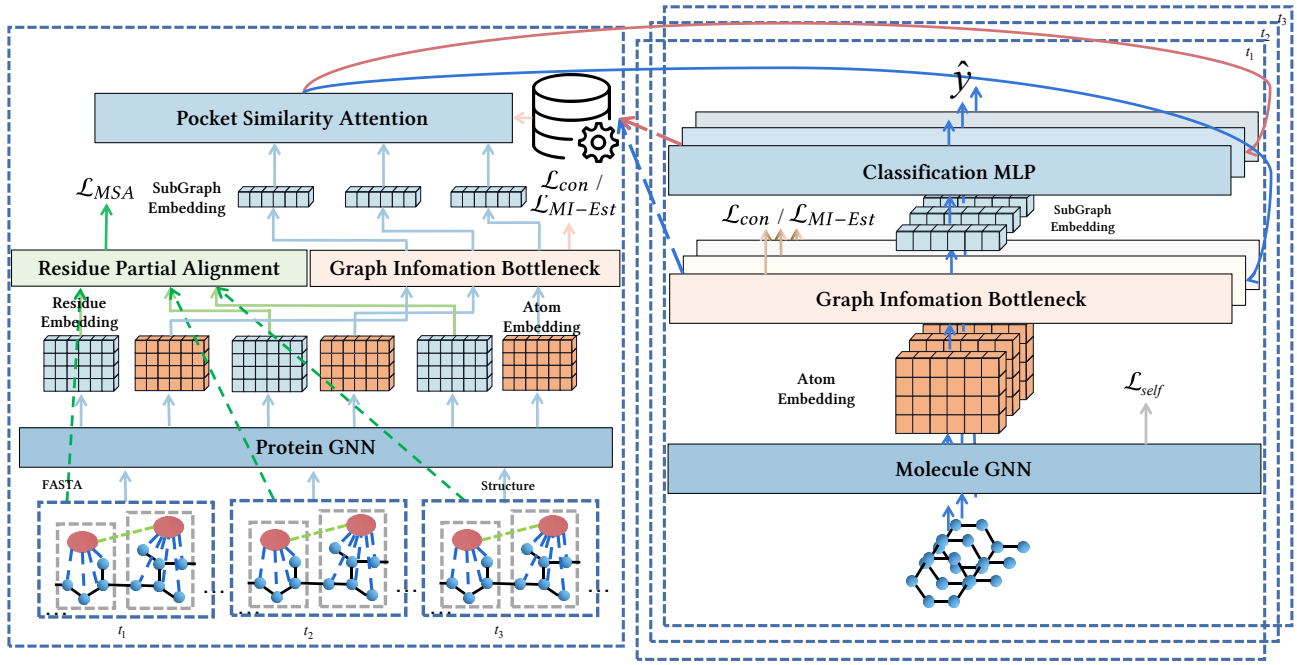
Problem 1 (Zero-shot Drug-Target Interaction Prediction). Given a DTI task t_p as defined above, DTI prediction problem can be formulated as learning a binary classification model F: $F(G_m, \Theta \mid G_p) = \hat{y}$, where Θ is the parameter of the model F, G_m ($m \in M_p$) is a molecule graph, G_p is the protein graph constructed with a target protein p, and $\hat{y}$ is the predicted DTI result. We denote $y \in \{0, 1\}$ as the ground truth (GT) label where 0 stands for decoy and 1 stands for active. For Zero-shot DTI Prediction, we aim to find an adaptive parameter Θ which can perform well on the potential molecule set of the target protein p without any training samples (M_p, y_p). To achieve this goal, we leverage the protein graph G_p as task-level information.

4 THE PCMS FRAMEWORK

4.1 Framework Overview

Fig. 2 shows the architecture of PCMS framework, which consists of a Master Learner and a Slave Learner. In a nutshell, PCMS uses Master Learner to generate parameters of some modules given a protein for fast adapting the Slave Learner and then utilizes Slave Learner for DTI prediction.

Specifically, for Slave Learner, the atom nodes of the input molecule are embedded by the Molecule GNN, and the resulting embeddings are sent to the Graph Information Bottleneck (GIB) [12] Module for key subgraph identification. Then, the embedding of the subgraph is sent to a Classification Multilayer Perceptron (MLP) layer for DTI prediction. Likewise, for Master Learner, a heterogeneous Protein Graph is fed to a hierarchical Protein GNN and the GIB module to find the key subgraph for the protein. Note that the hierarchical Protein GNN is a GNN model that embeds the residues and atoms with a heterogeneous graph derived from the target protein p. Finally, we generate the task-specific parameters utilizing only the information of the target protein p. The generated parameters are loaded into the corresponding modules of the Slave Learner and is used for the DTI prediction of the test set tasks.



(a) Master Learner (b) Slave Learner

Fig. 2. An overview of our Protein-Context enhanced Master/Slave (PCMS) framework. (a) Master Learner: learns the Atom embedding for key subgraph (pocket of the target protein) identification, in order to generate the parameters of modules in the Slave Learner. The residue embedding is generated for the Residue Partial Alignment task only for training. (b) Slave Learner: learns the Atom Embedding of the molecule for key subgraph (pharmacophore of the molecule) identification in order to make DTI prediction. The fine-tuned parameters are stored as the anchor parameters for the Master Learner during training. The generated parameters from the Master Learner are loaded by corresponding modules for inference.

More specifically, for meta-learning procedure, we can generate the parameters of Slave Learner after comparing the embeddings of the key protein subgraph with the stored subgraph embeddings of proteins in the training tasks. During training, we first use MAML [21] to train the Slave Learner on training tasks. For each training task, we store the parameters specifically fine-tuned on that task. Then we feed the proteins of these tasks to the Master Learner to model the relation between the similarities of the key substructures of their target protein p. Each generated parameter for the Slave Learner is an attentive weighted sum of the corresponding stored parameters of the proteins in the training tasks. The details of each module and the optimization process are presented in the following subsections.

4.2 Slave Learner

We build the architecture of Slave Learner inspired by Meta-MGNN [31]. Given a target protein p, the DTI prediction problem for each task is similar to the process of binary drug property classification problem. Meta-MGNN uses MAML [21] to train a GNN for drug-property prediction with several self-supervised tasks including graph reconstruction and atom type prediction with atom embeddings.

4.2.1 Model Structure

In Slave Learner, the Molecule GNN is a GNN model that embeds the nodes (i.e., the atoms of the molecule) with a graph derived from the molecule's 3D structure. For implementation, any GNN can be used as the basis GNN model for Molecule GNN. In our experiments, we adopt SchNet [10] since it is simple and can model 3D information of the molecule graph. The GIB is a graph assignment layer for finding the most informative subgraph. Similar to Meta-MGNN, first we embed a one-hot vector of atom chemical features into a feature space. Then we design two self-supervised task losses of node type prediction and link (chemical bond) prediction for learning better embeddings.

We first denote the initialization of atom nodes as $X_m^0 \in \mathbb{R}^{n_m \times d_m}$, where n_m is the number of the atoms in a molecule. Thus, the output of Molecule GNN (denoted as GNN_m) can be represented as:

$$X_m = GNN_m(X_m^0, E_m), \tag{1}$$

where $X_m = [x_m^1, x_m^2, \ldots, x_m^{n_m}]$ and E_m is edge set of G_m.

Specially, following the pre-training strategy for GNN [32], we design two self-supervised task losses for learning better atom node embeddings of molecules. First, we seek to recover the bonds between atoms based on the learned embeddings, whose reconstruction loss is defined as:

$$\hat{e}_{ab} = \text{Sigmoid}(x_m^a, x_m^b), \tag{2}$$

$$\mathcal{L}_{recon} = (1-e_{ab})\hat{e}_{ab}\log(1-\hat{e}_{ab}) + e_{ab}(1-\hat{e}_{ab})\log(\hat{e}_{ab}), \tag{3}$$

where $\hat{e}_{ab}$ denotes the predicted probability of whether a bond exists between atom a and atom b, and e_{ab} is the corresponding ground truth. Here we do not take all edges into account but apply the negative sampling to them, as there are many more negative edges than positive ones in a real G_m. Second, due to the over-smoothing problem of neighboring nodes in deep GNN models, we seek to maintain the node differences by predicting the atom type based on the output embedding x_m^a:

$$\hat{v}_a = MLP(x_m^a), \tag{4}$$

$$\mathcal{L}_{node} = (1-v_a)\hat{v}_a\log(1-\hat{v}_a) + v_a(1-\hat{v}_a)\log(\hat{v}_a). \tag{5}$$

Here, we use an MLP layer to predict the atom node type $\hat{v}_a$, and take the Cross-Entropy loss as the loss function of the GT v_a. Thus, the overall self-supervised loss can be computed as the following equation:

$$\mathcal{L}_{self} = \mathcal{L}_{recon} + \mathcal{L}_{node}. \quad (6)$$

After learning the embeddings of the atoms by the Molecule GNN, we apply a module to identify a key subgraph of the molecule as the Graph Information Bottleneck (GIB). Specifically, the GIB module aims to find the subgraph with the maximum compression of the original graph and the maximum correlation with the target protein p. In the DTI prediction situation, the interactions of a target protein and the molecule depend on the integrating degree of the pocket (of the target protein) and pharmacophore (of the molecule). From this perspective, the pocket and pharmacophore can be considered as the GIB of the protein graph G_p and the molecule graph G_m, respectively. Similarly, the objective of the GIB module on the protein graph is to find the pocket. The subsequent DTI prediction should be much more efficient by matching these important GIB sub-structures. Besides, the GIB will also provide good interpretability by identifying the potential pharmacophore of the molecule and the pocket of the target protein. In this way, the result of the prediction can be explained by bioinformatics mechanism.

Formally, the Graph Information Bottleneck problem can be formulated by the following equation:

$$\max_{G^{sub}}(I(y, G^{sub}) - \beta \cdot I(G, G^{sub})), \quad (7)$$

where $I()$ represents the mutual information of the two input embeddings, β is a hyper-parameter for loss adjustment, y is the ground truth label of the DTI prediction, and G^{sub} represents subgraph embeddings of the graph with embedding G. In Eq. (7), $I(y, G^{sub})$ can be computed as one minus the classification loss by subgraph embedding G^{sub} (i.e., $1-$ classification loss). But, it is difficult to find a tractable upper bound for $I(G, G^{sub})$ since there are an exponential number of subgraphs in a graph. Thus, it is hard to estimate the distribution of $I(G, G^{sub})$. Yu *et al.* [12] introduced the Donsker-Varadhan representation [13] of the KL-divergence, estimated $I(G, G^{sub})$ using a statistics network, and utilized a graph assignment module to propose potential subgraphs. Following their work, we apply a graph assignment MLP layer to assign the atom nodes of the proposed potential key subgraphs, as:

$$Assign = MLP_{assign}(X_m), \quad (8)$$

$$G^{sub} = Assign[0] \cdot X_m, \quad (9)$$

$$G_m = (Assign[0] \cdot X_m + Assign[1] \cdot X_m)/2, \quad (10)$$

where $Assign \in \mathbb{R}^{n_m \times 2}$ is a probability matrix in which each entry indicates whether an atom node belongs to the key subgraph, and $Assign[j]$ is the j-th column ($j \in 0, 1$) of $Assign$ as j equals 0 and 1 in Eq.9 and Eq.10, respectively. Thus, G^{sub} can be seen as the weighted average pool of the assigned subgraph node embeddings and is a single-dimension vector used for the final prediction. We let Θ^{assign} denote the parameters of the MLP_{assign}. Since pharmacophore is a connected substructure of a molecule, we apply a connected loss as follows:

$$\mathcal{L}_{con} = \parallel Norm(Assign^T \cdot A_m \cdot Assign) - I_2 \parallel_F, \quad (11)$$

where $\parallel \cdot \parallel_F$ is the Frobenius norm, Norm() is row-wise normalization, A_m is the adjacent matrix of the Molecule Graph, and I_2 is a 2×2 identity matrix. For $I(y, G^{sub})$, Yu *et al.* [12] proposed an Adversarial Learning method where in the inner loop, a discriminator $Disc$ distinguishes the matched graph and its subgraph, while in the outer loop, the GIB module mixes up the embeddings:

$$\mathcal{L}_{MI-Est} = \sum Disc(G, G^{sub}) - \log \sum e^{Disc(G, \hat{G}^{sub})}, \quad (12)$$

where $\hat{G}^{sub}$ is the unmatched subgraph embedding of graph with embedding G. Again, we apply negative sampling by keeping the number of unmatched embedding pairs and the number of matched embedding pairs the same.

Since the interactions between molecules and the target protein p depend on the integrating degree of the pharmacophore and the pocket regardless of the rest of the structure, we feed the molecule subgraph embeddings to the classification MLP layer. In this way, after obtaining subgraph embedding G^{sub}, the final DTI prediction can be computed as the following function:

$$\hat{y} = MLP_{pred}(G^{sub}), \quad (13)$$

$$\mathcal{L}_{cls} = (1-y)\hat{y}\log(1-\hat{y}) + y(1-\hat{y})\log\hat{y}. \quad (14)$$

Here we denote the parameters of MLP_{pred} as Θ^{pred}. The loss of Slave Learner can be defined as:

$$\mathcal{L}_{slave} = \mathcal{L}_{cls} + \lambda_1(\mathcal{L}_{self} + \mathcal{L}_{con} + \beta'\mathcal{L}_{MI-Est}), \quad (15)$$

where λ_1 is a loss weight for tuning the importance of these self-supervised tasks for better embeddings, β' is a hyper-parameter for adjusting the Graph Information Bottleneck weight, and β in Eq. (7) is equal to $\beta' \times \lambda_1$. $\mathcal{L}_{con}$ is the connectivity loss for regularizing the identified subgraph to be connected and $\mathcal{L}_{MI-Est}$ is the mutual information loss estimated by the discriminator.

4.2.2 Meta Training of the Slave Learner

We present the pseudo-code for meta-training of Slaver Learner in Algorithm 1 (lines 3-6). During the meta-training, we first use MAML [21] to train the Slave Learner to obtain a medium parameter for all the DIT tasks. Then we fine-tune MLP_{assign} and MLP_{pred} on query set M_q of each task t_q while fixing the parameters of the Molecule GNN. We store the parameters Θ_p^{assign} and Θ_p^{pred} as anchor parameters and GT parameters. Note that during the inference phase (instead of the meta training phase), the values of Θ^{pred} will be determined by the Master Learner which takes the subgraph embedding of the protein as input. More details of the parameter generation will be presented in Section 4.3.3.

4.3 Master Learner

As shown in Fig. 2, the architecture of the Master Learner is partially the same as the Slave Learner, which mainly consists of three parts. The first part is a hierarchical Protein GNN along with a Residue Partial Alignment for the embedding of the target protein p. The second part is a Graph Information Bottleneck module with a graph assignment

layer for the important subgraph (i.e., pocket) identification. Since the Graph Information Bottleneck module here is the same as the GIB module in the Slave Learner, we will not repeat the description of it again in this subsection. The last part is a pocket similarity-based parameter generation module which generates parameters for the DTI task on the test set. In general, we adopt a parameter generation method based on the similarity between the key sub-structures of the protein, which is modeled by a hierarchical Protein GNN, to deal with zero-shot DTI prediction.

4.3.1 Hierarchical Protein GNN

Given a heterogeneous Protein Graph G_p, there are two types of nodes: atom nodes and residue nodes. We initialize the embeddings of atom nodes with a one-hot vector of atom chemical features, and initialize the embeddings of residue nodes with ProtTrans [14] (a pre-trained FASTA sequence embedding model). We summarize the protein structure into three main relations: 1) Atom-bond-Atom, which is the basic structure of all chemical molecules; 2) Residue-nextto-Residue, which is the basic structure of a residue sequence of a protein; 3) Atom-belong-residue (Residue-consist-Atom), which shows that a residue is made up of several atoms. For each layer in the hierarchical Protein GNN, we apply different message-passing schemes for these three different edge relations. We then sum up all the messages as aggregation and update.

Atom-bond-atom: Like the Slave Learner, we adopt a GNN to generate messages from atom nodes to atom nodes. We denote the GNN model as GNN_p. The atom-atom message of the l-th layer, $M^l_{atom-atom}$, can be calculated by:

$$M^l_{atom-atom} = GNN_p(x^l_p, E_p, R_p). \tag{16}$$

For the atom level, the GNN_p can be replaced with any Graph Neural Networks. Specially, as 3D positional information are essential for atom embedding of molecules, in our experiment, we use SchNet [10] to obtain the 3D topological information of the as they are connected by chemical bonds in 3D dimensions.

Residue-nextto-residue: For the residue sequence level, the residues of the protein are connected by the peptide bond with alpha carbon of the residue. Thus, the residues form a sequential chain as the main structure of the protein. Transformer [43] is powerful to model the sequential data by considering the former and latter residues with global self-attention. Thus, we use the self-attention mechanism of the Transformer-Conv [15] as the message-passing method to pass the information from the neighboring residue nodes and model the residue sequence level information. The residue-residue message of the l-th layer is calculated by the functions below:

$$\alpha_{a,b} = Softmax(\frac{(W_{trans1} \cdot x^{a,l}_p)^T \cdot (W_{trans2} \cdot x^{b,l}_p)}{\sqrt{d_p}}), \tag{17}$$

$$M^{a,l}_{residue-residue} = W_{trans3} \cdot x^{a,l}_p + \sum_{b \in N(a)} (\alpha_{a,b} \cdot W_{trans4} \cdot x^{b,l}_p), \tag{18}$$

where W_{trans1}, W_{trans2}, W_{trans3}, and W_{trans4} are the weights of the TransformerConv method, and $N(a)$ denotes the set of neighboring nodes of a node a in G_p.

Atom-belong-residue (residue-consist-atom): Different from the residue sequence, the residue as well as the corresponding atoms form a star-shape graph to formalize the affiliation relationship. However, the importance of the atoms consisting of the residue varies from each other (e.g. the hydrogen atom is relatively less important compared with the carbon atom in the residue). In order to model the importance of the consisting atoms, we use GAT to model the atom-residue relationship as GAT [16] naturally models the importance attention of the neighbors. The atom-residue message of the l-th layer is computed as:

$$\gamma = Softmax(LeakyRelu(\omega^T \cdot [\theta \cdot x^a \parallel \theta \cdot x^b])),\ b \in N(a), \tag{19}$$

$$M_{atom-residue} = \gamma \cdot \theta x^b,\ b \in N(a), \tag{20}$$

where θ and ω are weights for Graph Attention, and x^v is the learned embedding of a node v in G_p.

For aggregation and update, we sum up all messages to update the node embeddings. Using the heterogeneous Protein GNN with the message-passing schemes above, we can effectively extract information from the atom and residue levels in a hierarchical perspective.

4.3.2 Residue Partial Alignment

We further propose a Residue Partial Alignment [17] module to align embeddings of the residue nodes for information extraction in the training phase for better protein embedding. The same residue fragments in different proteins often maintain similar properties to some extent. Thus, we align the residue nodes of the same type in different protein graphs to incorporate more bioinformatics knowledge. Note that usually, this alignment is done by Multiple Sequence Alignment (MSA). But, in our application scenario, MSA cannot be applied since there are few homologous proteins in the inference (a.k.a. test) process. In this way, we only incorporate this extra MSA task during training phase and do not envolve the result of MSA during inference phase.

Here we assume that the residues to be aligned by the MSA process have the same embeddings. We treat the result of Sequence Alignment as an Alignment Graph $G_s = \{V_r, E_s\}$, in which the aligned residues in V_r are connected by edges in the edge set E_s. Given the output residue embeddings X_r, we construct the Alignment Graph to align the embeddings of the aligned residues, as:

$$\bar{e}_{ab} = \mathrm{Sigmoid}(x^a_r, x^b_r), \tag{21}$$

$$\mathcal{L}_{MSA} = (1 - \check{e}_{ab})\bar{e}_{ab} \log(1 - \bar{e}_{ab}) + \check{e}_{ab}(1 - \bar{e}_{ab}) \log(\bar{e}_{ab}), \tag{22}$$

where $\bar{e}_{ab}$ is the predicted probability of whether the representations of residues a and b are aligned, and $\check{e}_{ab}$ is the GT label. Here we apply the negative sampling to balance the numbers of positive samples and negative samples. In this way, we can fully leverage the protein-level information.

4.3.3 Parameter Generation for Zero-shot Learning

The final step is to generate parameters for the Slave Learner based on the pocket similarity between the target protein p and the proteins in the training tasks. This is done by the pocket similarity-based Parameter Generation module.

During the meta training of the Slave Learner, we store the fine-tuned parameters of each task and the corresponding protein subgraph embeddings as the anchor parameters

and anchor subgraph embeddings. The anchor subgraph embeddings and anchor parameters act as the base vectors of the protein subgraph similarity feature space and the corresponding task-specific fine-tuned parameter feature space, respectively. Next, by computing the similarity between the subgraph embeddings of the target protein p and the proteins in the training tasks, we can map the position of the target parameter and thus generate the task-specific fine-tuned parameter of the test task. Finally, we load the generated parameters into the corresponding modules of the Slave Learner for DTI prediction.

Here we define the learned protein subgraph embeddings G_p^{sub} in training set as the anchor subgraph embeddings $G_{p_k}^{sub,anch}(k = 1, 2, \ldots)$. Meanwhile, we define the parameters Θ^{assign} of MLP_{assign} and parameters Θ^{pred} of MLP_{pred} of the Slave Learner fine-tuned by corresponding task molecules as anchor parameters $\Theta_{p_k}^{assign,anch}$ and $\Theta_{p_k}^{pred,anch}$, respectively. During training, we store the learned anchor subgraph embeddings $G_{p_k}^{sub,anch}$ and anchor parameters $\Theta_{p_k}^{assign,anch}$ and $\Theta_{p_k}^{pred,anch}$. The anchor subgraph embeddings and anchor parameters are taken as the base vectors of the target proteins' subgraph feature space and fine-tuned parameter feature space. Here we seek to map the subgraph feature space to the fine-tuned parameter feature space. By this means, for a subgraph embedding of a certain target protein, $G_{p_0}^{sub}$, we can measure the similarity of the subgraph of the protein p_0 with the anchor subgraphs of the proteins p_k, and thus use this similarity weight to derive the adaptive parameters for MLP_{assign} and MLP_{pred} in Slave Learner of the target task.

We denote the stored subgraph embeddings, parameters of MLP_{assign}, and parameters of MLP_{pred} as $G^{sub,anch}$, $\Theta^{assign,anch}$, and $\Theta^{pred,anch}$, respectively. When dealing with a new target protein task with the target protein graph G_{p_0}, we first calculate its subgraph embedding:

$$G_{p_0}^{sub} = GIB_p(GNN_p(G_{p_0})), \tag{23}$$

where GIB_p is the GIB module in the Master Learner and GNN_p is the Protein GNN. Then we apply attention mechanism to learn the similarity between the given pocket and the anchor pockets:

$$\alpha_{pocket} = Softmax([[G_{p_0}^{sub} \parallel G_{p_1}^{anch}], [G_{p_0}^{sub} \parallel G_{p_2}^{sub,anch}], \ldots]), \tag{24}$$

$$\Theta_{p_0}^{assign'} = \alpha_{pocket} \odot [\Theta_{p_1}^{assign,anch}, \Theta_{p_2}^{assign,anch}, \ldots], \tag{25}$$

$$\Theta_{p_0}^{pred'} = \alpha_{pocket} \odot [\Theta_{p_1}^{pred,anch}, \Theta_{p_2}^{pred,anch}, \ldots], \tag{26}$$

where $\Theta_{p_0}^{assign'}$ and $\Theta_{p_0}^{pred'}$ are generated parameters of MLP_{assign} and MLP_{pred} for task p_0, respectively. Suppose the GT parameters for MLP_{assign} and MLP_{pred} of task p are $\Theta_{p_0}^{assign}$ and $\Theta_{p_0}^{pred}$. Then the parameter generation loss $\mathcal{L}_{gen}$ can be defined as:

$$\mathcal{L}_{gen1} = \sum SmoothL1(\Theta_{p_0}^{assign}, \Theta_{p_0}^{assign'}), \tag{27}$$

$$\mathcal{L}_{gen2} = \sum SmoothL1(\Theta_{p_0}^{pred}, \Theta_{p_0}^{pred'}), \tag{28}$$

$$\mathcal{L}_{gen} = \mathcal{L}_{gen1} + \mathcal{L}_{gen2}, \tag{29}$$

where $SmoothL1$ is the loss function to measure the distance between the stored and generated parameters. We use the $SmoothL1$ loss here to avoid the gradient explosion problem of the Mean Square Error loss when the error is too large and to improve the converging performance of the Mean Absolute Error loss when the error is too small. Our loss for the Master Learner is defined as:

$$\mathcal{L}_{master} = \mathcal{L}_{gen} + \lambda_2(\mathcal{L}_{MSA} + \mathcal{L}_{con}^p + \beta' \mathcal{L}_{MI-Est}^p), \tag{30}$$

where $\mathcal{L}_{con}^p$ and $\mathcal{L}_{MI-Est}^p$ are the connected loss and MI-Est loss for GIB_p, and λ_2 is a weight for the extra losses.

4.4 Meta Training of the Master Learner

We present the pseudo-code for the meta-training of the Master Learner as in the following Algorithm 1.

Algorithm 1 Meta Training for PCMS

Input: A DTI task training set T_{train}, the Master Learner model (denoted as $ML(\cdot)$ for short), and the Slave Learner model (denoted as $SL(\cdot)$ for short).

while not done **do**
  Sample a batch of tasks, $T_{train}^{batch} \sim p(T_{train})$;
  $\Theta^{slave} = MAML(\mathcal{L}_{slave}(SL(T_{train}^{batch})))$;
  for $t_p \in T_{train}^{batch}$ **do**
    $\Theta_p^{assign}, \Theta_p^{pred} = Finetune(SL(t_p), MLP_{assign}, MLP_{pred})$;
  end for
  for $t_p \in T_{train}^{batch}$ **do**
    $\Theta_p^{assign'}, \Theta_p^{pred'}, \mathcal{L}_p^{master} = ML(t_p)$;
    load($\Theta_p^{assign'}, MLP_{assign}$);
    load($\Theta_p^{pred'}, MLP_{pred}$);
    $\mathcal{L}_p^{slave'} = SL(t_p)$;
  end for
  $\mathcal{L} = \sum_{t_p \in T_{train}} (\mathcal{L}_p^{master} + \mathcal{L}_p^{slave'})$
  Update($ML, \mathcal{L}$);
end while

During training the Master Learner, in each step, we take one of the training tasks as a sample, whose fine-tuned parameters are taken as the GT while the rest of the parameters as the anchor parameters, to compute $\mathcal{L}_{master}$. We also load the generated parameters to test on the molecules of the task taken in this step and update the sum of $\mathcal{L}_{master}$ and $\mathcal{L}_{slave}$ while fixing the parameters of the Slave Learner. In this way, we mimic the circumstances of the zero-shot DTI prediction, and use the prediction error as the training loss to update the model.

4.5 Inference

For testing, we directly feed G_p to the Master Learner to compute the similarity between the assigned subgraph and the anchor subgraph embeddings to generate parameters $\Theta_p^{assign'}$ and $\Theta_p^{pred'}$. Specifically, we store the subgraph embeddings and parameters for all the tasks in the training set as the anchor subgraph embeddings and anchor parameters to generate parameters for the test target protein p. We then load the generated parameters into the Slave Learner and test the DTI classification.

TABLE 1
Performance evaluation of our PCMS framework with the baselines on the DUD-E and DEKOIS2.0 datasets. The best results are marked in **bold**, and the second best results are underlined. The numbers outside and inside ( ) stand for the average and the standard deviation in a metric with 5-fold cross validation, respectively. Similarly hereinafter.

Data	Model	Zero-shot		5-shot	
		ACC	F1	ACC	F1
DUD-E	LR	0.4981 (0.0000)	0.3983 (0.0000)	0.4978 (0.0029)	0.3622 (0.0549)
	SVM	0.5019 (0.0000)	0.3111 (0.0000)	0.4957 (0.0074)	0.3067 (0.0494)
	LRF-DTI	0.5200 (0.0026)	0.0756 (0.0047)	0.5019 (0.0000)	0.0000 (0.0000)
	DeepConv-DTI	0.6026 (0.1095)	**0.6013 (0.0060)**	0.5812 (0.0153)	0.4808 (0.5633)
	GraphDTA	0.5886 (0.0091)	0.4078 (0.0092)	0.6994 (0.0178)	0.6185 (0.0220)
	IRM(GraphDTA)	0.5766 (0.0059)	0.4602 (0.0096)	0.5914 (0.0051)	0.5282 (0.0093)
	MolTrans	0.5113 (0.0022)	0.0448 (0.0068)	0.5097 (0.0026)	0.0484 (0.0117)
	Meta-MGNN	0.5703 (0.0043)	0.3270 (0.0083)	0.6714 (0.0158)	0.6580 (0.0206)
	PCMS	**0.6402 (0.0028)**	0.5283 (0.0070)	**0.7288 (0.0190)**	**0.7333 (0.0127)**
DEKOIS2.0	LR	0.5013 (0.0009)	0.2756 (0.0007)	0.5016 (0.0105)	0.2684 (0.0225)
	SVM	0.5421 (0.0006)	0.1876 (0.0006)	0.5426 (0.0067)	0.1871 (0.0141)
	LRF-DTI	0.6453 (0.0025)	0.0667 (0.0227)	0.6466 (0.0000)	0.0000 (0.0000)
	DeepConv-DTI	0.5575 (0.0072)	0.4306 (0.0046)	0.5352 (0.0261)	0.3945 (0.0429)
	GraphDTA	0.5247 (0.0071)	0.4858 (0.0043)	0.6355 (0.0095)	0.5299 (0.0168)
	IRM(GraphDTA)	0.5862 (0.0021)	0.4128 (0.0042)	0.5977 (0.0040)	0.4660 (0.0057)
	MolTrans	0.5939 (0.0467)	0.2028 (0.1076)	0.6154 (0.0251)	0.2024 (0.1464)
	Meta-MGNN	0.5179 (0.0020)	0.4731 (0.0010)	0.6441 (0.0127)	0.5665 (0.0051)
	PCMS	**0.6834 (0.0043)**	**0.5399 (0.0028)**	**0.6778 (0.0067)**	**0.5715 (0.0027)**

5 EXPERIMENTS

5.1 Experimental Setting

5.1.1 Datasets

We evaluate baseline methods and our PCMS Framework on DUD-E[1] dataset [39] and DEKOIS2.0[2] dataset [44] with parameters trained on DUD-E. DUD-E is a benchmark dataset for docking with 102 protein targets and the corresponding actives and decoys. However, DUD-E was also reported as a biased dataset [45], [46], on which a model could achieve good performance with only actives and decoys and without the target structure. Thus, we also introduce the unbiased dataset DEKOIS2.0 to directly evaluate the models with parameters trained on the DUD-E dataset.

For DUD-E dataset, we download the PDB files (the general file type to store the 3D protein structure) from the PDB website[3] [40] based on the given PDB IDs in DUD-E dataset for experiments. We eventually obtain 81 DTI tasks. We eliminate the actives and decoys that cannot be correctly loaded with the RDKit [41] and BioPython [42] packages, and eliminate the whole DTI task whose protein cannot be correctly loaded by RDKit. Since the objective of meta-learning methods is to learn knowledge from tasks with numerous data and transfer it to similar tasks with fewer samples, we select the top 60% of DTI tasks with the most actives and decoys as the training set, one half of the remaining DTI tasks as the evaluation set, and the other half of the remaining DTI tasks as the test set.

For the DEKOIS2.0 dataset, we also download the PDB files based on the PDB IDs provided in the appendix table of the paper [44]. With a similar pre-processing procedure, we eventually obtain 57 DTI tasks. All these tasks are used as unbiased test set tasks to evaluate the performances of models trained on the DUD-E dataset.

To support the zero-shot adaption capability, we analyze the homology sequence similarity between the target proteins from the training set and testing set of DUD-E dataset as well as the proteins in training set of DUD-E dataset and the DEKOIS2.0 dataset . It proves that these two pairs of datasets vary from each other and thus the inference test on the testing set of DUD-E dataset and DEKOIS2.0 dataset with parameters trained from the training set from the DUD-E dataset can test the capability of zero-shot adaption of our proposed PCMS. The details of the introduction of the two datasets, the analyzing results and the metrics used to evaluate the baselines and model performance are shown in the Appendix.

For every step, we randomly select 5 active and 5 decoys as the support set, and select 64 actives and 64 decoys to form the query set. When given a newly discovered target protein, there could be few known actives or decoys with a large number of molecules with unknown interaction results. Thus, here we use more actives and decoys in the query set than in the the support set in order to mimic the real few-shot virtual screening situation during training.

5.1.2 Baselines

We compare our PCMS with several state-of-the-art DTI prediction baselines: Support Vector Machine (SVM), Logistic Regression (LR), and LRF-DTIs [37] as representatives of machine learning methods, DeepConv-DTI [35] for CNN methods, GraphDTA [38] for GNN methods, Meta-MGNN [31] for self-supervised Meta-Learning methods, and the latest MolTrans [33] for a more complex protein embedding method using a state-of-the-art universal method of Transformer [43]. Due to the recent studies of Out of Domain (OOD) [56] models on the drug discovery area, we also incorporate IRM [57] as a baseline method. The details of these baselines are elaborated in Appendix.

5.2 Performance Evaluation and Analysis

We first compare the performances of our PCMS with the above baseline methods. As shown in Table 1, our approach outperforms most of the baseline methods in ACC and F1 metrics in both zero-shot learning and 5-shot learning circumstances, showing the effectiveness of our proposed

1. http://dude.docking.org
2. http://www.pharmchem.uni-tuebingen.de/dekois/
3. https://www.rcsb.org/

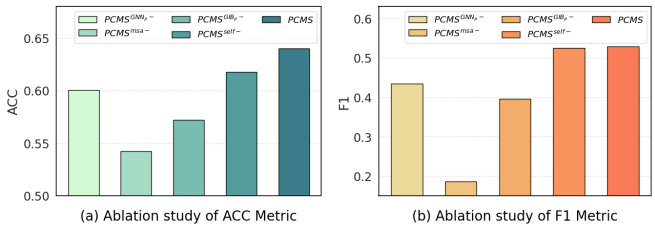
framework. Specifically, though DeepConv-DTI achieves the best F1 results for zero-shot learning on the DUD-E dataset (while pure PCMS still attains the second-best performance), it does not perform well on the DEKOIS2.0 dataset. This suggests that some methods may rely on the biased actives and decoys of the DUD-E dataset to achieve such performances, which can also be seen from the AUC metric results in Appendix. Yet PCMS can perform better on various external datasets with even zero-shot fine-tuning.

In particular, we observe that the performances of machine learning methods are unsatisfactory. A possible reason for this phenomenon is that these methods cannot distinguish active-target pairs and decoy-active pairs very well. This is because (1) compared with DL methods, simple machine learning methods cannot fit so few drug-target pairs of known interactions; (2) molecule fingerprint and residue embeddings omit complex information of the protein and molecule structures, which may significantly impair the information used. Among the machine learning methods, LRF-DTI utilizes the Lasso regularization for the complex structures without much irrelevant information, and thus its performances are better than the other such baselines.

Alternatively, DL methods use different ways to incorporate more structure information of the target protein and the molecules, and thus yield better performances. Specifically, DeepConv-DTI uses a CNN to focus on local information of the neighboring residues, and MolTrans applies a Natural Language Processing (NLP) based algorithm (i.e., FCS) to find the embeddings of frequent sub-structure combinations. These methods, as well as our proposed PCMS, all pay attention to the sub-structures of the proteins or molecules to refine utilized information, which is effective for DTI prediction. However, in the meta-learning scenario with few homologous proteins and known interactions, MolTrans performs not so well since there are too few proteins and molecule structures available. Hence, frequently appearing combinations of sub-structures may not be the key substructures providing effect in DTI prediction.

On the other hand, GraphDTA and Meta-MGNN leverage a GNN to model the 3D topological structures of the molecule and the target protein. In particular, the drug-property-based few-shot prediction method, Meta-MGNN, performs a little better since it takes few-shot learning into account. However, it omits information from the target protein when constructing the DTI task, and hence does not perform as well as our proposed PCMS. Specially, with the help of IRM, we can see that the GraphDTA method performs better with such few data samples on both two datasets. Still, IRM (GraphDTA) does not perform as well as our proposed method. We believe that IRM (GraphDTA) still requires a large number of samples to model the overall distribution of the protein domain. On the other hand, our proposed PCMS digs into the essence of the DTI process and fits the low-resource situation better, while the existing methods like CNN in DeepConv-DTI somehow look for pattern similarity between the residues, which potentially utilizes homologous proteins for embeddings and lead to worse performance in such circumstance.

In summary, PCMS improves 5.9% in ACC and 11.1% in F1 score with zero-shot on the DEKOIS2.0 dataset compared with the best baselines. Specifically, PCMS framework



(a) Ablation study of ACC Metric (b) Ablation study of F1 Metric

Fig. 3. Performance evaluation of ablation study with zero-shot learning on the DUD-E dataset.

utilizes a hierarchical Protein GNN and a Molecule GNN to gather as much information of the structure as possible. We also apply a GIB module for key subgraph identification based on Information Bottleneck Theory rather than the frequency. Besides, by modeling the similarity between the pocket structures, our PCMS achieves better performances with little homologous protein information.

5.3 Ablation Study

To examine the effects of the key components of our framework, we conduct ablation study on the DUD-E dataset by omitting each key component from our model. As DEKOIS2.0 dataset act only as a test dataset, we do not apply ablation study on it. $PCMS^{GNN_p-}$ stands for PCMS without Protein GNN while the sum of residue embeddings from ProtTrans still takes place; $PCMS^{msa-}$ stands for the PCMS Framework without the residue partial alignment; $PCMS^{GIB_p-}$ stands for the PCMS Framework without Graph Information Bottleneck for the protein; $PCMS^{self-}$ is for the performance of the PCMS Framework without the self-supervised module of molecules. Since the Master Learner works only for zero-shot learning and does not take any effect in the few-shot fine-tuning (which relies only on the molecules), we do not incorporate 5-shot experiments in this ablation study. The results are shown in Fig. 3.

For $PCMS^{msa-}$, without the residue alignment task, the Master Learner cannot perform even as well as $PCMS^{GNN_p-}$, which confirms that the target protein is too large and too complex for GNN to model. Therefore, providing more supplementary information (such as the 3D structures of atoms and MSA relations between residues) can compensate for this shortcoming for protein embedding, which is what PCMS does. Similarly, comparing $PCMS^{GIB_p-}$ with PCMS, the embedding of the whole protein is not as effective as the key subgraph embedding, which shows the effectiveness of GIB as well. Since $PCMS^{GNN_p-}$ utilizes residue features instead of the protein structure, it performs worse than PCMS as it uses only the FASTA information rather than the 3D topological structure. Lastly, for $PCMS^{self-}$, we can infer that the self-supervised module of the molecule embedding does give some enhancement, but this is not the major reason why PCMS can achieve its good performance. Consequently, these designed modules are all needed parts of our proposed PCMS, contributing to its final performance.

5.4 Robustness Study

We then investigate the robustness of our PCMS on the DUD-E dataset by varying the sizes of the support set and query set. Specifically, we run PCMS with groups of the

TABLE 2
Performances with different hyper-parameter values (the sizes of the support (spt) sets and query (qry) sets) on the DUD-E dataset.

Model	Zero-shot		5-shot	
	ACC	F1	ACC	F1
PCMS (3spt/32qry)	0.5222 (0.0027)	0.1036 (0.0056)	0.6843 (0.0097)	0.6750 (0.0220)
PCMS (3spt/64qry)	0.5782 (0.0073)	0.3641 (0.0169)	0.7226 (0.0121)	0.7374 (0.0245)
PCMS (5spt/32qry)	0.6025 (0.0028)	0.4405 (0.0044)	**0.7400 (0.0086)**	**0.7501 (0.0035)**
PCMS (5spt/64qry)	**0.6402 (0.0028)**	**0.5283 (0.0070)**	0.7288 (0.0190)	0.7333 (0.0127)

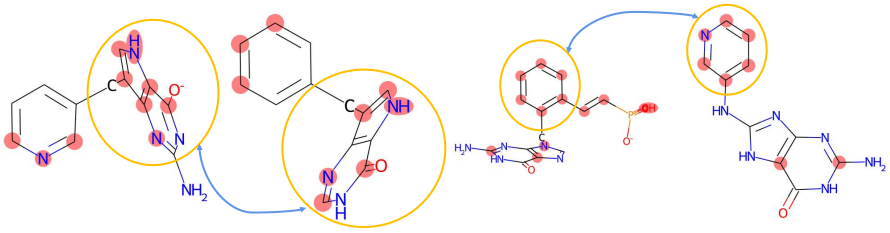


Fig. 4. Visualization of key subgraphs of the actives for a target protein PNPH (the DUD-E target name). The highlighted atoms in the orange circles belong to the identified common key sub-structures of the active molecules. The blue arrows indicate the identified similar sub-structure of the molecule, which is consistent with the function of pharmacophore in bioinformatics.

support set with 5 actives/decoys or support set with 3 actives/decoys, and the query set with 64 actives/decoys or query set with 32 actives/decoys. The results are shown in Table 2. One can see that although with the decrease of training samples in the support set and query set, the performance decreases simultaneously, PCMS can still achieve good performances after few-shot fine-tuning, which shows the robustness of our proposed PCMS framework.

5.5 Visualized Examples

Finally, we provide visualized examples of the identified key sub-graphs, i.e., the pharmacophore of the molecule. The visualization results are given in Fig. 4. As one can see, the highlighted atom sets in the orange circles share a similar structure, which agrees with the function of the pharmacophore, a common effective substructure of active molecules for DTI. Note that there are still a few isolated atoms highlighted in Fig. 4 that are outside of the orange circle. This does not mean that these atoms alone are the most important substructures. As DL methods are still commonly based on probabilistic models, the highlighted atoms may act as a hint that they are quite important to DTI prediction to some extent. The surrounding structure could be helpful to DTI prediction as well.

5.6 Inference Time Comparison

In this part, we provide the comparison of the inference time of our proposed PCMS framework and the baseline models. The results are shown in the following Table 3. Here we calculate the zero-shot and five-shot inference time for comparison. As we process the whole protein structure for each task while the other baselines use a simpler protein feature, the time consumption of our proposed method seems a little longer than the other baselines. However, in practice, we only need to calculate the protein embedding once for all molecule predictions, which will significantly decrease the calculation time.

TABLE 3
Inference time of our PCMS framework with the baselines on the DUD-E datasets. (The abbreviation ZSPGT stands for Zero-shot parameter generation time (s) per protein on DEKOIS 2.0 dataset, ZSIT stands for Zero-shot inference time (s) on DEKOIS2.0 dataset per molecule samples and FSIT stands for 5-shot inference time (s) on DEKOIS2.0 dataset per molecule samples.)

Model	ZSPGT (s)	ZSIT (s)	FSIT (s)
DeepConv-DTI	\	0.0025 ± 0.0000	0.0019 ± 0.0000
GraphDTA	\	0.0043 ± 0.0000	0.0046 ± 0.0000
IRM(GraphDTA)	\	0.0043 ± 0.0000	0.0046 ± 0.0000
MolTrans	\	0.0029 ± 0.0000	0.0046 ± 0.0000
Meta-MGNN	\	0.0021 ± 0.0000	0.0023 ± 0.0000
PCMS	**1.7448 ± 0.1050**	**0.0116 ± 0.0000**	**0.0102 ± 0.0000**

6 Conclusions

In this paper, we investigated how to improve drug target interaction (DTI) prediction when few known interactions and homologous proteins are available for a newly discovered protein. Instead of following traditional strategies for drug-target pair prediction, we formulated the DTI prediction problem as a meta-learning problem where each task is a protein-specific binary classification problem. Based on this formulation, we proposed a new parameter generation zero-shot learning master/slave framework, PCMS, which takes advantage of the protein pocket information to fast adapt the Slave Learner to the test task. Moreover, we proposed a hierarchical Protein GNN which leverages atom level, residue level, and protein level information to embed the protein. Finally, we conducted extensive experiments and analyses on two benchmark datasets, which showed the effectiveness of our approach compared with state-of-the-art DTI prediction methods.

Acknowledgments

This research was partially supported by National Natural Science Foundation of China under Grant No. 92259202, Zhejiang Provincial Key R&D Program of China under Grant No. 2023C03053, and GuangZhou City's Key R&D Program of China under Grant No. 2024B01J1301.

References

[1] M. Bagherian, E. Sabeti, K. Wang, M. A. Sartor, Z. Nikolovska-Coleska, and K. Najarian, "Machine learning approaches and databases for prediction of drug-target interaction: A survey paper," Briefings in Bioinformatics, vol. 22, no. 1, pp. 247–269, 2021.

[2] K. Abbasi, P. Razzaghi, A. Poso, S. Ghanbari-Ara, and A. Masoudi-Nejad, "Deep learning in drug target interaction prediction: Current and future perspectives," Current Medicinal Chemistry, vol. 28, no. 11, pp. 2100–2113, 2021.

[3] L. S. Jung and Y.-R. Cho, "Survey of network-based approaches of drug-target interaction prediction," in BIBM. IEEE, 2020, pp. 1793–1796.

[4] J. Palanisamy, S. Malarvizhi, and D. Shayamala, "A survey on drug discovery in medical application using artificial intelligence," Humanities, vol. 7, no. 4, pp. 99–102, 2020.

[5] K. Atz, F. Grisoni, and G. Schneider, "Geometric deep learning on molecular representations," Nature Machine Intelligence, pp. 1–10, 2021.

[6] D. Barnum, J. Greene, A. Smellie, and P. Sprague, "Identification of common functional configurations among molecules," Journal of Chemical Information and Computer Sciences, vol. 36, no. 3, pp. 563–571, 1996.

[7] A. Smellie, S. L. Teig, and P. Towbin, "Poling: Promoting conformational variation," Journal of Computational Chemistry, vol. 16, no. 2, pp. 171–187, 1995.

[8] H. Li, J. Sutter, and R. Hoffmann, "HypoGen: An automated system for generating 3D predictive pharmacophore models," Pharmacophore Perception, Development, and Use in Drug Design, vol. 2, p. 171, 2000.

[9] S. Li, J. Zhou, T. Xu, L. Huang, F. Wang, H. Xiong, W. Huang, D. Dou, and H. Xiong, "Structure-aware interactive graph neural networks for the prediction of protein-ligand binding affinity," in KDD, 2021, pp. 975–985.

[10] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, "SchNet – a deep learning architecture for molecules and materials," The Journal of Chemical Physics, vol. 148, no. 24, p. 241722, 2018.

[11] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko et al., "Highly accurate protein structure prediction with AlphaFold," Nature, vol. 596, no. 7873, pp. 583–589, 2021.

[12] J. Yu, T. Xu, Y. Rong, Y. Bian, J. Huang, and R. He, "Graph information bottleneck for subgraph recognition," in ICLR, 2021.

[13] M. D. Donsker and S. S. Varadhan, "Asymptotic evaluation of certain markov process expectations for large time, I," Communications on Pure and Applied Mathematics, vol. 28, no. 1, pp. 1–47, 1975.

[14] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rihawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger et al., "ProtTrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.

[15] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," in IJCAI, 2020.

[16] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in ICLR, 2018.

[17] M. Jiang, "Cross-network learning with partially aligned graph convolutional networks," in KDD, 2021, pp. 746–755.

[18] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," NIPS, vol. 30, 2017.

[19] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra et al., "Matching networks for one shot learning," NIPS, vol. 29, pp. 3630–3638, 2016.

[20] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in CVPR, 2018, pp. 1199–1208.

[21] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in ICML, 2017, pp. 1126–1135.

[22] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," in NIPS, 2016, pp. 3981–3989.

[23] J. Jiménez, M. Skalic, G. Martinez-Rosell, and G. De Fabritiis, "k_{DEEP}: Protein–ligand absolute binding affinity prediction via 3D-convolutional neural networks," Journal of Chemical Information and Modeling, vol. 58, no. 2, pp. 287–296, 2018.

[24] Y. Kwon, W.-H. Shin, J. Ko, and J. Lee, "AK-Score: Accurate protein-ligand binding affinity prediction using an ensemble of 3D-convolutional neural networks," International Journal of Molecular Sciences, vol. 21, no. 22, p. 8424, 2020.

[25] M. Ragoza, J. Hochuli, E. Idrobo, J. Sunseri, and D. R. Koes, "Protein – ligand scoring with convolutional neural networks," Journal of Chemical Information and Modeling, vol. 57, no. 4, pp. 942–957, 2017.

[26] J. Xu, J. Zhou, Y. Jia, J. Li, and X. Hui, "An adaptive master-slave regularized model for unexpected revenue prediction enhanced with alternative data," in ICDE, 2020, pp. 601–612.

[27] Y. Yamanishi, M. Araki, A. Gutteridge, W. Honda, and M. Kanehisa, "Prediction of drug-target interaction networks from the integration of chemical and genomic spaces," Bioinformatics, vol. 24, no. 13, pp. i232–i240, 2008.

[28] F. Cheng, C. Liu, J. Jiang, W. Lu, W. Li, G. Liu, W. Zhou, J. Huang, and Y. Tang, "Prediction of drug-target interactions and drug repositioning via network-based inference," PLoS Computational Biology, vol. 8, no. 5, p. e1002503, 2012.

[29] X. Chen, M.-X. Liu, and G.-Y. Yan, "Drug-target interaction prediction by random walk on the heterogeneous network," Molecular BioSystems, vol. 8, no. 7, pp. 1970–1978, 2012.

[30] J. Klicpera, J. Groß, and S. Günnemann, "Directional message passing for molecular graphs," in ICLR, 2019.

[31] Z. Guo, C. Zhang, W. Yu, J. Herr, O. Wiest, M. Jiang, and N. V. Chawla, "Few-shot graph learning for molecular property prediction," in WebConf, 2021, pp. 2559–2567.

[32] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, "Strategies for pre-training graph neural networks," in ICLR, 2020.

[33] K. Huang, C. Xiao, L. M. Glass, and J. Sun, "MolTrans: Molecular interaction transformer for drug-target interaction prediction," Bioinformatics, vol. 37, no. 6, pp. 830–836, 2021.

[34] H. Wang, G. Zhou, S. Liu, J.-Y. Jiang, and W. Wang, "Drug-target interaction prediction with graph attention networks," arXiv preprint arXiv:2107.06099, 2021.

[35] I. Lee, J. Keum, and H. Nam, "DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences," PLoS Computational Biology, vol. 15, no. 6, p. e1007129, 2019.

[36] P. J. Ballester and J. B. Mitchell, "A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking," Bioinformatics, vol. 26, no. 9, pp. 1169–1175, 2010.

[37] H. Shi, S. Liu, J. Chen, X. Li, Q. Ma, and B. Yu, "Predicting drug-target interactions using Lasso with random forest based on evolutionary information and chemical structure," Genomics, vol. 111, no. 6, pp. 1839–1852, 2019.

[38] T. Nguyen, H. Le, T. P. Quinn, T. Nguyen, T. D. Le, and S. Venkatesh, "GraphDTA: Predicting drug-target binding affinity with graph neural networks," Bioinformatics, vol. 37, no. 8, pp. 1140–1147, 2021.

[39] M. M. Mysinger, M. Carchia, J. J. Irwin, and B. K. Shoichet, "Directory of useful decoys, enhanced (DUD-E): Better ligands and decoys for better benchmarking," Journal of Medicinal Chemistry, vol. 55, no. 14, pp. 6582–6594, 2012.

[40] "Protein data bank: The single global archive for 3D macromolecular structure data," Nucleic Acids Research, vol. 47, no. D1, pp. D520–D528, 2019.

[41] G. Landrum, "RDKit: A software suite for cheminformatics, computational chemistry, and predictive modeling," 2013.

[42] P. J. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski et al., "Biopython: Freely available Python tools for computational molecular biology and bioinformatics," Bioinformatics, vol. 25, no. 11, pp. 1422–1423, 2009.

[43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," NIPS, vol. 30, 2017.

[44] M. R. Bauer, T. M. Ibrahim, S. M. Vogel, and F. M. Boeckler, "Evaluation and optimization of virtual screening workflows with DEKOIS 2.0 – a public library of challenging docking benchmark sets," Journal of Chemical Information and Modeling, vol. 53, no. 6, pp. 1447–1462, 2013.

[45] L. Chen, A. Cruz, S. Ramsey, C. J. Dickson, J. S. Duca, V. Hornak, D. R. Koes, and T. Kurtzman, "Hidden bias in the DUD-E dataset leads to misleading performance of deep learning in structure-based virtual screening," PLOS One, vol. 14, no. 8, p. e0220113, 2019.

[46] J. Yang, C. Shen, and N. Huang, "Predicting or pretending: Artificial intelligence for protein-ligand interactions lack of sufficiently

large and unbiased datasets," Frontiers in Pharmacology, vol. 11, p. 69, 2020.
[47] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, and T. Langer, "A compact review of molecular property prediction with graph neural networks," Drug Discovery Today: Technologies, vol. 37, pp. 1–12, 2020.
[48] J. Shen and C. A. Nicolaou, "Molecular property prediction: Recent trends in the era of artificial intelligence," Drug Discovery Today: Technologies, vol. 32, pp. 29–36, 2019.
[49] A. Cereto-Massagué, M. J. Ojeda, C. Valls, M. Mulero, S. Garcia-Vallvé, and G. Pujadas, "Molecular fingerprint similarity search in virtual screening," Methods, vol. 71, pp. 58–63, 2015.
[50] G. B. Goh, N. O. Hodas, C. Siegel, and A. Vishnu, "SMILES2Vec: An interpretable general-purpose deep neural network for predicting chemical properties," arXiv preprint arXiv:1712.02034, 2017.
[51] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," Pattern Recognition, vol. 30, no. 7, pp. 1145–1159, 1997.
[52] B. Rozemberczki, C. T. Hoyt, A. Gogleva, P. Grabowski, K. Karis, A. Lamov, A. Nikolov, S. Nilsson, M. Ughetto, Y. Wang et al., "ChemicaIX: A deep learning library for drug pair scoring," in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 3819–3828.
[53] X. Ru, X. Ye, T. Sakurai, and Q. Zou, "NerLTR-DTA: Drug–target binding affinity prediction based on neighbor relationship and learning to rank," Bioinformatics, vol. 38, no. 7, pp. 1964–1971, 2022.
[54] Z. Zhang, L. Chen, F. Zhong, D. Wang, J. Jiang, S. Zhang, H. Jiang, M. Zheng, and X. Li, "Graph neural network approaches for drug-target interactions," Current Opinion in Structural Biology, vol. 73, p. 102327, 2022.
[55] Y. Ding, J. Tang, F. Guo, and Q. Zou, "Identification of drug-target interactions via multiple kernel-based triple collaborative matrix factorization," Briefings in Bioinformatics, vol. 23, no. 2, 2022.
[56] Y. Ji, L. Zhang, J. Wu, B. Wu, L.-K. Huang, T. Xu, Y. Rong, L. Li, J. Ren, D. Xue et al., "Drugood: Out-of-distribution (ood) dataset curator and benchmark for ai-aided drug discovery–a focus on affinity prediction problems with noise annotations," arXiv preprint arXiv:2201.09637, 2022.
[57] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, "Invariant risk minimization," arXiv preprint arXiv:1907.02893, 2019.

Yuyang Xu received his B.S. degree from Jilin University, in 2021. He is now studying for the Ph.D. degree in the College of Computer Science and Technology, Zhejiang University. His major research interest is data mining in the medical field.

Jingbo Zhou is a staff research scientist and research manager of Baidu Research, in charge of the Business Intelligence Lab. He holds a Ph.D. in Computer Science from the National University of Singapore (2014) and a B.E. degree from Shandong University (2009). His research focuses on spatio-temporal big data, deep geometric learning, knowledge graph, and their applications in areas such as urban computing, drug discovery, and intelligent healthcare. He has published over 50 papers in top conferences and journals, including KDD, SIGMOD, ICDE, AAAI, Nature Machine Intelligence, Nature Sustainability, Bioinformatics, and TKDE. Jingbo regularly serves as a committee member to prestigious conferences such as KDD, AAAI, IJCAI, ACL, and NeurIPS. Some of his research and development accomplishments have been applied across multiple Baidu products, including Baidu Maps, Baidu Search Ads, Baidu City Brain, and Baidu Computational Biology Platform.

Haochao Ying is currently an associate professor in the School of Public Health, Zhejiang University. He received the Ph.D. degree in the College of Computer Science from Zhejiang University in 2019, and the B.S. degree in computer science and technology from Zhejiang University of Technology in 2014. His research interests include data mining for healthcare and personalized recommender system. He has authored some papers at prestigious international conferences and journals, such as TKDE, TMI, IJCAI, ICML, and CVPR.

Jintai Chen is currently a Ph.D. candidate in the College of Computer Science and Technology, Zhejiang University. His research interests include AI for healthcare, computer vision and tabular learning. He has published over 20 papers at top-tier international conferences and journals, such as ICML, ICLR, CVPR, AAAI, IJCAI, JBHI.

Wei Chen is a professor in the State Key Lab of CAD&CG, Zhejiang University. His research interests include visualization and visual analysis, and has published more than 80 IEEE/ACM Transactions and IEEE VIS papers. He actively served as several associate editors of ACM/IEEE Transactions.

Danny Z. Chen received the B.S. degrees in Computer Science and in Mathematics from the University of San Francisco, California in 1985, and the M.S. and Ph.D. degrees in Computer Science from Purdue University, West Lafayette, Indiana in 1988 and 1992, respectively. He is a Professor in the Department of Computer Science and Engineering, the University of Notre Dame. His main research interests include computational biomedicine, biomedical imaging, computational geometry, algorithms and data structures, machine learning, data mining, and VLSI. He has published many papers in these areas, and holds 8 US patents for technology development in biomedical applications. He is a Fellow of IEEE and a Distinguished Scientist of ACM.

Jian Wu is a full Professor at Zhejiang University. He is currently the director of Real Doctor AI Research Centre of Zhejiang University. His research interests include Artificial Intelligence, Data Mining and their applications in healthcare and biomedicine. He received his Ph.D. degree in Computer Science and Technology from Zhejiang University. He has published more than 200 papers in some prestigious refereed journals and conference proceedings such as IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MEDICAL IMAGING, CVPR, IJCAI, AAAI, ICML, MICCAI. He is a Distinguished Member of CCF.