Improving Pareto Front Learning via Multi-Sample Hypernetworks

Long P. Hoang[1,2*], **Dung D. Le**[1], **Tran Anh Tuan**[2], **Tran Ngoc Thang**[2]

[1] College of Engineering and Computer Science, VinUniversity
[2] School of Applied Mathematics and Informatics, Hanoi University of Science and Technology
long.hp@vinuni.edu.vn, dung.ld@vinuni.edu.vn, tuan.ta181295@sis.hust.edu.vn, thang.tranngoc@hust.edu.vn

Abstract

Pareto Front Learning (PFL) was recently introduced as an effective approach to obtain a mapping function from a given trade-off vector to a solution on the Pareto front, which solves the multi-objective optimization (MOO) problem. Due to the inherent trade-off between conflicting objectives, PFL offers a flexible approach in many scenarios in which the decision makers can not specify the preference of one Pareto solution over another, and must switch between them depending on the situation. However, existing PFL methods ignore the relationship between the solutions during the optimization process, which hinders the quality of the obtained front. To overcome this issue, we propose a novel PFL framework namely PHN-HVI , which employs a hypernetwork to generate multiple solutions from a set of diverse trade-off preferences and enhance the quality of the Pareto front by maximizing the Hypervolume indicator defined by these solutions. The experimental results on several MOO machine learning tasks show that the proposed framework significantly outperforms the baselines in producing the trade-off Pareto front.

1 Introduction

Multi-objective optimization has been shown prevalent in many machine learning applications with conflicting objectives such as in computer vision (**???**), speech & natural language processing (**??**), and recommender system (**????**), etc. (**?**) states that multi-task learning can be formulated as a multi-objective optimization (MOO) problem, in which the optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. Typically in MOO problems, the set of optimal solutions is called Pareto front, in which each solution on the front represents different trade-off between the objectives. However, most of the recent MOO algorithms must know the trade-off in advance or require a separate model to be trained for each point on the Pareto front (**?**). This limits the flexibility as there are many situations in which the user can only make a trade-off decision on the fly, such as selecting between the shortest but congested route and the farther but more sparse one, or adapting the investment strategy when the marketplace changes.

For that reason, (**???**) recently explore and develop a research direction called Pareto Front Learning (PFL), which attempts to approximate the entire Pareto front. However, these works only update the weights in the optimization algorithms by a random preference vector at each iteration, bringing with them restrictions in MOO that render them inappropriate for the PFL problem (see Section 5 for more details), and ignoring the dynamic relationship between the Pareto optimal solutions. We argue that this information is crucial in producing a high-quality Pareto front because having a variety of solutions representing different trade-off preferences between the objective functions offers us a global view of the current Pareto front. Therefore, we propose to use hypervolume indicator to guide the optimization process. Specifically, we build upon the work of (**?**) to allow the derivation of multiple Pareto optimal solutions from multiple preference vectors, followed by the hypervolume (defined by the obtained solutions) maximization step to improve the learnt Pareto front.

Contributions. Our contributions can be summarized in the following:

- *Firstly,* we provide a mathematical formulation of PFL with Hypernetwork, which originally introduced in (**?**).
- *Secondly,* we propose a novel framework, namely PHN-HVI that improves PFL problem using multi-sample hypernetworks and hypervolume indicator maximization.
- *Thirdly,* we conduct comprehensive experiments on several multi-task learning datasets to validate the effectiveness of PHN-HVI compared to baseline methods.

Organization. Section 2 provides the background knowledge for the PFL problem. In Section 3, we describe our framework PHN-HVI and its efficient variant with the partitioning algorithm. In Section 4, we review related work in the literature and briefly discuss both of their accomplishments and shortcomings. In Section 5, we conduct experiments from comparison and provide a model behavior analysis. We conclude the paper in Section 6 and discuss possible research directions as future work.

2 Preliminary

Multi-task learning seeks to find $\theta^* \in \Theta$ to optimize J loss functions as described in the following:

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(x,y)\sim p_D} \mathcal{L}\left(y, f\left(x;\theta\right)\right) \tag{1}$$

$$\mathcal{L}\left(y, f\left(x;\theta\right)\right) = \left\{\mathcal{L}_1\left(y, f\left(x;\theta\right)\right), \ldots, \mathcal{L}_J\left(y, f\left(x;\theta\right)\right)\right\}$$

*The author started this paper as part of the graduation thesis.

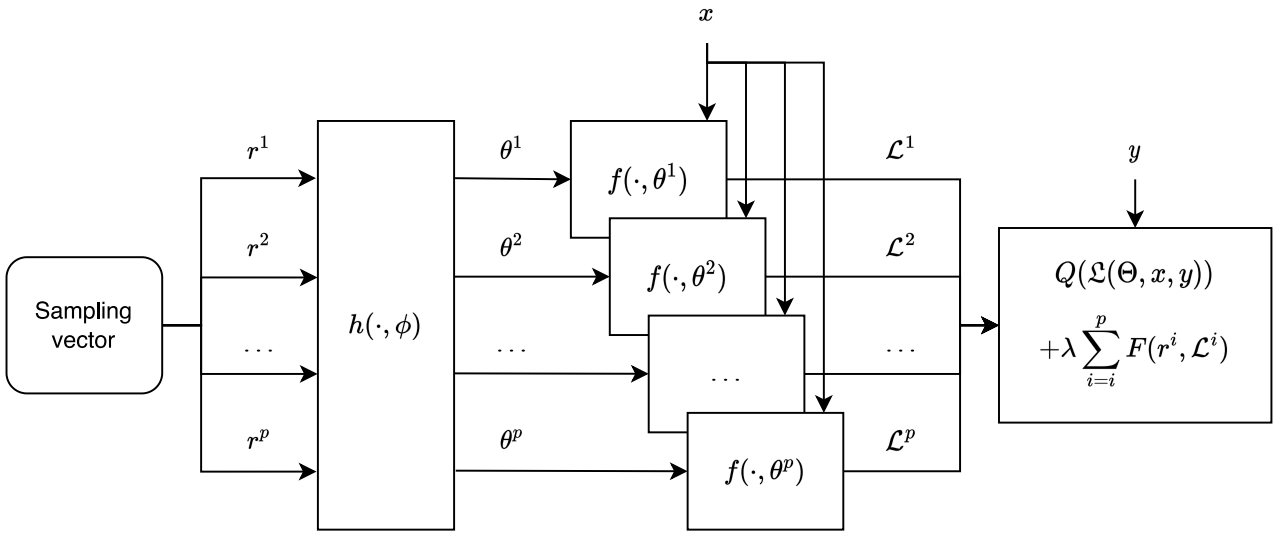

Figure 1: Multi-Sample Hypernetwork framework

in which, p_D denotes the data distribution, $\mathcal{L}_j(\cdot,\cdot) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{>0}, \forall j \in \{1,\dots,J\}$ denotes the $j-th$ loss function, and $f(x;\theta) : \mathcal{X} \times \vartheta \to \mathcal{Y}$ denotes the neural network with parameter θ.

Definition 1 (Dominance)**.** *A solution θ^a is said to dominate another solution θ^b if*

$$\mathcal{L}^j(y, f(x,\theta^a)) \leq \mathcal{L}^j(y, f(x,\theta^b)), \forall j \in \{1, ..., J\}$$

and $\mathcal{L}(y, f(x,\theta^a)) \neq \mathcal{L}(y, f(x,\theta^b))$. We denote this relationship as $\theta^a \prec \theta^b$.

Definition 2 (Pareto optimal solution)**.** *A solution θ^a is called Pareto optimal solution if there exists no θ^b such as $\theta^b \prec \theta^a$.*

Definition 3 (Pareto front)**.** *The set of Pareto optimal is Pareto set, denoted by $\mathcal{P}$, and the corresponding images in objectives space are Pareto front $\mathcal{P}_f = \mathcal{L}(y, f(x, \mathcal{P}))$.*

In (**?**), the authors introduced the term Pareto Front Learning. Here we provide a mathematical formulation of Pareto Front Learning with Hypernetwork, which sets the basis for our proposed framework in Section 3:

Definition 4 (PFL with Hypernetwork)**.** *Pareto Front Learning is a one-shot optimization approach to approximate the entire Pareto front by solving problem:*

$$\begin{aligned} &\phi^* = \arg\min_{\phi \in \mathbb{R}^n} \mathbb{E}_{r \sim p_{\mathcal{S}^J}, (x,y) \sim p_D} \; F(\mathcal{L}(y, f(x, \theta^r)), r) \\ &\text{s.t. } \theta^r = h(r, \phi) \in \mathcal{P}, h(\Omega, \phi^*) = \mathcal{P} \end{aligned} \tag{2}$$

where $h : \mathcal{S}^J \times \Phi \to \Theta$, random variable r is the preference vector that formulates a trade-off between loss functions, $\mathcal{S}^J = \{\lambda \in \mathbb{R}^J_{>0} : \sum_j \lambda_j = 1\}$ is the set feasible values of random variable r and $p_{\mathcal{S}^J}$ is a random distribution on $\mathcal{S}^J$ and $F(\cdot,\cdot) : \mathcal{S}^J \times \mathbb{R}^J \to \mathbb{R}$ is an extra criterion function, which helps us map a given preference vector with a Pareto optimal solution.

3 Multi-Sample Hypernetwork

Multi-Sample Hypernetwork samples p preference vectors r^i and use $h(\cdot,\phi)$ to generate p target networks $f(\cdot,\theta^i), i \in \{1,\dots,p\}$ where $\theta^i = h(r^i,\phi)$. Denote $\mathcal{L}^i = (\mathcal{L}_1(y, f\left(x;\theta^i\right)), \dots, \mathcal{L}_J(y, f\left(x;\theta^i\right)))$ is the vector of loss values, and $\mathfrak{L}(\Theta, x, y) = \left[\mathcal{L}^1, \dots, \mathcal{L}^p\right]$. Choose $p_{\mathcal{S}^J}$ as $\text{Dir}(\alpha)$. Different from (**?**), Multi-Sample Hypernetwork is designed to solve:

$$\min_{\phi} \; \mathbb{E}_{\substack{r^i \sim \text{Dir}(\alpha) \\ (x,y) \sim p_D}} Q(\mathfrak{L}(\Theta, x, y)) + \sum_{i=1}^{p} F(r^i, \mathcal{L}^i) \tag{3}$$

where $Q : \mathbb{R}^J \to \mathbb{R}$ is a monotonically decreasing function, meaning $\forall \theta^b \in \Theta^B \exists \theta^a \in \Theta^A : \theta^a \prec \theta^b \Rightarrow Q(\mathfrak{L}(\Theta^A, x, y)) < Q(\mathfrak{L}(\Theta^B, x, y))$ or $\nabla Q(\mathfrak{L}(\Theta, x, y))$ create a Pareto improvement, and $F : \mathcal{S}^J \times \mathbb{R}^J \to \mathbb{R}$ is a complete function, for example, $\{\arg\min_{\mathcal{L} \in \mathcal{P}_f} F(r, \mathcal{L}) : \forall r \in \mathcal{S}^J\} = \mathcal{P}_f$.

HV Indicator Hypernetwork

HV Indicator Hypernetwork (PHN-HVI) solves the problem:

$$\min_{\phi} -\mathbb{E}_{\substack{r^i \sim \text{Dir}(\alpha) \\ (x,y) \sim p_D}} \text{HV}\left(\mathfrak{L}(\Theta; x, y)\right) + \lambda \sum_{i=i}^{p} \cos(\vec{r^i}, \vec{\mathcal{L}^i}) \tag{4}$$

Lemma 1 ((**?**))**.** *If* $\text{HV}\left(\mathfrak{L}(\Theta; x, y)\right)$ *is maximal, then* $\theta^i \in \mathcal{P}, i \in \{1,\dots,p\}$.

The Hypervolume (HV) function will maximize the hypervolume of the Pareto front, pushing the Pareto front approximated by hypernetwork to the truth Pareto front. Meanwhile, the cosine similarity penalty function will spread the Pareto front and get the Pareto optimal solution close to the given preference vector. It is clear that cosine similarity is a function with a complete property, HV is a monotonic function.

Using hypervolume gradient algorithm of the independent p neural networks described in (**?**), we obtain the following, which is scalable for Domination-Ranked Fronts (**??**):

$$\nabla_{\mathcal{L}^i}\left[-\mathrm{HV}\left(\mathfrak{L}(\Theta;x,y)\right)\right] = \left(-\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\Theta;x,y)\right)}{\partial \mathcal{L}_1(y,f(x,\theta^i))},\ldots,-\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\Theta;x,y)\right)}{\partial \mathcal{L}_p(y,f(x,\theta^i))}\right) \quad (5)$$

We can approximate the descent direction of HV function for parameter ϕ based on (5) by:

$$d = -\sum_{i=1}^{p}\sum_{j=1}^{J}\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\Theta;x,y)\right)}{\partial \mathcal{L}_j(\theta^i,f(x,\theta^i))}\frac{\partial \mathcal{L}(\theta^i,f(x,\theta^i))}{\partial \phi} \quad (6)$$

The final update direction of HV Indicator Hypernetwork is:

$$d_{update} = d - \lambda\sum_{i=1}^{p}\frac{\partial}{\partial\phi}\left(\frac{\vec{r^i}\vec{\mathcal{L}}(y,f\left(x;\theta^i\right))}{\|\vec{r^i}\|\|\vec{\mathcal{L}}(y,f\left(x;\theta^i\right))\|}\right) \quad (7)$$

It should be noted that ϕ is the only parameter for PHN-HVI that has to be optimized. On the inference phase, PHN-HVI just needs one vector r to provide a corresponding Pareto local optimum θ^r.

Partitioning Algorithm

It is necessary to sample the preference vectors $r^1,\ldots,r^p$ to cover the objective space evenly to make HV and cosine similarity interact well. This is quite simple in 2D space using $p+1$ vectors $\{\cos(\frac{i\pi}{2p}),\sin(\frac{i\pi}{2p})\}$ to partition the object space into p subregions $\Omega^i, i=1,...p$ and randomly sample the vector r^i of subregion Ω^i. However, how to take a partitioned random sample in any dimensional J space? A good partition with N subregions must satisfy:

$$\cup_{i=1}^{N}(\Omega^i) = S^J \text{ and } \Omega^{i'}\cap_{i'\neq i}\Omega^i = \emptyset \quad (8)$$

Base on (**?**), we define subregions Ω^i by points $u = (u_1,\ldots,u_J)\in U\subset \mathcal{S}^J$ such that:

$$u_1\in\{0,\delta,2\delta,\ldots,...,1\}\text{ s.t }\frac{1}{\delta}=k\in\mathbb{N}^* \quad (9)$$

If $1<j<J-1$, $m_i=\frac{\delta}{u_i}, 1\leq i<j-1$, we have:

$$u_j\in\{0,\delta,\ldots,(p-\sum_{i=1}^{j-1}m_i)\delta\},\ u_J = 1-\sum_{j=1}^{J-1}u_j \quad (10)$$

Using the Delaunay triangulation algorithm (**?**) for these points, we obtain the required partition. Figure 2 is the illustrative result of the algorithm in 3D space. However, the number of points $u\in U$ in the space $\mathbb{R}^J$ and $k=\frac{1}{\delta}$ is $\binom{J+k-1}{k}$. Therefore, we will not be able to freely set the number of rays p for the HV Indicator Hypernetwork, and the number of partitions also increases exponentially as J and k increase, resulting in a huge amount of computation. Therefore, we will only use the partition sampling algorithm for the 2-objective optimization problem and still randomly sample the preference vectors $r^1,\ldots,r^p\sim\mathrm{Dir}(\alpha)$ with optimization problems of 3 or more objective functions.

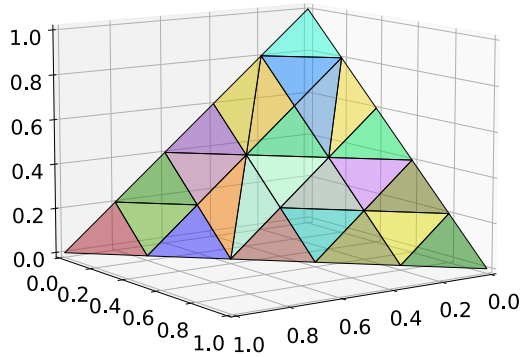

Figure 2: Partitioning algorithm with $J=3,\delta=0.2$

Algorithm 1: PHN-HVI optimization algorithm

while not converged **do**
  if $J=2$ **then**
    $r^1,\ldots,r^p\sim$ partition_sampling
  else
    $r^1,\ldots,r^p\sim\mathrm{Dir}(\alpha)$
  end if
  Compute $\left[\theta^i:=h(r^i,\phi)\right]_{i=1}^{p}$
  Compute d by Equation (6)
  $d_{update}:=d-\lambda\sum_{i=1}^{p}\frac{\partial}{\partial\phi}\left(\frac{\vec{r^i}\vec{\mathcal{L}}(y,f\left(x;\theta^i\right))}{\|\vec{r^i}\|\|\vec{\mathcal{L}}(y,f(x;\theta^i))\|}\right)$
  $\phi\leftarrow\phi-\eta\times d_{update}$
end while
return ϕ

Stein Variational Hypernetwork

Beside Hypervolume maximization, it is also possible to integrate the multi-sample hypernetwork with Stein variational gradient descent (**?**) for profiling the Pareto front. Similar to PHN-HVI , Stein variational hypernetwork generates p target networks.

Denote $g(\phi,r^i)$ is min norm vector in the convex hull $\mathcal{CH}_{\mathcal{L}^i}$ of the gradients $\nabla_\phi\mathcal{L}^i$:

$$g(\phi,r^i)=\arg\min_{g\in\mathcal{CH}_{\mathcal{L}^i}}\|g\|^2 \quad (11)$$

We can approximate the update direction for an input trade-off vector r^i as follows:

$$d^i=\sum_{i'=1}^{p}\left[g(\phi,r^{i'})k(\mathcal{L}^i,\mathcal{L}^{i'})-\nabla_\phi k(\mathcal{L}^i,\mathcal{L}^{i'})\right] \quad (12)$$

where $k(\mathcal{L}^i,\mathcal{L}^{i'})=\det(2\pi\sigma^2 I)^{-\frac{1}{2}}\exp(-\frac{1}{2\sigma^2}\|\mathcal{L}^i-\mathcal{L}^{i'}\|^2)$. The update direction of the hypernetwork's parameter is:

$$d_{update}=\sum_{i}^{p}d^i-\lambda\sum_{i=1}^{p}\frac{\partial}{\partial\phi}\left(\frac{\vec{r^i}\vec{\mathcal{L}}(y,f\left(x;\theta^i\right))}{\|\vec{r^i}\|\|\vec{\mathcal{L}}(y,f\left(x;\theta^i\right))\|}\right) \quad (13)$$

This demonstrates the flexibility of using Multi-Sample Hypernetworks to enable the iterative update of a set of solution points simultaneously to push them toward the Pareto front. However, we reserve the investigation of this aspect for future work.

4 Related Work

Multi-Objective Optimization. Rather than a single solution, MOO aims to find a set of Pareto optimal solutions with different trade-offs. For small-scale multi-objective optimization problems, genetic algorithms such as VEGA (**?**), NSGA-III (**?**), etc. are popular methods for finding a set of well-distributed Pareto optimal solutions in a single run. Scalarization algorithms (the Tchebysheff method (**?**) and its variants) use weighted functions to transform several objectives into a single objective, but, these methods need convex function conditions to approximate the entire Pareto front. Therefore, (**?**) used a monotonic optimization approach to obtain an approximation of the weakly Pareto optimal set for solving strictly quasiconvex multi-objective programming, a general case of convex multi-objective programming. Another approach, as developed by (**?**), (**?**) find a common descent direction of all objectives at each iteration, as a result, they cannot take in decision-makers preferences on the problem.

Multi-Task Learning. The use of MOO in machine learning, particularly for Multi-Task Learning, is not new. Some algorithms approach task balancing, such as (**?**) (homoscedastic uncertainty), (**?**) (balance learning), etc. (**?**) used Frank-Wofle algorithm to solve the constrained optimization problem of MGDA (**?**), while (**?**) corrects the direction of conflict gradients. Due to their approach to balanced solutions, these methods are not suitable for simulating trade-offs between objectives.

Using preference vectors, (**?**) may identify a variety of solutions by breaking down a multitask learning problem into several subproblems of many subspaces, whereas (**?**) can precisely identify the Pareto optimal solution belonging to inverse ray. (**?**) has the ability to locate Pareto optimal solutions close to a given Pareto optimal solution. The Pareto front with finite points approximated by several neural networks is another research direction. (**?**) has done this using Hypervolume Maximization. The techniques for calculating the gradient of the hypervolume were created by (**?**), and (**??**) further improved them. To increase the entropy of these points, (**?**) combined MGDA (**?**) with Stein Variational Gradient Descent (**?**). Unfortunately, as the number of preference vectors and objective functions rises, the computational cost of these algorithms becomes impractical.

Hypernetwork and Pareto Front Learning. Hypernetwork, which generates weights for other networks (target networks), can be applied to large-scale problems by using chunking (**?**). The result in (**?**) is poor since they constructed a hypernetwork but did not explicitly map a preference vector with a Pareto optimal solution. (**?**) presented PHN-LS and PHN-EPO, which combine optimization techniques like Linear Scalarization (LS) and EPO (**?**) with hyperwork, to approximate the complete Pareto front in a single training. As a result, the two models still contain the features and drawbacks of these techniques. In order to tackle PFL, (**?**) combined preference vectors with data using a single network and used the LS loss function together with cosine similarity. However, it is still unclear how to concatenate preference vectors with arbitrary data, and not all situations allow for efficient interaction between LS and cosine similarity.

5 Experiments

For all methods base on hypernetwork, we use a feed-forward network with various outputs to parameterize $h(\phi, r)$. The target network's weight tensor is produced by each output in a distinct way. Specifically, the input r is first mapped using a multi-layer perceptron network to a higher dimensional space in order to create shared features. A weight matrix for each layer in the target network is created by passing these features across fully connected layers. Experiments demonstrate that PHN-HVI outperforms other methods.

Baselines: We compare PHN-HVI [1] with the current state-of-the-art of PFL methods: **PHN-LS**, **PHN-EPO** (**?**), and **COSMOS** (**?**).

Evaluation Metric: The area dominated by Pareto front is known as Hypervolume (**?**). The higher Hypervolume, the better Pareto front.

Training Settings: Our target network has the same architecture as the baselines in all experiments. For toy examples, the optimization processes are run for 10,000 iterations, and 200 evenly distributed preference vectors are used for testing. On multi-task problems, the dataset is split into three subsets: training, validation, and testing. The model with the highest HV in the validation set will be evaluated. All methods are evaluated by the same well-spread preference vectors.

All experiments and methods in this paper are implemented with Pytorch (**?**) and trained on a single NVIDIA GeForce RTX3090.

Toy Examples

In this section, we will investigate the quality of the Pareto front generated by PHN-HVI and the PFL baselines using the following toy examples:

Problem 1 (?)

$$\mathcal{L}_1(\theta) = (\theta)^2 \, , \mathcal{L}_2(\theta) = (\theta - 1)^2, \text{ s.t. } \theta \in \mathbb{R} \tag{14}$$

Problem 2 (?)

$$\begin{aligned} &\mathcal{L}_1(\theta) = 1 - \exp\{-\|\theta - 1/\sqrt{d}\|_2^2\}, \\ &\mathcal{L}_2(\theta) = 1 - \exp\{-\|\theta + 1/\sqrt{d}\|_2^2\} \\ &\text{s.t } \theta \in \mathbb{R}^d, d = 100 \end{aligned} \tag{15}$$

Problem 3

$$\begin{aligned} &\mathcal{L}_1(\theta) = \cos^2(\theta_1) + 0.2, \\ &\mathcal{L}_2(\theta) = 1.3 + \sin^2(\theta_2) - \cos(\theta_1) - 0.1 \sin^5(22\pi \cos^2 \theta_1) \\ &\text{s.t } \theta \in \mathbb{R}^2 \end{aligned} \tag{16}$$

[1] We publish the code at https://github.com/longhoangphi225/MultiSample-Hypernetworks

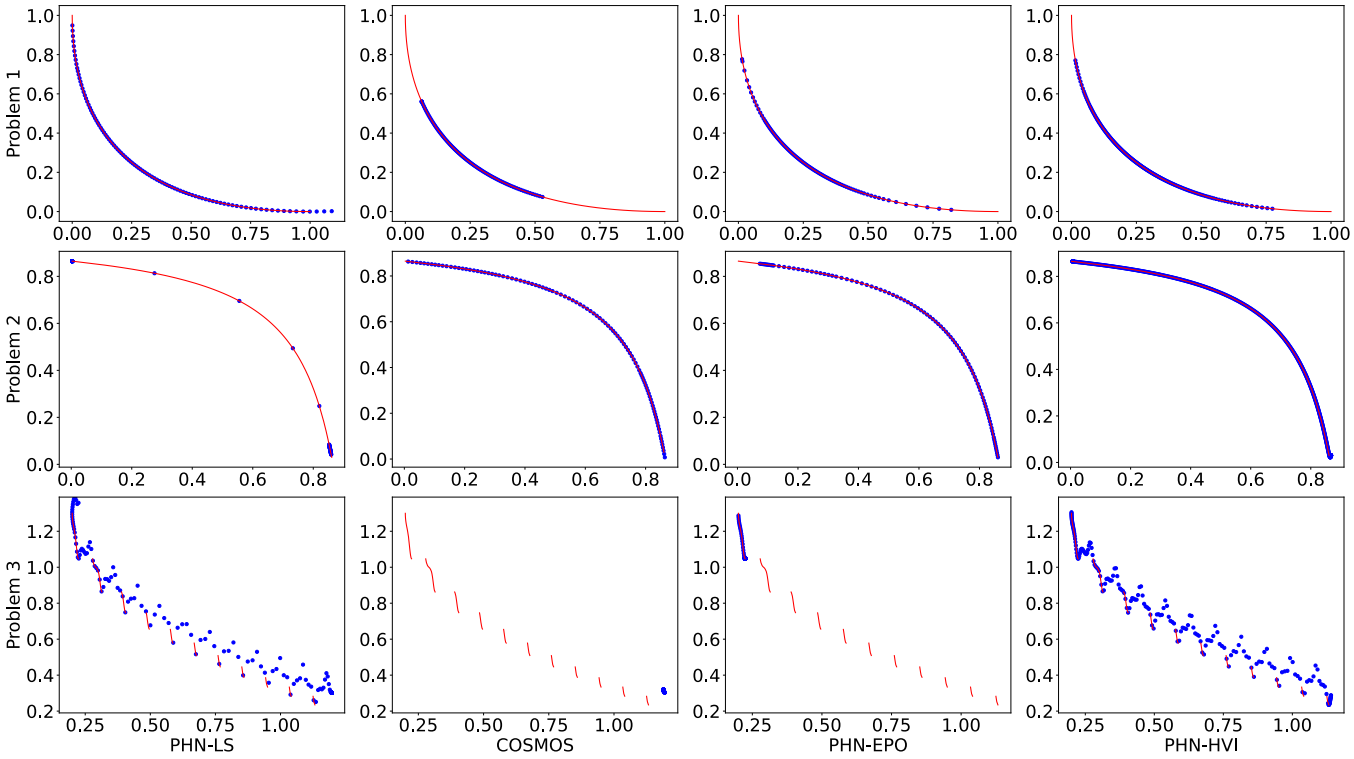

Figure 3: Toy 2D plots. The red curves are truth Pareto front. The blue points are approximate Pareto fronts

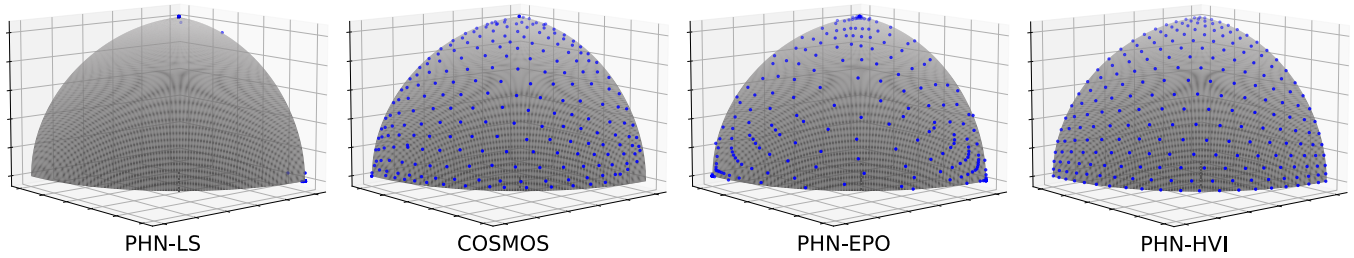

Figure 4: Toy 3D plots

Problem 4 (DTLZ4 with 3 Objective Functions)

$$\mathcal{L}_1(\theta) = \cos(\theta_1 \frac{\pi}{2}) \cos(\theta_2 \frac{\pi}{2})(\sum_{i=3}^{10}(\theta_i - 0.5)^2 + 1), \quad (17)$$

$$\mathcal{L}_2(\theta) = \cos(\theta_1 \frac{\pi}{2}) \sin(\theta_2 \frac{\pi}{2})(\sum_{i=3}^{10}(\theta_i - 0.5)^2 + 1),$$

$$\mathcal{L}_3(\theta) = \sin(\theta_1 \frac{\pi}{2})(\sum_{i=3}^{10}(\theta_i - 0.5)^2 + 1),$$

$$\text{s.t } \theta \in \mathbb{R}^{10}, 0 \leq \theta_i \leq 1$$

Since the objective function in Problem 1 is convex, practically all methodsin particular PHN-LSperform well. Due to the concave form of the Pareto front in Problem 2, all of the solutions produced by PHN-LS gravitate to two ends, but COSMOS, a method that combines LS with consine similarity, can be effective in this situation. However, it is highly problematic in Problem 3 when LS and cosine similarity function are in direct conflict.

As a result of its disconnected Pareto front, Problem 3 presents a difficult challenge. In view of hypernetwork-based method, we have $\mathcal{L}_j(h(\phi, r))$ is a smooth function, because $\mathcal{L}_j(\theta)$ is smooth, and $h(\phi, r)$ is smooth (because h is represented by a neural network). So on, if $\theta = h(\phi, r), \exists d_0, d_j, \epsilon_j \in \mathbb{R}_+, \forall r' \in \Omega : \|r' - r\| < d_0 \Rightarrow \theta' \in \mathcal{B}(\theta, d_i) : \|\mathcal{L}_j(\theta') - \mathcal{L}_j(\theta)\| < \epsilon_i, j \in \{1, \ldots, J\}$. Therefore as a consequence, in order to approximate the entire Pareto front, PHN-LS and PHN-HVI must give some point that is not a Pareto optimal solution. And EPO Search, which combines uniformity function, common direction gradient descent, and a controlled increase or decrease in the objective function, prevents PHN-EPO from generating a non-Pareto Optimal solution, resulting in the generation of just a portion of the Pareto front. In this case, PHN-LS and PHN-HVI are better than PHN-EPO, because they still can

Method	Multi-MNIST	Multi-Fashion	Fash.+MNIST	Drug	Jura	SARCOS
PHN-EPO	2.868	2.238	2.815	1.226	0.933	0.932
PHN-LS	2.859	2.219	2.764	1.208	0.932	0.934
COSMOS	2.959	2.324	2.838	NA	0.933	0.830
PHN-HVI	**3.012**	**2.408**	**2.967**	**1.294**	**0.946**	**0.949**

Table 1: Results compared to the state-of-the-art methods on Hypervolume

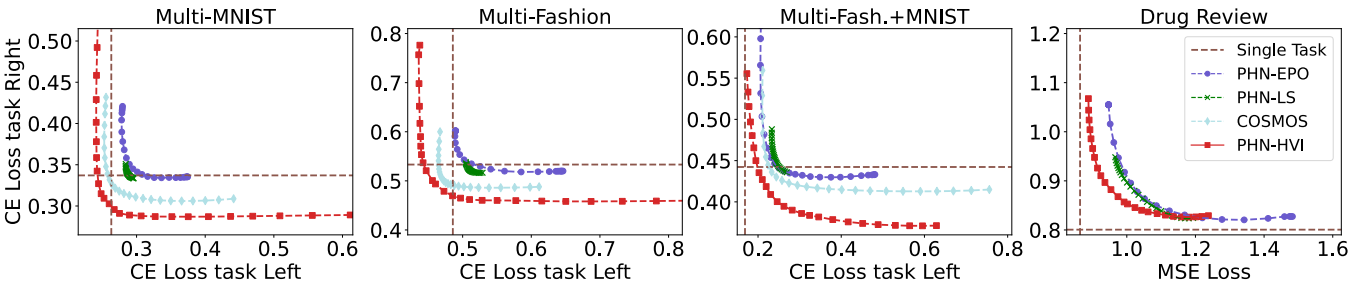

Figure 5: Pareto fronts are generated by methods

profile the Pareto front by removing the dominated solutions.

We use the sigmoid function to address constraints in Problem 4. Using the sampling technique which is described in Section 3, we use 231 uniform 3D rays. In Figure 4, PHN-LS entirely fails, COSMOS and PHN-EPO offer solutions that are widespread but a little chaotic, while PHN-HVI produces solutions that result in a pretty uniform distribution.

Image Classification

Three benchmark datasetsMulti-MNIST, Multi-Fashion, and Multi-Fashion+MNIST (**?**) are used in our evaluation. For each dataset, we have a two-objective multitask learning problem that asks us to categorize the top-left (task left) and bottom-right (task right) items (task right). Each dataset has 20,000 test set instances and 120,000 training examples. 10% of the training data are used for the validation split. Multi-LeNet (**?**) is the network that all approaches aim to reach. We set $p = 16, \lambda = 5$ on the Multi-MNIST dataset, $p = 16, \lambda = 4$ on the Multi-Fashion dataset, and Multi-Fashion+MNIST dataset for PHN-HVI .

In Quadrant I, we evaluate every methods using 25 uniformly distributed preference vectors. The results are shown in Table 1 and Figure 5. The HV was calculated by reference point (2, 2). The Pareto front of PHN-HVI outperforms and covers the baselines entirely. The Pareto Front is generated by other techniques, especially PHN-LS, that are not widely dispersed. There are numerous solutions on the Pareto front of PHN-HVI that may be achieved that are superior to a single task.

Text Classification and Regression

In this investigation, we concentrated on the Drug Review dataset (**?**). This dataset comprises user evaluations of particular medications, details on pertinent ailments, and a user score that shows general satisfaction. We examine two tasks: (1) regression-based prediction of the drug's rating and (2) classify the patient's condition.

This dataset consists of 215063 samples. 10% of the data which has conditions with insufficient user feedback are removed. Following that are 100 condition labels and 188155 samples. The dataset has a ratio of 0.65/0.10/0.25 for train/-val/test. Target network is TextCNN (**?**). The hyperparameters for the PHN-HVI model are $p = 16, \lambda = 4$. Test rays is 25 evenly preference vectors. The reference point for hypervolume is (2, 2).

In this case, PHN-HVI has a greater hypervolume than previous techniques and still permits the Pareto front to move deeper. For COSMOS, due to the mapping from a preference vector r, to the huge dimensionality of the embedding feature, it does not converge.

Multi-Output Regression

To demonstrate the viability of our strategy in high-dimensional space, we conduct experiments on 2 datasets:

- **Jura** (**?**): In this experiment, the goal variables are zinc, cadmium, copper, and lead (4 tasks), whereas the predictive features are the other metals, the type of land use, the type of rock, and the position coordinates at 359 different locations. The dataset has a ratio of 0.65/0.15/0.20 for train/val/test.

- **SARCOS** (**?**): The goal is predict pertinent 7 joint torques (7 tasks) from a 21-dimensional input space (7 joint locations, 7 joint velocities, 7 joint accelerations). There are 4449 and 44484 examples on testing/training set. As validation set, 10% of the training data are used.

The target network in both experiments is a Multi-Layer Perceptron with 4 hidden layers containing 256 units. We set $p = 8, \lambda = 0.001$ for PHN-HVI on both two datasets. The reference point for calculating HV is $(1, 1, \ldots, 1)$. PHN-HVI outperforms all other baselines in terms of Hypervolume.

Ablation Study

Number of Rays p. Figure 6 demonstrates that the quality of the Pareto front increases with the number of rays, but up to a certain point, adding more rays no longer significantly improves the results. That means our framework doesn't require too many sampling rays to get a good performance.

Partition. As shown in Figure 7, partitioning algorithm makes it easier for the cosine similarity function and the HV function to cooperate and enhances PHN-HVI performance.

Cosine Similarity. The cosine similarity function is critical in the convergence of PHN-HVI and helps in the spread of the Pareto Front. In Figure 8, if λ is very large ($\lambda = 100$), Pareto Front is very widely dispersed, but it is quite shallow. If λ is very small ($\lambda = 0.1$), PHN-HVI can't generate Pareto Front. Therefore, selecting a suitable lambda that balances the HV function and the cosine similarity function is critical for the PHN-HVI to work effectively.

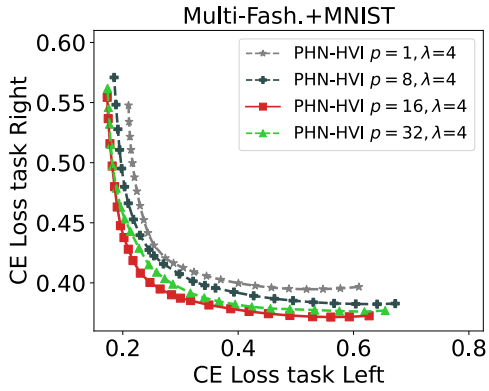

Figure 6: Performance of PHN-HVI when p varies

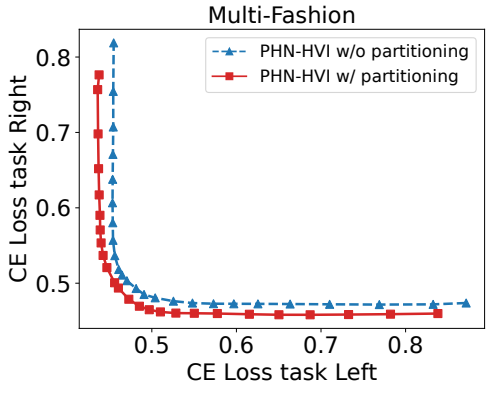

Figure 7: The effect of loss space partitioning

6 Conclusion and Future Work

In this paper, we propose PHN-HVI with Multi-Sample Hypernetwork, which utilizes a variety of trade-off vectors simultaneously, followed by hypervolume maximization to improve the PFL problem. This approach also opens up a wide range of potential research directions. On one hand,

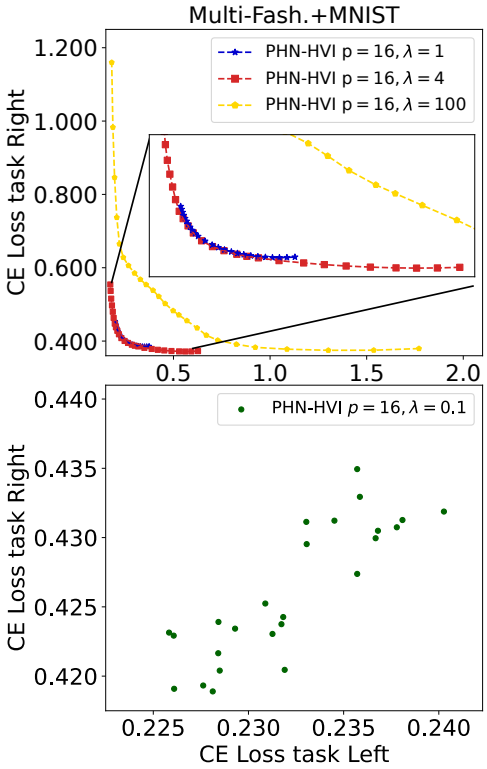

Figure 8: The impact of hyperpareter λ

it is necessary to investigate theoretically for which objective functions the hypernetwork-based PFL methods will guarantee the convergence. On the other hand, it is shown that hypernetwork-based PFL can not approximate well disconnected-Pareto fronts. Hence, the question of whether PFL may be solved effectively without hypernetwork is very crucial to consider.

Facilis molestiae in aspernatur voluptate, maxime corporis est harum nisi eos, esse eos itaque amet sit.Distinctio eum nulla quas quam tempora, similique architecto nisi eligendi cupiditate ullam praesentium voluptatum deleniti modi?Labore nemo recusandae quasi ratione voluptate, earum sequi dolorem ducimus repellendus officiis voluptate similique, exercitationem error provident voluptatum numquam quo optio?Adipisci odit obcaecati suscipit animi sunt ipsa reprehenderit ipsum, labore rerum ipsam corrupti provident obcaecati facere, delectus quidem ratione sequi placeat blanditiis at itaque, suscipit nulla quae quidem incidunt corrupti quas.Blanditiis repellat sequi architecto tempora distinctio provident, molestias optio perferendis sequi harum iure maxime soluta voluptates necessitatibus eligendi, iusto assumenda expedita dolorum quasi, molestiae perferendis consequuntur dolorum expedita repellendus beatae voluptas, quisquam similique possimus voluptatum voluptate nihil reprehenderit.Minima dolor labore deleniti nulla, sunt quos esse unde quam facere.Provident sint rem voluptatem, tempora mollitia repudiandae nobis ea dicta rerum vitae dolor, fugiat aliquid magnam labore aperiam iure doloremque praesentium accusamus iusto pariatur, velit reprehenderit sunt

architecto atque quod, nihil inventore delectus iste officiis sed eos?Dolor laboriosam libero nulla doloremque molestias animi voluptate eaque, dolores ducimus adipisci