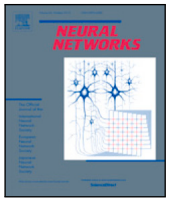
Neural Networks 179 (2024) 106571

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Full Length Article

A Hyper-Transformer model for Controllable Pareto Front Learning with Split Feasibility Constraints

Tran Anh Tuan, Nguyen Viet Dung, Tran Ngoc Thang *

Faculty of Mathematics and Informatics, Hanoi University of Science and Technology; Center for Digital Technology and Economy (BK Fintech), Hanoi University of Science and Technology, Hanoi, Vietnam

ARTICLE INFO

Keywords:
Multi-objective optimization
Controllable pareto front learning
Transformer
Hypernetwork
Split feasibility problem

ABSTRACT

Controllable Pareto front learning (CPFL) approximates the Pareto optimal solution set and then locates a non-dominated point with respect to a given reference vector. However, decision-maker objectives were limited to a constraint region in practice, so instead of training on the entire decision space, we only trained on the constraint region. Controllable Pareto front learning with Split Feasibility Constraints (SFC) is a way to find the best Pareto solutions to a split multi-objective optimization problem that meets certain constraints. In the previous study, CPFL used a Hypernetwork model comprising multi-layer perceptron (Hyper-MLP) blocks. Transformer can be more effective than previous architectures on numerous modern deep learning tasks in certain situations due to their distinctive advantages. Therefore, we have developed a hyper-transformer (Hyper-Trans) model for CPFL with SFC. We use the theory of universal approximation for the sequence-to-sequence function to show that the Hyper-Trans model makes MED errors smaller in computational experiments than the Hyper-MLP model.

1. Introduction

Multi-objective optimization (MOO), an advanced solution for modern optimization problems, is increasingly driven by the need to find optimal solutions in real-world situations with multiple criteria. Addressing the complex trade-offs inherent in decision-making problems resolves the challenges of simultaneously optimizing conflicting objectives on a shared optimization variable set. The advantages of MOO have been recognized in several scientific domains, including chemistry (Cao et al., 2019), biology (Lambrinidis & Tsantili-Kakoulidou, 2021), and finance, specifically investing (Vuong & Thang, 2023). Specifically, its recent accomplishment in deep multitask learning (Sener & Koltun, 2018) has attracted attention.

Split Feasibility Problem (SFP) is an idea that Censor and Elfving (1994) initially proposed. It requires locating a point in a nonempty closed convex subset in one space whose image is in another nonempty closed convex subset in the image space when subjected to a particular operator. While projection algorithms that are frequently employed have been utilized to solve SFP, they face challenges associated with computation, convergence on multiple sets, and strict conditions. The SFP is used in many real-world situations, such as signal processing, image reconstruction (Byrne, 2003; Stark, Yang, & Yang, 1998), and intensity-modulated radiation therapy (Brooke, Censor, & Gibali, 2021; Censor, Elfving, Kopf, & Bortfeld, 2005).

It is noteworthy that our work is the first to consider the connection between SFP and CPFL. Specifically, we can consider the solution set of SFP to be the Pareto optimal solution set of the corresponding MOP. From this, we have a multi-objective optimization problem with a split feasibility constraint. This problem has not been studied before because the Pareto solution set is usually a non-convex set with a complex structure (Kim & Thang, 2013). Previous methods tackled the entire Pareto front; one must incur an impracticably high cost due to the exponential growth in the number of solutions required in proportion to the number of objectives. Several proposed algorithms, such as evolutionary and genetic algorithms, aim to approximate the Pareto front partially (Jangir, Heidari, & Chen, 2021). Despite these algorithms' potential, only small-scale tasks (Murugan, Kannan, & Baskar, 2009) can be used in practice. Moreover, these methods limit adaptability because the decision-maker cannot flexibly adjust priorities in real-time. After all, the corresponding solutions are only sometimes readily available and must be recalculated for optimal performance (Lin, Zhen, Li, Zhang, & Kwong, 2019; Mahapatra & Rajan, 2021; Momma, Dong, & Liu, 2022). Based on the self-attention mechanism (Vaswani et al., 2017) to clarify the trade-offs between objectives in multi-objective optimization problems and the theory of sequence-to-sequence models behind the transformer (Jiang, Li, Li, & Wang, 2023; Yun, Bhojanapalli, Rawat, Reddi, & Kumar, 2019). We developed a Hyper-transformer

* Corresponding author.
E-mail addresses: tuan.ta222171m@sis.hust.edu.vn (T.A. Tuan), dung.nv232215m@sis.hust.edu.vn (N.V. Dung), thang.tranngoc@hust.edu.vn (T.N. Thang).

https://doi.org/10.1016/j.neunet.2024.106571
Received 6 February 2024; Received in revised form 6 July 2024; Accepted 23 July 2024
Available online 26 July 2024

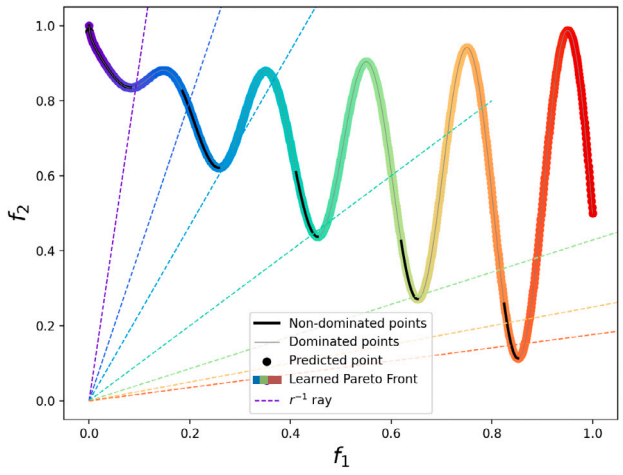


Fig. 1. Pareto Front Learning.

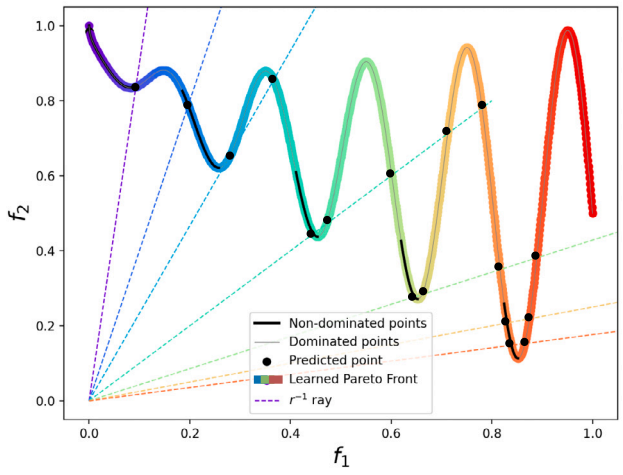


Fig. 2. Controllable Pareto Front Learning.

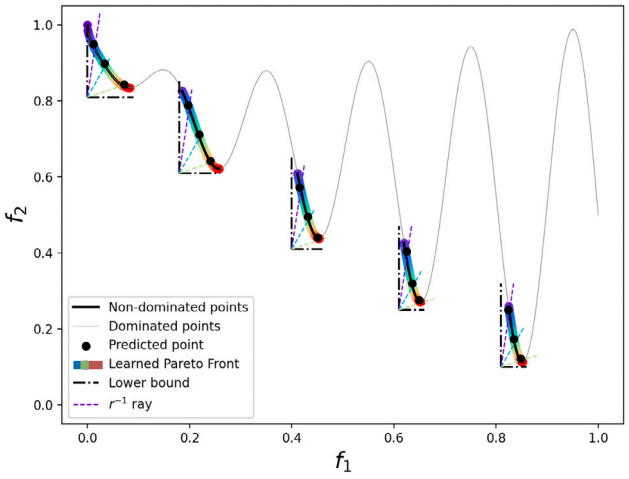


Fig. 3. Controllable Pareto Front Learning with Split Feasibility Constraints.

model to solve the Controllable Pareto Front Learning problem with Split Feasibility Constraints.

Our main contributions include:

- In this study, we express a split multi-objective optimization problem. From there, we focus on solving the controllable Pareto front learning problem with split feasibility constraints based on scalarization theory and the split feasibility problem. In reality, when decision-makers want their goals to be within the area limited by bounding boxes, this allows them to control resources and provide more optimal criteria for the Pareto optimal solution set.
- We propose a novel hypernetwork architecture based on a transformer encoder block for the controllable Pareto front learning problem with split feasibility constraints. Our proposed model shows superiority over MLP-based designs for multi-objective optimization and multi-task learning problems.
- We also integrate joint input and a mixture of expert architectures to enhance the hyper-transformer network for learning disconnected Pareto front. This helps bring great significance to promoting other research on the controllable disconnected Pareto front of the hypernetwork.

Summarizing, the remaining sections of the paper are structured in the following manner: Section 2 will summarize the shortcomings of existing work. Section 3 will provide an overview of the foundational knowledge required for multi-objective optimization. Section 4 presents the optimization problem over the Pareto set with splitting feasibility box constraints. Section 5 describes the optimization problem over the Pareto set as a controllable Pareto front learning problem using a hypernetwork, and we also introduce a hypernetwork based on the Transformer model (Hyper-Transformer). Section 6 explains the two fundamental models used in the Hyper-Transformer architecture within Disconnected Pareto Front Learning. In Section 7, we apply the proposed techniques to Multi-task learning, and Section 8 will detail the experimental synthesis, present the results, analyze the performance of the proposed model, and implement additional experiments. The last section addresses the findings and potential future endeavors.

2. Related works

Researchers have raised recent inquiries regarding the approximability of the solution to the priority vector. While prior research has suggested using a hypernetwork to approximate the entire Pareto front (Hoang, Le, Tuan, & Thang, 2023; Lin, Yang, Zhang, & Kwong, 2020; Navon, Shamsian, Chechik, & Fetaya, 2020), Pareto front learning (PFL) algorithms are incapable of generating solutions that precisely match the reference vectors input into the hypernetwork. The paper on controllable Pareto front learning with complete scalarization functions (Tuan, Hoang, Le, & Thang, 2023) explains how hypernetworks create precise connections between reference vectors and the corresponding Pareto optimal solution. The term "controllable" refers to the adjustable trade-off between objectives with respect to the reference vector. In such a way, one can find an efficient solution that satisfies his or her desired trade-off.

In Fig. 1, Pareto Front Learning uses a hypernetwork to approximate the entire Pareto front, including dominated points. In Fig. 2, Controllable Pareto Front Learning with Completed Scalarization Function uses a single hypernetwork model, mapping any given preference vector to its corresponding solution on the Pareto front; these solutions may not be unique. However, in Fig. 3, Controllable Disconnected Pareto Front Learning with Split Feasibility Constraints by a Robust Hypernetwork helps avoid dominated points.

Before our research, Raychaudhuri et al. (2022) exploited hypernetworks to achieve a controllable trade-off between task performance and network capacity in multi-task learning. The network architecture, therefore, can dynamically adapt to the compute budget variation. Chen et al. (2023) suggests a controllable multi-objective reranking (CMR) method that uses a hypernetwork to create parameters

for a re-ranking model based on different preference weights. In this way, CMR can adapt the preference weights according to the changes in the online environment without any retraining. These approaches, however, only apply to the multi-task learning scenario and require a complicated training paradigm. Moreover, they do not guarantee the exact mapping between the preference vector from user input and the optimal Pareto point.

Primarily, problems involving entirely connected Pareto fronts are the focus of the current research. Unfortunately, this is unrealistic in real-world optimization scenarios (Ishibuchi, He, & Shang, 2019), whereas the performance can significantly deteriorate when the PF consists of disconnected segments. If we use the most recent surrogate model's regularity information, we can see that the PFs of real-world applications are often shown as disconnected, incomplete, degenerated, and badly scaled. This is partly because the relationships between objectives are often complicated and not linear. Chen and Li (2023) proposed a data-driven EMO algorithm based on multiple-gradient descent to explore promising candidate solutions. It consists of two distinctive components: the MGD-based evolutionary search and the Infill criterion. While the D2EMO/MGD method demonstrated strong performance on specific benchmarking challenges involving unconnected PF segments, it needs more computational efficiency and flexibility to meet real-time system demands. In our research, we developed two different neural network architectures to help quickly learn about disconnected PF problems with split feasibility constraints.

3. Preliminaries

Multi-objective optimization aims to find $\mathbf{x} \in X$ to optimize m objective functions:

$$\min_{\mathbf{x}\in X} \mathcal{F}(\mathbf{x}), \tag{MOP}$$

where $\mathcal{F}(\cdot) : X \to Y \subset \mathbb{R}^m, \mathcal{F}(\mathbf{x}) = \{f_1(\mathbf{x}), \dots, f_m(\mathbf{x})\}$, $X \subset \mathbb{R}^n$ is nonempty convex set, and objective functions $f_i(\cdot) : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, m$ are convex functions and bounded below on X. We denote $Y := \mathcal{F}(X) = \{\mathbf{y} \in \mathbb{R}^m | \exists \mathbf{x} \in \mathbb{R}^n, \mathcal{F}(\mathbf{x}) = \mathbf{y}\}$ the outcome set or the value set of Problem (MOP).

Definition 3.1 (*Dominance*)**.** A solution $\mathbf{x}_1$ dominates $\mathbf{x}_2$ if $f_i(\mathbf{x}_1) \leq f_i(\mathbf{x}_2), \forall i$ and $f_i(\mathbf{x}_1) \neq f_i(\mathbf{x}_2)$. Denote $\mathcal{F}(\mathbf{x}_1) \prec \mathcal{F}(\mathbf{x}_2)$.

Definition 3.2 (*Pareto Optimal Solution*)**.** A solution $\mathbf{x}_1$ is called Pareto optimal solution (efficient solution) if $\nexists \mathbf{x}_2 : \mathcal{F}(\mathbf{x}_2) \preceq \mathcal{F}(\mathbf{x}_1)$.

Definition 3.3 (*Weakly Pareto Optimal Solution*)**.** A solution $\mathbf{x}_1$ is called weakly Pareto optimal solution (weakly efficient solution) if $\nexists \mathbf{x}_2 : \mathcal{F}(\mathbf{x}_2) \prec \mathcal{F}(\mathbf{x}_1)$.

Definition 3.4 (*Pareto Stationary*)**.** A point $\mathbf{x}^*$ is called Pareto stationary (Pareto critical point) if $\nexists d \in X : \langle J\mathcal{F}(\mathbf{x}^*), d\rangle < 0$ or $\forall d \in X : \langle J\mathcal{F}(\mathbf{x}^*), d\rangle \not< 0$, corresponding:

$$\max_{i=1,\dots,m} \nabla f_i(\mathbf{x}^*)^\top d \geq 0, \quad \forall d \in X,$$

where $J\mathcal{F}(\mathbf{x}^*) = \left[\nabla f_1(\mathbf{x}^*)^T, \dots, \nabla f_m(\mathbf{x}^*)^T\right]^T$ is Jacobian matrix of $\mathcal{F}$ at $\mathbf{x}^*$.

Definition 3.5 (*Pareto Set and Pareto Front*)**.** The set of Pareto optimal solutions is called the Pareto optimal solution set, denoted by X_E, and the corresponding images in objectives space are Pareto outcome set $Y_E := \{\mathbf{y} \in \mathbb{R}^m | \mathbf{y} = \mathcal{F}(\mathbf{x}) \text{ for some } x \in X_E\}$ or Pareto front (PF_E). Similarly, we can define the weakly Pareto set X_{WE} and weakly Pareto outcome set Y_{WE}.

Proposition 3.1. *$\mathbf{x}^*$ is Pareto optimal solution to Problem (MOP) $\Leftrightarrow$ $\mathbf{x}^*$ is Pareto stationary point.*

Definition 3.6 (*Mangasarian, 1994*)**.** The differentiable function $f : \mathbb{R}^m \to \mathbb{R}$ is said to be

- convex on X if for all $\mathbf{x}_1, \mathbf{x}_1 \in X$, $\lambda \in [0, 1]$, it holds that
$$f(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) \leq \lambda f(\mathbf{x}_1) + (1-\lambda) f(\mathbf{x}_2).$$
- pseudoconvex on X if for all $\mathbf{x}_1, \mathbf{x}_2 \in X$, it holds that
$$f(\mathbf{x}_2) < f(\mathbf{x}_1) \Rightarrow \langle \nabla f(\mathbf{x}_1), \mathbf{x}_2 - \mathbf{x}_1\rangle < 0.$$

Let f be a numerical function defined on some open X set in $\mathbb{R}^n$, let $\overline{\mathbf{x}} \in X$, and let f be differentiable at $\overline{\mathbf{x}}$. If f is convex at $\overline{\mathbf{x}}$, then f is pseudoconvex at $\overline{\mathbf{x}}$, but not conversely (Mangasarian, 1994).

Definition 3.7 (*Luc, 2005*)**.** A function φ is specified on convex set $X \subset \mathbb{R}^n$, which is called:

1. nondecreasing on X if $\mathbf{x}_1 \succeq \mathbf{x}_2$ then $\varphi(\mathbf{x}_1) \geq \varphi(\mathbf{x}_2)$, $\forall \mathbf{x}_1, \mathbf{x}_2 \in X$.
2. weakly increasing on X if $\mathbf{x}_1 \succ \mathbf{x}_2$ then $\varphi(\mathbf{x}_1) \geq \varphi(\mathbf{x}_2)$, $\forall \mathbf{x}_1, \mathbf{x}_2 \in X$.
3. monotonically increasing on X if $\mathbf{x}_1 \succ \mathbf{x}_2$ then $\varphi(\mathbf{x}_1) > \varphi(\mathbf{x}_2)$, $\forall \mathbf{x}_1, \mathbf{x}_2 \in X$.

The Pareto front's structure and optimal solution set of Problem (MOP) have been investigated by numerous authors in the field (Helbig, 1990; Luc, 1989; Naccache, 1978; Xunhua, 1994). In certain situations, Y_E is weakly connected or connected (Benoist, 2001; Luc, 1989). Connectedness and contractibility are noteworthy topological properties of these sets due to their ability to enable an uninterrupted transition from one optimal solution to another along only optimal alternatives and their assurance of numerical algorithm stability when subjected to limiting processes.

4. Multi-objective optimization problem with split feasibility constraints

4.1. Split multi-objective optimization problem

In 1994, Censor and Elfving (1994) first introduced the Split Feasibility Problem (SFP) in finite-dimensional Hilbert spaces to model inverse problems arising from phase retrievals and medical image reconstruction. In this setting, the problem is stated as follows:

$$\text{Find} \quad \mathbf{x}^* \in C : \mathcal{F}(\mathbf{x}^*) \in Q, \tag{SFP}$$

where C is a convex subset in $\mathbb{R}^n$, Q is a convex subset in $\mathbb{R}^m$, and a smooth linear function $\mathcal{F}(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$. The classical linear version of the split feasibility problem takes $\mathcal{F}(x) = Ax$ for some $m \times n$ matrix A (Censor & Elfving, 1994). Other typical examples of the constraint set Q are defined by the constraints $\mathcal{F}(x) = b, \|\mathcal{F}(x) - b\| \leq r$, or $c \leq \mathcal{F}(x) - b \leq d$, where $b, c, d, r \in \mathbb{R}^m$.

Some solution methods were studied for Problem (SFP) when C and/or Q are solution sets of some other problems such as fixed point, optimization, variational inequality, equilibrium (Anh & Muu, 2016; Byrne, 2002; Censor, Gibali, & Reich, 2012; López, Martín-Márquez, Wang, & Xu, 2012). These works focus on the assumptions when C is a convex set or $\mathcal{F}$ is linear (Godwin, Izuchukwu, & Mewomo, 2023; Xu, Chi, Yang, & Lange, 2018; Yen, Huyen, & Muu, 2019). According to our survey, there has been no research on Problem (SFP) where the set C is the solution set of a multi-objective optimization problem.

In the paper, we study Problem (SFP) where C is the weakly Pareto optimal solution set of Problem (MOP), that is

$$\text{Find} \quad \mathbf{x}^* \in X_{WE} : \mathcal{F}(\mathbf{x}^*) \in Q \tag{SMOP}$$
$$\text{with} \quad X_{WE} := \text{Argmin}\{\mathcal{F}(\mathbf{x}) | \mathbf{x} \in X\}.$$

This problem is called *a split multi-objective optimization problem*. This problem helps us find the Pareto optimal solution that satisfies the constraint $\mathcal{F}(x) \in Q$, which means the corresponding non-dominated point in the Pareto front is located within a given region Q. In a simple

case, the set Q can be a box or a sphere in the image space $\mathbb{R}^p$. In the case where Q is a box, the decision-maker aims to find Pareto optimal solutions such that the objectives are achieved within a specified range of values. For example, consider a bi-objective optimization problem of maximizing profit and minimizing risk. The box constraint Q corresponds to ensuring that the risk is within an acceptable range and the profit achieved falls within a certain range.

It is well known that X_{WE} is, in general, a non-convex set, even in the special case when X is a polyhedron and $\mathcal{F}$ is linear on $\mathbb{R}^n$ (Kim & Thang, 2013). Then the set C of Problem (SFP) is non-convex even when (MOP) is a linear multi-objective optimization problem. Therefore, unlike previous studies, in this study, we consider the more challenging case of Problem (SFP) where C is a non-convex set and $\mathcal{F}$ is nonlinear. This challenge is overcome using an outcome space approach to transform the non-convex form into a convex form, in which the constraint sets of Problem (SFP) are convex sets. This will be presented in Section 4.2 below.

4.2. Optimizing over the solution set of problem (SMOP)

MOO aims to find Pareto optimal solutions corresponding to different trade-offs between objectives (Ehrgott, 2005). Optimizing over the Pareto set in multi-objective optimization allows us to make informed decisions when dealing with multiple, often conflicting, objectives. It is not just about finding feasible solutions but also about understanding and evaluating the trade-offs between different objectives to select the most appropriate solution based on specific criteria or preferences. In a similar vein, we consider optimizing over the Pareto set of Problem (SMOP) as follows:

$$\min_{\mathbf{x}} \ S(\mathcal{F}(\mathbf{x})) \qquad \text{(SP)}$$
$$\text{s.t. } \mathbf{x} \in X_{WE} : \mathcal{F}(\mathbf{x}) \in Q,$$

where the function $S(\cdot) : Y \to \mathbb{R}$ is a monotonically increasing function and pseudoconvex on Y. Recall that Y is the outcome set of X through the function $\mathcal{F}$.

Following the outcome-space approach, the reformulation of Problem (SP) is given by:

$$\min \ S(\mathbf{y}) \qquad \text{(OSP)}$$
$$\text{s.t. } \mathbf{y} \in Y_{WE} : \mathbf{y} \in Q,$$

where Y_{WE} is the weakly Pareto outcome set of Problem (MOP).

Proposition 4.1. *Problem (SP) and Problem (OSP) are equivalent, i.e., if $\mathbf{x}^*$ is the optimal solution of Problem (SP) then $\mathbf{y}^* = \mathcal{F}(\mathbf{x}^*)$ is the optimal solution of Problem (OSP); conversely, if $\mathbf{y}^*$ is the optimal solution of Problem (OSP) then $\mathbf{x}^* \in X$ such that $\mathcal{F}(\mathbf{x}^*) \leq \mathbf{y}^*$ and $\mathcal{F}(\mathbf{x}^*) \in Q$ is the optimal solution of Problem (SP).*

Proof. Indeed, if $\mathbf{x}^* \in X_{WE}$ is a global optimal solution to Problem (SP), then any $\mathbf{x} \in X_{WE} : \mathcal{F}(\mathbf{x}) \in Q$ such that $S(\mathcal{F}(\mathbf{x}^*)) \leq S(\mathcal{F}(\mathbf{x}))$. We imply $S(\mathbf{y}^*) \leq S(\mathbf{y})$ with $\forall \mathbf{y} \in Y_{WE} : \mathbf{y} \in Q$, and $\mathbf{y}^* = \mathcal{F}(\mathbf{x}^*)$ belongs to the feasible domain of Problem (OSP). Hence, $\mathbf{y}^*$ is the optimal solution of Problem (OSP).

On the contrary, if $\mathbf{y}^* \in Y_{WE}$ is a global optimal solution to Problem (OSP), then any $\mathbf{x}^* \in X$ such that $\mathcal{F}(\mathbf{x}^*) \leq \mathbf{y}^*$. We imply $S(\mathcal{F}(\mathbf{x}^*)) \leq S(\mathbf{y}^*) \leq S(\mathbf{y})$ with $\forall \mathbf{y} \in Y_{WE} : \mathbf{y} \in Q$, i.e, and $S(\mathcal{F}(\mathbf{x}^*)) \leq S(\mathcal{F}(\mathbf{x}))$. From the definition, we have $\mathbf{x}^*$ as a global optimal solution to Problem (SP). □

Let $Y^+ = Y + \mathbb{R}^m_+ = \{\mathbf{y} \in \mathbb{R}^m | \mathbf{y} \geq \mathbf{q} \text{ with } \mathbf{q} \in Y\}$. When X is a convex set and $\mathcal{F}$ is a nonlinear function, the image set $Y = \mathcal{F}(X)$ is not necessarily a convex set. Therefore, instead of considering the set Y, we consider the set Y^+, which is an effective equivalent set (i.e., the set of effective points of Y and Y^+ coincide), and Y^+ has nicer properties; for example, Y^+ is a convex set. This is illustrated in Proposition 4.2.

Besides, we also define a set $G \subset \mathbb{R}^m$ is called normal if for any two points $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^m$ such that $\mathbf{x}' \leq \mathbf{x}$, if $\mathbf{x} \in G$, then $\mathbf{x}' \in G$. Similarly, a set $H \subset \mathbb{R}^m$ is called reverse normal if $\mathbf{x}' \geq \mathbf{x} \in H$ implies $\mathbf{x}' \in H$.

Proposition 4.2 (Kim & Thang, 2013)**.** *We have:*

(i) $Y_{WE} = Y^+_{WE} \cap Y$;
(ii) $\partial Y^+ = Y^+_{WE}$;
(iii) Y^+ *is a closed convex set and is a reverse normal set.*

Hence, we transform Problem (OSP) into:

$$\min \ S(\mathbf{y}) \qquad \text{(OSP}^+\text{)}$$
$$\text{s.t. } \mathbf{y} \in Y^+_{WE} : \mathbf{y} \in Q.$$

The equivalence of problems (OSP) and (OSP$^+$) is shown in the following Proposition 4.3.

Proposition 4.3. *If $\mathbf{y}^*$ is the optimal solution of Problem (OSP), then $\mathbf{y}^*$ is the optimal solution of Problem (OSP$^+$). Conversely, if $\mathbf{y}^*$ is the optimal solution of Problem (OSP$^+$) and $\mathbf{q}^* \in Y_{WE} \cap Q$ such that $\mathbf{y}^* \geq \mathbf{q}^*$ then $\mathbf{q}^*$ is the optimal solution of Problem (OSP).*

Proof. If $\mathbf{y}^*$ is the optimal solution of Problem (OSP), then $S(\mathbf{y}^*) \leq S(\mathbf{y}), \forall \mathbf{y} \in Y_{WE} \cap Q$ and $\mathbf{y}^* \in Y_{WE} \cap Q$. With each of $\overline{\mathbf{y}} \in Y^+_{WE} \cap Q$, following the definition of Y_{WE^+}, there exists $\mathbf{y} \in Y_{WE} \cap Q$ such that $\overline{\mathbf{y}} \geq \mathbf{y}$. S is a monotonically increasing function on Y, so $S(\overline{\mathbf{y}}) \geq S(\mathbf{y})$. Hence $S(\mathbf{y}^*) \leq S(\overline{\mathbf{y}}), \forall \overline{\mathbf{y}} \in Y^+_{WE} \cap Q$. Moreover, $\mathbf{y}^* \in Y_{WE} \cap Q$ means $\mathbf{y}^* \in Y^+_{WE} \cap Q$. We imply that $\mathbf{y}^*$ is the optimal solution of Problem (OSP$^+$).

Conversely, if $\mathbf{y}^*$ is the optimal solution of Problem (OSP$^+$), then $S(\mathbf{y}^*) \leq S(\mathbf{y}), \forall \mathbf{y} \in Y^+_{WE} \cap Q$. Assume that there exists $\mathbf{q}^* \in Y_{WE} \cap Q$ such that $\mathbf{y}^* \geq \mathbf{q}^*$. S is a monotonically increasing function on Y, then $S(\mathbf{q}^*) \leq S(\mathbf{y}^*) \leq S(\mathbf{y})$. With each of $\mathbf{y} \in Y_{WE} \cap Q$, then $\mathbf{y} \in Y^+_{WE} \cap Q$. Hence $S(\mathbf{q}^*) \leq S(\mathbf{y}), \forall \mathbf{y} \in Y_{WE} \cap Q$, i.e. $\mathbf{q}^*$ is the optimal solution of Problem (OSP). □

The problem (OSP$^+$) is a difficult problem because normally, the set Y^+_{WE} is a non-convex set. Thanks to the special properties of the objective functions S and Y^+, we can transform the problem (OSP$^+$) into an equivalent problem, where the constraint set of this problem is a convex set, as follows:

$$\min \ S(\mathbf{y}) \qquad \text{(}\overline{\text{OSP}}\text{)}$$
$$\text{s.t. } \mathbf{y} \in Y^+ \cap Q,$$

with the explicit form

$$\min_{(\mathbf{x},\mathbf{y})} \ S(\mathcal{F}(\mathbf{x})) \qquad \text{(}\overline{\text{ESP}}\text{)}$$
$$\text{s.t. } \mathbf{x} \in X, \mathbf{y} \in Q$$
$$\mathcal{F}(\mathbf{x}) \leq \mathbf{y}.$$

Proposition 4.4. *Assume $Q \subset \mathbb{R}^m_+$ is a normal set. The optimal solution sets of Problems (OSP$^+$) and ($\overline{\text{OSP}}$) are identical.*

Proof. From Proposition 11 (Tuy, 2000), the minimum of S over $Y^+ \cap Q$, if it exists, is attained on $\partial Y^+ \cap Q$. Assume $\mathbf{y}^*$ is the optimal solution of Problem ($\overline{\text{OSP}}$), then $\mathbf{y}^* \in \partial Y^+ \cap Q$. Use Proposition 4.2, which implies $\mathbf{y}^* \in Y^+_{WE} \cap Q$. Therefore, the optimal solution sets of Problems (OSP$^+$) and ($\overline{\text{OSP}}$) are identical. □

Proposition 4.5. *Problem ($\overline{\text{OSP}}$) is a pseudoconvex programming problem with respect to $\mathbf{y}$, and Problem ($\overline{\text{ESP}}$) is a pseudoconvex programming problem with respect to $(\mathbf{x}, \mathbf{y})$.*

Proof. Because each $f_i(\mathbf{x})$ is a convex function on a nonempty convex set X, and Y^+ is a full-dimension closed convex set. Moreover, S is a

monotonically increasing function and pseudoconvex on Y. Therefore, Problem (OSP) is a pseudoconvex programming problem with respect to $\mathbf{y}$.

If $f_i(\mathbf{x})$ are convex functions, then $\mathcal{F}(\mathbf{x}) - \mathbf{y}$ is convex constraint, and $\mathcal{S}(\mathcal{F}(\mathbf{x}))$ is a convex function on X, Y. Hence, $\mathcal{S}(\mathcal{F}(\mathbf{x}))$ is a convex function with respect to $(\mathbf{x}, \mathbf{y})$. Furthermore, X, Q are nonempty convex sets in $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. Problem (ESP) is a pseudoconvex programming problem with respect to $(\mathbf{x}, \mathbf{y})$. □

From Proposition 4.5, Problem (ESP) is a pseudoconvex programming problem. Therefore, each local minimization solution is also a global minimization solution (Mangasarian, 1994). So, we can solve it using gradient descent algorithms, such as Thang and Hai (2022), or neurodynamics methods, such as Bian, Ma, Qin, and Xue (2018), Liu, Wang, and Qin (2022), Xu, Chai, Qin, Wang, and Feng (2020).

These methods solely assist in locating the Pareto solution associated with the provided reference vector. In numerous instances, however, we are concerned with whether the resulting solution is controllable and whether we are interested in more than one predefined direction because the trade-off is unknown before training or the decision-makers decisions vary. Designing a model that can be applied at inference time to any given preference direction, including those not observed during training, continues to be a challenge. Furthermore, the model should be capable of dynamically adapting to changes in decision-maker references. This issue is referred to as controllable Pareto front learning (CPFL) and will be elaborated upon in the following section.

5. Controllable pareto front learning with split feasibility constraints

Tuan et al. (2023) was the first to introduce Controllable Pareto Front Learning. They train a single hypernetwork to produce a Pareto solution from a collection of input reference vectors using scalarization problem theory. Our study uses a weighted Chebyshev function based on the coordinate transfer method to find Pareto solutions that align with how DM's preferences change over time with $\mathcal{S}(\mathcal{F}(\mathbf{x}), \mathbf{r}) := \max_{i=1,\dots,m} \left\{ r_i \left(f_i(\mathbf{x}) - \mathbf{a}_i \right) \right\}$. Moreover, we also consider $Q = Q_1 \times Q_2 \times \dots \times Q_m$ where Q_i is a box constraint such that $f_i(\mathbf{x}) \in [\mathbf{a}_i, \mathbf{b}_i], \mathbf{a}_i \geq 0, i = 1, \dots, m$. From the definition of the normal set, then Q is a normal set. Therefore, the controllable Pareto front learning problem is modeled in the following manner by combining the properties of split feasibility constraints:

$$\begin{aligned} \phi^* = \arg\min_{\phi} \mathbb{E}_{\mathbf{r} \sim Dir(\alpha)} \left[\max_{i=1,\dots,m} \left\{ \mathbf{r}_i \left(f_i(h(\mathbf{r}, \phi)) - \mathbf{a}_i \right) \right\} \right] \quad \text{(LP)} \\ \text{s.t. } h(\mathbf{r}, \phi) \in X \\ \mathcal{F}(h(\mathbf{r}, \phi)) \leq \mathbf{b}, \end{aligned}$$

where $h : \mathcal{P} \times \mathbb{R}^n \to \mathbb{R}^n$ is a hypernetwork, and $Dir(\alpha)$ is Dirichlet distribution with concentration parameter $\alpha > 0$.

Theorem 5.1. *If $\mathbf{x}^*$ is an optimal solution of Problem (SMOP), then there exists a reference vector $\mathbf{r}$ $(\mathbf{r}_i > 0)$ such that $\mathbf{x}^*$ is also an optimal solution of Problem (LP).*

The pseudocode that solves Problem (LP) is presented in Algorithm 1. In contrast to the algorithm proposed by Tuan et al. (2023), our approach incorporates upper bounds $\mathbf{b}$ and lower bounds $\mathbf{a}$ during model training by regularizing the objective function following the paper by Jiang and Yang (2017)

$$f_i - \mathbf{a}_i = \frac{f_i - \mathbf{a}_i}{\mathbf{b}_i - \mathbf{a}_i}.$$

The model can weed out non-dominated Pareto solutions and solutions that do not meet the split feasibility constraints by adding upper-bounds constraints during post-processing. Moreover, we propose building a hypernetwork based on the Transformer architecture instead of the MLP architecture used in other studies (Hoang et al., 2023; Navon et al., 2020; Tuan et al., 2023). Take advantage of the universal approximation theory of sequence-to-sequence function and the advantages of Transformer's Attention Block over traditional CNN or MLP models (Cordonnier, Loukas, & Jaggi, 2019; Li, Chen, He, & Hsieh, 2021).

5.1. Hypernetwork-based multilayer perceptron

We define a Hypernetwork-Based Multilayer Perceptron (Hyper-MLP) h is a function of the form:

$$\begin{aligned} \mathbf{x}_\mathbf{r} &= h_{\text{mlp}}(\mathbf{r}; [\boldsymbol{W}, \boldsymbol{b}]) \quad \text{(Hyper-MLP)} \\ &= W^k \cdot \sigma \left(W^{k-1} \dots \sigma \left(W^1 \mathbf{a} + b^1 \right) + b^{k-1} \right), \end{aligned}$$

with weights $W^i \in \mathbb{R}^{k_{i+1} \times k_i}$ and biases $b^i \in \mathbb{R}^{k_{i+1}}$, for some $k_i \in \mathbb{N}$. In addition, $\phi = [\boldsymbol{W}, \boldsymbol{b}]$ accumulates the parameters of the hypernetwork. The function σ is a non-linear activation function, typically ReLU, logistic function, or hyperbolic tangent. An illustration is shown in Fig. 4a.

Theorem 5.2 (*Cybenko, 1989*)**.** *Let σ be any continuous sigmoidal function. Then finite sums of the form*

$$g(x) = \sum_{j=1}^{N} \alpha_j \sigma \left(y_j^{\mathrm{T}} x + \theta_j \right),$$

are dense in $C(I_n)$. In other words, given any $f \in C(I_n)$ and $\varepsilon > 0$, there is a sum, $g(x)$, of the above form, for which

$$|g(x) - f(x)| < \varepsilon \quad \text{for all} \quad x \in I_n.$$

It has been known since the 1980s (Cybenko, 1989; Hornik, Stinchcombe, & White, 1989) that feed-forward neural nets with a single hidden layer can approximate essentially any function if the hidden layer is allowed to be arbitrarily wide. Such results hold for a wide variety of activations, including ReLU. However, part of the recent renaissance in neural nets is the empirical observation that deep neural nets tend to achieve greater expressivity per parameter than their shallow cousins.

Theorem 5.3 (*Hanin & Sellke, 2017*)**.** *For every continuous function $f : [0,1]^{d_{in}} \to \mathbb{R}^{d_{out}}$ and every $\varepsilon > 0$ there is a Hyper-MLP h with ReLU activations, input dimension d_{in}, output dimension d_{out}, hidden layer widths $d_{in} = d_1, d_2, \dots, d_k, d_{k+1} = d_{out}$ that ε-approximates f :*

$$\sup_{\mathbf{x} \in [0,1]^{d_{in}}} \|f(\mathbf{x}) - h(\mathbf{r})\| \leq \varepsilon.$$

5.2. Hypernetwork-based transformer block

The reference vectors indicate the anticipated outcomes or significance of the objectives. The reference vectors and weights have similar compositional purposes but possess distinct physical interpretations and exert diverse influences on the search process (Wang, Olhofer, & Jin, 2017). Formally, in order for a solution to be considered Pareto optimum, it must satisfy the condition that for any two objectives, if the reference value for one objective is more than the reference value for the other objective, then the corresponding objective value must be less than the other objective value. The interdependence of the $\mathbf{r}_i$ guarantees that the sequence-to-sequence model, equipped with the attention mechanism, may effectively learn this connection and identify a Pareto optimal solution that meets the desired criteria.

The paper (Yun et al., 2019) gave a clear mathematical explanation of contextual mappings and showed that multi-head self-attention layers can correctly calculate contextual mappings for input sequences. They show that the capacity to calculate contextual mappings and the value mapping capability of the feed-forward layers allows transformers to serve as universal approximators for any permutation equivariant

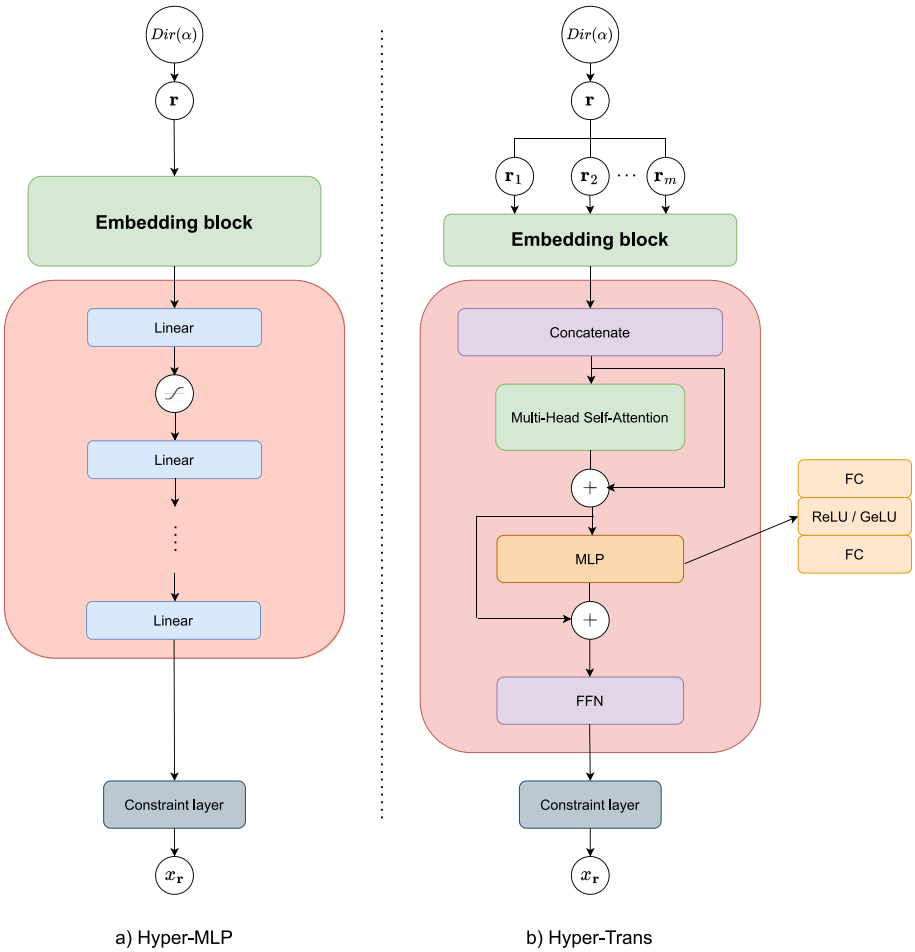


Fig. 4. Hyper-MLP (*left*) receives an input reference vector, Hyper-Trans (*right*) receives each coordinate of the input reference vector and outputs the corresponding Pareto optimal solution.

sequence-to-sequence function. There are well-known results for approximation, like how flexible Transformer networks are at it (Yun et al., 2019). Its sparse variants can also universally approximate any sequence-to-sequence function (Yun et al., 2020).

A transformer block is a sequence-to-sequence function mapping $\mathbb{R}^{d\times n}$ to $\mathbb{R}^{d\times n}$. It consists of two layers: a multi-head self-attention layer and a token-wise feed-forward layer, with both layers having a skip connection. More concretely, for an input $\mathbf{r} \in \mathbb{R}^{d\times m}$ consisting of d-dimensional embeddings of m tasks, a Transformer block with multiplicative or dot-product attention (Luong, Pham, & Manning, 2015) consists of the following two layers. We propose a hypernetwork-based transformer block (Hyper-Trans) as follows:

$$\mathbf{x}_\mathbf{r} = h_{\text{trans}}(\mathbf{r};\phi) = \text{MultiHeadAttn}(\mathbf{r}) + \text{MLP}(\mathbf{r}), \qquad \text{(Hyper-Trans)}$$

with:

$$\text{MultiHeadAttn}(\mathbf{r}) = \mathbf{r} + \sum_{i=1}^{h} \boldsymbol{W}^i_O \boldsymbol{W}^i_V \mathbf{r} \cdot \sigma\left[\left(\boldsymbol{W}^i_K \mathbf{r}\right)^T \boldsymbol{W}^i_Q \mathbf{r}\right],$$

$$\text{MLP}(\mathbf{r}) = \boldsymbol{W}_2 \cdot \text{ReLU}\left(\boldsymbol{W}_1 \cdot \text{MultiHeadAttn}(\mathbf{r}) + \boldsymbol{b}_1 \mathbf{1}_n^T\right) + \boldsymbol{b}_2 \mathbf{1}_n^T,$$

where $\boldsymbol{W}^i_O \in \mathbb{R}^{d\times k}, \boldsymbol{W}^i_V, \boldsymbol{W}^i_K, \boldsymbol{W}^i_Q \in \mathbb{R}^{k\times d}, \boldsymbol{W}_2 \in \mathbb{R}^{d\times r}, \boldsymbol{W}_1 \in \mathbb{R}^{r\times d}, \boldsymbol{b}_2 \in \mathbb{R}^d, \boldsymbol{b}_1 \in \mathbb{R}^r$, and MLP($\mathbf{r}$) is multilayer perceptron block with a ReLU activation function. Additionally, we can also replace the ReLU function with the GeLU function. The number of heads e and the head size k are two main parameters of the attention layer, and l denotes the hidden layer size of the feed-forward layer.

We would like to point out that our definition of the Multi-Head Attention layer is the same as Vaswani et al. (2017), in which they combine attention heads and multiply them by a matrix $\boldsymbol{W}_O \in \mathbb{R}^{d\times ke}$. One difference in our setup is the absence of layer normalization, which simplifies our analysis while preserving the basic architecture of the transformer.

We define transformer networks as the composition of Transformer blocks. The family of the sequence-to-sequence functions corresponding to the Transformers can be defined as:

$$\mathcal{T}^{e,k,l} := \{h\},$$

where $h : \mathbb{R}^{d\times m} \to \mathbb{R}^{d\times m}$ is a composition of Transformer blocks $t^{e,k,l} : \mathbb{R}^{d\times m} \to \mathbb{R}^{d\times m}$ denotes a Transformer block defined by an attention layer with e heads of size k each, and a feed-forward layer with l hidden nodes. An illustration is shown in Fig. 4b.

Theorem 5.4 (*Yun et al., 2019*). *Let $\mathbb{F}$ be the sequence-to-sequence function class, which consists of all continuous permutation equivariant functions with compact support that map $\mathbb{R}^{d\times m} \to \mathbb{R}^{d\times m}$. For $1 \le p < \infty$ and $\epsilon > 0$, then for any given $f \in \mathbb{F}$, there exists a Transformer network $h \in \mathcal{T}^{2,1,4}$, such that:*

$$\mathrm{d}_p(h, f) := \left(\int \|h(\mathbf{r}) - f\|_p^p \, d\mathbf{r}\right)^{1/p} < \epsilon.$$

5.3. Solution constraint layer

In many real-world applications, there could be constraints on the solution structure $\mathbf{x}$ across all preferences. The hypernetwork model can properly handle these constraints for all solutions via constraint layers (Lin, Zhang, Yang, & Zhang, 2023).

We first begin with the most common constraint: that the decision variables are explicitly bounded. In this case, we can simply add a transformation operator to the output of the hypernetwork:

$$\mathbf{x}_\mathbf{r} = c\left(h\left(\mathbf{r}, \phi\right)\right),$$

where $h(\mathbf{r}, \phi)$ is a hypernetwork and $c : \mathbb{R}^n \to \mathbb{R}^n$ is an activation function that maps arbitrary model output $h(\mathbf{r}; \cdot) \in (-\infty, \infty)^n$ into the desired bounded range. The activation function should be differentiable, and hence, we can directly learn the bounded hypernetwork by the gradient-based method proposed in the main paper. We introduce three typical bounded constraints and the corresponding activation functions in the following:

Non-Negative Decision Variables. We can set $c(\cdot)$ as the rectified linear function (ReLU):

$$\mathbf{x}_\mathbf{r} = c(\mathbf{x}_\mathbf{r}) = \max\{0, \mathbf{x}_\mathbf{r}\}.$$

Which will keep the values for all non-negative inputs and set the rest to 0. In other words, all the output of hypernetwork will now be in the range $[0, \infty)$.

Box-bounded Decision Variables. We can set $c(\cdot)$ as the sigmoid function:

$$\mathbf{x}_\mathbf{r} = c(\mathbf{x}_\mathbf{r}) = \frac{1}{1 + e^{-\mathbf{x}_\mathbf{r}}}.$$

Now, all the decision variables will range from 0 to 1. It is also straightforward to other bounded regions with arbitrary upper and lower bounds for each decision variable.

Simplex Constraints. In some applications, a fixed amount of resources must be arranged for different agents or parts of a system. We can use the Softmax function $c(\cdot)$ where

$$\mathbf{x}_\mathbf{r} = c(\mathbf{x}_\mathbf{r}) = \frac{e^{\mathbf{x}_\mathbf{r}^i}}{\sum_{j=1}^n e^{\mathbf{x}_\mathbf{r}^j}}, \forall i \in \{1, 2, \ldots, n\},$$

such that all the generated solutions are on the simplex $\{\mathbf{x} \in \mathbb{R}^n \mid \sum_{i=1}^n \mathbf{x}_i = 1 \text{ and } \mathbf{x}_i > 0 \text{ for } i = 1, \ldots, n\}$.

These bounded constraints are for each individual solution. With the specific activation functions, all (infinite) generated solutions will always satisfy the structure constraints, even for those with unseen contexts and preferences. This is also an anytime feasibility guarantee during the whole optimization process.

Algorithm 1 : Hypernetwork training for Connected Pareto Front.

Input: Init $\phi_0, t = 0$, $\mathbf{a}, \mathbf{b}, \alpha$, model-type.
Output: ϕ^*.
while *not converged* **do**
  $\mathbf{r}_t = Dir(\alpha)$
  if *model-type is 'MLP'* **then**
    $\mathbf{x}_{\mathbf{r}_t} = h_{\text{mlp}}\left(\mathbf{r}_t, \phi\right)$
  else
    $\mathbf{x}_{\mathbf{r}_t} = h_{\text{trans}}\left(\mathbf{r}_t, \phi\right)$
  end
  $\phi_{t+1} = \phi_t - \xi \nabla_\phi S\left(\mathcal{F}\left(\mathbf{x}_{\mathbf{r}_t}\right), \mathbf{r}_t, \mathbf{a}\right)$
  $t = t + 1$
end
$\phi^* = \phi_t$

Theorem 5.5. *Let neural network h be a set of multilayer perceptron or transformer blocks with σ activation. Assume that ϕ^* is stationary point of Algorithm 1 and $\nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*) \neq 0$. Then $\mathbf{x}(\hat{\mathbf{r}}) = h(\hat{\mathbf{r}}, \phi^*)$ is a global optimal solution to Problem (LP), and there exists a neighborhood U of $\hat{\mathbf{r}}$ and a smooth mapping $\mathbf{x}(\mathbf{r})$ such that $\mathbf{x}(\mathbf{r}^*)_{\mathbf{r}^* \in U}$ is also a global optimal solution to Problem (LP).*

Proof. Assume that $\mathbf{x}(\hat{\mathbf{r}})$ is not a local optimal solution to Problem (LP). Indeed, by using universal approximation Theorems 5.3 and 5.4, we can approximate smooth function $\mathbf{x}(\hat{\mathbf{r}})$ by a network $h(\hat{\mathbf{r}}, \phi^*)$. Since $S\left(\mathcal{F}(\mathbf{x}), \mathbf{r}\right) := \max_{i=1,\ldots,m} \left\{r_i \left(f_i(\mathbf{x}) - \mathbf{a}_i\right)\right\}$ is pseudoconvex on X and $\nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*) \neq 0$, we imply:

$$\exists \mathbf{x}' \in X, \mathbf{x}' \neq \mathbf{x} : S(\mathbf{x}') < S(\mathbf{x}) \Rightarrow \begin{bmatrix} \nabla S(\mathbf{x})(\mathbf{x}' - \mathbf{x}) \nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*) < 0 \\ \nabla S(\mathbf{x})(\mathbf{x}' - \mathbf{x}) \nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*) > 0 \end{bmatrix}. \quad (1)$$

Besides ϕ^* is stationary point of Algorithm 1, hence:

$$\nabla S(\mathbf{x}) \nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*) = 0,$$

then:

$$\nabla_\mathbf{x} S(\mathbf{x}) = 0. \quad (2)$$

Combined with $\mathbf{x}' \neq \mathbf{x}$, we have:

$$\nabla S(\mathbf{x}) \nabla_\phi \mathbf{x}(\hat{\mathbf{r}}; \phi^*)(\mathbf{x}' - \mathbf{x}) = 0. \quad (3)$$

From (1), (2), and (3), we have $\mathbf{x}(\hat{\mathbf{r}}) = h(\hat{\mathbf{r}}, \phi^*)$ is a stationary point or a local optimal solution to Problem (LP). With S is pseudoconvex on X, then $\mathbf{x}(\hat{\mathbf{r}})$ is a global optimal solution to Problem (LP) (Mangasarian, 1994). We choose any $\mathbf{r}^* \in U$ that is neighborhood of $\hat{\mathbf{r}}$, i.e. $\mathbf{r}^* \in \mathcal{P}$. Reiterate the procedure of optimizing Algorithm 1 we have $\mathbf{x}(\mathbf{r}^*)$ is a global optimal solution to Problem (LP). □

Remark 5.1. Via Theorem 5.5, we can see that the optimal solution set of Problem (SMOP) can be approximated by Algorithm 1. From Theorem 5.1, it guarantees that any reference vector $\mathbf{r}\left(\mathbf{r}_i > 0\right)$ of Dirichlet distribution $Dir(\alpha)$ always generates an optimal solution of Problem (SMOP) such that split feasibility constraints. Then the Pareto front is also approximated accordingly by mapping $\mathcal{F}(\cdot)$ respectively.

6. Learning disconnected pareto front with hyper-transformer network

The PF of some MOPs may be discontinuous in real-world applications due to constraints, discontinuous search space, or complicated shapes. Existing methods are mostly built upon an evolutionary searching algorithm, which requires massive computation to give acceptable solutions. In this work, we introduce two transformer-based methods to effectively learn the irregular Pareto Front, which we shall call Hyper-Transformer with Joint Input and Hyper-Transformer with Mixture of Experts.

6.1. Hyper-transformer with joint input

However, to guarantee real-time and flexibility in the system, we re-design adaptive model joint input for split feasibility constraints as follows:

$$\phi^* = \arg\min_\phi \mathop{\mathbb{E}}_{\substack{\mathbf{r} \sim Dir(\alpha) \\ \mathbf{a} \sim U(0,1)}} \left[S\left(\mathcal{F}\left(h_{\text{trans-joint}}\left(\mathbf{r}, \mathbf{a}, \phi\right)\right), \mathbf{r}, \mathbf{a}\right)\right] \quad \text{(Joint-Hyper-Trans)}$$

$$\text{s.t. } h_{\text{trans-joint}}\left(\mathbf{r}, \mathbf{a}, \phi\right) \in X$$
$$\mathcal{F}\left(h_{\text{trans-joint}}\left(\mathbf{r}, \mathbf{a}, \phi\right)\right) \leq \mathbf{b},$$

where $U(0, 1)$ is uniform distribution.

6.2. Hyper-transformer with mixture of experts

Despite achieving notable results in the continuous Pareto front, the joint input approach fails to achieve the desired MED in the discontinuous scenario. We, therefore, integrate the idea from the mixture of experts (Noam Shazeer et al., 2017) into the transformer-based model and assume that each Pareto front component will be learned by one expert.

In its simplest form, the MoE consists of a set of k experts (neural networks) $e_i : \mathcal{X} \to \mathbb{R}^u, i \in \{1, 2, \ldots, k\}$, and a gate $g : \mathcal{X} \to \mathbb{R}^n$ that assigns weights to the experts. The gate's output is assumed to be a

probability vector, i.e., $g(x) \geq 0$ and $\sum_i^k g(x)_i = 1$, for any $x \in \mathcal{X}$. Given an example $x \in \mathcal{X}$, the corresponding output of the MoE is a weighted combination of the experts:

$$\sum_i^n e_i(x) g(x)_i.$$

In most settings, the experts are usually MLP modules and the gate g is chosen to be a *Softmax* gate, and then the top-k expert with the highest values will be chosen to process the inputs associated with the corresponding value. As shown in Fig. 5b, our model takes $r_i, i = 1, 2, \ldots, m$ as input, the corresponding reference vector for ith constraints. We follow the same architecture design for the expert networks but omit the gating mechanism by fixing the routing of r_i to the ith expert. This allows each expert to specialize in a certain region of the image space in which may lie a Pareto front. By this setting, our model resembles a multi-model approach but has much fewer parameters and is simpler.

We adapt this approach for Hyper-Transformer as follows:

$$\phi^* = \arg\min_{\phi} \mathop{\mathbb{E}}_{\mathbf{r} \sim Dir(\alpha)} \left[\mathcal{S}\left(\mathcal{F}\left(h_{\text{trans-expert}}\left(\mathbf{r}, ID, \phi\right)\right), \mathbf{r}, \mathbf{a}[ID] \right) \right] \tag{Expert-Hyper-Trans}$$

$$\text{s.t. } h_{\text{trans-expert}}\left(\mathbf{r}, ID, \phi\right) \in X$$
$$\mathcal{F}\left(h_{\text{trans-expert}}\left(\mathbf{r}, ID, \phi\right)\right) \leq \mathbf{b},$$

where $h_{\text{trans-expert}}\left(\mathbf{r}, ID, \phi\right) = \left[\sum_{i=1}^{k} \text{MLP}_i\left(h_{\text{trans}}(\mathbf{r}; \phi)\right)\right] g(ID)$ with $g(ID) = (0_1, \ldots, 1_{ID}, \ldots, 0_k)$.

Hypernetwork h with architecture corresponding to Joint Input and Mixture of Experts was illustrated in Figs. 5a and 5b. The pseudocode that solves Problem (Joint-Hyper-Trans) and Problem (Expert-Hyper-Trans) is presented in Algorithm 2.

Indeed, Algorithm 2 with model-type is 'Expert' requires the number of experts to be determined in advance. In the current study, we are setting the number of experts as a fixed parameter that needs to be estimated from the beginning. To determine the number of experts, we need to identify the number of connected components of the Pareto Front. This is not an easy task in practice. This issue will be clarified in our subsequent research. Although we currently have some ideas to address this problem, one of them is using heuristic algorithms to search for connected components. This approach has been proposed in some works on MOEA. Coello and Sierra (2003) proposed a two-phase algorithm: Phase 1 searches for connected components; Phase 2 finds efficient points on the Pareto surface in each connected component. In summary, due to the scope of the paper and the focus on the main issue, we will study and clarify the problem you raised in subsequent research. Another approach to this problem is through deep learning, as presented by Zhao et al. (2021). In our MoEs architecture, the gating mechanism is not entirely discarded; we use a rule-based gating mechanism similar to the Hash Layers (Roller, Sukhbaatar, Weston, et al., 2021), where the gating function is not a neural network but simply a hashing function. After dividing the decision space into separate regions, we can determine the box $[\mathbf{a}, \mathbf{b}]$ for each region and through the box $[\mathbf{a}, \mathbf{b}]$ to decide which expert the ray will go into.

7. Application of controllable Pareto front learning in multi-task learning

7.1. Multi-task learning as multi-objectives optimization

Denotes a supervised dataset $(\mathbf{x}, \mathbf{y}) = \left\{\left(x_j, y_j\right)\right\}_{j=1}^{N}$ where N is the number of data points. They specified the MOO formulation of Multi-task learning from the empirical loss $\mathcal{L}^i(\mathbf{y}, g(\mathbf{x}, \theta))$ using a vector-valued loss $\mathcal{L}$:

$$\theta = \arg\min_{\theta} \mathcal{L}\left(\mathbf{y}, g\left(\mathbf{x}, \theta\right)\right),$$

Algorithm 2 : Hyper-Transformer training for Disconnected Pareto Front.

Input: Init $\phi_0, t = 0$, **idxs** $= [0, \ldots, k]$, $\mathbf{a}, \mathbf{b}, \alpha$, model-type.
Output: ϕ^*.
for ID *in* **idxs do**
  while *not converged* **do**
    $\mathbf{a}_t = \mathbf{a}[ID]$
    $\mathbf{r}_t = Dir(\alpha)$
    if *model-type is 'Joint input'* **then**
      $\mathbf{x}_{\mathbf{r}_t} = h_{\text{trans-joint}}\left(\mathbf{r}_t, \mathbf{a}_t, \phi_t\right)$
    else
      $\mathbf{x}_{\mathbf{r}_t} = h_{\text{trans-expert}}\left(\mathbf{r}_t, ID, \phi_t\right)$
    end
    $\phi_{t+1} = \phi_t - \xi \nabla_{\phi} \mathcal{S}\left(\mathcal{F}\left(\mathbf{x}_{\mathbf{r}_t}\right), \mathbf{r}_t, \mathbf{a}_t\right)$
    $t = t + 1$
  end
end
$\phi^* = \phi_t$

$$\mathcal{L}\left(\mathbf{y}, g\left(\mathbf{x}, \theta\right)\right) = \left(\mathcal{L}_1\left(\mathbf{y}, g\left(\mathbf{x}, \theta\right)\right), \ldots, \mathcal{L}_m\left(\mathbf{y}, g\left(\mathbf{x}, \theta\right)\right)\right)^T$$

where $g(\mathbf{x}; \theta) : \mathcal{X} \times \Theta \rightarrow \mathcal{Y}$ represents to a Target network with parameters θ.

7.2. Controllable Pareto front learning in multi-task learning

Controllable Pareto Front Learning in Multi-task Learning by solving the following:

$$\phi^* = \arg\min_{\phi} \mathop{\mathbb{E}}_{\substack{\mathbf{r} \sim Dir(\alpha) \\ (\mathbf{x}, \mathbf{y}) \sim p_D}} \mathcal{S}(\mathcal{L}(\mathbf{y}, g(\mathbf{x}, \theta_{\mathbf{r}})), \mathbf{r}, \mathbf{a})$$
$$\text{s.t. } \theta_{\mathbf{r}} = h(\mathbf{r}, \phi^*)$$
$$\mathcal{L}(\mathbf{y}, g(\mathbf{x}, \theta_{\mathbf{r}})) \leq \mathbf{b},$$

where $h : \mathcal{P} \times \Phi \rightarrow \Theta$ represents to a hypernetwork, $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_m), \mathbf{a}_i \geq 0$ is the lower-bound vector for the loss vector $\mathcal{L}(\mathbf{y}, g(\mathbf{x}, \theta_{\mathbf{r}}))$, and the upper-bound vector denoted as $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_m), \mathbf{b}_i \geq 0$ is the desired loss value. The random variable $\mathbf{r}$ is a preference vector, forming the trade-off between loss functions.

8. Computational experiments

The code is implemented in Python language programming and the Pytorch framework (Paszke et al., 2019). We compare the performance of our method with the baseline method (Tuan et al., 2023) and provide the setting details and additional experiments. Our source code is available at https://github.com/tuantran23012000/CPFL_Hyper_Transformer.git.

8.1. Experiment details

8.1.1. Computational analysis

Hyper-Transformer consists of two blocks: the Self-Attention mechanism and the Multilayer Perceptron. With Hypernetwork w/o join input architect, we assume dimension of three matrices $\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V$ is d, number of heads is 2. Besides, we also assume the input and output of the MLP block with d dimension. Hence, the total parameters of the Hyper-Transformer is $4d^2+4d$. The Hyper-MLP architect uses six hidden linear layers with d dimension input and output. Therefore, the total parameters of Hyper-MLP is $6d^2 + 6d$.

Although the total parameters of Hyper-MLP are larger than Hyper-Transformer, the number of parameters that need to be learned for the MOP examples is the opposite when incorporating the Embedding

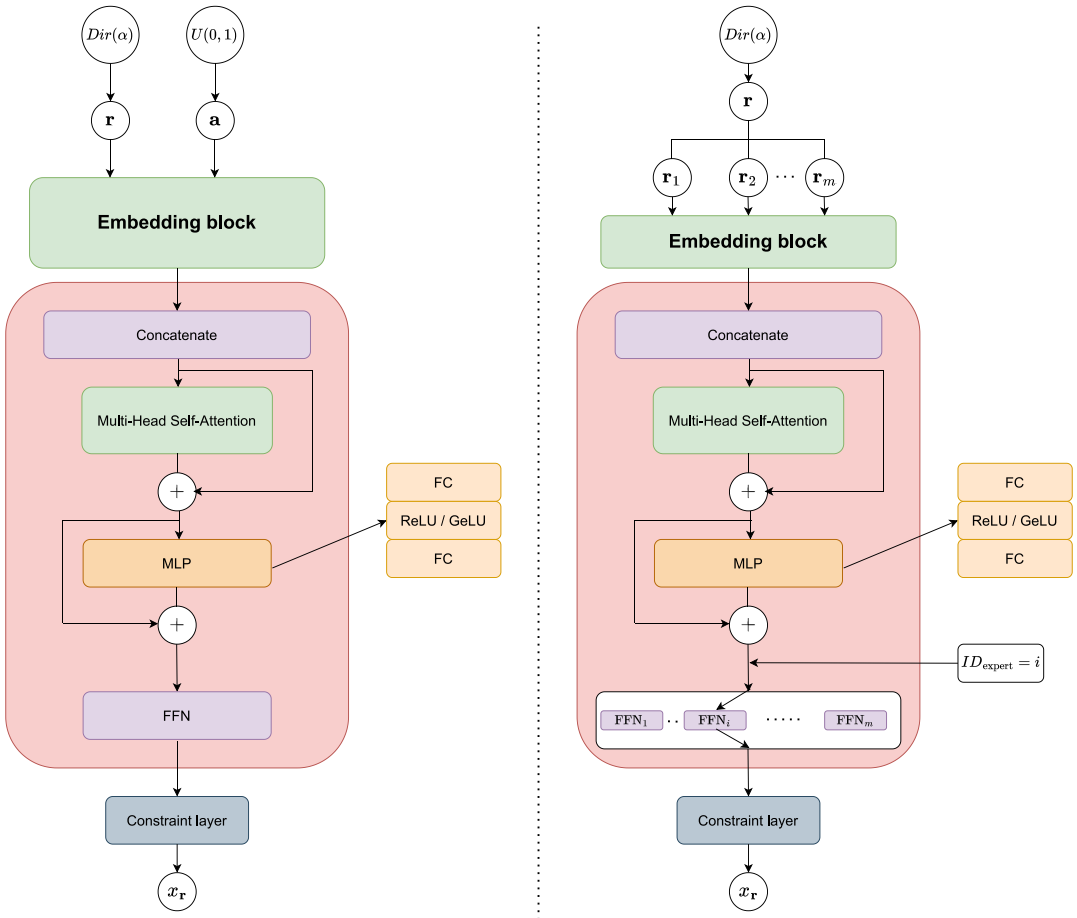


Fig. 5. Proposed Transformer-based Hypernetwork. *Left*: The Joint Input model takes reference vectors and objective function's lower bounds corresponding to each Pareto front component. *Right*: Mixture of Experts integrated model which inputs reference vectors.

Table 1
Information of MOO problems.

Problem	n	m	Objective function	Pareto-optimal	Pareto front
(CVX1)	1	2	convex	convex	connected
(CVX2)	2	2	convex	convex	connected
(CVX3)	3	3	convex	convex	connected
(ZDT1)	30	2	non-convex	convex	connected
(ZDT2)	30	2	non-convex	non-convex	connected
(ZDT3)	30	2	non-convex	non-convex	disconnected
(ZDT3*)	30	2	non-convex	non-convex	disconnected
(DTLZ2)	10	3	non-convex	non-convex	connected
(DTLZ7)	10	3	non-convex	non-convex	disconnected

block. In the two architectures described in Figs. 4a and 4b, the parameters to be learned of Hyper-Trans are:

$$6d^2 + 6d + 2md + (d+1)n,$$

and with Hyper-MLP are:

$$6d^2 + 6d + (m+1)d + (d+1)n.$$

From there, we see that the difference in the total parameters to be learned $(m-1)d$ between the Hyper-Trans and Hyper-MLP models is insignificant. It only depends on the width of the hidden layers d and the number of objective functions $m \geq 2$.

8.1.2. Training setup

The experiments MOO were implemented on a computer with CPU - Intel(R) Core(TM) i7-10700, 64-bit CPU @2.90 GHz, and 16 cores. Information on MOO test problems is illustrated in Table 1.

We use Hypernetwork based on multi-layer perceptron (MLP), which has the following structure:

$$\begin{aligned} h_{\text{mlp}}(\mathbf{r}, \phi) : \mathbf{Input}(\mathbf{r}) &\to \mathbf{Linear}(m, d) \to \mathbf{ReLU} \to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \\ &\to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \\ &\to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \\ &\to \mathbf{Linear}(d, d) \to \mathbf{ReLU} \to \mathbf{Linear}(d, n) \\ &\to \mathbf{Constraintlayer} \to \mathbf{Output}(\mathbf{x_r}). \end{aligned}$$

Toward Hypernetwork based on the Transformer model, we use the structure as follows:

$$\begin{aligned} h_{\text{trans}}(\mathbf{r}, \phi) : \mathbf{Input}(\mathbf{r}) &\to \begin{bmatrix} \mathbf{Linear}(1, d) \\ \dots \\ \mathbf{Linear}(1, d) \end{bmatrix} \to \mathbf{Concatenate} \\ &\to \begin{bmatrix} \mathbf{Multi-HeadSelf-Attention} \\ \mathbf{Identitylayer} \end{bmatrix} \\ &\to \mathbf{Sum} \to \begin{bmatrix} \mathbf{MLP} \\ \mathbf{Identitylayer} \end{bmatrix} \to \mathbf{Sum} \to \mathbf{Linear}(d, n) \\ &\to \mathbf{Constraintlayer} \to \mathbf{Output}(\mathbf{x_r}). \end{aligned}$$

8.2. Evaluation metrics

Mean Euclid Distance (MED). How well the model maps preferences to the corresponding Pareto optimal solutions on the Pareto front serves as a measure of its quality. To do this, we use the Mean Euclidean

Table 2
Hyperparameters for training MOO problems.

Problem	Hyperparameters
(CVX1)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0, 0.8], [0.1, 0.6], [0.2, 0.4], [0.35, 0.22], [0.6, 0.1]]$
(CVX2)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0, 0.6], [0.02, 0.4], [0.16, 0.2], [0.2, 0.15], [0.4, 0.02]]$
(CVX3)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0.15, 0.2, 0.7], [0.2, 0.5, 0.6], [0.2, 0.7, 0.4], [0.35, 0.6, 0.22], [0.6, 0.1, 0.46]]$
(ZDT1)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0, 0.8], [0.1, 0.6], [0.2, 0.4], [0.35, 0.22], [0.6, 0.1]]$
(ZDT2)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0.1, 0.9], [0.1, 0.6], [0.2, 0.4], [0.35, 0.22], [0.6, 0.1]]$
(ZDT3)	Adam optimizer, $\alpha = 0.6, d = 30, iter = 20000, lr = 0.001, a = [[0.01, 0.81], [0.16, 0.61], [0.4, 0.41], [0.62, 0.23], [0.81, 0.1]]$
(ZDT3*)	Adam optimizer, $\alpha = 0.6, d = 10, iter = 20000, lr = 0.001, a = [[0.8, 0.62], [0.01, 0.7]], A = 2, \gamma = 3, \beta = \frac{1}{3}$
(DTLZ2)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0.15, 0.2, 0.7], [0.2, 0.5, 0.6], [0.2, 0.7, 0.4], [0.35, 0.6, 0.22], [0.6, 0.1, 0.46]]$
(DTLZ7)	Adam optimizer, $\alpha = 0.6, d = 20, iter = 20000, lr = 0.001, a = [[0.62, 0.62, 0.4], [0.01, 0.62, 0.5], [0.01, 0.01, 0.82], [0.62, 0.01, 0.6]]$

Distance (MED) (Tuan et al., 2023) between the truth corresponding Pareto optimal solutions $\mathcal{F}^* = \{f(\mathbf{x}_\mathbf{r}^*)\}$ and the learned solutions $\hat{\mathcal{F}} = \{f(h(\mathbf{r};\phi))\}$.

$$MED(\mathcal{F}^*, \hat{\mathcal{F}}) = \frac{1}{|\mathcal{F}^*|}\left(\sum_{i=1}^{|\mathcal{F}^*|}\left\|\mathcal{F}_i^* - \hat{\mathcal{F}}_i\right\|_2\right).$$

Hypervolume (HV). Hypervolume (Zitzler & Thiele, 1999) is the area dominated by the Pareto front. Therefore, the quality of a Pareto front is proportional to its hypervolume. Given a set of k points $\mathcal{M} = \{m^j | m^j \in \mathbb{R}^m; j = 1, \dots, k\}$ and a reference point $\rho \in \mathbb{R}_+^m$. The Hypervolume of S is measured by the region of non-dominated points bounded above by $m \in \mathcal{M}$, and then the hypervolume metric is defined as follows:

$$HV(S) = VOL\left(\bigcup_{m\in\mathcal{M}, m\prec\rho} \Pi_{i=1}^m [m_i, \rho_i]\right).$$

Hypervolume Difference(HVD). The area dominated by the Pareto front is known as Hypervolume. The higher the Hypervolume, the better the Pareto front quality. For evaluating the quality of the learned Pareto front, we employ Hypervolume Difference (HVD) between the Hypervolumes computed by the truth Pareto front $\mathcal{F}$ and the learned Pareto front $\hat{\mathcal{F}}$ as follows:

$$HVD(\mathcal{F}^*, \hat{\mathcal{F}}) = HV(\mathcal{F}^*) - HV(\hat{\mathcal{F}}).$$

8.3. Synthesis experiments

We utilized a widely used synthesis multi-objective optimization benchmark problem in the following to evaluate our proposed method with connected and disconnected Pareto front. For ease of test problems, we normalize the PF to $[0, 1]^m$.

8.3.1. Problems with connected Pareto front

CVX1 (Tuan et al., 2023):

$$\min\left\{x, (x-1)^2\right\} \quad \text{s.t. } 0 \le x \le 1. \tag{CVX1}$$

CVX2 (Binh & Korn, 1997):

$$\min\left\{f_1, f_2\right\} \quad \text{s.t. } x_i \in [0, 5], i = 1, 2 \tag{CVX2}$$

where

$$f_1 = \frac{x_1^2 + x_2^2}{50}, f_2 = \frac{(x_1-5)^2 + (x_2-5)^2}{50}.$$

CVX3 (Thang, Solanki, Dao, Thi Ngoc Anh, & Van Hai, 2020):

$$\min\left\{f_1, f_2, f_3\right\} \quad \text{s.t. } x_1^2 + x_2^2 + x_3^2 = 1,\ x_i \in [0, 1], i = 1, 2, 3 \tag{CVX3}$$

where

$$f_1 = \frac{x_1^2 + x_2^2 + x_3^2 + x_2 - 12x_3 + 12}{14},$$

$$f_2 = \frac{x_1^2 + x_2^2 + x_3^2 + 8x_1 - 44.8x_2 + 8x_3 + 44}{57},$$

$$f_3 = \frac{x_1^2 + x_2^2 + x_3^2 - 44.8x_1 + 8x_2 + 8x_3 + 43.7}{56}.$$

Moreover, we experiment with the additional Non-Convex MOO problems, including ZDT1-2 (Zitzler, Deb, & Thiele, 2000), and DTLZ2 (Deb, Thiele, Laumanns, & Zitzler, 2002).

ZDT1 (Zitzler et al., 2000): It is a classical multi-objective optimization benchmark problem with the form:

$$f_1(\mathbf{x}) = x_1$$
$$f_2(\mathbf{x}) = g(\mathbf{x})\left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}\right], \tag{ZDT1}$$

where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n-1} x_{i+1}$ and $0 \le x_i \le 1$ for $i = 1, \dots, n$.

ZDT2 (Zitzler et al., 2000): It is a classical multi-objective optimization benchmark problem with the form:

$$f_1(\mathbf{x}) = x_1$$
$$f_2(\mathbf{x}) = g(\mathbf{x})\left(1 - \left(f_1(\mathbf{x})/g(\mathbf{x})\right)^2\right), \tag{ZDT2}$$

where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n-1} x_{i+1}$ and $0 \le x_i \le 1$ for $i = 1, \dots, n$.

DTLZ2 (Deb et al., 2002): It is a classical multi-objective optimization benchmark problem in the form:

$$f_1(\mathbf{x}) = (1 + g(\mathbf{x}))\cos\frac{\pi x_1}{2}\cos\frac{\pi x_2}{2}$$
$$f_2(\mathbf{x}) = (1 + g(\mathbf{x}))\cos\frac{\pi x_1}{2}\sin\frac{\pi x_2}{2}$$
$$f_3(\mathbf{x}) = (1 + g(\mathbf{x}))\sin\frac{\pi x_1}{2} \tag{DTLZ2}$$

where $g(\mathbf{x}) = \sum_{i=1}^{n-2}\left(x_{i+2}\right)^2$ and $0 \le x_i \le 1$ for $i = 1, \dots, n$.

The statistical comparison results of the connected Pareto front problems of the MED scores between our proposed Hyper-Trans and the Hyper-MLP (Tuan et al., 2023) are given in Table 3. These results show that the MED scores of Hyper-Trans are statistically significantly better than those of Hyper-MLP in all comparisons. The state trajectories of $\mathcal{F}(x)$ are shown in Figs. 6, 7, 8, 9, 10, and 11. These were calculated using Hyper-Transformer and Hyper-MLP for 2D problems where $\mathbf{x}$ is generated at $\mathbf{r} = [0.5, 0.5]$ and for 3D problems where $\mathbf{x}$ is generated at $\mathbf{r} = [0.4, 0.3, 0.3]$. The number of iterations needed to train the model goes up, and the fluctuation amplitude of the objective functions produced by the hyper-transformer goes down compared to the best solution.

8.3.2. Problems with disconnected Pareto front

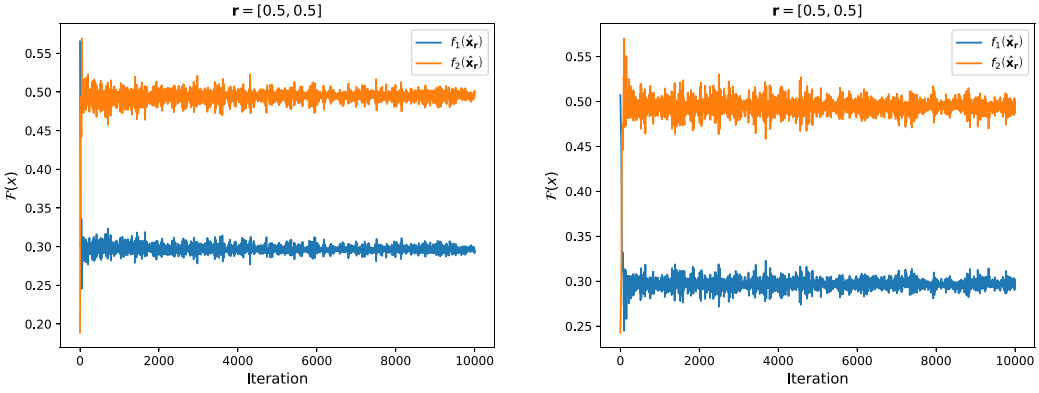
ZDT3 (Zitzler et al., 2000): It is a classical multi-objective optimization benchmark problem with the form:

$$f_1(\mathbf{x}) = x_1$$
$$f_2(\mathbf{x}) = g(\mathbf{x})\left(1 - \sqrt{\left(f_1(\mathbf{x})/g(\mathbf{x})\right)} - \left(f_1(\mathbf{x})/g(\mathbf{x})\right)\sin 10\pi f_1\right), \tag{ZDT3}$$

where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n-1} x_{i+1}$ and $0 \le x_i \le 1$ for $i = 1, \dots, n$.

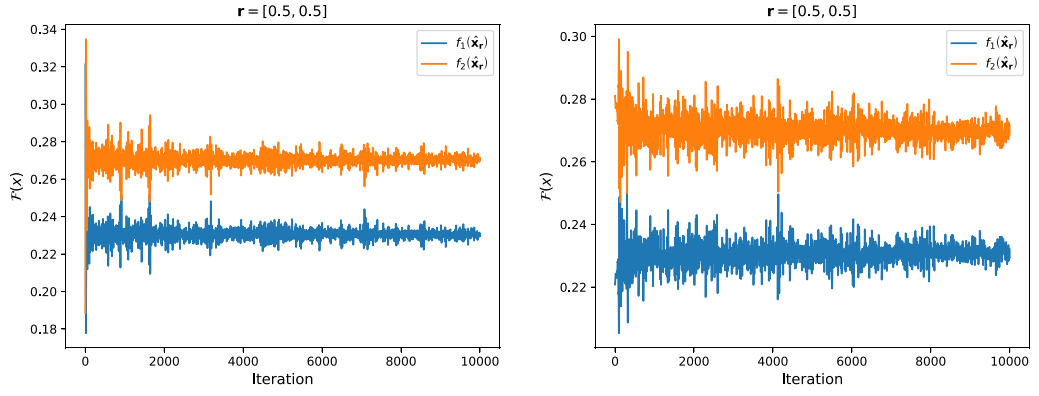
Table 3
We evaluate 30 random seed with lower bounds in Table 1.

Example	Constraint layer	Hyper-MLP (Tuan et al., 2023)	Hyper-Trans (ours)	Params	MED⇓
(CVX1)	sigmoid	✓		5 × 5701	0.00229 ± 0.00119
			✓	5 × 5731	**0.00161 ± 0.00129**
(CVX2)	sigmoid	✓		5 × 5732	0.00353 ± 0.00144
			✓	5 × 5762	**0.00258 ± 0.00127**
(CVX3)	softmax + sqrt	✓		5 × 5793	0.01886 ± 0.00784
			✓	5 × 5853	**0.00827 ± 0.00187**
(ZDT1)	sigmoid	✓		5 × 6600	0.00682 ± 0.00385
			✓	5 × 6630	**0.00219 ± 0.00049**
(ZDT2)	sigmoid	✓		5 × 6600	0.00859 ± 0.00476
			✓	5 × 6630	**0.00692 ± 0.00304**
(ZDT3)	sigmoid	✓		5 × 3210	0.18741 ± 0.00653
			✓	5 × 3230	**0.00767 ± 0.00414**
(ZDT3*)	sigmoid	✓		2 × 6600	0.00641 ± 0.00594
			✓	2 × 6630	**0.00391 ± 0.00404**
(DTLZ2)	sigmoid	✓		5 × 2810	0.06217 ± 0.01528
			✓	5 × 2850	**0.01083 ± 0.00142**
(DTLZ7)	sigmoid	✓		4 × 6010	0.03439 ± 0.02409
			✓	4 × 6070	**0.01116 ± 0.00217**



(a) Hyper-Trans. (b) Hyper-MLP.

Fig. 6. CVX1 problem.



(a) Hyper-Trans. (b) Hyper-MLP.

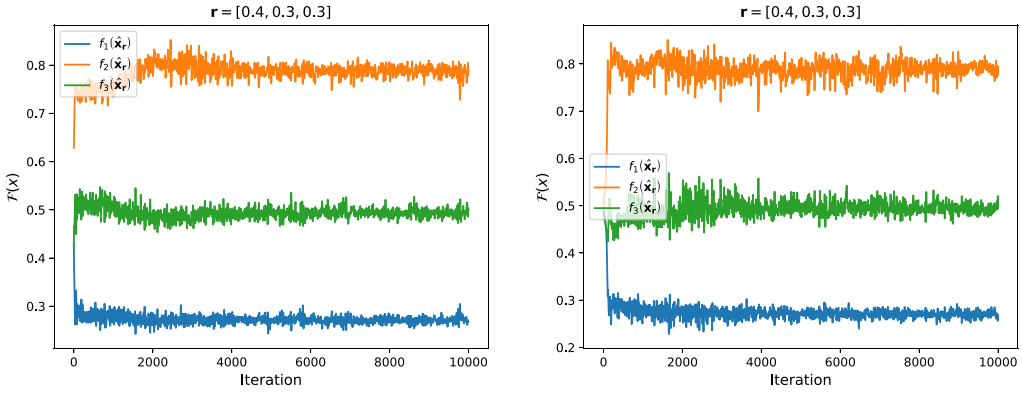
Fig. 7. CVX2 problem.

ZDT3* (Chen & Li, 2023): It is a classical multi-objective optimization benchmark problem in the form:

$$f_1(\mathbf{x}) = x_1 \qquad \text{(ZDT3*)}$$

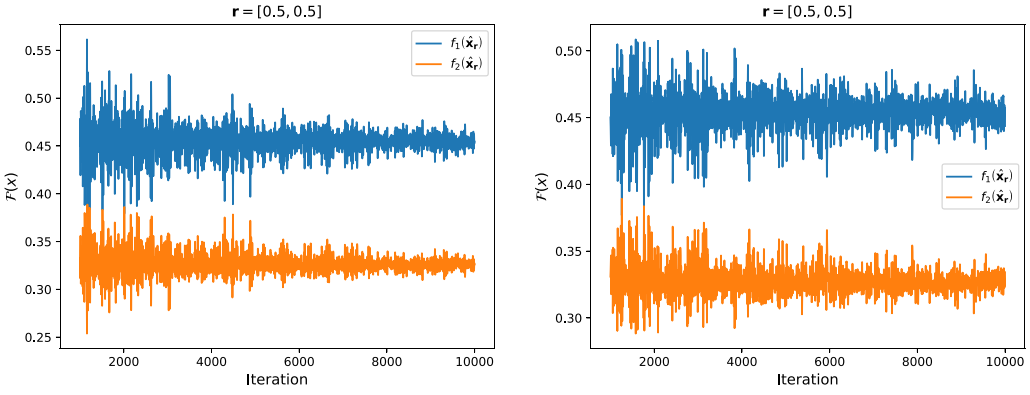
$$f_2(\mathbf{x}) = g(\mathbf{x})\left(1 - \sqrt{\left(f_1(\mathbf{x})/g(\mathbf{x})\right)} - \left(f_1(\mathbf{x})^{\gamma}/g(\mathbf{x})\right)\sin A\pi f_1^{\beta}\right),$$

where $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n-1} x_{i+1}$ and $0 \le x_i \le 1$ for $i = 1, \ldots, n$. The A determines the number of disconnected regions of the PF. γ controls the overall shape of the PF where $\gamma > 1, \gamma < 1$, and $\gamma = 1$ lead to a concave, a convex, and a linear PF, respectively. β influences the location of the disconnected regions.



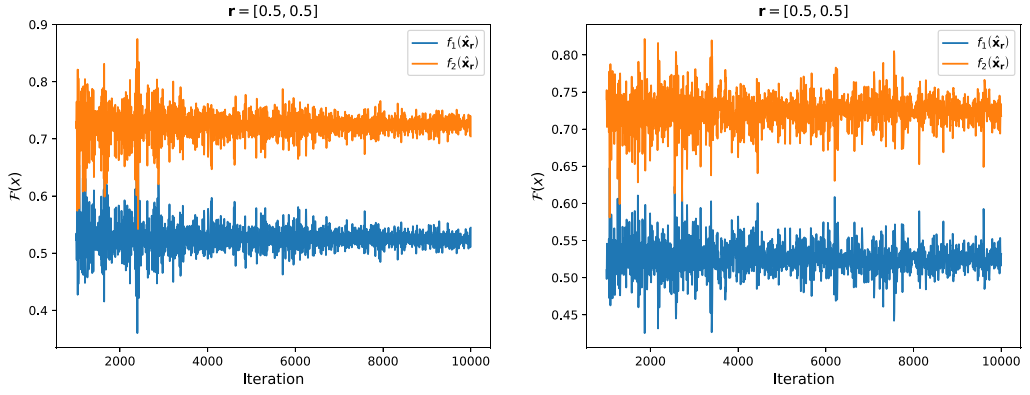
(a) Hyper-Trans. (b) Hyper-MLP.

Fig. 8. CVX3 problem.



(a) Hyper-Trans. (b) Hyper-MLP.

Fig. 9. ZDT1 problem.



(a) Hyper-Trans. (b) Hyper-MLP.

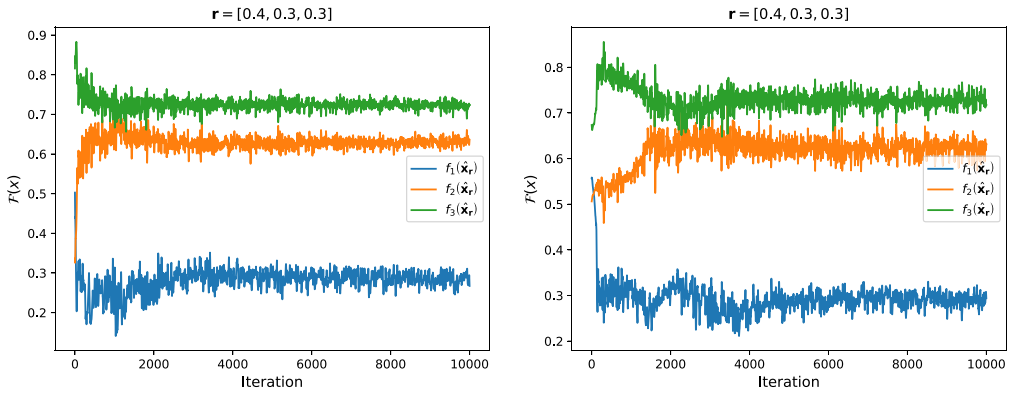
Fig. 10. ZDT2 problem.

DTLZ7 (Deb et al., 2002): It is a classical multi-objective optimization benchmark problem with the form:

$$
\begin{aligned}
&f_1(\mathbf{x}_1) = x_1 \\
&f_2(\mathbf{x}_2) = x_2 \\
&\vdots \\
&f_{m-1}(\mathbf{x}_{m-1}) = x_{m-1} \\
&f_m(\mathbf{x}_m) = \frac{(1+g(\mathbf{x}_m))h\left(f_1, f_2, \ldots, f_{m-1}, g\right)}{6},
\end{aligned}
\tag{DTLZ7}
$$

where $g(\mathbf{x}_m) = 1 + \frac{9}{|\mathbf{x}_m|}\sum_{x_i \in \mathbf{x}_m} x_i$, $h\left(f_1, f_2, \ldots, f_{m-1}, g\right) = m - \sum_{i=1}^{m-1}\left[\frac{f_i}{1+g}\left(1+\sin 3\pi f_i\right)\right]$ and $0 \le x_i \le 1$ for $i = 1, \ldots, n$. The functional g requires $k = |\mathbf{x}_m| = n - m + 1$ decision variables.

The disparity between the Hypervolume calculated utilizing the actual Pareto front $\mathcal{F}$ and the learned Pareto front $\hat{\mathcal{F}}$ of the Joint Input model is illustrated in Table 4 and Fig. 12. The outcomes of the Joint Input model surpass those of the Mixture of Experts structure. However, this distinction is not statistically significant. In addition, the hyper-transformer model with MoE still gets a much lower MED score



(a) Hyper-Trans. (b) Hyper-MLP.

Fig. 11. DTLZ2 problem.

Table 4
We evaluate 30 random seed with lower bounds in Table 1.

Example	Constraint layer	Model	Joint input	Mixture of experts	Params	HVD⇓	MED⇓
(ZDT3)	sigmoid	Hyper-Trans	✓		22 230	0.04088	0.52587 ± 0.37795
				✓	20 250	**0.00091**	**0.24787 ± 0.22053**
(ZDT3*)	sigmoid	Hyper-Trans	✓		4110	**−0.00472**	0.35923 ± 0.35548
				✓	3900	−0.00466	**0.12789 ± 0.09911**
(DTLZ7)	sigmoid	Hyper-Trans	✓		7990	**0.00265**	0.52847 ± 0.31352
				✓	7880	0.00302	**0.12338 ± 0.07585**

Table 5
Testing hypervolume on Multi-MNIST, Multi-Fashion, and Multi-Fashion+MNIST datasets with 10 folds split.

	Multi-MNIST	Multi-Fashion	Fashion-MNIST	
Method	HV ⇑	HV⇑	HV⇑	Params
Hyper-MLP (Tuan et al., 2023)	2.860 ± 0.027	2.164 ± 0.045	2.781 ± 0.039	8.66M
Hyper-Trans + ReLU (**ours**)	**2.883 ± 0.029**	2.166 ± 0.059	**2.806 ± 0.041**	8.66M
Hyper-Trans + GeLU (**ours**)	2.879 ± 0.017	**2.196 ± 0.046**	2.802 ± 0.049	8.66M

than the joint input when comparing disconnected Pareto front tests. Using complex MoE designs for the Hyper-Transformer model shows that Controllable Disconnected Pareto Front Learning could have good future results.

8.4. Multi-task learning experiments

The dataset is split into two subsets in MTL experiments: training and testing. Then, we split the training set into ten folds and randomly picked one fold to validate the model. The model with the highest HV in the validation fold will be evaluated. All methods are evaluated with the same well-spread preference vectors based on Das and Dennis (2000). The experiments MTL were implemented on a computer with CPU - Intel(R) Xeon(R) Gold 5120 CPU @ 2.20 GHz, 32 cores, and GPU - VGA NVIDIA Tesla V100-PCIE with VRAM 32 GB.

Image Classification. Our experiment involved the application of three benchmark datasets from Multi-task Learning for the image classification task: Multi-MNIST (Sabour, Frosst, & Hinton, 2017), Multi-Fashion (Xiao, Rasul, & Vollgraf, 2017), and Multi-Fashion+MNIST (Lin et al., 2019). We compare our proposed Hyper-Trans model with the Hyper-MLP model based on Multi-LeNet architecture (Tuan et al., 2023), and we report results in Table 5.

Scene Understanding. The NYUv2 dataset (Silberman, Hoiem, Kohli, & Fergus, 2012) serves as the basis experiment for our method. This dataset is a collection of 1449 RGBD images of an indoor scene that have been densely labeled at the per-pixel level using 13 classifications. We use this dataset as a 2-task MTL benchmark for depth estimation and semantic segmentation. The results are presented in Table 6 with (3, 3) as hypervolume's reference point. Our method, which includes ReLU and GeLU activations, achieves the best HV on the NYUv2 dataset with the same parameters as Hyper-MLP.

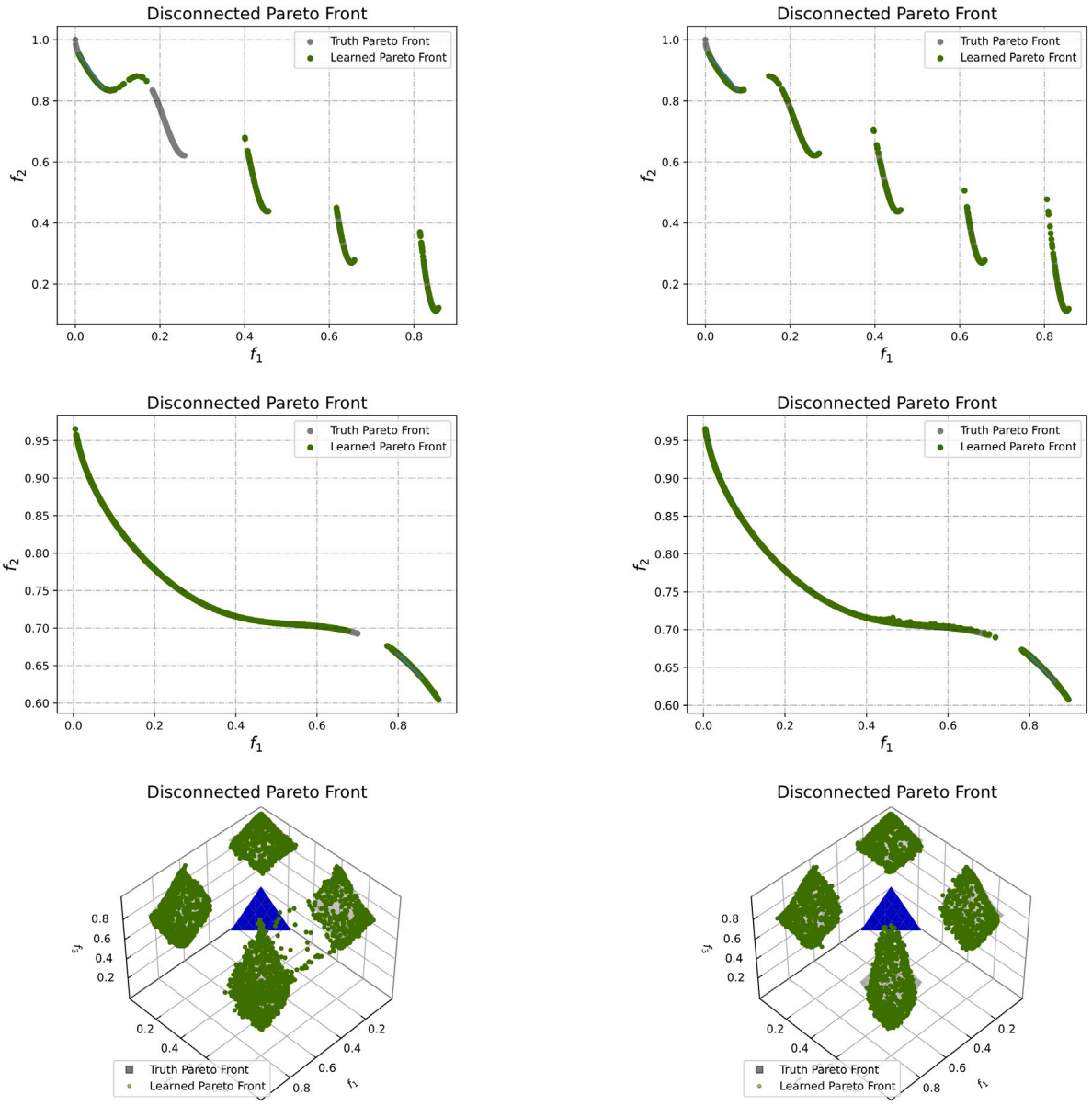
Table 6
Testing hypervolume on NYUv2 dataset.

NYUv2		
Method	HV ⇑	Params
Hyper-MLP (Tuan et al., 2023)	4.058	31.09M
Hyper-Trans + ReLU (**ours**)	4.093	31.09M
Hyper-Trans + GeLU (**ours**)	**4.135**	31.09M

Table 7
Testing hypervolume on SARCOS dataset with ten folds split.

SARCOS		
Method	HV ⇑	Params
Hyper-MLP (Tuan et al., 2023)	0.6811 ± 0.227	7.1M
Hyper-Trans + ReLU (**ours**)	0.7107 ± 0.0236	7.1M
Hyper-Trans + GeLU (**ours**)	**0.7123 ± 0.0134**	7.1M

Multi-Output Regression. We conduct experiments using the SARCOS dataset (Vijayakumar, 2000) to illustrate the feasibility of our methods in high-dimensional space. The objective is to predict seven relevant joint torques from a 21-dimensional input space (7 tasks) (7 joint locations, seven joint velocities, and seven joint accelerations). In Table 7, our proposed model shows superiority over Hyper-MLP in terms of hypervolume value.



(a) Hyper-Transformer with joint input

(b) Hyper-Transformer with Mixture of Experts

Fig. 12. *Left*: Pareto Front is approximated by the Joint Input model. *Right*: Pareto Front is approximated by the Mixture of Experts model in example (ZDT3) (top), example (ZDT3$^+$) (middle), and example (DTLZ7) (bottom).

Table 8
Testing hypervolume on ten hard-tasks CelebA dataset.

CelebA		
Method	HV ⇑	Params
Hyper-MLP (Tuan et al., 2023)	0.003995	11.09M
Hyper-Trans + ReLU (**ours**)	0.003106	11.09M
Hyper-Trans + GeLU (**ours**)	**0.004719**	11.09M

Multi-Label Classification. Continually investigate our proposed architecture in MTL problem, we solve the problem of recognizing 40 facial attributes (40 tasks) in 200 K face images on CelebA dataset (Liu, Luo, Wang, & Tang, 2015) using a big Target network: Resnet18 (11M parameters) of He, Zhang, Ren, and Sun (2016). Due to the very high dimensional scale (40 dimensions), we only test hypervolume value on ten hard-tasks CelebA datasets, including 'Arched Eyebrows,' 'Attractive,' 'Bags Under Eyes,' 'Big Lips,' 'Big Nose,' 'Brown Hair,' 'Oval Face,' 'Pointy Nose,' 'Straight Hair,' 'Wavy Hair.' Table 8 shows that the Hyper-Trans model combined with the GeLU activation function gives the highest HV value with the reference point (1,...,1).

8.5. Additional experiments

8.5.1. Number of heads and hidden dim

To understand the impact of the number of heads and the dimension of hidden layers, we analyzed the MED error based on different numbers of heads and hidden dims in Fig. 13.

We compare the Hyper-Transformer and Hyper-MLP models based on the MED score, where the dimension of the hidden layers $d = [16, 32, 64, 128]$, and the number of heads $e = [1, 2, 4, 8, 16]$.

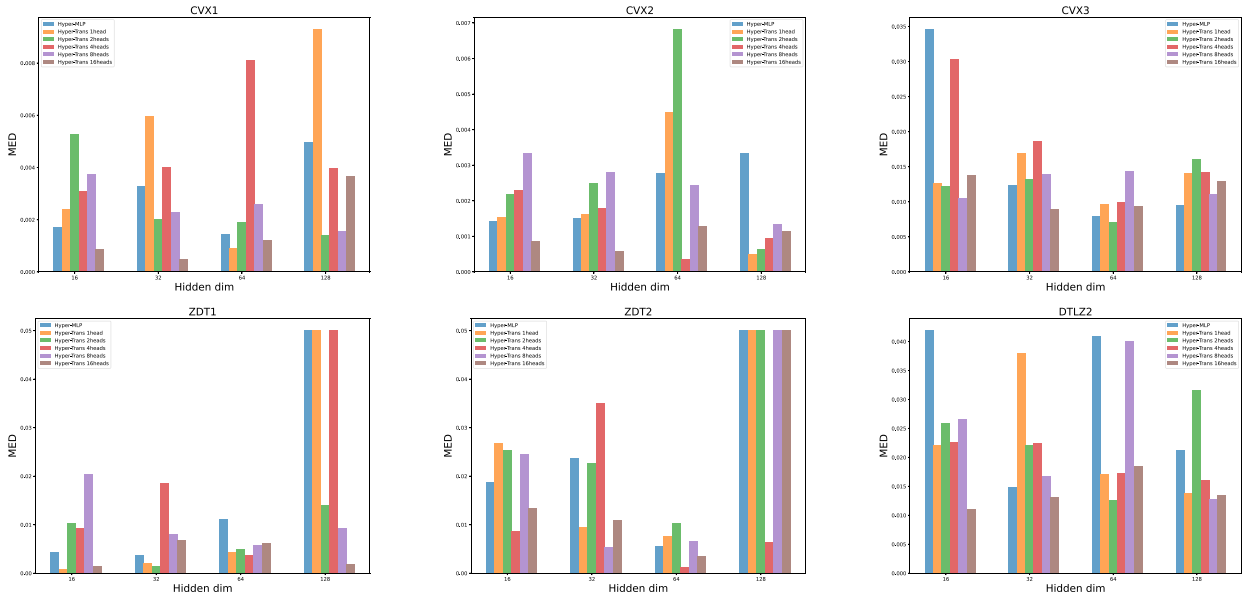


Fig. 13. MED score of Hyper-Transformer and Hyper-MLP across the number of Heads and the dimension of Hidden layers.

8.5.2. *Feature maps weight generated by hypernetworks*

We compute feature maps of the first convolutional from the weights generated by Hyper-Trans in Fig. 14. Briefly, we averaged feature maps of this convolutional layer across all its filter outputs. This accounts for the weights learned by Multi-Lenet through hypernetworks.

8.5.3. *Exactly mapping of hypernetworks*

We utilize Hypernetwork to generate an approximate efficient solution from a reference vector created by Dirichlet distribution with $\alpha = 0.6$. We trained all completion functions using an Adam optimizer (Kingma & Ba, 2014) with a learning rate of $1e-3$ and 20 000 iterations. In the test phase, we sampled three preference vectors based on each lower bound in Table 2. Besides, we also illustrated target points and predicted points from the pre-trained Hypernetwork in Figs. 15, 16, and 17.

9. Conclusion and future work

This paper presents a novel approach to tackle controllable Pareto front learning with split feasibility constraints. Additionally, we provide mathematical explanations for accurately mapping a priority vector to the corresponding Pareto optimal solution by hyper-transformers based on a universal approximation theory of the sequence-to-sequence function. Furthermore, this study represents the inaugural implementation of Controllable Disconnected Pareto Front Learning. Besides, we provide experimental computations of controllable Pareto front learning with a MED score to substantiate our theoretical reasoning. The outcomes demonstrate that the hypernetwork model, based on a transformer architecture, exhibits superior performance in the connected Pareto front and disconnected Pareto front problems compared to the multi-layer perceptron model.

Although the early results are promising, several obstacles must be addressed. Multi-task learning studies show promise for real-world multi-objective systems that need real-time control and involve difficulties with split feasibility constraints. Nevertheless, more enhancements are required for our suggested approach to addressing disconnected Pareto Front issues. This is due to the need for the model to possess prior knowledge of the partition feasibility limitations, which restricts the model's capacity to anticipate non-dominated solutions. Future research might involve the development of a resilient MoEs hyper-transformer that can effectively adjust to various split feasibility limitations and prevent the occurrence of dominated points. Moreover, we expect to use penalty function algorithms to handle more complex constraints in gradient computation (Liu et al., 2022).

CRediT authorship contribution statement

Tran Anh Tuan: Writing – review & editing, Writing – original draft. **Nguyen Viet Dung:** Validation, Writing – review & editing. **Tran Ngoc Thang:** Writing – original draft, Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Tran Ngoc Thang reports financial support was provided by Rikkeisoft Corporation and supported by the Center for Digital Technology and Economy (BK Fintech), Hanoi University of Science and Technology. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The authors thank the anonymous peer reviewers and the editor for their constructive comments which helped to improve the paper. This work was funded by Rikkeisoft Corporation and supported by the Center for Digital Technology and Economy (BK Fintech), Hanoi University of Science and Technology, Viet Nam.

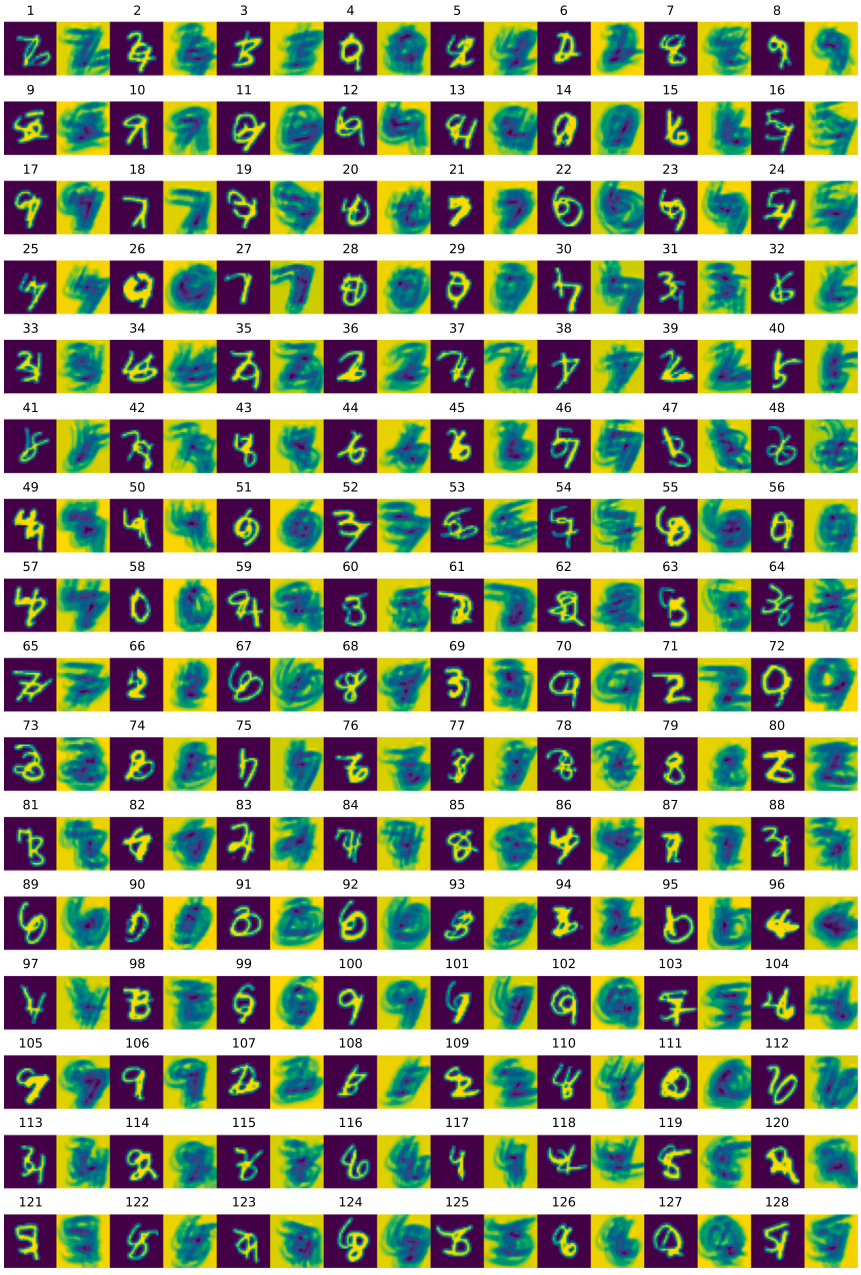
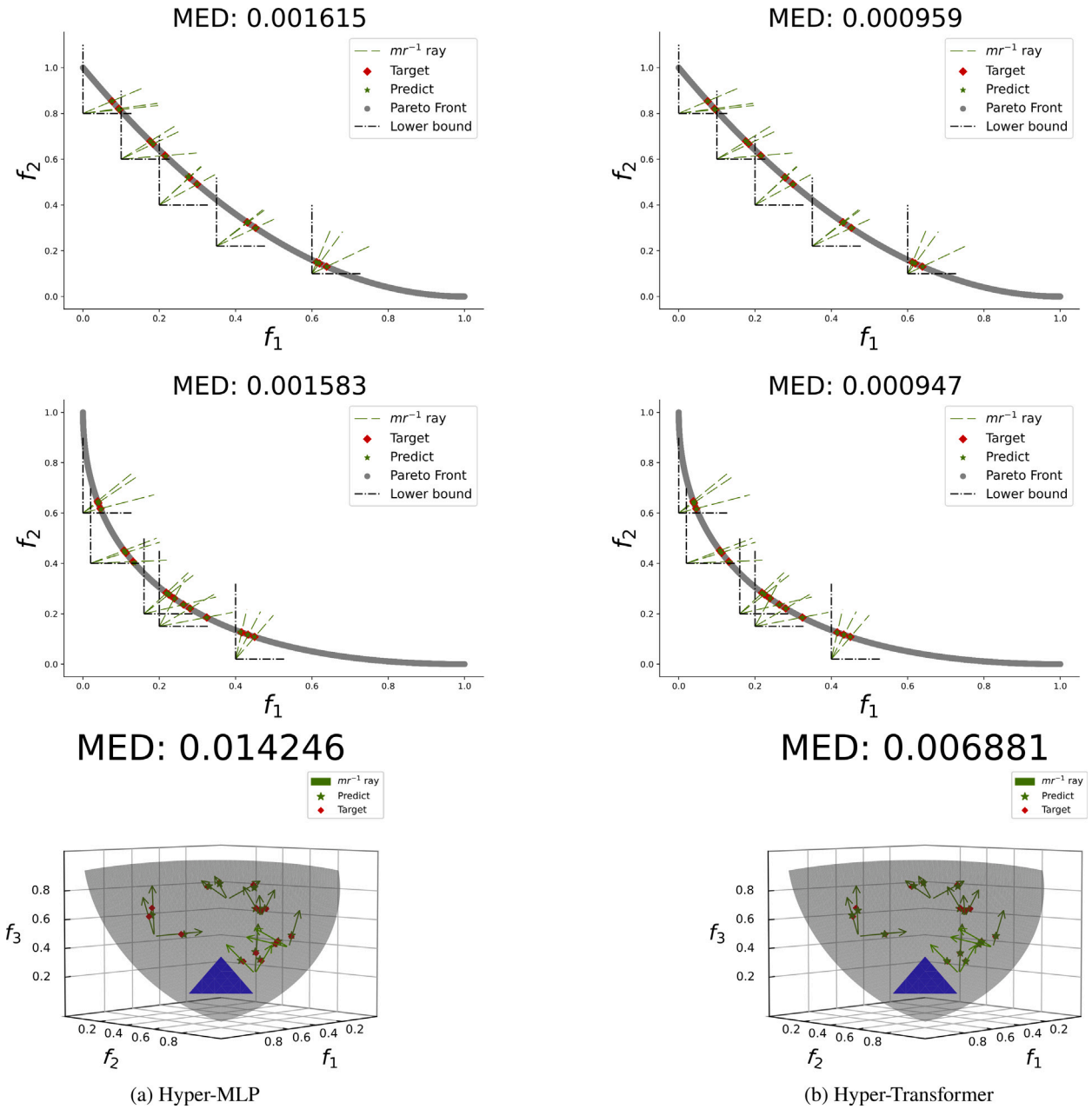
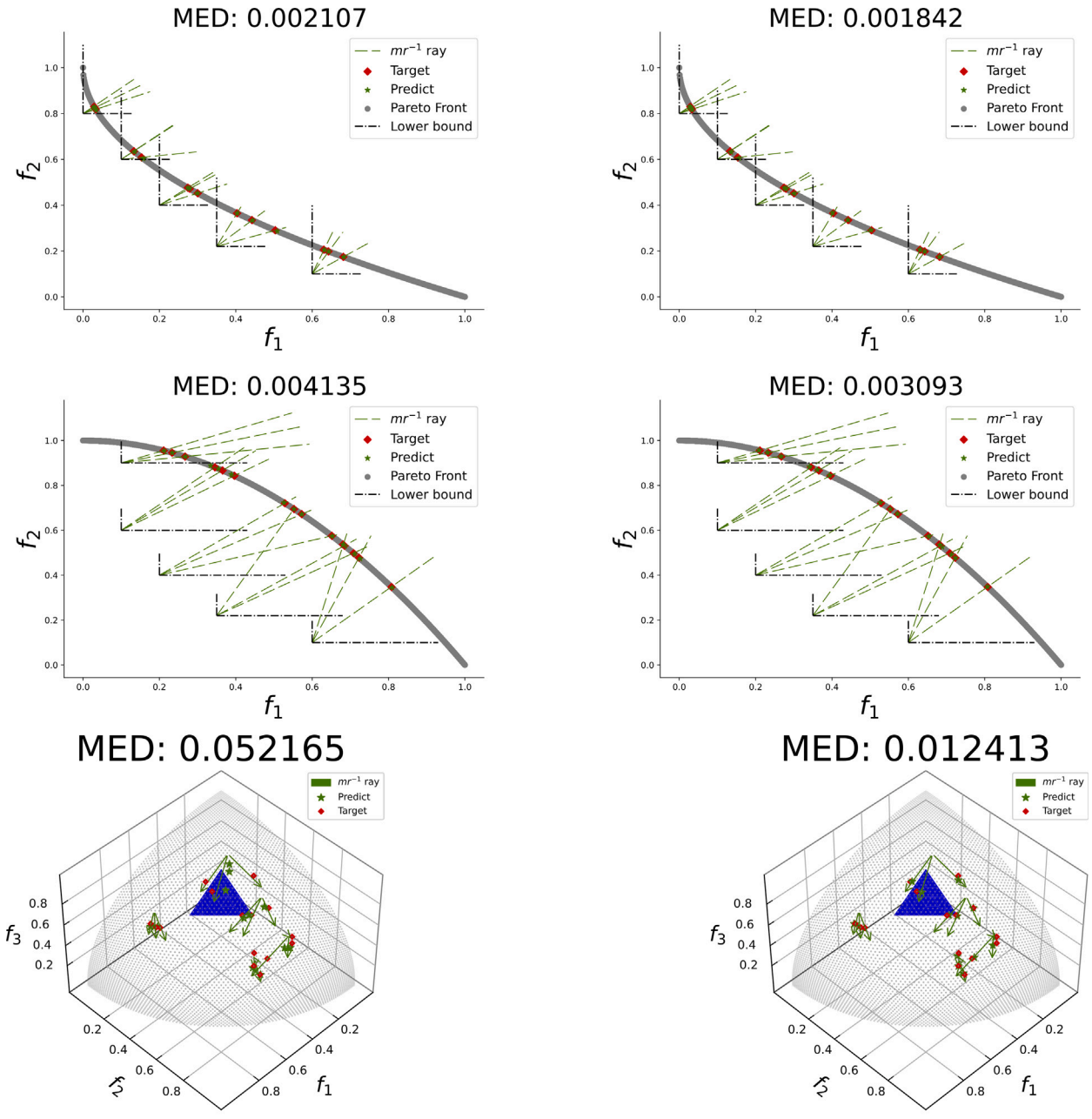


Fig. 14. Feature maps of Target network with weights that Hyper-Transformer generated.



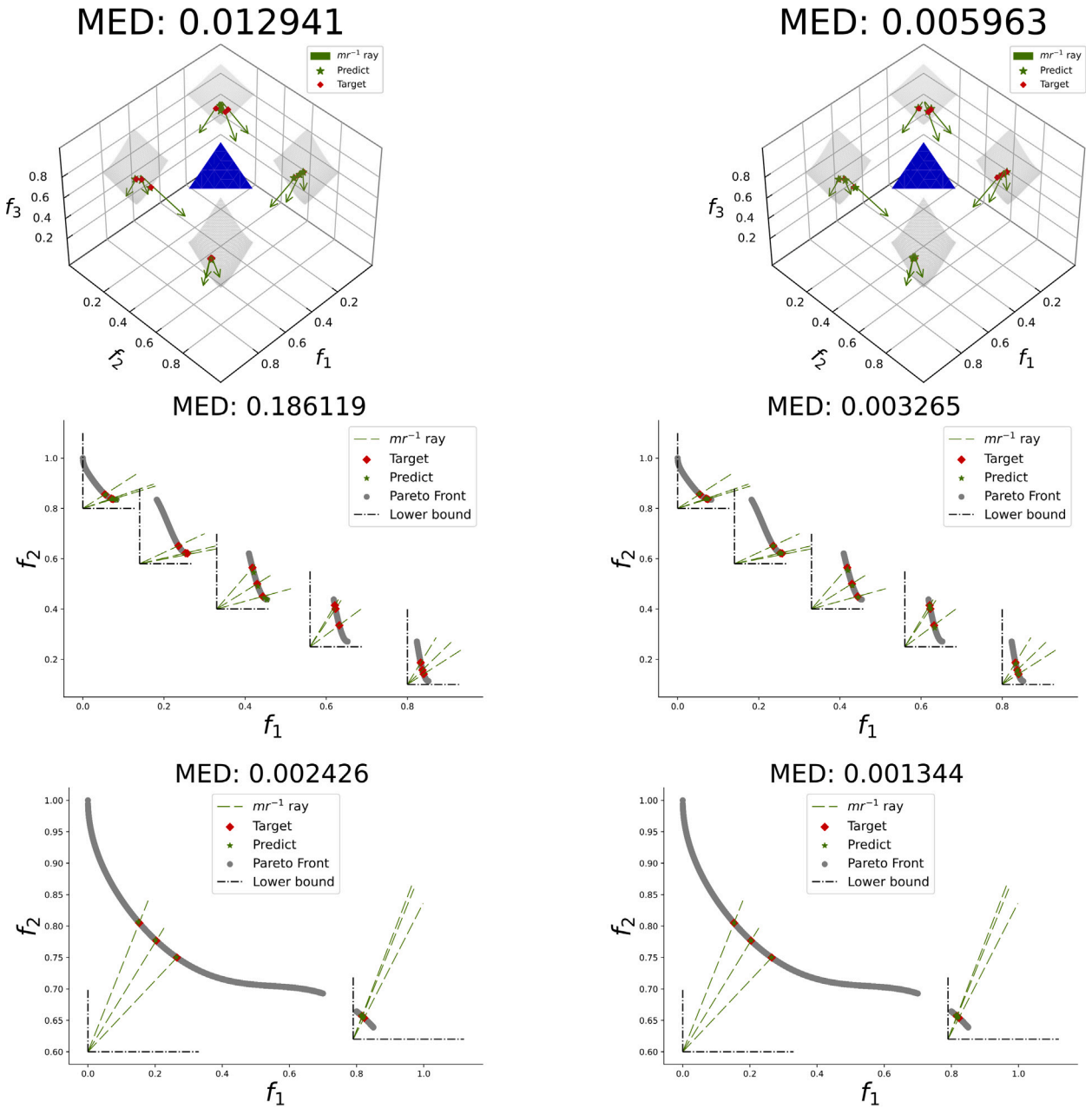
(a) Hyper-MLP (b) Hyper-Transformer

Fig. 15. The Controllable Pareto Front Learning by Split Feasibility Constraints method achieves an exact mapping between the predicted solution of Hypernetwork and the truth solution, as illustrated in Examples (CVX1) (top), (CVX2) (middle), and (CVX3) (bottom).



(a) Hyper-MLP (b) Hyper-Transformer

Fig. 16. The Controllable Pareto Front Learning by Split Feasibility Constraints method achieves an exact mapping between the predicted solution of hypernetwork and the truth solution, as illustrated in Examples (ZDT1) (top), (ZDT2) (middle), and (DTLZ2) (bottom).



(a) Join Input Hyper-Transformer

(b) Mixture of Experts Hyper-Transformer

Fig. 17. The Controllable Pareto Front Learning by Split Feasibility Constraints method achieves an exact mapping between the predicted solution of Hypernetwork and the truth solution, as illustrated in Examples (DTLZ7) (top), (ZDT3) (middle), and (ZDT3*) (bottom).

References

Anh, T. V., & Muu, L. D. (2016). A projection-fixed point method for a class of bilevel variational inequalities with split fixed point constraints. *Optimization*, *65*(6), 1229–1243.

Benoist, J. (2001). Contractibility of the efficient set in strictly quasiconcave vector maximization. *Journal of Optimization Theory and Applications*, *110*(2), 325–336.

Bian, W., Ma, L., Qin, S., & Xue, X. (2018). Neural network for nonsmooth pseudoconvex optimization with general convex constraints. *Neural Networks*, *101*, 1–14.

Binh, T. T., & Korn, U. (1997). MOBES: A multiobjective evolution strategy for constrained optimization problems. In *The third international conference on genetic algorithms, vol. 25* (p. 27).

Brooke, M., Censor, Y., & Gibali, A. (2021). Dynamic string-averaging CQ-methods for the split feasibility problem with percentage violation constraints arising in radiation therapy treatment planning. *International Transactions in Operational Research*, *30*(1), 181–205.

Byrne, C. (2002). Iterative oblique projection onto convex sets and the split feasibility problem. *Inverse Problems*, *18*(2), 441.

Byrne, C. (2003). A unified treatment of some iterative algorithms in signal processing and image reconstruction. *Inverse Problems*, *20*(1), 103.

Cao, X., Jia, S., Luo, Y., Yuan, X., Qi, Z., & Yu, K.-T. (2019). Multi-objective optimization method for enhancing chemical reaction process. *Chemical Engineering Science*, *195*, 494–506.

Censor, Y., & Elfving, T. (1994). A multiprojection algorithm using Bregman projections in a product space. *Numerical Algorithms*, *8*, 221–239.

Censor, Y., Elfving, T., Kopf, N., & Bortfeld, T. (2005). The multiple-sets split feasibility problem and its applications for inverse problems. *Inverse Problems*, *21*(6), 2071.

Censor, Y., Gibali, A., & Reich, S. (2012). Algorithms for the split variational inequality problem. *Numerical Algorithms*, *59*, 301–323.

Chen, R., & Li, K. (2023). Data-driven evolutionary multi-objective optimization based on multiple-gradient descent for disconnected Pareto fronts. In *International conference on evolutionary multi-criterion optimization* (pp. 56–70). Springer.

Chen, S., Wang, Y., Wen, Z., Li, Z., Zhang, C., Zhang, X., et al. (2023). Controllable multi-objective re-ranking with policy hypernetworks. In *Proceedings of the 29th*

ACM SIGKDD conference on knowledge discovery and data mining (pp. 3855–3864). New York, NY, USA: Association for Computing Machinery, ISBN: 9798400701030, http://dx.doi.org/10.1145/3580305.3599796.

Coello, C. C., & Sierra, M. R. (2003). A coevolutionary multi-objective evolutionary algorithm. In *The 2003 congress on evolutionary computation, 2003, vol. 1* (pp. 482–489). IEEE.

Cordonnier, J.-B., Loukas, A., & Jaggi, M. (2019). On the relationship between self-attention and convolutional layers. arXiv preprint arXiv:1911.03584.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, *2*(4), 303–314.

Das, I., & Dennis, J. (2000). Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, *8*, http://dx.doi.org/10.1137/S1052623496307510.

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2002). Scalable multi-objective optimization test problems. In *Proceedings of the 2002 congress on evolutionary computation. CEC'02 (cat. no. 02TH8600)*: *vol. 1*, (pp. 825–830). IEEE.

Ehrgott, M. (2005). *Multicriteria optimization: vol. 491*, Springer Science & Business Media.

Godwin, E. C., Izuchukwu, C., & Mewomo, O. T. (2023). Image restorations using a modified relaxed inertial technique for generalized split feasibility problems. *Mathematical Methods in the Applied Sciences*, *46*(5), 5521–5544.

Hanin, B., & Sellke, M. (2017). Approximating continuous functions by relu nets of minimal width. arXiv preprint arXiv:1710.11278.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Helbig, S. (1990). On the connectedness of the set of weakly efficient points of a vector optimization problem in locally convex spaces. *Journal of Optimization Theory and Applications*, *65*, 257–270.

Hoang, L. P., Le, D. D., Tuan, T. A., & Thang, T. N. (2023). Improving pareto front learning via multi-sample hypernetworks. In *Proceedings of the AAAI conference on artificial intelligence, vol. 37, no. 7* (pp. 7875–7883).

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366.

Ishibuchi, H., He, L., & Shang, K. (2019). Regular Pareto front shape is not realistic. In *2019 IEEE congress on evolutionary computation* (pp. 2034–2041). IEEE.

Jangir, P., Heidari, A. A., & Chen, H. (2021). Elitist non-dominated sorting Harris hawks optimization: Framework and developments for multi-objective problems. *Expert Systems with Applications*, *186*, Article 115747.

Jiang, H., Li, Q., Li, Z., & Wang, S. (2023). A brief survey on the approximation theory for sequence modelling. arXiv preprint arXiv:2302.13752.

Jiang, S., & Yang, S. (2017). A strength Pareto evolutionary algorithm based on reference direction for multiobjective and many-objective optimization. *IEEE Transactions on Evolutionary Computation*, *21*(3), 329–346.

Kim, N. T. B., & Thang, T. N. (2013). Optimization over the efficient set of a bicriteria convex programming problem. *Pacific Journal of Optimization*, *9*(1), 103–115.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Lambrinidis, G., & Tsantili-Kakoulidou, A. (2021). Multi-objective optimization methods in novel drug design. *Expert Opinion on Drug Discovery*, *16*(6), 647–658.

Li, S., Chen, X., He, D., & Hsieh, C.-J. (2021). Can vision transformers perform convolution? arXiv preprint arXiv:2111.01353.

Lin, X., Yang, Z., Zhang, Q., & Kwong, S. (2020). Controllable pareto multi-task learning. arXiv preprint arXiv:2010.06313.

Lin, X., Zhang, X., Yang, Z., & Zhang, Q. (2023). Evolutionary Pareto set learning with structure constraints. arXiv preprint arXiv:2310.20426.

Lin, X., Zhen, H.-L., Li, Z., Zhang, Q., & Kwong, S. (2019). Pareto multi-task learning. In *Thirty-third conference on neural information processing systems* (pp. 12037–12047).

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision* (pp. 3730–3738).

Liu, N., Wang, J., & Qin, S. (2022). A one-layer recurrent neural network for nonsmooth pseudoconvex optimization with quasiconvex inequality and affine equality constraints. *Neural Networks*, *147*, 1–9.

López, G., Martín-Márquez, V., Wang, F., & Xu, H.-K. (2012). Solving the split feasibility problem without prior knowledge of matrix norms. *Inverse Problems*, *28*(8), Article 085004.

Luc, D. T. (1989). Scalarization and stability. In *Theory of vector optimization* (pp. 80–108). Berlin, Heidelberg: Springer Berlin Heidelberg, ISBN: 978-3-642-50280-4, http://dx.doi.org/10.1007/978-3-642-50280-4_4.

Luc, D. T. (2005). Generalized convexity in vector optimization. In *Handbook of generalized convexity and generalized monotonicity* (pp. 195–236). Springer.

Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025.

Mahapatra, D., & Rajan, V. (2021). Exact Pareto optimal search for multi-task learning: Touring the Pareto front. arXiv preprint arXiv:2108.00597.

Mangasarian, O. L. (1994). *Nonlinear programming*. SIAM.

Momma, M., Dong, C., & Liu, J. (2022). A multi-objective / multi-task learning framework induced by Pareto stationarity. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, & S. Sabato (Eds.), *Proceedings of machine learning research*: *vol. 162*, *Proceedings of the 39th international conference on machine learning* (pp. 15895–15907). PMLR, URL https://proceedings.mlr.press/v162/momma22a.html.

Murugan, P., Kannan, S., & Baskar, S. (2009). NSGA-II algorithm for multi-objective generation expansion planning problem. *Electric Power Systems Research*, *79*(4), 622–628.

Naccache, P. (1978). Connectedness of the set of nondominated outcomes in multicriteria optimization. *Journal of Optimization Theory and Applications*, *25*, 459–467.

Navon, A., Shamsian, A., Chechik, G., & Fetaya, E. (2020). Learning the pareto front with hypernetworks. arXiv preprint arXiv:2010.04104.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, *32*.

Raychaudhuri, D. S., Suh, Y., Schulter, S., Yu, X., Faraki, M., Roy-Chowdhury, A. K., et al. (2022). Controllable dynamic multi-task architectures. In *2022 IEEE/CVF conference on computer vision and pattern recognition* (pp. 10945–10954). Los Alamitos, CA, USA: IEEE Computer Society, http://dx.doi.org/10.1109/CVPR52688.2022.01068.

Roller, S., Sukhbaatar, S., Weston, J., et al. (2021). Hash layers for large sparse models. *Advances in Neural Information Processing Systems*, *34*, 17555–17566.

Sabour, S., Frosst, N., & Hinton, G. E. (2017). Dynamic routing between capsules. http://dx.doi.org/10.48550/ARXIV.1710.09829, URL https://arxiv.org/abs/1710.09829.

Sener, O., & Koltun, V. (2018). Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, *31*.

Shazeer, N. M., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G. E., et al. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. ArXiv, arXiv:1701.06538. URL https://api.semanticscholar.org/CorpusID:12462234.

Silberman, N., Hoiem, D., Kohli, P., & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *European conference on computer vision* (pp. 746–760). Springer.

Stark, H., Yang, Y., & Yang, Y. (1998). *Vector space projections: A numerical approach to signal and image processing, neural nets, and optics*. John Wiley & Sons, Inc..

Thang, T. N., & Hai, T. N. (2022). Self-adaptive algorithms for quasiconvex programming and applications to machine learning. http://dx.doi.org/10.48550/ARXIV.2212.06379, URL https://arxiv.org/abs/2212.06379.

Thang, T. N., Solanki, V. K., Dao, T. A., Thi Ngoc Anh, N., & Van Hai, P. (2020). A monotonic optimization approach for solving strictly quasiconvex multiobjective programming problems. *Journal of Intelligent & Fuzzy Systems*, *38*(5), 6053–6063.

Tuan, T. A., Hoang, L. P., Le, D. D., & Thang, T. N. (2023). A framework for controllable pareto front learning with completed scalarization functions and its applications. *Neural Networks*.

Tuy, H. (2000). Monotonic optimization: Problems and solution approaches. *SIAM Journal on Optimization*, *11*(2), 464–494.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Vijayakumar, S. (2000). The sarcos dataset. http://dx.doi.org/10.48550/ARXIV.1708.07747, URL http://www.gaussianprocess.org/gpml/data.

Vuong, N. D., & Thang, T. N. (2023). Optimizing over Pareto set of semistrictly quasiconcave vector maximization and application to stochastic portfolio selection. *Journal of Industrial and Management Optimization*, *19*(3), 1999–2019.

Wang, H., Olhofer, M., & Jin, Y. (2017). A mini-review on preference modeling and articulation in multi-objective optimization: current status and challenges. *Complex & Intelligent Systems*, *3*, 233–245.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.

Xu, C., Chai, Y., Qin, S., Wang, Z., & Feng, J. (2020). A neurodynamic approach to nonsmooth constrained pseudoconvex optimization problem. *Neural Networks*, *124*, 180–192.

Xu, J., Chi, E. C., Yang, M., & Lange, K. (2018). A majorization–minimization algorithm for split feasibility problems. *Computational Optimization and Applications*, *71*, 795–828.

Xunhua, G. (1994). Connectedness of the efficient solution set of a convex vector optimization in normed spaces. *Nonlinear Analysis. Theory, Methods & Applications*, *23*(9), 1105–1114.

Yen, L. H., Huyen, N. T. T., & Muu, L. D. (2019). A subgradient algorithm for a class of nonlinear split feasibility problems: application to jointly constrained Nash equilibrium models. *Journal of Global Optimization*, *73*, 849–868.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S. J., & Kumar, S. (2019). Are transformers universal approximators of sequence-to-sequence functions? arXiv preprint arXiv:1912.10077.

Yun, C., Chang, Y.-W., Bhojanapalli, S., Rawat, A. S., Reddi, S., & Kumar, S. (2020). O (n) connections are expressive enough: Universal approximability of sparse transformers. *Advances in Neural Information Processing Systems*, *33*, 13783–13794.

Zhao, Y., Wang, L., Yang, K., Zhang, T., Guo, T., & Tian, Y. (2021). Multi-objective optimization by learning space partitions. arXiv preprint arXiv:2110.03173.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, *8*(2), 173–195.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257–271.