Multi-objective Learning Using HV Maximization

Deist, Timo M.; Grewal, Monika; Dankers, Frank J.W.M.; Alderliesten, Tanja; Bosman, Peter A.N.

DOI
10.1007/978-3-031-27250-9_8

Publication date
2023

Document Version
Final published version

Published in
Evolutionary Multi-Criterion Optimization - 12th International Conference, EMO 2023, Proceedings

Multi-objective Learning Using HV Maximization

Timo M. Deist[1](✉), Monika Grewal[1], Frank J.W.M. Dankers[2],
Tanja Alderliesten[2], and Peter A.N. Bosman[1,3]

[1] Centrum Wiskunde and Informatica, Amsterdam, The Netherlands
timo.deist@cwi.nl
[2] Leiden University Medical Center, Leiden, The Netherlands
[3] Delft University of Technology, Delft, The Netherlands

Abstract. Real-world problems are often multi-objective, with decision-makers unable to specify a priori which trade-off between the conflicting objectives is preferable. Intuitively, building machine learning solutions in such cases would entail providing multiple predictions that span and uniformly cover the Pareto front of all optimal trade-off solutions. We propose a novel approach for multi-objective training of neural networks to approximate the Pareto front during inference. In our approach, we train the neural networks multi-objectively using a dynamic loss function, wherein each network's losses (corresponding to multiple objectives) are weighted by their hypervolume maximizing gradients. Experiments on different multi-objective problems show that our approach returns well-spread outputs across different trade-offs on the approximated Pareto front without requiring the trade-off vectors to be specified a priori. Further, results of comparisons with the state-of-the-art approaches highlight the added value of our proposed approach, especially in cases where the Pareto front is asymmetric.

Keywords: Multi-objective optimization · Neural networks · Pareto front · Hypervolume · Multi-objective learning

1 Introduction

Multi-objective (MO) optimization refers to finding Pareto optimal solutions for multiple, often conflicting, objectives. In MO optimization, a solution is Pareto optimal if none of the objectives can be improved without a simultaneous detriment in performance on at least one of the other objectives [35]. MO optimization is used for MO decision-making in many real-world applications [32] e.g., e-commerce recommendation [21], treatment plan optimization [25,27], and aerospace engineering [29]. In this paper, we focus on learning-based MO decision-making i.e., MO training of machine learning (ML) models so that MO decision-making is possible during inference. Specifically, we focus on training neural networks to generate a finite number of Pareto optimal solutions for each

T. M. Deist and M. Grewal—contributed equally.

M. Emmerich et al. (Eds.): EMO 2023, LNCS 13970, pp. 103–117, 2023.
https://doi.org/10.1007/978-3-031-27250-9_8

sample[1], so that they together provide a discrete approximation of the Pareto front[2].

The most straightforward approach for MO optimization is linear scalarization, i.e., optimizing a linear combination of different objectives according to scalarization weights. The scalarization weights are based on the desired trade-off between multiple objectives which is often referred to as 'user preference'. A major issue with linear scalarization is that user preferences cannot always be straightforwardly translated to linear scalarization weights. Recently proposed approaches have tackled this issue and find solutions on the average Pareto front for conflicting objectives according to a pre-specified user preference vector [20,23]. However, in many real-world problems, the user preference vector cannot be known a priori and decision-making is only possible *a posteriori*, i.e., after multiple solutions are generated that are (near) Pareto optimal for a specific sample[3]. For example, in neural style transfer [11] where photos are manipulated to imitate an art style from a selected painting, the user preference between the amount of semantic information (the photo's content) and artistic style can only be decided by looking at multiple different resultant images on the Pareto front (Fig. 5). Moreover, defining multiple trade-offs, typically by defining multiple scalarizations, to evenly cover the Pareto front is far from trivial, e.g., if the Pareto front is asymmetric. Here, we define asymmetry in Pareto fronts as asymmetry in the distribution of Pareto optimal solutions in the objective space on either side of the 45°-line, the line which represents the trade-off of equal marginal benefit along all objectives (see Pareto fronts in Fig. 1). We demonstrate and discuss this further in Sect. 4. To enable a posteriori decision-making per sample, multiple solutions spanning the Pareto front need to be generated without requiring the user preference vectors beforehand.

Despite many developments in the direction of MO training of neural networks with pre-specified user preferences, research on MO learning allowing for a posteriori decision-making is still scarce. Here, we present a novel method to multi-objectively train a set of neural networks to this end, leveraging the concept of hypervolume. Although we present our approach for training neural networks, the proposed formulation can be used for a wide range of ML models.

The hypervolume (HV) – the objective space dominated by a given set of solutions – is a popular metric to compare the quality of different sets of solutions approximating the Pareto front. It has its origins in the field of evolutionary algorithms [39], which are commonly accepted to be state of the art for multi-objective optimization. Theoretically, if the HV is maximal for a set of solutions, these solutions are on the Pareto front [9]. Additionally, HV not only encodes the proximity of a set of solutions to the Pareto front but also their diversity, which means that HV maximization provides a straightforward way for finding diverse solutions on the Pareto front. Therefore, we use hypervolume maximization for

[1] Note that, during inference, only *near* Pareto optimal solutions can be generated due to the generalization gap between training and inference.

[2] The Pareto front is the set of all Pareto optimal solutions in objective space.

[3] For more information on a posteriori decision-making, please refer to [14].

MO training of neural networks. We train the set of neural networks with a dynamically weighted combination of loss functions corresponding to multiple objectives, wherein the weight of each loss is based on the HV-maximizing gradients. In summary, our paper has the following main contributions:

- An MO approach for training neural networks
    - using gradient-based HV maximization
    - predicting Pareto optimal and diverse solutions on the Pareto front per sample without requiring specification of user preferences
    - enabling learning-based a posteriori decision-making.
- Experiments to demonstrate the added value of the proposed approach, specifically in asymmetric Pareto fronts.

2 Related Work

MO optimization has been used in machine learning for hyperparameter tuning of machine learning models [2,18], multi-objective classification of imbalanced data [33], and discovering the complete Pareto set starting from a single Pareto optimal solution [22]. [15] used MO optimization for finding configurations of deep neural networks for conflicting objectives. [13] proposed optimizing the weights of an autoencoder multi-objectively for finding the Pareto front of sparsity and reconstruction error. [24] used the Tchebycheff procedure for multi-objective optimization of a single neural network with multiple heads for multi-task text classification. Although we do not focus on these directions, our proposed approach can be used in similar applications.

MO training of a set of neural networks such that their predictions approximate the Pareto front of multiple objectives is closely related to the work presented in this paper. Similar to our work, [20,23] describe approaches with dynamic loss formulations to train multiple networks such that the predictions from these multiple networks together approximate the Pareto front. However, in these approaches, the trade-offs between conflicting objectives are required to be known in advance whereas our proposed approach does not require knowing the set of trade-offs beforehand. Other approaches [19,28] involve training a "hypernetwork" to predict the weights of another neural network based on a user-specified trade-off. Recently, it has been proposed to condition a neural network for an input user preference vector to allow for predicting multiple points near the Pareto front during inference [31]. While these approaches can approximate the Pareto front by iteratively predicting neural network weights or outputs based on multiple user preference vectors, the process of sampling the user preference vectors may still be intensive for an unknown Pareto front shape. Another approach proposes growing dense Pareto fronts from sparse Pareto optimal solutions [22], for which our approach can provide baseline solutions.

Gradient-based HV maximization is a key component of our approach. [26] have described gradient-based HV maximization for single networks and formulated a dynamic loss function treating each sample's error as a separate loss. [1]

applied this concept for training in generative adversarial networks. HV maximization is also applied in reinforcement learning [34,38]. While these approaches use HV maximizing gradients to optimize the weights of a single neural network, our proposed approach formulates a dynamic loss based on HV maximizing gradients for a *set* of neural networks. Different from our approach, other concurrent approaches for HV maximization are based on transformation to $(m-1)D$ (where m is the number of objectives) integrals by use of polar coordinates [7], random scalarization [12], and a q-Expected hypervolume improvement function [3].

3 Approach

MO learning of a network parameterized by a vector θ can be formulated as minimizing a vector of n losses $\mathcal{L}(\theta, s_k) = [L_1(\theta, s_k), \ldots, L_n(\theta, s_k)]$ for a given set of samples $S = \{s_1, \ldots, s_k, \ldots, s_{|S|}\}$. These loss functions form the loss space, wherein the subspace attainable by a sample's losses is bounded by its Pareto front. To learn multiple networks with loss vectors on each sample's Pareto front, we replace θ by a set of parameters $\Theta = \{\theta_1, \ldots, \theta_p\}$, where each parameter vector θ_i represents a network. The corresponding set of loss vectors is $\{\mathcal{L}(\theta_1, s_k), \ldots, \mathcal{L}(\theta_p, s_k)\}$ and is represented by a stacked loss vector $\mathfrak{L}(\Theta, s_k) = [\mathcal{L}(\theta_1, s_k), \ldots, \mathcal{L}(\theta_p, s_k)]$. **Our goal is to learn a set of p networks such that loss vectors in $\mathfrak{L}(\Theta, s_k)$ corresponding to the networks' predictions for sample s_k lie on and span the Pareto front of loss functions for sample** s_k. In other words, each network's loss vector is Pareto optimal and lies in a distinct subsection of the Pareto front for each sample. To achieve this goal, we train networks so that the loss subspace Pareto dominated by the networks' predictions (i.e., the HV) is maximal.

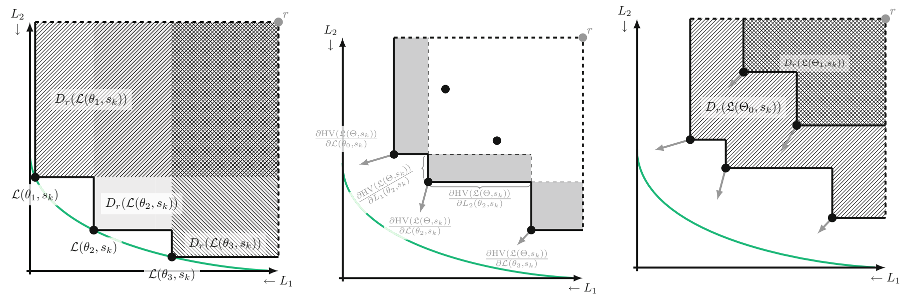
The HV of a loss vector $\mathcal{L}(\theta_i, s_k)$ for a sample s_k is the volume of the subspace $D_r(\mathcal{L}(\theta_i, s_k))$ in loss space dominated by $\mathcal{L}(\theta_i, s_k)$. This is illustrated in Fig. 1a. To keep this volume finite, the HV is computed with respect to a reference point r which bounds the space to the region of interest[4]. Subsequently, the HV of multiple loss vectors $\mathfrak{L}(\Theta, s_k)$ is the HV of the union of dominated subspaces $D_r(\mathcal{L}(\theta_i, s_k)), \forall i \in \{1, 2, ..., p\}$. The MO learning problem to maximize the mean HV of all $|S|$ samples is as follows:

$$\text{maximize} \frac{1}{|S|} \sum_{k=1}^{|S|} \text{HV}\left(\mathfrak{L}(\Theta, s_k)\right) \tag{1}$$

The update direction of gradient ascent for parameter vector θ_i of network i is:

$$\frac{\partial \frac{1}{|S|} \sum_{k=1}^{|S|} \text{HV}(\mathfrak{L}(\Theta, s_k))}{\partial \theta_i} \tag{2}$$

[4] The reference point is generally set to large coordinates in loss space to ensure that it is always dominated by all loss vectors.


L_2
r
$D_r(\mathcal{L}(\theta_1, s_k))$
$\mathcal{L}(\theta_1, s_k)$
$D_r(\mathcal{L}(\theta_2, s_k))$
$\mathcal{L}(\theta_2, s_k)$
$D_r(\mathcal{L}(\theta_3, s_k))$
$\mathcal{L}(\theta_3, s_k)$
$\leftarrow L_1$
L_2
r
$\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial \mathcal{L}(\theta_0,s_k)}$
$\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial L_1(\theta_2,s_k)}$
$\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial L_2(\theta_2,s_k)}$
$\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial \mathcal{L}(\theta_2,s_k)}$
$\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial \mathcal{L}(\theta_3,s_k)}$
$\leftarrow L_1$
L_2
r
$D_r(\mathfrak{L}(\Theta_1, s_k))$
$D_r(\mathfrak{L}(\Theta_0, s_k))$
$\leftarrow L_1$

(a) Dominated subspaces

(b) HV gradients

(c) Domination-ranked fronts

Fig. 1. **(a)** Three Pareto optimal loss vectors $\mathcal{L}(\theta_i, s)$ on the Pareto front (green) with dominated subspaces $D_r(\mathcal{L}(\theta_i, s_k))$ with respect to reference point r. The union of dominated subspaces is the dominated hypervolume (HV) of $\mathfrak{L}(\Theta, s_k)$. **(b)** Gray markings illustrate the computation of the HV gradients $\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s))}{\partial \mathcal{L}(\theta_i,s)}$ (gray arrows) in the three non-dominated solutions. **(c)** The same five solutions grouped into two domination-ranked fronts Θ_0 and Θ_1 with corresponding HV, equal to their dominated subspaces $D_r(\mathcal{L}(\theta_i, s_k))$, and HV gradients. (Color figure online)

By exploiting the chain rule decomposition of HV gradients as described in [8], the update direction in Eq. (2) for parameter vector θ_i of network i can be written as follows:

$$\frac{1}{|S|} \sum_{k=1}^{|S|} \frac{\partial \mathrm{HV}\left(\mathfrak{L}(\Theta, s_k)\right)}{\partial \mathcal{L}(\theta_i, s_k)} \cdot \frac{\partial \mathcal{L}(\theta_i, s_k)}{\partial \theta_i} \quad \forall i \in \{1, \ldots, p\} \tag{3}$$

The dot product of $\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial \mathcal{L}(\theta_i,s_k)}$ (the HV gradients with respect to loss vector $\mathcal{L}(\theta_i, s_k)$) in loss space, and $\frac{\partial \mathcal{L}(\theta_i,s_k)}{\partial \theta_i}$ (the matrix of loss vector gradients in the network i's parameters θ_i) in parameter space, can be decomposed to

$$\frac{1}{|S|} \sum_{k=1}^{|S|} \sum_{j=1}^{n} \frac{\partial \mathrm{HV}\left(\mathfrak{L}(\Theta, s_k)\right)}{\partial L_j(\theta_i, s_k)} \frac{\partial L_j(\theta_i, s_k)}{\partial \theta_i} \quad \forall i \in \{1, \ldots, p\} \tag{4}$$

where $\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial L_j(\theta_i,s_k)}$ is the scalar HV gradient in the single loss function $L_j(\theta_i, s_k)$, and $\frac{\partial L_j(\theta_i,s_k)}{\partial \theta_i}$ are the gradients used in gradient descent for single-objective training of network i for loss $L_j(\theta_i, s_k)$. Based on Eq. (4), one can observe that mean HV maximization of loss vectors from a set of p networks for $|S|$ samples can be achieved by weighting their gradient descent directions for loss functions $L_j(\theta_i, s_k)$ with their corresponding HV gradients $\frac{\partial \mathrm{HV}(\mathfrak{L}(\Theta,s_k))}{\partial L_j(\theta_i,s_k)}$ for all i, j. In other terms, the MO learning of a set of p networks can be achieved by minimizing[5] the following dynamic loss function for each network i:

[5] Minimizing the dynamic loss function maximizes the HV because the reference point r is in the positive quadrant ("to the right and above 0").

$$\frac{1}{|S|}\sum_{k=1}^{|S|}\sum_{j=1}^{n}\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\varTheta, s_k)\right)}{\partial L_j(\theta_i, s_k)}L_j(\theta_i, s_k) \quad \forall i \in \{1,\ldots,p\} \tag{5}$$

The computation of the HV gradients $\frac{\partial \mathrm{HV}(\mathfrak{L}(\varTheta,s_k))}{\partial L_j(\theta_i,s_k)}$ is illustrated in Fig. 1b. These HV gradients are equal to the marginal decrease in the subspace dominated only by $\mathcal{L}(\theta_i, s_k)$ when increasing $L_j(\theta_i, s_k)$.

Note that Eq. 5 maximizes the HV for each sample's losses instead of first averaging losses on the set of samples as commonly done in learning tasks. Consequently, the neural networks are trained on each sample's Pareto front separately, instead of on the front of averages losses. In [5], we experimentally illustrate that learning an average front may lead to undesired results for non-convex fronts.

3.1 HV Maximization of Domination-Ranked Fronts

A relevant caveat of gradient-based HV maximization is that HV gradients $\frac{\partial \mathrm{HV}(\mathfrak{L}(\varTheta,s_k))}{\partial L_j(\theta_i,s_k)}$ in strongly dominated solutions are zero (because no movement in any direction will affect the HV, Fig. 1b) and in weakly dominated solutions are undefined [8]. To resolve this issue, we follow [37]'s approach, which avoids the problem of dominated solutions by sorting all loss vectors into separate fronts $\varTheta_l$ of mutually non-dominated loss vectors and optimizing each front separately (Fig. 1c). l is the domination rank, and $q(i)$ is the mapping of network i to domination rank l. By maximizing the HV of each front, trailing fronts with domination rank > 0 eventually merge with the non-dominated front $\varTheta_0$ and a single front is maximized by determining optimal locations for each loss vector on the Pareto front.

Furthermore, we normalize the HV gradients $\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\varTheta_{q(i)},s_k)\right)}{\partial \mathcal{L}(\theta_i,s_k)}$ as in [6] such that their length in loss space is 1. The dynamic loss function with domination-ranking of fronts and HV gradient normalization is:

$$\frac{1}{|S|}\sum_{k=1}^{|S|}\sum_{j=1}^{n}\frac{1}{w_i}\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\varTheta_{q(i)}, s_k)\right)}{\partial L_j(\theta_i, s_k)}L_j(\theta_i, s_k) \quad \forall i \in \{1,\ldots,p\} \tag{6}$$

where $w_i = \left\lVert \frac{\partial \mathrm{HV}\left(\mathfrak{L}(\varTheta_{q(i)},s_k)\right)}{\partial \mathcal{L}(\theta_i,s_k)} \right\rVert$.

3.2 Implementation

We implemented the HV maximization of losses from multiple networks, as defined in Eq. (6), in Python[6]. The neural networks were implemented using the PyTorch framework [30]. We used [10]'s HV computation reimplemented by Simon Wessing, available from [36]. The HV gradients $\frac{\partial \mathrm{HV}\left(\mathfrak{L}(\varTheta_{q(i)},s_k)\right)}{\partial L_j(\theta_i,s_k)}$ were

[6] Code is available at https://github.com/timodeist/multi_objective_learning.

computed following the algorithm by [8]. Networks with identical losses were assigned the same HV gradients. For non-dominated networks with one or more identical losses (which can occur in training with three or more losses), the left- and right-sided limits of the HV function derivatives are not the same [8], and they were set to zero. Non-dominated sorting was implemented based on [4].

3.3 A Toy Example

Consider an example of MO regression with two conflicting objectives: given a sample $x_k \in S$, from input variable $X \in [0, 2\pi]$, predict the corresponding output z_k that matches y_k^1 from target variable Y_1 and y_k^2 from target variable Y_2, simultaneously. The relation between X, Y_1, and Y_2 is as follows:

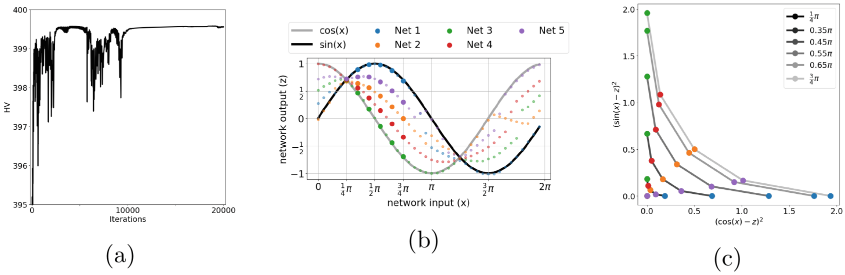
$$Y_1 = \cos(X), \quad Y_2 = \sin(X)$$

The corresponding mean square error formulations for loss functions are $L_j = \frac{1}{|S|}\sum_{k=1}^{|S|}(y_k^j - z_k)^2; j \in \{1, 2\}$. We generated 200 samples of input and target variables for training and validation each. We trained five neural networks for 20000 iterations each with two fully connected linear layers of 100 neurons followed by ReLU nonlinearities. Figure 2a shows the HV over training iterations for the set of networks, which stabilizes visibly. Figure 2b shows predictions (y-axis) for validation samples evenly sampled from $[0, 2\pi]$ (x-axis). These predictions by five neural networks constitute Pareto front approximations for each sampled x_k, and correspond to precise predictions for $\cos(X)$ and $\sin(X)$, and trade-offs between both target functions. A network may generate predictions with changing trade-offs for different samples, as demonstrated Networks 2–5 in Fig. 2b for $x \in [\frac{3/2}{\pi}, 2\pi]$. Figure 2c shows the predictions for the highlighted samples in Fig. 2b in loss space, wherein they seem to be evenly distributed on the approximated Pareto front. It becomes clear from Figs. 2b & 2c that each x_k has a differently sized Pareto front which the networks are able to predict. Figure 2c also demonstrates an a posteriori decision-making scenario. Upon visualizing the different Pareto fronts per sample, a user might decide to select predictions corresponding to different trade-offs for different samples.

4 Experiments

We performed experiments with two MO problems: MO regression with differently shaped Pareto fronts and neural style transfer.[7] We compared the performance of our approach with **linear scalarization** and two state-of-the-art approaches:

[7] Additional experiments are provided in [5]: multi-observer medical image segmentation, MO regression with three losses, multi-style transfer, and a counter-example for initial loss normalization.



(a)

(b)

(c)

Fig. 2. MO regression on two losses. (a) HV values for a set of networks over training iterations. (b) Network outputs for $X \in [0, 2\pi]$. (c) Generated Pareto front predictions for a selection of six samples from $[\frac{1}{4}\pi, \frac{3}{4}\pi]$ in loss space.

Pareto MTL [20] and **EPO** [23]. Pareto MTL and EPO try to find Pareto optimal solutions on the average Pareto front for a given trade-off vector using dynamic loss functions. For a consistent comparison, we used the trade-offs used in the original experiments of EPO for Pareto MTL, EPO, and as fixed weights in linear scalarization.

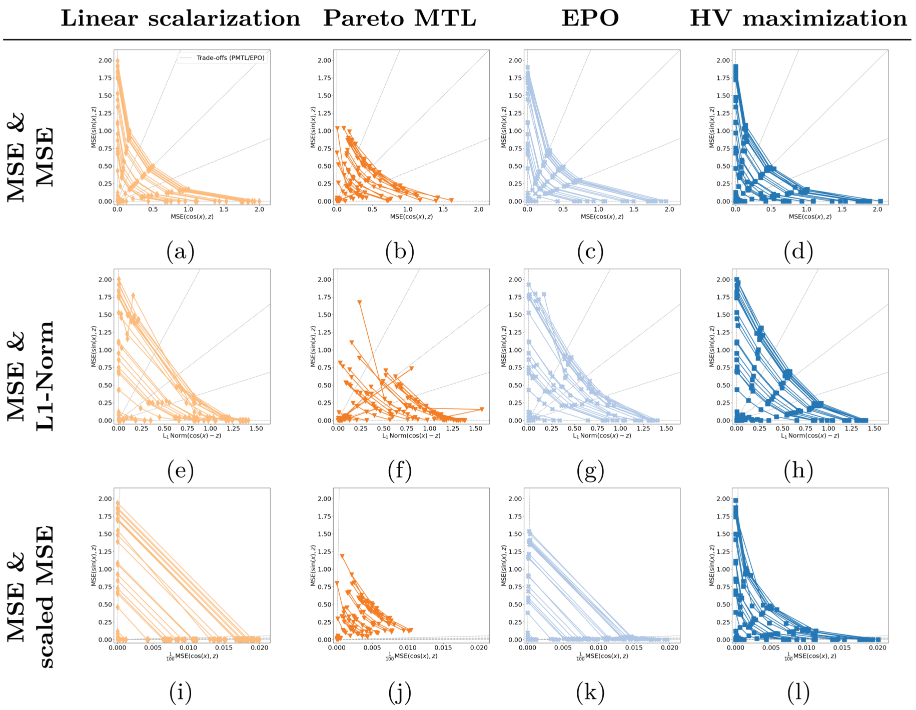
Experiments were run on systems using Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz with NVIDIA GeForce RTX 2080Ti, or Intel(R) Core(R) i5-3570K @ 3.40 Ghz with NVIDIA GeForce GTX 1060 6 GB. For training, the Adam optimizer [17] was used. The learning rate and β_1 of Adam were tuned for each approach separately based on the maximal HV of validation loss vectors.

4.1 MO Regression

We considered three cases for the MO regression toy problem described in Sect. 3.3 each demonstrating a different Pareto front shape: the symmetric case with two MSE losses as in Fig. 2, and two asymmetric cases each with MSE as one loss and L1-norm or MSE scaled by $\frac{1}{100}$ as the second loss. The reference point for our proposed approach was set to $(20, 20)$.

Figure 3 shows Pareto front approximations for all three cases. Figures 3a & 3c show that fixed linear scalarizations and EPO produce networks generating well-distributed outputs with low losses that predict a sample's symmetric Pareto front for two conflicting MSE losses. The positions on the front approximated by linear scalarization seem to be far from the pre-specified trade-offs (gray lines). This is expected because, by definition of linear scalarization, the solutions should lie on the approximated Pareto front where the tangent is perpendicular to the search direction specified by the trade-offs. For Pareto MTL, networks are clustered closer to the center of the approximated Pareto front.

Optimizing MSE and L1-Norm (Figs. 3e–3h) results in an asymmetric Pareto front approximation. The predictions by our HV maximization-based approach remain well distributed across the fronts. EPO also still provides a decent spread albeit less uniform across samples whereas linear scalarization and Pareto MTL tend to both or mostly the lower extrema, respectively.



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

Fig. 3. Pareto front approximations on a random subset of validation samples by sets of five neural networks trained using four approaches. Three different pairs of loss functions are used: (a–d) MSE and MSE, (e–h) MSE and L1-Norm, and (i–l) MSE and scaled MSE.

The difficulty of manually pre-specifying the trade-offs without knowledge of the Pareto front becomes more evident when optimizing losses with highly different scales (Figs. 3i–3l). The pre-specified trade-offs do not evenly cover the Pareto fronts. Consequently, the networks trained by EPO do not cover the Pareto front evenly despite following the pre-specified trade-offs. Further, the networks optimized by Pareto MTL cover only the upper part of the fronts. Networks trained with fixed linear scalarizations tend towards both extrema. On the other hand, our approach trains networks that follow well-distributed trade-offs on the Pareto front. Normalizing losses from differing scales as in Figs. 3i–3l might not sufficiently improve methods based on pre-specified trade-offs (Pareto MTL, EPO) or fixed linear scalarizations [5].

The mean HV over 200 validation samples is computed for all approaches and Table 1 displays the median and inter-quartile ranges (IQR) over 25 runs. The magnitude of the HV is largely determined by the position of the reference point. For $r = (20, 20)$ the maximal HV equals 400 minus the area bounded by the utopian point $(0, 0)$ and a sample's Pareto front. Even poor approximations of a sample's Pareto front can yield a HV ≥ 390. For these reasons, HVs in Table 1 appear large and minuscule differences between HVs are relevant. As

Table 1. Comparison of HV across different approaches. The maximal median HV in each column is **highlighted**. Small increases in HV close to the maximum (10^6 or 400) matter: see Sect. 4.1. A statistically significant one-sided Wilcoxon signed rank test with correction for multiple comparison is indicated by: LS vs HV max. (*), PMTL vs HV max. (†), and EPO vs HV max. (‡). **Columns 1–3:** Median (inter-quartile range) values of the mean HV of the approximated Pareto fronts for 200 validation samples from 25 runs of MO regression problem are reported. **Column 4:** Median (inter-quartile range) HV of the approximated Pareto fronts for the 25 image sets used in neural style transfer are reported.

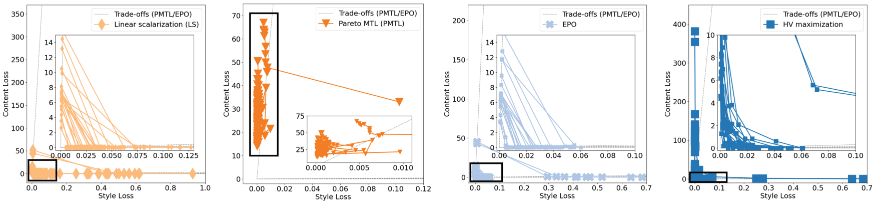
	MSE & MSE	MSE & L1-Norm	MSE & scaled MSE	Style & content
Linear scalarization (LS)	**399.5929*** **(399.5776, 399.6018)**	399.2909 (399.2738, 399.3045)	399.9859 (399.9857, 399.9864)	999990.7699 (999988.6580, 999992.5850)
Pareto MTL (PMTL)	397.1356 (396.3212, 397.6288)	392.2956 (392.0377, 393.4942)	398.3159 (397.4799, 398.6699)	997723.8748 (997583.5152, 998155.6837)
EPO	399.5135 (399.5051, 399.5348)	399.0884 (398.998, 399.1743)	399.9885 (399.9883, 399.9889)	999988.4297 (999984.4808, 999989.8338)
HV maximization	399.5823$^{\dagger\ \ddagger}$ (399.5619, 399.6005)	**399.3795**$^{*\ \dagger\ \ddagger}$ **(399.3481, 399.4039)**	**399.9954**$^{*\ \dagger\ \ddagger}$ **(399.9927, 399.9957)**	**999999.7069 (999999.4543, 999999.8266)**$^{*\ \dagger\ \ddagger}$

expected, our approach finds higher HV values for the case of asymmetric front shapes (Table 1 columns 2 and 3, and Figs. 3e–3l). In case of the symmetric front shape (Fig. 3a), since the pre-specified trade-offs appear to span the Pareto front shape well, linear scalarization's training based on fixed loss weights is more efficient than training on a dynamic loss with varying weights as used by HV maximization. This increased efficiency of training using fixed weights that are suitable for symmetric MSE losses presumably results in a slightly higher HV for linear scalarization (Table 1 column 1).

4.2 Neural Style Transfer

We further considered the MO optimization problem of neural style transfer as defined in [11] (we reused and adjusted Pytorch's neural style transfer implementation [16]), where pixels of an image are optimized to minimize content loss (semantic similarity with a target image) and style loss (artistic similarity with a style image) simultaneously. We performed experiments with 25 image pairs (image sources as in [5]), obtained by combining 5 content and 10 style images to generate 6 solutions on the Pareto front. The reference point in our approach was chosen as (100, 10000) based on preliminary runs.

Figure 4 shows the obtained Pareto front estimates for 25 image sets by each approach. Linear scalarization (a) and EPO (c) determine solutions close to or on the chosen user preferences which, however, do not diversely cover the range of possible trade-offs. Pareto MTL (b) achieves sets of clustered and partly



(a) (b) (c) (d)

Fig. 4. Pareto front estimates in loss space by different approaches for neural style transfer using four approaches: (a) Linear scalarization (b) Pareto MTL, (c) EPO, and (d) HV maximization. Sections within the black frames are magnified.

dominated solutions, which do not cover trade-offs with low content loss. On the other hand, HV maximization (d) returns Pareto front estimates that broadly cover diverse trade-offs between style and content loss across different image sets without having to specify user preferences. This is also reflected in the significantly larger median HVs reported in Table 1.

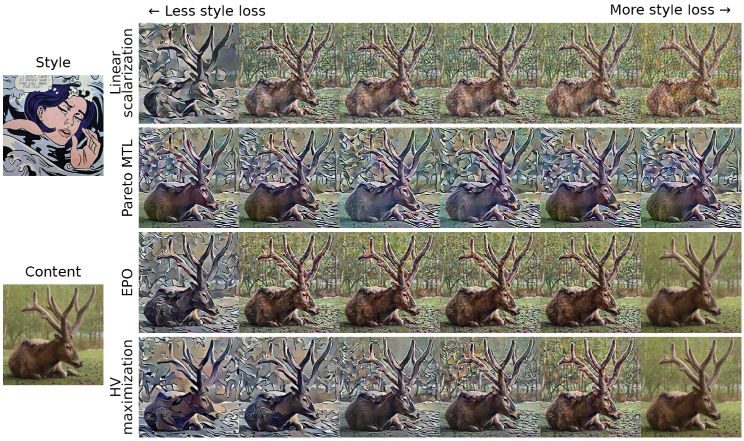


Fig. 5. Neural style transfer example by all four approaches for one image set.

Figure 5 shows the images generated by each approach for one of the image sets. This case was manually selected for its aesthetic appeal.[8] The images seen here match observations from Fig. 4, e.g., Pareto MTL's images show little diversity in style and content, many images by linear scalarization of EPO have too little style match ('uninteresting' images), and images by HV maximization show most interesting diversity.

[8] Generated images for all 25 image sets are available at https://github.com/timodeist/multi_objective_learning.

5 Discussion

We have proposed an approach to train a set of neural networks such that they jointly predict Pareto front approximations for each sample during inference, without requiring user-specified trade-offs. Our approach translates the concept of gradient-based HV maximization from MO optimization to MO learning. We provide experimental comparisons with existing approaches that require a priori specification of the trade-offs. The results highlight the advantage of our HV maximization approach, especially in MO problems that exhibit asymmetric Pareto front.

Our HV maximization based approach does not require specifying p trade-offs a priori (based on the number of predictions, p, required on the Pareto front), which essentially are $p(n-1)$ hyperparameters of the learning process for n losses. Choosing these trade-offs well requires knowledge of the Pareto front shapes, which is often not known a priori. HV maximization, however, introduces the n-dimensional reference point r and thus n additional hyperparameters. However, choosing a reference point such that the entire Pareto front gets approximated is not complex. It often suffices to use losses of randomly initialized networks rescaled by a factor ≥ 1 as the reference point. If only a specific section of the Pareto front is relevant and this is known a priori, the reference point can be chosen so that the Pareto front approximation only spans the chosen section.

HV-based training for sets of neural networks can, in theory, be applied to any number of networks, p, and loss functions, n. In practice, the time complexity of exact HV (exponential in n, [10]) and HV gradient (quadratic in p with $n \leq 4$, [8]) computations is limiting but may be overcome by algorithmic improvements using, e.g., HV approximations. Further, we train a separate network corresponding to each prediction. This increases computational load linearly if more predictions on the Pareto front are desired. We train separate networks instead of one multi-headed network for the sake of simplicity in experimentation and clarity when demonstrating our approach. It is expected that the HV maximization formulation would work similarly if the parameters of some of the neural network layers are shared, which would decrease computational load.

In conclusion, we describe MO training of neural networks using HV maximization for learning-based a posteriori MO decision-making. Our approach provided the desired well-spread Pareto front approximations on artificial MO regression problems. On the MO style transfer problem, our method yielded encouraging results that emphasize its usefulness for a posteriori decision-making.

Acknowledgements. We thank dr. Marco Virgolin (Chalmers University of Technology) for his valuable contributions. The research is funded by Open Technology Programme (15586) of the Dutch Research Council (NWO), Elekta, and Xomnia, and the public-private partnership allowance for top consortia for knowledge and innovation from the Ministry of Economic Affairs.

References

1. Albuquerque, I., Monteiro, J., Doan, T., Considine, B., Falk, T., Mitliagkas, I.: Multi-objective training of generative adversarial networks with multiple discriminators. arXiv preprint arXiv:1901.08680 (2019)
2. Avent, B., Gonzalez, J., Diethe, T., Paleyes, A., Balle, B.: Automatic discovery of privacy-utility Pareto fronts. Proc. Priv. Enh. Technol. **2020**(4), 5–23 (2020)
3. Daulton, S., Balandat, M., Bakshy, E.: Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization. arXiv preprint arXiv:2006.05078 (2020)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
5. Deist, T.M., Grewal, M., Dankers, F.J., Alderliesten, T., Bosman, P.A.: Multi-objective learning to predict Pareto fronts using hypervolume maximization. arXiv preprint arXiv:2102.04523 (2021)
6. Deist, T.M., Maree, S.C., Alderliesten, T., Bosman, P.A.N.: Multi-objective optimization by uncrowded hypervolume gradient ascent. In: Bäck, T., et al. (eds.) PPSN 2020. LNCS, vol. 12270, pp. 186–200. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58115-2_13
7. Deng, J., Zhang, Q.: Approximating hypervolume and hypervolume contributions using polar coordinate. IEEE Trans. Evol. Comput. **23**(5), 913–918 (2019)
8. Emmerich, M., Deutz, A.: Time complexity and zeros of the hypervolume indicator gradient field. In: Schuetze, O., et al. (eds.) EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III. Studies in Computational Intelligence, vol. 500, pp. 169–193. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-319-01460-9_8
9. Fleischer, M.: The measure of Pareto optima applications to multi-objective metaheuristics. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Thiele, L., Deb, K. (eds.) EMO 2003. LNCS, vol. 2632, pp. 519–533. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36970-8_37
10. Fonseca, C.M., Paquete, L., López-Ibánez, M.: An improved dimension-sweep algorithm for the hypervolume indicator. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 1157–1163. IEEE (2006)
11. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2414–2423 (2016)
12. Golovin, D., et al.: Random hypervolume scalarizations for provable multi-objective black box optimization. arXiv preprint arXiv:2006.04655 (2020)
13. Gong, M., Liu, J., Li, H., Cai, Q., Su, L.: A multiobjective sparse feature learning model for deep neural networks. IEEE Trans. Neural Netw. Learn. Syst. **26**(12), 3263–3277 (2015)
14. Hwang, C.L., Masud, A.S.M.: Multiple Objective Decision Making—Methods and Applications: A State-of-the-Art Survey, vol. 164. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-45511-7
15. Iqbal, M.S., Su, J., Kotthoff, L., Jamshidi, P.: FlexiBO: cost-aware multi-objective optimization of deep neural networks. arXiv preprint arXiv:2001.06588 (2020)
16. Jacq, A.: Neural style transfer using Pytorch (2017). https://pytorch.org/tutorials/advanced/neural_style_tutorial.html
17. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

18. Koch, P., Wagner, T., Emmerich, M.T., Bäck, T., Konen, W.: Efficient multi-criteria optimization on noisy machine learning problems. Appl. Soft Comput. **29**, 357–370 (2015)
19. Lin, X., Yang, Z., Zhang, Q., Kwong, S.: Controllable Pareto multi-task learning. arXiv preprint arXiv:2010.06313 (2020)
20. Lin, X., Zhen, H.L., Li, Z., Zhang, Q.F., Kwong, S.: Pareto multi-task learning. Adv. Neural. Inf. Process. Syst. **32**, 12060–12070 (2019)
21. Lin, X., et al.: A Pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp. 20–28 (2019)
22. Ma, P., Du, T., Matusik, W.: Efficient continuous Pareto exploration in multi-task learning. In: International Conference on Machine Learning, pp. 6522–6531. PMLR (2020)
23. Mahapatra, D., Rajan, V.: Multi-task learning with user preferences: gradient descent with controlled ascent in Pareto optimization. In: International Conference on Machine Learning, pp. 6597–6607. PMLR (2020)
24. Mao, Y., Yun, S., Liu, W., Du, B.: Tchebycheff procedure for multi-task text classification. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 4217–4226 (2020)
25. Maree, S.C., et al.: Evaluation of bi-objective treatment planning for high-dose-rate prostate brachytherapy—a retrospective observer study. Brachytherapy **18**(3), 396–403 (2019)
26. Miranda, C.S., Von Zuben, F.J.: Single-solution hypervolume maximization and its use for improving generalization of neural networks. arXiv preprint arXiv:1602.01164 (2016)
27. Müller, B., Shih, H., Efstathiou, J., Bortfeld, T., Craft, D.: Multicriteria plan optimization in the hands of physicians: a pilot study in prostate cancer and brain tumors. Radiat. Oncol. **12**(1), 1–11 (2017)
28. Navon, A., Shamsian, A., Chechik, G., Fetaya, E.: Learning the Pareto front with hypernetworks. arXiv preprint arXiv:2010.04104 (2020)
29. Oyama, A., Liou, M.S.: Multiobjective optimization of rocket engine pumps using evolutionary algorithm. J. Propul. Power **18**(3), 528–535 (2002)
30. Paszke, A., et al.: Automatic differentiation in PyTorch. Adv. Neural Inf. Process. Syst. (2017). https://github.com/pytorch/pytorch
31. Ruchte, M., Grabocka, J.: Efficient multi-objective optimization for deep learning. arXiv preprint arXiv:2103.13392 (2021)
32. Stewart, T., et al.: Real-world applications of multiobjective optimization. In: Branke, J., Deb, K., Miettinen, K., Słowiński, R. (eds.) Multiobjective Optimization. LNCS, vol. 5252, pp. 285–327. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88908-3_11
33. Tari, S., Hoos, H., Jacques, J., Kessaci, M.-E., Jourdan, L.: Automatic configuration of a multi-objective local search for imbalanced classification. In: Bäck, T., et al. (eds.) PPSN 2020. LNCS, vol. 12269, pp. 65–77. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58112-1_5
34. Van Moffaert, K., Nowé, A.: Multi-objective reinforcement learning using sets of Pareto dominating policies. J. Mach. Learn. Res. **15**(1), 3483–3512 (2014)
35. Van Veldhuizen, D.A., Lamont, G.B.: Multiobjective evolutionary algorithms: analyzing the state-of-the-art. Evol. Comput. **8**(2), 125–147 (2000)
36. Wang, H., Deutz, A., Bäck, T., Emmerich, M.: Code repository: Hypervolume indicator gradient ascent multi-objective optimization. https://github.com/wangronin/HIGA-MO

37. Wang, H., Deutz, A., Bäck, T., Emmerich, M.: Hypervolume indicator gradient ascent multi-objective optimization. In: Trautmann, H., et al. (eds.) EMO 2017. LNCS, vol. 10173, pp. 654–669. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54157-0_44
38. Xu, J., Tian, Y., Ma, P., Rus, D., Sueda, S., Matusik, W.: Prediction-guided multi-objective reinforcement learning for continuous robot control. In: International Conference on Machine Learning, pp. 10607–10616. PMLR (2020)
39. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)