Holistic deep learning

Machine Learning
https://doi.org/10.1007/s10994-023-06482-y

Holistic deep learning

Dimitris Bertsimas[1] · Kimberly Villalobos Carballo[2] · Léonard Boussioux[2] · Michael Lingzhi Li[3] · Alex Paskov[2] · Ivan Paskov[2]

Received: 15 March 2023 / Revised: 7 August 2023 / Accepted: 24 October 2023

Abstract
This paper presents a novel holistic deep learning framework that simultaneously addresses the challenges of vulnerability to input perturbations, overparametrization, and performance instability from different train-validation splits. The proposed framework holistically improves accuracy, robustness, sparsity, and stability over standard deep learning models, as demonstrated by extensive experiments on both tabular and image data sets. The results are further validated by ablation experiments and SHAP value analysis, which reveal the interactions and trade-offs between the different evaluation metrics. To support practitioners applying our framework, we provide a prescriptive approach that offers recommendations for selecting an appropriate training loss function based on their specific objectives. All the code to reproduce the results can be found at https://github.com/kimvc7/HDL.

Keywords Deep learning · Optimization · Robustness · Sparsity · Stability · Regularization

Editor: Nathalie Japkowicz.

✉ Dimitris Bertsimas
dbertsim@mit.edu

Kimberly Villalobos Carballo
kimvc@mit.edu

Léonard Boussioux
leobix@mit.edu

Michael Lingzhi Li
mlli@mit.edu

Alex Paskov
apaskov@mit.edu

Ivan Paskov
ipaskov@mit.edu

[1] Sloan School of Management and Operations Research Center, Massachusetts Institute of Technology, Cambridge 02139, MA, USA

[2] Operations Research Center, Massachusetts Institute of Technology, Cambridge 02139, MA, USA

[3] Technology and Operations Management, Harvard Business School, Boston 02163, MA, USA

Published online: 07 December 2023

1 Introduction

Neural networks have become increasingly popular due to their remarkable achievements in computer vision and natural language processing. Their generalization power has been demonstrated in wide-ranging applications, from classifying photos to recommending products. However, neural networks face challenges in real-world applications for high-stakes decision-making, including healthcare, policy-making, and autonomous driving.

First, many standard neural networks are not robust – they can be easily fooled by natural or artificial noise in the input data (Szegedy et al., 2014), making them vulnerable to perturbations that may arise in real-world applications. Moreover, neural networks, similar to other machine learning models, often suffer from instability during the training process – different train-validation splits could generate models with very different performance (May et al., 2010; Xu & Goodacre, 2018). This reduces the policymakers' trust in these models and hinders post-hoc interpretations. Another critical difficulty is that neural networks are not sparse – the high number of parameters utilized for neural networks prevents efficient computation and storage (Thompson et al., 2020). Most neural networks have millions of non-zero parameters to be stored and accessed for evaluation. This is problematic in many decision-making settings with limitations or restrictions on hardware capabilities. Reducing the number of parameters could make them more applicable in a broader range of scenarios (Changpinyo et al., 2017; Narang et al., 2017).

The questions around improving robustness, stability, and sparsity metrics have all been previously studied in the neural network literature. However, they have been almost exclusively studied in isolation, with a limited understanding of the tradeoffs between these desired qualities and their effect on natural accuracy (accuracy with respect to the unperturbed data samples). This paper aims to simultaneously address all these objectives through a novel comprehensive methodology named Holistic Deep Learning (HDL). In particular, HDL carefully combines state-of-the-art techniques that address these individual challenges and demonstrates their collective efficacy through extensive experiments on diverse data sets. Our findings provide a promising pathway toward developing efficient and reliable machine learning models across many dimensions for real-world applications.

Specifically, our contributions are as follows:

1. We design HDL, a novel framework that jointly optimizes for neural network robustness (adversarial accuracy), stability (worst accuracy across train-validation splits), and sparsity (parameters with value zero) metrics by appropriately modifying the objective function.
2. Through extensive ablation experiments and SHAP value analysis (Lundberg & Lee, 2017) across 45 UCI data sets (Dua & Graff, 2017) and 3 image data sets (MNIST (Deng, 2012), Fashion MNIST (Xiao et al., 2017) and CIFAR10 (Krizhevsky et al., 2009)), we analyze the individual performance of each metric as well as the interactions and trade-offs between them. We corroborate that imposing robustness, stability, and sparsity improves the corresponding metrics across all data sets. In addition, we show that:
    - Imposing stability and sparsity further improves robustness,
    - Imposing stability and robustness further improves sparsity,
    - Imposing robustness further improves stability,

- Imposing stability and robustness further improves natural accuracy.

  The effect of sparsity on natural accuracy is more complex and highly varies across data sets. However, we show that it is often possible to simultaneously improve robustness, stability, and sparsity without sacrificing performance on natural accuracy.

3. We propose a prescriptive approach to provide recommendations on selecting the appropriate loss function depending on the practitioner's objective. In particular, simultaneously imposing robustness, stability and sparsity in the loss function leads to the best results when jointly optimizing for all the metrics.

The paper is organized as follows: Sect. 2 outlines the current literature of robust, sparse, and stable methods; Sect. 3 describes the Holistic Deep Learning framework, and Sect. 4 shows the results of the computational experiments.

2 Related work

2.1 Robust neural networks

Many state-of-the-art deep neural networks are highly vulnerable to small perturbations in the input data (Szegedy et al., 2014), which can be a threat to some real-world applications like self-driving cars or face recognition. Adversarial robustness evaluates a neural network's resistance against these altered inputs, referred to as adversarial examples, intentionally designed to worsen the network's performance (Goodfellow et al., 2014; Carlini & Wagner, 2017; Madry et al., 2017).

Multiple methods have been developed in recent years to enhance the adversarial robustness of neural networks. One of the most popular heuristics is augmenting the data set during training with adversarial examples (Madry et al., 2017; Goodfellow et al., 2014). Others include neuron randomization (Prakash et al., 2018; Xie et al., 2017), input space projections (Lamb et al., 2018; Kabilan et al., 2021; Ilyas et al., 2017) and regularization (Bertsimas et al., 2021; Ross & Doshi-Velez, 2018; Hein & Andriushchenko, 2017; Yan et al., 2018). A less common but more theoretically rigorous approach is to minimize a provable upper bound of the loss achieved with adversarial examples (Raghunathan et al., 2018; Singh et al., 2018; Zhang et al., 2018; Weng et al., 2018; Dvijotham et al., 2018; Lecuyer et al., 2019; Cohen et al., 2019; Anderson et al., 2020; Bertsimas et al., 2021).

While these methods successfully improve the network's robustness, the extent to which they do so often depends on the data set, the network size, and the magnitude of the input perturbations. In particular, heuristic methods generally work well for small perturbations, while the upper bound methods yield better results when the input noise is larger (Bertsimas et al., 2021; Athalye et al., 2018). However, there is a trade-off between effectiveness and efficiency. The methods providing the strongest adversarial robustness are often computationally demanding, making it challenging to implement them for large data sets or complex network architectures.

2.2 Sparse neural networks

In machine learning, sparse models make predictions based on a limited number of parameters. Sparsity is often desirable, as it may save memory, enhance model interpretability, and reduce overfitting (Bertsimas et al., 2020).

There are two typical approaches to sparsity in deep learning. The first one, *train-then-sparsify*, consists of removing unnecessary neurons or connections after training the network, sometimes followed by retraining (Janowsky, 1989; LeCun et al., 1989). This approach has been widely investigated, and several schemes exist to choose which connections to prune (Hoefler et al., 2021). Han et al. (2015), for example, propose to prune the connections with the smallest weights. Other methods include formulating a convex optimization problem (Aghasi et al., 2020), removing filters for which the total absolute sum is low (Li et al., 2016), and eliminating channels that have limited impact on the network's discriminatory ability (Zhuang et al., 2018). The second approach, *sparsify-during-training*, is achieved by learning a sparse architecture while training the network. Multiple methodologies exist (Bellec et al., 2017; Mocanu et al., 2017; Mostafa & Wang, 2019), including the method to approximate the ℓ_0 norm with continuous functions and add a regularization term to the loss function (Louizos et al., 2017; Savarese et al., 2020). We refer the reader to Gale et al. (2019) and Hoefler et al. (2021) for more comprehensive surveys on sparsity.

2.3 Stable neural networks

The stochastic nature of data samples can lead to instability and high dependence of machine learning models on the specific train-validation split. This can negatively impact the interpretability of the resulting model and its ability to make reliable predictions (Bertsimas & Paskov, 2020), a key factor to establishing trust in any algorithm.

The sensitivity of machine learning models to the choice of training split has mostly been studied through the lens of cross-validation and distributionally robust optimization. Cross-validation can be used to measure the variability from the selection of training split but at a significant increase in computational cost (Krogh & Vedelsby, 1994; Hastie et al., 2001) that is often intractable for deep learning settings. Distributionally robust optimization has been used to quantify the worst-case generalization error in the presence of shifts in distribution or regime (Staib & Jegelka, 2019; Goldwasser et al., 2020; Sagawa et al., 2019), but it often requires pre-defined groups over the training data and expensive group annotations for each data sample to avoid overly pessimistic uncertain distributions (Sagawa et al., 2019; Liu et al., 2021). A different approach has been studied by Bertsimas and Paskov (2020) and Bertsimas et al. (2022), who instead optimize over the worst training set of fixed size without making any probabilistic assumptions. Although their method was presented in the context of linear and tree-based models, their framework also applies to neural networks.

3 The holistic deep learning approach

3.1 The HDL framework

We introduce the HDL framework for a classification problem over points $\mathbf{x} \in \mathbb{R}^M$ whose target $y \in [K]$ is one of K different classes (we use the notation [n] to denote the set $\{1, \ldots, n\}$). We illustrate our approach over a fully-connected neural network for simplicity of notation, but the framework remains the same for convolutional neural networks.

For $\mathbf{x} \in \mathbb{R}^M$, define $[\mathbf{x}]_m^+ = \max\{0, x_m\}$ (the ReLU function). Given weight matrices $\mathbf{W}^\ell \in \mathbb{R}^{r_{\ell-1} \times r_\ell}$ and bias vectors $\mathbf{b}^\ell \in \mathbb{R}^{r_\ell}$ for $\ell \in [L]$, such that $r_0 = M, r_L = K$, the corresponding feed-forward neural network with L layers and ReLU activation functions is defined by the equations:

$$\mathbf{z}^1(\mathcal{W}, \mathbf{x}) = \mathbf{W}^1 \mathbf{x} + \mathbf{b}^1, \tag{1}$$

$$\mathbf{z}^\ell(\mathcal{W}, \mathbf{x}) = \mathbf{W}^\ell [\mathbf{z}^{\ell-1}(\mathcal{W}, \mathbf{x})]^+ + \mathbf{b}^\ell, \quad \forall\, 2 \le \ell \le L, \tag{2}$$

where $\mathcal{W}$ denotes the parameters $(\mathbf{W}^\ell, \mathbf{b}^\ell)$ for all $\ell \in [L]$. Consider a data set $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where $y_n \in [K]$ is the target class of $\mathbf{x}_n$. For each point $\mathbf{x}_n$, the class predicted by the network is $\arg\max_k z_k^L(\mathcal{W}, \mathbf{x}_n)$.

The nominal DL approach is to minimize the cross-entropy loss of the network $\mathbf{z}^L$ described in Eq. (2), which can be written as:

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W}, \mathbf{x})} \right), \tag{3}$$

where $\Delta \boldsymbol{e}_k^{y_n} = \boldsymbol{e}_k - \boldsymbol{e}_{y_n}$ and $\boldsymbol{e}_k$ refers to the one-hot vectors with a 1 in the k^{th} coordinate and 0 everywhere else. In our HDL framework, we propose instead to minimize the following optimization problem:

$$\min_{s,\theta,\mathcal{W}} \underbrace{\lambda \sum_{j=1}^{|\mathcal{W}|} \sigma(\beta s_j)}_{\text{Sparsity}} + \underbrace{\theta}_{\text{Stability}}$$
$$+ \frac{1}{a} \sum_{n=1}^{N} \left[\log \sum_{k=1}^{K} e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W} \odot \sigma(\beta \boldsymbol{s}), \mathbf{x}) + \overbrace{\rho \left\| \nabla_{\mathbf{x}} \left(\Delta \boldsymbol{e}_k^{y_n} \right)^\top \mathbf{z}^L(\mathcal{W} \odot \sigma(\beta \boldsymbol{s}), \mathbf{x}) \right\|_1}^{\text{Robustness}}} - \underbrace{\theta}_{\text{Stability}} \right]^+, \tag{4}$$

where $\odot$ corresponds to the element-wise product, σ is the standard sigmoid function, $\mathbf{z}^L(\cdot, \mathbf{x})$ was defined in (2), λ (resp. ρ) is the regularization coefficient corresponding to the sparsity (resp. robustness) loss component, and a is the size of the data subsets used for the stability requirement (see Sect. 2.3). We observe that robustness adds a term to the output, while stability and sparsity add new parameters (θ and $\boldsymbol{s}$ respectively) to be optimized. This

loss function allows us to simultaneously train robust, sparse, and stable feed-forward neural networks at scale. In the next section, we provide more details about each metric.

3.2 Robustness

This section describes our method to introduce the robust component into neural network training. Since our ultimate goal is to incorporate the sparsity, robustness, and stability of neural networks together in a tractable way, we avoid algorithms that improve robustness at the expense of a significant increase in the training time or the algorithm's complexity (for instance, the algorithms that perform training with adversarial examples usually require significantly longer times to optimally find such examples at each gradient descent iteration (Madry et al., 2017; Bertsimas et al., 2021)). We follow the approach from Bertsimas et al. (2021) of using a linear approximation of the neural network to estimate the robust objective. This approach is simple to implement, produces good adversarial accuracy, and does not require the extensive training time of other algorithms.

For a given $(\mathbf{x}, y)$ pair, the robust problem using the cross-entropy loss and the ℓ_∞-norm uncertainty sets can be upper bounded as:

$$\max_{\delta:\|\delta\|_\infty \le \rho} \log \sum_k e^{(\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}+\boldsymbol{\delta})} \le \log \sum_k e^{\max_{\delta:\|\delta\|_\infty \le \rho} (\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}+\boldsymbol{\delta})}. \tag{5}$$

Since $z_k^L(\mathcal{W}, \mathbf{x})$ is piece-wise linear, we expect the outputs $z_k^L(\mathcal{W}, \mathbf{x})$ and $z_k^L(\mathcal{W}, \mathbf{x}+\boldsymbol{\delta})$ to be in the same linear piece when $\mathbf{x}+\boldsymbol{\delta}$ is close to $\mathbf{x}$. In other words, the linear approximation

$$z_k^L(\mathcal{W}, \mathbf{x}+\boldsymbol{\delta}) \approx z_k^L(\mathcal{W}, \mathbf{x}) + \boldsymbol{\delta}^\top \nabla_{\mathbf{x}} z_k^L(\mathcal{W}, \mathbf{x}) \tag{6}$$

is exact for small enough $\boldsymbol{\delta}$. Therefore, we approximate the upper bound in (5) as

$$\begin{aligned} &\log \sum_k e^{\max_{\delta:\|\delta\|_\infty \le \rho} (\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}+\boldsymbol{\delta})} \\ &\approx \log \sum_k e^{\max_{\delta:\|\delta\|_\infty \le \rho} (\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}) + \boldsymbol{\delta}^\top \nabla_{\mathbf{x}} (\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})} \\ &= \log \sum_k e^{(\Delta \boldsymbol{e}_k^y)^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}) + \rho \|(\Delta \boldsymbol{e}_k^y)^\top \nabla_{\mathbf{x}} \mathbf{z}^L(\mathcal{W},\mathbf{x})\|_1}. \end{aligned} \tag{7}$$

Even though the expression in Eq. (7) is not always an upper bound of Eq. (5) for an arbitrary value of ρ, Bertsimas et al. (2021) experimentally show that generally the average loss obtained using this expression is indeed an upper bound of the average adversarial loss. In fact, for small ρ, their experiments demonstrate that this approach achieves competitive results with state-of-the-art methods while requiring significantly less computational time across various tabular and image data sets. However, we emphasize that the methodology developed in this paper could also be performed with other methods for robust training, like adversarial training or upper bound minimization, which might be more appropriate for large uncertainty sets.

3.3 Sparsity

In this work, we use the specific retraining procedure proposed by Savarese et al. (2020), which deterministically approximates the ℓ_0 regularization utilizing a sequence of sigmoid

functions and adding them as a penalty term in the loss function. Notably, the implementation is easily compatible with our robustness and stability requirements, since this methodology relies on a penalty term added in the loss function. Therefore, we can use gradient descent to simultaneously optimize the objective function comprising the robustness, stability, and sparsity penalties.

Adding ℓ_0 regularization explicitly penalizes the number of non-zero weights in the model to induce sparsity. However, the ℓ_0-norm induces a priori a non-convex and non-differentiable loss function $\mathcal{R}(\mathcal{W})$, as follows:

$$\mathcal{R}(\mathcal{W}) = \frac{1}{N}\left(\sum_{n=1}^{N} \mathcal{L}\big(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)\big)\right) + \lambda \|\mathcal{W}\|_0, \quad \|\mathcal{W}\|_0 = \sum_{j=1}^{|\mathcal{W}|} \mathbb{I}\big[w_j \neq 0\big], \tag{8}$$

where $|\mathcal{W}|$ is the number of parameters, w_j is the j^{th} coordinate of $\mathcal{W}$, λ is the regularization weight and $\mathcal{L}$ a loss function (e.g., cross-entropy loss).

The goal is to relax the discrete nature of the ℓ_0 penalty to preserve an efficient continuous optimization while allowing for exact zeros in the neural network weights. To do this, Savarese et al. (2020) propose to first parameterize the weights $w_j = H(s_j)$ where $H(\cdot)$ is the Heaviside step function, and then approximate the non-differentiable step function with the sigmoid function: $\sigma(\beta s_j) \to H(s_j)$ when $\beta \to \infty$. Therefore, β is the hardness parameter that controls how close the approximation is to the ℓ_0 regularization, and the final loss function can be written as:

$$\mathcal{R}(\mathcal{W}) \approx \frac{1}{N}\left(\sum_{n=1}^{N} \mathcal{L}\big(y_n, \mathbf{z}^L(\mathcal{W} \odot \sigma(\beta \boldsymbol{s}), \mathbf{x}_n)\big)\right) + \lambda \sum_{j=1}^{|\mathcal{W}|} \sigma(\beta s_j). \tag{9}$$

To achieve a sparse network, we use this loss function (9) over multiple training rounds to gradually reach a sparse initialization before training the final sparse neural network. To obtain each initialization before a new training round, we start with our initialized auxiliary sparsity $\boldsymbol{s}_0$ and hardness $\beta = 1$ parameters. Over the T training iterations, we gradually increase β until it reaches a maximum value $\bar{\beta}$ when the training procedure is completed with sparsity $\boldsymbol{s}_T$. Then, we take $\boldsymbol{s}_0' = \min(\bar{\beta}\boldsymbol{s}_T, \boldsymbol{s}_0)$ to generate the new initialization for the next round of training. This minimization function essentially keeps the information of the suppressed weights, i.e., $\sigma(\beta s_j) \approx 0$, while reverting those not suppressed to their starting position. This process is completed over multiple rounds to find better and sparser initializations for the neural network.

We implement the methodology as suggested by Savarese et al. (2020). In the results section, we measure sparsity in terms of the percentage of neuron connections (weights) set to 0.

3.4 Stability

Using the measure of stability defined in Sect. 2.3, we apply the methodology developed in Bertsimas et al. (2022) for building stable neural networks. At a high level, this corresponds to constructing a model that is robust to the specific subset of data used to train it. One way to think about this is to view the training data set as a sample from the true data distribution and then require the model to be robust to the specific sample. Considering the partition of the data into train-validation sets as a sampling mechanism from this true data distribution

(each split choice gives a different training set), we desire to build models that are robust to every partition.

To achieve this, we first associate each observation $(\mathbf{x}_n, y_n)$ with a binary variable z_n, $n \in [N]$ that indicates whether or not $(\mathbf{x}_n, y_n)$ is part of the training set. We then choose the network's parameters as to minimize the worst-case loss over all possible allocations of these z_n's, resulting in a model that is explicitly built to do well not just over one training set, but over all possible training sets. We start from the same minimization problem introduced in Sect. 3.1, i.e.,

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)).$$

To obtain network stability we require the model to be robust to every training set of fixed size a, which results in the following optimization problem:

$$\begin{aligned} &\min_{\mathcal{W}} \max_{z \in \mathcal{Z}} \frac{1}{a} \sum_{n=1}^{N} z_n \mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)), \\ &\text{where } \mathcal{Z} = \left\{ z : \sum_{n=1}^{N} z_n = a, \quad z_n \in \{0,1\},\ n \in [N] \right\}. \end{aligned} \tag{10}$$

The value of a indicates the desired proportion between the size of the training and validation sets. For example, by setting $a = 0.7N$ we recover the typical 70/30 train-validation split. Since the inner maximization problem is linear in z, the problem is equivalent to optimizing over the convex hull of $\mathcal{Z}$. This implies that the binary constraints on z_n can be relaxed to $0 \le z_n \le 1$, and we obtain a linear maximization problem in the variables z_n. Computing its dual problem we obtain that the value of the inner maximization problem is equivalent to:

$$\min_{\theta, u_n} \ \theta + \frac{1}{a} \sum_{n=1}^{N} u_n \quad \text{subject to} \quad \theta + u_n \ge \mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)), \quad u_n \ge 0,\ n \in [N].$$

Therefore, the stability problem becomes

$$\min_{\mathcal{W}, \theta, u_n} \ \theta + \frac{1}{a} \sum_{n=1}^{N} u_n \quad \text{subject to} \quad \theta + u_n \ge \mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)),\ u_n \ge 0,\ n \in [N].$$

Note that the variables u_n can be solved in closed form as $u_n = [\mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)) - \theta]^+$. The final minimization problem with stability then becomes:

$$\min_{\mathcal{W}, \theta} \ \theta + \frac{1}{a} \sum_{n=1}^{N} \left[\mathcal{L}(y_n, \mathbf{z}^L(\mathcal{W}, \mathbf{x}_n)) - \theta \right]^+,$$

which is now an unconstrained problem that can be solved with standard gradient descent optimization algorithms.

4 Experiments

This section presents extensive computational experiments comparing the nominal DL approach (abbreviated DL) with 7 other models resulting from our holistic methodology. We showcase the merit of our HDL framework and investigate the influence of each studied component – robustness, sparsity, and stability – on the overall performance across 4 evaluation metrics:

- *Natural accuracy*: Average accuracy on the testing set across the 10 different train-validation splits with respect to the original input data.
- *Adversarial robustness*: Average adversarial accuracy on the testing set across the 10 different train-validation splits with respect to adversarial attacks resulting from perturbations of the original input data. We consider only attacks bounded in the L_∞ norm by some radius ρ using Projected Gradient Descent as in Madry et al. (2017).
- *Stability*: Worst accuracy on the testing set across the 10 different train-validation splits with respect to the original input data.
- *Sparsity*: Percentage of network parameters with value 0.

The exact optimization problem solved for each model results from combinations of the loss functions described in the previous section, and the specific formulations can be found in Table 1.

Data

We computed experiments on classification tasks with 45 UCI data sets from the UCI Machine Learning Repository (Dua & Graff, 2017). These data sets give various problem sizes and difficulties to form a representative sample of real-world tabular problems, with the largest data set having 245,056 observations and the highest number of features being

Table 1 Loss functions used for DL and all methods in the HDL framework

Method	Optimization Problem
DL	$\min_{\mathcal{W}} \frac{1}{N}\sum_{n=1}^{N} \log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}_n)}\right)$
Robust	$\min_{\mathcal{W}} \frac{1}{N}\sum_{n=1}^{N} \log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})+\rho\|\nabla_{\mathbf{x}}(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})\|_1}\right)$
Stable	$\min_{\mathcal{W},\theta} \theta + \frac{1}{a}\sum_{n=1}^{N}\left[\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x}_n)}\right) - \theta\right]^+$
Sparse	$\min_{\mathcal{W},s} \lambda\sum_{j=1}^{\lvert\mathcal{W}\rvert}\sigma(\beta s_j) + \frac{1}{N}\sum_{n=1}^{N}\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W}\odot\sigma(\beta s),\mathbf{x}_n)}\right)$
Robust + Sparse	$\min_{\mathcal{W},s} \lambda\sum_{j=1}^{\lvert\mathcal{W}\rvert}\sigma(\beta s_j)$ $+\frac{1}{N}\sum_{n=1}^{N}\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W}\odot\sigma(\beta s),\mathbf{x})+\rho\|\nabla_{\mathbf{x}}(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})\|_1}\right)$
Stable + Sparse	$\min_{\mathcal{W},\theta,s} \lambda\sum_{j=1}^{\lvert\mathcal{W}\rvert}\sigma(\beta s_j) + \theta + \frac{1}{a}\sum_{n=1}^{N}\left[\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W}\odot\sigma(\beta s),\mathbf{x}_n)}\right) - \theta\right]^+$
Stable + Robust	$\min_{\mathcal{W},\theta} \theta + \frac{1}{a}\sum_{n=1}^{N}\left[\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})+\rho\|\nabla_{\mathbf{x}}(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})\|_1}\right) - \theta\right]^+$
HDL	$\min_{\mathcal{W},\theta,s} \lambda\sum_{j=1}^{\lvert\mathcal{W}\rvert}\sigma(\beta s_j) + \theta$ $+\frac{1}{a}\sum_{n=1}^{N}\left[\log\left(\sum_k e^{(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W}\odot\sigma(\beta s),\mathbf{x})+\rho\|\nabla_{\mathbf{x}}(\Delta \boldsymbol{e}_k^{y_n})^\top \mathbf{z}^L(\mathcal{W},\mathbf{x})\|_1}\right) - \theta\right]^+$

856. We also benchmarked our methodologies on three image data sets: MNIST, Fashion-MNIST, and CIFAR10.

Implementation

Our code is written in Python 3.8 (Van Rossum & Drake, 2009). Neural networks are coded using Tensorflow v1 (Abadi et al., 2015). We trained each model on a system equipped with an Intel Xeon Gold 6248 processor, which included 4 CPU cores and one Nvidia Volta V100 GPU.

Training methodology

For each data set, we used 20% of the data to obtain a fixed test set, and we randomly generated 10 different 80%-20% train-validation splits with the remaining data points. The same 10 train-validation partitions were used across all methods for a fair comparison. Given a choice of model and evaluation metric, we selected the hyperparameters that led to the best average performance in the validation set for the metric in question. We then reported the average performance of the chosen parameter configuration on the test set with respect to the given metric. For all evaluation metrics, the average performance is computed over the 10 train-validation splits initially generated.

Neural network architectures

For our experiments on UCI data sets, we used a feedforward neural network architecture with 2 hidden layers, each with 128 neurons and ReLU activations. For our experiments on the image data sets, we used a convolutional neural network with the AlexNet architecture (Krizhevsky et al., 2012). We used the Glorot uniform initialization (Glorot & Bengio, 2010) for the network weights $\mathcal{W}$ and $\mathbf{0}$ as initialization for the sparsity variable s_0. The choice of architecture and initialization was made to reflect typical settings utilized in the machine learning community (e.g. Madry et al. (2017); Savarese et al. (2020); Bertsimas et al. (2021)) while maintaining moderate size networks that facilitate exhaustive experimentation across dozens of data sets. Importantly, the same architecture is used across all methods been evaluated.

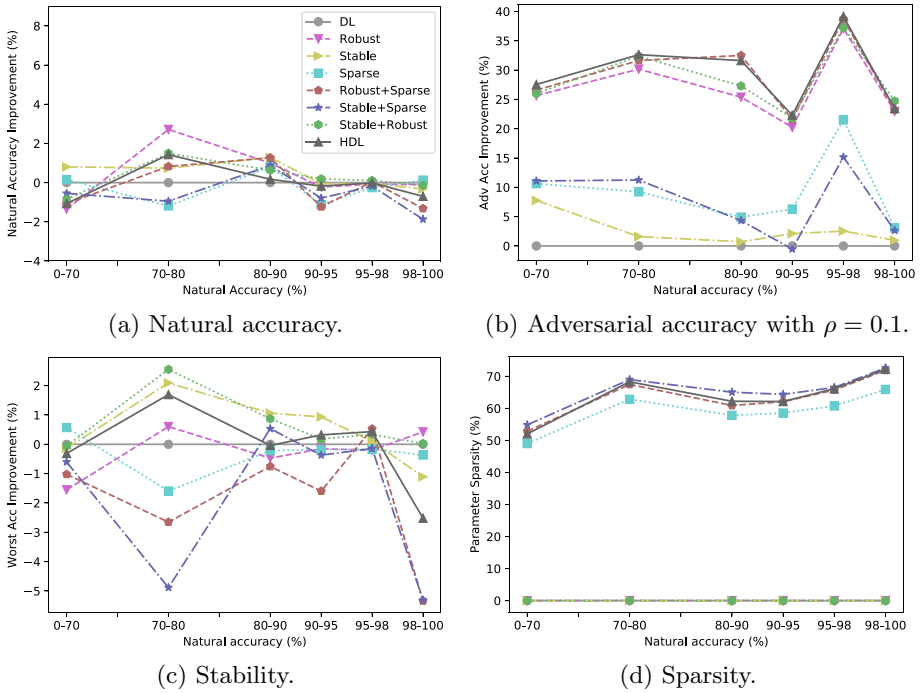
Hyperparameter search

For each model, we cross-validated the values of the following hyperparameters:

- Adam learning rate: $\{1e^{-2}, 1e^{-3}\}$ for UCI data sets, $\{1e^{-3}, 1e^{-4}\}$ for image data sets.
- Number of epochs: 150 for UCI data sets, 50 for vision data sets.
- Batch Size: 32 for UCI data sets, 64 for image data sets.
- Robustness radius ρ: $\{1e^{-1},\ 1e^{-2},\ 1e^{-3},\ 1e^{-4},\ 1e^{-5}\}$.
- Sparsity regularization parameter λ: $\{1e^{-6},\ 1e^{-8},\ 1e^{-10}\}$.
- Sparsity temperature parameter $\bar{\beta}$: $\{200, 1000\}$.
- Stability parameter a: $\{0.7, 0.8, 0.9, 1\}$.

4.1 UCI data sets

We split the 45 UCI data sets into 6 roughly even-sized groups based on their difficulty level. Specifically, we consider the ranges 0%-70%, 70%-80%, 80%-90%, 90%-95%, 95%-98% and 98%-100% of natural accuracy achieved by the nominal DL approach. We first



(a) Natural accuracy.

(b) Adversarial accuracy with $\rho = 0.1$.

(c) Stability.

(d) Sparsity.

Fig. 1 Evaluation of the different methods depending on the natural accuracy of the nominal DL approach on the UCI data sets

investigate the performance of the HDL framework with respect to a single evaluation metric. In Fig. 1, we evaluate all methods in terms of natural accuracy, adversarial accuracy with $\rho = 0.1$, stability, and sparsity.

Figure 1a and c show that those data sets for which the nominal approach achieves accuracy in the 70%-90% range are the ones that benefit the most from the HDL framework (especially the Robust, Stable, and Stable+Robust models) when the evaluation metric corresponds to natural accuracy or stability. For the data sets with natural accuracy above 90%, none of the models significantly improve over the natural accuracy or stability achieved by the nominal DL model. However, for data sets in the 98%-100% range sparsity slightly improves accuracy and robustness slightly helps for stability.

Figure 1b shows the adversarial robustness achieved with perturbation parameter $\rho = 0.1$. We see a substantial adversarial robustness improvement in all methods that included the robust component. Moreover, combining robustness with stability and/or sparsity leads to higher adversarial accuracy than that achieved with robustness alone. In terms of parameter sparsity, Fig. 1d shows that all models with imposed sparsity (Sparse, Stable+Sparse, Robust+Sparse, and HDL) have a much lower percentage of nonzero parameters

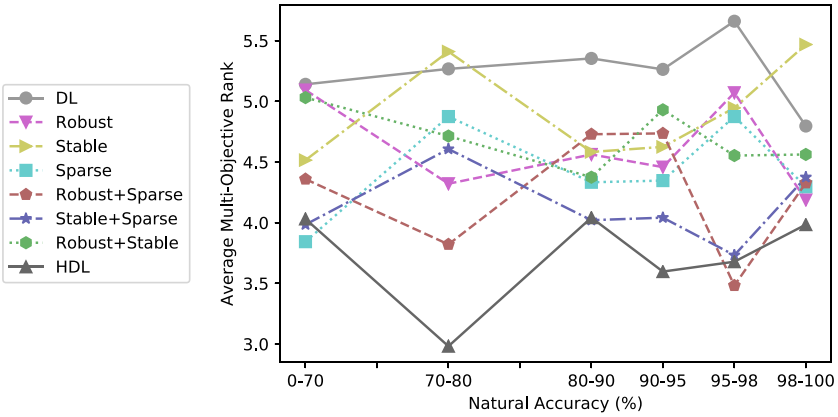


Fig. 2 Average multi-objective rank

Table 2 Results for the Fashion-MNIST data set. For each method, the parameters with the highest average rank in the validation set were chosen

Method	DL	Rob.	Stab.	Sparse	Rob. + Sparse	Stab. + Sparse	Stab. + Rob.	HDL
Avg. accuracy	91.8	92.0	91.9	91.4	**92.1**	91.4	92.0	**92.1**
Adv. acc. $\rho = 0.01$	78.7	81.1	78.3	80.8	86.9	80.2	86.8	**87.1**
Stability	91.5	91.7	**91.8**	91.3	91.9	91.2	91.7	**91.8**
Sparsity	0	0	0	36.2	26.6	**48.4**	0	26.8

Bold numbers correspond to the highest value in each row

Table 3 Results for the MNIST data set. For each method, the parameters with the highest average rank in the validation set were chosen

Method	DL	Rob.	Stab.	Sparse	Rob. + Sparse	Stab. + Sparse	Stab. + Rob.	HDL
Avg. Accuracy	99.2	**99.3**	99.2	99.1	99.2	99.2	**99.3**	**99.3**
Adv. Acc. $\rho = 0.1$	49.6	78.4	51.5	42.6	74.7	27.7	**79.5**	76.0
Stability	99.1	99.2	99.2	99.1	99.1	99.0	**99.3**	99.2
Sparsity	0	0	0	16.1	27.9	**31.7**	0	28.0

Bold numbers correspond to the highest value in each row

compared to the models without it. And importantly, both robustness and stability help achieve sparser models when combined with sparsity.

Since we are also interested in models that are simultaneously accurate, sparse, robust, and stable, we consider a multi-objective metric using the rank of each method (ranks start at 1, with lower ranks corresponding to better performance). For each method, we use the natural accuracy, adversarial accuracy, stability, and sparsity achieved in the validation set respectively to rank all its hyperparameter configurations 4 times. Then for each hyperparameter configuration, we compute the average rank across the 4 metrics and select

Table 4 Results for the CIFAR10 data set. For each method, the parameters with the highest average rank in the validation set were chosen

Method	DL	Rob.	Stab.	Sparse	Rob. + Sparse	Stab. + Sparse	Stab. + Rob.	HDL
Avg. Accuracy	**70.1**	**70.1**	**70.1**	69.8	69.2	69.3	69.8	69.3
Adv. Acc. $\rho = 0.01$	26.7	27.1	26.6	27.3	27.4	27.7	29.1	**30.6**
Stability	69.7	69.7	**69.8**	69.3	68.9	68.5	69.4	68.8
Sparsity	0	0	0	28.7	47.2	**47.8**	0	**47.8**

Bold numbers correspond to the highest value in each row

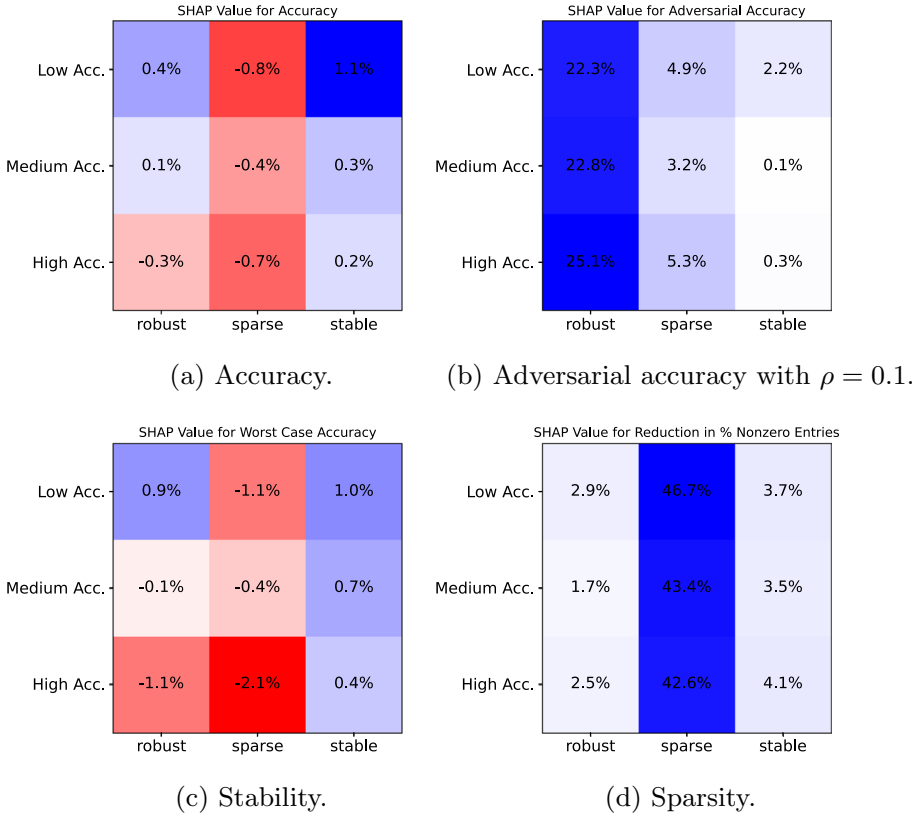
the configuration that leads to the method's highest average rank. Finally, we rank the 8 selected models (for the 8 different methods) with respect to each evaluation metric on the testing set to obtain their out-of-sample average rank. As shown in Fig. 2, all 7 models from the HDL framework outperform the nominal DL approach with respect to this holistic metric. Moreover, the HDL model typically achieves the best results across data set complexities.

4.2 Image data sets

In this section, we evaluate all methods using the MNIST, Fashion-MNIST, and CIFAR10 data sets. For each method, we select the parameters based on the multi-objective metric utilized for the UCI data sets in the validation set and report the performance across metrics. In Tables 2 and 3, we see that for MNIST and Fashion-MNIST, the HDL model outperforms the DL model for all objectives. In particular, HDL achieves higher accuracy using only around 70% of the parameters. The results for the CIFAR10 data set (Table 4) are a bit different since adding sparsity slightly hurts natural accuracy. However, the accuracy achieved by the HDL model is comparable to those achieved by the non-sparse models and the number of parameters is reduced by 47%.

Table 5 Average slowdown factors of computational time with respect to the nominal DL method

Method	Batches/sec Slowdown factor	No. iterations Increase factor	Total slowdown factor
Robust	2.7	1	2.7
Stable	1.0	1	1.0
Sparse	1.2	5	5.9
Robust+Sparse	3.2	5	16.1
Stable+Sparse	1.1	5	5.5
Stable+Robust	2.7	1	2.7
HDL	3.2	5	16.2



(a) Accuracy. (b) Adversarial accuracy with $\rho = 0.1$.

(c) Stability. (d) Sparsity.

Fig. 3 SHAP values on various metrics across different UCI data set categories. Blue/red indicates that the feature has a positive/negative SHAP value on a specific category of UCI data set

4.3 Computational times

Since modifying the loss function often affects the training computational time, we quantify the slowdown effect for all the methods in the HDL framework. Specifically, for each of the 45 UCI data sets as well as the 3 image data sets introduced in the previous section, we calculate how many times slower each method is when compared to the nominal DL approach in terms of batches per second as well as number of iterations needed. The average slowdown factors are shown in Table 5. We observe that robustness and sparsity both decrease the number of batches per second by approximately a factor of 3, while stability preserves the same speed as the DL approach. In addition, since we used 5 training rounds for the methods incorporating sparsity, they require 5 times as many training iterations as the other methods. On average, the HDL method is only 16 times slower, and methods that don't optimize for sparsity only increase the computational time by less than 3 times.

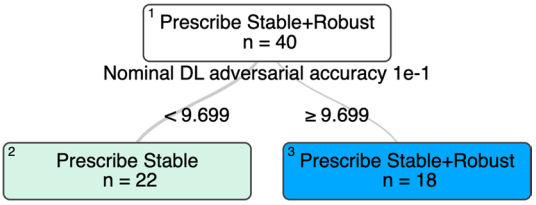


Fig. 4 Optimal policy tree for maximizing natural accuracy

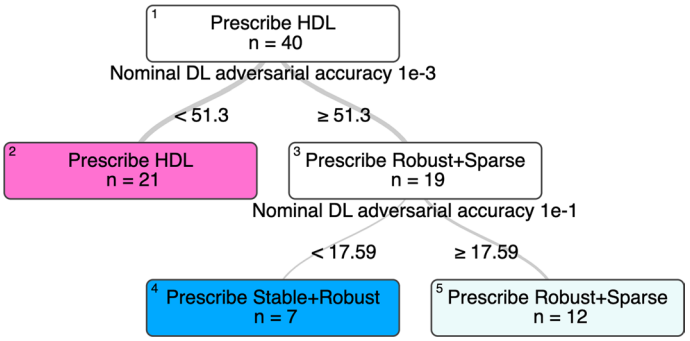


Fig. 5 Optimal policy tree for maximizing robustness ($\rho = 0.1$)

4.4 SHAP values

To gain a deeper understanding of the interplay between individual loss components (robustness, stability, sparsity) and the metrics we measure, we employ the SHAP values method (Lundberg & Lee, 2017). We compute the SHAP values for each UCI data set and average the results over three data set categories: Low Accuracy (< 80%), Medium Accuracy (80%-95%), and High Accuracy (> 95%), with 15 data sets each. The results are shown in Fig. 3.

Our findings confirm that robustness, stability, and sparsity techniques improve the corresponding metrics across all data set categories. More intriguingly, these techniques also positively impact metrics beyond their intended purposes. For example, sparsity and stability enhance adversarial accuracy, while robustness and stability yield sparser networks. This indicates that combining techniques does not necessarily result in any adverse effects and that it is feasible to attain networks with good performance across all metrics. Additionally, the benefits of these techniques are more pronounced in data sets with low initial accuracy, particularly for the accuracy and stability metrics. Lastly, we observe that sparsity generally hurts accuracy and stability, although this highly varies across data sets, as observed in Sect. 4.1.

4.5 Prescriptive approach

In this section, we develop a prescriptive approach that allows users to choose a training loss function based on the specific objective they wish to maximize, which can be a single

Table 6 Performance of prescription trees on the testing set

Objective	Method	Test data sets objective value				
		Cnae-9	Hill-valley	Libras-move	Magic-gamma	Thyroid-ann
Nat. Acc.	DL	93.70	47.21	79.44	87.11	98.86
	Prescribed	94.07	53.61	80.00	87.28	99.05
	Optimal	94.07	53.61	82.50	87.28	99.05
Robustness ($\rho = 0.1$)	DL	0.00	7.54	0.00	15.07	48.42
	Prescribed	3.80	36.39	2.50	64.59	91.79
	Optimal	3.80	40.16	4.72	64.59	91.79
Stability	DL	91.20	43.44	75.00	86.65	98.28
	Prescribed	93.06	45.08	81.94	87.01	98.54
	Optimal	93.06	49.18	81.94	87.01	98.81
Sparsity	DL	0.00	0.00	0.00	0.00	0.00
	Prescribed	73.43	34.89	71.00	57.52	53.22
	Optimal	73.43	41.94	71.00	57.52	53.22
(Nat. Acc. +Robustness +Stability +Sparsity)/4	DL	46.25	24.55	38.61	47.21	61.39
	Prescribed	57.75	39.35	52.76	58.06	73.03
	Optimal	62.07	40.66	52.82	59.62	73.03

evaluation metric or a weighted combination of several metrics. Depending on the data set characteristics and the performance scores of the nominal DL model, we propose a tree-based recommendation model to suggest the most suitable HDL loss function for optimal results with respect to the desired objective.

We train our models using an Optimal Policy Tree (OPT) algorithm (Amram et al., 2022), which uses observational data of the form $(\mathbf{x}_i, y_i, z_i)$. While it is possible to include variability and complexity indicators of the data set as part of $\mathbf{x}_i$ (Lorena et al., 2019), given the extensive and diverse range of data sets in consideration, we choose to capture complexity using the performance metrics achieved by the nominal DL approach on the corresponding classification tasks. In our case, each observation (i.e., data set) i encompasses:

- Data set features $\mathbf{x}_i \in \mathbb{R}^8$: number of features, number of target classes, nominal DL accuracy, nominal DL adversarial accuracy with $\rho = 0.001$, nominal DL adversarial accuracy with $\rho = 0.01$, nominal DL adversarial accuracy with $\rho = 0.1$, nominal DL stability, nominal DL worst case accuracy.
- Prescriptions $z_i \in 1, \ldots, 8$: DL, Robust, Stable, Sparse, Robust+Sparse, Stable+Sparse, Stable+Robust, HDL.
- Outcomes $\boldsymbol{y}_i \in \mathbb{R}^8$, which represent the performance improvement of each method compared to the nominal DL model with respect to the metric set by the user.

Our prescriptive task is to find the optimal policy that, given the information $\mathbf{x}$ of a data set, prescribes the method z leading to the best metric score y. We randomly split the 45 UCI

Table 7 Significance results for the null hypothesis that the average performance achieved by the prescribed method is equal to that one achieved by the nominal DL approach, with alternative hypothesis corresponding to the average performance achieved by the prescribed method being higher

Objective	Leaf prescription	p-value
Nat. Acc.	Stable	0.025
	Stable+Robust	0.0462
Robustness ($\rho = 0.1$)	HDL	$1.508\ e^{-6}$
	Stable+Robust	0.001
	Robust+Sparse	$1.727\ e^{-5}$
Stability	Stable+Robust	0.0161
Sparsity	Stable+Sparse	$1.188\ e^{-38}$
(Nat. Acc. +Robustness +Stability +Sparsity)/4	HDL	$4.472\ e^{-26}$

data sets into a training set (40 data sets) and a test set (5 data sets from different difficulty levels). We cross-validated the optimal tree depth and complexity using the training set.

Figures 4 and 5 represent the OPTs obtained for maximizing two different objectives: natural accuracy and adversarial accuracy. The tree in Fig. 4 highlights that the Stable and Stable+Robust methods are the best suited to obtain high natural accuracy, with the former being preferred when the nominal DL approach has very low adversarial accuracy ($\rho = 0.1$). To maximize robustness, the tree in Fig. 5 prescribes HDL, Stable+Robust, or Robust+Sparse depending on the adversarial accuracy achieved by the nominal DL method.

In addition, we obtained single-leaf trees when maximizing the stability and sparsity objectives. The recommended methods are Stable+Robust for optimizing stability and Stable+Sparse for maximizing sparsity. Lastly, HDL was always the prescribed method when the desired objective was the equally weighted average of all 4 previous metrics.

Finally, Table 6 reports the out-of-sample performance of these prescription trees on the 5 UCI data sets from the test set (cnae-9, hill-valley, libras-movement, magic-gamma, and thyroid-ann). We emphasize that the performance of the prescribed methods is higher than that of the nominal DL approach across all objectives and data sets, and it often matches the performance of the best method.

4.6 Significance analysis

To further validate the improvements achieved by the HDL framework, we analyze the significance of our results with one-sided Welch's t-tests with different variance groups. Specifically, for each evaluation metric and each leaf of the corresponding optimal prescriptive tree, we consider all the UCI and image data sets that fall within that leaf. For those data sets, we test the null hypothesis that the average performance achieved by the prescribed method is equal to that one achieved by the nominal DL approach, with alternative hypothesis corresponding to the average performance achieved by the prescribed method being higher. As shown in Table 7, all p-values are below the 0.05 significance level, concluding that the prescribed methods have statistically significant higher performance than the nominal DL approach across all performance metrics.

5 Conclusions

This paper presents a unifying methodology to obtain deep learning models that are accurate, robust, stable, and sparse by appropriately modifying the objective function to be minimized. Across multiple computational experiments, we show how these 4 metrics interact and demonstrate that we can often train models that simultaneously improve adversarial accuracy, worst-case accuracy, and parameter sparsity without sacrificing natural accuracy. Finally, we provide prescriptive trees that use general features of the data set (e.g. dimension, number of target classes, nominal accuracy, etc.) to recommend which method is more appropriate depending on the desired objective to be maximized, and we show that the improvements achieved by the prescribed methods are statistically significant.

For future research we aim to explore how HDL performs with respect to other data set indicators like variability and complexity, as this could offer further guidance on which method to select. We would also like to test our framework in real world applications; for instance in the area of healthcare, where trustworthy models are crucial and memory constraints are often required for practical use. Consequently, improving the interpretability of the HDL framework would be essential to make it more suitable for such applications. We deem adversarial robustness, stability and sparsity as critical qualities in the development of more reliable machine learning algorithms, and we hope this work will motivate further research in this important field.

Appendix A Results tables

We present the evaluation results for natural accuracy, adversarial accuracy, stability, and sparsity on the test sets across all data sets and methods discussed in the paper. The natural accuracy results can be found in Table 8, adversarial accuracy results in Table 9, stability results in Table 10, and sparsity results in Table 11.

Table 8 Natural accuracy results for all UCI and vision data sets, where n denotes the data size, p denotes the data dimension, and k denotes the number of classes. Darker blue corresponds to higher nominal DL natural accuracy for the UCI data sets

Name	n	p	k	DL	Rob.	St.	Sp.	Rob. +Sp.	St. +Sp.	St. +Rob.	HDL
echocardiogram	131	7	3	40.00	42.22	42.96	40.00	42.22	42.22	37.78	36.30
hill-valley	605	100	2	47.21	47.54	53.61	47.70	47.87	52.30	51.48	51.64
planning-relax	181	12	2	53.51	52.97	54.05	54.05	54.05	54.05	54.59	54.05
poker-hand	25009	10	10	54.87	55.24	55.09	54.89	54.61	54.41	55.19	54.17
hill-valley-noise	605	100	2	56.23	53.44	59.67	57.87	51.31	53.11	53.61	50.82
yeast	1483	8	10	58.45	59.06	59.12	59.80	60.88	59.80	59.46	60.54
haberman-survival	305	3	2	62.58	61.94	61.61	63.23	61.61	61.94	60.97	62.90
glass-identification	213	9	6	64.65	59.53	65.58	61.40	61.40	63.72	61.40	61.40
brst-cancer-ws-prog	197	32	2	71.00	70.50	72.50	71.50	73.00	69.50	71.50	66.50
hayes-roth	131	4	3	74.81	79.26	77.78	73.33	82.22	77.78	74.81	79.26
spectf-heart	79	44	2	76.25	86.25	73.75	73.75	80.00	75.00	83.75	87.50
hepatitis	154	19	2	78.71	80.00	77.42	78.71	79.35	79.35	77.42	79.35
connectionist-bench	989	10	11	79.19	80.30	83.54	78.08	72.32	74.44	83.23	76.67
libras-movement	359	90	15	79.44	82.50	80.00	75.83	80.56	77.22	79.44	80.00
bld-transf-serv-ctr	747	4	2	79.47	79.07	79.07	79.33	77.20	78.93	79.20	79.60
connect-bench-sonar	207	60	2	83.33	86.67	89.52	86.19	85.71	88.10	83.81	86.67
image-segmentation	209	19	7	83.81	87.14	84.29	83.33	89.52	84.76	87.14	83.81
ecoli	335	7	8	84.71	84.41	85.29	85.59	85.59	85.29	84.71	85.00
qsar-biodegradation	1054	41	2	85.12	85.69	85.21	84.55	84.74	85.21	84.93	84.36
parkinsons	194	21	2	86.67	85.64	87.18	88.72	86.15	86.15	86.67	85.13
magic-gamma-tel	19019	10	2	87.11	87.20	86.98	87.17	86.66	86.50	87.28	86.80
letter-recognition	19999	16	26	90.26	89.57	91.15	82.63	87.60	86.31	90.84	88.79
statlog-proj-landsat	4434	36	6	90.39	90.67	90.64	90.44	89.97	90.53	90.46	90.03
wall-robot-nav-24	5455	24	4	92.36	92.23	92.11	92.49	92.89	92.88	92.47	93.08
spambase	4600	57	2	92.73	93.36	93.42	92.94	92.81	93.03	93.12	92.83
seeds	209	7	3	92.86	92.38	91.43	93.33	93.33	92.86	92.38	93.33
ozone-level-eight	2533	72	2	93.69	93.06	93.89	93.37	93.73	93.49	93.69	93.96
cnae-9	1079	856	9	93.70	93.98	94.07	92.87	93.24	92.59	93.89	92.87
balance-scale	624	4	3	94.24	92.96	93.60	91.68	88.64	93.12	94.24	94.56
ionosphere	350	34	2	94.93	94.93	94.37	95.21	91.83	93.24	95.77	94.08
brst-cancer-ws-orig	698	9	2	96.00	96.29	95.86	96.14	96.86	96.00	96.43	96.29
brst-cancer-ws-diag	568	30	2	97.37	96.49	97.37	96.49	96.49	96.14	97.19	96.67
ozone-level-one	2535	72	2	97.40	97.13	97.20	97.01	97.44	97.44	97.28	97.28
wall-robot-nav-4	5455	4	4	97.77	97.82	97.69	98.04	98.44	98.13	97.80	98.33
climate-simu-crash	539	18	2	97.78	97.59	97.78	96.67	96.85	96.67	97.59	97.04
optical-recog-digits	3822	64	10	97.83	98.01	97.73	97.59	98.09	98.07	98.14	97.93
wall-robot-nav-2	5455	2	4	97.88	98.13	98.02	98.30	98.13	98.30	98.33	98.39
dermatology	365	34	6	98.38	98.38	98.38	98.65	98.38	98.38	98.38	98.92
thyroid-disease-new	214	5	3	98.60	99.07	98.14	98.60	99.07	96.74	99.07	97.67
thyroid-disease-ann	3771	21	3	98.86	98.91	98.78	98.70	98.89	98.86	99.05	98.91
wine	177	13	3	98.89	98.33	98.89	100.00	97.78	97.22	97.78	98.33
pen-recog-digits	7493	16	10	99.23	99.05	99.16	99.11	99.41	99.31	99.19	99.41
skin-segmentation	245056	3	2	99.91	99.90	99.90	99.90	99.88	99.89	99.91	99.89
banknote-authent	1371	4	2	99.93	100.00	100.00	99.85	97.16	89.75	100.00	95.71
iris	149	4	3	100.00	99.33	98.00	100.00	92.67	98.67	99.33	99.33
MNIST	70000	784	10	99.19	99.29	99.21	99.13	99.22	99.16	99.31	99.30
Fashion-MNIST	70000	784	10	91.77	91.88	91.93	91.54	92.12	91.43	92.00	92.15
CIFAR10	60000	1024	10	68.42	69.15	68.25	66.60	57.74	69.82	68.78	69.03

Table 9 Adversarial accuracy results for all UCI and vision data sets, where n denotes the data size, p denotes the data dimension, and k denotes the number of classes. We use $\rho = 0.1$ for all data sets except CIFAR10 and Fashion-MNIST, for which we set $\rho = 0.01$. Darker blue corresponds to higher nominal (DL) natural accuracy

Name	n	p	k	DL	Rob.	St.	Sp.	Rob. +Sp.	St. +Sp.	St. +Rob.	HDL
echocardiogram	131	7	3	4.44	44.44	7.41	17.78	44.44	17.78	41.48	44.44
hill-valley	605	100	2	7.54	40.16	28.52	20.00	39.02	20.16	38.52	36.39
planning-relax	181	12	2	21.62	54.05	44.86	49.19	54.05	54.05	54.05	54.05
poker-hand	25009	10	10	50.70	50.70	50.70	50.70	50.70	50.70	50.70	51.95
hill-valley-noise	605	100	2	11.31	21.48	22.30	12.79	30.00	20.98	24.10	34.43
yeast	1483	8	10	0.47	32.26	0.27	0.20	32.26	0.00	32.26	32.26
haberman-survival	305	3	2	17.74	59.68	20.97	47.74	59.68	34.19	59.68	59.68
glass-identification	213	9	6	4.65	20.93	5.58	5.12	20.93	9.30	25.58	25.58
brst-cancer-ws-prog	197	32	2	18.00	72.50	21.00	43.50	72.50	39.50	72.50	72.50
hayes-roth	131	4	3	9.63	32.59	8.89	5.19	36.30	13.33	40.74	40.74
spectf-heart	79	44	2	27.50	25.00	32.50	40.00	25.00	13.75	21.25	25.00
hepatitis	154	19	2	5.81	70.97	9.03	21.94	70.97	29.68	70.97	70.97
connectionist-bench	989	10	11	4.04	1.52	4.34	1.41	6.46	3.84	8.79	8.99
libras-movement	359	90	15	0.00	0.83	0.28	0.00	2.50	0.00	4.72	2.50
bld-transf-serv-ctr	747	4	2	0.00	72.67	0.13	17.60	72.67	43.60	72.67	72.67
connect-bench-sonar	207	60	2	5.71	20.00	18.10	32.38	43.81	33.81	22.86	40.00
image-segmentation	209	19	7	4.29	0.95	2.86	2.38	9.52	3.33	9.52	8.10
ecoli	335	7	8	0.59	41.18	1.76	2.06	51.47	0.88	41.18	51.47
qsar-biodegradation	1054	41	2	36.21	72.04	24.08	31.75	72.04	29.67	72.04	72.04
parkinsons	194	21	2	58.97	74.36	63.08	63.08	74.36	68.72	74.36	74.36
magic-gamma-tel	19019	10	2	15.07	64.59	15.28	18.62	64.59	10.36	64.59	64.59
letter-recognition	19999	16	26	0.76	3.73	1.11	0.52	3.68	0.34	3.57	3.74
statlog-proj-landsat	4434	36	6	8.66	25.48	10.35	7.51	25.48	5.77	20.79	25.39
wall-robot-nav-24	5455	24	4	3.04	39.69	3.28	1.63	39.69	3.06	39.69	39.65
spambase	4600	57	2	40.67	58.41	48.08	45.06	58.41	49.25	58.41	58.41
seeds	209	7	3	23.81	24.29	31.90	27.14	32.38	15.24	31.43	30.95
ozone-level-eight	2533	72	2	49.59	94.48	48.88	94.48	94.48	56.69	94.48	94.48
cnae-9	1079	856	9	0.00	1.02	0.09	0.83	1.94	2.50	3.80	3.80
balance-scale	624	4	3	17.44	40.80	13.28	8.16	43.84	14.72	49.92	49.92
ionosphere	350	34	2	19.44	57.75	25.35	34.65	57.75	10.99	57.75	57.75
brst-cancer-ws-orig	698	9	2	17.71	60.71	23.71	31.29	60.71	15.14	60.71	60.71
brst-cancer-ws-diag	568	30	2	25.44	47.02	25.61	47.02	58.77	13.33	47.02	58.77
ozone-level-one	2535	72	2	73.78	97.44	81.22	97.44	97.44	89.17	97.44	97.44
wall-robot-nav-4	5455	4	4	3.42	39.69	4.85	7.69	39.69	10.49	39.69	39.69
climate-simu-crash	539	18	2	0.00	94.44	2.41	75.56	94.44	83.33	94.44	94.44
optical-recog-digits	3822	64	10	1.36	7.48	1.39	0.76	7.16	0.60	8.94	10.27
wall-robot-nav-2	5455	2	4	5.26	39.69	5.48	17.75	39.69	21.03	39.69	39.69
dermatology	365	34	6	4.59	20.27	2.97	5.68	20.27	10.81	20.27	20.27
thyroid-disease-new	214	5	3	9.77	65.12	10.23	13.02	65.12	13.02	65.12	65.12
thyroid-disease-ann	3771	21	3	48.42	91.79	54.97	54.12	91.79	55.23	91.79	91.79
wine	177	13	3	1.67	44.44	1.11	3.89	37.78	5.56	43.33	31.67
pen-recog-digits	7493	16	10	2.76	10.13	2.63	2.05	10.13	1.80	8.71	10.27
skin-segmentation	245056	3	2	79.30	79.30	79.30	79.30	79.30	79.28	79.30	79.30
banknote-authent	1371	4	2	39.20	54.25	42.25	54.25	54.25	43.13	54.25	54.25
iris	149	4	3	12.00	16.00	12.00	10.00	24.67	10.00	32.67	32.67
MNIST	70000	784	10	49.60	78.37	51.52	43.80	74.67	39.94	79.50	75.97
Fashion-MNIST	70000	784	10	78.70	87.74	78.30	80.76	87.07	80.24	87.80	86.91
CIFAR10	60000	1024	10	28.81	48.61	28.75	34.87	43.98	33.73	47.32	43.96

Table 10 Stability (worst case accuracy) results for all UCI and vision data sets, where n denotes the data size, p denotes the data dimension, and k denotes the number of classes. Darker blue corresponds to higher nominal (DL) natural accuracy

Name	n	p	k	DL	Rob.	St.	Sp.	Rob. +Sp.	St. +Sp.	St. +Rob.	HDL
echocardiogram	131	7	3	40.74	37.04	37.04	40.74	33.33	33.33	40.74	33.33
hill-valley	605	100	2	43.44	43.44	49.18	43.44	46.72	50.82	45.08	49.18
planning-relax	181	12	2	51.35	48.65	54.05	54.05	54.05	51.35	54.05	54.05
poker-hand	25009	10	10	54.60	54.32	54.56	54.48	53.20	53.52	54.70	52.32
hill-valley-noise	605	100	2	54.92	47.54	52.46	54.92	47.54	47.54	48.36	50.00
yeast	1483	8	10	57.58	56.90	56.57	57.24	59.60	58.92	56.90	58.92
haberman-survival	305	3	2	59.68	59.68	59.68	59.68	59.68	59.68	59.68	59.68
glass-identification	213	9	6	55.81	58.14	53.49	58.14	55.81	58.14	58.14	58.14
brst-cancer-ws-prog	197	32	2	67.50	67.50	67.50	67.50	62.50	60.00	72.50	67.50
hayes-roth	131	4	3	70.37	74.07	70.37	70.37	70.37	74.07	70.37	74.07
spectf-heart	79	44	2	68.75	68.75	75.00	68.75	68.75	62.50	75.00	75.00
hepatitis	154	19	2	70.97	70.97	70.97	70.97	74.19	70.97	70.97	77.42
connectionist-bench	989	10	11	75.25	77.78	80.81	71.72	63.13	60.61	76.26	70.71
libras-movement	359	90	15	75.00	75.00	79.17	68.06	75.00	70.83	81.94	75.00
bld-transf-serv-ctr	747	4	2	79.33	77.33	78.00	78.67	74.67	74.00	78.00	79.33
connect-bench-sonar	207	60	2	78.57	80.95	83.33	80.95	83.33	83.33	80.95	83.33
image-segmentation	209	19	7	78.57	78.57	80.95	76.19	73.81	83.33	78.57	78.57
ecoli	335	7	8	80.88	77.94	83.82	82.35	83.82	83.82	83.82	82.35
qsar-biodegradation	1054	41	2	84.83	84.83	83.89	81.99	83.41	83.89	84.36	83.89
parkinsons	194	21	2	84.62	82.05	82.05	84.62	79.49	76.92	84.62	79.49
magic-gamma-tel	19019	10	2	86.65	86.93	86.44	86.75	85.73	86.01	87.01	86.25
letter-recognition	19999	16	26	86.98	86.75	89.60	82.27	84.47	83.85	90.20	86.55
statlog-proj-landsat	4434	36	6	88.84	88.05	90.08	90.19	88.61	89.40	89.29	88.39
wall-robot-nav-24	5455	24	4	92.03	91.12	91.58	92.31	92.03	92.22	90.75	92.40
spambase	4600	57	2	92.18	92.73	92.62	92.29	92.40	92.51	92.40	92.62
seeds	209	7	3	90.48	90.48	92.86	92.86	92.86	90.48	90.48	92.86
ozone-level-eight	2533	72	2	93.10	94.08	93.10	93.29	94.48	92.31	93.10	93.49
cnae-9	1079	856	9	91.20	93.06	92.59	92.13	92.59	92.13	93.06	91.20
balance-scale	624	4	3	91.20	91.20	92.00	90.40	72.80	91.20	91.20	92.80
ionosphere	350	34	2	92.96	90.14	92.96	91.55	94.37	91.55	90.14	91.55
brst-cancer-ws-orig	698	9	2	93.57	93.57	95.71	95.71	95.71	95.00	95.71	95.71
brst-cancer-ws-diag	568	30	2	95.61	94.74	95.61	94.74	95.61	93.86	94.74	95.61
ozone-level-one	2535	72	2	97.24	96.46	96.85	96.85	97.44	97.44	96.85	97.05
wall-robot-nav-4	5455	4	4	97.44	97.16	97.34	97.25	97.71	97.62	97.62	97.80
climate-simu-crash	539	18	2	96.30	96.30	95.37	94.44	96.30	93.52	96.30	96.30
optical-recog-digits	3822	64	10	97.39	97.39	96.99	97.12	97.91	97.91	97.78	97.65
wall-robot-nav-2	5455	2	4	96.98	97.53	97.34	97.25	97.53	98.17	97.89	97.44
dermatology	365	34	6	97.30	97.30	97.30	98.65	95.95	97.30	97.30	94.59
thyroid-disease-new	214	5	3	97.67	97.67	95.35	93.02	95.35	93.02	97.67	93.02
thyroid-disease-ann	3771	21	3	98.28	98.81	98.01	98.54	98.68	98.41	98.54	98.28
wine	177	13	3	94.44	97.22	94.44	94.44	97.22	88.89	97.22	97.22
pen-recog-digits	7493	16	10	98.93	98.67	99.00	99.07	99.13	99.33	99.07	99.07
skin-segmentation	245056	3	2	99.90	99.89	99.89	99.89	99.86	99.87	99.90	99.87
banknote-authent	1371	4	2	99.64	100.00	100.00	99.64	87.27	70.18	100.00	87.27
iris	149	4	3	100.00	100.00	93.33	100.00	70.00	96.67	96.67	96.67
MNIST	70000	784	10	99.14	99.24	99.15	99.06	99.07	98.86	99.25	99.19
Fashion-MNIST	70000	784	10	91.53	91.36	91.75	91.29	91.65	91.15	91.38	90.19
CIFAR10	60000	1024	10	68.28	65.44	68.13	63.14	56.04	61.76	63.76	56.48

Table 11 Sparsity results for all UCI and vision data sets, where n denotes the data size, p denotes the data dimension, and k denotes the number of classes. Darker blue corresponds to higher nominal (DL) natural accuracy

Name	n	p	k	DL	Rob.	St.	Sp.	Rob. +Sp.	St. +Sp.	St. +Rob.	HDL
echocardiogram	131	7	3	0.0	0.0	0.0	52.17	62.57	65.69	0.0	65.49
hill-valley	605	100	2	0.0	0.0	0.0	49.00	41.94	34.89	0.0	28.80
planning-relax	181	12	2	0.0	0.0	0.0	54.47	63.00	69.13	0.0	70.56
poker-hand	25009	10	10	0.0	0.0	0.0	29.56	49.23	51.31	0.0	49.83
hill-valley-noise	605	100	2	0.0	0.0	0.0	50.67	39.73	45.04	0.0	29.28
yeast	1483	8	10	0.0	0.0	0.0	53.08	53.41	54.71	0.0	54.36
haberman-survival	305	3	2	0.0	0.0	0.0	47.21	49.40	53.34	0.0	53.10
glass-identification	213	9	6	0.0	0.0	0.0	56.14	63.65	64.86	0.0	64.85
brst-cancer-ws-prog	197	32	2	0.0	0.0	0.0	61.87	67.34	69.49	0.0	67.78
hayes-roth	131	4	3	0.0	0.0	0.0	61.56	67.49	66.39	0.0	67.38
spectf-heart	79	44	2	0.0	0.0	0.0	81.02	85.84	86.10	0.0	85.58
hepatitis	154	19	2	0.0	0.0	0.0	64.56	68.71	74.01	0.0	72.74
connectionist-bench	989	10	11	0.0	0.0	0.0	57.95	63.20	63.34	0.0	63.45
libras-movement	359	90	15	0.0	0.0	0.0	66.76	70.90	71.00	0.0	70.73
bld-transf-serv-ctr	747	4	2	0.0	0.0	0.0	46.21	48.90	52.65	0.0	50.52
connect-bench-sonar	207	60	2	0.0	0.0	0.0	76.09	80.04	81.10	0.0	80.07
image-segmentation	209	19	7	0.0	0.0	0.0	60.95	66.47	69.23	0.0	67.24
ecoli	335	7	8	0.0	0.0	0.0	54.91	62.25	63.26	0.0	62.31
qsar-biodegradation	1054	41	2	0.0	0.0	0.0	44.89	44.32	51.82	0.0	47.65
parkinsons	194	21	2	0.0	0.0	0.0	59.42	60.95	67.40	0.0	63.23
magic-gamma-tel	19019	10	2	0.0	0.0	0.0	50.80	51.43	57.52	0.0	52.84
letter-recognition	19999	16	26	0.0	0.0	0.0	58.77	63.84	65.32	0.0	64.86
statlog-proj-landsat	4434	36	6	0.0	0.0	0.0	45.65	50.60	52.06	0.0	51.53
wall-robot-nav-24	5455	24	4	0.0	0.0	0.0	51.85	55.96	56.59	0.0	55.53
spambase	4600	57	2	0.0	0.0	0.0	56.81	56.56	59.90	0.0	55.84
seeds	209	7	3	0.0	0.0	0.0	65.56	71.45	72.73	0.0	72.33
ozone-level-eight	2533	72	2	0.0	0.0	0.0	66.09	68.13	67.86	0.0	67.38
cnae-9	1079	856	9	0.0	0.0	0.0	61.94	62.06	73.43	0.0	61.91
balance-scale	624	4	3	0.0	0.0	0.0	57.25	63.41	63.35	0.0	63.14
ionosphere	350	34	2	0.0	0.0	0.0	63.16	66.76	68.17	0.0	66.74
brst-cancer-ws-orig	698	9	2	0.0	0.0	0.0	48.29	53.44	55.78	0.0	55.97
brst-cancer-ws-diag	568	30	2	0.0	0.0	0.0	68.91	71.38	71.46	0.0	70.69
ozone-level-one	2535	72	2	0.0	0.0	0.0	66.22	68.25	67.97	0.0	68.58
wall-robot-nav-4	5455	4	4	0.0	0.0	0.0	52.89	61.43	60.75	0.0	60.50
climate-simu-crash	539	18	2	0.0	0.0	0.0	59.75	64.56	65.88	0.0	66.29
optical-recog-digits	3822	64	10	0.0	0.0	0.0	65.42	68.85	71.25	0.0	69.22
wall-robot-nav-2	5455	2	4	0.0	0.0	0.0	63.85	72.12	72.68	0.0	71.74
dermatology	365	34	6	0.0	0.0	0.0	67.03	73.49	74.61	0.0	74.21
thyroid-disease-new	214	5	3	0.0	0.0	0.0	67.25	74.48	74.62	0.0	74.12
thyroid-disease-ann	3771	21	3	0.0	0.0	0.0	48.81	50.15	53.22	0.0	50.36
wine	177	13	3	0.0	0.0	0.0	74.62	81.05	80.60	0.0	80.59
pen-recog-digits	7493	16	10	0.0	0.0	0.0	59.13	62.58	63.33	0.0	63.23
skin-segmentation	245056	3	2	0.0	0.0	0.0	63.77	71.27	72.15	0.0	72.56
banknote-authent	1371	4	2	0.0	0.0	0.0	76.88	84.17	84.97	0.0	84.98
iris	149	4	3	0.0	0.0	0.0	69.89	78.03	77.56	0.0	77.79
MNIST	70000	784	10	0.0	0.0	0.0	39.20	75.06	44.77	0.0	76.05
Fashion-MNIST	70000	784	10	0.0	0.0	0.0	45.41	68.94	48.40	0.0	69.18
CIFAR10	60000	1024	10	0.0	0.0	0.0	50.51	82.34	55.58	0.0	81.41

Acknowledgements We would like to thank the editor and the reviewers of the paper for their valuable comments, which contributed to enhance the paper.

Author contributions DB directed overall research and edited the manuscript. KVC led model development, wrote code, designed and performed experiments, analyzed results, and wrote the manuscript. LB wrote code, designed and performed experiments, analyzed results, and wrote the manuscript. MLL designed and performed experiments, analyzed results, and wrote the manuscript, AP, IP performed experiments and helped write the manuscript.

Funding 'Open Access funding provided by the MIT Libraries'. The authors have no relevant financial or non-financial interests to disclose.

Data and code availability All the code and data to reproduce the results can be found at https://github.com/kimvc7/HDL.

Declarations

Conflict of interest The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Ethical approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Abadi, M., Agarwal, A., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org/, software available from tensorflow.org.

Aghasi, A., Abdi, A., & Romberg, J. (2020). Fast convex pruning of deep neural networks. *SIAM Journal on Mathematics of Data Science, 2*(1), 158–188.

Amram, M., Dunn, J., & Zhuo, Y. D. (2022). Optimal policy trees. *Machine Learning, 111*, 2741–2768.

Anderson, R., Huchette, J., Ma, W., et al. (2020). Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming* (pp. 1–37).

Athalye, A., Carlini, N., & Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. arXiv:1802.00420.

Bellec, G., Kappel, D., Maass, W., et al. (2017). Deep rewiring: Training very sparse deep networks. *CoRR*. arXiv:1711.05136.

Bertsimas, D., & Paskov, I. (2020). Stable regression: On the power of optimization over randomization in training regression problems. *Journal of Machine Learning Research, 21*(230), 1–25.

Bertsimas, D., Pauphilet, J., & Parys, B. V. (2020). Sparse regression: Scalable algorithms and empirical performance. *Statistical Science, 35*(4), 555–578. https://doi.org/10.1214/19-STS701

Bertsimas, D., Boix, X., Carballo, K. V., et al. (2023). Robust Upper Bounds for Adversarial Training. arXiv:2112.09279.

Bertsimas, D., Dunn, J., & Paskov, I. (2022). Stable classification. *Journal of Machine Learning Research, 23*(296), 1–53.

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)* (pp. 39–57).

Changpinyo, S., Sandler, M., & Zhmoginov, A. (2017). The power of sparsity in convolutional neural networks. arXiv:1702.06257.

Cohen, J., Rosenfeld, E., & Kolter, Z. (2019). Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning, PMLR* (pp. 1310–1320).

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine, 29*(6), 141–142.

Dua, D., & Graff, C. (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml.

Dvijotham, K., Stanforth, R., Gowal, S., et al. (2018). A dual approach to scalable verification of deep networks. In *UAI*, p 3.

Gale, T., Elsen, E., & Hooker, S. (2019). The state of sparsity in deep neural networks. *CoRR*. arXiv:1902.09574.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In: Teh YW, Titterington M (eds) *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, vol 9. PMLR, Chia Laguna Resort, Sardinia, Italy* (pp. 249–256), https://proceedings.mlr.press/v9/glorot10a.html.

Goldwasser, S., Kalai, A. T., Kalai, Y., et al. (2020). Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. *Advances in Neural Information Processing Systems, 33*, 15859–15870.

Goodfellow, IJ., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv:1412.6572.

Han, S., Pool, J., Tran, J., et al. (2015). Learning both weights and connections for efficient neural network. *Advances in Neural Information Processing Systems, 28*.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). The Elements of Statistical Learning. Springer Series in Statistics, Springer New York Inc.

Hein, M., & Andriushchenko, M. (2017). Formal guarantees on the robustness of a classifier against adversarial manipulation. *Advances in Neural Information Processing Systems, 30*.

Hoefler, T., Alistarh, D., Ben-Nun, T., et al. (2021). Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *The Journal of Machine Learning Research, 22*(1), 10882–11005.

Ilyas, A., Jalal, A., Asteri, E., et al. (2017). The robust manifold defense: Adversarial training using generative models. *CoRR*. arXiv:1712.09196.

Janowsky, S. A. (1989). Pruning versus clipping in neural networks. *Physical Review A, 39*, 6600–6603. https://doi.org/10.1103/PhysRevA.39.6600

Kabilan, V. M., Morris, B., Nguyen, H. P., et al. (2021). Vectordefense: Vectorization as a defense to adversarial examples. *Soft Computing for Biomedical Applications and Related Topics* (pp. 19–35).

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images. Tech. rep.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. Curran Associates Inc., Red Hook, NY, USA, NIPS'12* (pp. 1097–1105).

Krogh, A., & Vedelsby, J. (1994). Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems, 7*.

Lamb, A., Binas, J., Goyal, A., et al. (2018). Fortified networks: Improving the robustness of deep networks by modeling the manifold of hidden representations. arXiv:1804.02485.

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. *Advances in Neural Information Processing Systems, 2*.

Lecuyer, M., Atlidakis, V., Geambasu, R., et al. (2019). Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP), IEEE* (pp. 656–672).

Li, H., Kadav, A., Durdanovic, I., et al. (2016). Pruning filters for efficient convnets. *CoRR*. arXiv:1608.08710.

Liu, E. Z., Haghgoo, B., Chen, A. S., et al. (2021). Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning, PMLR* (pp. 6781–6792).

Lorena, A. C., Garcia, L. P., Lehmann, J., et al. (2019). How complex is your classification problem? A survey on measuring classification complexity. *ACM Computing Surveys (CSUR), 52*(5), 1–34.

Louizos, C., Welling, M., & Kingma, D. P. (2017). Learning sparse neural networks through l_0 regularization. arXiv:1712.01312.

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems, 30*.

Madry, A., Makelov, A., Schmidt, L., et al. (2017). Towards deep learning models resistant to adversarial attacks. arXiv:1706.06083.

May, R. J., Maier, H. R., & Dandy, G. C. (2010). Data splitting for artificial neural networks using SOM-based stratified sampling. *Neural Networks, 23*(2), 283–294.

Mocanu, D. C., Mocanu, E., Stone, P., et al. (2017). Evolutionary training of sparse artificial neural networks: A network science perspective. *CoRR*. arXiv:1707.04780.

Mostafa, H., & Wang, X. (2019). Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning, PMLR* (pp. 4646–4655).

Narang, S., Elsen, E., Diamos, G., et al. (2017). Exploring sparsity in recurrent neural networks. arXiv:1704.05119.

Prakash, A., Moran, N., Garber, S., et al. (2018). Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 8571–8580).
Raghunathan, A., Steinhardt, J., & Liang, P. S. (2018). Semidefinite relaxations for certifying robustness to adversarial examples. *Advances in Neural Information Processing Systems, 31*.
Ross, A., & Doshi-Velez, F. (2018). Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
Sagawa, S., Koh, P. W., Hashimoto, T. B., et al. (2019). Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *CoRR*. arXiv:1911.08731.
Savarese, P., Silva, H., & Maire, M. (2020). Winning the lottery with continuous sparsification. *Advances in Neural Information Processing Systems, 33*, 11380–11390.
Singh, G., Gehr, T., Mirman, M., et al. (2018). Fast and effective robustness certification. *NeurIPS, 1*(4), 6.
Staib, M., & Jegelka, S. (2019). Distributionally robust optimization and generalization in kernel methods. *Advances in Neural Information Processing Systems, 32*.
Szegedy, C., Zaremba, W., Sutskever, I., et al. (2014). Intriguing properties of neural networks. In *International Conference on Learning Representations*, arXiv:1312.6199.
Thompson, N. C., Greenewald, K. H., Lee, K., et al. (2020). The computational limits of deep learning. *CoRR*. arXiv:2007.05558.
Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley: CreateSpace.
Weng, L., Zhang, H., Chen, H., et al. (2018). Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning, PMLR* (pp. 5276–5285).
Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747.
Xie, C., Wang, J., Zhang, Z., et al. (2017). Mitigating adversarial effects through randomization. arXiv:1711.01991.
Xu, Y., & Goodacre, R. (2018). On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of Analysis and Testing, 2*(3), 249–262.
Yan, Z., Guo, Y., & Zhang, C. (2018). Deep defense: Training dnns with improved adversarial robustness. *Advances in Neural Information Processing Systems, 31*.
Zhang, H., Weng, T. W., Chen, P. Y., et al. (2018). Efficient neural network robustness certification with general activation functions. *Advances in Neural Information Processing Systems, 31*.
Zhuang, Z., Tan, M., Zhuang, B., et al. (2018). Discrimination-aware channel pruning for deep neural networks. *Advances in Neural Information Processing Systems, 31*.