2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

A minimax optimal control approach for robust neural ODEs

Cristina Cipriani[1], Alessandro Scagliotti[2] and Tobias Wöhrer[3]

***Abstract*— In this paper, we address the adversarial training of neural ODEs from a robust control perspective. This is an alternative to the classical training via empirical risk minimization, and it is widely used to enforce reliable outcomes for input perturbations. Neural ODEs allow the interpretation of deep neural networks as discretizations of control systems, unlocking powerful tools from control theory for the development and the understanding of machine learning. In this specific case, we formulate the adversarial training with perturbed data as a minimax optimal control problem, for which we derive first order optimality conditions in the form of Pontryagin's Maximum Principle. We provide a novel interpretation of robust training leading to an alternative weighted technique, which we test on a low-dimensional classification task.**

I. INTRODUCTION

The recent advances in deep neural networks (DNNs) combined with the rapidly increasing supply of computing power have lead to numerous breakthroughs in science and technology [1]. DNN algorithms have shown to possess remarkable data generalization properties and flexible application settings [2], [3]. Nonetheless, it has become evident that many machine learning (ML) models suffer from a severe lack of robustness against data manipulation. *Adversarial attacks* describe inputs with minuscule changes, e.g. pictures with added pixel noise that is invisible in low resolution, which result in dramatic changes in the model outputs [4], [5]. Such attack vulnerabilities limit the implementations of ML algorithms in areas that demand high reliability, such as self-driving vehicles or automated content filtering. The robustness crisis further highlights the general lack of theoretical understanding and interpretability of neural network algorithms.

A central theoretical achievement of recent years is the interpretation of DNNs, such as residual NNs, as dynamical systems [6], [7]: The input-to-output information flow through a network with an infinite amount of layers can be formulated in the continuum limit. This leads to nonlinear *neural ODEs* (nODE) [8], [9], where time takes the role of the continuous-depth variable. The nODE vantage point has proven to be a powerful tool that allows to interpret the learning problems (or parameter training) of DNNs as continuous-time control problems and to formulate *Pontryagin's Maximum Principle* (PMP) [10] in various network settings [11], [8], [12]. The link between the training of DNNs with batches of input and simultaneous control led to numerous additional theoretical insights into neural networks [13], [14], [15].

In the context of nODEs, an adversarial attack in predefined norm $\|\cdot\|$ of *adversarial budget* $\varepsilon > 0$ takes the form of a perturbed initial data point $\tilde{x}^0 = x^0 + \alpha$ where $\|\alpha\| \leq \varepsilon$. As a result, the *adversarially robust learning problem* has been established [16], [17] as the minimax control problem

$$\min_u \mathbb{E}_{(x^0,y)\sim\mu}\left[\max_{\|\alpha\|\leq\varepsilon} \mathrm{Loss}(u, x^0+\alpha, y)\right] \tag{1}$$

where the initial data and labels (x^0, y) are drawn from an underlying (and generally unknown) data distribution μ and where the function $\mathrm{Loss}(u, x^0, y)$ quantifies the prediction accuracy resulting from the model control u. While such saddle-point problems have a long history [18], their non-smooth, non-convex nature pose challenges in obtaining theoretical result. Additionally, the inner maximization in (1) is x^0 dependent and in applications often high-dimensional. Numerical implementations are therefore typically substituting the maximization by first-order guesses of worst-case adversarial attacks [5], [16], [19], [20].

Contributions: In this work, we interpret the forward flow of DNNs as nonlinear neural ODEs and take a control theoretical approach. We investigate the non-smooth and non-convex robust learning problem (1) by formulating the corresponding PMP for a finite ensemble of data points. We provide a proof of this classical result by means of separation of Boltyanski approximating cones. In spite of being a well-known result, the novelty consists in relating the solution of the resulting PMP to the one of another smooth and properly weighted control problem. Inspired by this interpretation, we develop a numerical method to address the robust learning problem, and we test it on a bidimensional classification task.

II. ROBUST CONTROL MODEL

In this paper, we consider the simultaneous and robust control problem of a finite number of particles in $\mathbb{R}^d$, whose initial positions are subject to perturbations (also known as *attacks* in the machine learning literature). Namely, the control system driving the dynamics of a single particle is:

$$\begin{cases} \dot{x}(t) = F(x(t), u(t)), & \text{a.e. } t \in [0,T], \\ x(0) = x^0, \end{cases} \tag{2}$$

where $x^0 \in \mathbb{R}^d$ represents the unperturbed initial datum, $F : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$ is a continuous function (with Lipschitz dependence in the first variable), and $u \in \mathcal{U} := L^2([0,T], U)$ is an admissible control taking values in any compact set $U \subset \mathbb{R}^m$. Here, we consider a collection (or *batch*) of M

[1,2,3]Technical University Munich (TUM), Department of Mathematics, Munich, Germany.
[1]Munich Data Science Institute (MDSI), Munich, Germany
[1,2]Munich Center for Machine Learning (MCML), Munich, Germany
cristina.cipriani@ma.tum.de, scag@ma.tum.de, tobias.woehrer@tum.de

initial data points $\{x_1^0, \dots, x_M^0\}$ for the system (2), which are *simultaneously* driven by the same admissible control $u \in \mathcal{U}$. In addition, each of these Cauchy data points is affected by N additive perturbations, i.e.,

$$\mathcal{A}_i^N := \{\alpha_i^1, \dots, \alpha_i^N\}, \quad \forall i = 1, \dots, M,$$

where $\alpha_i^j \in \mathbb{R}^d$ $(j = 1, \dots, N)$ is the j-th perturbation of x_i^0. We denote by $x_i^j : [0,T] \to \mathbb{R}^d$ the solution of (2) corresponding to the initial condition $x_i^j(0) = x_i^0 + \alpha_i^j$, while we refer to the collection of the N perturbations of the $i-$th particle using $X_i(\cdot) := (x_i^1(\cdot), \dots, x_i^N(\cdot)) \in \mathbb{R}^{d\times N}$. Finally, we employ the variable $X = (X_1, \dots, X_M) \in (\mathbb{R}^{d\times N})^M$ to describe the states of the whole system, i.e., the ensemble of every perturbation of every particle. We are interested in the minimization of the functional $J : \mathcal{U} \to \mathbb{R}$, defined as

$$\begin{aligned} J(u) &:= \max_{\substack{j_1=1,\dots,N \\ \dots \\ j_M=1,\dots,N}} \frac{1}{M} \sum_{i=1}^M g_i\big(x_i^{j_i}(T)\big) \\ &= \frac{1}{M} \sum_{i=1}^M \max_{j=1,\dots,N} g_i\big(x_i^j(T)\big) = \frac{1}{M} \sum_{i=1}^M \tilde{g}_i(X_i(T)), \end{aligned} \tag{3}$$

where $x_i^j(T)$ is the terminal-time value of the solution of (2) starting from $x_i^0 + \alpha_i^j$ and steered by the control $u \in \mathcal{U}$, while, for every $i = 1, \dots, M$, $\tilde{g}_i : \mathbb{R}^{d\times N} \to \mathbb{R}$ is the non-smooth function defined as $\tilde{g}_i(X_i) = \max(g_i(x_i^1), \dots, g_i(x_i^N))$, and $g_i : \mathbb{R}^d \to \mathbb{R}$ is a smooth convex function. Notice that for each datum x_i, we must consider various perturbations α_i^j, depending on the initial datum. This motivates the need to compute the maximum over different indices j_i, which depend on the datum i. However, in the second identity of (3), we exploit the fact that each perturbation is independent from the others to swap the sum and the maximization.
In machine learning, the minimization of (3) can be rephrased as the empirical loss minimization of the robust training problem (1), where g_i incorporates the i-th data label, which has been denoted with the variable y in (1).

Definition 1. Given $u \in \mathcal{U}$ and the corresponding collection of trajectories $X : [0,T] \to (\mathbb{R}^{d\times N})^M$ that solve (2) with the perturbed initial data, we call the couple (u, X) an *admissible process*. Moreover an admissible process $(\bar{u}, \bar{X})$ is a *strong local minimizer* for (3) if there exists an $\varepsilon > 0$ such that, for every other admissible process (u, X) satisfying $\|X(\cdot) - \bar{X}(\cdot)\|_{C^0} \le \varepsilon$, we have

$$\frac{1}{M} \sum_{i=1}^M \tilde{g}_i(\bar{X}_i(T)) \le \frac{1}{M} \sum_{i=1}^M \tilde{g}_i(X_i(T)).$$

Moreover, we further introduce

$$\dot{X}(t) = \mathcal{F}(X(t), u(t)), \quad \text{a.e. } t \in [0,T], \tag{4}$$

to denote the evolution of the whole ensemble of trajectories in $(\mathbb{R}^{d\times N})^M$, where the mapping $\mathcal{F} : (\mathbb{R}^{d\times N})^M \times \mathbb{R}^m \to (\mathbb{R}^{d\times N})^M$ should be understood as the application of $F(\cdot, u)$ component-wise. This allows us to conveniently define the Hamiltonian $H : (\mathbb{R}^{d\times N})^M \times ((\mathbb{R}^{d\times N})^M)^\star \times \mathbb{R}^m \to \mathbb{R}$ as

$$H(X, P, u) := P \cdot \mathcal{F}(X, u) = \sum_{i=1}^M \sum_{j=1}^N p_i^j \cdot F\big(x_i^j, u\big),$$

where $P = (P_1, \dots, P_M)$, $P_i = (p_i^1, \dots, p_i^N)$, and $p_i^j \in \mathbb{R}^d$ for every $i = 1, \dots, M$ and every $j = 1, \dots, N$.

A. Pontryagin Maximum Principle

Below, we specify the hypotheses needed to formulate the necessary conditions for strong local minimizers of (3).

Assumption 2. We assume that

(A1) the function $F : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$ that defines the dynamics in (2) is continuous, and it is C^2-regular in the first variable. Moreover, there exists $C > 0$ such that

$$|F(x, \omega) - F(y, \omega)|_2 \le C(|\omega|_2 + 1)|x - y|_2,$$

for every $x, y \in \mathbb{R}^d$ and for every $\omega \in \mathbb{R}^m$;

(A2) for every $i = 1, \dots, M$, the function $g_i : \mathbb{R}^d \to \mathbb{R}$ related to the terminal-cost in (3) is convex, C^1-regular, bounded from below and never equal to $+\infty$.

From now on, when we consider a convex function $f : \mathbb{R}^d \to \mathbb{R}$, we assume that it never attains the value $+\infty$. Let us now present the main result which motivates our numerical approach presented in Section IV.

Theorem 3 (PMP for minimax)**.** *Let $(\bar{u}, \bar{X})$ be a strong local minimizer for the minimax optimal control problem related to (3). Then, there exist coefficients $(\gamma_i^j)_{i=1,\dots,M}^{j=1,\dots,N}$ satisfying $\gamma_i^j \ge 0$ and $\sum_{j=1}^N \gamma_i^j = 1 \quad \forall i = 1, \dots M$, and there exists a curve $P : [0,T] \to \big((\mathbb{R}^{d\times N})^M\big)^\star$ whose components solve the adjoint equations*

$$\begin{cases} \dot{p}_i^j(t) = p_i^j(t) \nabla_x F\big(\bar{x}_i^j(t), \bar{u}(t)\big), & a.e.\ t \in [0,T], \\ p_i^j(T) = -\frac{1}{M} \gamma_i^j \nabla_x g_i\big(\bar{x}_i^j(T)\big), \end{cases} \tag{5}$$

such that, for a.e. $t \in [0,T]$, it holds that

$$H(\bar{X}(t), P(t), \bar{u}(t)) = \max_{\omega \in U} H(\bar{X}(t), P(t), \omega). \tag{6}$$

Moreover, for every $i = 1, \dots, M$, if $\gamma_i^j > 0$, then

$$j \in \arg\max_{j=1,\dots,N} g_i\big(x_i^j(T)\big).$$

Remark 4. In Theorem 3 only normal extremals are considered, since the problem of minimizing the functional $J : \mathcal{U} \to \mathbb{R}$ defined in (3) does not admit abnormal extremals.

Remark 5. Here we consider controls $u : [0,T] \to \mathbb{R}^m$ taking values in an arbitrary compact set $U \subset \mathbb{R}^m$. However, it is possible to remove this constraint, and to add in the definition (3) of the functional J the penalization $\beta \|u\|_{L^2}^2$ on the energy of the control, where $\beta > 0$ is a parameter tuning this regularization. In this case, Theorem 3 holds in the same form with a slightly different Hamiltonian, which has now to take into account the running cost.

Remark 6. In (5) it is possible to assign the following alternative terminal-time Cauchy datum:

$$p_i^j(T) = -\nabla_x g_i\big(x_i^j(T)\big)$$

for every $i = 1, \ldots, M$ and $j = 1, \ldots, N$. In this case, we introduce the probability measure $d\Gamma_i$ on the set $\mathcal{A}_i^N = \{\alpha_i^1, \ldots, \alpha_i^N\}$, defined as $d\Gamma_i(\alpha_i^j) = \gamma_i^j$, where $(\gamma_i^j)_{i=1,\ldots,M}^{j=1,\ldots,N}$ are the coefficients prescribed by Theorem 3. Therefore, we can rephrase (6) as

$$\frac{1}{M}\sum_{i=1}^{M}\int_{\mathcal{A}_i^N} p_i^j(t)\cdot F\big(\bar{x}_i^j(t),\bar{u}(t)\big)\,d\Gamma_i(\alpha_i^j)$$
$$= \max_{\omega\in U}\frac{1}{M}\sum_{i=1}^{M}\int_{\mathcal{A}_i^N} p_i^j(t)\cdot F(\bar{x}_i^j(t),\omega)\,d\Gamma_i(\alpha_i^j).$$

This formulation is particularly suitable to generalize to the case of infinitely many perturbations, that has been investigated in [21]. Furthermore, the case of infinitely many particles with finitely many perturbations per particle can be treated with a mean-field analysis, as recently done in [22], [13] in the smooth framework.

B. *Interpretation of the PMP*

An important message conveyed by Theorem 3 is that *any strong local minimizer of the minimax problem related to the functional J in (3) is an extremal for another smooth and properly weighted optimal control problem.* Namely, let us consider the functional $J_\Gamma : \mathcal{U} \to \mathbb{R}$ defined as

$$J_\Gamma(u) = \frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}\gamma_i^j g_i\big(x_i^j(T)\big), \tag{7}$$

where $x_i^j : [0,T] \to \mathbb{R}^d$ are the same as in (3), and where $(\gamma_i^j)_{i=1,\ldots,M}^{j=1,\ldots,N}$ are precisely the non-negative coefficients prescribed by Theorem 3 for the strong local minimizer $(\bar{u},\bar{X})$ of J. Then, it turns out that $(\bar{u},\bar{X})$ satisfies the corresponding PMP conditions for the optimal control problem associated to J_Γ, which is *smooth* with respect to the terminal-states. This fact is extremely interesting from a numerical perspective, since the minimization of a functional with smooth terminal cost can be, in general, more easily addressed. On the other hand, the coefficients $(\gamma_i^j)_{i=1,\ldots,M}^{j=1,\ldots,N}$ are not explicitly available a priori, as they are themselves part of the task of finding a strong local minimizer of J. In Section IV, we propose and numerically test a procedure to iteratively adjust these coefficients.

III. Proof of the PMP

The proof of Theorem 3 is classical and can be deduced, e.g., from [23, Theorem 6.4.1]. However, in that case, the proof is rather sophisticated since it is developed in a very general framework. In this section, we propose a more direct proof based on an *Abstract Maximum Principle*, for which we refer to the recent expository paper [24].

A. *Abstract Maximum Principle*

In order to prove Theorem 3, we first need to collect some well known results of convex and non-convex analysis.

Lemma 7 (Result from [25], Lemma 2.1.1 (Part D))**.** *Given a convex function $f : \mathbb{R}^n \to \mathbb{R}$ and $x \in \mathbb{R}^n$, then*

$$f(x+h) = f(x) + \sigma_{\partial f(x)}(h) + o(|h|) \quad \textit{as } h \to 0, \tag{8}$$

where $\partial f(x)$ denotes the subdifferential of f at x.

We recall that, given a convex and compact set $K \subset \mathbb{R}^n$, the *support function* $\sigma_K : \mathbb{R}^n \to \mathbb{R}$ is the convex sublinear function:

$$\sigma_K(x) := \sup_{p\in K}\langle p, x\rangle.$$

For a thorough discussion on the properties of support functions, we recommend [25, Part C].

Lemma 8. *Given N convex functions $f_j : \mathbb{R}^d \to \mathbb{R}$, $j = 1, \ldots, N$, let us define $\Phi : \mathbb{R}^{d\times N} \to \mathbb{R}$ as $\Phi(\xi) = \max_j f_j(x^j)$ for every $\xi = (x^1, \ldots, x^N)$. Then, for every $\xi \in \mathbb{R}^{d\times N}$ we have that*

$$\partial\Phi(\xi) \subset \big\{\big(\gamma^1\partial f_1(x^1), \ldots, \gamma^N\partial f_N(x^N)\big) :$$
$$\gamma\in\Gamma, \gamma^j = 0 \textit{ if } f_j(x^j) < \Phi(x)\big\},$$

where $\Gamma := \{\gamma = (\gamma^1, \ldots, \gamma^N) \in \mathbb{R}^N : \gamma^j \geq 0, \sum_j \gamma^j = 1\}$.

Proof. See [25, Corollary 4.3.2 (Part D)]. □

We also recall the definition of Boltyanski approximating cones from [24, Definition 2.1].

Definition 9. Consider a subset $\mathcal{K} \subseteq \mathbb{R}^n$ and a point $y \in \mathcal{K}$, and let $\mathbf{K}$ be a convex cone in $\mathbb{R}^n$. We say that $\mathbf{K}$ is a *Boltyanski approximating cone to $\mathcal{K}$ at y* if $\mathbf{K} = LC$ where,

(i) for an integer $m > 0$, $C \subseteq \mathbb{R}^m$ is a convex cone,
(ii) $L : \mathbb{R}^m \to \mathbb{R}^n$ is a linear mapping,
(iii) there exists $\delta > 0$ and a continuous map $\mathsf{F} : C \cap B_\delta(0) \to \mathcal{K}$ such that

$$\mathsf{F}(c) = y + Lc + o(|c|), \quad \text{as } c \to 0. \tag{9}$$

In the next proposition we provide the construction of a Boltyanski approximating cone to the epigraph of a non-smooth convex function. This represents a slight generalization w.r.t. the framework of [24], where the cost function is assumed to be differentiable. Indeed, in that case, it is sufficient to consider the *graph* of the cost, and to take the tangent hyperplane as a Boltyanski approximating cone.

Proposition 10. *Let $\mathcal{R} \subseteq \mathbb{R}^n$ be a set and, for $y \in \mathcal{R}$, let $\mathbf{R} \subseteq \mathbb{R}^n$ be a Boltyanski approximating cone at y to $\mathcal{R}$. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function, and let $\mathrm{epi}(f_{|\mathcal{R}}) := \{(x,\eta) \in \mathbb{R}^{n+1} : x \in \mathcal{R}, \eta \geq f(x)\}$. Then, a Boltyanski approximating cone at $(y, f(y))$ to $\mathrm{epi}(f_{|\mathcal{R}})$ is*

$$\tilde{\mathbf{R}} := \{(v, \sigma_{\partial f(y)}(v) + \eta) : v \in \mathbf{R}, \eta \geq 0\}.$$

Proof. Since $\mathbf{R}$ is an approximating cone at y to $\mathcal{R}$, there exists a convex cone $C \subseteq \mathbb{R}^m$, a linear mapping $L_\mathbf{R} : \mathbb{R}^m \to \mathbb{R}^n$, and a continuous function $\mathsf{F}_\mathbf{R} : C \to \mathcal{R}$ such that

$$\mathsf{F}_\mathbf{R}(c) = y + L_\mathbf{R}c + o_1(|c|) \text{ as } c \to 0\text{ , and } \mathbf{R} = L_\mathbf{R}C.$$

Thus, we set $\tilde{C} := \{(c, \sigma_{\partial f(y)}(L_{\mathbf{R}}c) + \eta) : c \in C, \eta \geq 0\}$. We observe that $\tilde{C}$ is a convex cone, since it is the epigraph of the support function $c \mapsto \sigma_{\partial f(y)}(L_{\mathbf{R}}c)$ restricted to C. Moreover, let us set $\tilde{L} : \mathbb{R}^{m+1} \to \mathbb{R}^{n+1}$ such that $\tilde{L}(c,\mu) := (L_{\mathbf{R}}c, \mu)$ where $(c,\mu) \in \tilde{C}$, and let us define the continuous function $\tilde{\mathsf{F}} : \tilde{C} \to \operatorname{epi}(f|_{\mathcal{R}})$ as:

$$\tilde{\mathsf{F}}(c,\mu) := \big(\mathsf{F}_{\mathbf{R}}(c), \mu - \sigma_{\partial f(y)}(L_{\mathbf{R}}c) + f(\mathsf{F}_{\mathbf{R}}(c))\big).$$

We observe that

$$\begin{aligned} f(\mathsf{F}_{\mathbf{R}}(c)) &= f(y + L_{\mathbf{R}}c + o_1(|c|)) \\ &= f(y) + \sigma_{\partial f(y)}(L_{\mathbf{R}}c) + o_2(|c|), \end{aligned}$$

where we have used Lemma 7 and the boundedness of $\partial f(y)$. Hence, for every $(c,\mu) \in \tilde{C}$, we have

$$\begin{aligned} \tilde{\mathsf{F}}(c,\mu) &= (y, f(y)) + (L_{\mathbf{R}}c, \mu) + (o_1(|c|), o_2(|c|)) \\ &= (y, f(y)) + \tilde{L}(c,\mu) + o(|c|), \end{aligned}$$

and we set $\tilde{\mathbf{R}} := \tilde{L}\tilde{C}$. Recalling that, for every $(c,\mu) \in \tilde{C}$, $\tilde{L}(c,\mu) = \tilde{L}(c, \sigma_{\partial f(y)}(L_{\mathbf{R}}c) + \eta) = (L_{\mathbf{R}}c, \sigma_{\partial f(y)}(L_{\mathbf{R}}c) + \eta)$, we deduce the thesis. □

Finally, we report here an extension of the Abstract Maximum Principle presented in [24, Section 5], aimed at encompassing the case of a non-smooth convex cost.

Theorem 11. *Let us consider a metric space $\mathcal{U}$, a continuous map $y : \mathcal{U} \to \mathbb{R}^n$, a convex function $\Psi : \mathbb{R}^n \to \mathbb{R}$, and a set $\mathcal{S} \subset \mathbb{R}^n$. Let $(\bar{u}, y(\bar{u})) \in \mathcal{U} \times \mathbb{R}^n$ be a* strong local minimizer[1] *for*

$$\min_{u \in \mathcal{U}} \Psi(y(u)) \quad \textit{subject to } y(u) \in \mathcal{S}.$$

Setting $\bar{y} := y(\bar{u})$, we define $\mathcal{R} := \{y(u) : u \in \mathcal{U}\}$, and let $\mathbf{R}, \mathbf{K}$ be Boltyanski approximating cones at $\bar{y}$ to $\mathcal{R}$ and $\mathcal{S}$, respectively. Then, there exists $(\lambda, \lambda_c) \in (\mathbb{R}^{n+1})^\star$ such that

$$\begin{aligned} &(a) \quad (\lambda, \lambda_c) \neq (0,0), \\ &(b) \quad \lambda \in -\mathbf{K}^\perp \quad \textit{and } \lambda_c \leq 0, \\ &(c) \quad \max_{v \in \mathbf{R}} \left(\min_{p \in \partial\Psi(\bar{y})} \langle \lambda + \lambda_c p, v \rangle \right) = 0. \end{aligned}$$

Proof. Following the proof of [24, Theorem 5.1], let us introduce the sets $\tilde{\mathcal{S}} := \{(x,\eta) \in \mathbb{R}^{n+1} : x \in \mathcal{S}, \eta < \Psi(\bar{y})\} \cup \{(\bar{y}, \Psi(\bar{y}))\}$ and $\tilde{\mathcal{R}} := \operatorname{epi}(\Psi_{|\mathcal{R}})$, and their respective Boltyanski approximating cones at $(\bar{y}, \Psi(\bar{y}))$, i.e., $\tilde{\mathbf{K}} = \{(w,\nu) : w \in \mathbf{K}, \nu \geq 0\}$ and, owing to Proposition 10, $\tilde{\mathbf{R}} := \{(v, \sigma_{\partial\Psi(\bar{y})}(v) + \eta) : v \in \mathbf{R}, \eta \geq 0\}$. In virtue of [24, Corollary 4.1], since $\tilde{\mathcal{R}}$ and $\tilde{\mathcal{S}}$ are locally separate (see [24, Lemma 5.1]), it follows that $\tilde{\mathbf{R}}$ and $\tilde{\mathbf{K}}$ are linearly separable, i.e., there exists $(\lambda, \lambda_c) \in (\mathbb{R}^{n+1})^\star$ s.t. $(\lambda, \lambda_c) \neq (0,0)$,

$$\begin{aligned} \langle (\lambda, \lambda_c), \tilde{w} \rangle \geq 0 \qquad & \forall \tilde{w} \in \tilde{\mathbf{K}}, \\ \langle (\lambda, \lambda_c), \tilde{v} \rangle \leq 0 \qquad & \forall \tilde{v} \in \tilde{\mathbf{R}}. \end{aligned}$$

From the first inequality, we deduce point (b) of the thesis, while the second inequality yields

$$\langle \lambda, v \rangle + \lambda_c(\sigma_{\partial\Psi(\bar{y})}(v) + \eta) \leq 0$$

for every $v \in \mathbf{R}$ and $\eta \geq 0$, which implies for $\eta = 0$ that $\langle \lambda, v \rangle + \lambda_c \sigma_{\partial\Psi(\bar{y})}(v) \leq 0$ for every $v \in \mathbf{R}$. Since the equality is attained for $v = 0$, we deduce that

$$\max_{v \in \mathbf{R}} \big(\langle \lambda, v \rangle + \lambda_c \sigma_{\partial\Psi(\bar{y})}(v)\big) = 0.$$

Finally, recalling that $\lambda_c \leq 0$, the thesis follows from (8). □

B. Proof of Theorem 3

We finally prove the necessary optimality conditions associated to (3) using Theorem 11. In our framework, $\mathcal{U}$ is the space of admissible controls, $\mathbb{R}^n = (\mathbb{R}^{d \times N})^M$, and $y : \mathcal{U} \to \mathbb{R}^n$ maps every control $u \in \mathcal{U}$ to the final-time state of the corresponding solution of (4). Since we do not deal with final-time constraints, we have that $\mathcal{S} = (\mathbb{R}^{d \times N})^M$. Moreover, we set $\mathcal{R} := y(\mathcal{U})$. As a direct application of [24, Corollary 6.1], we obtain the approximating cones to $\mathcal{R}$.

Proposition 12. *Let $(\bar{u}, \bar{X})$ be a strong local minimizer for (3), and for $r \geq 1$ let $\{t_1, \ldots, t_r\} \subseteq [0,T]$ be arbitrary distinct Lebesgue points for $t \mapsto \mathcal{F}(\bar{X}(t), \bar{u}(t))$, and let us take $\omega_1, \ldots, \omega_r \in U$. Then, if we set $W_k := \mathcal{F}(\bar{X}(t_k), \omega_k) - \mathcal{F}(\bar{X}(t_k), \bar{u}(t_k))$,*

$$\mathbf{R} := \operatorname{span}^+_{k=1,\ldots,r} \{\mathcal{M}(t_k, T) W_k\} \tag{10}$$

is a Boltyanski approximating cone to $\mathcal{R}$ at $\bar{X}(T)$, where $\mathcal{M}(\cdot,\cdot)$ is the fundamental matrix[2] associated to the linearized equation $\dot{V}(t) = \nabla_X \mathcal{F}\left(\bar{X}(t), \bar{u}(t)\right) V(t)$.

We observe that the Boltyanski cone (10) reflects the particular structure of the dynamics (4). Indeed, we can write $\mathbf{R} = \prod_{i=1}^M \left(\prod_{j=1}^N \operatorname{span}^+_{k=1,\ldots,r} \left\{ M^{i,j}(t_k, T) w_k^{i,j} \right\} \right)$, where $w_k^{i,j} := F(\bar{x}_i^j(t_k), \omega_k) - F(\bar{x}_i^j(t_k), \bar{u}(t_k))$, and $M^{i,j}(\cdot,\cdot)$ is the fundamental matrix associated to the linearization of the trajectory $\bar{x}_i^j$, i.e., $\dot{v}(t) = \nabla_x F(\bar{x}_i^j(t), \bar{u}(t)) v(t)$. We are now in position to present the proof of Theorem 3.

Proof of Theorem 3. In virtue of Theorem 11, since $\mathcal{S} = (\mathbb{R}^{d \times N})^M$, we have that $\mathbf{K} = (\mathbb{R}^{d \times N})^M$ and $\mathbf{K}^\perp = \{0\}$. Moreover, using the same notations as in Section II, we have that $\Psi(X) = \frac{1}{M}\sum_{i=1}^M \tilde{g}_i(X_i)$. Hence, by setting $\lambda_c = -1$, we consider the approximating cone $\mathbf{R}$ to $\mathcal{R}$ at $\bar{X}(T)$ constructed in Proposition 12 with Lebesgue points $\{t_1, \ldots, t_r\}$, and from Theorem 11 it follows that

$$0 = \max_{V \in \mathbf{R} \cap \overline{B_1(0)}} \; \min_{P \in -\partial\Psi(\bar{X}(T))} \langle P, V \rangle,$$

since the maximum is attained at $0 \in \mathbf{R} \cap \overline{B_1(0)}$. Moreover, by using von Neumann's Minimax Theorem (see, e.g., [23, Theorem 3.4.6]) we obtain that

$$\min_{P \in -\partial\Psi(\bar{X}(T))} \; \max_{V \in \mathbf{R} \cap \overline{B_1(0)}} \langle P, V \rangle = 0,$$

i.e., there exists $P^* \in -\partial\Psi(\bar{X}(T))$ such that $\langle P^*, V \rangle \leq 0$ for every $V \in \mathbf{R} \cap \overline{B_1(0)}$. Since the last inequality is invariant by positive rescaling, we deduce that

$$\langle P^*, \mathcal{M}(t_k, T)\left(\mathcal{F}(\bar{X}(t_k), \omega_k) - \mathcal{F}(\bar{X}(t_k), \bar{u}(t_k))\right)\rangle \leq 0.$$

[1]This concept is the natural generalization of Definition 1. See also [24, Definition 5.1 and Remark 5.2].

[2]See, e.g., [26, Section 2.2].

From the structure of the cost function Ψ and Lemma 8, it follows that the covector $P^{*}= (P_1^*, \dots, P_M^*)$ has the form $P_i^* = -\frac{1}{M}\left(\gamma_i^1 \nabla_x g_i\big(\bar{x}_i^1(T)\big), \dots, \gamma_i^N \nabla_x g_i\big(\bar{x}_i^N(T)\big)\right)$. Then, the thesis follows from a classical infinite intersection argument, as detailed e.g. in the proof of [27, Theorem 5.7.1]. □

IV. NUMERICS

In this section, we present numerical experiments designed to validate our interpretation of the PMP in the context of robust neural network training.
Here, we approach the robust training of deep neural networks from a control perspective. Specifically, we interpret neural networks as discretized representations of controlled ODEs such as the one depicted in (2). Then, the network parameter training is viewed as a control problem. Our objective is to minimize the cost function (3), which is defined in order to obtain a network which robustly classifies the input data. In particular, we employ the PMP outlined in Theorem 3 and derived in Section III, to identify the controls/network parameters that yield effective and robust classification performance. The utilization of the PMP for neural network training has already been successfully applied and explored in [11], [22]. In practice, this approach typically involves employing a shooting method, such as the one outlined in Algorithm 1, which consists of repeating the forward evolution of the trajectories, the backward evolution of the adjoint variables, and the update of the controls, until convergence. For further insights into this methodology, we direct readers to [28]. It is crucial to emphasize that the novelty of our approach lies in the selection of the weights to be incorporated into (7), rather than in the use of the PMP for training neural networks. To keep the analysis simple, we focus on a classification task in 2d, but we defer the extension to higher-dimensional experiments to future research. The primary objective here is to robustly train a network to distinguish between data points separated by a nonlinear boundary.

Algorithm 1: Shooting method

```
Data:
$u^0$ : initial guess for controls; iter_max : number of shooting iterations;
$\Delta t$ : time-discretization of the interval $[0,T]$; $\tau$ : memory parameter;
Result: $u^{\text{iter\_max}}$
time_nodes ← $T/\Delta t$;
for $k = 0, \dots,$ iter_max do
    for $i = 1, \dots, M,\ j = 1 \dots, N$ do
        for $n = 1, \dots,$ time_nodes do
            $x_i^j(t_{n+1}) \leftarrow x_i^j(t_n) + \Delta t\, F(x_i^j(t_n), u^k(t_n))$;
        end
    end
    for $i = 1, \dots, M,\ j = 1, \dots, N$ do
        $p_i^j(t_{\text{time\_nodes}}) \leftarrow -\frac{\gamma_i^j}{M} \nabla_x g_i\left(x_i^j(t_{\text{time\_nodes}})\right)$;
        for $n =$ time_nodes, ..., 1 do
            $p_i^j(t_{n-1}) \leftarrow$
              $p_i^j(t_n) + \Delta t\, p_i^j(t_n) \cdot \nabla_x F\big(x_i^j(t_n), u^k(t_n)\big)$;
        end
    end
    for $n =$ time_nodes, ..., 1 do
        $X(t_{n-1}) \leftarrow (x_i^j)_{i=1,\dots,M}^{j=1,\dots,N}$ and $P(t_{n-1}) \leftarrow (p_i^j)_{i=1,\dots,M}^{j=1,\dots,N}$;
        $u^{k+1}(t_{n-1}) \leftarrow \max_{\omega\in U} [H(X(t_{n-1}), P(t_{n-1}), \omega)$
                                   $- \frac{1}{2\tau}\|\omega - u^k(t_{n-1})\|_2^2]$;
    end
    If needed, update $(\gamma_i^j)_{i=1,\dots,M}^{j=1,\dots,N}$;
end
```

A. *The Classification task*

The training data: We generate two classes by uniformly sampling $M = 200$ data points in the domain $[0,1]^2$ excluding a separation region (shaded yellow in Fig. 1). These data points are labeled according to their positions ($y = 1 =$ 'above' or $y = -1 =$ 'below') w.r.t. a predetermined nonlinear boundary (depicted as a yellow line in Fig. 1). The variable y introduced in (1) denotes the label associated with each data point. Since this variable is incorporated in the cost g_i for every datum i, it does not explicitly appear in Algorithm 1.

The network: We employ a residual neural network of 20 layers, which corresponds to a discretization of the interval $[0, T]$ with $T = 1$ and $\Delta t = 0.05$. The network is interpreted as an explicit Euler discretization of the $2d$-controlled ODE

$$\begin{cases} \dot{x}(t) = F(x(t), u(t)) = \sigma\big(W(t)x(t) + b(t)\big), \\ x(0) = \xi^0. \end{cases} \tag{11}$$

Here, σ denotes an activation function acting component-wise, set to be the hyperbolic tangent in our experiments. The state-space of this system is $\mathbb{R}^2$, while the control variable is $u(t) = (W(t), b(t)) \in \mathbb{R}^{2\times 2} \times \mathbb{R}^2$. Here, given an admissible control $t \mapsto (W(t), b(t))$, evaluating the neural ODE written above on our dataset means solving (11) with $\xi^0 = x_i^0 + \alpha_i^j$, with $i = 1, \dots, M$ and $j = 1, \dots, N$.

The adversarial budget: We fix $N = 4$ equidistant perturbations with adversarial budget $\epsilon = 0.02$ around each training point. This budget is chosen sensibly with the class separation region (yellow in Fig. 1) in mind: The perturbations may populate the region but an unwanted overlap of opposite labels is prevented. We remark that in high-dimensional implementations, such as image classification, the attack's budget is typically assumed to be very small compared to the class distance, so that the labels are unaffected.

The cost function: As loss functions g_i, we use a cost that promotes separation and clustering of the two classes. In ML, it is typical to append a trainable linear layer at the end of the network to project the classifier's output, and then to employ a penalization on the mismatch. However, in order to maintain an optimal control formulation and for more accurate comparison of the weight choices below, we prefer to incorporate a fixed projection within the loss function. Consequently, we use the following cost:

$$g_i(x_i^j(T)) = \begin{cases} e^{v\cdot x_i^j(T)} & y_i = 1, \\ e^{-v\cdot x_i^j(T)} & y_i = -1, \end{cases}$$

where $v = (1, -1)$. It is important to note that while our method is applicable to any type of cost function, including quadratic or more complex formulations, we have chosen the above expression for visualization purposes.

The training: In Algorithm 1, we provide the shooting method used for the numerical solution of the PMP, and hence for training the network. The connection between the shooting method and the training of neural networks with gradient descent has been investigated in [11], [29]. Notice that the extra term in the Hamiltonian (at line 17) comes from [28] for stability reasons. We set the initial guess u^0 by randomly generating standard Gaussian values for each entry and for each time-node, while the number of shooting iterations is $\text{iter\_max} = 1000$. It is worth noting that the maximization step in line 16 of Algorithm 1 can be performed in various ways, such as gradient ascent. However, we opt for the approach used in [13], which employs a fixed-point method to find the point where the gradient of the augmented Hamiltonian vanishes.

B. The choice of the weights

One crucial challenge is that the PMP derived in Theorem 3 does not specify how to choose the weights γ_i^j at line 19 of Algorithm 1. Therefore, we compare three options.

Uniform robust: The first approach involves assigning uniform weights to all perturbations. In this method, we consider the perturbations as training data points and minimize the loss based on them. However, this straightforward approach is not connected to a minimax problem and does not align with the theoretical framework, where the weights play a specific role in emphasizing the perturbations that achieve the maximum.

Weighted robust: The second option is to introduce the following weight functions for the perturbations:

$$\gamma_i^j = \frac{1}{C_i} e^{c g_i(x_i^j(T))} \quad \text{with } C_i = \sum_{j=1}^{N} e^{c g_i(x_i^j(T))}, \qquad (12)$$

where $c > 0$ is a constant chosen a priori. This choice is inspired by Laplace's principle [30] and it is reminiscent of Gibbs measures. These weights must be updated each time the control is adjusted, and they are determined based on the cost g_i of the network's output $x_i^j(T)$, which we aim to maximize across perturbations. Subsequently, these weights are normalized according to the cost achieved by the other perturbations. The benefit of this approach is that if two perturbations (almost) achieve the maximum, they will have (almost) equal weights, leading to a more stable method. This peculiarity stands in contrast to the instability of the worst-case approach described below. The constant c needs to be chosen appropriately. Setting it to zero (or too small) is equivalent to the uniform weighting method described above. Similar considerations have already been done in [31], motivated by the relaxation of the max function by the LogSumExp.

Worst-case robust: The third approach consists of explicitly computing the worst-case scenario at each step — which involves identifying for every datum the perturbation that achieves the maximum — and in updating the controls using exclusively these data-points. During the execution of the shooting method, the worst-case perturbations are typically different at each iteration. As a matter of fact, the cost is highly oscillating along the iterative steps, since improving the performance on the worst-cases might result in deteriorating the behavior of the others. At each iteration of the shooting method, this approach is equivalent to set the coefficients $(\gamma_i^j)_{i=1,\dots,M}^{j=1,\dots,N}$ according to (12), and then to send $c \to \infty$.

C. Results

In Fig. 1 we display the classification prediction probabilities of the trained models. We compare all the four methods described above, namely the non-robust approach using the unperturbed set of training data, the uniform robust method, the weighted robust approach with weights as (12) with $c = 100$ and finally the worst-case robust method. For the robust methods, as training input, we use the perturbed version of the data employed in the non-robust method, i.e. $M \times N = 800$ training points, with unchanged parameter initialization. Upon initial visual analysis, it is apparent that while the robust methods may not significantly outperform the non-robust method in precisely identifying the decision boundary, they excel in recognizing areas with higher risk of misclassification and making more cautious predictions. To provide a more quantitative comparison of all methods, we present in Table I several metrics obtained by averaging over multiple experiments. These metrics are: test accuracy (on $[0,1]^2$), classification margin accuracy of the excluded margin of unperturbed training points, and high-confidence mistakes, which determine the share of all misclassifications in $[0,1]^2$ with prediction confidence of more than 70%. This last metric highlights the algorithm's capability to identify areas of ambiguity. Finally, we provide a comparison of

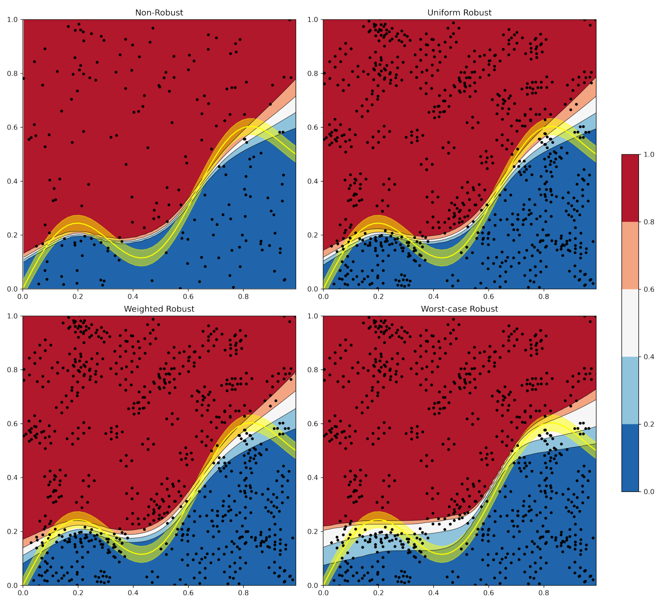

Fig. 1. Classification level-sets on $[0,1]^2$: the color bar indicates the confidence of prediction of one class (red above the yellow margin) or the other class (blue below the yellow margin).

	Test Accuracy	Margin Accuracy	High-confidence Mistakes
Non-robust	94,12 %	56,52 %	78,50 %
Uniform robust	94,26 %	56,80 %	72,57 %
Weighted robust	94,12 %	57,14 %	65,13 %
Worst-case robust	94,68 %	58,31 %	55,76 %

TABLE I

robustness in Fig. 2, where we examine the value of J as defined in (3). To achieve robustness, the objective is to minimize this value, motivated by the minimax formulation (1). The key observation here is that the worst-case robust method achieves the lowest loss, which aligns with the results presented in Table I. However, this method exhibits considerable oscillations due to the instability of the maxima. Indeed, when handling scenarios involving multiple perturbations that are close to the maximum, the worst-case method would select a single perturbation and adjust the parameters accordingly. This approach may favor one direction while neglecting the others. In contrast, the weighted robust method considers the influence of all perturbations. This is confirmed by the plot in Fig. 2: our proposed weighted robust approach generally exhibits slightly reduced robustness compared to the worst-case robust method, but it offers more stability during training.

V. CONCLUSIONS

In this work, we investigate the minimax optimal control problem of neural ODEs in order to improve adversarially robust training of deep neural networks. In the continuous ODE setting, we provide a proof of the first order optimality conditions in form of the PMP. Inspired by these conditions, we present a numerical scheme for training neural networks that incorporates weighted adversarial attacks for each training point. This method achieves promising results in low dimensions: it surpasses the uniform robust method in terms of accuracy and exhibits greater stability in terms of training behavior compared to the worst-case approach. Our novel approach has potential for higher dimensional settings, where the precise computation of the local maximum is infeasible, but a weighted approximation could improve robustness.

ACKNOWLEDGMENT

C.C. is partially supported by the DFG SPP 2298. A.S. is partially supported by INdAM-GNAMPA. T.W. is supported by the Austrian Science Fund (FWF) grant no. J 4681-N. Finally, the authors want to thank Prof. Franco Rampazzo.

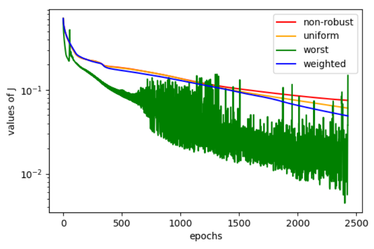

Fig. 2. Robustness measure displayed on a semilogarithmic scale.

REFERENCES

[1] J. Jumper, R. Evans, A. Pritzel, *et al.* Highly accurate protein structure prediction with alphafold. In: Nature, 596:583–589, 2021.
[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In: 2016 IEEE CVPR, 770–778, 2016.
[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.
[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus. Intriguing properties of neural networks. In: 2nd ICLR, 2014.
[5] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. In: 3rd ICLR, 2015.
[6] W. E. A proposal on machine learning via dynamical systems. In: Comm. Math. Stat., 1(5):1–11, 2017.
[7] E. Haber, L. Ruthotto. Stable architectures for deep neural networks. In: Inverse problems, 34(1), 2018.
[8] R.T.Q. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud. Neural Ordinary Differential Equations. In: Advances in Neural Information Processing Systems, 2018.
[9] E. Dupont, A. Doucet, Y. W. Teh. Augmented neural odes. Advances in Neural Information Processing Systems, 2019.
[10] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, E. F. Mishchenko. The mathematical theory of optimal processes. In: Interscience Publishers John Wiley & Sons Inc., 1962.
[11] Q. Li, L. Chen, C. Tai, W. E. Maximum principle based algorithms for deep learning. In: J. Mach. Learn. Res., 18(1):5998—6026, 2017.
[12] C. G. Trillos, N. G. Trillos. On the regularized risk of distributionally robust learning over deep neural networks. In: Res. Math. Sci., 9(54), 2022.
[13] C. Cipriani, M. Fornasier, A. Scagliotti. From NeurODEs to AutoencODEs: a mean-field control framework for width-varying neural networks. In: Eur. J. Appl. Math., 2024.
[14] C. Esteve, B. Geshkovski, D. Pighin, E. Zuazua. Large-time asymptotics in deep learning. In: arXiv:2008.02491
[15] D. Ruiz-Balet, E. Zuazua. Neural ode control for classification, approximation, and transport. In: SIAM Review, 65(3):735–773, 2023.
[16] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu. Towards deep learning models resistant to adversarial attacks. In: 6th ICLR, 2018.
[17] U. Shaham, Y. Yamada, S. Negahban. Understanding adversarial training: Increasing local stability of supervised models through robust optimization. In: Neurocomputing, 307:195–204, 2018.
[18] A. Wald. Statistical decision functions which minimize the maximum risk. In: Annals of Mathematics, 46(2):265–280, 1945.
[19] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, P. McDaniel. The space of transferable adversarial examples. In: arXiv:1704.03453, 2017.
[20] N. S. Frank, J. Niles-Weed. Existence and minimax theorems for adversarial surrogate risks in binary classification. In: arXiv:2206.09098, 2022.
[21] R.B. Vinter. Minimax optimal control. In: SIAM J. Control Optim., 4(3):939-968, 2005.
[22] B. Bonnet, C. Cipriani, M. Fornasier, H. Huang. A measure theoretical approach to the mean-field maximum principle for training NeurODEs. In: Nonlinear Analysis, 227:113-161, 2023.
[23] R.B. Vinter. Optimal control. In: Birkhäuser, 1(1), 2000.
[24] M. Motta, F. Rampazzo. An Abstract Maximum Principle for constrained minimum problems. In: arXiv:2310.09845, 2023.
[25] J.-B. Hiriart-Urruty, C. Lemaréchal. Fundamentals of Convex Analysis. In: Springer Science & Business Media, 2004.
[26] A. Bressan, B. Piccoli. Introduction to the mathematical theory of control. In: AIMS, Springfield, 2004.
[27] H.J. Sussmann. Geometry and Optimal Control. In: Baillieul, J., Willems, J.C. (eds) Mathematical Control Theory. Springer, 1999.
[28] Y. Sakawa, Y. Shindo. On global convergence of an algorithm for optimal control. In: IEEE Trans. Automat. Contr. 25(6):1149-1153, 1980.
[29] M. Benning, E. Celledoni, M. Ehrhardt, B. Owren, C. B. Schhönlieb. Deep learning as optimal control problems: models and numerical methods. In: Journal of Computational Dynamics, 2019.
[30] A. Dembo and O. Zeitouni. Large deviations techniques and applications. In: Springer Science & Business Media, 38, 2009.
[31] T. Li, A. Beirami, M. Sanjabi, V. Smith. Tilted Empirical Risk Minimization. In: 8th ICLR, 2020.