A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning

Sijia Liu, Member, IEEE, Pin-Yu Chen, Member, IEEE, Bhavya Kailkhura, Member, IEEE, Gaoyuan Zhang, Alfred Hero, Fellow, IEEE, and Pramod K. Varshney, Life Fellow, IEEE

Abstract—Zeroth-order (ZO) optimization is a subset of gradient-free optimization that emerges in many signal processing and machine learning applications. It is used for solving optimization problems similarly to gradient-based methods. However, it does not require the gradient, using only function evaluations. Specifically, ZO optimization iteratively performs three major steps: gradient estimation, descent direction computation, and solution update. In this paper, we provide a comprehensive review of ZO optimization, with an emphasis on showing the underlying intuition, optimization principles and recent advances in convergence analysis. Moreover, we demonstrate promising applications of ZO optimization, such as evaluating robustness and generating explanations from black-box deep learning models, and efficient online sensor management.

Index Terms—Zeroth-order (ZO) optimization, nonconvex optimization, gradient estimation, black-box adversarial attacks, machine learning, deep learning

I. INTRODUCTION

Many signal processing, machine learning (ML) and deep learning (DL) applications involve tackling complex optimization problems that are difficult to solve analytically. Often the objective function itself may not be in analytical closed form, only permitting function evaluations but not gradient evaluations. Optimization corresponding to these types of problems falls into the category of zeroth-order (ZO) optimization with respect to black-box models, where explicit expressions of the gradients are difficult to compute or infeasible to obtain. ZO optimization methods are gradient-free counterparts of first-order (FO) optimization methods. They approximate the full gradients or stochastic gradients through function value based gradient estimates. Interest in ZO optimization has grown rapidly in the past few years since the concept of gradient estimation by finite difference approximations was proposed in the 1950s and 1980s [1], [2].

It is worth noting that derivative-free methods for black-box optimization had been studied by the optimization community long before they had impact on signal processing and ML/DL. Traditional derivative-free optimization (DFO) methods can be classified into two categories: direct search-based methods (DSMs) and model-based methods (MBMs) [3]–[6]. DSMs include the Nelder-Mead simplex method [7], the coordinate search method [8], and the pattern search method [9], to name a few. MBMs contain model-based descent methods [10] and trust region methods [11]. Evolutionary optimization is another class of generic population-based meta heuristic DFO algorithms, and includes particle swarm optimization methods [12] and genetic algorithms [13]. Some Bayesian optimization (BO) methods [14] tackle black-box optimization problems by modelling the objective function as a Gaussian process (GP) that is learned.
from the history of function evaluations. However, learning an accurate GP model is computationally intensive.

Conventional DFO methods have two main shortcomings. First, they are often difficult to scale to large-size problems. For example, the off-the-shelf DFO solver COBYLA [15] only supports problems with a maximum of $2^{16}$ variables (SciPy Python library [16]), which is smaller than the size of a single ImageNet image [17]. Second, DFO methods lack a convergence rate analysis and they may require a significant amount of effort to be customized to the particular applications. ZO optimization has three main advantages over DFO: a) ease of implementation with only small modification of commonly-used gradient-based algorithms, b) computationally efficient approximations to derivatives when they are difficult to compute, and c) comparable convergence rates to FO algorithms [18]–[21]. An illustrative example of ZO optimization versus FO optimization is shown in Figure 1.

ZO optimization has attracted increasing attention due to its success in solving emerging signal processing and ML/DL problems. First, ZO optimization serves as a powerful and practical tool for evaluating adversarial robustness of ML/DL systems [22]. We note that the research in adversarial robustness is receiving increased attention in recent years. ZO based methods for exploring vulnerability of DL to black-box adversarial attacks are able to reveal the most susceptible features. Such ZO methods can be as effective as state-of-the-art white-box attacks, despite only having access to the inputs and outputs of the targeted deep neural networks (DNNs) [23], [24]. Moreover, ZO optimization can generate explanations and provide interpretations of prediction results in a gradient-free and model-agnostic manner [25]. Furthermore, ZO optimization can also be used to solve automated ML problems, e.g., automated backpropagation in DL, where the gradients with respect to ML pipeline configuration parameters are intractable [26]. ZO optimization is also applicable to ML applications where the full gradient must be kept private [27]. In addition, ZO optimization provides computationally-efficient alternatives for second-order optimization such as robust training by curvature regularization [28], meta-learning [29], transfer learning [30], and online network management [27].

In this paper, we provide a comprehensive review of recent development in ZO optimization for signal processing and ML. In Sections II and III, we review various types of ZO gradient estimators as well as ZO algorithms. Section IV presents a promising connection between ZO optimization and adversarial ML. Section V illustrates an application of ZO optimization to online sensor management. More applications are provided in Section VI. We discuss open issues and state our conclusions in Sections VII and VIII respectively.

II. GRADIENT ESTIMATION VIA ZO ORACLE

In this section, we provide an overview of gradient estimation techniques for optimization with a black-box objective function. The resulting gradient estimate forms the basis for constructing the descent direction used in ZO optimization algorithms. We categorize the ZO gradient estimates into two types, 1-point, and multi-point estimates, based on the number of queried function evaluations. As the number of function evaluations increases, a more accurate gradient estimate is expected but at the cost of increased query complexity.

A. 1-point estimate

We start by the principles of randomized gradient estimation in the context of 1-point estimation. Let $f(x)$ be a continuously differentiable objective function on a $d$-dimension variable $x \in \mathbb{R}^d$. The 1-point gradient estimate of $f$ has the generic form

$$\hat{\nabla} f(x) := \frac{\phi(d)}{\mu} f(x + \mu u),$$

where $u \sim p$ is a random direction vector drawn from a certain distribution $p$, which is typically chosen as either the standard multivariate normal distribution $\mathcal{N}(0, I)$ [19] or the multivariate uniform distribution $\mathcal{U}(S(0, 1))$ on a unit sphere centered at 0 with radius 1 [20]. $\mu > 0$ is a perturbation radius (also called a smoothing parameter), and $\phi(d)$ denotes a certain dimension-dependent factor related to the choice of the distribution $p$. If $p = \mathcal{N}(0, I)$, then $\phi(d) = 1$; If $p = \mathcal{U}(S(0, 1))$, then $\phi(d) = d$.

The rationale behind (1) is that it is an unbiased estimate of the gradient of the smoothed version of $f$ over a random perturbation $u \sim p'$ with smoothing parameter $\mu$,

$$f_\mu(x) := \mathbb{E}_{u \sim p'}[f(x + \mu u)],$$

where $p'$ is specified as $\mathcal{N}(0, I)$ if $p = \mathcal{N}(0, I)$ in (1), or the multivariate uniform distribution on a unit ball $\mathcal{U}(B(0, 1))$ if
which satisfies the unbiasedness condition (3) for any $f$. Thus, variance cannot be reduced even if $\phi$ becomes smaller. However, in a practical system, if $\mu$ is too small, then the function difference could be dominated by system noise and it may fail to represent the differential. Thus, careful selection of the smoothing parameter $\mu$ is important for convergence of ZO optimization methods. Second, different from the first-order stochastic gradient estimate, the ZO gradient estimate yields a dimension-dependent variance that increases as $O(d\|\nabla f(x)\|^2)$. Thus, variance cannot be reduced even if $\mu \to 0$. Thus, some recent work has focused on the design of variance-reduced gradient estimates.

Mini-batch sampling is the most commonly-used approach to reduce the variance of ZO gradient estimates. Instead of using a single random direction, the average of $b$ i.i.d. samples $\{u_i\}_{i=1}^b$ drawn from $p$ are used for gradient estimation, leading to the multi-point estimate

$$
\hat{\nabla} f(x) := \frac{\phi(d)}{\mu} \sum_{i=1}^b [(f(x + \mu u_i) - f(x))u_i],
$$

with the approximation error $O\left(\frac{\mu^2 d^3}{\phi(d)b}\right)$.

In (6), the first two terms correspond to the reduced variance of the 2-point estimate $\mathbb{E}[\|\hat{\nabla} f(x) - \nabla f_\mu(x)\|^2]$ due to the drawing of $b$ random direction vectors. And the third term, independent of $b$, corresponds to the approximation error due to the gradient of the smoothed function $\|\nabla f(x) - \nabla f_\mu(x)\|^2$.

When the number of function evaluations reaches the problem dimension $d$ in (6), then instead of using randomized directions $\{u_i\}_{i=1}^d$, one can employ the deterministic coordinate-wise gradient estimate

$$
\frac{\partial}{\partial x_i} \sum_{i=1}^d [(f(x + \mu e_i) - f(x))e_i],
$$

which yields a lower approximation error, of order $O(d\mu^2)$.

Here $e_i \in \mathbb{R}^d$ denotes the $i$th elementary basis vector, with 1 at the $i$th coordinate and 0s elsewhere. In practice, the multi-point gradient estimate (6) is usually implemented with $2 \leq b \leq d$. The previously introduced multi-point estimates are computed by using forward differences of function values. An alternative is the central difference variant that uses

$$
(f(x + \mu u) - f(x - \mu u)),
$$

where $u$ can be either randomized or deterministic. These central difference estimates have similar approximation errors to the forward difference estimates.

### III. ZO Optimization Algorithms

In this section, we present a unified algorithmic framework covering many commonly-used ZO optimization methods. We provide a thorough overview of existing algorithms in different problem settings and delve into the factors that influence their convergence.

#### A. The generic form of the ZO algorithm

Let us consider a stochastic optimization problem

$$
\min_{x \in X} f(x) := \mathbb{E}_\xi f(x; \xi),
$$

where $x \in \mathbb{R}^d$ are optimization variables, $X$ is a closed convex set, $f$ is a possibly nonconvex objective function, and $\xi$ is a
certain random variable that captures stochastic data samples or noise. If \( \xi \) obeys a uniform distribution over \( n \) empirical samples \( \{\xi_i\}_{i=1}^n \), then problem (8) reduces to a finite-sum formulation with objective function \( f(x) = \frac{1}{n} \sum_{i=1}^n f(x; \xi_i) \). And if \( \mathcal{X} = \mathbb{R}^d \), then problem (8) simplifies to the unconstrained optimization problem.

Algorithm 1 Generic form of ZO optimization

1. **Gradient estimation:**
   \[ \hat{g}_t = \phi(\{f(x_t; \xi_j)\}_{j=1}^{\Omega_t}), \]
   where \( \Omega_t \) denotes a set of mini-batch stochastic samples used at iteration \( t \).

2. **Descent direction computation:**
   \[ m_t = \psi(\{g_j\}_{j=1}^t), \]
   where \( \psi(\cdot) \) denotes a point updating operation subject to the constraint \( x \in \mathcal{X} \).

3. **Point updating:**
   \[ x_{t+1} = \Pi_{\mathcal{X}}(x_t - \eta_t m_t), \]
   where \( \Pi_{\mathcal{X}} \) denotes a point updating operation subject to the constraint \( x \in \mathcal{X} \).

Most ZO optimization methods mimic their first-order counterparts, and involve three steps, shown in Algorithm 1, gradient estimation (9), descent direction computation (10), and point updating (11). Without loss of generality, we specify (9) as a variant of (6) built on a mini-batch of empirical samples \( \{\xi_j\}_{j=1}^{\Omega_t} \),

\[ \hat{g}_t = \phi(\{f(x_t; \xi_j)\}_{j=1}^{\Omega_t}), \quad \alpha = \frac{1}{|\Omega_t|} \sum_{j \in \Omega_t} \nabla f(x_t; \xi_j), \]

where \( \nabla f(x_t; \xi_j) \) is given by (6) as the gradient of the function \( f(\cdot; \xi) \), and \( |\Omega_t| \) denotes the cardinality of the set of mini-batch samples at iteration \( t \).

Next, we elaborate on the descent direction computation and the point updating step used in many ZO algorithms.

1) **ZO algorithms for unconstrained optimization:** We consider the ZO stochastic gradient descent (ZO-SGD) method [18], the ZO sign-based SGD (ZO-signSGD) [36], the ZO stochastic variance reduced gradient (ZO-SVRG) method [32], [37]–[39], and the ZO Hessian-based (ZO-Hess) algorithm [40]. These algorithms employ the same point updating rule (11).

\[ x_t = x_{t-1} - \eta_t m_t. \]

However, they adopt different strategies to form the descent direction \( m_t \) in (10).

- **ZO-SGD** [18]: The descent direction \( m_t \) is set as the current gradient estimate \( m_t = g_t \). Note that ZO-SGD becomes the ZO stochastic coordinate descent (ZO-SCD) method [33] as the coordinate-wise gradient estimate is used. Moreover, if the full batch of stochastic samples are used, then ZO-SGD becomes ZO gradient descent (ZO-GD) [19].

- **ZO-SVRG** [32], [37]–[39]: The descent direction \( m_t \) is formed by breaking \( g_t \) into a control variate of reduced variance, \( m_t = g_t - c_t + \xi \hat{g}_t \), where \( c_t \) denotes a control variate, which is commonly given by a gradient estimate evaluated at \( x_{t-1} \) but the entire dataset of \( n \) empirical samples.

- **ZO-Hess** [40]: The descent direction \( m_t \) incorporates the approximate Hessian \( \hat{H}_t \), where \( \hat{H}_t = \hat{H}_t^{-1/2} g_t \), where \( \hat{H}_t \) is constructed either by the second-order Gaussian Stein’s identity [41] or the diagonalization-based Hessian approximation [40]. The former approach was used in [41] to develop the ZO stochastic cubic regularized Newton (ZO-SCRN) method.

2) **ZO algorithms for constrained optimization:** We next present the ZO projected SGD (ZO-PSGD) [43], the ZO stochastic mirror descent (ZO-SMD) [21], the ZO stochastic conditional gradient (ZO-SCG) algorithm [41], [44], and the ZO adaptive momentum method (ZO-AdaMM) [45] for constrained optimization. The aforementioned algorithms, with the exception of the ZO-AdaMM, specify the descent direction (10) as the current gradient estimate \( m_t = \hat{g}_t \). Their key difference lies in how they implement the point updating step (11).

- **ZO-PSGD** [43]: By letting \( \Pi_{\mathcal{X}} \) be the Euclidean distance based projection operation, the point updating step (11) is given by \( x_t = \arg \min_{x \in \mathcal{X}} \|x - (x_{t-1} - \eta_t m_t)\|^2 \).

- **ZO-SMD** [21]: Upon defining a Bregman divergence \( h_n(x, y) \) with respect to a strongly convex and differentiable function \( h \), \( D_h(x, y) = h(x) - h(y) - (x - y)^T \nabla h(y) \), the point updating step (11) is given by \( x_t = \arg \min_{x \in \mathcal{X}} \|m_t^T x + \frac{1}{\eta_t} D_h(x, x_t) \|^2 \). For example, if \( h(x) = \frac{1}{2} \|x\|^2 \), then \( D_h(x, y) = \frac{1}{2} \|x - y\|^2 \), and \( x_t = \arg \min_{x \in \mathcal{X}} \|x - (x_{t-1} - \eta_t m_t)\|^2 \), which reduces to ZO-PSGD.

- **ZO-SCG** [41], [44]: The point updating step (11) calls for...
a linear minimization oracle \[41\], \(z_t = \arg \min_{x \in X} m_t^T x\), and forms a feasible point update through the linear combination \(x_t = (1 - \eta_t) x_{t-1} + \eta_t z_t\). Similar algorithms, known as ZO Frank-Wolfe, were also developed in \[46\], \[47\].

- ZO-AdaMM \[45\]: Different from ZO-PSGD, ZO-SMD and ZO-SCG, ZO-AdaMM adopts a momentum-type descent direction (rather than the current estimate \(\hat{g}_t\)), an adaptive learning rate (rather than the constant rate \(\eta_t\)), and a projection operation under Mahalanobis distance (rather than Euclidean distance). ZO-AdaMM can strike a balance between the convergence speed and accuracy. However, it requires tuning extra algorithmic hyperparameters in addition to the learning rate and smoothing parameters \[45\], \[48\].

B. ZO optimization in complex settings

Here we review ZO algorithms for composite optimization, min-max optimization, distributed optimization, and structured high-dimensional optimization.

1) ZO composite optimization: Consider the following problem, with a smooth+non-smooth composite objective function, 
\[
\min_{x \in \mathbb{R}^d} f(x) + g(x),
\]
where \(f\) is a black-box smooth function (possibly nonconvex), and \(g\) is a white-box non-smooth regularization function. The form of problem (13) arises in many sparsity-promoted applications, e.g., adversarial attack generation \[49\] and online sensor management \[27\]. The ZO proximal SGD (ZO-ProxSGD) algorithm \[43\] and ZO (stochastic) alternating direction method of multipliers (ZO-ADMM) \[27\], \[50\], \[51\] were developed to solve problem (13). We remark that problem (8) can also be cast as (13) by introducing the indicator function of the constraint \(x \in X\) in the objective of (8) by letting \(g(x) = 0\) if \(x \in X\) and \(\infty\) if \(x \notin X\).

2) ZO min-max optimization: By min-max, we mean that the problem is a composition of inner maximization and outer minimization of the objective function,
\[
\min_{x \in X} \max_{y \in Y} f(x, y)
\]
where \(x \in \mathbb{R}^{d_x}\) and \(y \in \mathbb{R}^{d_y}\) are optimization variables (for ease of notation, let \(d_x = d_y = d\)), \(f\) is a black-box objective function, and \(X\) and \(Y\) are compact convex sets. One motivating application behind problem (14) is the design of black-box poisoning attack \[52\], where the attacker deliberately influences the training data (by injecting poisoned samples) to manipulate the results of a black-box predictive model. To solve the problem posed in (14), the work \[52\], \[53\] presented efficient ZO min-max algorithms for stochastic and deterministic bi-level optimization with nonconvex outer minimization over \(x\) and strongly concave inner maximization over \(y\). They proved the convergence rate to be sub-linear when gradient estimation is integrated with alternating (projected) stochastic gradient descent-ascent methods.

3) ZO distributed optimization: Consider the minimization of a network cost, given by the sum of local objective functions \(\{f_i\}\) at multiple agents
\[
\min_{\{x_i \in X\}} \sum_{i=1}^N f_i(x_i)
\]
subject to \(x_i = x_j, \forall j \in N(i)\).
Here \(N(i)\) denotes the set of neighbors of agent/node \(i\), and the underlying network/graph is connected, namely, there exists a path between every pair of distinct nodes. Some recent works have started to tackle the distributed optimization problem (15) with black-box objectives. In \[54\], a distributed Kiefer Wolfowitz type ZO algorithm was proposed along with convergence analysis, for the case that the objective functions \(\{f_i\}\) are strongly convex. In \[55\], \[56\], the ZO distributed (sub)gradient algorithm and the ZO distributed mirror descent algorithm were developed for nonsmooth convex optimization. In \[57\], \[58\], the convergence of consensus-based distributed ZO algorithms was established for nonconvex (unconstrained) optimization.

4) Structured high-dimensional optimization: Compared to FO algorithms, ZO algorithms typically suffer from a slowdown (proportional to the problem size \(d\) in convergence. Thus, some recent works attempt to mitigate this limitation when solving high-dimensional (large \(d\)) problems. The work \[59\] explored the functional sparsity structure, under which the objective function \(f\) depends only on a subset of \(d\) coordinates. This assumption also implies the gradient sparsity, which enabled the development of a LASSO based algorithm for gradient estimation, and eventually yielded poly-logarithmically dependent on \(d\) when \(f\) is convex. And the work \[41\] established the convergence rate of ZO-SGD which depends on \(d\) only poly-logarithmically under the assumption of gradient sparsity. In addition, the work \[60\] proposed a direct search based algorithm, which yields the convergence rate that is poly-logarithmically dependent on dimensionality for any monotone transform of a smooth and strongly convex objective with a low-dimensional structure. That is, the objective function \(f(x)\)
is supported on a low dimensional manifold $\mathcal{X}$. Another work \[61\] studied the problem of ZO optimization on Riemannian manifolds, and proposed algorithms that only depend on the intrinsic dimension of the manifold by using ZO Riemannian gradient estimates.

C. Convergence rates

We first elaborate on the criteria used to analyze the convergence rate of ZO algorithms under different problem settings.

1) Convex optimization: The convergence error is measured by the optimality gap of function values $E[f(x_T) - f(x^*)]$ for a convex objective $f$, where $x_T$ denotes the updated point at the final iteration $T$, $x^*$ denotes the optimal solution, and the expectation is taken over the full probability space, e.g., random gradient approximation and stochastic sampling.

2) Online convex optimization: The cumulative regret \[62\] is typically used in place of the optimality gap, namely, $E\left[\sum_{t=1}^{T} f_t(x_t) - \min_{x} \sum_{t=1}^{T} f_t(x)\right]$ for an online convex cost function $f_t$, e.g., $f_t(\cdot) = f(\cdot; \xi_t)$ in problem (8).

3) Unconstrained nonconvex optimization: The convergence is evaluated by the first-order stationary condition in terms of the squared gradient norm $\frac{1}{T} \sum_{t=1}^{T} E[\|\nabla f(x_t)\|^2]$ for the nonconvex objective $f$. Since first-order stationary points could be saddle points of a nonconvex optimization problem, the second-order stationary condition is also used to ensure the local optimality of a first-order stationary point (namely, escaping saddle-points) \[41\], \[63\]. The work \[41\] and \[63\] focused on stochastic optimization and deterministic optimization, respectively.

4) Constrained nonconvex optimization: The criterion for convergence is commonly determined by detecting a sufficiently small squared norm of the gradient mapping \[43\], \[64\]. $P_X(x_t, \nabla f(x_t), \eta_t) := \frac{1}{\eta_t} [x_t - \Pi_X (x_t - \eta_t \nabla f(x_t))]$, where the notation follows \[11\]. $P_X(x_t, \nabla f(x_t), \eta_t)$ can naturally be interpreted as the projected gradient, which offers a feasible update from the previous point $x_t$. The Frank-Wolfe duality gap is another commonly-used convergence criterion \[41\], \[44\], \[46\], \[47\], given by max$_{x \in \mathcal{X}} \langle x - x_t, -\nabla f(x_t) \rangle$. It is always non-negative, and becomes 0 if and only if $x_t \in \mathcal{X}$ is a stationary point.

More generally, given a convergence measure $\mathcal{M}(\cdot)$, $x$ is called an $\epsilon$-optimal solution if $\mathcal{M}(x) \leq \epsilon$. The convergence error is typically expressed as a function of the number of iterations $T$, relating the convergence rate to the iteration complexity. Since the convergence analysis of existing ZO algorithms varies under different problem domains and algorithmic parameter settings, in Table I we compare the convergence performance of ZO algorithms covered in this section from 5 perspectives: problem structure, type of gradient estimates, smoothing parameter, convergence error, and function query complexity.

IV. APPLICATION: ADVERSARIAL EXAMPLE GENERATION

In this section, we present the application of ZO optimization to the generation of prediction-evasive adversarial examples to fool DL models. Adversarial examples, also known as evasion attacks, are inputs corrupted with imperceptible adversarial perturbations (to be designed) toward misclassification (namely, prediction different from true image labels) \[22\], \[65\].

Most studies on adversarial vulnerability of DL have been restricted to the white-box setting where the adversary has complete access and knowledge of the target system (e.g., DNNs) \[22\], \[65\]. However, it is often the case that the internal states/configurations and the operating mechanism of DL systems are not revealed to the practitioners (e.g., Google Cloud Vision API). This gives rise to the problem of black-box adversarial attacks \[23\], \[24\], \[66\]–\[69\], where the only mode of interaction of the adversary with the system is via submission of inputs and receiving the corresponding predicted outputs.

More formally, let $z$ denote a legitimate example, and $z' := z + x$ denote an adversarial example with the adversarial perturbation $x$. Given the learned ML/DL model $\theta$, the problem of adversarial example generation can be cast as an optimization problem of the following generic form \[49\], \[65\] minimize $f(x + x; \theta) + \lambda g(x)$, subject to $\|x\|_{\infty} \leq \epsilon$, $z' \in [0, 1]^d$. where $f(z'; \theta)$ denotes the (black-box) attack loss function for fooling the model $\theta$ using the perturbed input $z'$ (see \[23\] for a specific formulation), $g(x)$ is a regularization function that penalizes the sparsity or the structure of adversarial perturbations, e.g., group sparsity in \[70\]. $\lambda \geq 0$ is a regularization parameter, the $\ell_\infty$ norm enforces similarity between $z'$ and $z$, and the input space of ML/DL systems is normalized to $[0, 1]^d$. If $\lambda \neq 0$, then problem (16) is in the
form of composite optimization, and ZO-ADMM is a well-suited optimizer. If \( \lambda = 0 \), a solution to problem (16) is known as a black-box \( \ell_\infty \) attack \[24\], which can be obtained using ZO methods for constrained optimization.

In Table [1], we present black-box \( \ell_\infty \) attacks with respect to 5 ImageNet images against the Inception V3 model \[71\]. The adversarially perturbed images are obtained from 4 ZO methods including ZO-PSGD, ZO-SMD, ZO-AdaMM, and ZO-NES (a projected version of ZO-signSGD that is used in practice \[24\]). We demonstrate the attack performance of different ZO algorithms in terms of the \( \ell_2 \) norm of the generated perturbations and the number of queries needed to achieve a first successful black-box attack. As we can see, ZO-PSGD typically has the fastest speed of converging to a valid adversarial example, while ZO-AdaMM has the best convergence accuracy in terms of the smallest distortion required to fool the neural network.

### V. APPLICATION: ONLINE SENSOR MANAGEMENT

ZO optimization is also advantageous when it is difficult to compute the first-order gradient of an objective function. Online sensor management provides an example of such a scenario \[27\], \[72\]. Sensor selection for parameter estimation is a fundamental problem in smart grids, communication systems, and wireless sensor networks \[73\]. The goal is to seek the optimal tradeoff between sensor activations and the estimation accuracy over a time period.

We consider the cumulative loss for online sensor selection \[27\]

\[
\begin{align*}
\text{minimize} & & 1/T \sum_{t=1}^{T} \left[ -\log \left( \sum_{i=1}^{d} x_i a_{i,t} a_{i,t}^T \right) \right] \\
\text{subject to} & & 1^T x = m_0, \quad 0 \leq x \leq 1, \\
& & \text{where } x \in \mathbb{R}^d \text{ is the optimization variable, } d \text{ is the number of sensors, } a_{i,t} \in \mathbb{R}^n \text{ is the observation coefficient of sensor } i \text{ at time } t, \text{ and } m_0 \text{ is the number of selected sensors. The objective function of (17) can be interpreted as the log determinant of the error covariance matrix associated with the maximum likelihood.}
\end{align*}
\]
A similar explanation holds for other rows, except that different \( ZO \in \{ \leq \} \) for sensor selection. In the top plots, we show the mean squared error (MSE) of parameter estimation using different number of selected sensors \( m_0 \) in [17]. As we can see, ZO-ADMM yields almost the same MSE as FO-ADMM in the context of parameter estimation using \( m_0 \) activated sensors, determined by the hard thresholding of continuous sensor selection schemes, i.e., solutions of problem [17] obtained from ZO-ADMM and FO-ADMM.

![Fig. 2: Comparison between ZO-ADMM and FO-ADMM for solving the sensor selection problem [17]. Top: ADMM primal-dual residuals versus number of iterations. Bottom: Mean squared error of activated sensors for parameter estimation versus the total number of selected sensors.](image)

In this section we discuss some other recent applications of ZO optimization in signal processing and machine learning.

**A. Model-agnostic constrastive explanations.** Explaining the decision making process of a complex ML model is crucial to many ML-assisted high-stakes applications, such as job hiring, financial loan application and judicial sentence. When generating local explanations for the prediction of an ML model on a specific data sample, one common practice is to leverage the information of its input gradient for sensitivity analysis of the model prediction. For ML models that do not have explicit functions for computing input gradients, such as access-limited APIs or rule-based systems, ZO optimization enables the generation of local explanations using model queries without the knowledge of the gradient information. Moreover, even when the input gradient can be obtained via ML platforms

| True label | brambling | cannon | pag-dog | fly | armadillo | balloon |
|-----------|-----------|--------|---------|----|-----------|---------|
| Perturbed image | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) |
| Prediction label | goldfinch | plow | bucket | cicada | croquet ball | parachute |
| \( \ell_2 \) Distortion | 35.2137 87.7304 72.3397 101.068 172.719 35.9330 |
| \# of queries | 7640 150 130 50 210 5830 |

| Prediction label | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) |
| \( \ell_2 \) Distortion | 8.9708 26.0126 20.9504 30.0968 45.097 10.9023 |
| \# of queries | 29980 350 260 140 570 15830 |

| Prediction label | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) |
| \( \ell_2 \) Distortion | 5.9502 5.7359 4.5753 4.4956 6.3149 7.7405 |
| \# of queries | 53300 1710 790 540 2780 32900 |

| Prediction label | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) | ![Perturbed image](image) |
| \( \ell_2 \) Distortion | 54.9956 34.5852 28.7035 28.9158 40.1483 51.7116 |
| \# of queries | 22110 1080 830 430 2280 15830 |

**TABLE II:** Comparison of various ZO methods for generating untargeted adversarial attacks against Inception V3 model over 5 ImageNet images. Row 1: true labels of given images. Row 2-4: Results obtained using ZO-SMD, which include perturbed images, corresponding prediction labels, \( \ell_2 \) norm of perturbations, and the number of queries when achieving the first successful black-box attack. A similar explanation holds for other rows, except that different ZO methods are used.

estimator for parameter estimation [74]. The constraint \( 0 \leq x \leq 1 \) is a relaxed convex hull of the Boolean constraint \( x \in \{0, 1\}^m \), which encodes whether or not a sensor is selected.

Conventional methods such as projected gradient (first-order) and interior-point (second-order) algorithms can be used to solve problem [17]. However, both methods involve the calculation of inverses of large matrices that are necessary to evaluate the gradient of the cost function. The matrix inversion step is usually a bottleneck while acquiring the gradient information in high dimensions and it is particularly problematic in the online optimization setting. Since problem [17] involves mixed equality and inequality constraints, it has been shown [27] that ZO-ADMM is an effective ZO optimization method for circumventing the computational bottleneck.

In Figure 2 we compare the performance of ZO-ADMM and that of FO-ADMM [75] for sensor selection. In the top plots, we demonstrate the primal-dual residuals in ADMM against the number of iterations. As we can see, ZO-ADMM has a slower convergence rate than FO-ADMM, and it approaches the accuracy of FO-ADMM as the number of iterations increases. In the bottom plots, we show the mean squared error (MSE) of parameter estimation using different number of selected sensors \( m_0 \) in [17]. As we can see, ZO-ADMM yields almost the same MSE as FO-ADMM in the context of parameter estimation using \( m_0 \) activated sensors, determined by the hard thresholding of continuous sensor selection schemes, i.e., solutions of problem [17] obtained from ZO-ADMM and FO-ADMM.

VI. OTHER RECENT APPLICATIONS

In this section we discuss some other recent applications of ZO optimization in signal processing and machine learning.

A. Model-agnostic constrastive explanations. Explaining the decision making process of a complex ML model is crucial to many ML-assisted high-stakes applications, such as job hiring, financial loan application and judicial sentence. When generating local explanations for the prediction of an ML model on a specific data sample, one common practice is to leverage the information of its input gradient for sensitivity analysis of the model prediction. For ML models that do not have explicit functions for computing input gradients, such as access-limited APIs or rule-based systems, ZO optimization enables the generation of local explanations using model queries without the knowledge of the gradient information. Moreover, even when the input gradient can be obtained via ML platforms
such as TensorFlow and PyTorch, the gradient computation is platform-specific. In this case, ZO optimization has the advantage of alleviating platform dependency when developing multi-platform explanation methods, as it only depends on model inference results.

**Digit classification**

| Instance | PP  | PN |
|----------|-----|----|
| S        | S   |    |
| D        |     |    |

(a)

**Credit loan approval**

Alice’s loan application would be approved if
Consolidated risk markers changes from 65 to 72 AND
Average age of accounts in months changes from 52 to 68 AND
Months since most recent credit inquiry not within the last 7 days changes from 2 to 3

(b)

**Fig. 3:** Contrastive explanations generated by ZO optimization methods. (a) For hand-written digit classification, the red digit class on the corner of an input sample shows the model prediction of the instance. The pixels highlighted by the cyan color are the pertinent positive (PP) supporting the original prediction. The pixels highlighted by the purple color are the pertinent negative (PN) that will alter the model prediction when added to the original instance. (b) For the credit loan application, the PN of an applicant (Alice) is used to explain the necessary modifications on a subset of the original features in order to change the model prediction from ‘denial’ to ‘approval’.

Here, we apply ZO optimization to generating contrastive explanations [76] for two ML applications – handwritten digit classification and loan approval. Contrastive explanations consist of two components derived from a given data sample for explaining the model prediction, i.e., a pertinent positive (PP) that is minimally and sufficiently present to keep the same prediction of the original input sample, and a pertinent negative (PN) that is minimally and necessarily absent to alter the model prediction. The process of finding PP and PN is formulated as a sparsity-driven and data-perturbation based optimization problem guided by the model prediction outcomes [25], which can be solved by ZO optimization methods. Fig. 3 shows the contrastive explanations generated from black-box neural network models by ZO-GD using the objective functions in [25]. For hand-written digit classification, the PP identifies a subset of pixels such that their presence is minimally sufficient for model prediction. Moreover, the PN identifies a subset of pixels such that their absence is minimally necessary for altering model prediction. The PP and PN together constitute a contrastive explanation for interpreting model prediction. Similarly, for credit loan approval task trained on the FICO explainable machine learning challenge dataset [76] based on a neural network model, the PN generated for an applicant (Alice) can be used to explain how the model would alter the recommendation from ‘denial’ to ‘approval’ based on Alice’s loan application profile [76].

**B. Policy search in reinforcement learning.** Reinforcement learning aims to determine given a state which action to take (or policy) in order to maximize the reward. One of the most popular policy search approaches is the model-free policy search, where agent learns parameterized policies from sampled trajectories without needing to learn the model of the underlying dynamics. Model-free policy search updates the parameters such that trajectories with higher reward are more likely to be obtained when following the updated policy [77]. Traditional policy search methods, such as REINFORCE [78], rely on randomized exploration in the action space to compute an estimated direction of improvement. These methods (referred to as policy gradient methods) then leverage the first order information of the policy (or Jacobian) to update its parameters to maximize the reward. Note that the chance of finding a sequence of actions resulting in high total reward decreases as the horizon length increases and thus policy gradient methods often exhibit high variance and result in large sample complexity [79].

To alleviate these problems, ZO policy search methods, which directly optimize over policy parameter space, have emerged as an alternative to policy gradient. More specifically, ZO policy search methods seek to directly optimize the total reward in the space of parameters by employing finite-difference methods to compute estimates of the gradient with respect to policy parameters [77], [80] – [83]. These methods are fully zeroth-order, i.e., they do not exploit first-order information of the policy, the reward, or the dynamics. Interestingly, it has been observed that although policy gradient methods leverage more information, ZO policy search methods often perform better empirically. In particular, the work [82] characterized

https://community.fico.com/s/explainable-machine-learning-challenge

1 Please refer to https://aix360.mybluemix.net for more details.
the convergence rate of ZO policy optimization when applied to linear-quadratic systems. And the work [83] theoretically showed that the complexity of exploration in the action space (using policy gradients) depends on both the dimensionality of the action space and the horizon length, as opposed to, the complexity of exploration in the parameter space (using ZO methods) depends only on the dimensionality of the parameter space.

C. Automated ML. The success of ML relies heavily on selecting the right pipeline algorithms for the problem at hand, and on setting its hyperparameters. Automated ML (AutoML) automates the process of model selection and hyperparameter optimization. It offers the advantages of producing simpler solutions, faster creation of those solutions, and models that often outperform hand-designed machine learning models. One could view AutoML as the process of optimization of an unknown black-box function. Recently, several Bayesian optimization (BO) approaches have been proposed for AutoML [26], [84]. BO works by building a probabilistic surrogate via Gaussian process (GP) for the objective function, and then using an acquisition function defined from this surrogate to decide where to sample. However, BO suffers from a computational bottleneck: an internal first-order solver is required to determine the parameters of the GP model by maximizing the log marginal likelihood of the current function evaluations at each iteration of BO. The first-order solver is slow due to the difficulty of computing the gradient of the log-likelihood function with respect to the parameters of GP. To circumvent this difficulty, the ZO optimization algorithm can be used to determine the hyperparameters and thus to accelerate BO in AutoML [26]. In the context of meta-learning, ZO optimization has also been leveraged to obviate the need for determining computationally-intensive high-order derivatives during meta-training [29]. Lastly, we note that ZO optimization can be integrated with learning-to-optimize (L2O), which models the optimizer through a trainable DNN-based meta-learner [85], [86].

A. ZO optimization with non-smooth objectives. There exists a gap between the theoretical analysis of ZO optimizers and practical ML/DL applications with non-smooth objectives, where the former usually requires the smoothness of the objective function. There are two possible means of relaxing the smoothness assumption. First, the randomized smoothing technique ensures that the convolution of two functions is at least as smooth as the smoothest of the two original functions. Thus, $f_\mu$ is smooth even if $f$ is non-smooth in [2]. This motivates the technique of double randomization that approximates a subgradient of a non-smooth objective function [21], where an extra randomized perturbation is introduced to prevent drawing points from non-smooth regions of $f$. The downside of double randomization is the increase of function query complexity. Second, a model-based trust region method can be leveraged to approximate the subgradient/gradient using linear or quadratic interpolation [6], [31]. This leads to the general approach of gradient estimation without imposing extra assumptions on the objective function. However, it increases the computation cost due to the need to solve nested regression problems.

B. ZO optimization with black-box constraints. The current work on ZO optimization is restricted to black-box objective functions with white-box constraints. In the presence of black-box constraints, the introduction of barrier functions (instead of constraints) [87] in the objective could be a potential solution. One could also employ the method of multipliers to reformulate black-box constraints as regularization functions in the objective function [26].

C. ZO optimization for privacy-preserving distributed learning. To protect the sensitive information of data in the context of distributed learning, it is common to add ‘noise’ (randomness) into gradients of individual cost functions of agents, known as message-perturbing privacy strategy [88]. The level of privacy is often evaluated by differential privacy (DP). A high degree of DP prevents the adversary from gaining meaningful personal information of any individuals. Similarly, ZO optimization also conceals the gradient information and allows the use of noisy gradient estimates that are constructed from function values. Thus, one interesting question is: can ZO optimization be designed with privacy guarantees? In a more general sense, it would be worthwhile to examine what roles ZO optimization plays in the privacy-preserving and
When a sub-derivative is infeasible or difficult to compute, the variables, such as texts, graphs, sets and categorical data. In particular, when the high-order derivatives (beyond gradient) are required, e.g., model-agnostic meta-learning [90], ZO optimization could help to overcome the derivative bottleneck.

E. ZO optimization for discrete variables. Many machine learning and signal processing tasks involve handling discrete variables, such as texts, graphs, sets and categorical data. In addition to the technique of relaxation to continuous values, it is worthwhile to explore and design ZO algorithms that directly operate on discrete domains.

F. Tight convergence rates of ZO methods. Although the optimal rate for ZO unconstrained convex optimization was studied in [21], there remain many open questions on seeking the optimal rates, or associated tight lower bounds, for general cases of ZO constrained nonconvex optimization.

VIII. CONCLUSIONS

In this survey paper, we discussed various variants of ZO gradient estimators and focused on their statistical modelling as this leads to general ZO algorithms. We also provided an extensive comparison of ZO algorithms and discussed their iteration and function query complexities. Furthermore, we presented numerous emerging applications of ZO optimization in signal processing and machine learning. Finally, we highlighted some unsolved research challenges in ZO optimization research and presented some promising future research directions.

ACKNOWLEDGEMENT

B. Kailkhura was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.
A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg, “A theoretical evaluation of adversarial examples,” ICML, 2015.

P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zero: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” in Proc. 10th ACM Workshop on Artificial Intelligence and Security, 2017, pp. 15–26.

A. Dhurandhar, T. Pedapati, A. Balakrishnan, et al., “Model agnostic contrastive explanations for structured data,” arXiv preprint arXiv:1906.00117, 2019.

S. Liu, P. Ram, D. Vijaykeerthy, et al., “An ADMM based framework for AutoML pipeline configuration,” AAAI, 2019.

S. Liu, J. Chen, P.-Y. Chen, and A. O. Hero, “Zeroth-order online ADMM: Convergence analysis and applications,” in AISTATS, 2018, vol. 84, pp. 288–297.

S.-M. Moosavi-Dezfooli, A. Fawzi, J. Uesato, and P. Frossard, “Robustness via curvature regularization, and vice versa,” in IEEE CVPR, 2019, pp. 9078–9086.

X. Song, W. Gao, Y. Yang, K. Choromanski, A. Pacchiano, and Y. Tang, “Es-maml: Simple hessian-free meta learning,” in ICLR, 2020.

Y.-Y. Tsai, P.-Y. Chen, and T.-Y. Ho, “Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources,” ICML, 2020.

A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg, “A theoretical and empirical comparison of gradient approximations in derivative-free optimization,” arXiv preprint arXiv:1905.01332, 2019.

S. Liu, B. Kailkhura, P.-Y. Chen, P. Ting, S. Chang, and L. Amini, “Zeroth-order stochastic variance reduction for nonconvex optimization,” NeurIPS, 2018.

J. C. Spall et al., “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation,” IEEE transactions on automatic control, vol. 37, no. 3, pp. 342–341, 1992.

X. Lian, H. Zhang, C.-J. Hsieh, Y. Huang, and J. Liu, “A comprehensive linear speedup analysis for asynchronous stochastic parallel optimization from zeroth-order to first-order,” in NeurIPS, 2016, pp. 3054–3062.

O. Shamir, “An optimal algorithm for bandit and zero-order convex optimization with two-point feedback,” JMLR, vol. 18, no. 52, pp. 1–11, 2017.

S. Liu, P.-Y. Chen, X. Chen, and M. Hong, “signSGD via zeroth-order oracle,” in ICLR, 2019.

B. Gu, Z. Huo, and H. Huang, “Zeroth-order asynchronous doubly stochastic algorithm with variance reduction,” arXiv preprint arXiv:1612.04245, 2016.

L. Liu, M. Cheng, C.-J. Hsieh, and D. Tao, “Stochastic zeroth-order optimization via variance reduction method,” arXiv preprint arXiv:1805.11811, 2018.

K. Ji, Z. Wang, Y. Zhou, and Y. Liang, “Improved zeroth-order variance reduced algorithms and analysis for nonconvex optimization,” in ICML, 2019, vol. 97, pp. 3100–3109.

H. Ye, Z. Huang, C. Fang, C. J. Li, and T. Zhang, “Hessian-aware zeroth-order optimization for black-box adversarial attack,” arXiv preprint arXiv:1812.11377, 2018.

K. Balasubramanian and S. Ghadimi, “Zeroth-order nonconvex stochastic optimization: Handling constraints, high-dimensionality, and saddle-points,” arXiv preprint arXiv:1809.06474, 2019.

M. Cheng, S. Singh, P.-Y. Chen, S. Liu, and C.-J. Hsieh, “Sign-OPT: A query-efficient hard-label adversarial attack,” ICLR, 2020.

S. Ghadimi, G. Lan, and H. Zhang, “Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization,” Mathematical Programming, vol. 155, no. 1-2, pp. 267–305, 2016.

K. Balasubramanian and S. Ghadimi, “Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates,” in NeurIPS, 2018, pp. 3455–3464.

X. Chen, S. Liu, K. Xu, X. Li, X. Lin, M. Hong, and D. Cox, “ZO-AdaMM: Zeroth-order adaptive momentum method for black-box optimization,” in NeurIPS, 2019, pp. 7202–7213.

J. Chen, J. Yi, and Q. Gu, “A Frank-Wolfe framework for efficient and effective adversarial attacks,” in AAAI, 2020.

A. K. Sahu, M. Zaheer, and S. Kar, “Towards gradient free and projection free stochastic optimization,” AISTATS, 2019.

X. Chen, S. Liu, R. Sun, and M. Hong, “On the convergence of a class of adaptively selected nonconvex optimization,” in ICLR, 2019.

P. Zhao, S. Liu, P.-Y. Chen, N. Huang, K. Xu, B. Kailkhura, and X. Lin, “On the design of black-box adversarial examples by leveraging gradient-free optimization and operator splitting method,” in ICCV, 2019.

X. Gao, B. Jiang, and S. Zhang, “On the information-adaptive variants of the ADMM: an iteration complexity perspective,” Optimization Online, vol. 12, 2014.

F. Huang, S. Gao, S. Chen, and H. Huang, “Zeroth-order stochastic alternating direction method of multipliers for nonconvex nonsmooth optimization,” IJCAI, 2019.

S. Liu, S. Lu, X. Chen, Y. Feng, K. Xu, A. Al-Dujaili, M. Hong, and U.-M. O’Reilly, “Min-max optimization without gradients: Convergence and applications to adversarial ML,” in ICML, 2020.

Z. Wang, K. Balasubramanian, S. Ma, and M. Razaviyayn, “Zeroth-order algorithms for nonconvex minimax problems with improved complexities,” arXiv preprint arXiv:2001.07819, 2020.

A. K. Sahu, D. Jakovetic, D. Bajovic, and S. Kar, “Distributed zeroth order optimization over random networks: A kiefer-wolfowitz stochastic approximation approach,” in IEEE CDC, 2018, pp. 4951–4958.

D. Yuan, D. W. C. Ho, and S. Xu, “Zeroth-order method for distributed optimization with approximate projections,” IEEE Trans. Neural Netw. Learn. Syst., vol. 27, no. 2, pp. 284–294, 2015.

Z. Yu, D. W. C. Ho, and D. Yuan, “Distributed randomized gradient-free mirror descent algorithm for constrained optimization,” arXiv preprint arXiv:1903.04157, 2019.

D. Hajinezhad, M. Hong, and A. Garcia, “Zone: Zeroth-order nonconvex multiagent optimization over networks,” IEEE Trans. Autom. Control, vol. 64, no. 10, pp. 3995–4010, 2019.

Y. Tang and N. Li, “Distributed zero-order algorithms for nonconvex multi-agent optimization,” in Allerton, 2019, pp. 781–786.

Y. Wang, S. Du, S. Balakrishnan, and A. Singh, “Stochastic zeroth-order optimization in high dimensions,” in AISTATS. April 2018, vol. 84, pp. 1356–1365, PMLR.

D. Golovin, J. Karro, G. Kochanski, C. Lee, X. Song, and Q. Zhang, “Gradientless descent: High-dimensional zeroth-order optimization,” in ICLR, 2020.

J. Li, K. Balasubramanian, and S. Ma, “Zeroth-order optimization on Riemannian manifolds,” arXiv preprint arXiv:2003.11238, 2020.
E. Hazan, “Introduction to online convex optimization,” Foundations and Trends® in Optimization, vol. 2, no. 3-4, pp. 157–325, 2016.

E.-V. Vlatakis-Gkaragkounis, L. Flokas, and G. Piliouras, “Efficiently avoiding saddle points with zero order methods: No gradients required,” in NeurIPS, 2019, pp. 10066–10077.

S. J. Reddi, S. Sra, B. Poczos, and A. J. Smola, “Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization,” in NeurIPS, 2016, pp. 1145–1153.

N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in IEEE Symposium on Security and Privacy, 2017, pp. 39–57.

M. Cheng, T. Le, P.-Y. Chen, J. Yi, H. Zhang, and C.-J. Hsieh, “Query-efficient hard-label black-box attack: An optimization-based approach,” ICML, 2019.

P. Zhao, P.-Y. Chen, S. Wang, and X. Lin, “Towards query-efficient black-box adversary with zeroth-order natural gradient descent,” AAAI, 2020.

K. Xu, S. Liu, P. Zhao, et al., “Structured adversarial attack: Towards general implementation and better interpretability,” in ICLR, 2019.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in IEEE CVPR, 2016, pp. 2818–2826.

T. Chen and G. B. Giannakis, “Bandit convex optimization for scalable and dynamic IoT management,” IEEE Internet of Things Journal, 2018.

A. O. Hero and D. Cochran, “Sensor management: Past, present, and future,” IEEE Sensors J., vol. 11, no. 12, pp. 3064–3075, 2011.

S. Joshi and S. Boyd, “Sensor selection via convex optimization,” IEEE Trans. Signal Process., vol. 57, no. 2, pp. 451–462, 2009.

T. Suzuki, “Dual averaging and proximal gradient descent for online alternating direction multiplier method,” in ICML, 2013, pp. 392–400.

A. Dhurandhar, P.-Y. Chen, et al., “Explanations based on the missing: Towards contrastive explanations with pertinent negatives,” in NeurIPS, 2018, pp. 592–603.

J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” The International Journal of Robotics Research, vol. 32, no. 11, pp. 1238–1274, 2013.

R. J Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” Machine learning, vol. 8, no. 3-4, pp. 229–256, 1992.

T. Zhao, H. Hachiya, G. Niu, and M. Sugiyama, “Analysis and improvement of policy gradient estimation,” in NeurIPS, 2011, pp. 262–270.

T. Salimans, H. Ho, X. Chen, S. Sidor, and I. Sutskever, “Evolution strategies as a scalable alternative to reinforcement learning,” arXiv preprint arXiv:1703.03864, 2017.

F. Sehnke, C. Osendorf, Thomas Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, “Parameter-exploring policy gradients,” Neural Networks, vol. 23, no. 4, pp. 551–559, 2010.