Risk-aware learning for scalable voltage optimization in distribution grids☆

Shanny Lin *, Shaohui Liu, Hao Zhu

Department of Electrical and Computer Engineering, The University of Texas at Austin, United States of America

ABSTRACT

Real-time coordination of distributed energy resources (DERs) is crucial for regulating the voltage profile in distribution grids. By capitalizing on a scalable neural network (NN) architecture, one can attain decentralized DER decisions to address the lack of real-time communications. This paper develops an advanced learning-enabled DER coordination scheme by accounting for the potential risks associated with reactive power prediction and voltage deviation. Such risks are quantified by the conditional value-at-risk (CVaR) using the worst-case samples only, and we propose a mini-batch selection algorithm to address the training speed issue in minimizing the CVaR-regularized loss. Numerical tests using real-world data on the IEEE 123-bus test case have demonstrated the computation and safety improvements of the proposed risk-aware learning algorithm for decentralized DER decision making, especially in terms of reducing feeder voltage violations.

1. Introduction

Rapid integration of distributed energy resources (DERs) opens up new opportunities of flexible and adaptive support to the operations of power distribution grids. The smart inverters of fast-acting DERs have been popularly advocated for the feeder voltage optimization by adjusting their reactive power outputs [1]. It is crucial to design a scalable coordination framework for these heterogeneous DERs under limited real-time communications available in distribution systems.

Coordinating reactive power setpoints of DERs can be viewed as an optimal power flow (OPF) problem that requires all feeder-wide information. To bypass the high communication overhead of a centralized solution, several distributed and decentralized optimization algorithms have attracted significant interest in the last years; see e.g., [2–4]. While those algorithms can greatly simplify the communication graph to have information exchange only among neighbors or with the control center, they still require high-rate bi-directional communications to cope up with the fast dynamics in distribution grids. With the increasing volume of available data, the recent trend is to leverage machine learning (ML) tools to attain decentralized decision rules that map from the system operating condition (OC) to the optimal decisions [5–9]. By and large, these ML-based approaches have been designed to reduce the sample average of prediction error, or the *average loss*. Accordingly, they may fail to address the largest sample losses or worst-case scenarios in terms of prediction error or even the violation of voltage limits.

Our goal is to develop a risk-aware learning framework to improve the safety of scalable decision rules developed for distribution grid voltage optimization. Inspired by earlier work [5–9], we design a scalable neural network architecture such that each nodal inverter can learn its optimal reactive power from local measurements and selected information only. To address the aforementioned issue of optimizing average loss only, it is possible to adopt a post-analysis approach that reduces the worst-case errors of the resultant NN models by restricting the input domain [10]. Alternatively, we propose to *systematically* incorporating the worst-case measure into the training process by introducing a regularization term on the *conditional value-at-risk (CVaR)* [11] of the predicted decisions. The CVaR metric corresponds to the average of the largest sample losses, or the worst-case scenarios, and thus optimizing it can improve the safety associated with the reactive power decisions. Specifically for our voltage optimization task, we also use the CVaR metric to quantify the worst-case voltage deviation performance. This way, the NN design is also physics-informed by incorporating the system-wide voltage limits; see .e.g., [12]. Notably, the CVaR metric is defined over a subset of samples (worst ones) and thus can suffer from the learning efficiency issue. We further propose to accelerate the mini-batch gradient descent for NN training [13,14] by selecting those mini-batches attaining a certain CVaR threshold. Accordingly, gradient updates are performed only on the mini-matches of statistical significance, thus reducing the number of epochs. Our algorithm has effectively improved the computation time for the training process and reduced the voltage violations based on numerical tests.

It is worth mentioning the proposed risk-aware learning framework is very related to a recent work [6] on addressing the CVaR risk of voltage deviations in OPF-based NN training. While this paper incorporates

CVaR in a stochastic optimization framework, our proposed learning approach here directly uses the scenarios generated by OPF solutions. Thus, the present work can be easily generalized to incorporate other operational considerations in real-time feeder decision making. Notably, we also present an algorithmic solution to effectively accelerate the NN training process.

The rest of this paper is organized as follows. Section 2 presents the system modeling and the centralized voltage optimization problem. In Section 3, we first formulate the risk-aware learning problem by introducing the CVaR losses for prediction and voltage deviations. A mini-batch selection scheme is developed to accelerate the training process. Numerical tests using real-world data on the single-phase equivalent of the IEEE-123 bus test case are presented in Section 4 to demonstrate the computation and safety improvements of the proposed algorithm. The paper is wrapped up in Section 5.

Notation: Upper (lower) boldface symbols stand for matrices (vectors); $(\cdot)^{\mathsf{T}}$ stands for matrix transposition; $\|\cdot\|_2$ denotes the L_2-norm; $|\cdot|$ denotes the absolute value; ∇_x denotes the gradient with respect to x; $\mathbb{1}$ denotes the indicator function; and $\mathbf{0}$ stands for an all-zero vector of appropriate size.

2. System modeling

Consider a radial distribution feeder consisting of $(N+1)$ buses with bus 0 denoting the reference bus at the feeder head. For simplicity, this work focuses on single-phase feeders, while results can be extended to multi-phase systems following earlier approaches such as [15]. Let vectors $\mathbf{p} \in \mathbb{R}^N$ and $\mathbf{q} \in \mathbb{R}^N$ collect the net active and reactive power injections, respectively, at all non-reference buses. The net reactive power consists of DER generation $\mathbf{q}^g$ (controllable) and load consumption $\mathbf{q}^c$ (non-flexible) such that $\mathbf{q} := \mathbf{q}^g - \mathbf{q}^c$; similarly, the net active power injection is $\mathbf{p} = \mathbf{p}^g - \mathbf{p}^c$.

The distribution voltage optimization task aims to coordinate the controllable $\mathbf{q}^g$ from DERs to support system operations in terms of feeder voltage regulation [2,3] or phase balance in multi-phase systems [16,17]. We formulate a general centralized problem of minimizing a system-wide operational objective in order to satisfy voltage limit constraints, while adhering to reactive power limits Q, as

$$\mathbf{z} = \min_{\mathbf{q}^g \in Q} \quad o(\mathbf{q}^g) \tag{1a}$$

$$\text{s. to} \quad \mathbf{g}(\mathbf{q}^g) \leq \mathbf{0} \tag{1b}$$

where $\mathbf{z}$ denotes the optimal value of $\mathbf{q}^g$ by solving (1). To model the system-wide power flow in (1), one can simplify the accurate nonlinear power flow model by adopting the linearized DistFlow (LDF) approximation [18]. The LDF model represents feeder voltage to be linear with respect to (wrt) power injections $\mathbf{p}$ and $\mathbf{q}$, and has been shown very effective for developing algorithms of distribution monitoring [19,20] and voltage optimization [21,22]. The per-unit (pu) voltage *deviation* from the reference bus voltage is approximated by the LDF model as

$$\mathbf{v} \approx \mathbf{R}\mathbf{p} + \mathbf{X}\mathbf{q} \tag{2}$$

where matrices $\mathbf{R} \in \mathbb{R}^{N \times N}$ and $\mathbf{X} \in \mathbb{R}^{N \times N}$ depend on the feeder topology and line parameters.

For the system-wide objective, one can consider the feeder ohmic loss which is quadratic wrt $\mathbf{p}$ and $\mathbf{q}$ [23]. This is because the line power flow under LDF equivalently aggregates the total down-stream power injections. Thus, the objective in (1a) becomes a convex quadratic function of $\mathbf{q}^g$, given by

$$o(\mathbf{q}^g) = (\mathbf{q}^g)^{\mathsf{T}}\mathbf{R}\mathbf{q}^g - 2(\mathbf{q}^c)^{\mathsf{T}}\mathbf{R}\mathbf{q}^g. \tag{3}$$

In regard to the constraint set Q, each node n has a reactive power limit depending on the apparent power rating and active power output of its own inverter. Collectively, the reactive power limit is set to be

$$Q := \{\mathbf{q}^g : |q_n^g| \leq \bar{q}_n^g, \ \forall n = 1, \ldots, N\}. \tag{4}$$

Note that non-controllable nodes can be easily included too by setting the corresponding limits $\bar{q}_n^g = 0$. The inequality constraint in (1b) limits all non-reference bus voltage deviations $\mathbf{v}$ to be within a fixed range $[\underline{\mathbf{v}}, \overline{\mathbf{v}}]$, by

$$\mathbf{g}(\mathbf{q}^g) = \begin{bmatrix} \mathbf{X}\mathbf{q}^g + \mathbf{h}(\mathbf{y}) - \overline{\mathbf{v}} \\ -\mathbf{X}\mathbf{q}^g - \mathbf{h}(\mathbf{y}) + \underline{\mathbf{v}} \end{bmatrix} \leq \mathbf{0} \tag{5}$$

where $\mathbf{h}(\mathbf{y}) = \mathbf{R}(\mathbf{p}^g - \mathbf{q}^c) - \mathbf{X}\mathbf{q}^c$ with $\mathbf{y} := [\mathbf{p}^c; \ \mathbf{p}^q; \ \mathbf{q}^c]$ capturing all system-wide inputs to problem (1) for determining the feeder's operating condition (OC).

Clearly, the voltage optimization problem (1) is a linearly-constrained quadratic program (LCQP) which can be efficiently solved given the full feeder model and system-wide OC. We will introduce a risk-aware learning framework to attain scalable and safe decision rules for real-time DER operations under minimal system-wide information. Before that, the following remark discusses the generalizability of the presented models and formulation for voltage optimization.

Remark 1 (*Multi-Phase and Nonlinear Models*)**.** The optimization problem (1) can be extended to multi-phase systems by using the general multi-phase LDF model; see e.g., [15]. Basically, the network matrices $\mathbf{R}$ and $\mathbf{X}$ are formed to capture phase-to-phase connections in multi-phase lines, in a similar fashion to the single-phase case. In addition, the problem (1) can be formulated using different types of linearized approximation as well, such as the fixed-point linearization in [24]. Last, one can formulate the problem using nonlinear ac power flow model too, thanks to the popular convex relaxation based approaches [25,26]. Regardless of the underlying problem modeling, the proposed learning-enabled framework can work by using any data samples of $\{\mathbf{y}, \mathbf{z}\}$ generated by solving the specific voltage optimization problem.

3. Risk-aware learning

Machine learning (ML) techniques have been recently utilized to attain scalable solutions such that individual nodes can form their own optimal decisions using minimal real-time information. The centralized problem (1) is efficiently solvable but requires a wide deployment of communication resources to connect the DERs. Lacking real-time communications, each node n may resort to a decentralized architecture by forming its optimal z_n from local measurements and possibly limited information elsewhere. Recent work [5,6] has proposed to obtain decentralized decision rules through supervised learning approaches such as kernel learning and neural networks (NNs). With a pre-trained model, these approaches have extremely high computation efficiency during real-time implementation. Nonetheless, most ML-enabled approaches for end-to-end distribution system learning aim to minimize the average losses in predicting the optimal $\mathbf{z}$. This may fall short in reducing the statistical risks of the resultant decision rules. We consider a risk-aware learning framework to address this issue of ML-based DER operations.

Specifically, the key of the ML-based solutions is to obtain the predictive model $\Phi(\mathbf{y}) \rightarrow \mathbf{z}$ from the OC $\mathbf{y}$ to the optimal $\mathbf{z}$, such that it follows a pre-specified *scalable* structure. Considering fully local decision rules for example, we can enforce the model $\{\Phi_n(\cdot)\}_n$, one for each node n, to use local measurements only as the input features; i.e., the nodal prediction is $\hat{z}_n = \Phi_n(\mathbf{y}_n) = \Phi_n([p_n^g, p_n^c, q_n^c])$. Local input $\mathbf{y}_n$ can also include other measurements such as nodal voltage or current magnitude. If certain real-time communications such as broadcasted messages from the feeder head or other key nodes are possible, then $\mathbf{y}_n$ can also include the aggregated power flow or current magnitude in these feeder locations. This ML-enabled framework is very scalable to large networks and flexible to varying communication scenarios.

To construct the individual mapping $\Phi_n(\cdot)$, one can adopt the NN model known for its superior nonlinear approximation capability [27]. As a multi-layer perceptron (MLP), the NN model is basically a layered network with a linear transformation followed by nonlinear activation

per layer. Per node n, using the input to the first layer $\mathbf{y}_n^0$ as a vector embedding of $\mathbf{y}_n$, each layer t is given by

$$\mathbf{y}_n^{t+1} = \sigma(\mathbf{W}_n^t \mathbf{y}_n^t + \mathbf{b}_n^t), \quad \forall t = 0, \ldots, T-1 \tag{6}$$

where $\mathbf{W}_n^t$ and $\mathbf{b}_n^t$ are parameters to be learned, while $\sigma(\cdot)$ is the nonlinear activation such as ReLU. Thus, the training processes is parallelizable as each mapping $\Phi_n(\cdot)$ can be obtained individually. For simplicity, one can set the parameters to be the same for a subset of nodes. For example, all the DER nodes within a feeder, or those DER nodes belonging to the same lateral, can use the same set of parameters. This way, the nodes with the same parameters will be jointly trained by a single MLP. For notational simplicity, the rest of paper presents the individual NN training with $\boldsymbol{\varphi}$ collecting all the learnable parameters $\{\mathbf{W}_n^t, \mathbf{b}_n^t\}$. Hence, there is a one-to-one mapping between the model $\Phi(\cdot)$ and parameter $\boldsymbol{\varphi}$, and the goal becomes to obtain the NN parameter $\boldsymbol{\varphi}$.

Given the scalable structure of $\Phi(\cdot) := \{\Phi_n(\cdot)\}$, a centralized entity can learn $\boldsymbol{\varphi}$ through offline training using K data samples. Each sample k consists of all available measurements in $\mathbf{y}_k$ as the input, and the corresponding optimal $\mathbf{z}_k$ as the output. For simplicity, the rest of paper will use k to index samples. The sample inputs can be from historic measurements, or the latest load forecasting, while the outputs are obtained by solving (1) for each input scenario. To learn $\boldsymbol{\varphi}$, a popular metric is to minimize the average loss in predicting $\mathbf{z}$ over all K samples, as given by

$$\min_{\boldsymbol{\varphi}} f(\boldsymbol{\varphi}) := \frac{1}{K} \sum_{k=1}^{K} \ell\left(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k\right) \tag{7}$$

where $\ell(\cdot)$ denotes the loss function for each sample's predicting value based on the NN parameter $\boldsymbol{\varphi}$. Under the L_2-norm based quadratic loss given by

$$\ell\left(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k\right) = \left\|\Phi(\mathbf{y}_k; \boldsymbol{\varphi}) - \mathbf{z}_k\right\|_2^2,$$

we form the mean-squared error (MSE) of prediction in (7). Other error norms such as the L_1-norm or Huber loss can be used as well. Due to the nonlinearity of Φ wrt $\boldsymbol{\varphi}$, the nonconvex problem (7) is typically minimized through gradient descent iterations that use backpropagation to efficiently compute the gradient. From a statistical perspective, this average loss metric approaches the expected loss if K is large enough. As detailed soon in Remark 2, it does not capture the dispersion of losses or represent the worst-case scenarios such as the maximum loss. As shown by Fig. 1, the tail of the sample loss distribution is not directly dependent on the average value, and thus minimizing the average loss does not guarantee the reduction of worst-case prediction losses. Under a small K or high variability of the samples, the worst-case prediction losses can be very large, even if the average is reasonably small. For the voltage optimization problem, these worst-case scenarios may lead to high mismatch in $\mathbf{q}^g$ prediction or even severe violations of voltage limits.

To tackle this issue, we propose to develop a risk-aware learning approach by including other statistical measures to improve the safety guarantees of the resultant solutions. To quantify the risk of the sample distribution, one possible measure is the *value-at-risk* (VaR), popularly used in finance for portfolio optimization [11,28]. For a given *significance level* $\alpha \in (0, 1)$, the α-VaR represents the threshold value for the $(1-\alpha)$-quantile of a random distribution, as indicated by the blue dashed line of Fig. 1. Hence, reducing the VAR can directly mitigate the worst-case sample losses, but unfortunately is difficult to optimize by using samples due to its lack of smoothness and convexity [11]. Instead, we will consider the *conditional value-at-risk (CVaR)*, a risk measure more widely used as in robust optimization and safe reinforcement learning problems; see e.g., [29–31]. Intuitively, the α-CVaR represents the average over the top αK sample errors, which is thus an upper bound of α-VaR, as shown by Fig. 1. Given $\boldsymbol{\varphi}$, the α-CVaR is analytically formed by all K samples, as

$$\gamma_\alpha(\boldsymbol{\varphi}) := \frac{1}{\alpha K} \sum_{k=1}^{K} \ell(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k) \times \mathbb{1}\{\ell(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k) \geq v\} \tag{8}$$
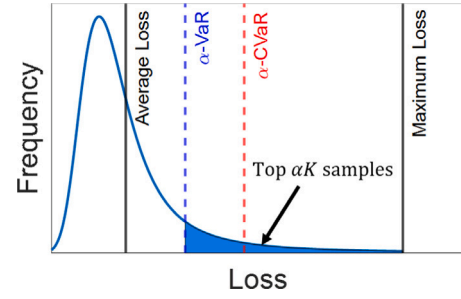

Fig. 1. Illustration of the α-VaR and α-CVaR as compared to average loss.

where v is the α-VaR while $\mathbb{1}(\cdot)$ denotes the indicator function. The CVaR metric can be easily computed after using bisection-typed line search to find v. Interestingly, it is equivalent to the following optimization problem, as shown by [11]

$$\gamma_\alpha(\boldsymbol{\varphi}) := \min_{\beta \in \mathbb{R}} \left\{ \beta + \frac{1}{\alpha K} \sum_{k=1}^{K} \left[\ell\left(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k\right) - \beta \right]_+ \right\} \tag{9}$$

where the positive projection operator $[a]_+ := \max\{0, a\}$. This is because the optimal β to problem (9) turns out to be the α-VaR. The objective function of (9) is convex and piecewise-linear wrt β. Note that if $\ell\left(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k\right)$ is linear wrt $\boldsymbol{\varphi}$, then (9) can be recast as a convex linear program by forming the epigraph problem and expressing $\left\{ \ell\left(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k\right) - \beta \right\}_{k=1}^{K}$ as linear inequality constraints. The convexity property makes CVaR a popular risk measure, while its gradient estimation is also possible. Here, we remark on the generalizability of using sample-based empirical approximation.

Remark 2 (*Empirical Versus Expected*). Both the average loss and CVaR in (7)–(8) are the empirical approximations to their expected-value counterparts, as given by

$$\check{f}(\boldsymbol{\varphi}) = \mathbb{E}_{(\mathbf{y},\mathbf{z})}\left[\ell\left(\Phi(\mathbf{y}; \boldsymbol{\varphi}), \mathbf{z}\right) \right],$$

$$\check{\gamma}_\alpha(\boldsymbol{\varphi}) = \mathbb{E}_{(\mathbf{y},\mathbf{z})}\left[\ell\left(\Phi(\mathbf{y}; \boldsymbol{\varphi}), \mathbf{z}\right) \Big| \ell\left(\Phi(\mathbf{y}; \boldsymbol{\varphi}), \mathbf{z}\right) \geq \check{v} \right]$$

where $\check{v}$ denotes the α-VaR of the underlying distribution. Without knowing the actual distribution, the expected terms are approximated using the sample-based metrics. It is well known such approximation asymptotically approaches the expected value as $K \to \infty$, and similarly for their respective minimizers. For finite K, it is also possible to bound the generalization error, or the difference between the minimizers to empirical and expected losses using the so-termed Rademacher complexity defined over the given samples [32]. More recently, the generalization bounds for the empirical CVaR minimizer have been similarly analyzed in [33], showing a fixed (slightly higher) scaling of the Rademacher complexity. Thus, optimizing the empirical CVaR can achieve guaranteed performance in obtaining the desiderata minimizer to $\check{\gamma}_\alpha(\cdot)$.

In addition to predicting $\mathbf{z}$, we can also represent the risk associated voltage violation wrt the constraint (5). Similarly, we can define the voltage-related CVaR loss as

$$\gamma_\alpha^v(\boldsymbol{\varphi}) := \frac{1}{\alpha K} \sum_{k=1}^{K} |v_n(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}))| \times \mathbb{1}\{|v_n(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}))| \geq v\} \tag{10}$$

where the α-VAR threshold v is constructed by the voltage samples corresponding to the predicted reactive power decision $\Phi(\mathbf{y}_k; \boldsymbol{\varphi})$. Hence, this loss captures the highest voltage deviations based on the $\mathbf{q}^g$ prediction. As voltage depends on the full system input [cf. (2)], minimizing the voltage risk would require the joint training of nodal prediction models $\{\Phi_n(\cdot)\}$.

We consider a general risk-regularized formulation for learning $\Phi(\cdot; \boldsymbol{\varphi})$, as

$$\min_{\boldsymbol{\varphi}} \quad f(\boldsymbol{\varphi}) + \lambda \gamma_\alpha(\boldsymbol{\varphi}) \tag{11}$$

where $\lambda > 0$ is a regularization hyperparameter. The CVaR term can correspond to either the risk of predicting $\mathbf{z}$ in (8) or the voltage risk in (10), or the combination of the two. This risk-aware learning approach also includes the CVaR-only minimization as a special case with $\lambda \to \infty$. Nonetheless, the hyperparameter λ nicely balances between the risk reduction and learning efficiency. Note that the CVaR function in (9) is estimated over αK samples, which is much smaller than K. Accordingly, it can incur high variance of estimation error under a small α value. Hence, purely minimizing CVaR may significantly affect the learning efficiency and potentially lead to much higher expected loss due to the trade-off between mean and variance.

3.1. Accelerated learning via mini-batch selection

To solve the risk-aware learning problem (11), we adopt the mini-batch gradient descent method [13,14], which is widely used in practice for reducing computation complexity and resources. Per iteration i, one can randomly select a subset of samples in $\mathcal{B}^i \subset \{1, \ldots, K\}$, and update the NN parameters according to this mini-batch as

$$\boldsymbol{\varphi}^{i+1} = \boldsymbol{\varphi}^i - \eta_\varphi \left(\nabla f(\boldsymbol{\varphi}^i) + \lambda \nabla \gamma_\alpha(\boldsymbol{\varphi}^i) \right) \tag{12}$$

where η_φ is a positive step-size (or, learning rate) that is chosen to be sufficiently small for convergence. The step-size can also be adaptively learned for better convergence rates, using e.g., the ADAM method [34]. Computing the gradient of the average loss in (7) is the backbone of regular NN training. Similar to the average loss, its gradient can be formed by averaging over the mini-batch $\mathcal{B}^i$, as

$$\nabla f(\boldsymbol{\varphi}^i) = \frac{2}{|\mathcal{B}^i|} \sum_{k \in \mathcal{B}^i} (\Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i) - \mathbf{z}_k)^\top \nabla \Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i), \tag{13}$$

by evaluating the sample gradient $\nabla \Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i)$ using the backpropagation algorithm.

Unfortunately, the gradient for CVaR requires the knowledge of the actual distribution of the sample loss $\ell(\Phi(\mathbf{y}_k; \boldsymbol{\varphi}), \mathbf{z}_k)$ based on $\boldsymbol{\varphi}^i$ [35], with the worst samples depending on $\boldsymbol{\varphi}$ in a non-smooth fashion. Hence, a naive approach by truncating the gradient estimates for the worst samples may not be numerically stable. One possible solution is to use the equivalent CVaR definition in (9), which does not explicitly use the worst samples. This way, β should be included as an optimization variable and updated per iteration i, as well. In addition, by approximating $[a]_+$ using the smooth softplus function $\log(1 + e^a)$ as in [36], one can use the gradient $\nabla_a \log(1 + e^a) = 1/(1 + e^{-a})$ to evaluate

$$\nabla \gamma_\alpha(\boldsymbol{\varphi}^i) = \frac{2}{\alpha |\mathcal{B}^i|} \sum_{k \in \mathcal{B}^i} (\Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i) - \mathbf{z}_k)^\top \nabla \Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i)$$
$$\frac{1}{1 + \exp(-\|\Phi(\mathbf{y}_k; \boldsymbol{\varphi}^i) - \mathbf{z}_k\|_2^2 + \beta^i)}, \tag{14}$$

In addition, the gradient for β^i can be formed to update the auxiliary β variable. Similar to (13), this smooth CVaR gradient update can be implemented using the backpropagation algorithm. In the numerical tests later on, we will directly use the Pytorch library in Python that directly implements backpropagation along with automatic differentiation (AD) to compute the gradient for CVaR loss.

Notably, the use of mini-batch $\mathcal{B}^i$ may make the learning efficiency issue more evident for the computation of CVaR's gradient. This is because by randomly selecting a subset of samples, the worst-case samples may not be evenly represented by every mini-batch. This is a known issue for risk-aware learning. Recently, [37] has proposed an adaptive sampling approach that selects data points more likely to be the worst cases for computing the gradient for CVaR.

Inspired by this idea, we propose a mini-batch selection scheme for accelerating the learning process under CVaR loss. Intuitively, for a

mini-batch $\mathcal{B}^i$ with very small CVaR, it implies that the selected samples in $\mathcal{B}^i$ do not well represent the worst-case scenarios of the full dataset. Therefore, this mini-batch could be disregarded in the gradient descent update for $\boldsymbol{\varphi}$. This mini-batch selection is simple to implement, and yet our numerical tests have shown that it can effectively reduce the number of gradient updates and training time. The algorithmic steps are tabulated in Algorithm 1. In practice, Algorithm 1 is performed sequentially in each epoch while multiple mini-batches can be generated simultaneously. Very recently, CVaR based risk-aware learning has been shown to potentially attain linear convergence rates even for non-convex loss functions [38]. We plan to pursue the design of NN architecture that could improve the convergence rate analysis in future.

Algorithm 1 Mini-batch Selection for Learning (11)

1: **Input:** Full data $\{(\mathbf{y}_k, \mathbf{z}_k)\}_{k=1}^K$, the NN architecture $\Phi(\cdot; \boldsymbol{\varphi})$, the hyper-parameters α, λ, and η_φ, as well as an iteration stopping threshold ϵ.
2: **Output:** The NN parameters in $\boldsymbol{\varphi}$.
3: **Initialize:** Set the iteration index $i = 0$ with the initial $\boldsymbol{\varphi}^0$ and CVaR threshold γ_α.
4: **while** $\|\boldsymbol{\varphi}^i - \boldsymbol{\varphi}^{i-1}\|_2 \geq \epsilon$ **do**
5:     Generate the mini-batch $\mathcal{B}^i$ and compute its CVaR.
6:     **if** CVaR$(\mathcal{B}^i) \geq \gamma_\alpha$ **then**
7:         Update the CVaR threshold $\gamma_\alpha \leftarrow$ CVaR$(\mathcal{B}^i)$.
8:         Calculate the gradients (13)–(14) over $\mathcal{B}^i$ using backpropagation.
9:         Update $\boldsymbol{\varphi}^{i+1}$ as in (12).
10:         Update $i \leftarrow i + 1$.
11:    **end if**
12: **end while**
13: **Return:** the NN model in $\Phi(\cdot; \boldsymbol{\varphi}^i)$

---

4. Numerical validations

We have tested on the single-phase equivalent of the IEEE 123-bus test case [39] to demonstrate the effectiveness of the proposed risk-aware learning algorithm in mitigating the risks of $\mathbf{q}^g$ prediction and voltage violation. The test system consists of 90 load nodes with nodes 66, 85, 96, 114, 151, and 250 equipped with inverter-based PV generation. Real-world active power data for PV output $\mathbf{p}^g$ and non-PV loads $\mathbf{p}^c$ at the minute-level resolution has been obtained from the Pecan Street Dataport [40]. The reactive power $\mathbf{q}^c$ for the loads was synthetically generated by randomly selecting a power factor in the range [0.9, 0.95]. The optimal reactive power $\mathbf{z}$ in (1) has been solved for each sample system OC $\mathbf{y}$ using the MATLAB® R2020b. 10 days of data have been used to generate 14,400 samples. Samples from the first 8 days are used to train the NN models in a batch setting, from which each mini-batch is randomly generated. The remaining 2 days are used to test the trained models in predicting the reactive power decisions from nodal measurements.

To attain scalable decision rules, all six PV nodes use the same NN model with the input $\mathbf{y}_n$ consisting of p/q data locally and from the feeder head's broadcast. To allow for efficient training, a simple graph NN model with the graph filter being an identity matrix (e.g., no information sharing among nodes) [41] and `relu` activation function has been used to set up $\Phi(\cdot)$ for this local architecture.

To evaluate the performance of the proposed algorithm, we have considered three loss objectives: (i) the (risk-neutral) MSE-only loss (7) for predicting $\mathbf{q}^g$; (ii) the risk-regularized MSE (11) with CVaR on the error in predicting $\mathbf{q}^g$, denoted as CVaR(qg); (iii) and the risk-regularized MSE with CVaR on both $\mathbf{q}^g$ prediction error and voltage deviation, denoted as CVaR(qg,dv). The CVaR parameter $\alpha = 0.2$ has been picked. For both CVaR-based risk-aware objectives, we have compared Algorithm 1 with the default mini-batch based algorithm.
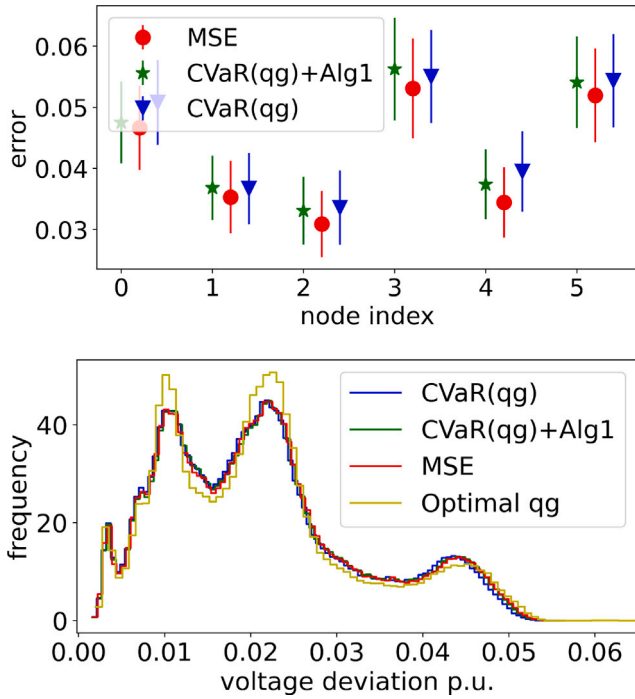
Fig. 2. Test Case 1: Comparisons of the three NN training approaches in terms of (top) nodal error in predicting q_n^g (mean $\pm$ standard deviation) and (bottom) the distribution of voltage deviations based on the training results.

Table 1
Computation time and error performance in Test Case 1.

Loss obj.	Epoch [s]	Total [s]	q^g error	Max $\mathbf{v}$
MSE	0.52	46.48	6.62%	5.65%
CVaR(qg)	1.07	38.70	6.78%	5.57%
CVaR(qg)+Alg 1	0.61	35.63	6.97%	5.64%

These algorithms have been implemented using the PyTorch library and tested on Google Colaboratory using the NVIDIA Tesla P100 GPU for training acceleration. All models converged in the training process, and the same convergence criterion for the loss is applied for all test cases. All results presented here have used the test set to generate the error performance.[1]

Test Case 1: We first compare the training objectives (i) MSE and (ii) CVaR(qg) to demonstrate the benefits of incorporating the CVaR loss for predicting $\mathbf{q}^g$, while the CVaR(qg) has also been implemented by the proposed Algorithm 1. Fig. 2 and Table 1 list the training time and test error performances. Fig. 2 plots the nodal prediction error (top) and the corresponding distribution of voltage deviations (bottom). In general, the error performance in both $\mathbf{q}^g$ prediction and voltage deviation is very close among the three methods, with the voltage distribution very similar to that of the optimal decisions. The CVaR loss has slightly increased the prediction error, also confirmed by the percentage prediction error and maximum voltage deviation in Table 1. This change is because the task of predicting $\mathbf{q}^g$ has been very accurate using the input features to the NN training. Interestingly, although the CVaR loss increases the average computation time per epoch, Table 1 shows that this regularization actually speeds up the overall training process as comparing to the MSE loss. More importantly, the proposed Algorithm 1 has attained the expected CVaR error performance while reducing the computation time for each epoch (by over 40%) and the
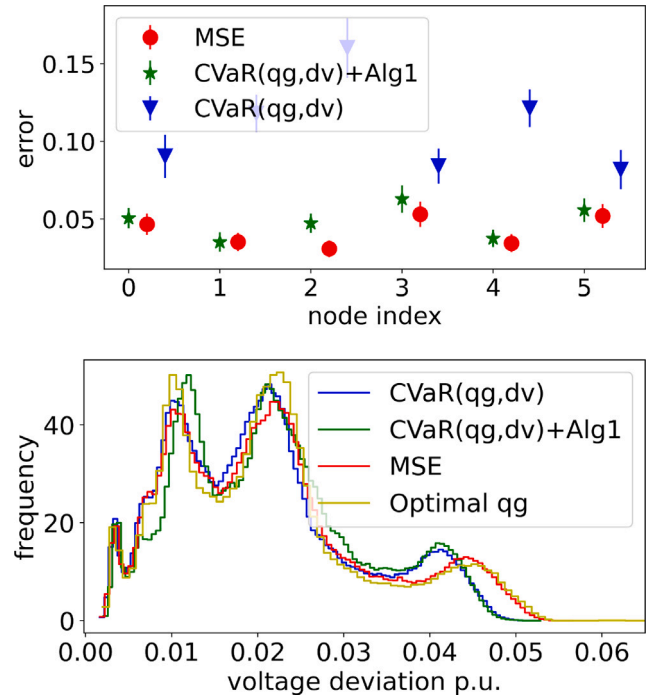

Fig. 3. Test case 2 with CVaR regularization on both $\mathbf{q}^g$ and voltage deviation $\mathbf{v}$: Comparison of the NN-based decentralized controller using pure MSE loss, the CVaR-regularized formulation as well as the CVaR-regularized controller trained using CVaR-based mini-batch selection algorithm on $\mathbf{q}^g$ in terms of (top figure) predicting $\mathbf{q}^g$ (mean $\pm$ standard deviation) and (bottom) the resulting histogram of voltage deviations.

Table 2
Computation time and error performance in Test Case 2.

Loss obj.	Epoch [s]	Total [s]	q^g error	Max $\mathbf{v}$
MSE	0.54	44.89	4.21%	5.65%
CVaR(qg,dv)	0.77	31.73	10.93%	5.27%
CVaR(qg,dv)+Alg 1	0.51	25.93	4.82%	5.28%

total training time. Thus, the proposed Algorithm 1 has attained faster learning speed for predicting $\mathbf{q}^g$.

Test Case 2: We further incorporate the CVaR loss associated with voltage deviation by comparing the training objectives (i) MSE and (iii) CVaR(qg,dv). Fig. 3 and Table 2 list the updated training time and test error performance for this comparison. Notably, Fig. 3 demonstrates that the CVaR loss introduces additional difficulty to optimize, leading to higher nodal prediction error over the MSE one. Thanks to the proposed acceleration scheme, Algorithm 1 has significantly mitigated the error bias issue by using more statistically important mini-batches. More importantly, using the additional voltage-risk (10), the proposed method can reduce the maximum voltage deviation by around 7% over the MSE one, as shown in Table 2. This is more evident in Fig. 3 where the worst-case voltage deviations (over 0.05) are effectively reduced in frequency thanks to the CVaR regularization on the worst-case voltage. Thus, the voltage-based CVaR metric is especially useful for improving the safety of the resultant decision rules. In addition, the training time improvement is more significant in this test case, with the total training time reduced by over 40% from the MSE one. Hence, the proposed risk-constrained learning framework for designing decentralized controllers has shown to be effective in attaining safe decision making for DERs to perform voltage optimization. To sum up, the proposed CVaR regularization and mini-batch selection scheme can effectively improve the training speed, while incorporating the voltage risk can help mitigate the worst-case voltage deviations attained by the NN-based decision rules.

5. Conclusions and future work

This paper developed a risk-aware learning framework for attaining scalable decision rules in the distribution grid voltage optimization problem. For learning the optimal reactive power decision rules using local data, we propose to account for the worst-case scenarios by considering the CVaR losses associated with prediction error and voltage deviation. To solve the resultant risk-regularized problem, we develop a mini-batch gradient descent algorithm by judiciously selecting the mini-batches to accelerate the training process. Numerical tests using real-world data have demonstrated the training accelerations by using the proposed mini-batch selection algorithm. In addition, the benefits of using voltage-associated risk have been validated in terms of mitigating the worst-case voltage deviations.

Several interesting future directions open up for this work. We are currently investigating the convergence properties of optimizing CVaR loss. In addition, it is interesting to incorporate the graph structure of distribution grids to generalize the scalable NN architecture and to investigate the effects of system topology on our proposed approach. Last, the proposed risk-aware learning framework can be extended to optimize active power resources and also dynamical DERs such as energy storage.

CRediT authorship contribution statement

Shanny Lin: Methodology, Software, Writing – original draft. **Shaohui Liu:** Methodology, Software, Validation. **Hao Zhu:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces, IEEE Std 1547-2018, Tech. Rep., 2018, pp. 1–138.

[2] D.K. Molzahn, F. Dörfler, H. Sandberg, S.H. Low, S. Chakrabarti, R. Baldick, J. Lavaei, A survey of distributed optimization and control algorithms for electric power systems, IEEE Trans. Smart Grid 8 (6) (2017) 2941–2962.

[3] K.E. Antoniadou-Plytaria, I.N. Kouveliotis-Lysikatos, P.S. Georgilakis, N.D. Hatziargyriou, Distributed and decentralized voltage control of smart distribution networks: Models, methods, and future research, IEEE Trans. Smart Grid 8 (6) (2017) 2999–3008.

[4] H.J. Liu, W. Shi, H. Zhu, Distributed voltage control in distribution networks: Online and robust implementations, IEEE Trans. Smart Grid 9 (6) (2017) 6106–6117.

[5] M. Jalali, V. Kekatos, N. Gatsis, D. Deka, Designing reactive power control rules for smart inverters using support vector machines, IEEE Trans. Smart Grid 11 (2) (2020) 1759–1770.

[6] S. Gupta, V. Kekatos, M. Jin, Controlling smart inverters using proxies: A chance-constrained DNN-based approach, 2021, arXiv preprint arXiv:2105.00429.

[7] Q. Yang, G. Wang, A. Sadeghi, G.B. Giannakis, J. Sun, Two-timescale voltage control in distribution grids using deep reinforcement learning, IEEE Trans. Smart Grid 11 (3) (2020) 2313–2323.

[8] R. Dobbe, O. Sondermeijer, D. Fridovich-Keil, D. Arnold, D. Callaway, C. Tomlin, Toward distributed energy services: Decentralizing optimal power flow with machine learning, IEEE Trans. Smart Grid 11 (2) (2020) 1296–1306.

[9] D. Cao, J. Zhao, W. Hu, N. Yu, F. Ding, Q. Huang, Z. Chen, Deep reinforcement learning enabled physical-model-free two-timescale voltage control method for active distribution systems, IEEE Trans. Smart Grid (2021) 1.

[10] A. Venzke, G. Qu, S. Low, S. Chatzivasileiadis, Learning optimal power flow: Worst-case guarantees for neural networks, in: 2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), 2020, pp. 1–7.

[11] R.T. Rockafellar, S. Uryasev, et al., Optimization of conditional value-at-risk, J. Risk 2 (2000) 21–42.

[12] R. Nellikkath, S. Chatzivasileiadis, Physics-informed neural networks for AC optimal power flow, 2021, arXiv preprint arXiv:2110.02672.

[13] A. Nemirovski, A. Juditsky, G. Lan, A. Shapiro, Robust stochastic approximation approach to stochastic programming, SIAM J. Optim. 19 (4) (2009) 1574–1609.

[14] S. Ghadimi, G. Lan, H. Zhang, Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization, Math. Program. 155 (1–2) (2016) 267–305.

[15] H. Zhu, H.J. Liu, Fast local voltage control under limited reactive power: Optimality and stability analysis, IEEE Trans. Power Syst. 31 (5) (2016) 3794–3803.

[16] M. Yao, I.A. Hiskens, J.L. Mathieu, Mitigating voltage unbalance using distributed solar photovoltaic inverters, IEEE Trans. Power Syst. (2020).

[17] K. Girigoudar, L.A. Roald, On the impact of different voltage unbalance metrics in distribution system optimization, Electr. Power Syst. Res. 189 (2020) 106656.

[18] M. Baran, F. Wu, Network reconfiguration in distribution systems for loss reduction and load balancing, IEEE Trans. Power Deliv. 4 (2) (1989) 1401–1407.

[19] P.L. Donti, Y. Liu, A.J. Schmitt, A. Bernstein, R. Yang, Y. Zhang, Matrix completion for low-observability voltage estimation, IEEE Trans. Smart Grid 11 (3) (2019) 2520–2530.

[20] J. Song, E. Dall'Anese, A. Simonetto, H. Zhu, Dynamic distribution state estimation using synchrophasor data, IEEE Trans. Smart Grid 11 (1) (2019) 821–831.

[21] H. Xu, A.D. Domínguez-García, V.V. Veeravalli, P.W. Sauer, Data-driven voltage regulation in radial power distribution systems, IEEE Trans. Power Syst. 35 (3) (2019) 2133–2143.

[22] Z. Tang, D.J. Hill, T. Liu, Fast distributed reactive power control for voltage regulation in distribution networks, IEEE Trans. Power Syst. 34 (1) (2018) 802–805.

[23] S. Taheri, M. Jalali, V. Kekatos, L. Tong, Fast probabilistic hosting capacity analysis for active distribution systems, IEEE Trans. Smart Grid (2020).

[24] A. Bernstein, C. Wang, E. Dall'Anese, J.-Y. Le Boudec, C. Zhao, Load flow in multiphase distribution networks: Existence, uniqueness, non-singularity and linear models, IEEE Trans. Power Syst. 33 (6) (2018) 5832–5843.

[25] E. Dall'Anese, H. Zhu, G.B. Giannakis, Distributed optimal power flow for smart microgrids, IEEE Trans. Smart Grid 4 (3) (2013) 1464–1475.

[26] S.H. Low, Convex relaxation of optimal power flow—Part I: Formulations and equivalence, IEEE Trans. Control Netw. Syst. 1 (1) (2014) 15–27.

[27] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[28] N. Larsen, H. Mausser, S. Uryasev, Algorithms for optimization of value-at-risk, in: Financial Engineering, E-Commerce and Supply Chain, Springer, 2002, pp. 19–46.

[29] V. Gabrel, C. Murat, A. Thiele, Recent advances in robust optimization: An overview, European J. Oper. Res. 235 (3) (2014) 471–483.

[30] Y. Chow, A. Tamar, S. Mannor, M. Pavone, Risk-sensitive and robust decision-making: a CVaR optimization approach, in: C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, R. Garnett (Eds.), Proc. Advances in Neural Information Processing Systems, Vol. 28, 2015.

[31] A.R. Cardoso, H. Xu, Risk-averse stochastic convex bandit, in: Proc. 22nd International Conference on Artificial Intelligence and Statistics, 2019, pp. 39–47.

[32] P.L. Bartlett, S. Mendelson, Rademacher and Gaussian complexities: Risk bounds and structural results, J. Mach. Learn. Res. 3 (Nov) (2002) 463–482.

[33] J. Lee, S. Park, J. Shin, Learning bounds for risk-sensitive learning, 2020, arXiv preprint arXiv:2006.08138.

[34] T. Goldstein, M. Li, X. Yuan, Adaptive primal-dual splitting methods for statistical learning and image processing, in: Proc. Advances in Neural Information Processing Systems, 2015, pp. 2089–2097.

[35] A. Tamar, Y. Glassner, S. Mannor, Optimizing the CVaR via sampling, in: Proc. of the AAAI Conference on Artificial Intelligence, Vol. 29, 2015, no. 1.

[36] T. Tarnopolskaya, Z. Zhu, CVaR-minimising hedging by a smoothing method, ANZIAM J. 52 (2010) C237–C256.

[37] S. Curi, K.Y. Levy, S. Jegelka, A. Krause, Adaptive sampling for stochastic risk-averse learning, in: Proc. Advances in Neural Information Processing Systems 33 (NeurIPS 2020), Vol. 33, 2020, pp. 1036–1047.

[38] D.S. Kalogerias, Noisy linear convergence of stochastic gradient descent for CV@R statistical learning under polyak- ojasiewicz conditions, 2020, arXiv preprint arXiv:2012.07785.

[39] IEEE Distribution Test Feeders, IEEE Power & Energy Society, [Online]. Available: https://site.ieee.org/pes-testfeeders/resources/.

[40] PecanStreet Dataport, Pecan Street Inc., [Online]. Available: https://www.pecanstreet.org/dataport/.

[41] S. Liu, C. Wu, H. Zhu, Graph neural networks for learning real-time prices in electricity market, in: Proc. Tackling Climate Change with Machine Learning Workshop, International Conference on Machine Learning (ICML), 2021.