Fast and Reliable $N - k$ Contingency Screening with Input-Convex Neural Networks

Nicolas Christianson NCHRISTI@CALTECH.EDU
Wenqi Cui WENQICUI@CALTECH.EDU
Steven Low SLOW@CALTECH.EDU
California Institute of Technology, Pasadena, CA, USA

Weiwei Yang WEIWEI.YANG@MICROSOFT.COM
Microsoft Research, Redmond, WA, USA

Baosen Zhang ZHANGBAO@ECE.UW.EDU
University of Washington, Seattle, WA, USA

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

Power system operators must ensure that dispatch decisions remain feasible in case of grid outages or contingencies to prevent cascading failures and ensure reliable operation. However, checking the feasibility of all $N - k$ contingencies – every possible simultaneous failure of k grid components – is computationally intractable even for small k, requiring system operators to resort to heuristic screening methods. Because of the increase in uncertainty and changes in system behaviors, heuristic lists might not include all relevant contingencies, generating false negatives in which unsafe scenarios are misclassified as safe. In this work, we propose to use input-convex neural networks (ICNNs) for contingency screening. We show that ICNN reliability can be determined by solving a convex optimization problem, and by scaling model weights using this problem as a differentiable optimization layer during training, we can learn an ICNN classifier that is both data-driven and has provably guaranteed reliability. That is, our method can ensure a zero false negative rate. We empirically validate this methodology in a case study on the IEEE 39-bus test network, observing that it yields substantial (10-20 $\times$) speedups while having excellent classification accuracy.

Keywords: Contingency analysis, reliable machine learning, differentiable convex optimization

1. Introduction

Power systems face increasing uncertainty due to increasing variable renewable generation and environmental factors such as extreme weather events and wildfires. To ensure reliable operation in the face of this growing uncertainty, power system operators must dispatch generation resources in a manner that anticipates and is robust to potential asset outages, such as the failure of a transmission line (Enns et al., 1982; Nakiganda et al., 2023). Failing to anticipate and prepare for such outages can lead to cascading failures that may necessitate load shedding, as occurred in the Texas blackouts in 2021 (Busby et al., 2021).

To assess and plan for the impacts of potential asset failures before they occur, system operators must perform contingency analysis to identify which failures will result in a post-failure operating point that is infeasible (Bienstock, 2015, Chapter 3). In particular, NERC regulations mandate that US power systems remain stable for all $N - 1$ contingencies – contingencies involving the loss of a single asset – and that system operators plan for the multi-element contingencies with the most severe consequences (NERC, 2010). Assessing the security of and planning for such $N - k$ contingencies – simultaneous losses of $k > 1$ assets – is crucial for reliable operation, as such correlated failures can cause severe blackouts, such as the 2003 Northeast blackout (Force, 2004). However, the complexity of contingency analysis grows exponentially in the number of simultaneous failures k that is considered: in a system with N components, the number of such contingencies is $\Omega(N^k)$, which quickly becomes intractable for $k > 1$ in large-scale power systems (Strbac et al., 2016).

To combat this intractability and enable efficient screening of $N - k$ contingencies for $k > 1$, several approaches have been developed in the recent literature. These methods fall into one of two categories: (1) heuristic approaches using, e.g., machine learning to predict contingency feasibility (Davis and Overbye, 2011; Crozier et al., 2022; Nakiganda et al., 2023), and (2) exact methods that reduce computational expense by certifying feasibility of a collection of contingencies using "representative" constraints (Jiachun Guo et al., 2009; Kaplunovich and Turitsyn, 2013, 2016; Yang et al., 2017). However, these methods fall short on two fronts. The heuristic approaches (1) come with no rigorous guarantees on prediction accuracy or reliability; thus, they might misclassify a critical contingency as feasible, causing system outages. On the other hand, the exact methods (2), while reliable, are typically hand-designed rules which cannot take advantage of historical data to accelerate contingency analysis by focusing on the most relevant contingencies for a particular power system. To enable reliable and efficient screening of higher-order $N - k$ contingencies in modern power systems, new approaches are needed to bridge the data-driven paradigm of machine learning with the strong reliability guarantees of exact methods.

1.1. Contributions

In this work, we confront this challenge, proposing a machine learning approach to screening $N-k$ contingencies that is fast, data-driven, and comes with *provable* reliability guarantees. In particular, we propose to use *input-convex neural networks* (ICNNs) to screen arbitrary collections of contingencies for infeasibility. We define a *reliable* classifier as one that never misclassifies an infeasible contingency as feasible – that is, one that makes no false negative predictions – and we show that ICNN reliability can be certified by solving a collection of convex optimization problems (Proposition 2). Furthermore, we show that an unreliable ICNN can be transformed into a reliable one with zero false negative rate by suitably scaling model parameters by the solution to a convex optimization problem (Theorem 3), and we propose a training methodology that enables learning over the restricted set of *provably reliable ICNNs* by applying this scaling during training via a differentiable convex optimization layer (Theorem 4, Algorithm 1). This fully differentiable approach ensures that the scaling procedure and its dependence on model parameters are accounted for during gradient descent updates; see Figure 1 for a diagram outlining this approach. We detail our contributions in the broader context of related work on contingency analysis and machine learning in the longer version of this paper (Christianson et al., 2024).

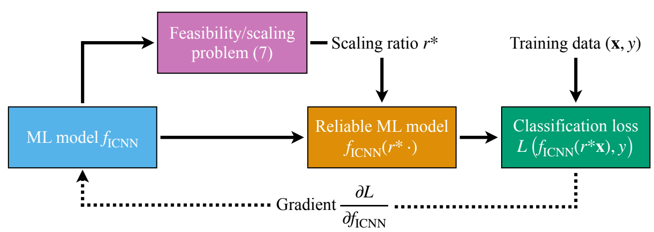

Figure 1: A schematic of our proposed methodology for training reliable classifiers for contingency screening; see Algorithm 1 for details.

Our approach allows for **trading off the online computational burden of contingency screening for an offline one**: it requires a significant computational investment during the training procedure to guarantee model reliability, but at deployment time, screening for contingencies only requires a single feedforward pass of the ICNN. We test our approach in a case study on the IEEE 39-bus test system, finding that it yields significant (10-20 $\times$) speedups in runtime while ensuring zero false negative rate and excellent (2-5%) false positive rate (Section 5.1). In addition, our approach yields an ICNN parametrizing an inner approximation to the set of network injections that are feasible across contingencies, which enables $10\times$ faster preventive dispatch via security-constrained optimal power flow (Section 5.2). We anticipate that our proposed approach to learning efficient data-driven inner approximations to complex feasible sets using ICNNs could be of broader interest for other applications in energy systems and control.

1.2. Notation

Let $\mathbb{R}_+$ denote the nonnegative reals. Given a vector $\mathbf{x} \in \mathbb{R}^n$, we denote its ith entry x_i; similarly, given a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, its ith row is denoted $\mathbf{m}_i$ and its ijth entry is denoted m_{ij}. Given $n \in \mathbb{N}$, we define $[n] = \{1, \ldots, n\}$, and given a set $\mathcal{X}$, we define $\mathcal{P}(\mathcal{X})$ as its power set. Given a set $\mathcal{A} \subseteq \mathbb{R}^n$, $\operatorname{int}\mathcal{A}$ denotes its interior and $\operatorname{vol}(\mathcal{A})$ denotes its volume.

2. Model and Preliminaries

We begin by reviewing power network economic dispatch via the DC-optimal power flow problem and the problem of screening for infeasible contingencies. We then describe our classification approach to contingency screening and the *input-convex* neural networks we employ to this end.

2.1. DC-OPF and Contingency Screening

Consider a power network represented by a graph $G = (V, E)$, where V is the set of buses and E is the set of transmission lines. Let $n = |V|$ be the number of buses and $m = |E|$ be the number of lines. Without loss of generality, we assume that each bus $i \in [n]$ has a single generator.

To dispatch generation while minimizing cost and satisfying demand and other constraints in large-scale transmission networks, system operators typically solve the DC-optimal power flow (OPF) problem, which considers a linearized model of power flow (Stott et al., 2009). In this problem, the system operator is faced with a known vector $\mathbf{d} \in \mathbb{R}^n$ of (net) demands across buses, and chooses generator dispatches $\mathbf{p} \in \mathbb{R}^n$ to minimize cost while satisfying several constraints:

$$
\begin{aligned}
\min_{\mathbf{p}\in\mathbb{R}^n} \quad & \sum_{i\in[n]} c_i(p_i) && \text{(1a)}\\
\text{s.t.} \quad & \underline{\mathbf{p}} \le \mathbf{p} \le \overline{\mathbf{p}} && \text{(1b)}\\
& \mathbf{1}^\top(\mathbf{p}-\mathbf{d}) = 0 && \text{(1c)}\\
& \underline{\mathbf{f}} \le \mathbf{H}(\mathbf{p}-\mathbf{d}) \le \overline{\mathbf{f}} && \text{(1d)}
\end{aligned}
$$

Here, $c_i(p_i)$ is the cost for the generation p_i on generator i, the constraint (1b) enforces lower and upper limits $\underline{\mathbf{p}}, \overline{\mathbf{p}} \in \mathbb{R}^n$ on generation, (1c) enforces supply-demand balance, and (1d) enforces the bounds $\underline{\mathbf{f}}, \overline{\mathbf{f}} \in \mathbb{R}^m$ on line power flows given the nodal net injection vector $\mathbf{p} - \mathbf{d}$. The matrix $\mathbf{H}$ mapping from nodal net power injections to line power flows is specifically defined as $\mathbf{H} := \mathbf{B}\mathbf{C}^\top\mathbf{L}^\dagger$, where $\mathbf{B} \in \mathbb{R}^{m\times m}$ is the diagonal matrix of line admittances, $\mathbf{C} \in \mathbb{R}^{n\times m}$ is a bus-by-line directed incidence matrix, and $\mathbf{L} = \mathbf{C}\mathbf{B}\mathbf{C}^\top$ is the admittance-weighted network Laplacian.

In the DC-OPF problem (1), the system operator solves for a feasible dispatch vector given a nominal network topology G. In practice, however, after a dispatch decision is chosen and the net nodal power injections $\mathbf{x} := \mathbf{p} - \mathbf{d}$ are fixed, the network topology might change due to the failure of one or more lines. As a result of this contingency, the matrix $\mathbf{H}$ mapping net power injections to line power flows will change, causing the line flows to redistribute and potentially violate the line flow limits (1d). Such violations may cause further lines to trip, causing a cascade of failures (Bienstock, 2015, Chapter 4). Thus, to ensure continued feasible and reliable operation, the system operator must determine which contingencies are infeasible and must be planned for. This is the *contingency analysis* problem, which is defined formally as follows.[1]

Problem 0 (Contingency Analysis) *Let $\mathcal{C} \subseteq \mathcal{P}([m])$ be a set of contingencies of interest, where each $c \in \mathcal{C}$ represents a set of failed lines, and let $\mathbf{x} = \mathbf{p} - \mathbf{d} \in \mathbb{R}^n$ be a vector of nodal net power injections. In the* ***contingency analysis*** *problem, the system operator seeks to determine whether $\mathbf{x}$ yields feasible line flows for each contingency $c \in \mathcal{C}$ – that is, whether $\underline{\mathbf{f}} \leq \mathbf{H}_c\mathbf{x} \leq \overline{\mathbf{f}}$ for each $c \in \mathcal{C}$, where $\mathbf{H}_c = \mathbf{B}_c\mathbf{C}_c^\top\mathbf{L}_c^\dagger$ is defined for the post-contingency network topology with lines $E \setminus c$.*

A standard choice for the set of reference contingencies $\mathcal{C}$ is the collection of all $N-k$ contingencies, or the set of all possible simultaneous failures of up to k lines; in this case, $\mathcal{C} = \{c \subseteq [m] : 1 \leq |c| \leq k\}$. In practice, however, it is impractical to check the feasibility of all possible $N-k$ contingencies in real time for even moderately small k: in a network with m lines, there are $\Omega(m^k)$ such possible contingencies, and so the complexity of $N-k$ contingency analysis grows exponentially with k. Instead, system operators typically only consider the $N-1$ case, augmented with a small number of representative or problematic higher-order contingencies selected via heuristic methods. Such heuristics work well most of the time, since typically only a small number of contingencies are likely either to occur or to cause system infeasibility. However, they give no guarantees on system (in)feasibility for the broader set of possible $N-k$ contingencies for $k > 1$.

In this work, we seek to develop methods that can efficiently check whether a net injection $\mathbf{x}$ is feasible for *all* contingencies in some general, large reference set $\mathcal{C}$, such as the set of all $N-k$ contingencies for $k > 1$. To this end, we introduce the *contingency screening* problem as a coarse-grained version of the contingency analysis problem.

Problem 1 (Contingency Screening) *Let $\mathcal{C} \subseteq \mathcal{P}([m])$ be a set of contingencies of interest, and let $\mathbf{x} \in \mathbb{R}^n$ be a vector of nodal net power injections. In the* ***contingency screening*** *problem, the system operator seeks to determine whether the net injection $\mathbf{x}$ is feasible for* ***all*** *contingencies $c \in \mathcal{C}$ – that is, whether $\mathbf{x}$ is in the* ***feasible region*** *$\mathcal{F}_\mathcal{C} := \left\{\mathbf{y} \in \mathbb{R}^n : \underline{\mathbf{f}} \leq \mathbf{H}_c\mathbf{y} \leq \overline{\mathbf{f}} \quad \forall c \in \mathcal{C}\right\}$, where each $\mathbf{H}_c = \mathbf{B}_c\mathbf{C}_c^\top\mathbf{L}_c^\dagger$ is defined for the post-contingency network topology with lines $E \setminus c$.*

The (true) feasible region $\mathcal{F}_\mathcal{C}$ is the set of all net injections which remain feasible under any contingency in the set $\mathcal{C}$. For notational convenience, we henceforth write this set abstractly as

$$\mathcal{F}_\mathcal{C} := \{\mathbf{y} \in \mathbb{R}^n : \mathbf{A}\mathbf{y} \leq \mathbf{b}\}\,, \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{2m|\mathcal{C}| \times n}$ and $\mathbf{b} \in \mathbb{R}^{2m|\mathcal{C}|}$ collect all of the constraints $\underline{\mathbf{f}} \leq \mathbf{H}_c\mathbf{y} \leq \overline{\mathbf{f}}$. We will assume that $\mathbf{A}$ contains no zero rows, since these would encode vacuous constraints. We will also make the following mild assumptions on the structure of $\mathcal{F}_\mathcal{C}$.

Assumption 1 *$\mathcal{F}_\mathcal{C}$ is a strict subset of $\mathbb{R}^n$ whose interior contains the origin: $\mathcal{F}_\mathcal{C} \subsetneq \mathbb{R}^n$ and $\mathbf{0} \in \operatorname{int}\mathcal{F}_\mathcal{C}$. Equivalently, $\mathbf{A}$ has at least one row and $\mathbf{A}\mathbf{0} < \mathbf{b}$.*

1. Given a change in network topology resulting from a contingency, infeasibility could arise in either the line flow limits (1d) or the supply-demand balance constraint (1c); the latter is possible only in the case of *islanding* contingencies which disconnect the network into multiple connected components. Because the set of islanding contingencies can be determined in advance, in this work we will restrict our focus only to the set of non-islanding contingencies and the feasibility of the line limits (1d).

Note that these assumptions are reasonable: the first simply means that $\mathcal{F}_\mathcal{C}$ encodes *some* constraint; if it doesn't, then there is no need to perform contingency screening. The second assumption amounts to the condition that the lower and upper line limits $\underline{\mathbf{f}}, \overline{\mathbf{f}}$ are bounded away from zero, which should hold in practice.

We can frame contingency *screening* as a binary classification problem where one seeks to classify a net injection vector $\mathbf{x} \in \mathbb{R}^n$ as feasible or infeasible, and true labels are given by the indicator function $f_\mathcal{C}$ where $f_\mathcal{C}(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathcal{F}_\mathcal{C}$ ($\mathbf{x}$ is feasible) and $f_\mathcal{C}(\mathbf{x}) = 1$ otherwise ($\mathbf{x}$ is infeasible). While at first glance this might appear to be a simpler problem than the full contingency analysis problem, determining whether some injection $\mathbf{x} \in \mathcal{F}_\mathcal{C}$ (i.e., computing the label $f_\mathcal{C}(\mathbf{x})$) still has complexity $\Omega(m|\mathcal{C}|)$ in general, as it requires checking the feasibility of each contingency in $\mathcal{C}$. This feasibility verification is tractable given sufficient time, but it will generally take too long for real-time operation when the network and contingency set are large. If approximations suffice, we could instead use techniques from machine learning to learn a more computationally efficient approximation of the function $f_\mathcal{C}$ in a data-driven fashion using, e.g., neural networks; however, this computational speedup will typically come at the expense of reduced classification accuracy.

In particular, a generic machine learning classifier might suffer *false negatives*, where it classifies injections as feasible when they are not. While false positives (misclassifying a feasible injection as infeasible) may simply cause increased caution, false negatives pose a serious threat to reliable power system operation, since an infeasible injection that is not identified as such could lead to a cascade of failures. To confidently deploy machine learning methods to contingency screening, they should ideally avoid any false negative predictions; we call such a classifier *reliable*.

Definition 1 *A classifier $f : \mathbb{R}^n \to \{0,1\}$ for the contingency screening problem (Problem 1) is said to be* ***reliable*** *if it has zero false negative rate, i.e., if $f(\mathbf{x}) = 0$ implies $\mathbf{x} \in \mathcal{F}_\mathcal{C}$ for any $\mathbf{x} \in \mathbb{R}^n$.*

Note that a reliable classifier f is exactly one whose *predicted feasible region* $\{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) = 0\}$ is contained inside the true feasible region $\mathcal{F}_\mathcal{C}$; that is, the predicted feasible region should be an inner approximation of the true feasible region. Our goal in this work is to develop an approach for training reliable ML classifiers for contingency screening that satisfy this property. For general machine learning models and classification problems, determining whether this containment property holds is not typically tractable. However, as we will see in Section 3, the convex polyhedral structure of the true feasible region $\mathcal{F}_\mathcal{C}$ enables tractably answering this question when we restrict to a class of *convex* neural networks.

2.2. Input-Convex Neural Networks

Input-convex neural networks (ICNNs) (Amos et al., 2017) are a restricted class of neural networks that parametrize convex functions. We consider feed-forward ICNNs $f_{\text{ICNN}} : \mathbf{x} \mapsto \mathbf{y}$ with k hidden layers of the form

$$\begin{aligned} \mathbf{z}_1 &= \operatorname{ReLU}(\mathbf{D}_1\mathbf{x} + \mathbf{b}_1) \\ \mathbf{z}_i &= \operatorname{ReLU}(\mathbf{W}_{i-1}\mathbf{z}_{i-1} + \mathbf{D}_i\mathbf{x} + \mathbf{b}_i) \qquad \text{for } i = 2, \dots, k \\ \mathbf{y} &= \mathbf{W}_k\mathbf{z}_k + \mathbf{D}_{k+1}\mathbf{x} + \mathbf{b}_{k+1}, \end{aligned} \tag{3}$$

where $\mathbf{z}_i$ is the ith hidden layer, the intermediate activation function is $\operatorname{ReLU}(x) = \max\{x, 0\}$, and the the weight matrices $\mathbf{W}_i$ are all assumed to have nonnegative entries (the weights $\mathbf{D}_i$ can have arbitrary entries). It is relatively straightforward to see that, under these assumptions (and more generally in the case of convex, nondecreasing activation functions), $f_{\text{ICNN}}(\mathbf{x})$ is convex in $\mathbf{x}$ (Amos et al., 2017, Proposition 1). Moreover, given sufficient depth and width, ICNNs can approximate *any* convex function arbitrarily well (Chen et al., 2019, Theorem 1).

In the remainder of this work, for our application to the contingency screening problem, we will consider ICNNs with input dimension n and output dimension 1. When using an ICNN to classify the feasibility of an injection $\mathbf{x}$, we will take its prediction to be $\sigma(f_{\text{ICNN}}(\mathbf{x}))$, where $\sigma(x) = (1 + e^{-x})^{-1}$ is a sigmoidal activation applied to the output of the ICNN. In this case, predictions less than 0.5 will correspond to a "feasible" classification (0), and those strictly greater than 0.5 will correspond to "infeasible" (1). With this setup, one readily observes that the predicted feasible region of an ICNN is exactly its 0-sublevel set, i.e., $\{\mathbf{x} \in \mathbb{R}^n : \sigma(f_{\text{ICNN}}(\mathbf{x})) \leq 0.5\} = \{\mathbf{x} \in \mathbb{R}^n : f_{\text{ICNN}}(\mathbf{x}) \leq 0\}$. Note that the universal convex function approximation property enjoyed by ICNNs implies that *any* convex set can be approximated arbitrarily well by the 0-sublevel set of an ICNN. *Thus, ICNNs are well-matched to the task of approximating the true feasible region $\mathcal{F}_{\mathcal{C}}$ for contingency screening, which is itself a convex set.*

Following Definition 1, a reliable ICNN classifier is one whose predicted feasible region is contained inside the true feasible region $\mathcal{F}_{\mathcal{C}}$. In the next section, we will discuss how the convex structure of an ICNN enables both (a) tractably determining whether this containment property holds and (b) scaling an ICNN's parameters to guarantee its reliability.

3. Certifying and Enforcing Reliability for ICNN Contingency Classifiers

As discussed in Section 2.1, a *reliable* classifier for the contingency screening problem is one that makes no false negative predictions, i.e., whose predicted feasible region is fully contained inside the true feasible region $\mathcal{F}_{\mathcal{C}}$. For an ICNN classifier f_{ICNN}, this amounts to the property that its 0-sublevel set is contained in $\mathcal{F}_{\mathcal{C}}$. An immediate question that arises is whether it is possible to certify that a given classifier f_{ICNN} satisfies this reliability criterion. Conveniently, we can show that certifying this property reduces to solving a collection of convex optimization problems.

Proposition 2 *An ICNN classifier for the contingency screening problem is reliable – i.e., has zero false negative rate – if and only if*

$$\left\{ \begin{array}{ll} \max\limits_{\mathbf{x} \in \mathbb{R}^n} & \mathbf{a}_j^\top \mathbf{x} \\ \text{s.t.} & f_{\text{ICNN}}(\mathbf{x}) \leq 0 \end{array} \right\} \leq b_j \quad \textit{for all } j \in [2m|\mathcal{C}|]\,. \tag{4}$$

A proof of this result can be found in the longer version of this paper (Christianson et al., 2024). This proposition provides a way of tractably certifying whether a given ICNN classifier is reliable, but it does not give a means of transforming an unreliable classifier into a reliable one. Since reliability of a classifier is exactly containment of its predicted feasible set inside the true feasible set, a natural approach to enforcing reliability would be to transform the classifier to translate and scale its predicted feasible set into the interior of $\mathcal{F}_{\mathcal{C}}$. As we show in the following theorem, it is possible to perform such a scaling efficiently by solving a collection of convex optimization problems, yielding a reliable classifier.

Theorem 3 *Let r^* and $\mathbf{v}^*$ be the optimal solutions to the optimization problem*

$$\min_{r \in \mathbb{R}_+, \mathbf{v} \in \mathbb{R}^n} \quad r \tag{5a}$$

$$\text{s.t.} \quad z_j^* \leq \mathbf{a}_j^\top \mathbf{v} + b_j r \quad \forall j \in [2m|\mathcal{C}|]\,, \tag{5b}$$

where $$z_j^* := \max_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{a}_j^\top \mathbf{x} \quad \text{s.t.} \quad f_{\text{ICNN}}(\mathbf{x}) \le 0 \tag{6}$$

for each $j \in [2m|\mathcal{C}|]$. *Then the transformed ICNN classifier* $\hat{f}_{\text{ICNN}}$ *defined as*

$$\hat{f}_{\text{ICNN}}(\mathbf{x}) := f_{\text{ICNN}}(r^*\mathbf{x} + \mathbf{v}^*)$$

has zero false negative rate. Moreover, (5) *has a feasible solution as long as the original predicted feasible set* $\{\mathbf{x} \in \mathbb{R}^n : f_{\text{ICNN}}(\mathbf{x}) \le 0\}$ *is bounded.*

A proof of Theorem 3 can be found in the longer version of this paper (Christianson et al., 2024). Note that the transformed classifier $\hat{f}_{\text{ICNN}}$ can be obtained from f_{ICNN} (as defined in (3)) by multiplying the weights $\mathbf{D}_i$ by r^* and adding $\mathbf{D}_i\mathbf{v}^*$ to the biases. As long as r^* is not infinite – that is, if (5) is feasible – then the predicted feasible set of $\hat{f}_{\text{ICNN}}$ will be nonempty (assuming that f_{ICNN}'s predicted feasible set is nonempty). We thus seek to minimize r to maximize the volume of $\hat{f}_{\text{ICNN}}$'s predicted feasible set, which will ensure reliability while minimizing the conservativeness of this classifier as an inner approximation of the true feasible set $\mathcal{F}_{\mathcal{C}}$. In Section 4 we will propose a methodology to further reduce this conservativeness and enforce classifier reliability *during ICNN training* by incorporating a version of the scaling problem (5) into the training process as a differentiable layer.

4. Training Reliable Classifiers with Differentiable Convex Optimization Layers

To avoid conservativeness resulting from the scaling proposed in the previous section, this scaling procedure must be incorporated into the training of the ICNN classifier and not solely applied after training. A natural approach is as follows: at each epoch of training, first solve the problems (5) and (6) to determine the optimal scaling parameters r^* and $\mathbf{v}^*$. Then, evaluate the training loss of the transformed ICNN classifier – for a single injection/label pair $(\mathbf{x}, y)$, we denote this loss $L\left(f_{\text{ICNN}}(r^*\mathbf{x} + \mathbf{v}^*), y\right)$, where L is some classification loss – and update the model f_{ICNN} using the gradient $\frac{\partial L}{\partial f_{\text{ICNN}}}$, where ∂f_{ICNN} refers to the gradient with respect to all the parameters of f_{ICNN}.

However, this approach is incomplete. In particular, note that the scaling parameters $r^*, \mathbf{v}^*$ resulting from (5) themselves depend on the parameters of f_{ICNN} through each z_j^*. Thus to compute the gradient of the loss L with respect to the parameters of the ICNN f_{ICNN}, it is necessary to also compute the gradients of the optimal solutions $r^*, \mathbf{v}^*$ of (5) with respect to each z_j^*, and the gradient of each optimal value z_j^* of (6) with respect to f_{ICNN}'s parameters. To compute these gradients, we can employ differentiable convex optimization layers (Agrawal et al., 2019), which automatically compute the gradient of a convex optimization problem with respect to problem parameters by differentiating through the Karush-Kuhn-Tucker (KKT) conditions of the problem, allowing the incorporation of such problems into machine learning training methodologies in a fully differentiable manner. By computing r^* and $\mathbf{v}^*$ using differentiable layers, we ensure that the training process is "aware" of the scaling procedure that is applied to f_{ICNN} to guarantee reliability.

While this fully differentiable approach ensures that the scaling procedure is accounted for when computing the loss gradient, it requires computing both the solution to (5) and the solutions to (6) for all $j \in [2m|\mathcal{C}|]$ using differentiable layers, which typically require additional computational overhead beyond a non-differentiable solution (Agrawal et al., 2019). Because we need to apply this scaling at each epoch of training to enforce reliability, reducing the number of differentiable optimization layers used at each step of training would improve computational efficiency.

Fortunately, as we show in the following theorem, it is possible to obtain a fully differentiable scaling procedure using just a *single* differentiable optimization problem.

Theorem 4 *Let z_j^* be defined as in (6) for each $j \in [2m|\mathcal{C}|]$, and let $j^* := \arg\max_j z_j^*/b_j$. Define r^* to be the optimal value of the following problem:*

$$r^* := \max_{\mathbf{x}\in\mathbb{R}^n} \quad \mathbf{a}_{j^*}^\top \mathbf{x}/b_{j^*} \quad \text{s.t.} \quad f_{\text{ICNN}}(\mathbf{x}) \le 0. \tag{7}$$

Then the transformed ICNN classifier $\hat{f}_{\text{ICNN}}$ defined as $\hat{f}_{\text{ICNN}}(\mathbf{x}) := f_{\text{ICNN}}(r^*\mathbf{x})$ has zero false negative rate. Moreover, (7) has a feasible solution as long as the original predicted feasible set $\{\mathbf{x} \in \mathbb{R}^n : f_{\text{ICNN}}(\mathbf{x}) \le 0\}$ is bounded.

Proof Consider the optimization problem (5), and fix $\mathbf{v} = \mathbf{0}$; this problem remains feasible, by the assumption that the predicted feasible set is bounded, and since $\mathbf{0} \in \text{int}\,\mathcal{F}_{\mathcal{C}}$ (Assumption 1) implies that $\mathbf{b} > \mathbf{0}$. The optimal solution r^* to (5) is the smallest value of r that satisfies the constraints (5b); this is exactly $r^* := \max_j z_j^*/b_j$. One can easily see that this r^* is identical to the one obtained by (7). Thus, the scaling obtained from (7) inherits the zero false negative rate yielded by (5). ■

In Theorem 4, the values z_j^* only need to be computed in order to determine the maximizing index j^*; then, the scaling ratio r^* is computed using just the single optimization problem (7). As such, all of the z_j^* can be computed in parallel in a non-differentiable fashion, and only (7) must be solved using a differentiable layer. Note additionally that the lack of a translation variable $\mathbf{v}$ in (7) won't yield any additional conservativeness during training, since the ICNN can learn biases that would imitate the impact of any possible $\mathbf{v}$.

We outline in Algorithm 1 a training methodology incorporating the fast, differentiable scaling procedure in Theorem 4. In this process, we begin by "warm-starting" the training for M_w epochs by performing standard gradient descent on the classification loss without scaling for reliability. Then, for each of the remaining M_s epochs, the model is scaled using a differentiable layer implementing (7) before evaluating the training loss. Note that after every gradient step, the ICNN's weights $\mathbf{W}_i$ must be clipped to the positive orthant to maintain convexity.

Algorithm 1 Training for reliable ICNN classifiers

Input: training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, initial ICNN f_{ICNN}, warm-start epochs M_w, scaling epochs M_s, batch size s

```
/* Warm-start without scaling */
```
for each *epoch in* $[M_w]$ **do**
- **for each** *mini-batch* $B \subset [N]$ **do**
  - Evaluate the loss.
  - Compute the gradient $\frac{\partial\,\text{loss}}{\partial f_{\text{ICNN}}}$ and use it to update f_{ICNN}.
- **end**

end

```
/* Train with scaling to enforce reliability */
```
for each *epoch in* $[M_s]$ **do**
- Compute z_j^* through (6) for each $j \in [2m|\mathcal{C}|]$.
- Set $j^* := \arg\max_j z_j^*/b_j$.
- Compute r^* through (7) using a differentiable convex optimization layer .
- Evaluate the loss $\frac{1}{s}\sum_{i\in B} L\left(f_{\text{ICNN}}(r^*\mathbf{x}_i), y_i\right)$ of the scaled model on a mini-batch B.
- Compute the gradient $\frac{\partial\,\text{loss}}{\partial f_{\text{ICNN}}}$ and use it to update f_{ICNN}.

end

5. Experimental Results

In this section, we describe the results of our ICNN training methodology (Algorithm 1) in a case study of $N-2$ contingency screening on the IEEE 39-bus test network (Athay et al., 1979; Pai, 1989). All experiments were performed on a MacBook Pro with 12-core M3 Pro processor, and the experimental code is available upon request. A full description of the setup, data, training details, and results can be found in the longer version of this paper (Christianson et al., 2024).

We used the IEEE 39-bus test network implemented in pandapower (Thurner et al., 2018). To construct the true feasible set $\mathcal{F}_{\mathcal{C}}$, we took the set of all $N-2$ contingencies and dropped any islanding contingencies as well as contingencies that were infeasible more than 90% of the time, since these should be handled separately. We show in Figure 2 a 2-dimensional slice of the true feasible region $\mathcal{F}_{\mathcal{C}}$ and the predicted feasible region of a 1-layer ICNN trained via our methodology. It is evident that the ICNN respects the inner approximation property as a result of the scaling procedure while learning to focus on the data-intensive region at the bottom of the true feasible region. The ICNN does not need to learn the shape of the entire true feasible region due to data sparsity at the top of this slice, enabling a more efficient representation.

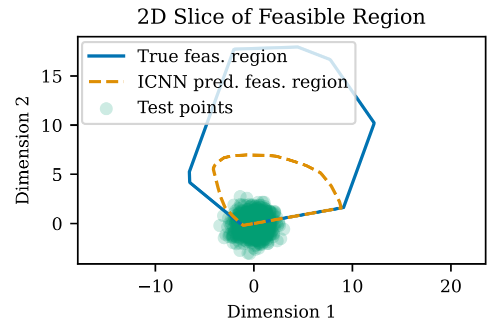

Figure 2: A 2-dimensional feasible region slice.

5.1. Contingency Screening Results

We show in Figure 3 a comparison of ICNNs with several hidden depths trained via our approach against standard, nonconvex NNs and the exhaustive method of checking all constraints individually for the contingency screening problem. The "Positive Weight" value denotes the weight assigned to elements of the positive class in the binary cross-entropy training loss, where weights less than 1 typically encourage lower false positive rates, and weights greater than 1 encourage lower false negative rates.

Notably, the ICNNs trained with our differentiable scaling procedure in Algorithm 1 achieve a speedup of 10-20 $\times$ over the exhaustive method, depending on the depth of the ICNN (Figure 3, top). Moreover, they uniformly achieve a false negative rate of 0 (Figure 3, middle), as guaranteed by our theoretical results, and a false positive rate between 2% and 5% (Figure 3, bottom). While the effect is not significant, it appears in the cases of hidden depth 1 and 3 that a lower positive weight may decrease the false positive rate of our approach, though further study is needed to understand whether this behavior holds in general.

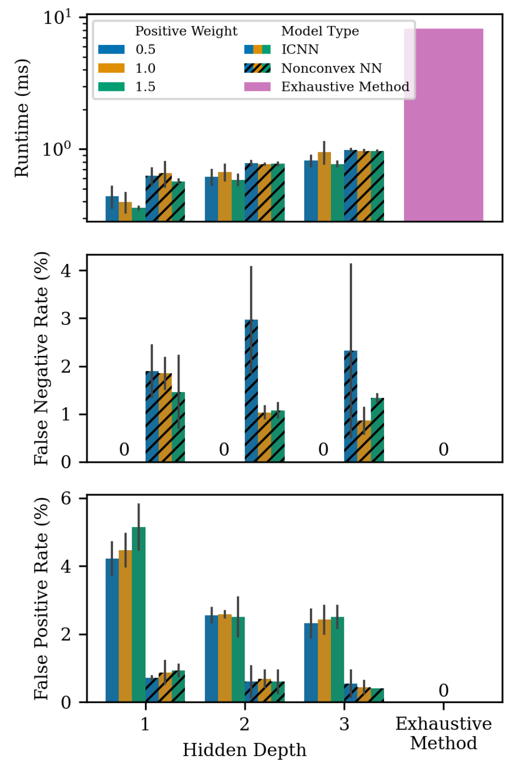

Figure 3: Comparison of ICNN-based contingency analysis against a nonconvex NN and exhaustive method.

In comparison, the nonconvex NNs achieve a better false positive rate, ranging between 0.5% and 1%, but suffer significant false negative rates of 1% to 3%, demonstrating that they could misclassify infeasible scenarios as feasible. Our approach thus enables significantly faster screening than the exhaustive method while ensuring the reliability that cannot be guaranteed by standard NNs.

5.2. Faster Preventive Dispatch via SC-OPF

In practice, power system operators often want to perform *preventive* dispatch to ensure that the chosen operating point will remain feasible in the case of contingencies. This problem, known as security-constrained (SC)-DC-OPF, adds to (1) the additional constraint that $\mathbf{x} := \mathbf{p} - \mathbf{d}$ should be feasible for all contingencies in the reference set $\mathcal{C}$ – that is, $\mathbf{p} - \mathbf{d} \in \mathcal{F}_\mathcal{C}$.

Because our ICNN approach to contingency screening yields an ICNN $f_{\text{ICNN}}(r^*\cdot)$ whose 0-sublevel set is an inner approximation to $\mathcal{F}_\mathcal{C}$, one might naturally consider replacing the security constraint in the full SC-OPF problem with the conservative inner approximation $\hat{f}_{\text{ICNN}}\left(r^*(\mathbf{p} - \mathbf{d})\right) \le 0$ in an attempt to accelerate the solution time of this problem, since the original set $\mathcal{F}_\mathcal{C}$ is typically high-dimensional. We test the performance of this approach and its impact on system cost and infeasibility using our ICNN models trained on the IEEE 39-bus system, and we display the results in Figure 4.

We see that, while the ICNNs with hidden depth 3 do not offer a speedup compared to solving the full SC-OPF exactly, the 2-layer ICNNs halve the runtime, and the shallowest 1-layer ICNNs speed up this problem by nearly a factor of 10 (Figure 4, top). Remarkably, they achieve this speedup while increasing the dispatch cost by no more than 0.1% on average over the full SC-OPF problem (Figure 4, middle), and increasing the share of infeasible demand instances by only $\sim 1\%$. It also appears that, for the ICNNs with hidden depth 1, decreasing the positive weight leads to better cost and less infeasibility. This agrees with intuition, since a lower positive weight encourages lower false positive rates, so the ICNN should be a less conservative inner approximation to the set $\mathcal{F}_\mathcal{C}$. However, further study is needed to determine whether this observation generalizes, as the trend falls within the error bars and the deeper models do not seem to exhibit this behavior.

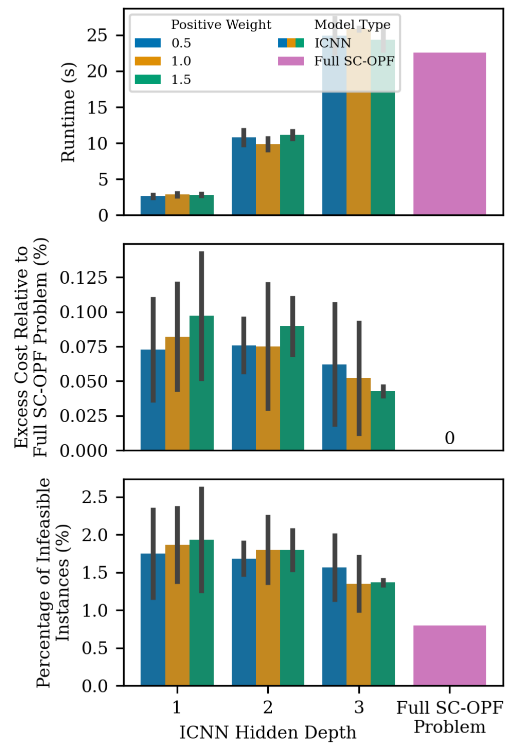

Figure 4: Results for the ICNN-based SC-OPF problem compared to the full SC-OPF problem.

6. Discussion and Conclusions

In this work, we proposed a methodology for data-driven training of input-convex neural network classifiers for contingency screening in power systems with zero false negative rate. We show that certifying and enforcing zero false negative rate – i.e., reliability – of an ICNN classifier can be achieved by solving a collection of optimization problems, and by incorporating these problems into a differentiable convex optimization layer during ICNN training, we can restrict training to be over the set of provably reliable models. Future directions include scaling up this approach to enable applications to larger-scale power systems and extending the methodology to other applications that require constructing tractable inner approximations to some complicated set, such as learning data-driven and safe inner approximations to AC-OPF feasible regions or electric vehicle aggregate flexibility sets.

Acknowledgments

The authors thank Carey Priebe, Hayden Helm, and Christopher Yeh for illuminating discussions, and Kate Lytvynets for technical assistance. The authors acknowledge support from an NSF Graduate Research Fellowship (DGE-2139433), NSF Grant ECCS-1942326, the Resnick Sustainability Institute, a Caltech S2I Grant, and C3.ai Award #11015.

References

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable Convex Optimization Layers. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html.

Brandon Amos, Lei Xu, and J. Zico Kolter. Input Convex Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 146–155. PMLR, July 2017. URL https://proceedings.mlr.press/v70/amos17b.html. ISSN: 2640-3498.

T. Athay, R. Podmore, and S. Virmani. A Practical Method for the Direct Analysis of Transient Stability. *IEEE Transactions on Power Apparatus and Systems*, PAS-98(2):573–584, March 1979. ISSN 0018-9510. doi: 10.1109/TPAS.1979.319407. URL https://ieeexplore.ieee.org/document/4113518.

Daniel Bienstock. *Electrical Transmission System Cascades and Vulnerability*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, December 2015. ISBN 978-1-61197-415-7. doi: 10.1137/1.9781611974164. URL https://epubs.siam.org/doi/book/10.1137/1.9781611974164.

Joshua W. Busby, Kyri Baker, Morgan D. Bazilian, Alex Q. Gilbert, Emily Grubert, Varun Rai, Joshua D. Rhodes, Sarang Shidore, Caitlin A. Smith, and Michael E. Webber. Cascading risks: Understanding the 2021 winter blackout in Texas. *Energy Research & Social Science*, 77:102106, July 2021. ISSN 2214-6296. doi: 10.1016/j.erss.2021.102106. URL https://www.sciencedirect.com/science/article/pii/S2214629621001997.

Yize Chen, Yuanyuan Shi, and Baosen Zhang. Optimal Control Via Neural Networks: A Convex Approach. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1MW72AcK7.

Nicolas Christianson, Wenqi Cui, Steven Low, Weiwei Yang, and Baosen Zhang. Fast and reliable $N - k$ contingency screening with input-convex neural networks. *arXiv preprint arXiv:2410.00796*, 2024.

Constance Crozier, Kyri Baker, Yuhan Du, Javad Mohammadi, and Meiyi Li. Data-driven Contingency Selection for Fast Security Constrained Optimal Power Flow. In *2022 17th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6, June 2022. doi: 10.1109/PMAPS53380.2022.9810574. ISSN: 2642-6757.

C. Matthew Davis and Thomas J. Overbye. Multiple Element Contingency Screening. *IEEE Transactions on Power Systems*, 26(3):1294–1301, August 2011. ISSN 1558-0679. doi: 10.1109/TPWRS.2010.2087366. Conference Name: IEEE Transactions on Power Systems.

Mark K. Enns, John J. Quada, and Bert Sackett. Fast Linear Contingency Analysis. *IEEE Transactions on Power Apparatus and Systems*, PAS-101(4):783–791, April 1982. ISSN 0018-9510. doi: 10.1109/TPAS.1982.317142. Conference Name: IEEE Transactions on Power Apparatus and Systems.

U.S.-Canada Power System Outage Task Force. Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations. Technical report, 2004. URL https://www.nerc.com/pa/rrm/ea/2003%20Blackout%20Final%20Report/ch1-3.pdf.

Jiachun Guo, Yong Fu, Zuyi Li, and M. Shahidehpour. Direct Calculation of Line Outage Distribution Factors. *IEEE Transactions on Power Systems*, 24(3):1633–1634, August 2009. ISSN 0885-8950, 1558-0679. doi: 10.1109/TPWRS.2009.2023273. URL http://ieeexplore.ieee.org/document/5130240/.

P. Kaplunovich and K. Turitsyn. Fast and Reliable Screening of N-2 Contingencies. *IEEE Transactions on Power Systems*, 31(6):4243–4252, November 2016. ISSN 1558-0679. doi: 10.1109/TPWRS.2015.2510586. Conference Name: IEEE Transactions on Power Systems.

P. A. Kaplunovich and K. S. Turitsyn. Fast selection of N-2 contingencies for online security assessment. In *2013 IEEE Power & Energy Society General Meeting*, pages 1–5, 2013. doi: 10.1109/PESMG.2013.6672792. URL http://arxiv.org/abs/1303.3938. arXiv:1303.3938 [math-ph, physics:physics].

Agnes M. Nakiganda, Catherine Cheylan, and Spyros Chatzivasileiadis. Topology-Aware Neural Networks for Fast Contingency Analysis of Power Systems, October 2023. URL http://arxiv.org/abs/2310.04213. arXiv:2310.04213 [cs, eess].

NERC. NERC Reliability Standard PRC-003-0a, 2010. URL https://www.nerc.com/pa/Stand/Reliability%20Standards/TPL-003-0a.pdf.

M. A. Pai. *Energy Function Analysis for Power System Stability*. Springer US, Boston, MA, 1989. ISBN 978-1-4612-8903-6 978-1-4613-1635-0. doi: 10.1007/978-1-4613-1635-0. URL http://link.springer.com/10.1007/978-1-4613-1635-0.

Brian Stott, Jorge Jardim, and Ongun Alsac. DC Power Flow Revisited. *IEEE Transactions on Power Systems*, 24(3):1290–1300, August 2009. ISSN 1558-0679. doi: 10.1109/TPWRS.2009.2021235. URL https://ieeexplore.ieee.org/document/4956966/?arnumber=4956966. Conference Name: IEEE Transactions on Power Systems.

Goran Strbac, Daniel Kirschen, Rodrigo Moreno, et al. Reliability standards for the operation and planning of future electricity networks. *Foundations and Trends® in Electric Energy Systems*, 1(3):143–219, 2016.

Leon Thurner, Alexander Scheidler, Florian Schäfer, Jan-Hendrik Menke, Julian Dollichon, Friederike Meier, Steffen Meinecke, and Martin Braun. Pandapower—An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems. *IEEE Transactions on Power Systems*, 33(6):6510–6521, November 2018. ISSN 1558-0679. doi: 10.1109/TPWRS.2018.2829021. URL https://ieeexplore.ieee.org/document/8344496.

Yafei Yang, Xiaohong Guan, and Qiaozhu Zhai. Fast Grid Security Assessment With N - k Contingencies. *IEEE Transactions on Power Systems*, 32(3):2193–2203, May 2017. ISSN 1558-0679. doi: 10.1109/TPWRS.2016.2608378.