1. Additional Experiment for Rebuttal

We conduct the following experiments for rebuttal: (i) construct an **irregular constraint benchmark** for IPNN training; (ii) conduct **sensitivity analysis** in high-dimension problems for IPNN hyper-parameters; (iii) solving **verification of IPNN** over QP, convex QCQP, and non-convex JCCIM.

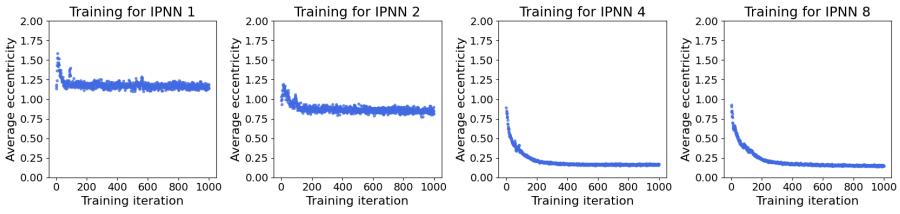
1.1. Benchmark Irregular Constraint Set

We construct the following constraint set as the benchmark to test IPNN:

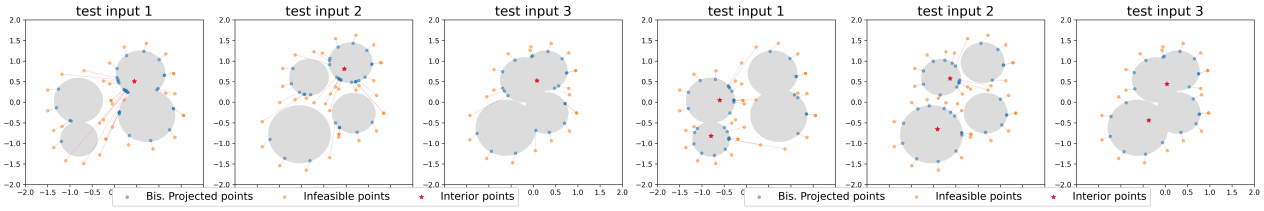
$$C_\theta = \cup_{i=1}^{4} \mathcal{B}(x_i, r_i) \tag{1}$$

where $x_i \in \mathbb{R}^2$, $r_i \in \mathbb{R}^+$, and the input parameter is defined as $\theta = \{x_i, r_i\}_{i=1}^4$. We remark that the geometry of this constraint set is highly irregular, like non-convex and disconnected, making it suitable for testing the BP framework. Further, the minimum eccentricity of 4 IPs is zero due to the symmetry of the balls, and the MEIPs are the centers of the balls.

- **IPNN training and testing with different IPs**. We first visualize the eccentricity convergence of IPNN training and test the IPNN under new input parameters to validate its performance, as shown in Fig. 1.
- **IPNN training without eccentricity loss**. To validate the necessity of IPNN training with minimizing eccentricity measure, we train IPNN with penalty term only and visualize its performance, as shown in Fig. 2.

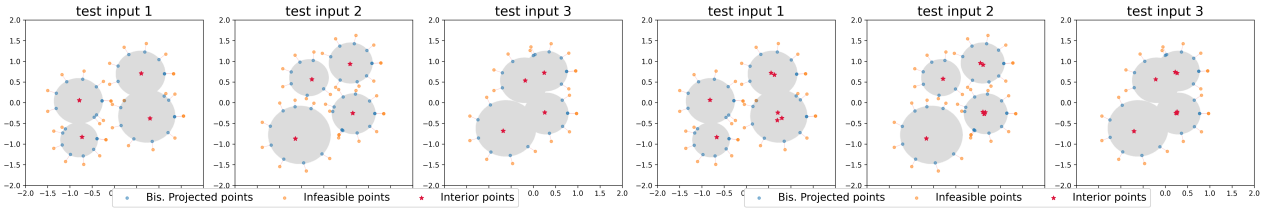


(a) Unsupervised IPNN training with different IP predictors.



(b) IPNN with 1 IP predictors and average eccentricity of 0.89

(c) IPNN with 2 IP predictors and average eccentricity of 0.21.



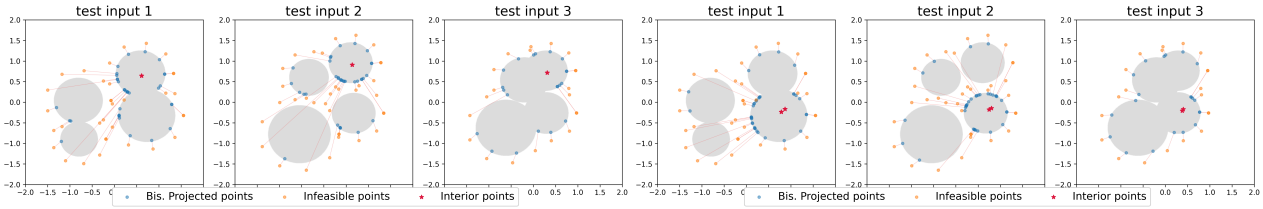
(d) IPNN with 4 IP predictors and average eccentricity of 0.16.

(e) IPNN with 8 IP predictors and average eccentricity of 0.14.

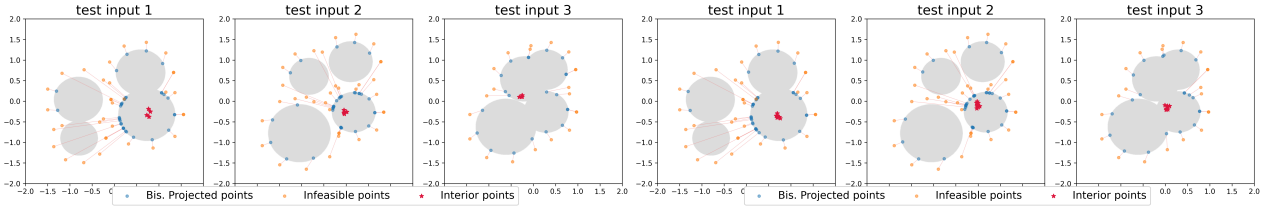
Figure 1: Training and testing IPNN for approximating MEIPs of non-convex sets.

1.2. Sensitivity Analysis for IPNN training

We present the sensitivity analysis for the BP framework under different numbers of IPs, random IPs, and NN structures, as well as the boundary sampling time statistics. Specifically, we select convex QCQP problem and the non-convex JCCIM problem as test beds to compare the sensitivity of different parameters. Further, we compare their performance in an adversarial setting with a trained NN predictor with poor feasibility and use its output as infeasible predictions for our



(a) IPNN with 1 IP predictors without minimizing eccentricity (b) IPNN with 2 IP predictors without minimizing eccentricity.



(c) IPNN with 4 IP predictors without minimizing eccentricity. (d) IPNN with 8 IP predictors without minimizing eccentricity.

Figure 2: Training and testing IPNN without minimizing eccentricity of non-convex sets.

bisection algorithm. The performance is shown in Table 1, which demonstrates the efficiency of training with penalty term to find interior points and the necessity of eccentricity to reduce optimality loss and reach feasibility guarantee.

Table 1: Sensitivity analysis for BP framework.

	Feasibility		**Optimality**		**Run-time**	**Training**
	feas. rate (%)	ineq. vio.	sol. err. (%)	obj. gap (%)	total (s)	per iteration (s)
	Convex QCQP: : $n = 400,\ d = 100,\ n_{\text{eq}} = 100,\ n_{\text{ineq}} = 100$					
NN	0.00%	11.05	3.10%	0.03%	0.001	0.06
Random-IPNN 1	98.34%	0.39	8.55%	6.67%	0.13	0.007
Random-IPNN 2	100%	0.00	6.23%	4.15%	0.27	0.005
Random-IPNN 4	100%	0.00	5.62%	3.66%	0.65	0.008
Random-IPNN 8	100%	0.00	5.32%	3.43%	0.82	0.01
ME-IPNN 1	100%	0.00	5.18%	3.63%	0.07	0.02
ME-IPNN 2	100%	0.00	5.01%	3.46%	0.34	0.02
ME-IPNN 4	100%	0.00	4.87%	3.30%	0.39	0.03
ME-IPNN 8	100%	0.00	4.80%	3.28%	0.65	0.03
	Non-convex JCC-IM: : $n = 400,\ d = 100,\ n_{\text{eq}} = 0,\ n_{\text{ineq}} = 10100$					
NN	6.54%	0.18	2.68%	1.66%	0.001	0.06
Random-IPNN 1	100%	0.00	6.22%	5.78%	0.39	0.09
Random-IPNN 2	100%	0.00	5.47%	4.94%	0.46	0.07
Random-IPNN 4	100%	0.00	5.21%	4.65%	0.48	0.07
Random-IPNN 8	100%	0.00	5.01%	4.43%	0.62	0.07
ME-IPNN 1	100%	0.00	4.74%	4.20%	0.42	0.77
ME-IPNN 2	100%	0.00	4.29%	3.74%	0.41	0.71
ME-IPNN 4	100%	0.00	3.91%	3.24%	0.50	0.69
ME-IPNN 8	100%	0.00	3.68%	2.97%	0.54	0.70

[1] Random-IPNN indicates it is trained with only penalty terms for constraint violation.
[2] ME-IPNN indicates it is trained with penalty terms and eccentricity measures jointly.
[3] Run time indicates the time for running the bisection algorithm to recover feasibility for infeasible cases out of 1024 test instances.
[4] Training time indicates the average per-iteration computational cost, consisting of boundary sampling time and NN forward-backward propagation.

1.3. Verification for IPNN

Without loss of generality, we focus on the IPNN with 1 IP predictor and the verification of multiple IP predictors can be solved independently. Recall the feasibility verification problem as:

$$\textbf{P1:} \quad \min_{\theta, x} \ t \tag{2}$$

$$\text{s.t. } g(x, \theta) \leq t,\ \theta \in \Theta,\ x = \text{IPNN}(\theta) \tag{3}$$

The optimal objective t^* can be viewed as the worst-case constraint violation for the output of NN given arbitrary input $\theta \in \Theta$. In general, this problem is non-convex and NP-hard, either due to the NN structure or non-convex constraint function. However, we then apply the relaxation techniques to reformulate the problem to derive the upper bound of constraint violation, denoted t^*, to verify the feasibility of NN:

$$\textbf{P2:} \quad \min_{\theta, x} \ t \tag{4}$$

$$\text{s.t. } \hat{g}(x, \theta) \leq t, \ \theta \in \Theta, \ x \in \text{Relax}(\text{IPNN}(\theta)) \tag{5}$$

Here we apply the relaxation (e.g., linear relaxation or SDP relaxation) for the ReLU-based NN and the convex restriction for the constraint function $g(x, \theta) \leq \hat{g}(x, \theta)$. Therefore, the relaxed problem is a convex programming in general and can be solved with polynomial complexity. Further, to tighten the relaxation, we compute the upper/lower bound for the per-layer output and add those constraints to the final verification problem (Tjeng et al., 2017; Zhao et al., 2023). We denote the optimal solution as $\hat{t}^*$, which is an upper bound of the worst-case constraint violation. The tightness of the relaxation depends on both NN reformulation and convex restriction, which might be problem-dependent for empirical experiments.

We then solve the upper bound constraint violation over the following problem with different constraint structures. The results are shown in Table 2, which demonstrates the feasibility guarantee for IPNN trained by minimizing eccentricity.

Table 2: Verification for IPNN over different constraint sets.

IPNN	**Constraint violation upper bound**
QP: : $n = 100, \ d = 50, \ n_{\text{eq}} = 50, \ n_{\text{ineq}} = 50$ **Input constraint:** $\theta \in [\theta_l, \theta_u]$**, output constraint:** $Ax = \theta, Gx \leq h$ **NN relaxation: linear, constraint restriction: N/A**	
Untrained	315.285
Training without minimizing eccentricity	1.815
Training with minimizing eccentricity	-0.622
Convex QCQP: : $n = 100, \ d = 50, \ n_{\text{eq}} = 50, \ n_{\text{ineq}} = 50$ **Input constraint:** $\theta \in [\theta_l, \theta_u]$**, output constraint:** $Ax = \theta$**,** $x^{\text{T}} H_i x + g_i x \leq h_i$ **NN relaxation: linear, constraint restriction: N/A**	
Untrained	1267.653
Training without minimizing eccentricity	0.871
Training with minimizing eccentricity	-2.67
Non-Convex JCCIM: : $n = 400, \ d = 100, \ n_{\text{eq}} = 100, \ n_{\text{ineq}} = 10100$ **Input constraint:** $\theta \in [\theta_l, \theta_u]$**, output constraint:** $\frac{1}{100} \sum_{k=1}^{100} \mathbf{I}(Ax \geq \theta + \omega_k) \geq 1 - \epsilon$**,** $Gx \leq h$ **NN relaxation: linear, constraint restriction: robust (Pagnoncelli et al., 2009)**	
Untrained	95.905
Training without minimizing eccentricity	-0.048
Training with minimizing eccentricity	-0.053

[1] We apply a 3-layer IPNN with ReLU activation, hidden dimension of n, and residual connection in hidden layers.
[2] We apply CVXPY to formulate all relaxed verification problems and use GUROBI to solve them optimally.

References

Pagnoncelli, B. K., Ahmed, S., and Shapiro, A. Sample average approximation method for chance constrained programming: theory and applications. *Journal of optimization theory and applications*, 142(2):399–416, 2009.

Tjeng, V., Xiao, K., and Tedrake, R. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Zhao, T., Pan, X., Chen, M., and Low, S. H. Ensuring dnn solution feasibility for optimization problems with convex constraints and its application to dc optimal power flow problems. In *International Conference on Learning Representations*, 2023.