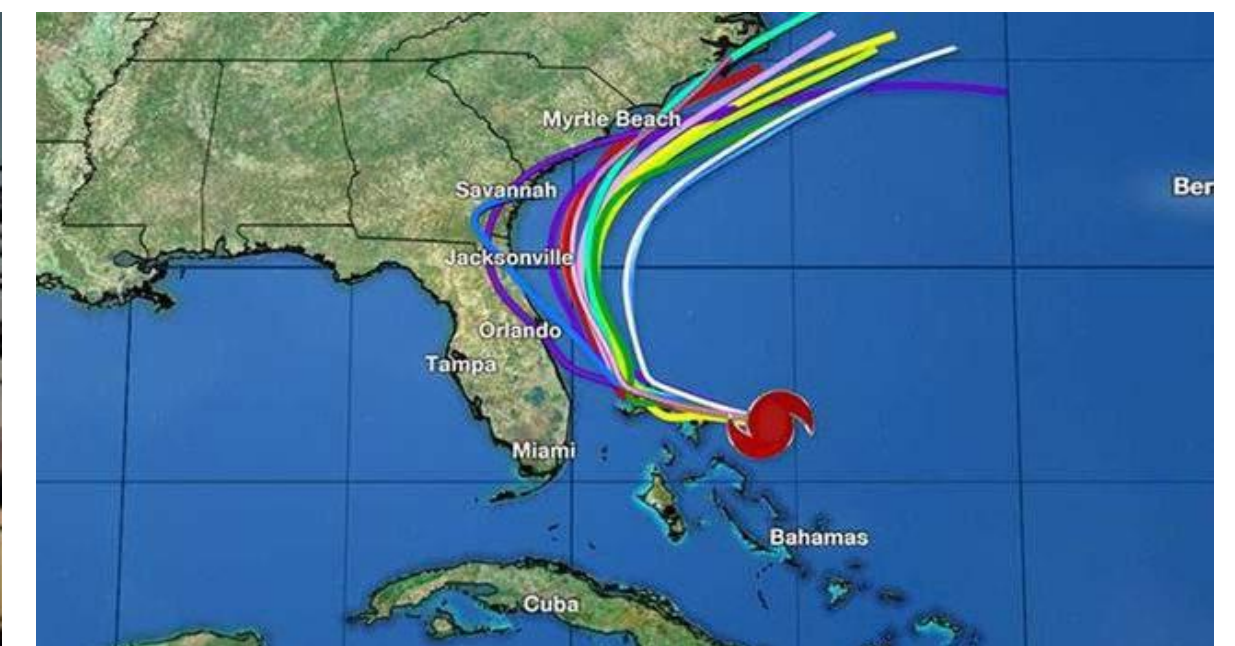


Deep Learning for Optimization

- Many computationally challenging optimization problems are solved repeatedly under different scenarios. Plenty of data is generated.
- Benefit from fast and accurate approximations.
- Deep learning solutions are particularly appealing to approximate the solution of these optimization problems.
- **Issue:** Presence of hard physical and engineering constraints.
 - Ohm's and Kirchhoff laws in power systems, Weymouth equations, in Gas networks, and the Navier-Stoke's equations for shallow water in flood mitigation.
- **Goal:** How to enable a deep learning model to take account of these constraints in its predictions?



Deep Constrained Learning

$$\mathcal{O}(d) : x^* = \underset{x}{\operatorname{argmin}} f(x, d)$$

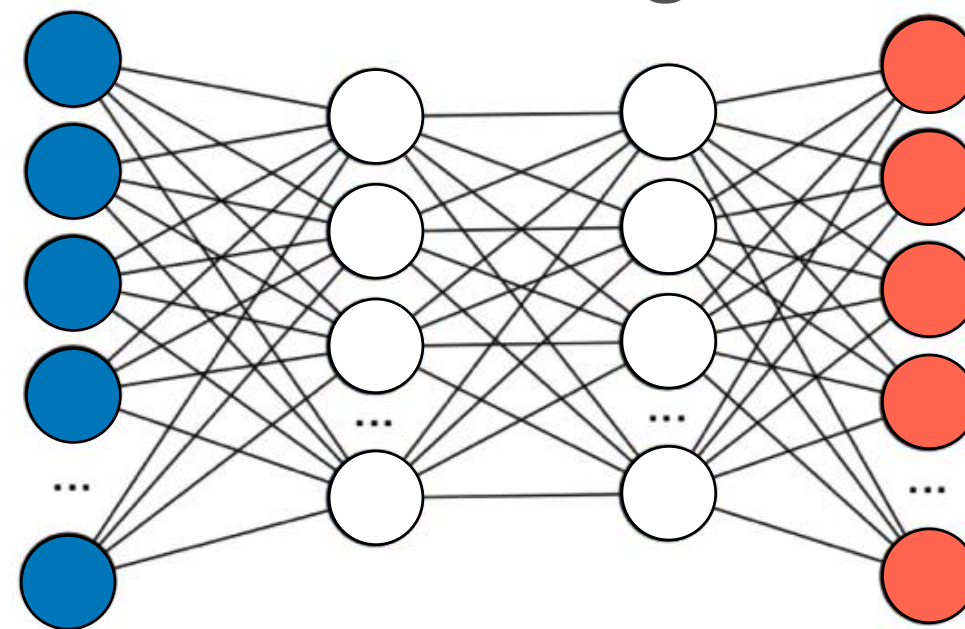
$$\text{s.t. } g(x, d) \leq 0$$



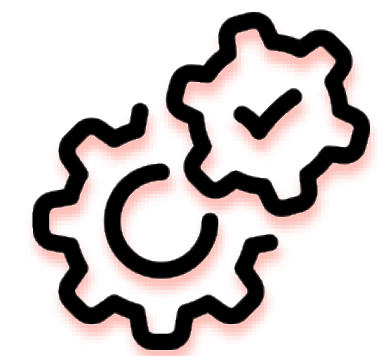
Data

$$D = \left\{ \left(d_i, x_i^* = \mathcal{O}(d_i) \right) \right\}_{i=1}^n$$

Training



Approximate
Optimizer



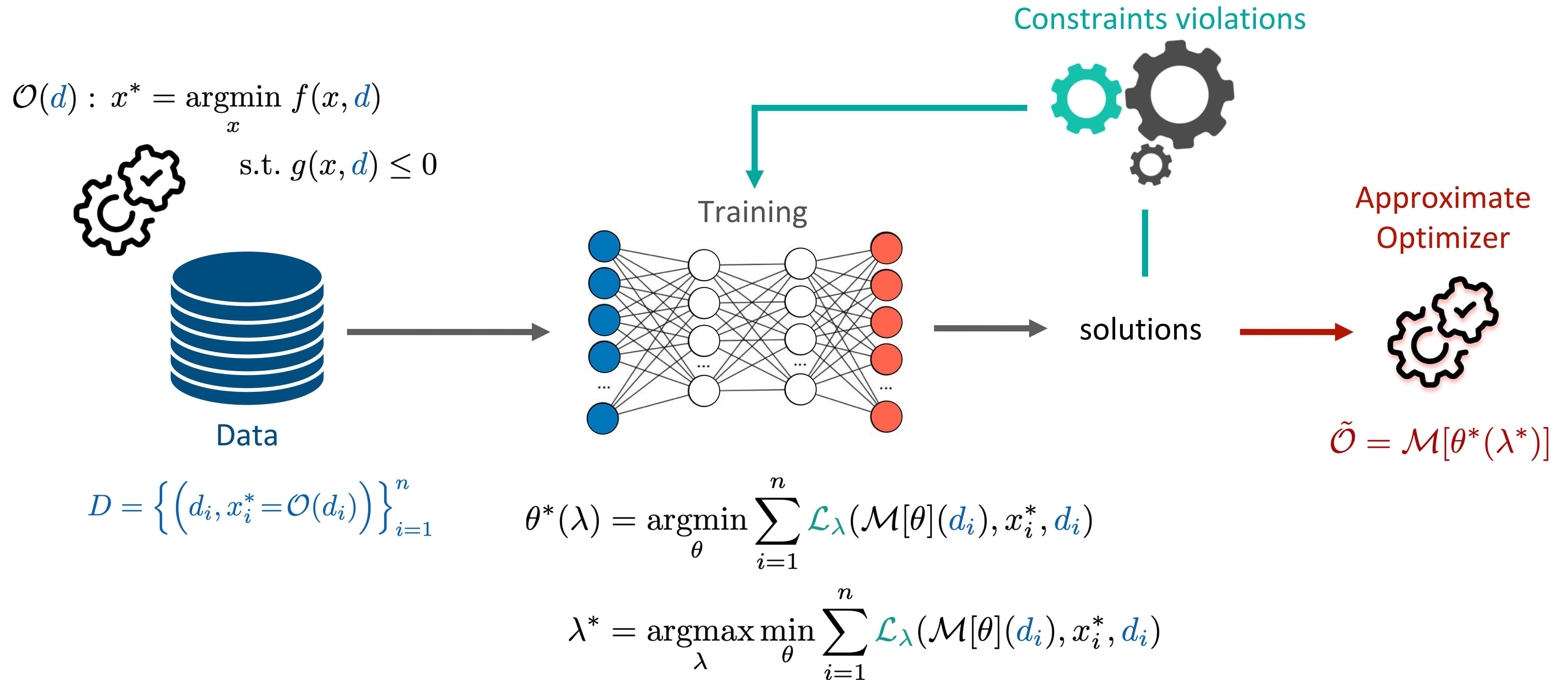
$$\tilde{\mathcal{O}} = \mathcal{M}[\theta^*]$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^n \mathcal{L}(\mathcal{M}[\theta](d_i), x_i^*)$$

$$\text{s.t. } g(\mathcal{M}[\theta](d_i), d_i) \leq 0 \quad (i \in [n])$$

challenge

Deep Constrained Learning



How does it works in Practice?

AC Optimal Power Flow Predictions

Solution Quality

Test Case		Dist. to load flow sol. (%)		
		DC	$\mathcal{M}^-$	$\mathcal{M}$
30_ieee	p^g	2.6972	2.0793	0.0007
	v	1.2929	83.138	0.0037
118_ieee	p^g	0.2011	0.1071	0.0038
	v	1.9971	3.4391	0.0866
300_ieee	p^g	0.1336	0.0447	0.0084
	v	3.8526	31.698	0.1994
	p^g	0.7751	0.9843	0.0197
	v	2.4284	36.288	0.1995
		Total Average		

Objective cost distance and runtime

Test Case	Dist. to AC-OPF cost (%)			Runtime (sec.)		
	DC	$\mathcal{M}^-$	$\mathcal{M}$	AC	DC	$\mathcal{M}$
30_ieee	7.9894	2.9447	0.0180	0.1024	0.0148	$< 10^4$
118_ieee	4.7455	1.0973	0.5408	0.4207	0.0785	0.0001
300_ieee	4.7508	1.9543	0.3011	8.0645	0.2662	0.0001
	4.5733	2.3706	0.2124	1x	30.3x	$> 10^4$ x
	Total Average			Min Speedup		

$\mathcal{M}$: AI-based model

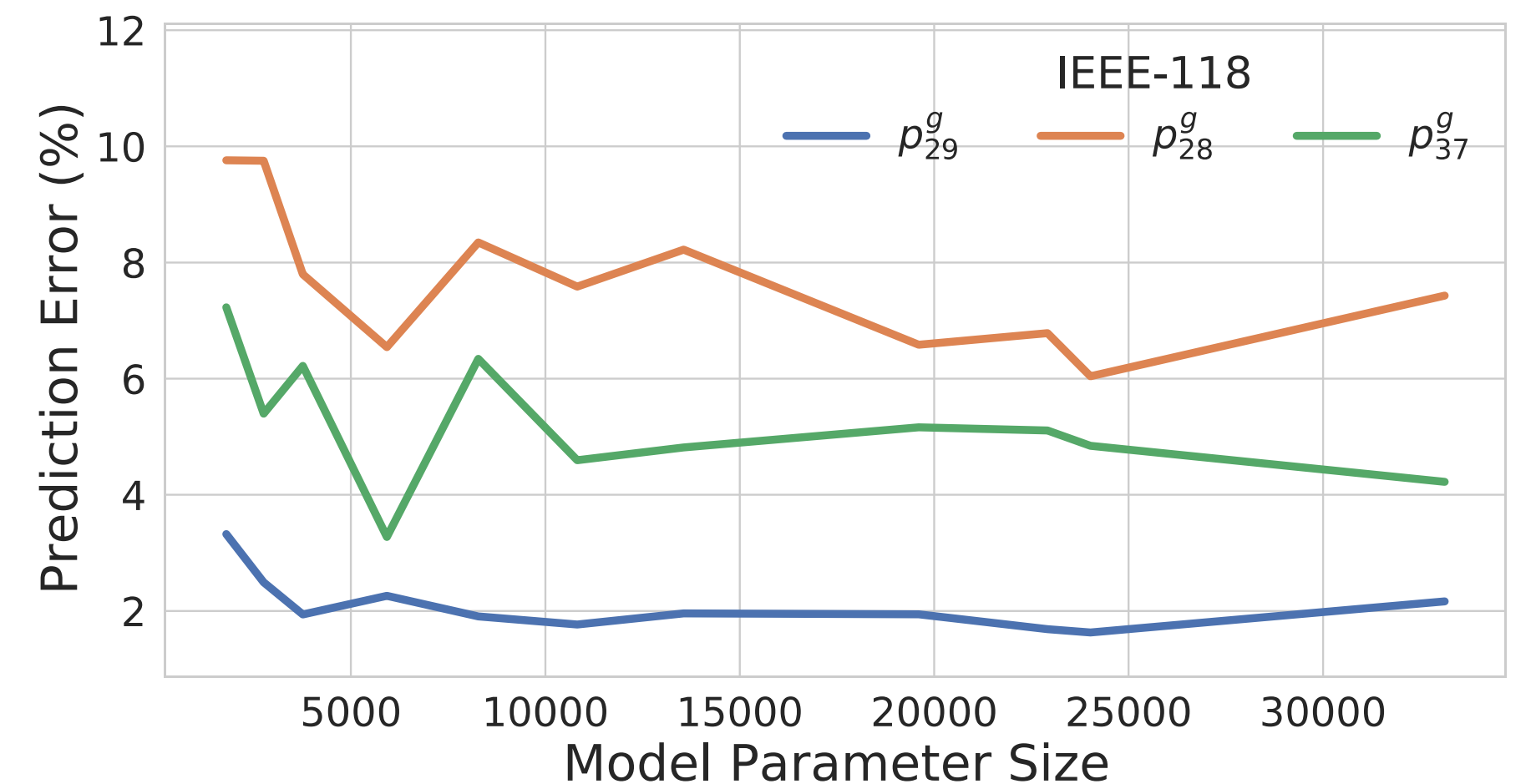
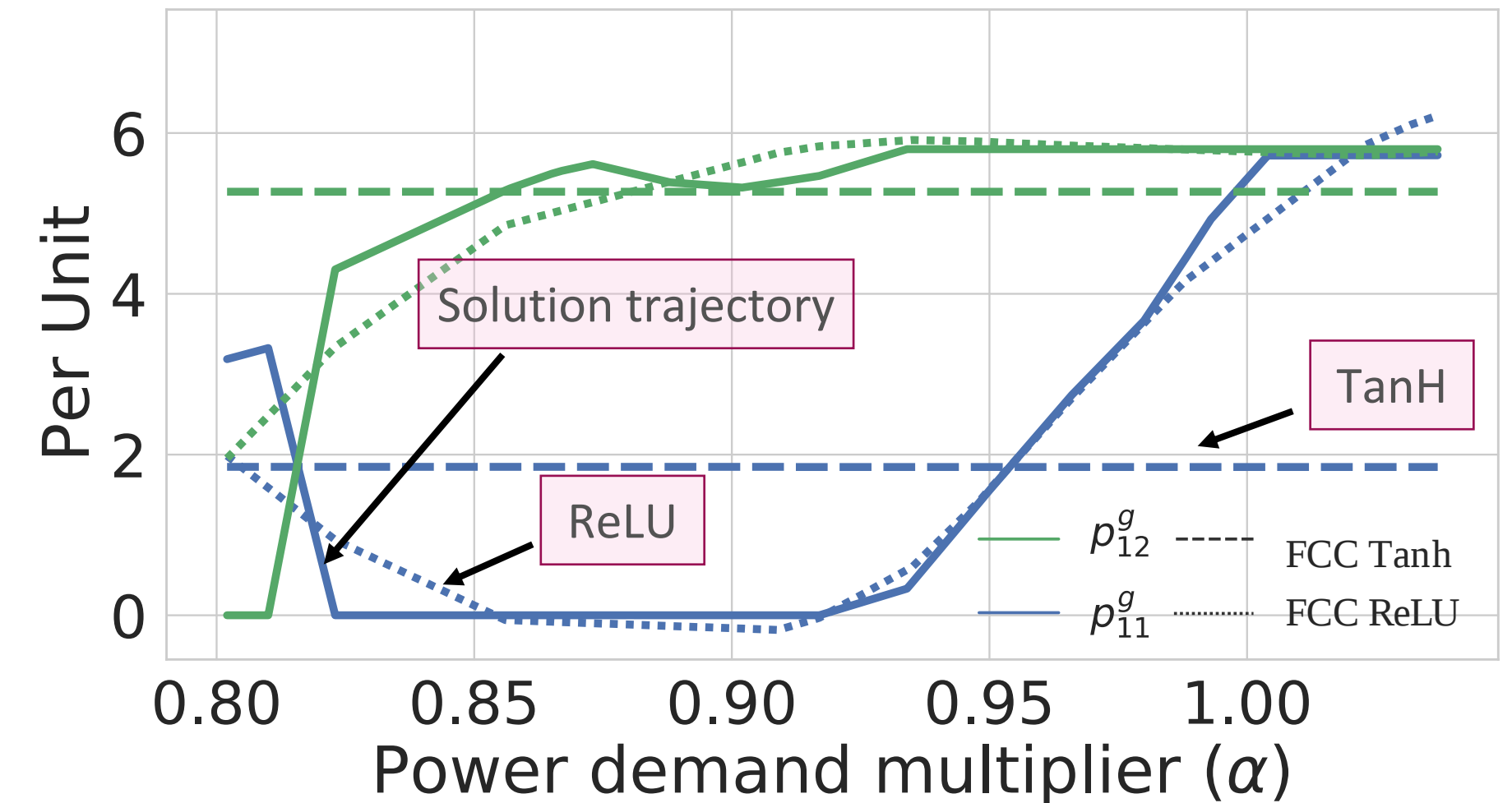
AC: full non-linear model

DC: linear approximation
(as used in industry)

Summary: AI-based model can predict quantities **several order of magnitude more accurately** and **faster** than the linear (DC) approximation (and a baseline learning model $\mathcal{M}^-$) and reports **significantly less constraint violations**.

Why does it work?

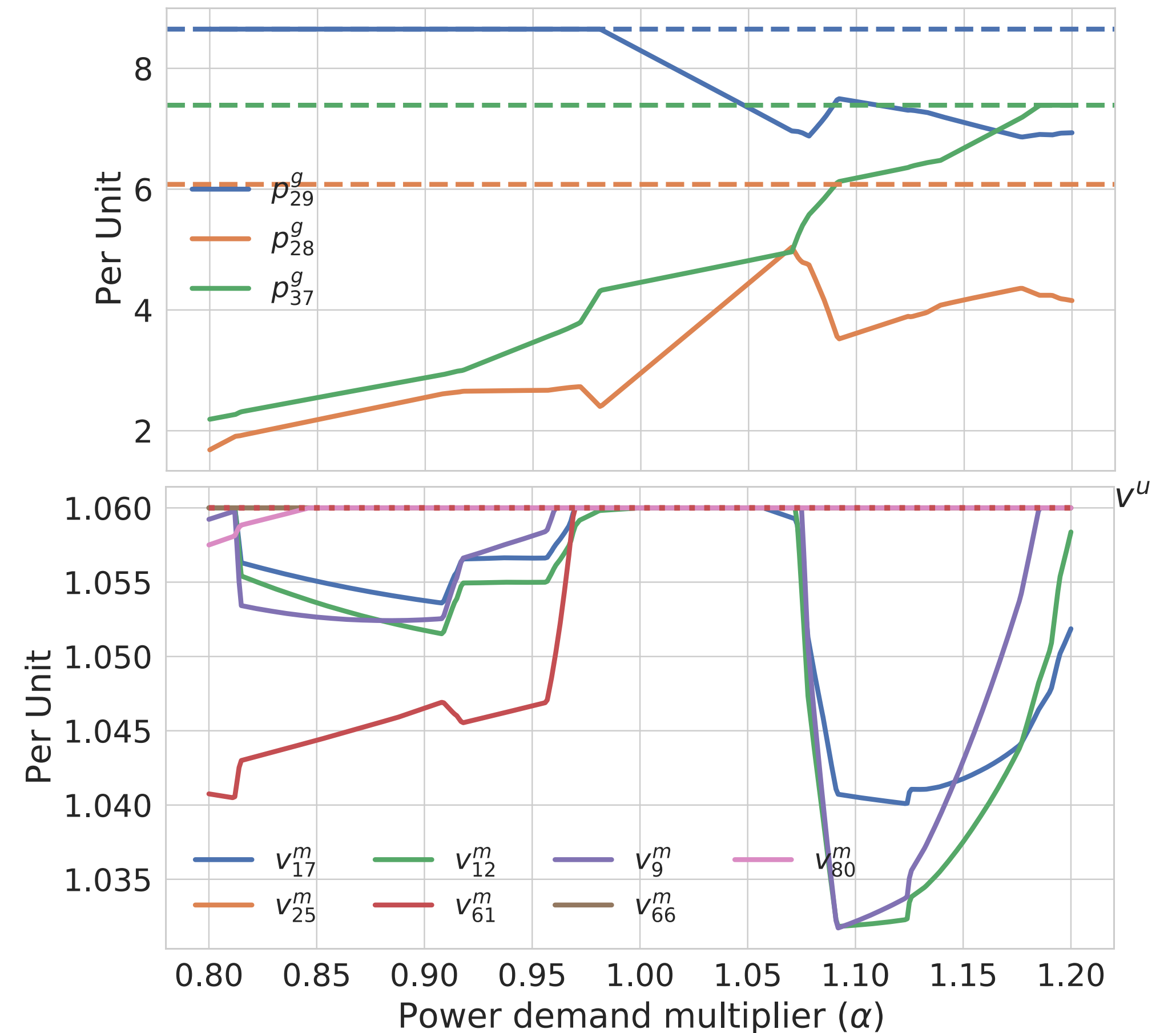
- Solution trajectories can be approximated by piecewise linear functions.
- ReLU neural networks have the ability to capture piecewise linear functions.
- When many variables have “simple” solution trajectories, highly accurate approximations can be obtained.
- **Thm (informal).** The approximation error of a ReLU network depends on the trajectory complexity (number of pieces and their total variations) and the network capacity.
- Dependency between complexity of the trajectories and prediction error, in some contexts, regardless of the model capacity.



Why does it work?

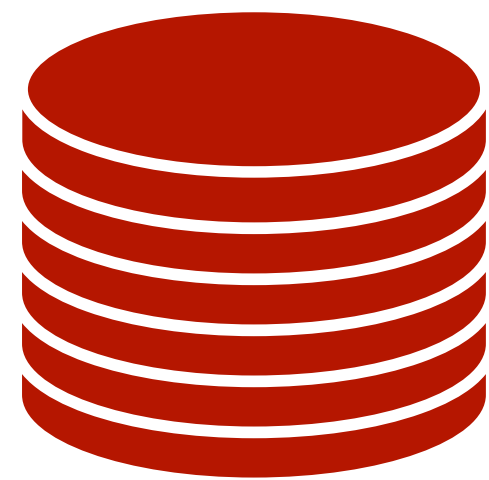
The importance of modeling constraints

- Introducing constraints using Lagrangian-based penalties is not a regularization term.
- It helps the model accurately learn different hidden features, i.e., to more accurately capture the dependencies across variables and their outputs.



Privacy and Security Concerns

$$\mathcal{O}(\mathbf{D}) = \operatorname{argmin}_x f(x, \mathbf{D})$$
$$\text{s.t. } g(x, \mathbf{D}) \leq 0$$

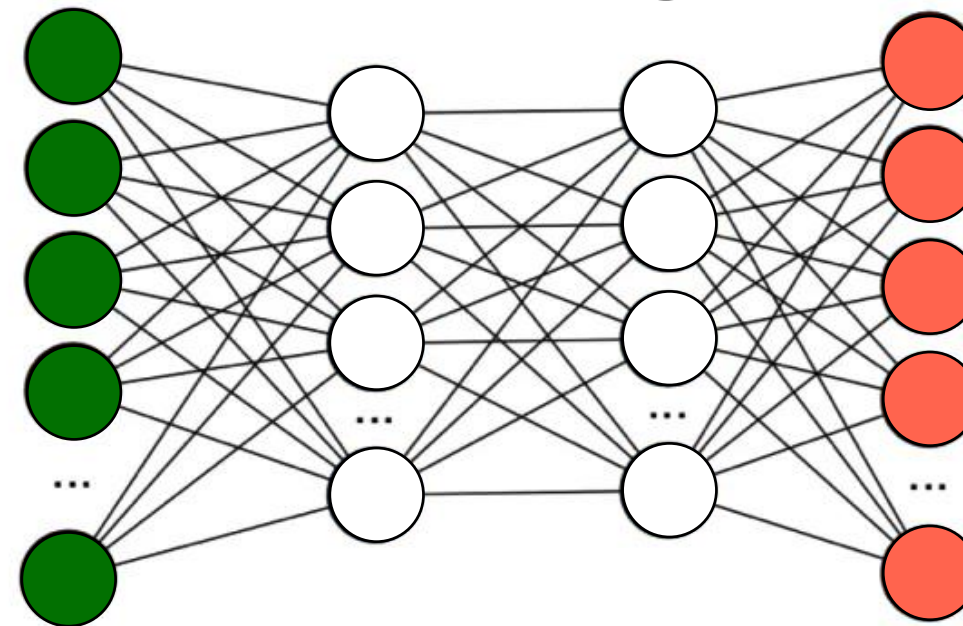


Sensitive Data

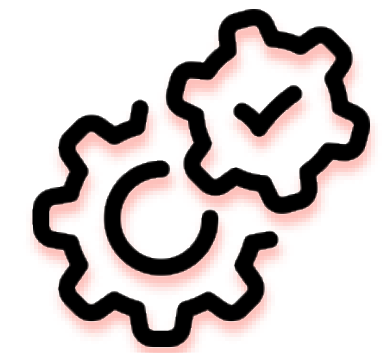
Differential
Privacy



Training

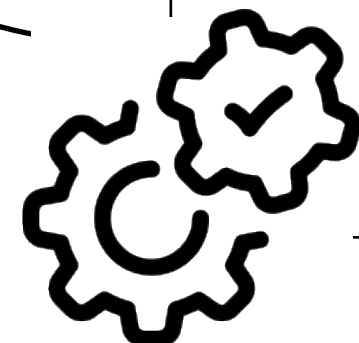


Approximate
Optimizer



- Privacy regulations
- National security

$$\min_{\hat{\mathbf{D}}, x} \|\hat{\mathbf{D}} - \tilde{\mathbf{D}}\|^2$$
$$\text{s.t. } |f(x, \hat{\mathbf{D}}) - f^*| \leq \alpha$$
$$g(x, \hat{\mathbf{D}}) \leq 0$$



Opportunities and Challenges

- Robustness guarantees about the solutions generated.
- Integration of physics simulators in the loop.
- Scalability and data availability.
- Lack of theoretical results.

Thank You

nandofioretto.com   twitter.com/nandofioretto

nandofioretto@gmail.com   github.com/nandofioretto