Data Analytics and Machine Learning Group
Department of Informatics
Technical University of Munich

Generalization of Neural Combinatorial Solvers Through the Lens of Adversarial Robustness

Simon Geisler*, Johanna Sommer*, Jan Schuchardt, Aleksandar Bojchevski and Stephan Günnemann

tl;dr: Neural Combinatorial Solvers are highly vulnerable to small perturbations of the problem instances

- We study **adversarial robustness** of neural combinatorial solvers
- We propose **sound and efficient** perturbation models for SAT and TSP
- Adversarial robustness is a more realistic **evaluation procedure** to measure local generalization

Generalization of Neural Combinatorial Solvers

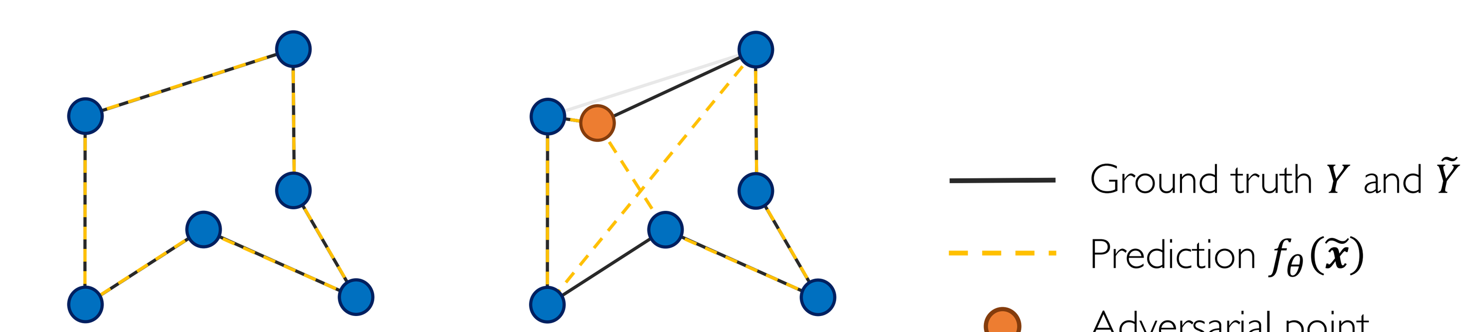
Goal: learn how to solve a wide range of combinatorial problems

- Learn from small problem instances and generalize to large problem instances
- Learn from randomly generated data and generalize to different domains

Adversarial Robustness

Find a perturbed problem instance $\tilde{\boldsymbol{x}}$ s.t. the prediction $\hat{Y} = f_\theta(\tilde{\boldsymbol{x}})$ is very different from the new optimal solution $\tilde{Y}$ (optionally with locality constraint between $\tilde{\boldsymbol{x}}$ and the original problem instance $\boldsymbol{x}$ with optimal solution Y).

- $Y \neq \tilde{Y}$ in the general case, even for perturbations with a very small budget!
- $\tilde{Y}$ should be obtained exactly and efficiently → **sound and efficient**

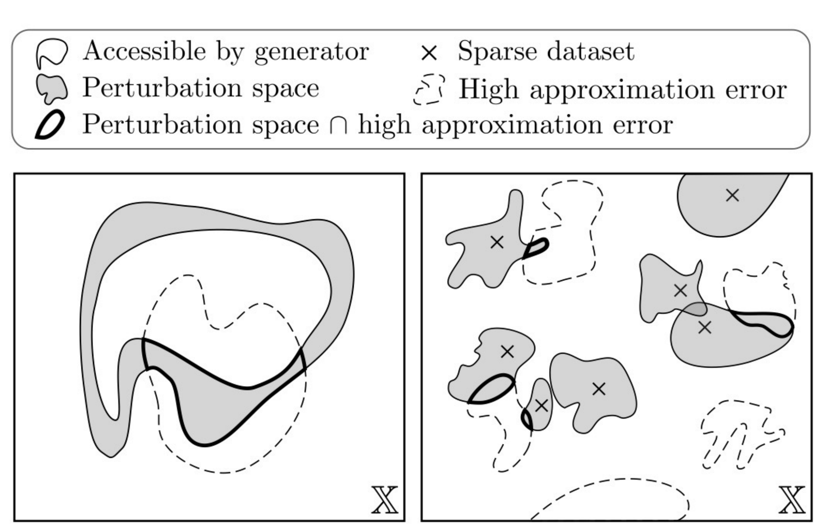


Original problem instance $\boldsymbol{x}$
optimal tour length: 270

Perturbed problem instance $\tilde{\boldsymbol{x}}$
optimal tour length: 350

Adversarial attacks can alleviate some of the challenges faced during dataset construction:

- An *efficient* data generator must be *incomplete* and a *complete* data generator must be *inefficient*
- A dataset with high coverage of the problem space is costly



Satisfiability of Boolean Expressions (SAT)

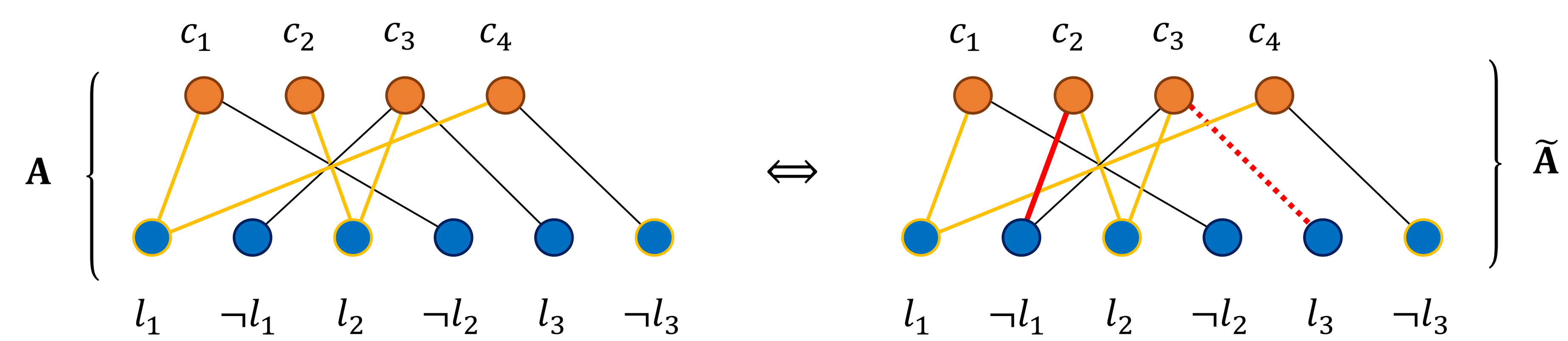
We construct the perturbed problem instance $\tilde{\boldsymbol{x}}$ s.t. it retains SATisfiability / UNSATisfiability

SAT Invariance

Removing or adding literals retains satisfiability if one literal in Y remains in each clause.

$$[l_1 \vee \neg l_2] \wedge [l_2] \wedge [\neg l_1 \vee l_2 \vee l_3] \wedge [l_1 \vee \neg l_3] = \text{True if } l_1, l_2 = \text{True and } l_3 = \text{False}$$
$$= [l_1 \vee \neg l_2] \wedge [\boldsymbol{l_1} \vee l_2] \wedge [\neg l_1 \vee l_2 \vee \cancel{l_3}] \wedge [l_1 \vee \neg l_3]$$

Graph representation:



DEL Invariance

Deleting literals from a problem will retain unsatisfiability (retain one literal per clause).

$$[l_1 \vee l_2] \wedge [\neg l_1 \vee \neg l_2] \wedge [l_1 \vee \neg l_2] \wedge [\neg l_1 \vee l_2] = \text{False}$$
$$= [l_1 \vee l_2] \wedge [\neg l_1 \vee \neg l_2] \wedge [l_1 \vee \neg l_2] \wedge [\cancel{\neg l_1} \vee l_2]$$

ADC Invariance

Adding clauses to a problem will retain unsatisfiability.

$$[l_1] \wedge [\neg l_1] = \text{False} = [l_1] \wedge [\neg l_1] \wedge [\boldsymbol{l_2} \wedge \neg \boldsymbol{l_3}]$$

Traveling Salesperson Problem (TSP)

We construct the perturbed problem instance $\tilde{\boldsymbol{x}}$ s.t. we can determine the updated route $\tilde{Y}$ efficiently:

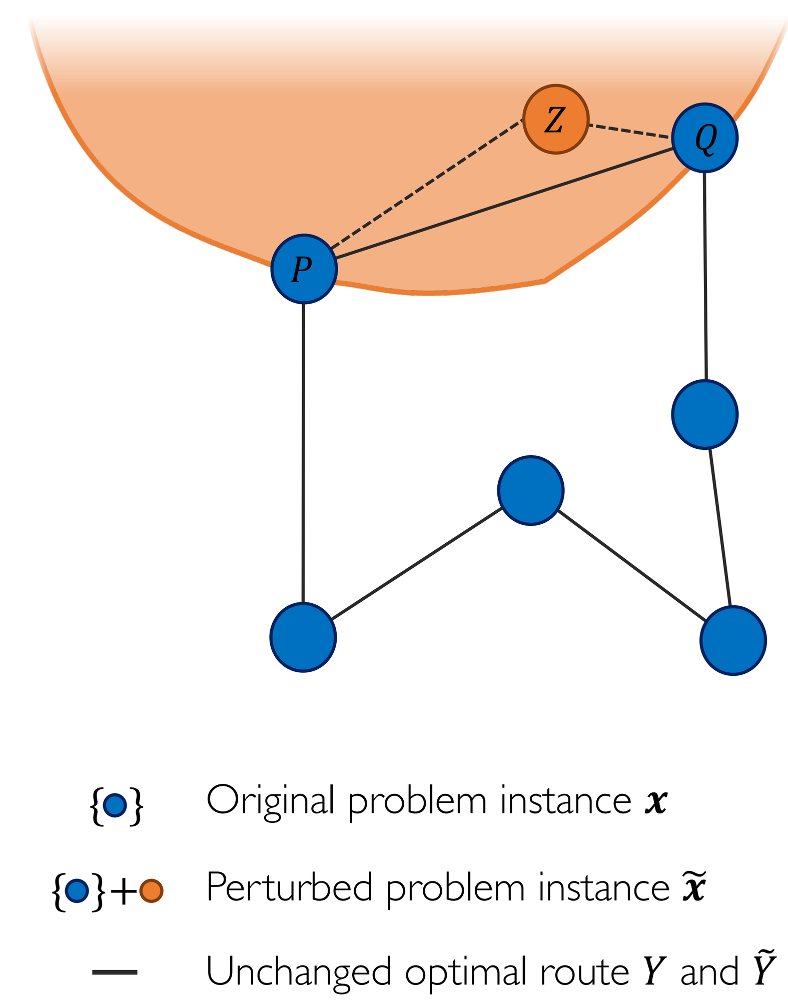
(1) Choose nodes P and Q that are **neighboring** on theoptimal route Y for the original problem instance $\boldsymbol{x}$.

(2) Choose **additional node** Z s.t. the **route** $P \rightarrow Z \rightarrow Q$ **is the smallest** among all pairs of nodes in $\boldsymbol{x}$ (constraint).

→ The optimal route $\tilde{Y}$ of $\tilde{x}$ is given via inserting Z between P and Q.

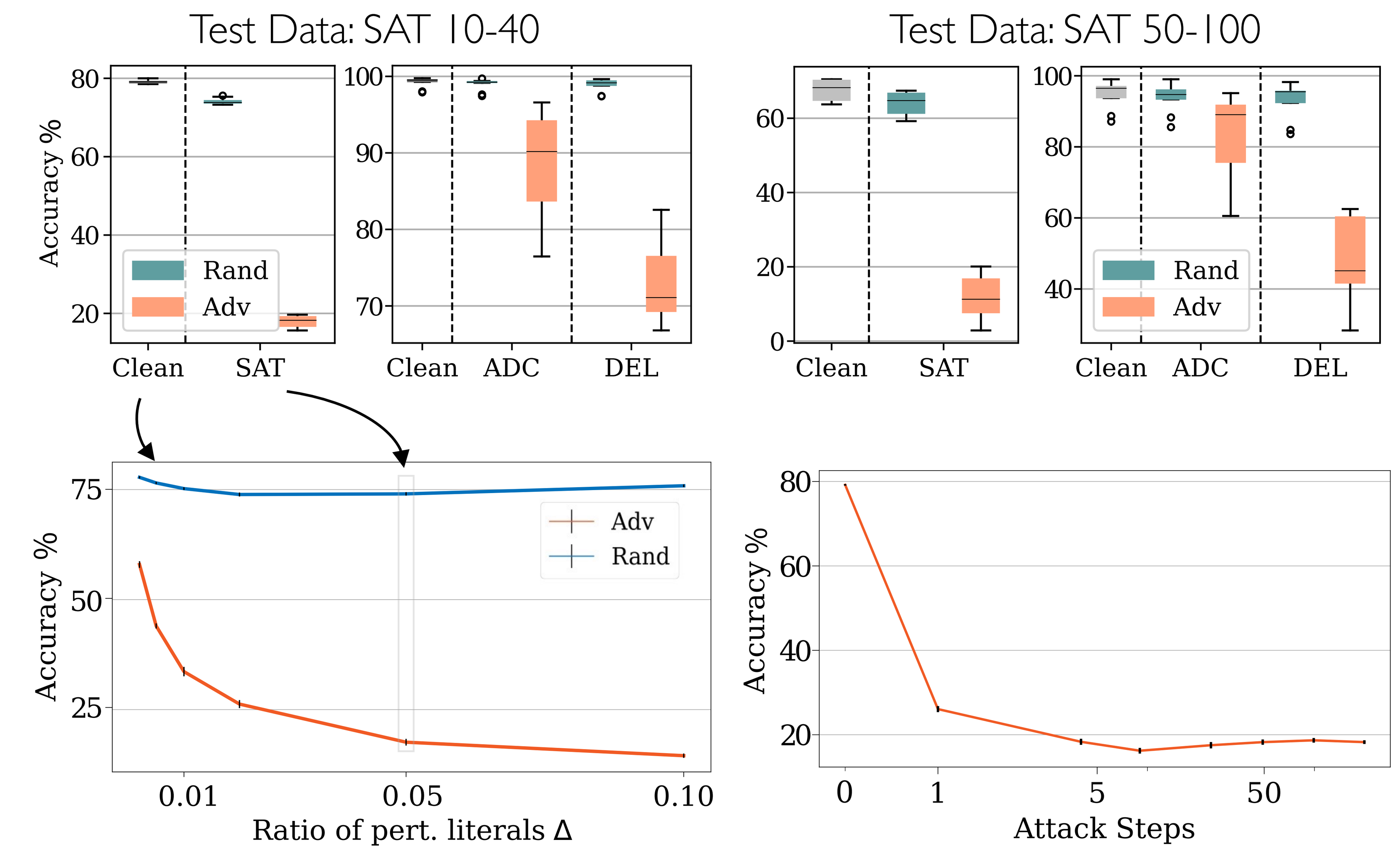
We can then use a variant of projected gradient descent to optimize over node Z's coordinates.

In the paper: How to add multiple adversarial nodes simultaneously.



Robustness of NeuroSAT

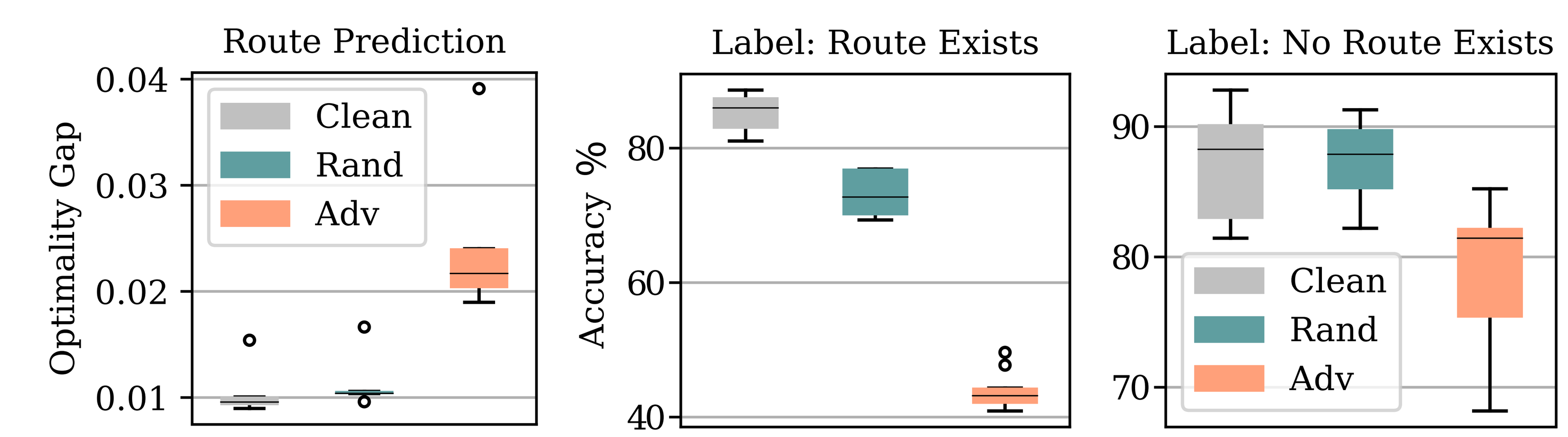
Budget: 5% of literals



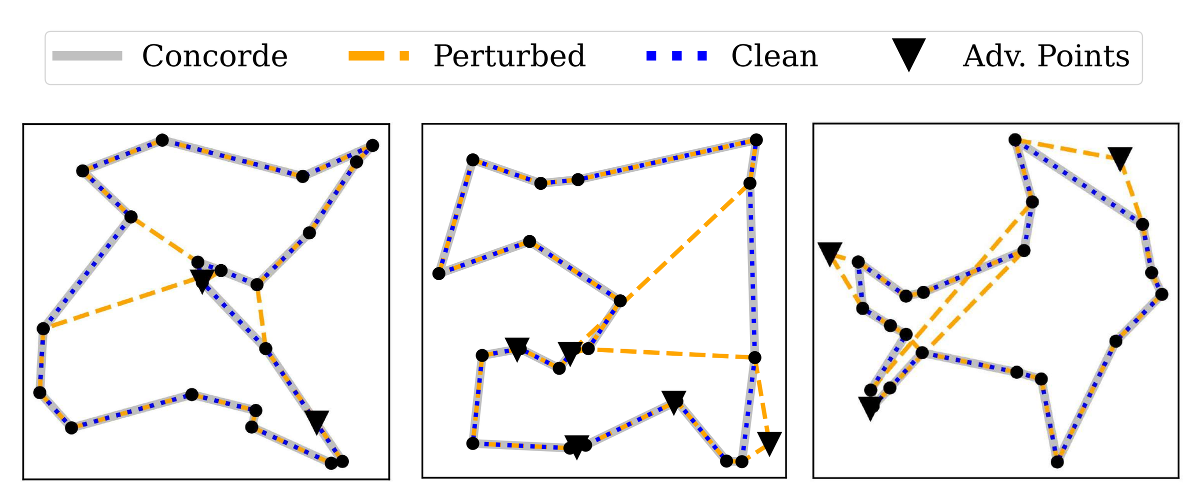
- **Our attacks are effective:** changing ~0.5% of the literals suffices to push the accuracy below 50%
- **Our attacks are efficient:** one attack step suffices to push the accuracy below 50%,

Robustness of Neural TSP Solvers

Neural TSP solvers are also **not robust to small perturbations** of the input.



↑ Route prediction, ↓ qualitative examples, and ↑ the decision problem



Project Page: