Improving robustness of jet tagging algorithms with adversarial training

Presented at the 21st International Workshop on Advanced Computing and Analysis Techniques in Physics Research
Bari, Italy, 23rd-28th October, 2022.

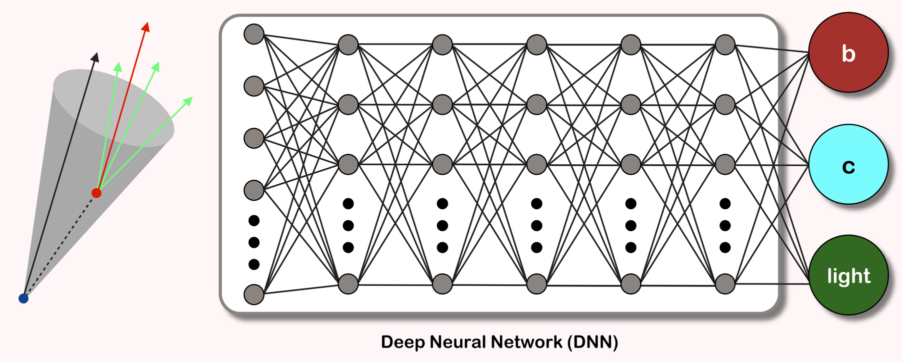
1 Physics Institute III A, RWTH AACHEN UNIVERSITY 2 BROWN

Annika Stein[1], Xavier Coubez[1,2], Spandan Mondal[1], Andrzej Novak[1], Alexander Schmidt[1]

Probing vulnerability of a nominal jet tagging algorithm with the Fast Gradient Sign Method (FGSM)

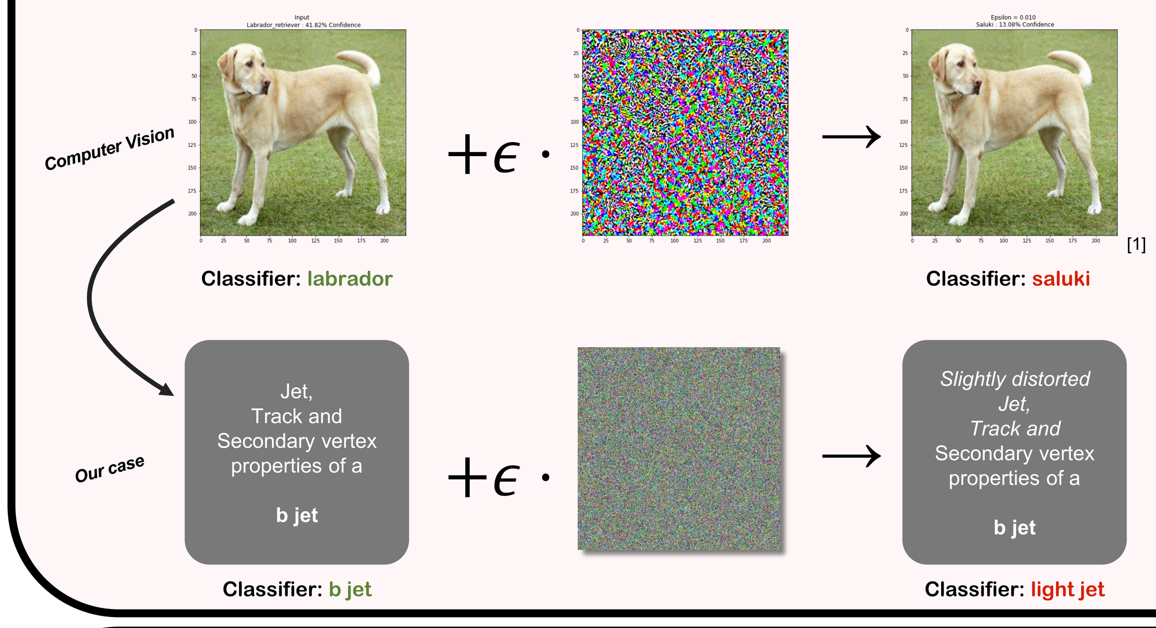
Goal of jet tagging algorithms: **identify flavor** of a jet's initiating particle (quark, gluon).

Exploit **deep learning** techniques, reliant on **accurate simulation!**

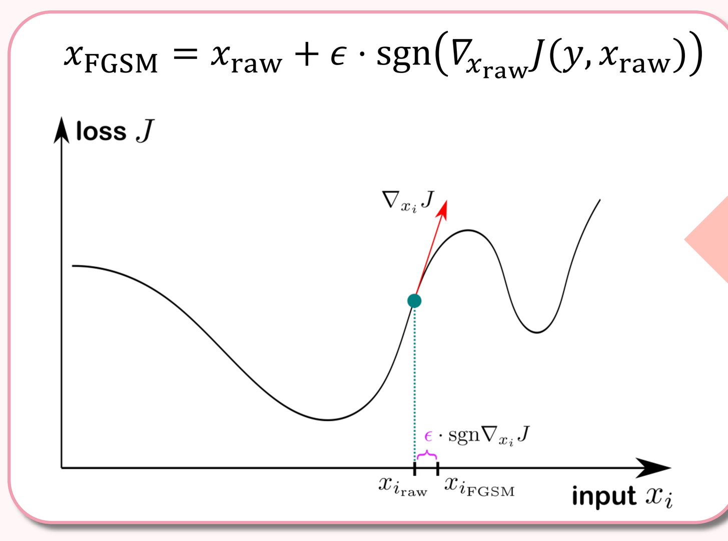


Physics analysis: Can validate each 1D input distribution within uncertainties. But what about mismodeled correlations?

Benchmark problem: apply **adversarial attacks** (e.g. FGSM) on inputs ➔ Introduce **"invisible" mismodelings**.



- **Fast Gradient Sign Method** maximizes loss function (with respect to inputs) ➔ worst-case scenario (~first order)



Attack!

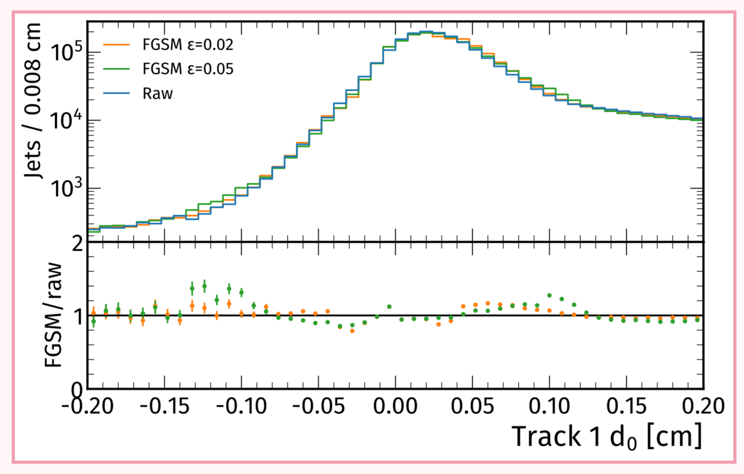
Effect

- **Drastic** effect on performance — yet only **minimal changes of the input features: Mimics invisible mismodelings!**
- Impact on input variables bound to 20%, discrepancies within uncertainties ⇒ **1D input distributions look "normal"**

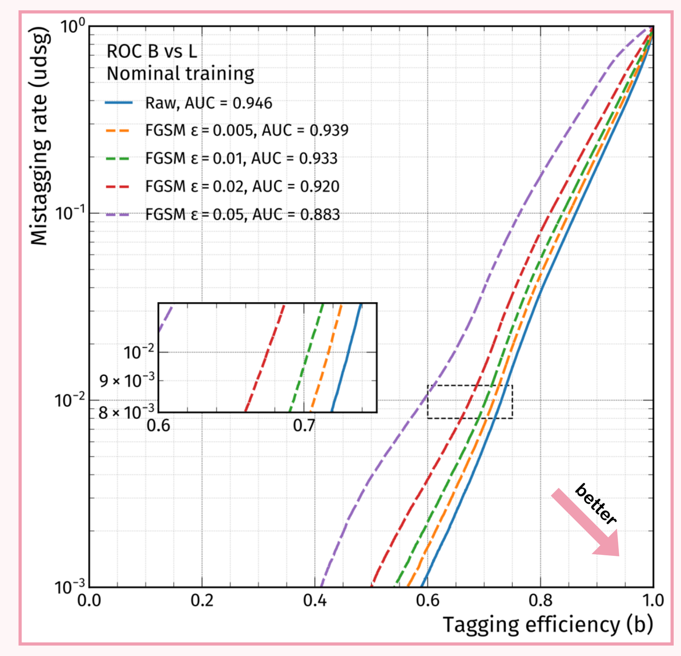
Example:

input feature

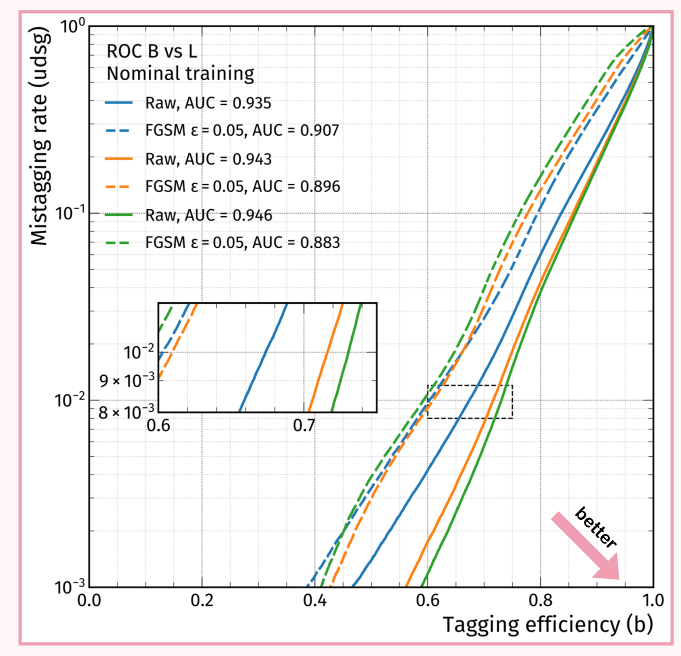
(signed impact parameter of the first track, in transverse plane)



- **Best performance** on **raw** samples
- The **larger** ϵ, the larger the **impact** on model performance



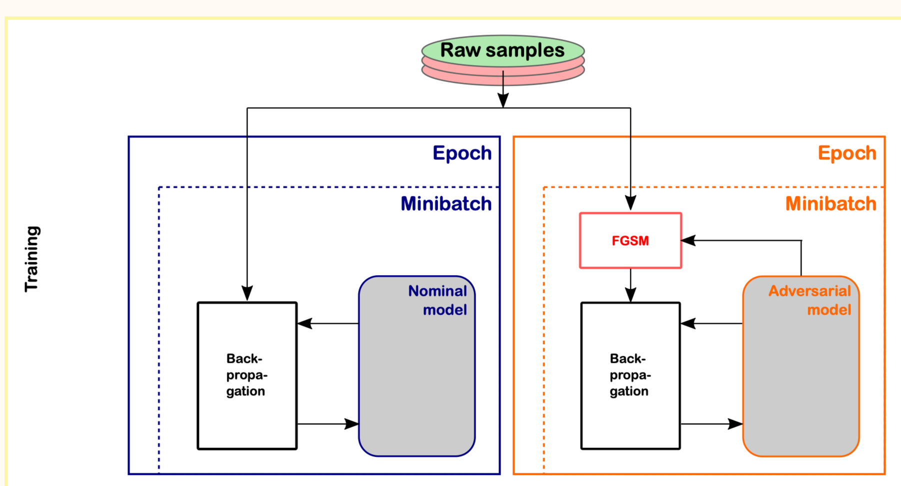
- More **training epochs** lead to **better performance** — but at the same time, the **susceptibility** towards adversarial attacks increases as well!
- Observe a **trade-off** between **performance** and **robustness**!
- Increased **gap** between raw performance (solid lines) and performance on distorted samples (dashed lines)



Adversarial training as a defense strategy

- **Inject distorted inputs already during training phase**
- **Idea**: model **never sees raw inputs** ➔ less likely to learn simulation-specific artefacts

```
FOR N EPOCHS:
    SPLIT WHOLE TRAINING SAMPLE INTO MINIBATCHES
    FOR EVERY MINIBATCH:
        DISTORT INPUTS ( = APPLY FGSM)
        EVALUATE MODEL (FORWARD)
        COMPUTE LOSS (AND APPLY LOSS WEIGHTING)
        ACCUMULATE GRADIENTS OF LOSS (BACKWARD)
        UPDATE MODEL PARAMETERS
```



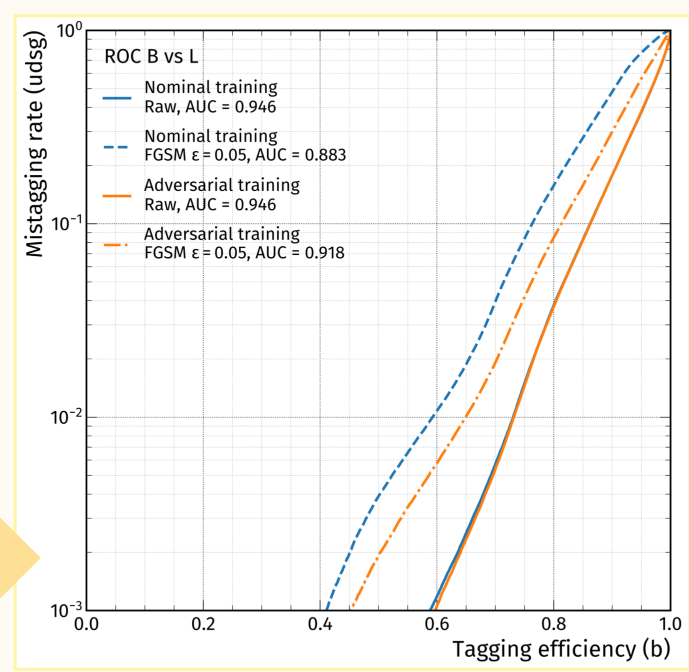
Comparison of nominal and adversarial **training** strategy
➔ difference: **FGSM prior to backpropagation**

- Expect higher **robustness** and better **generalization** by introducing a saddle point problem — so, let's check if that is indeed the case!
- **Evaluation** compares predictions of two trainings for nominal and systematically distorted test samples

Robustness against FGSM attacks:

(Attacks individually generated to cause worst possible impact)

≈ worst-case scenario



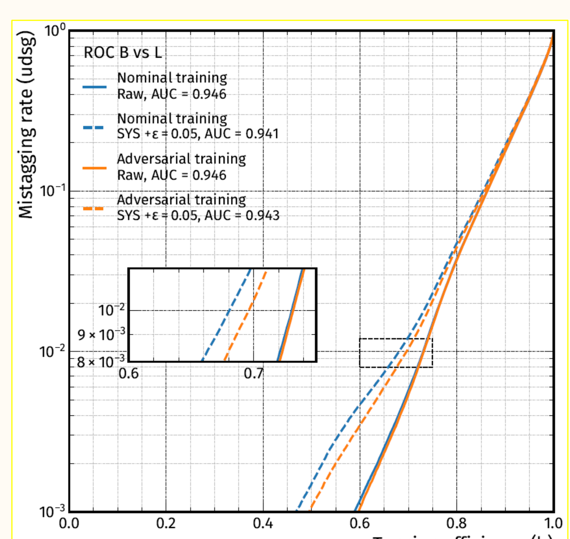
FGSM affects nominal training much more than adversarial training, with ~equal nominal performance!

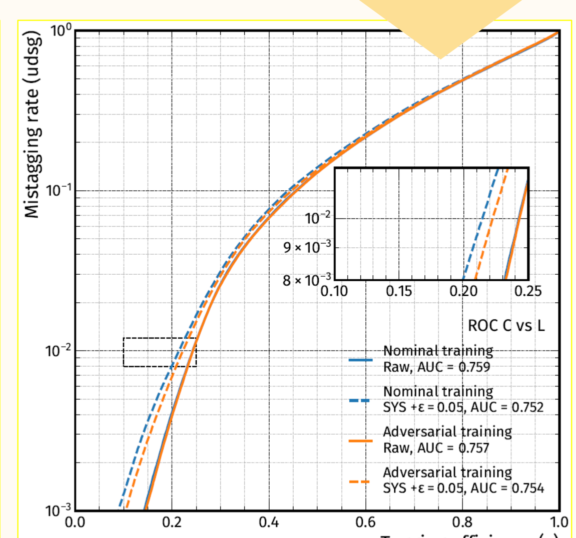
- Adversarial training **does well** on nominal samples although it has **never seen raw inputs** during training!

+ higher **robustness**, compared to nominal training

What about **usual systematic variations**?

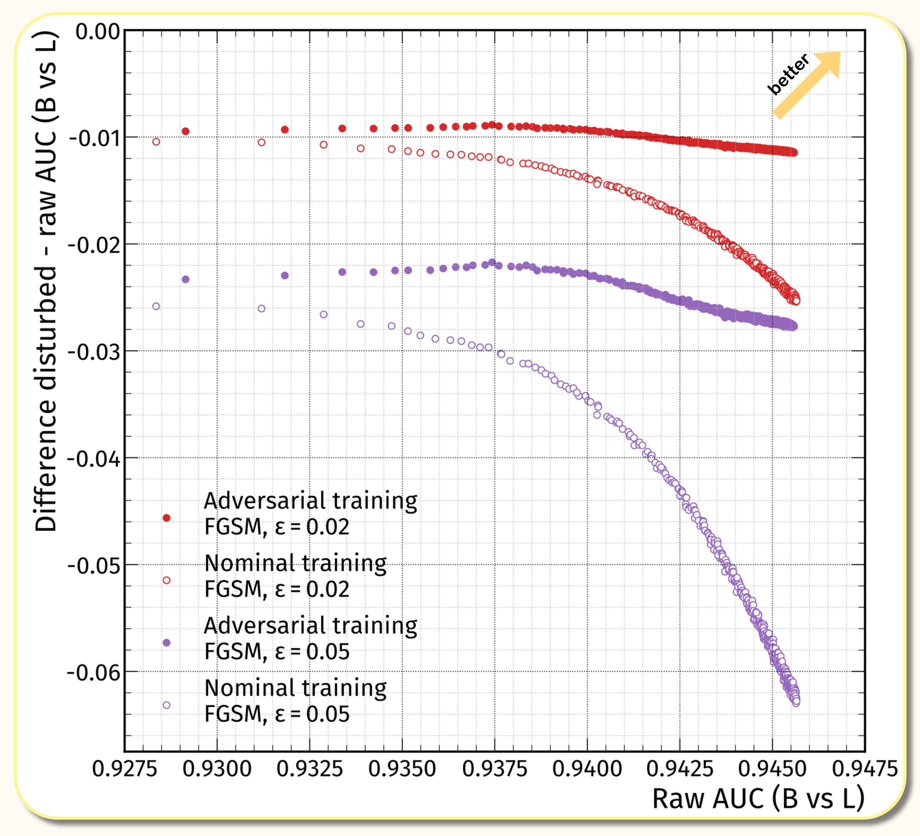
Real-istic scenario





Robustness as a function of training epochs

- Evaluate nominal and adversarial training after **several epochs / checkpoints** during training and record **raw performance** (with BvsL AUC) and **susceptibility towards adversarial attacks** (difference between disturbed and raw AUC)



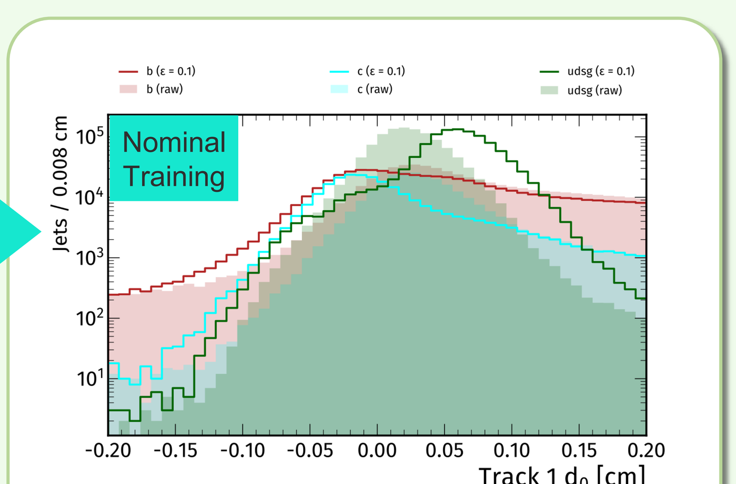
- High **density** of points at high performance: late stages of training with only small improvements, close to **convergence**
- **Nominal training**: **steep drop in robustness** towards **higher raw performance**
- **Adversarial training** maintains its **robustness even at high raw performance**, recovers robustness during training
- Trade-off is not entirely gone, but large improvement compared to nominal training

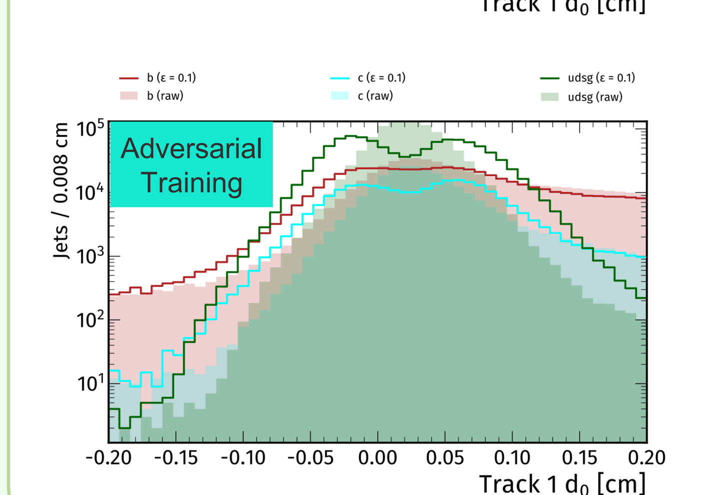
What makes the adversarial training robust? Exploring flavor dependence & geometric properties of the attack and defense

Example: d_0 of first track,

(20% distortion cap removed for visibility)

- **Raw samples**: filled histograms
- **FGSM-distorted samples**: lines





Physics:

- **b/c jets**: positive d_0 (meson secondary vertex)
- **Light jets:** d_0 peaks at zero (and is symmetric)

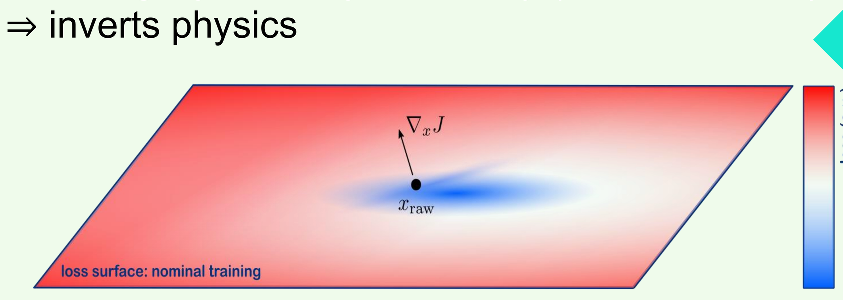
What the FGSM attack does:

- **In nominal training:** Pushes light to b/c (and vice versa)
- **In adversarial training:** Exhibits suppressed flavor-dependency

Nominal training ⊗ FGSM ➔ asymmetric shapes

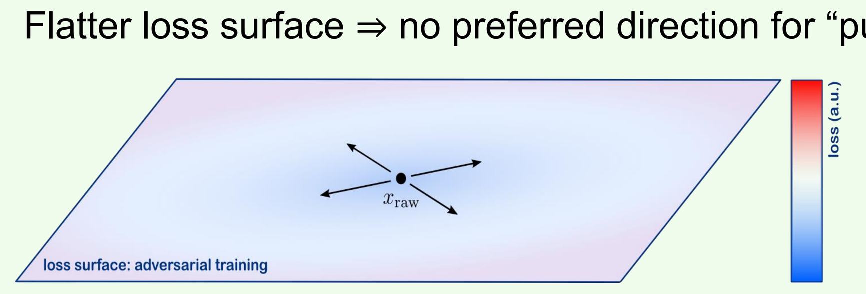
- FGSM "pushes out" from local minima of loss surface
- From "light jet" to "b jet" territory (and vice versa) ⇒ inverts physics

Illustration

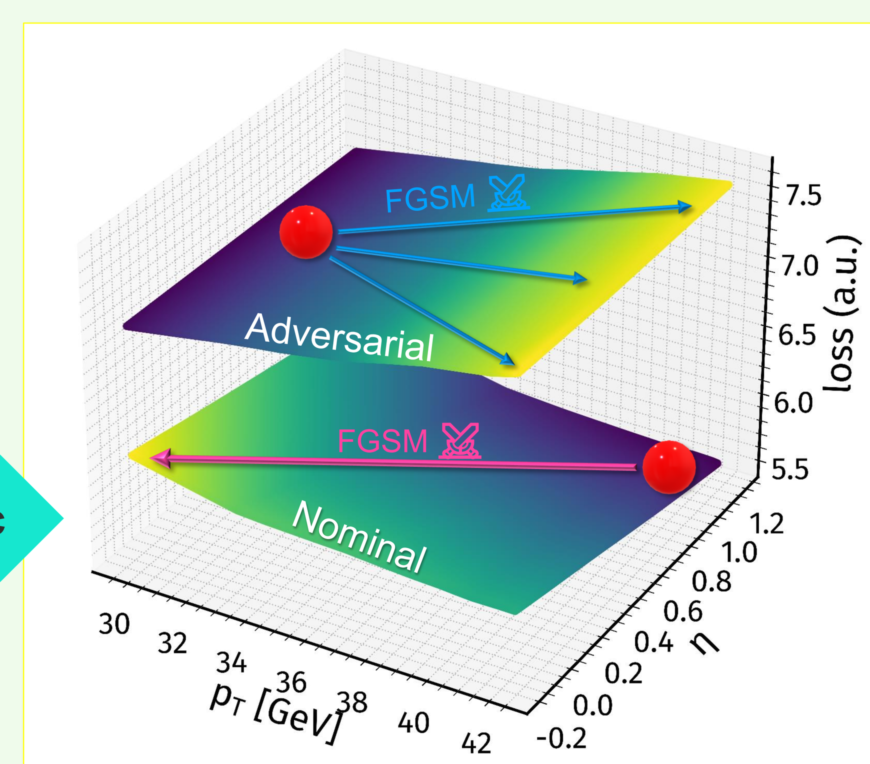


Adversarial training ⊗ FGSM ➔ symmetric shapes

- Flatter loss surface ⇒ no preferred direction for "pushing"



Realistic



- **Nominal:** Easy choice of direction for FGSM attack to "confuse" the classifier

Adversarially trained model expected to be less vulnerable to mismodelings in simulation

Conclusion

- Small disturbances of the inputs ➔ noticeable performance drops ➔ applicable & concerning for High Energy Physics
- Increased model performance comes with higher susceptibility towards adversarial attacks
- Robustness improves with adversarial training

Next steps

- Test also on detector data and investigate generalization capability
- Apply to more complex NN structures (e.g. convolutional, or graph NN)
- Check vulnerability as a function of input feature space dimension
- Use more harmful attacks and build stronger defense (e.g. train against Projected Gradient Descent, PGD)

More details in: *Comput Softw Big Sci* 6 (2022) 15

Click me!