Improving robustness of jet tagging algorithms with adversarial training

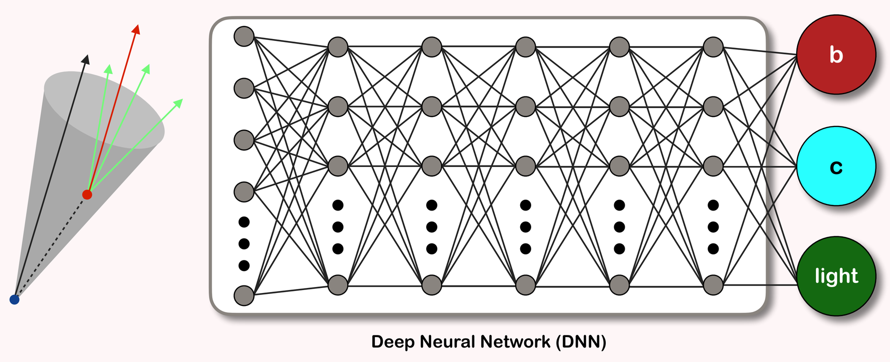
Presented at Center for Data and Computing in Natural Sciences (CDCS) Opening Symposium, 26. - 28. April 2022, Hamburg, Germany

Annika Stein, Xavier Coubez, Spandan Mondal, Andrzej Novak, Alexander Schmidt

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

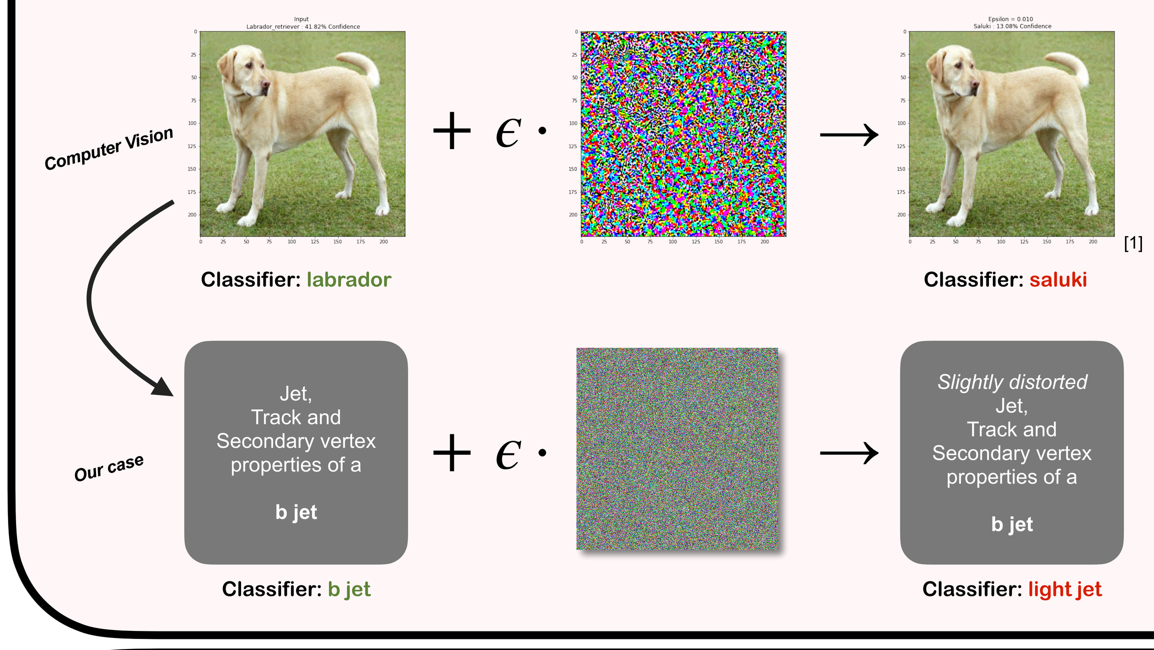
Goal of jet tagging algorithms: **identify flavor** of a jet's initiating particle (quark, gluon).

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



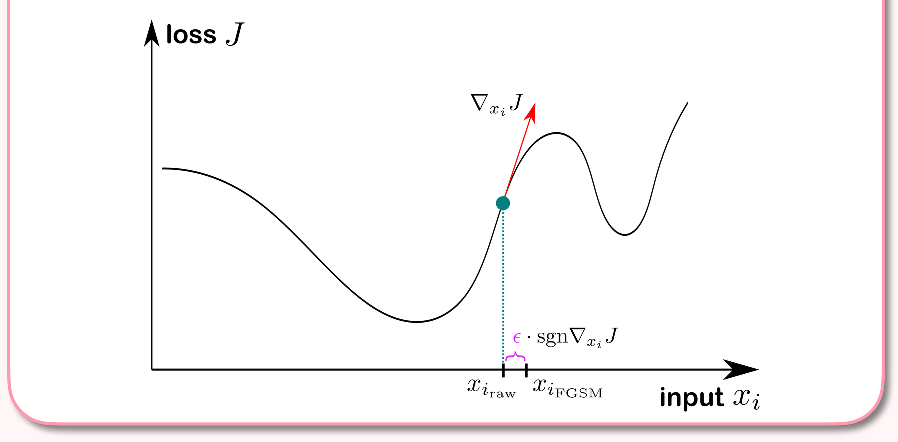
Physics analysis: evaluate tagger on measured detector **data**, requires **calibration**; but residual and invisible **mismodelings** can occur ➔ influence classifier's performance and robustness.

Benchmark problem: apply **adversarial attacks** (e.g. FGSM) on inputs ➔ investigate classifier response to injected mismodelings.



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

$$x_{\mathrm{FGSM}} = x_{\mathrm{raw}} + \epsilon \cdot \mathrm{sgn}\left(\nabla_{x_{\mathrm{raw}}} J(y, x_{\mathrm{raw}})\right)$$

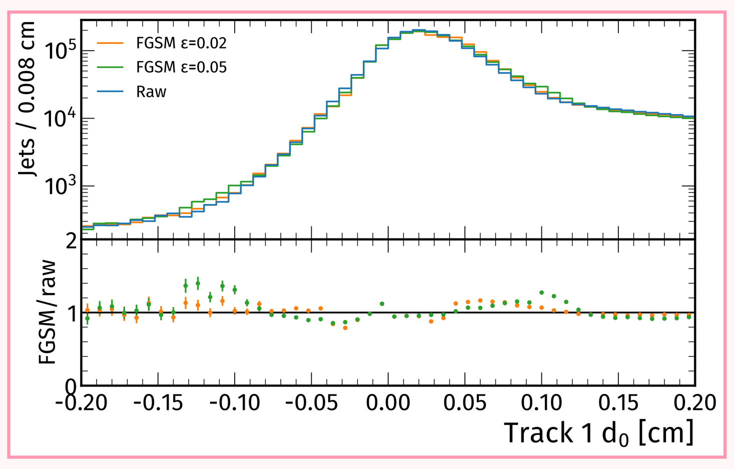


- **Systematic** and drastic effect on performance — yet only **minimal changes of the input features**
- Impact on input variables bound to 20% of the value, no discrepancies that could not be explained by uncertainties

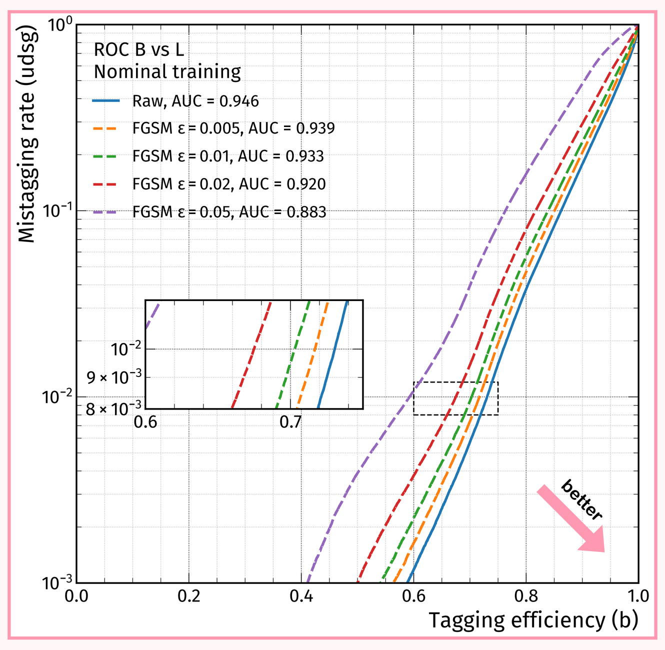
Example:

input feature

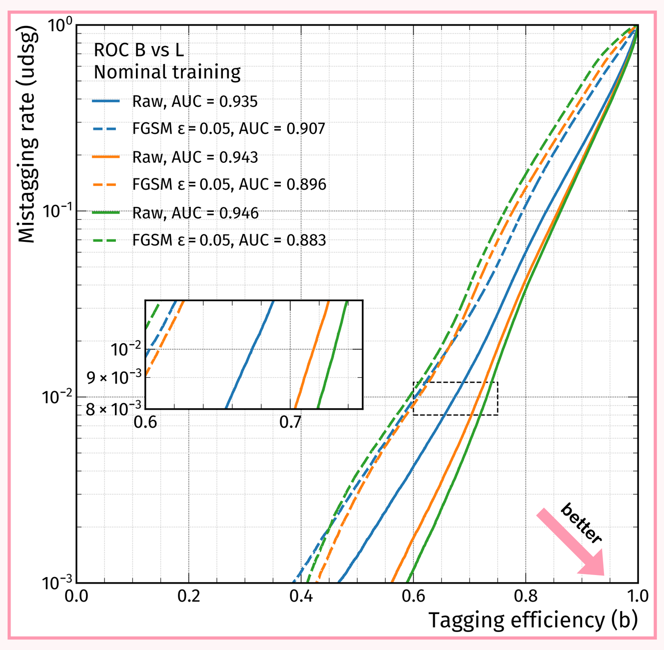
(signed impact parameter of the first track, in transverse plane)



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



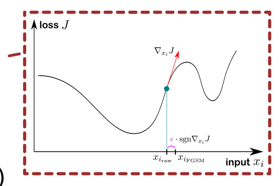
- More **training** leads to **better performance** — but at the same time, the **susceptibility** towards adversarial attacks increases as well!
- Observe a **trade-off** between **performance** and **robustness**!
- Increased **gap** between raw performance (solid lines) and performance on distorted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

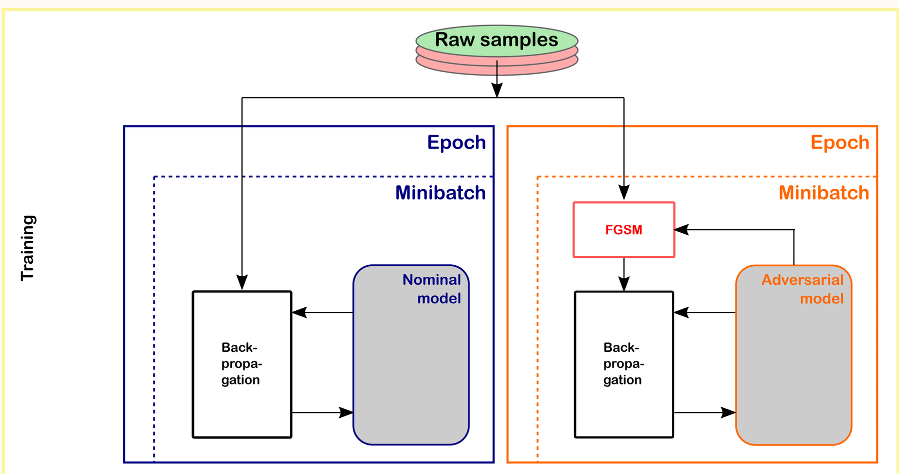


Adversarial training as a defense strategy

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

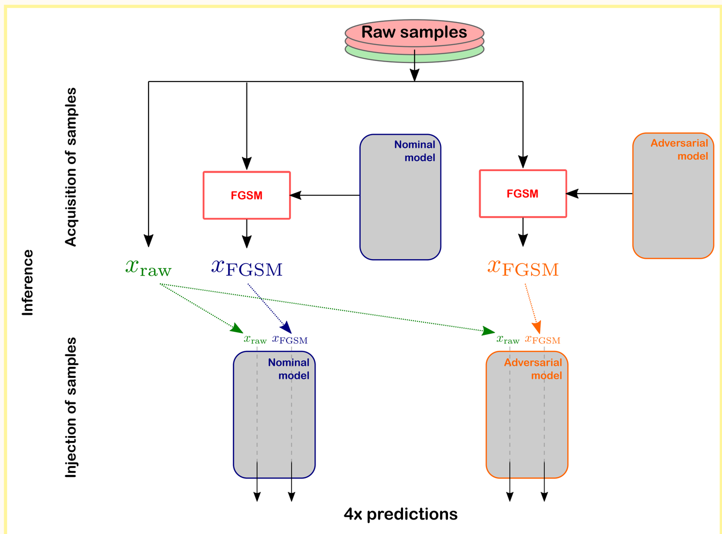
```
FOR N EPOCHS:
    SPLIT WHOLE TRAINING SAMPLE INTO MINIBATCHES
    FOR EVERY MINIBATCH:
        DISTORT INPUTS ( = APPLY FGSM)
        EVALUATE MODEL (FORWARD)
        COMPUTE LOSS (AND APPLY LOSS WEIGHTING)
        ACCUMULATE GRADIENTS OF LOSS (BACKWARD)
        UPDATE MODEL PARAMETERS
```



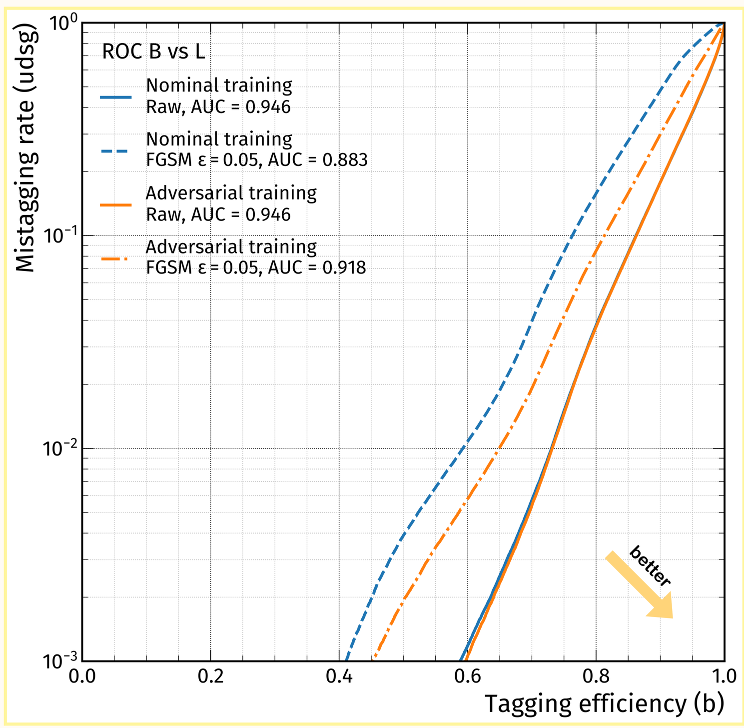


Comparison of nominal and adversarial **training** strategy ➔ difference: **FGSM prior to backpropagation**

- 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 — individually generated to cause worst possible impact (and to be fair to both contenders)
- **ROC curves** from **inference** step, after training has converged ➠



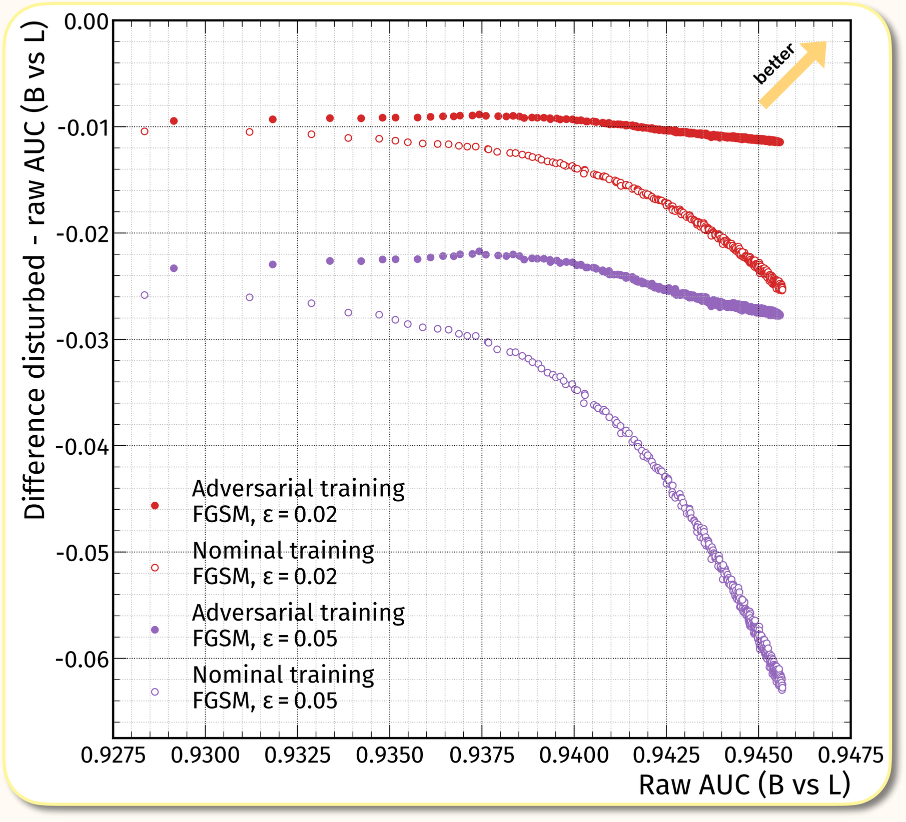
FGSM affects nominal training much more than adversarial training, with ≈ equal nominal performance!



- Adversarial training behaves **better than expected**, does well on nominal samples although it has **never seen raw inputs** during training!

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

- 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

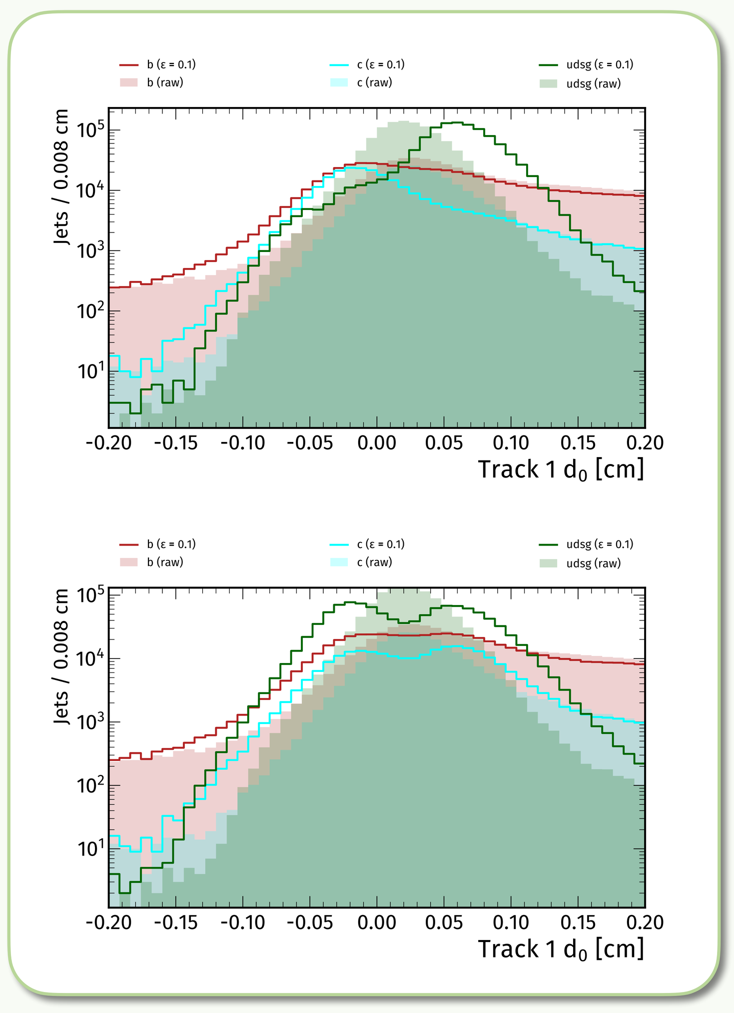
Exploring flavor dependence & geometric properties of the attack and defense, or: what makes the adversarial training robust?

Example: d_0 of first track, remove 20% cap for visibility

- **Nominal** distributions split by flavor: **filled** histograms in the background
- Systematically **distorted** samples: **lines** overlaid in foreground

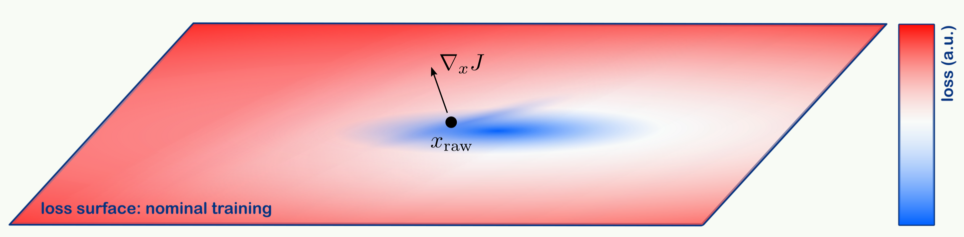
Discriminating power:

- Presence of **secondary vertex** for heavy-flavor jets ➔ displaced tracks for category b (partially also c), largest fraction in positive region
- No secondary vertex for light jets ➔ raw distribution of d_0 peaks at zero (and is symmetric)



Nominal training ⊗ FGSM ➔ asymmetric shapes

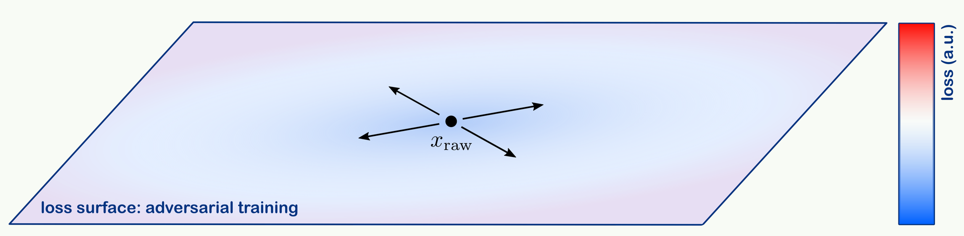
- Shifts light jets into heavy-flavor dominated region and vice-versa ➔ FGSM „inverts“ physics



- Clear direction for first-order worst-case adversarial inputs for nominal training due to geometry of the loss surface

Adversarial training ⊗ FGSM ➔ symmetric shapes

- Crafting adversarial inputs for adversarially trained model is almost like „coin-flipping“



- Assume flat loss surface ➔ no preferred direction for adversarial examples
- 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: arXiv:2203.13890

Follow the QR code! ➠

[1] Reproduced from work created and shared by Google and used according to terms described in the Creative Commons 4.0 Attribution License. (https://www.tensorflow.org/tutorials/generative/adversarial_fgsm). Labrador Retriever by Mirko CC-BY-SA 3.0 from Wikimedia Commons.