PyData
Delhi 2018

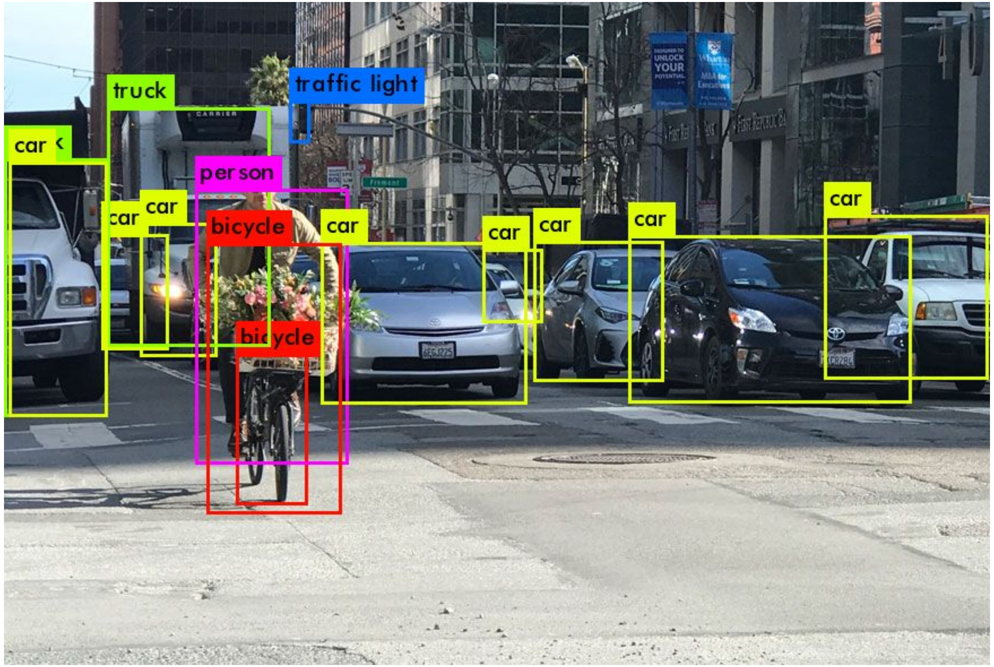
Fooling and Protecting Deep learning models

Divyam Madaan

Twitter: @divyam3897
Github: @divyam3897

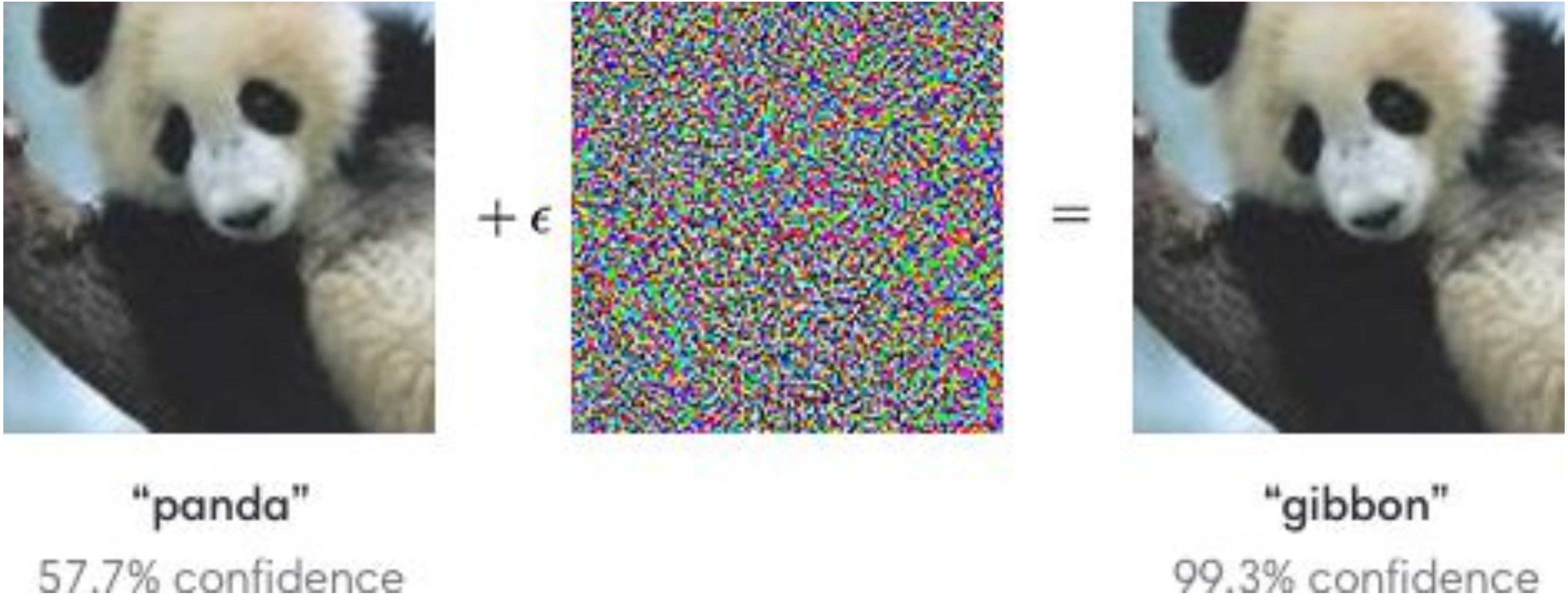
About me

- Final year undergraduate
- Deep Learning Researcher at FOR.ai
- Working on Adversarial Training and Robustness

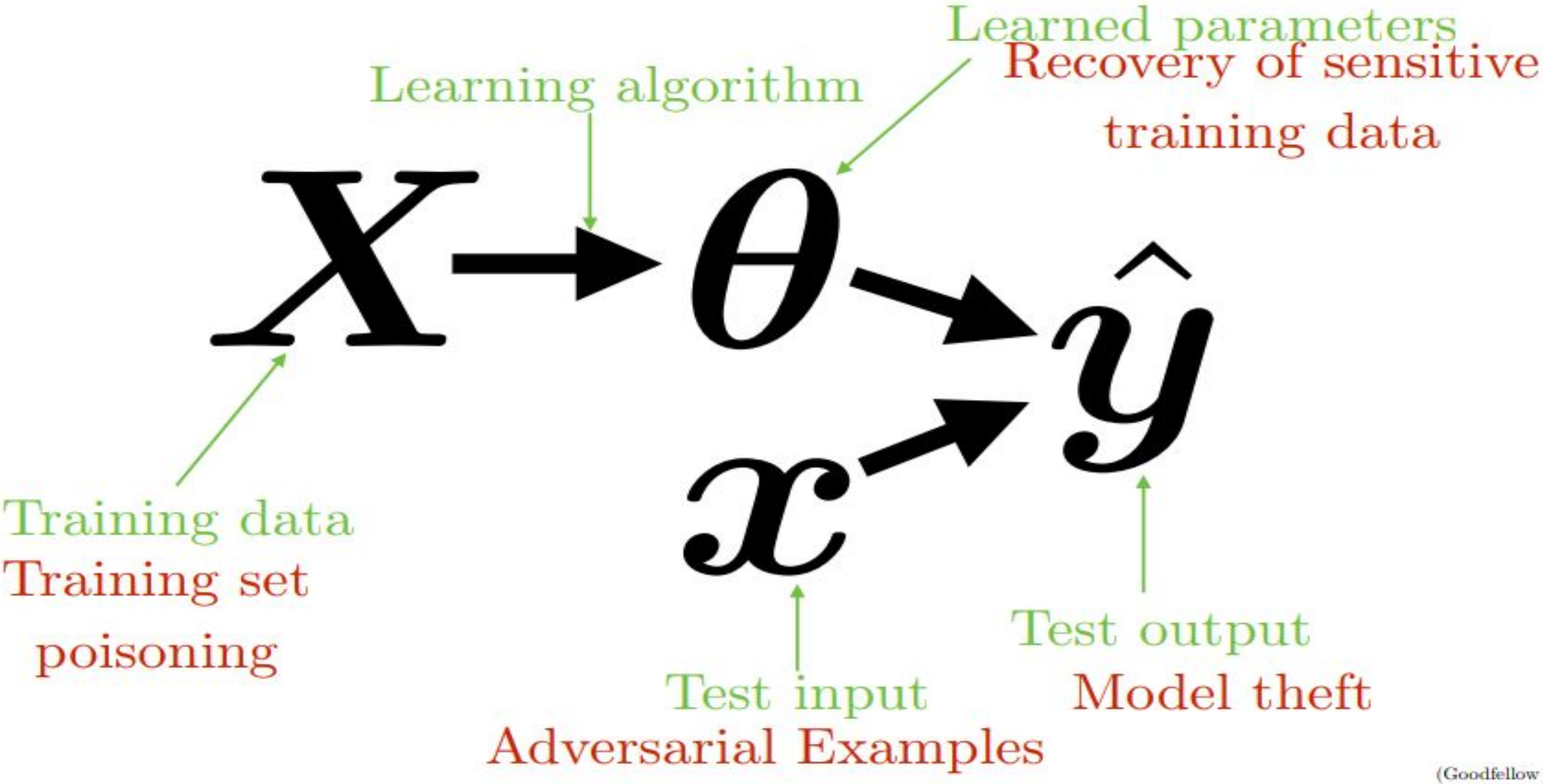


Machine learning reached state of the art on various tasks

Good models make mistakes



Attack on the machine learning pipeline



Learned parameters

Recovery of sensitive training data

Learning algorithm

$\boldsymbol{X} \rightarrow \boldsymbol{\theta} \rightarrow \hat{\boldsymbol{y}}$

$\boldsymbol{x} \rightarrow \hat{\boldsymbol{y}}$

Training data

Training set poisoning

Test input

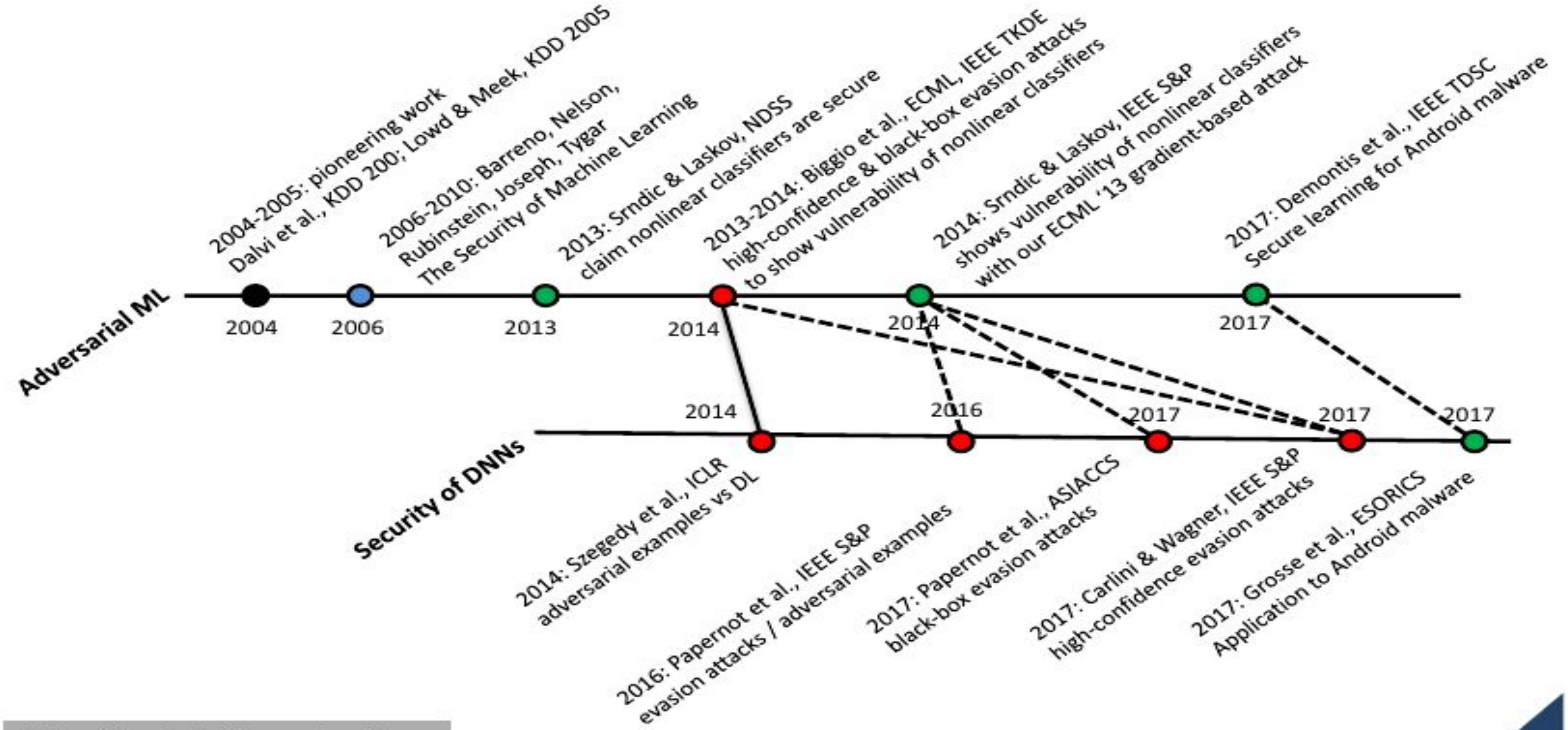
Adversarial Examples

Test output

Model theft

(Goodfellow 2018)

Timeline of Learning Security



Adversarial ML

2004-2005: pioneering work
Dalvi et al., KDD 200; Lowd & Meek, KDD 2005

2006-2010: Barreno, Nelson, Rubinstein, Joseph, Tygar
The Security of Machine Learning

2013: Srndic & Laskov, NDSS
claim nonlinear classifiers are secure

2013-2014: Biggio et al., ECML, IEEE TKDE
high-confidence & black-box evasion attacks
to show vulnerability of nonlinear classifiers

2014: Srndic & Laskov, IEEE S&P
shows vulnerability of nonlinear classifiers
with our ECML '13 gradient-based attack

2017: Demontis et al., IEEE TDSC
Secure learning for Android malware

Security of DNNs

2014: Szegedy et al., ICLR
adversarial examples vs DL

2016: Papernot et al., IEEE S&P
evasion attacks / adversarial examples

2017: Papernot et al., ASIACCS
black-box evasion attacks

2017: Carlini & Wagner, IEEE S&P
high-confidence evasion attacks

2017: Grosse et al., ESORICS
Application to Android malware

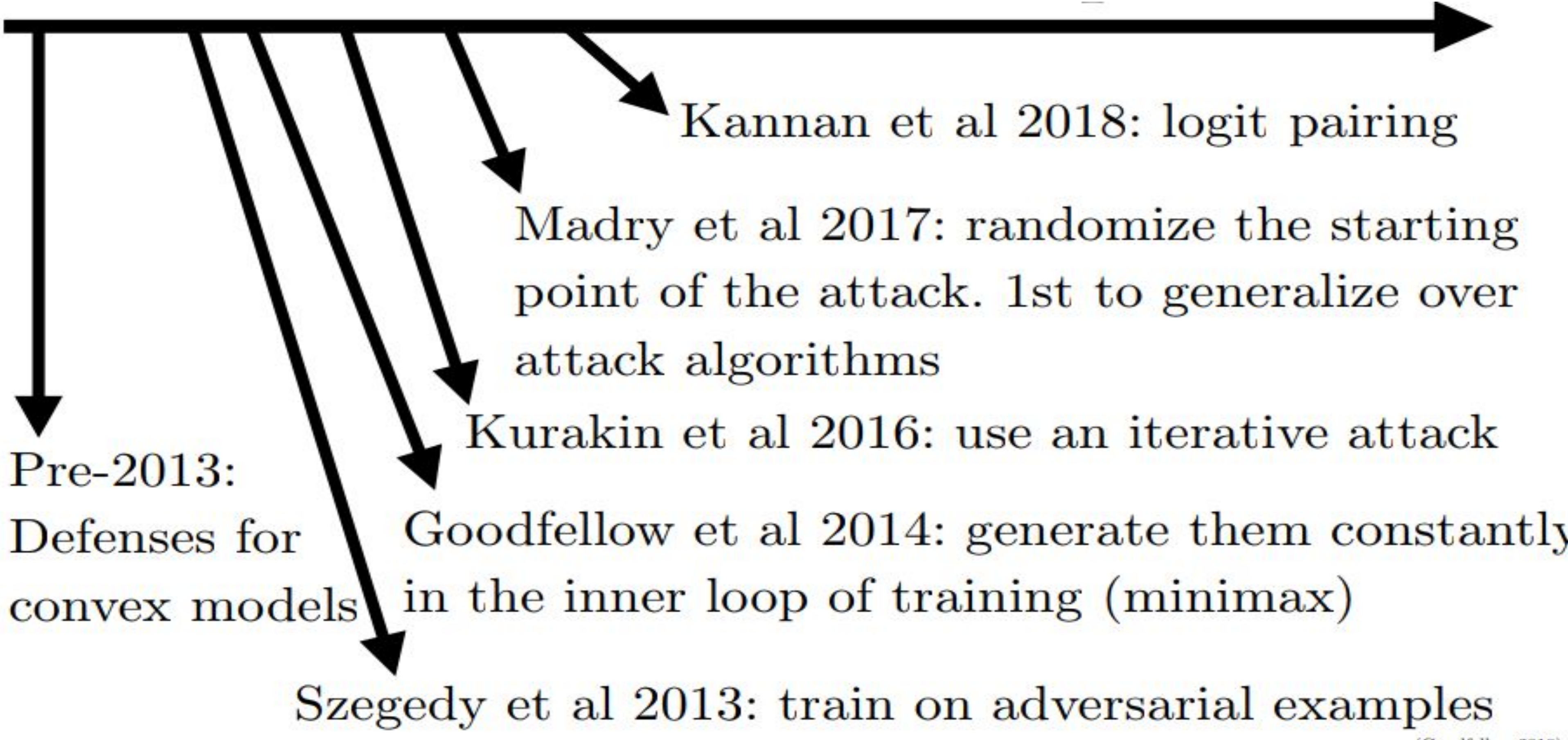
What this means for us?

- Deep learning algorithms (Machine learning in general) are susceptible to attacks
- Use with caution in critical deployments
- Evaluate a model's adversarial resilience - not just accuracy/precision/recall
- Spend effort to make model robust to tempering

Defending the machines

- Distillation (Train model 2x, feed first DNN output logits into second DNN input layer)
- Train models with adversarial samples i.e ironing out imperfect knowledge learnt in the model)
- Special regularization methods/loss functions (simulating adversarial content during training)

Timeline of Defences



Pre-2013: Defenses for convex models

Szegedy et al 2013: train on adversarial examples

Goodfellow et al 2014: generate them constantly in the inner loop of training (minimax)

Kurakin et al 2016: use an iterative attack

Madry et al 2017: randomize the starting point of the attack. 1st to generalize over attack algorithms

Kannan et al 2018: logit pairing

(Goodfellow 2018)

LIVE DEMO

Thank you
Questions?

Twitter: @divyam3897

Linkedin: https://www.linkedin.com/in/dmadaan3897

Github: @divyam3897