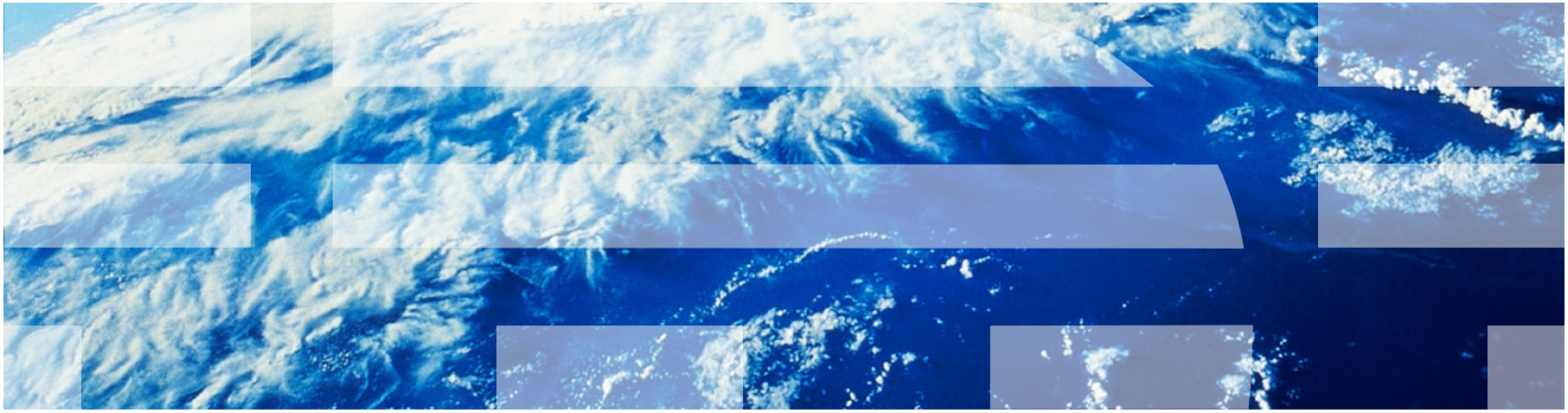
Some slides borrowed from Kexin Pei

Open Systems Challenges in ML



Many open systems/security challenges in ML

- Testing and debugging
- Version control and rolling back models
- Managing models on heterogeneous hardware
- Models leaking private information

- These are by and large open problems, no standard solutions/tools

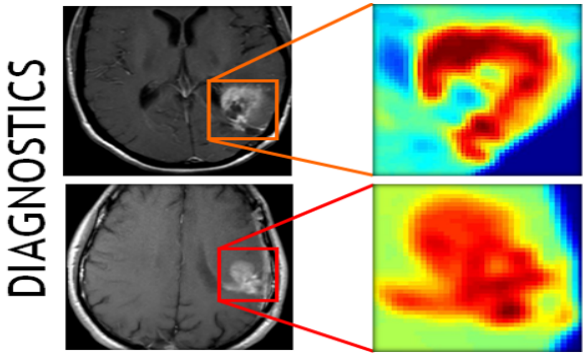
Many open systems/security challenges in ML

- **Testing and debugging**
- Version control and rolling back models
- Managing models on heterogeneous hardware
- Models leaking private information

Deep learning is used in safety-critical systems

- Deep learning correctness and security is crucial

Self-driving car



Medical diagnosis

Malware detection

Model Reliability

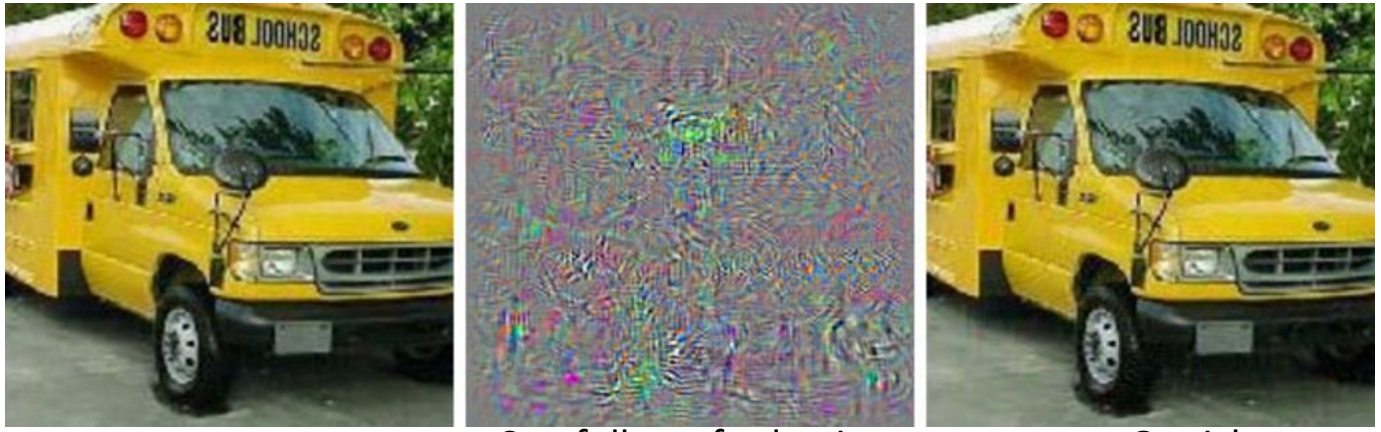
Tesla autopilot failed to recognize a white truck against bright sky leading to fatal crash

Very hard to test DL models

- Common practice: measure accuracy on a test input set of randomly chosen data
- Problem 1: how good is the coverage of the test set?
  - DL decision logic is incredibly complex
  - Are we sure we are testing all the corner cases?
- Problem 2: it requires expensive labeling effort
  - Data in test set must be manually labelled
  - To enlarge the test set, we need to manually label more data

Approach 1: Adverserial testing of DL models

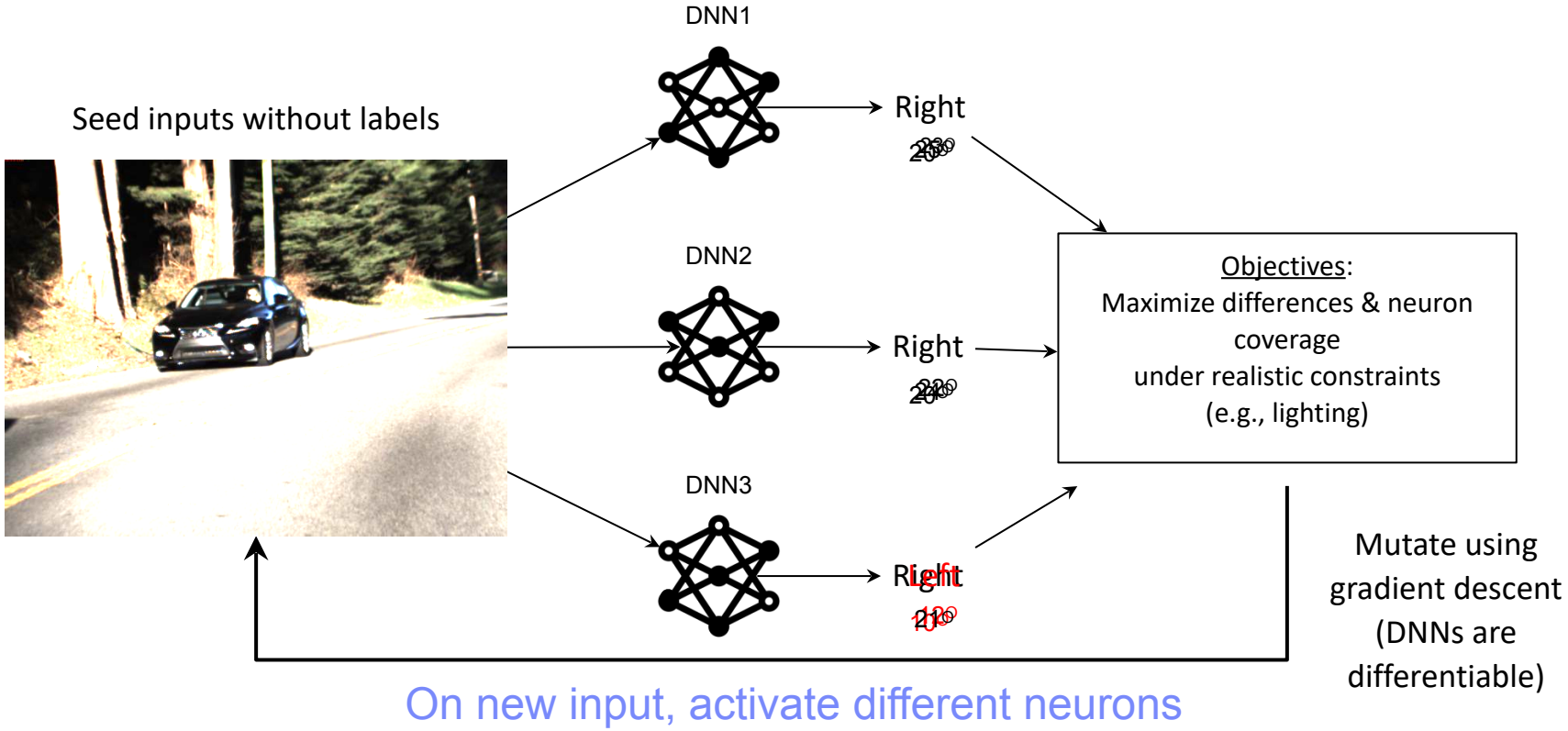
- Adversarial testing (Szegedy et al.): find corner-case inputs imperceptible to human but induce errors
  - Problem 1: how good is the coverage of the test set?
  - Problem 2: it requires expensive labeling effort
  - Problem 3: Not realistic. (Theoretical, assumes a very powerful adversary. [Sharif et al. CCS'16])



School bus | Carefully crafted noise | Ostrich

Approach 2: Testing as an optimization

- Deepxplore, Pei et al (Columbia research project): automatically optimize the testing inputs
  - Try to **optimize input** for “neuron coverage” (similar to code coverage)
    - Exercise all neurons, so that a ”rogue” neuron doesn’t cause unexpected results
  - Use multiple models, **optimize input** to find when outputs diverge
  - Use realistic inputs



On new input, activate different neurons

Many open systems/security challenges in ML

- Testing and debugging
- **Version control and rolling back models**
- Managing models on heterogeneous hardware
- Models leaking private information

Training a model, real-world example

- Let's say you're building a model that detects phishing emails like these:

From: "Alice Smith" <asmith@acne.com>

To: "Bob Barnes" <bbarnes@acme.com>

Subject: Vendor Payment

Hey Bob,

Are you around? I need to send a wire transfer ASAP to a vendor.

Alice Smith

CEO, Acme Corporation

Cell: 408-292-2222

From: "Alice Smith" <asmith@acme.com>

Reply-to: "Alice Smith" <ceo.executive@outlook.com>

To: "Bob Barnes" <bbarnes@acme.com>

Subject: At desk?

Bob, are you available for something urgent?

Models rely on a lot of “feature engineering” and specific data points

- Some features might be:
  - Is the email address of the sender different than the normal email address this person sends from?
  - Is the reply-to email address different from the sender email?
  - Is the email sent in a non-”normal” hour?
- Small tweaks to these features can substantially change the accuracy
- You don’t need to be careful only about the features, but also about data points to include in the training
  - Including a specific email example might change the accuracy significantly
  - For example, a rare (but important) example of a phishing email

Models keep evolving, hard to keep track of their versions

- New versions of models add new features
  - And new data
- Over time, it's hard to track/predict the importance of different features/datapoints
  - There's no record of old model performance
  - Which data was used to train old models
  - Difficult to reproduce results
  - Difficult to collaborate (new person doesn't know why a certain feature is in the model)

Same version of this problem exists with deep neural networks

- Companies often fine-tune existing open source models (e.g., Lama, Deepseek)
  - Creating many versions
- What if there is a ”bug” in the source model? How to update all versions efficiently?
- What if there is a bug in one of the fine-tuned models? How to uncover it and update only the relevant “branch”

Many open systems/security challenges in ML

- Testing and debugging
- Version control and rolling back models
- **Managing models on heterogeneous hardware**
- Models leaking private information

Models are increasingly being deployed on the “edge”

- Many time sensitive ML tasks need to occur on mobile devices / cameras / sensors rather than on the cloud
  - Google Translate
  - Self-driving cars
  - Face recognition for security cameras
  - Object recognition for drones
  - …

Inference accuracy and performance differs greatly across devices

- Inference accuracy varies widely across devices
  - Different devices apply different quantization
  - Different apply different compression on photos/videos
- Inference performance varies widely across devices
  - Inference can vary 10x or more between high end cloud GPU/TPU and GPU/TPU on edge device
- Need to account for power consumption

Challenges in managing edge models

- Does my model have lower accuracy on the edge?
- Is my model too slow on the edge?
- Version control for many models
- Training different models for different devices

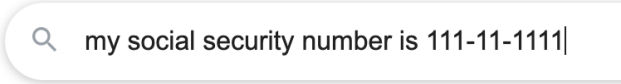
Many open systems/security challenges in ML

- Testing and debugging
- Version control and rolling back models
- Managing models on heterogeneous hardware
- **Models leaking private information**

Models can leak private information on individual users

- Models are thought to be privacy preserving, since they aggregate on a lot of data
- But there are many examples that have shown models can leak information

- Generative text models
  - Auto-completing ”My social security number is”



- Recommender models
  - If you have some auxiliary information of what a specific user did in the past, you can reconstruct what they bought/watched
- Classification models
  - Reconstruct whether a user is a member of a group (e.g., patients discharged from hospital)

Guaranteeing privacy with differential privacy

- Differential privacy is a technique that can guarantee a certain level of privacy [Dwork et al.]
  - An observer seeing the output of a differentially private model cannot tell if an individual's information was used in the computation
- Basic intuition: add some noise to the data or the training process so that no individual data point can be de-anonymized
  - Downside: reduces accuracy of algorithm
- Originally used for aggregating queries from a database
- Now being applied to ML
  - For example, PrivateKube [Tao et al.] (researchers from Columbia) and Tensorflow Privacy are efforts to enable differential privacy on continuously trained ML models