

# Causality and Generative Models

---

*Author: Anton Zhitomirsky*

## Contents

<b>1 Trustworthy AI/ML</b>	<b>3</b>
1.1 The need for data . . . . .	3
1.2 What are the hurdles to getting more data? . . . . .	4
1.3 Secure and Privacy-aware ML . . . . .	4
1.4 Secure and Privacy-preserving ML . . . . .	5
1.5 Federated learning . . . . .	6
1.5.1 Federated learning - SGD . . . . .	7
1.5.2 Federated Averaging . . . . .	8
1.5.3 Challenges . . . . .	9
1.6 Homomorphic Encryption . . . . .	10
1.6.1 ML in standard setting . . . . .	10
1.6.2 ML in client/server setting . . . . .	11
1.7 Homomorphic Encryption . . . . .	12
1.8 ML and Homomorphic Encryption . . . . .	12
1.9 Secure Multi-party Computation . . . . .	13
1.10 Secure Multi-Party Computation . . . . .	15
1.11 Trusted Execution Environments . . . . .	16
1.12 What is Privacy? . . . . .	16
1.13 k-anonymity . . . . .	17
1.14 Privacy attacks . . . . .	19
1.14.1 Model inversion . . . . .	19
1.14.2 Membership inference . . . . .	20
1.14.3 Attribute inference . . . . .	20
1.15 Differential Privacy . . . . .	21
1.15.1 Randomized responses . . . . .	22
1.16 Differential privacy . . . . .	23
1.17 Concrete mitigation . . . . .	25
1.17.1 Differentially private stochastic gradient descent (DP-SGD) . . . . .	25

<b>2 Interpretability and Explainability</b>	<b>26</b>
2.1 Why is it important? . . . . .	27
2.2 Common misunderstandings . . . . .	31
2.3 Types of methods . . . . .	33
2.4 How can we interpret an existing ML model? . . . . .	33
2.4.1 Occlusions . . . . .	38
2.4.2 Saliency maps . . . . .	39
2.4.3 Occlusions . . . . .	40
2.4.4 Saliency maps . . . . .	41
2.5 Lack of channel specificity . . . . .	44
2.6 Cam and Grad-Cam . . . . .	45
2.7 DeepDream / Inceptionism . . . . .	45
2.8 Inversion . . . . .	46
<b>3 Robustness: Adversarial Methods</b>	<b>47</b>
3.1 Adversarial Attacks . . . . .	47
3.1.1 Perturbation . . . . .	48
3.1.2 Fast Gradient Sign Method . . . . .	51
3.2 Adversarial attacks . . . . .	51
<b>A References</b>	<b>52</b>

# 1 Trustworthy AI/ML

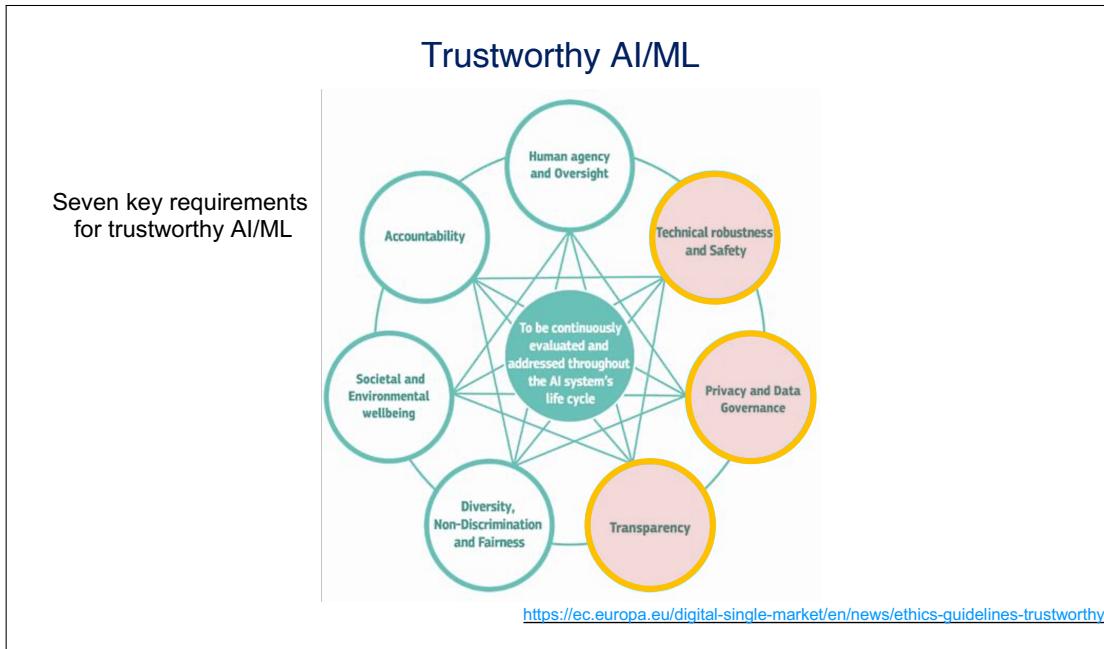


Figure 1: caption

## 1.1 The need for data

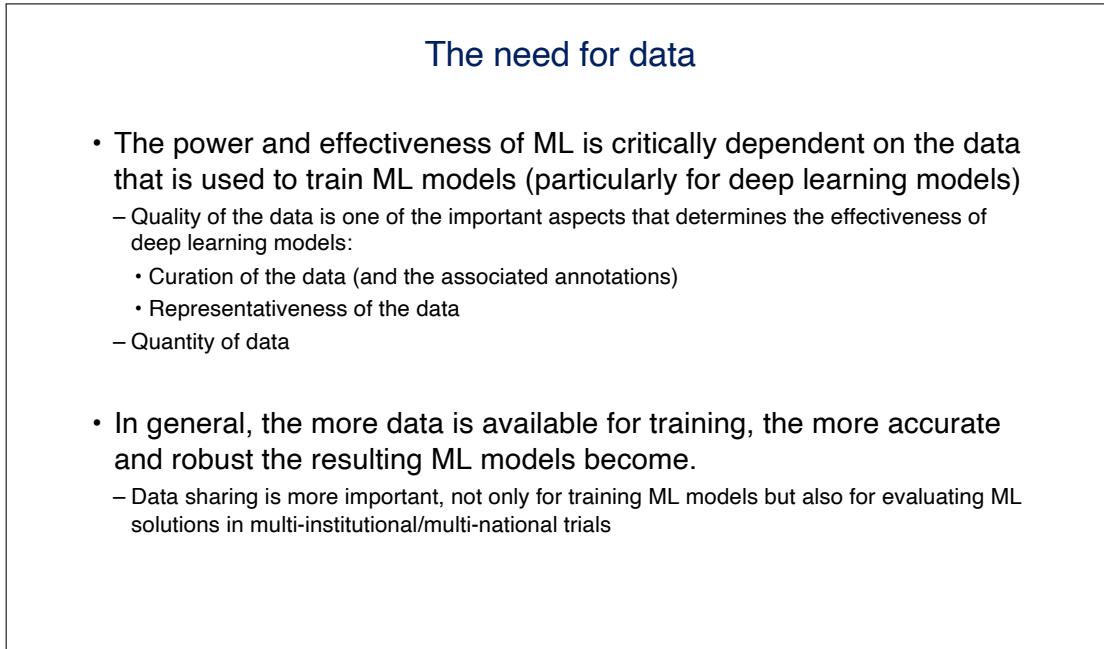


Figure 2: caption

## 1.2 What are the hurdles to getting more data?

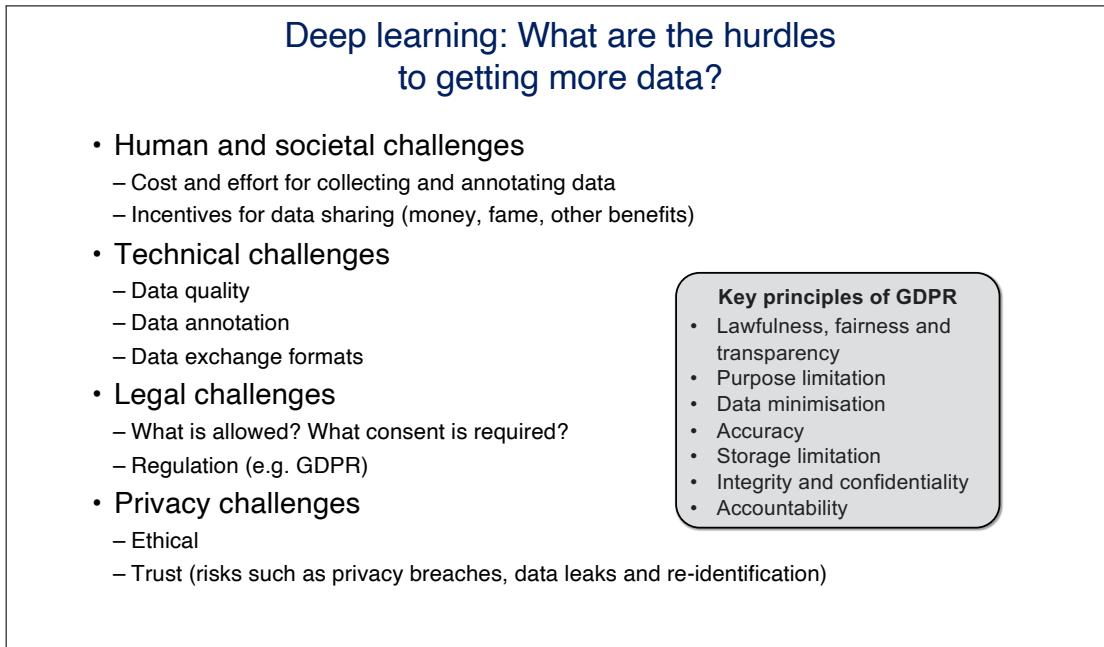


Figure 3: caption

## 1.3 Secure and Privacy-aware ML

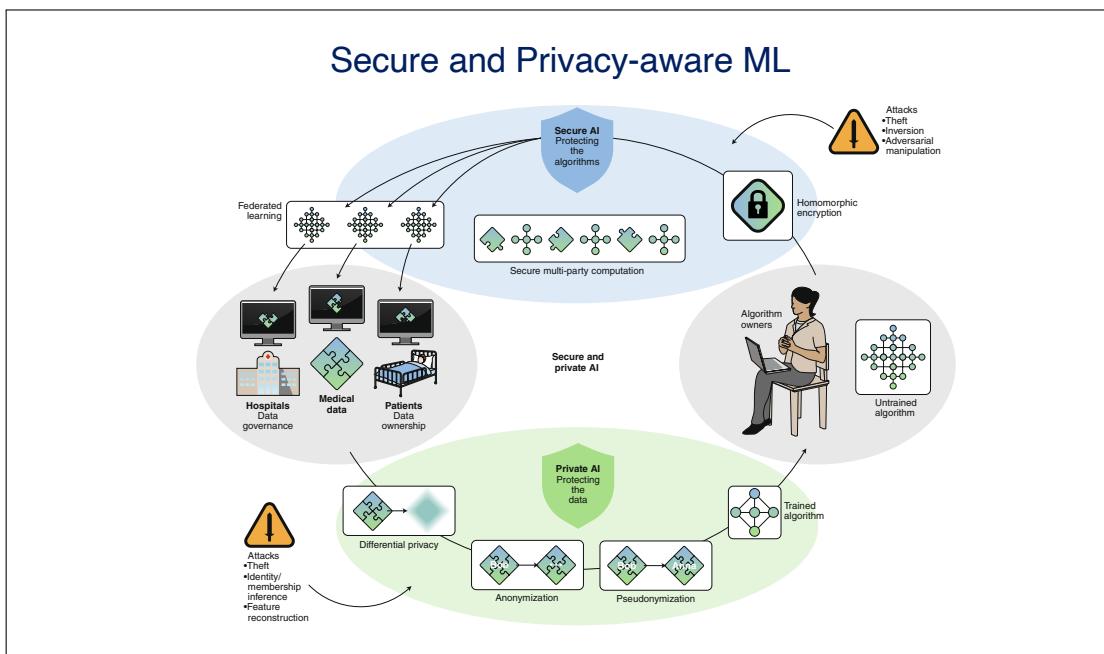


Figure 4: caption

## 1.4 Secure and Privacy-preserving ML

### Secure and privacy-preserving ML

- Optimal privacy preservation requires implementations that are secure by default so-called *privacy by design*

*Federated learning:* train a ML model across decentralized clients with local data, without exchanging them

- Requirements:

- Minimal or no data transfer
- Provision of theoretical and/or technical guarantees of privacy

*Differential privacy:* perturb the data so that information about the single individual is reduced while retaining the capability of learning

Figure 5: caption

### Secure and privacy-preserving ML

- Optimal privacy preservation requires implementations that are secure by default so-called *privacy by design*

- Requirements:

- Minimal or no data transfer
- Provision of theoretical and/or technical guarantees of privacy

- Other approaches for privacy-preserving AI:

- Homomorphic encryption which enables learning from encrypted data
- Secure multi-party computing where processing is performed on encrypted data shares, split among them in a way that no single party can retrieve the entire data on their own.
- Trusted execution environments

Figure 6: caption

## 1.5 Federated learning

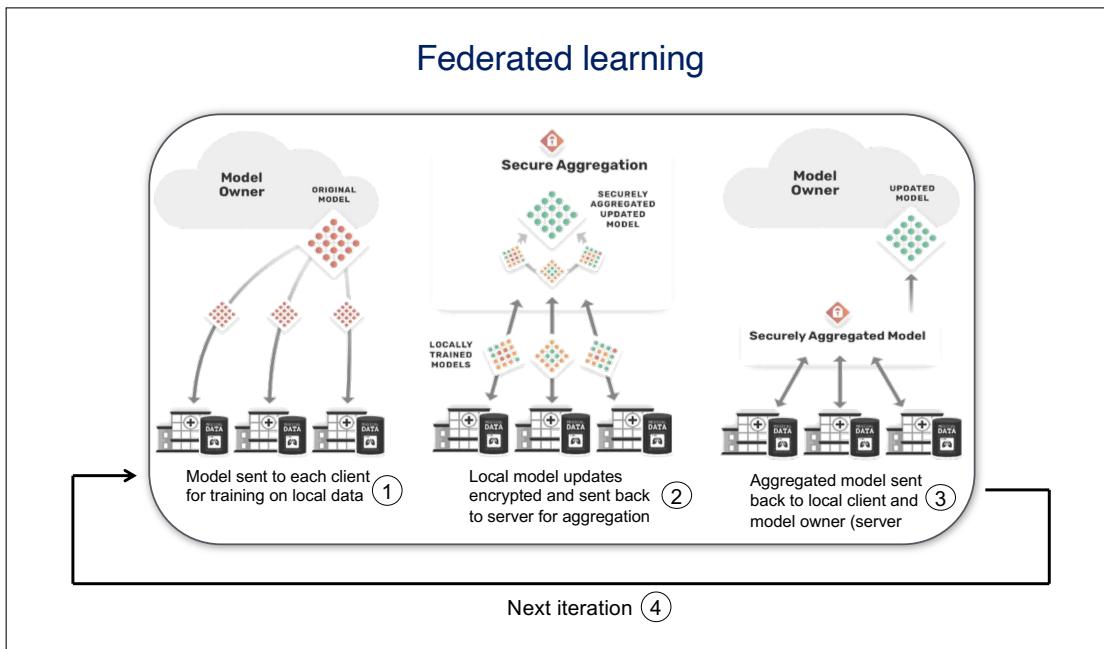


Figure 7: caption

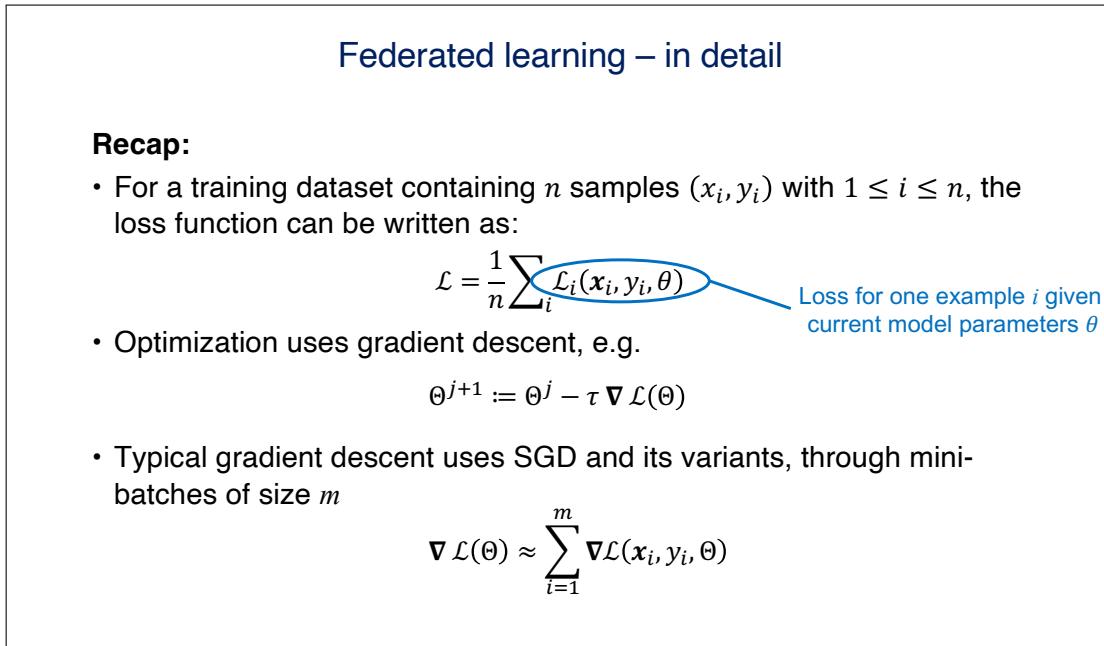


Figure 8: caption

## Federated learning – in detail

### In federated learning:

- Suppose  $N$  training samples are distributed to  $K$  clients,  $P_k$  is the set of indices of samples at client  $k$ , and  $n_k = |P_k|$

$$\mathcal{L}(\Theta) = \sum_{k=1}^K \frac{n_k}{N} \mathcal{L}_k(\Theta)$$

with

$$\mathcal{L}_k(\Theta) = \frac{1}{n_k} \sum_{i \in P_k} \mathcal{L}(\mathbf{x}_i, y_i, \Theta)$$

$$\mathbb{E}_{P_k}[\mathcal{L}_k] = \mathcal{L} \quad \text{iid setting}$$

$$\mathbb{E}_{P_k}[\mathcal{L}_k] \neq \mathcal{L} \quad \text{non-iid setting}$$

Figure 9: caption

### 1.5.1 Federated learning - SGD

## Federated SGD

### In federated learning:

- Suppose a  $C$  fraction of clients are selected at each round:  
 $C = 1$ : full-batch (non-stochastic) gradient descent  
 $C < 1$ : stochastic gradient descent (SGD)

- Each client computes:

$$\nabla \mathcal{L}_k(\Theta)$$

- Server computes:

$$\nabla \mathcal{L}(\Theta) = \sum_{k=1}^K \frac{n_k}{N} \nabla \mathcal{L}_k(\Theta)$$

Figure 10: caption

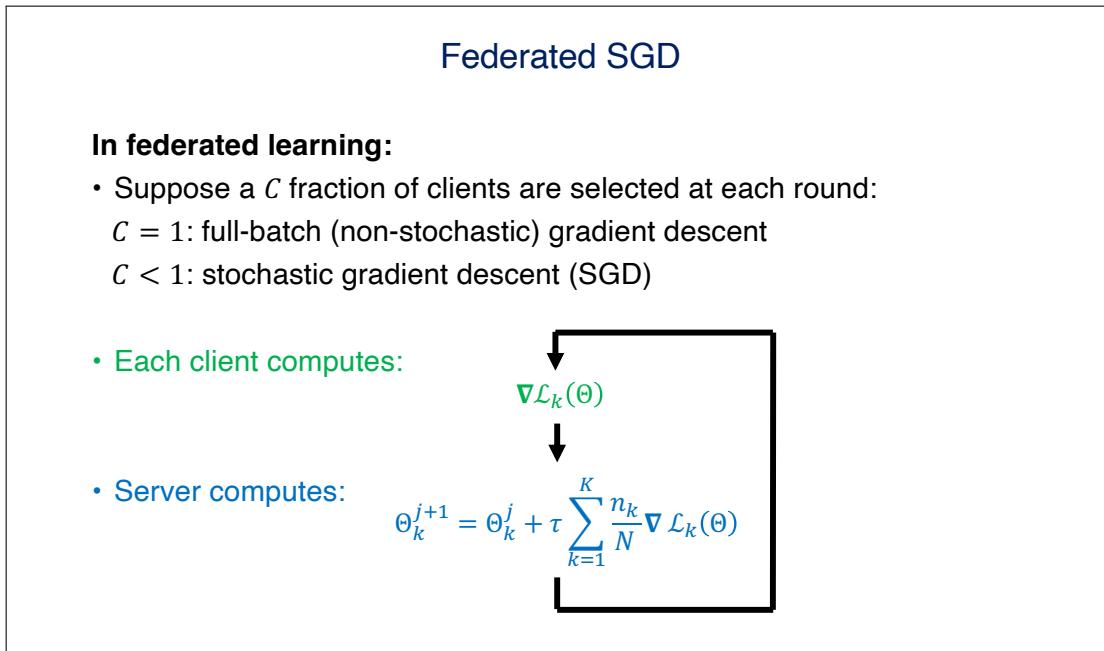


Figure 11: caption

### 1.5.2 Federated Averaging

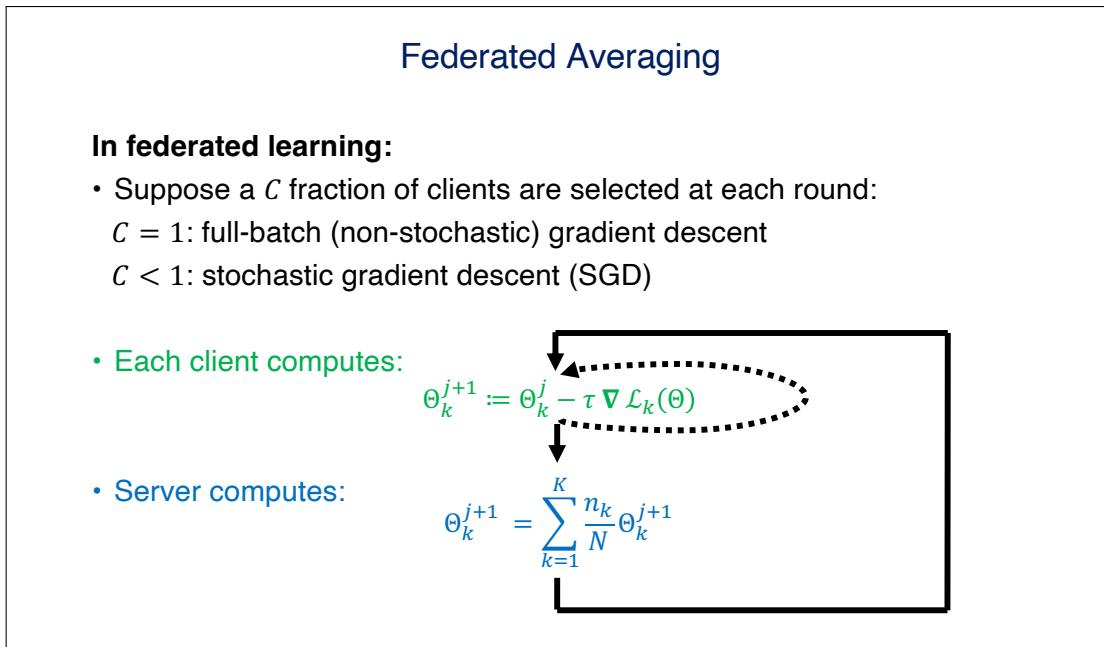
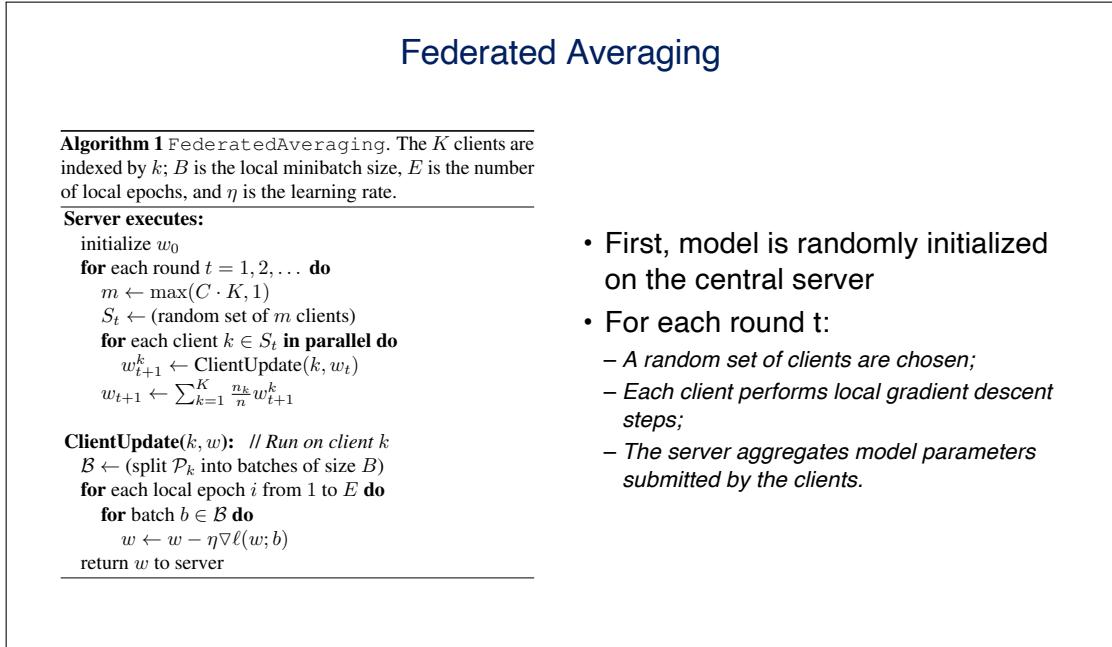
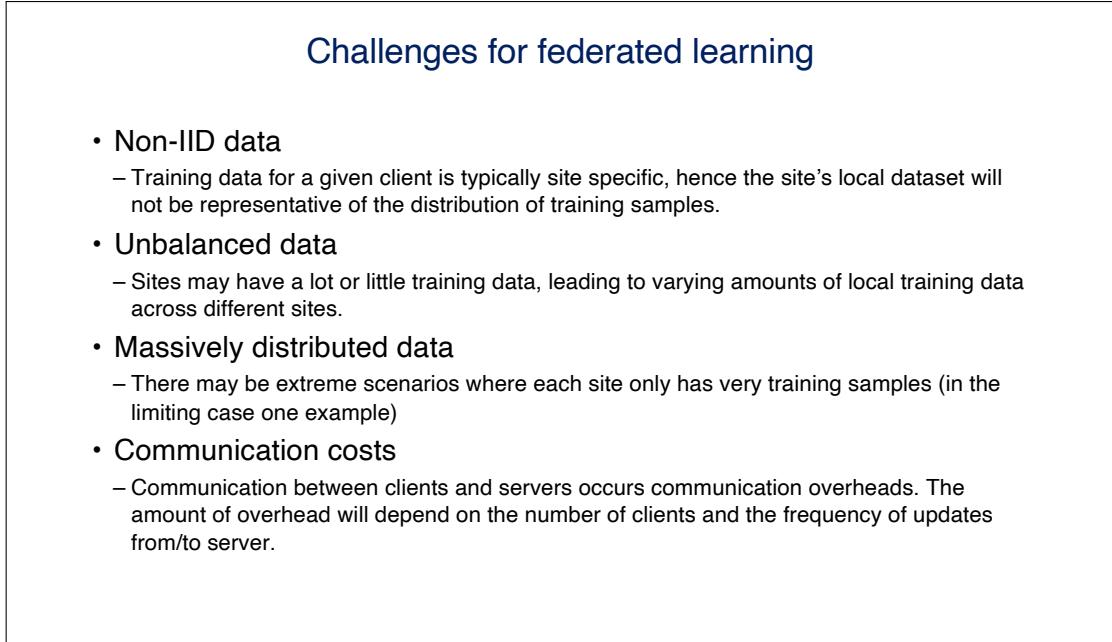


Figure 12: caption

**Figure 13:** caption

### 1.5.3 Challenges

**Figure 14:** caption

## 1.6 Homomorphic Encryption

### Homomorphic Encryption

- Based on the assumption that one can perform (basic) arithmetic on encrypted values, i.e.

$$[x] \oplus [y] = [x + y] \quad \text{and} \quad [x] \otimes [y] = [x \cdot y]$$

- Here:

$[xyz]$  encryption of some plaintext  $xyz$   
 $\oplus$  homomorphic addition operation in ciphertext space  
 $\otimes$  homomorphic multiplication operation in ciphertext space

Figure 15: caption

### 1.6.1 ML in standard setting

### ML in standard setting

Alice

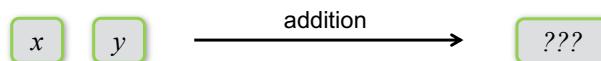
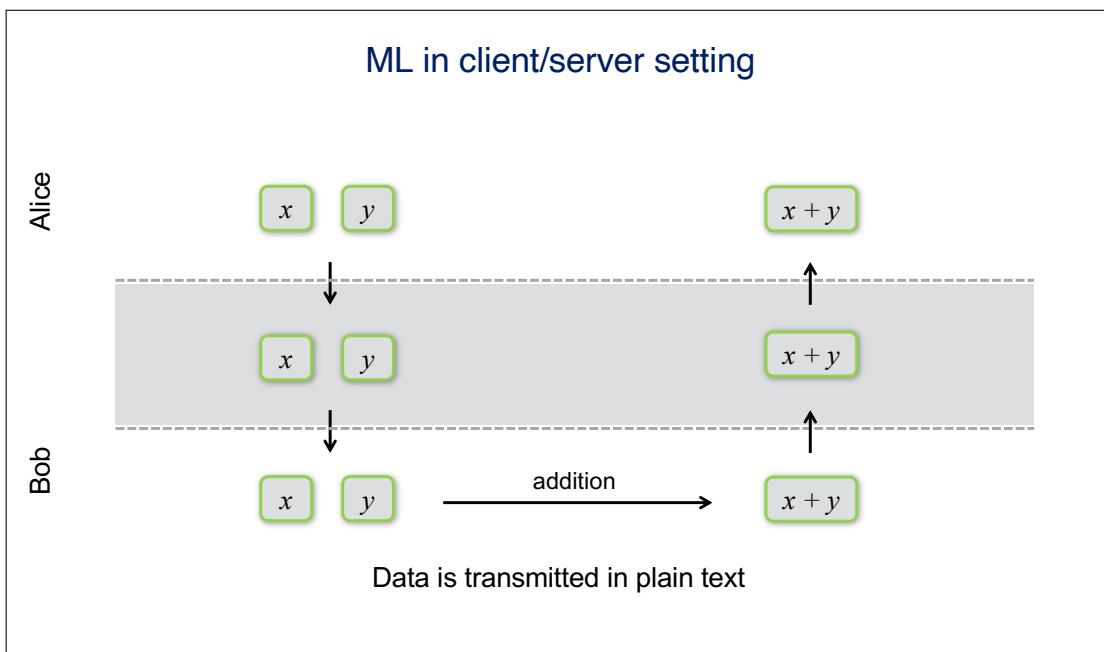
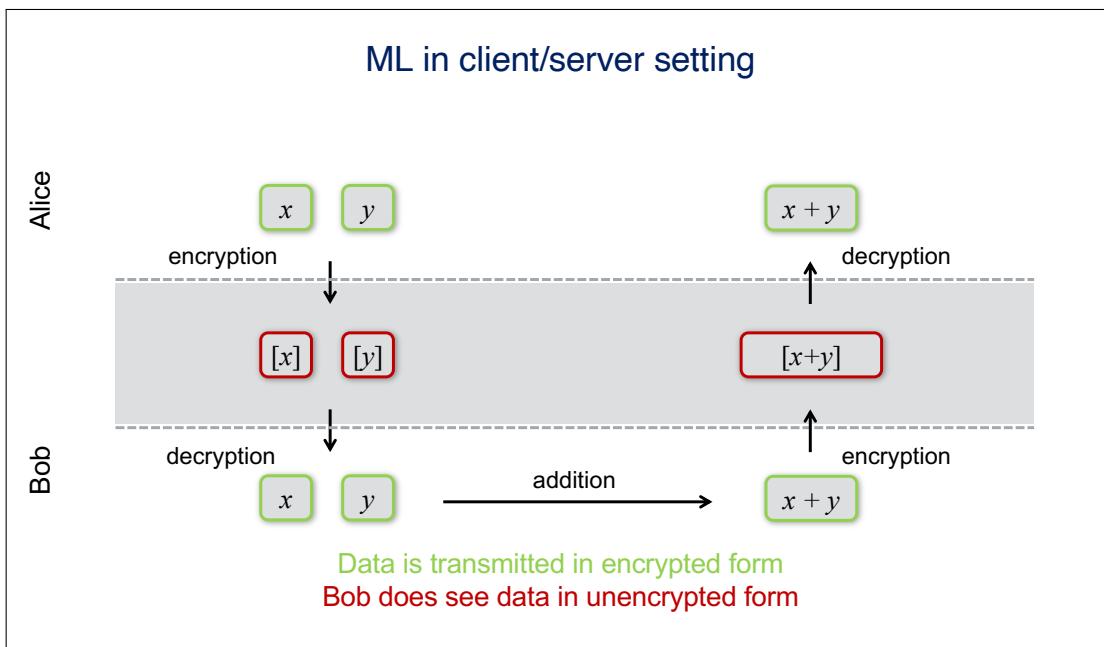


Figure 16: caption

### 1.6.2 ML in client/server setting



**Figure 17:** caption



**Figure 18:** caption

## 1.7 Homomorphic Encryption

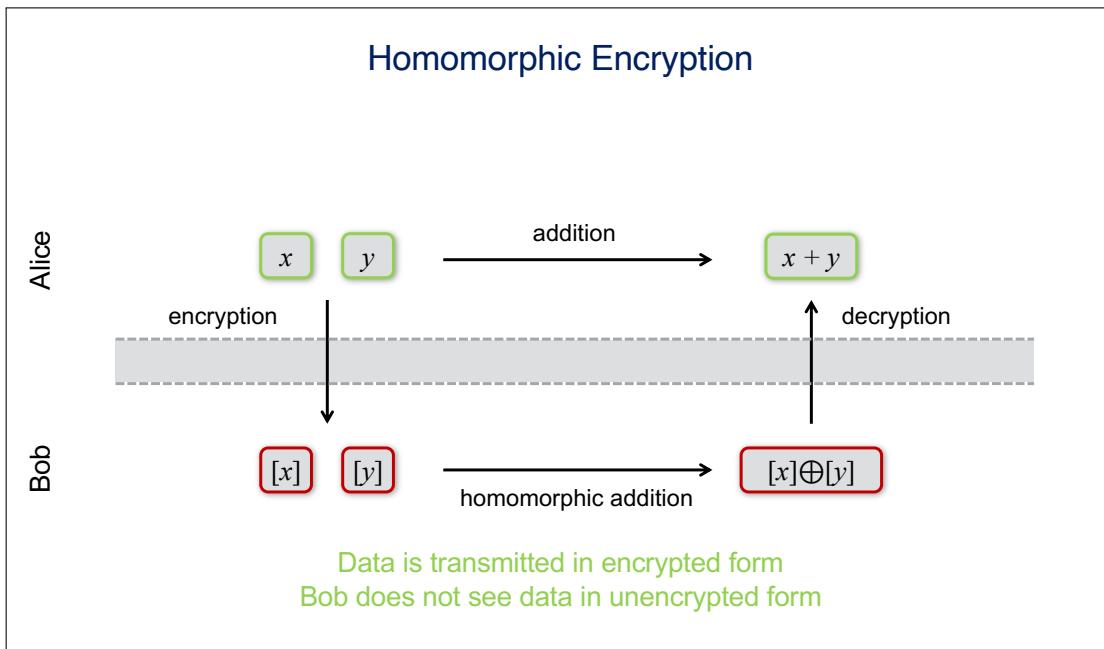


Figure 19: caption

## 1.8 ML and Homomorphic Encryption

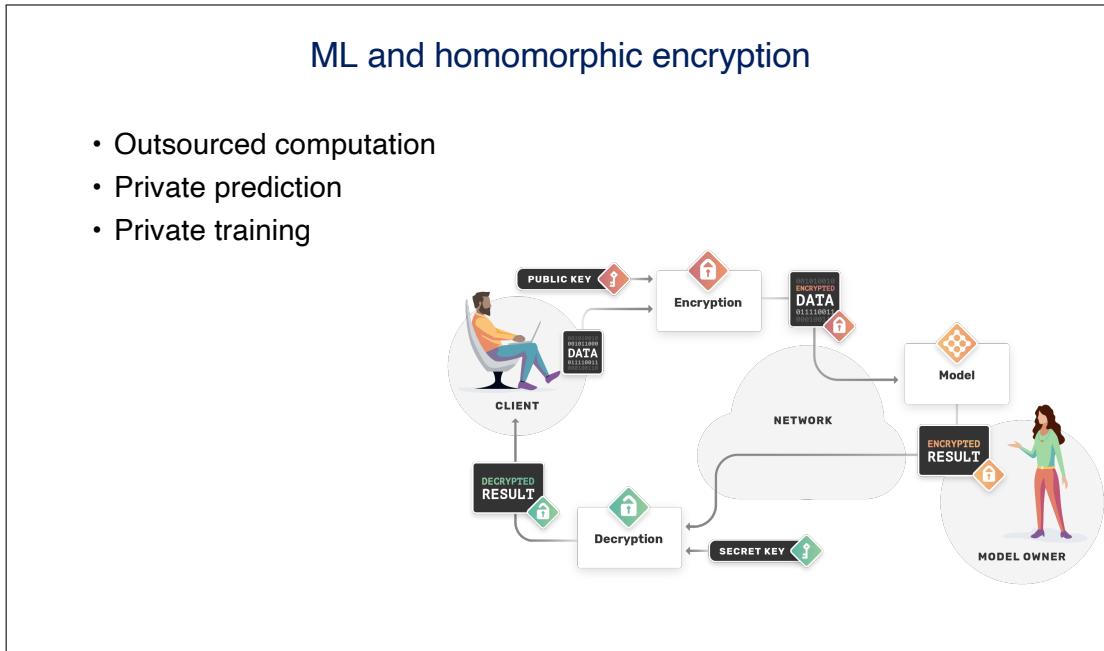


Figure 20: caption

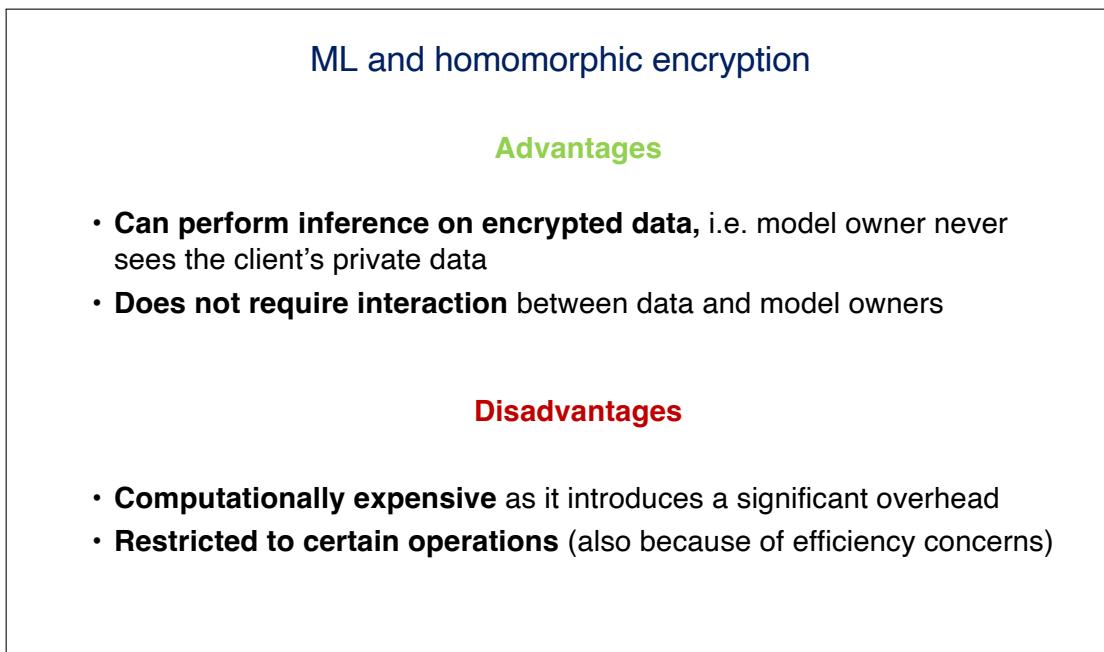


Figure 21: caption

## 1.9 Secure Multi-party Computation

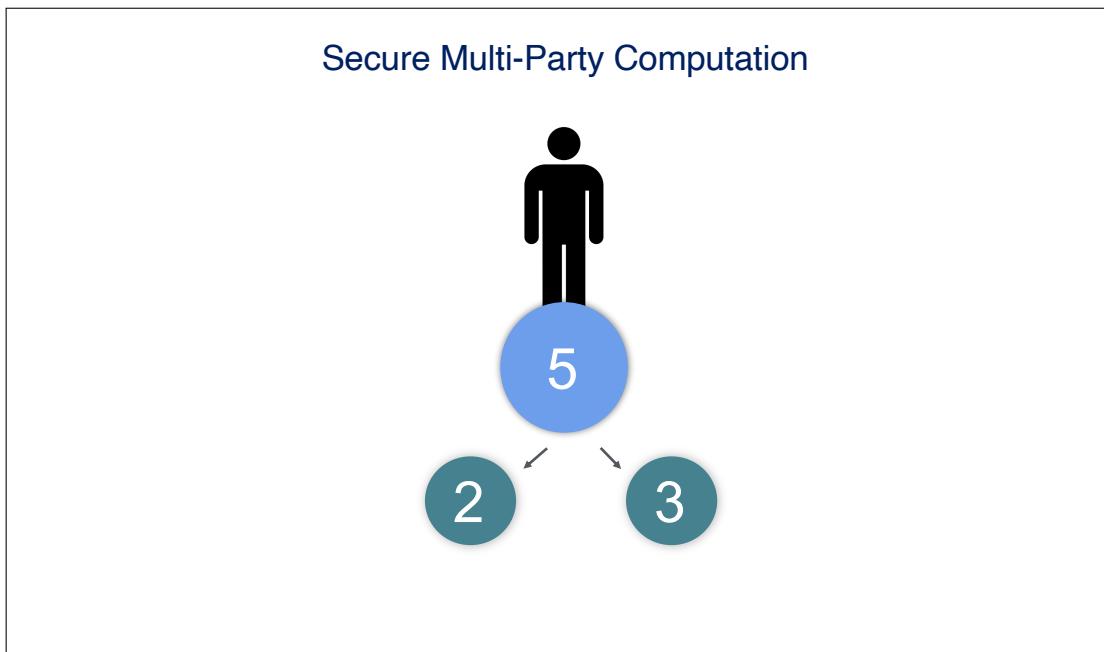


Figure 22: caption

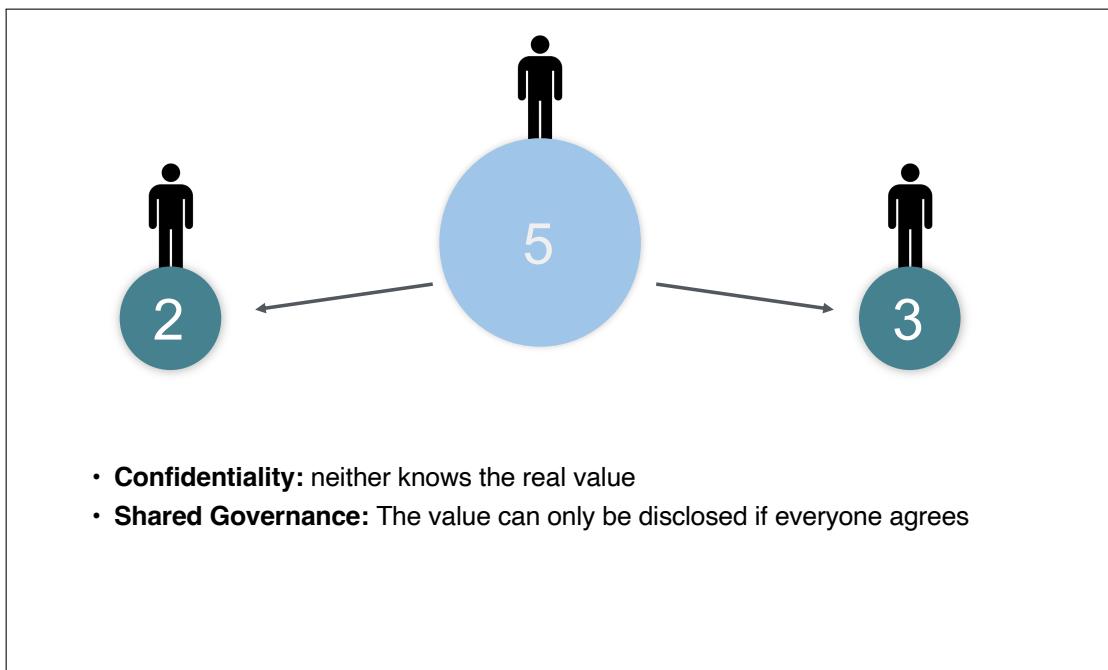


Figure 23: caption



Figure 24: caption

## 1.10 Secure Multi-Party Computation

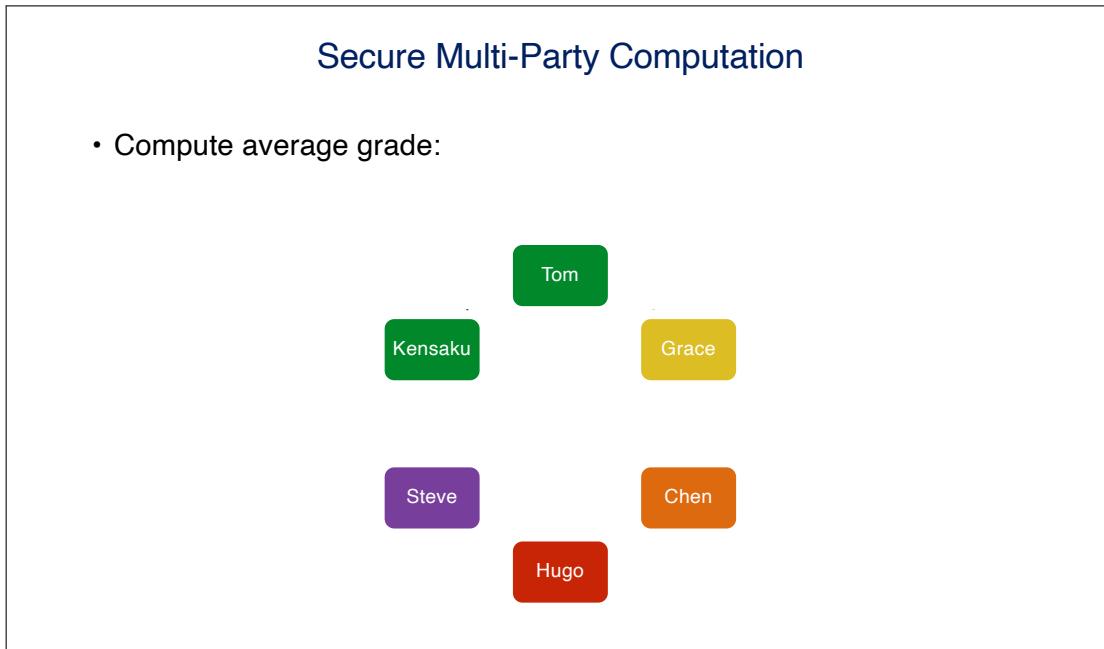


Figure 25: caption

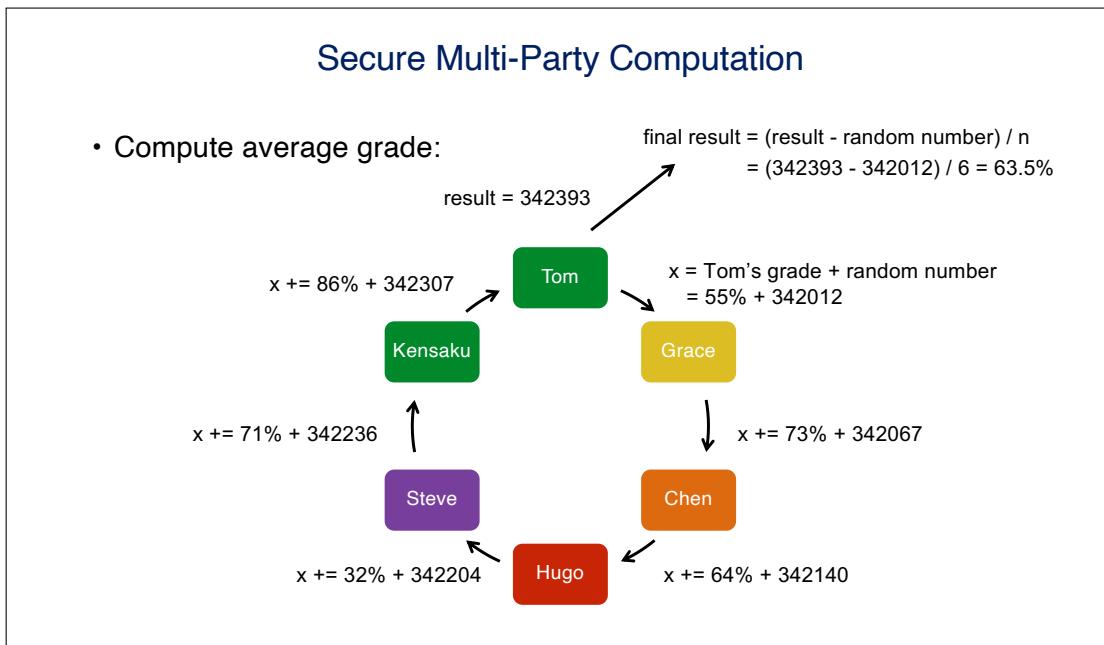


Figure 26: caption

## 1.11 Trusted Execution Environments

**Trusted Execution Environments**

- Set of **CPU instructions** to create enclaves in RAM, that **no one can access** - except code from the enclave itself
- Ensures **total confidentiality of data during computation** - decryption happens only inside the enclave

**Confidential Computing** BETA

Encrypt data in-use with Confidential VMs. Available in Beta for Google Compute Engine.

Figure 27: caption

## 1.12 What is Privacy?

**What is privacy?**

**Anonymization**

- The most straightforward approach is anonymization where identifying information is removed.
  - For example, a name may be removed from a medical record.
- Unfortunately, anonymization is rarely sufficient to protect privacy as the remaining information can be uniquely identifying.
  - For instance, given the gender, postal code, age, ethnicity and height, it may be possible to identify someone uniquely, even in a very large database.

Figure 28: caption

## 1.13 k-anonymity

### k-anonymity

- One approach to prevent linkage attacks is *k*-anonymity.
  - A dataset is said to be *k* anonymous if, for any person's record in a dataset, there are at least  $k - 1$  other records which are indistinguishable.
  - So if a dataset is *k*-anonymous, then the best a linkage attack could ever do is identify a group of *k* records which could belong to the person of interest.
  - Even if a dataset isn't inherently *k*-anonymous, it could be made so by removing entire fields of data (like names and addresses) and selectively censoring fields of individual people who are particularly unique.

Figure 29: caption

### k-anonymity: Example

Original Database to Disclose

ID	IDENTIFYING VARIABLE Name	QUASI-IDENTIFIERS		Test Result
		Gender	Year of Birth	
1	John Smith	Male	1959	+ve
2	Alan Smith	Male	1962	-ve
3	Alice Brown	Female	1955	-ve
4	Linda Green	Male	1959	-ve
5	Alicia Freds	Female	1942	-ve
6	Gill Stringer	Female	1975	-ve
7	Marie Kirkpatrick	Female	1966	+ve
8	Leslie Hall	Female	1987	-ve
9	Bill Nash	Male	1975	-ve
10	Ann Bravewell	Male	1978	-ve
11	Beverly McCusky	Female	1964	-ve
12	Douglas Henry	Male	1959	+ve
13	Freda Shields	Female	1975	-ve
14	Fred Thompson	Male	1967	-ve

2-Anonymization

ID	QUASI-IDENTIFIERS		Test Result
	Gender	Decade of Birth	
1	Male	1950-1959	+ve
2	Male	1960-1969	-ve
4	Male	1950-1959	-ve
6	Female	1970-1979	-ve
7	Female	1960-1969	+ve
9	Male	1970-1979	-ve
10	Male	1970-1979	-ve
11	Female	1960-1969	-ve
12	Male	1950-1959	+ve
13	Female	1970-1979	-ve
14	Male	1960-1969	-ve

Disclosed (k-Anonymized) Database ( $\zeta$ )

K. El Emam et al. Protecting privacy using k-anonymity. J Am Med Inform Assoc. 20c  
<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC252802>

Figure 30: caption

***k-anonymity***

### **Anonymization and linkage attacks**

- Famous example:
  - After an insurance group released health records which had been stripped of personal information like patient name and address, a CS student was able to “deanonymize” which records belonged to politicians (including the Governor of Massachusetts) by cross referencing with public voter registers.
  - This is an example of a **linkage attack**, where connections to other sources of information work to deanonymize a dataset.
  - Linkage attacks have been successful on a range of anonymized datasets including the Netflix challenge and genome data.

<https://arstechnica.com/tech-policy/2009/09/your-secrets-live-online-in-databases-of-ruin/>

The screenshot shows a news article from Ars Technica. The title is "Anonymized" data really isn't—and here's why not". The article discusses how companies continue to store and sometimes release vast databases of personal information. A quote from Nate Anderson states: "Companies continue to store and sometimes release vast databases of ...". Below the quote, it says: "The Massachusetts Group Insurance Commission had a bright idea back in the mid-1990s—it decided to release 'deanonymized' data on state employees that showed every single hospital visit. The goal was to help researchers, and the state spent time removing all obvious identifiers such as name, address, and Social Security number. But a graduate student in computer science saw a chance to make a point about the limits of anonymization." A note at the bottom says: "Latanya Sweeney requested a copy of the data and went to work on her 'reidentification' quest. It didn't prove difficult. Law professor Paul Ohm describes Sweeney's work."

**Figure 31:** caption

***k-anonymity***

- Unfortunately, *k*-anonymity isn't sufficient for anything but very large datasets with only small numbers of simple fields for each record.
- Intuitively, the more fields and the more possible entries there are in those fields, the more unique a record can be and the harder it is to ensure that there are *k* equivalent records.

**Figure 32:** caption

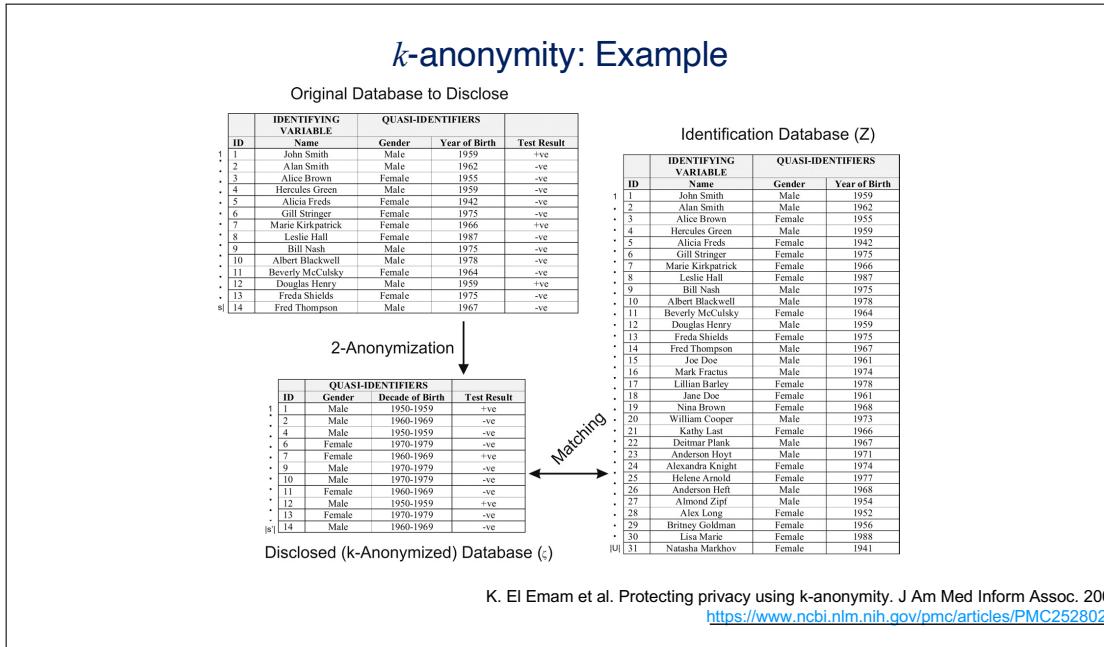


Figure 33: caption

## 1.14 Privacy attacks

### 1.14.1 Model inversion

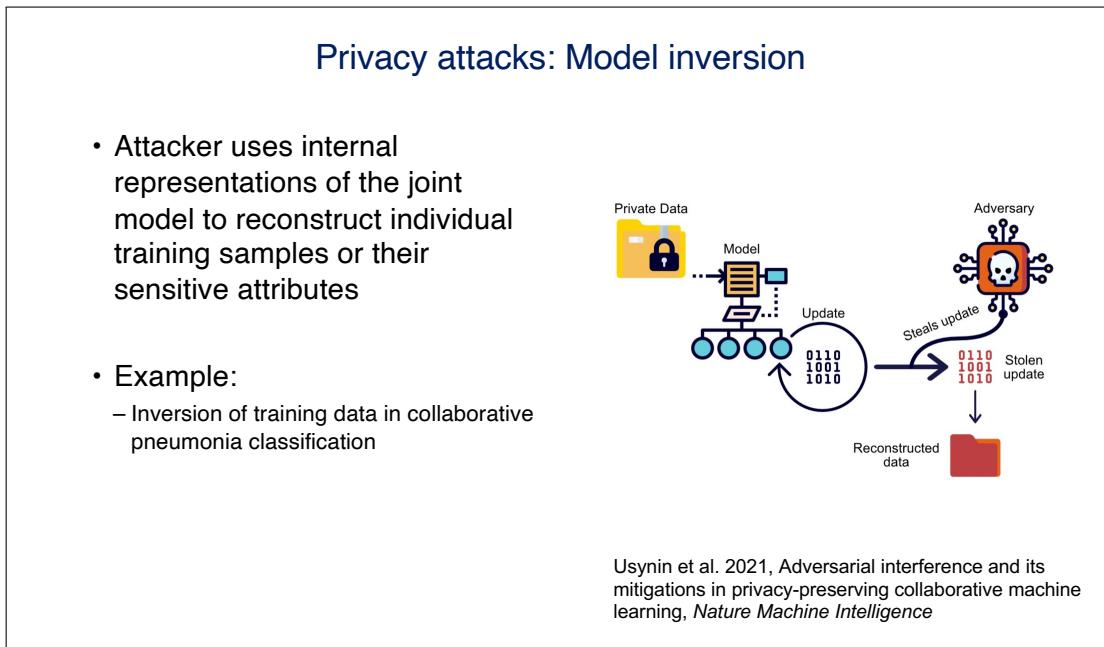
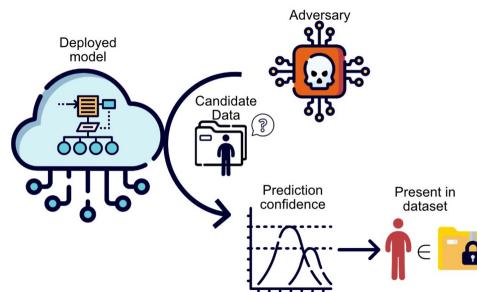


Figure 34: caption

### 1.14.2 Membership inference

**Privacy attacks: Membership inference**

- Attacker obtains a data record and determines if it was used to train a particular model
- Example:
  - Determining if a specific patient was part of the HIV-positive dataset



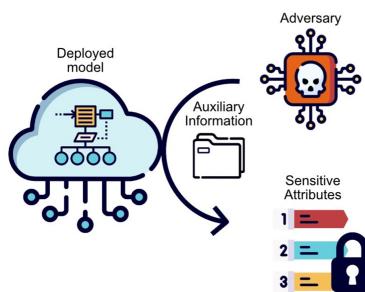
Usynin et al. 2021, Adversarial interference and its mitigations in privacy-preserving collaborative machine learning, *Nature Machine Intelligence*

**Figure 35:** caption

### 1.14.3 Attribute inference

**Privacy attacks: Attribute inference**

- Attacker uses model access and auxiliary information about the victim to obtain the sensitive values of their data
- Example:
  - Given access to a model trained on patient records and a specific patient's public information infer their HIV status



Usynin et al. 2021, Adversarial interference and its mitigations in privacy-preserving collaborative machine learning, *Nature Machine Intelligence*

**Figure 36:** caption

## 1.15 Differential Privacy

### Differential Privacy

#### Randomized responses

- Enables draw statistical conclusion from datasets without revealing information about individual data points.

Figure 37: caption

### Differential Privacy

#### Randomized responses

- Enables draw statistical conclusion from datasets without revealing information about individual data points.
- Realised by adding a controlled amount of noise

Figure 38: caption

### 1.15.1 Randomized responses



Figure 39: caption

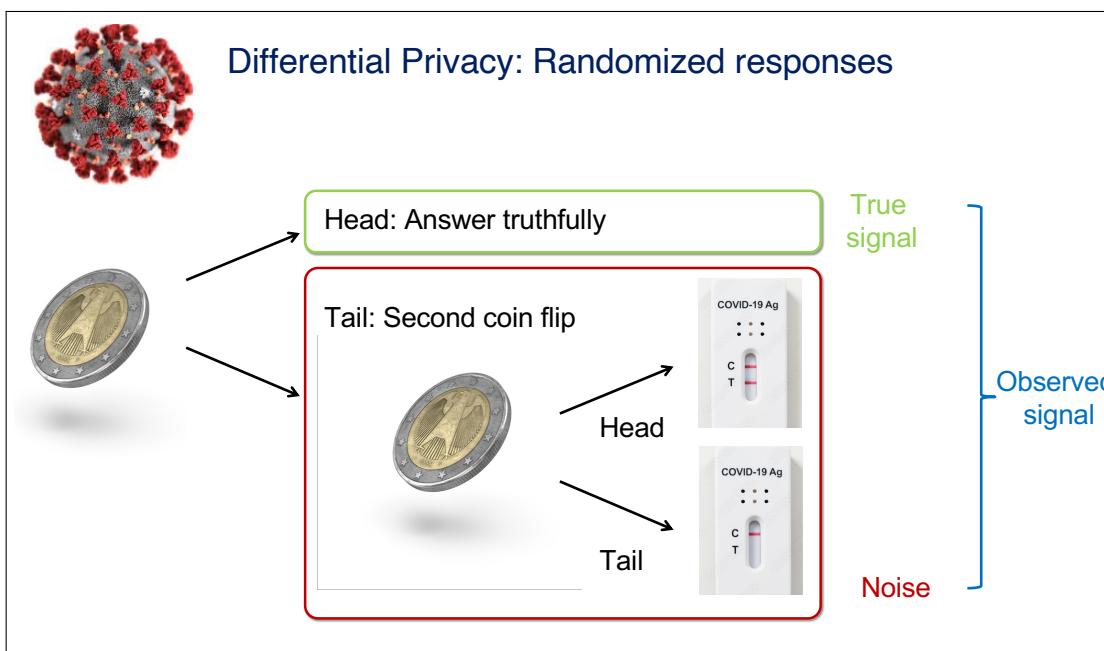


Figure 40: caption

### Differential Privacy: Randomized responses

- In this example, there is a parameter which is the probability that the true response is recorded:
  - If it's very likely that the true response is recorded, then there is less privacy protection.
  - Conversely, if it's unlikely that the true response is recorded, then there is more.
  - It's also clear that, regardless of the probability, if an individual is surveyed multiple times, then there will be less protection, even if their answer is potentially randomized every time.
- Differential privacy formalizes how we define, measure and track the privacy protection afforded to an individual as functions of factors like randomization probabilities and number of times surveyed.

Figure 41: caption

## 1.16 Differential privacy

### Differential privacy

- An algorithm can be made approximately invariant to inclusion of a single data sample
- This is achieved through the addition algorithm

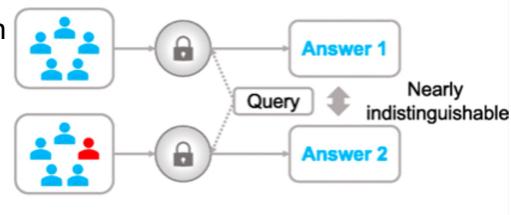


Figure 42: caption

### Differential privacy (continued)

- This can be mathematically defined using the following (relaxed) expression:

A process A is  $\epsilon$ -differentially private if for all databases  $D_1$  and  $D_2$  which differ in only one individual:

$$\mathbb{P}[A(D_1) = O] \leq e^\epsilon \cdot \mathbb{P}[A(D_2) = O]$$

Figure 43: caption

### Differential privacy (continued)

- This is done through the addition of noise to the output of the query so then the results in two datasets look approximately like this:

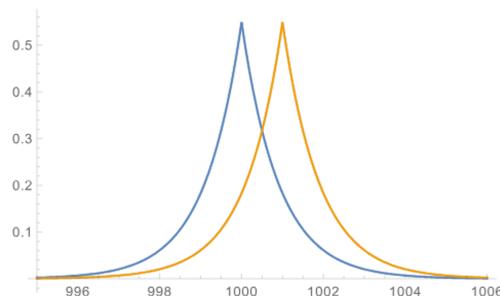


Figure 44: caption

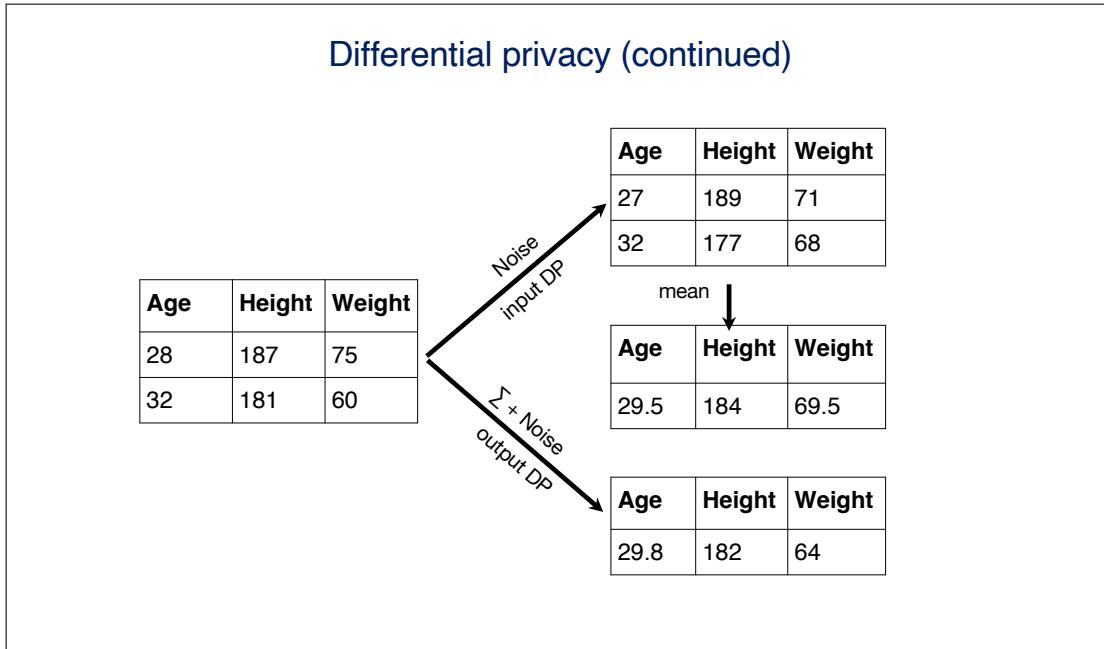


Figure 45: caption

## 1.17 Concrete mitigation

### 1.17.1 Differentially private stochastic gradient descent (DP-SGD)

#### Concrete mitigation: Differentially private stochastic gradient descent (DP-SGD)

1. Compute gradients for each individual sample (they represent independent clients)
2. Clip the calculated gradients to obtain a known sensitivity
3. Add the noise scaled by the sensitivity from step 2
4. Perform the gradient descent step

Abadi, Martin, et al. "Deep learning with differential privacy." *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016.

---

**Algorithm 1** Differentially private SGD (Outline)

**Input:** Examples  $\{x_1, \dots, x_N\}$ , loss function  $\mathcal{L}(\theta) = \frac{1}{N} \sum_i \mathcal{L}(\theta, x_i)$ . Parameters: learning rate  $\eta_t$ , noise scale  $\sigma$ , group size  $L$ , gradient norm bound  $C$ .

**Initialize**  $\theta_0$  randomly

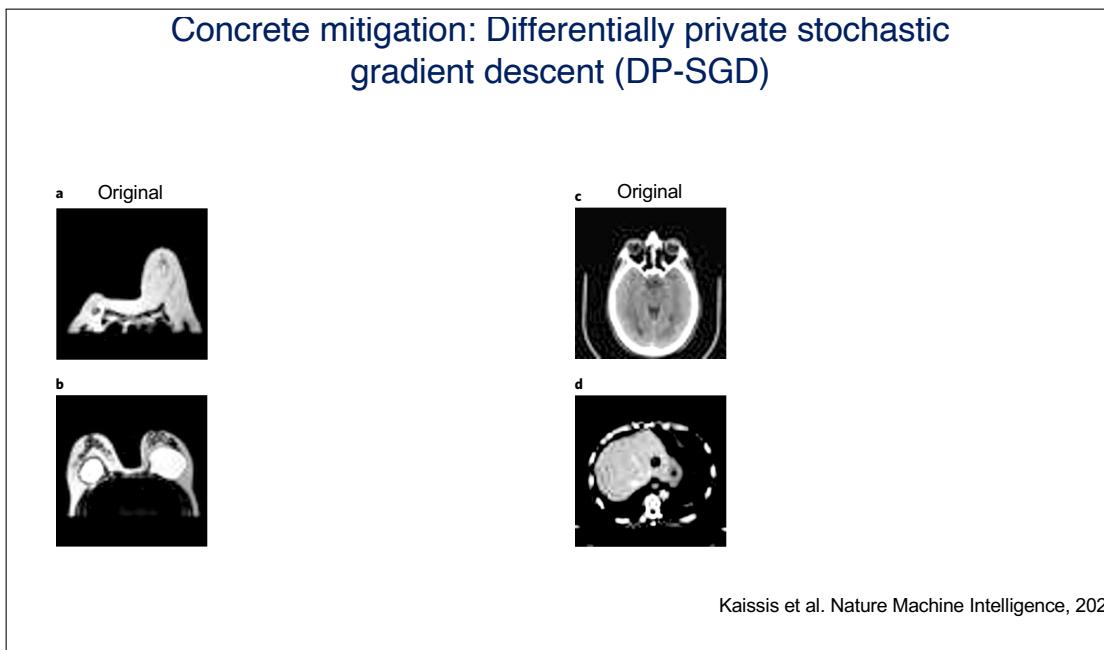
**for**  $t \in [T]$  **do**

- Take a random sample  $L_t$  with sampling probability  $L/N$
- Compute gradient**  
For each  $i \in L_t$ , compute  $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$
- Clip gradient**  
 $\tilde{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C})$
- Add noise**  
 $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L} \sum_i (\tilde{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))$
- Descent**  
 $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

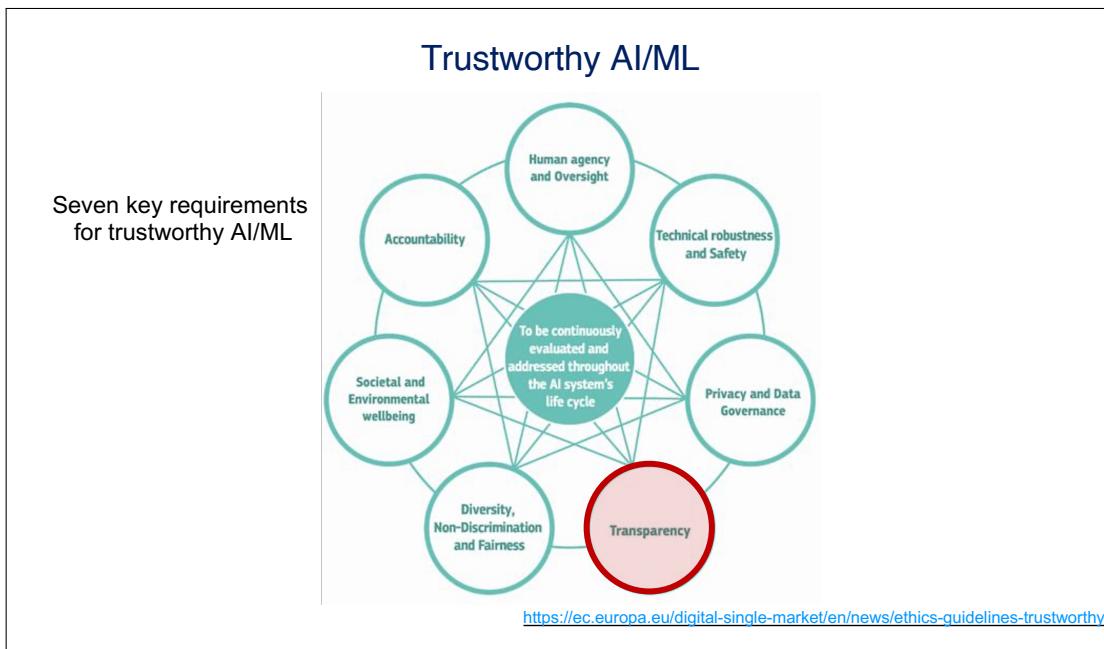
**Output**  $\theta_T$  and compute the overall privacy cost  $(\varepsilon, \delta)$  using a privacy accounting method.

---

Figure 46: caption

**Figure 47:** caption

## 2 Interpretability and Explainability

**Figure 48:** caption

## 2.1 Why is it important?

### Interpretability: Why is this important?

- This is not a new problem, so why now?
  - Complexity and prevalence!
  - Safety and robustness is critical
  - Generating knowledge

Figure 49: caption

### Interpretability: Why is this important?



Safety



Science



Robustness

Figure 50: caption

### Interpretability: Why is this important?

- This is not a new problem, so why now?
  - Complexity and prevalence!
  - Safety is critical
  - Generating knowledge
- Debugging machine learning
  - Why does my model not train?
  - Why does my model exhibit unexpected/unintuitive behaviour?

Figure 51: caption

### Interpretability: Why is this important?

- Debugging machine learning models

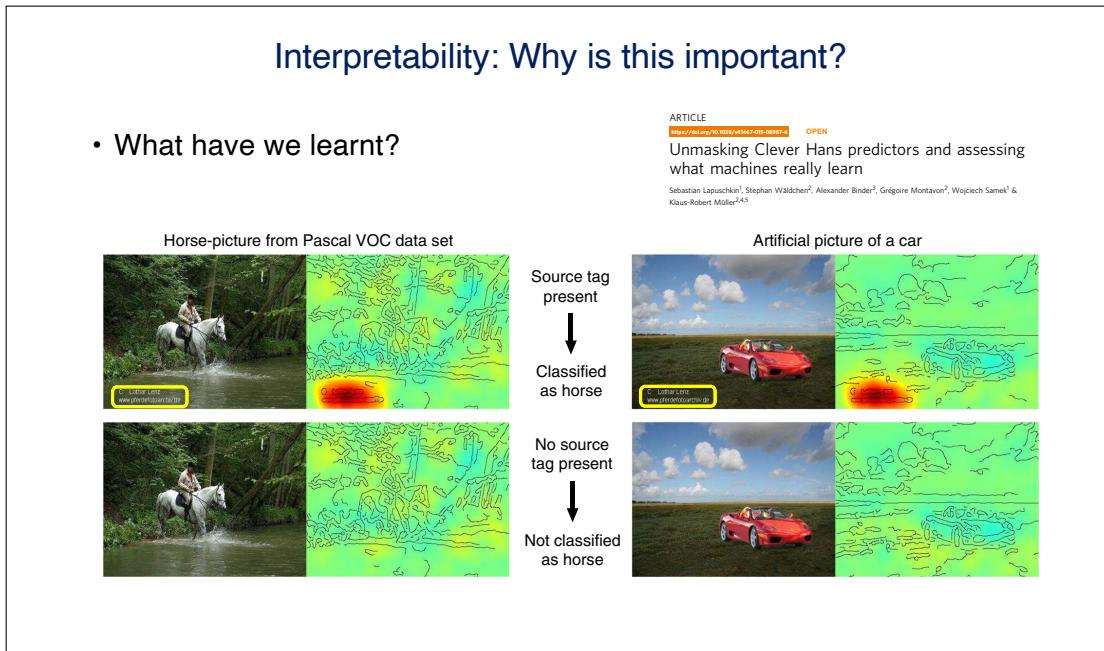
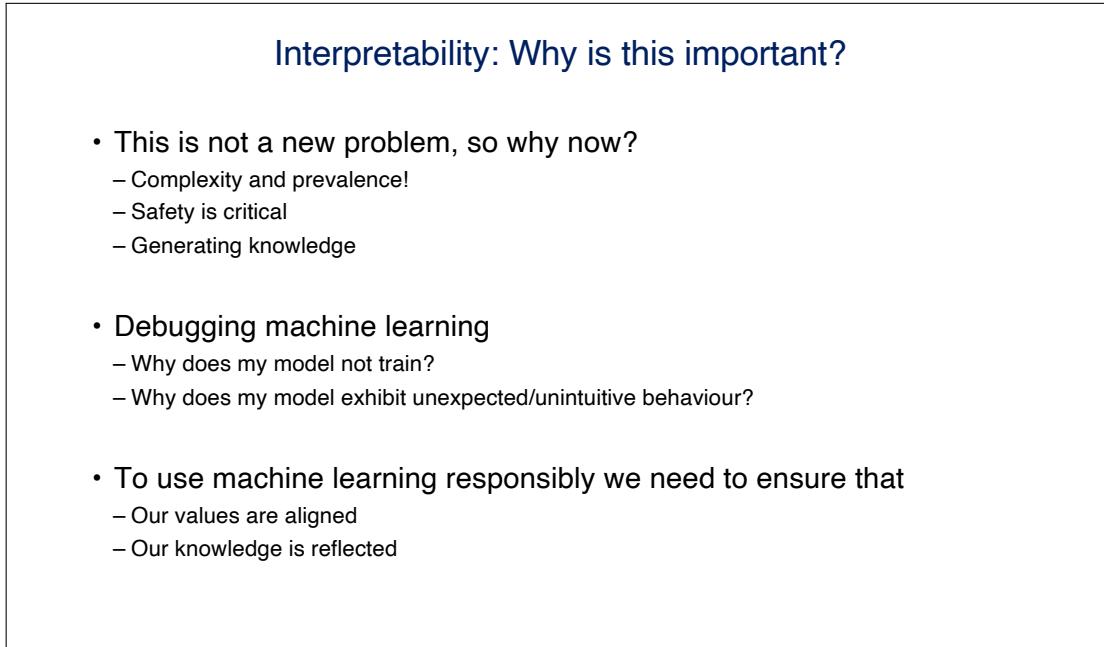


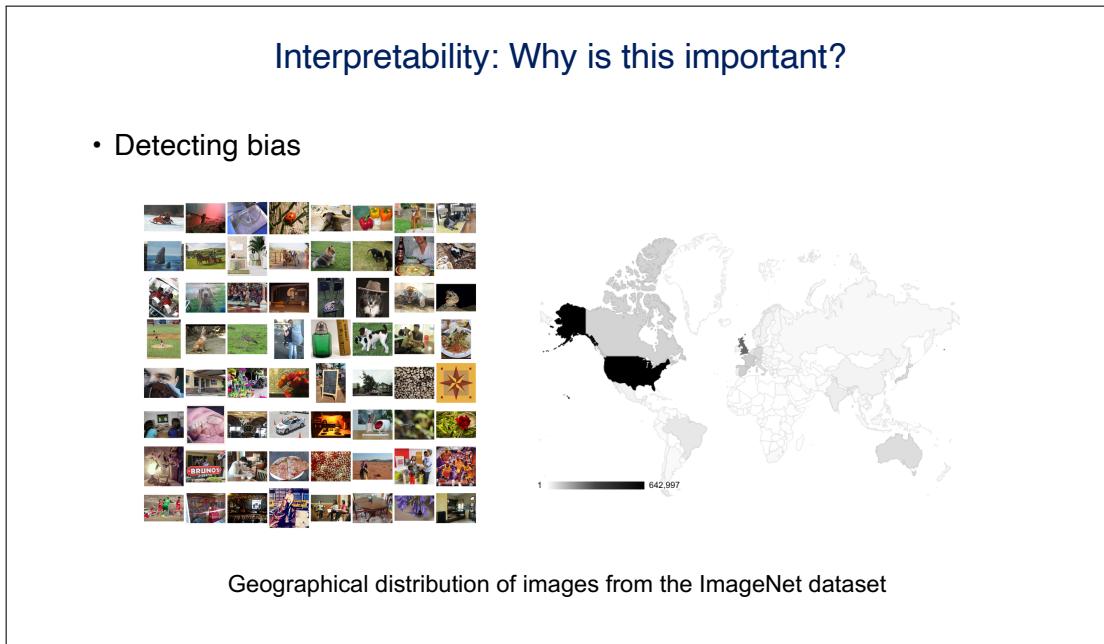
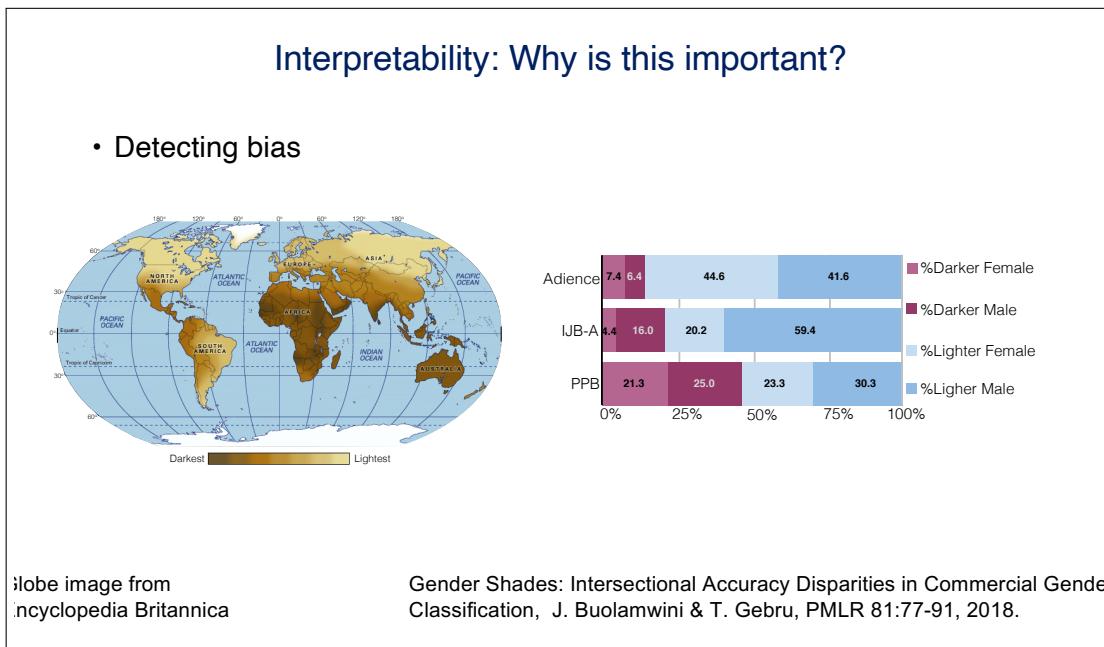
Data during training

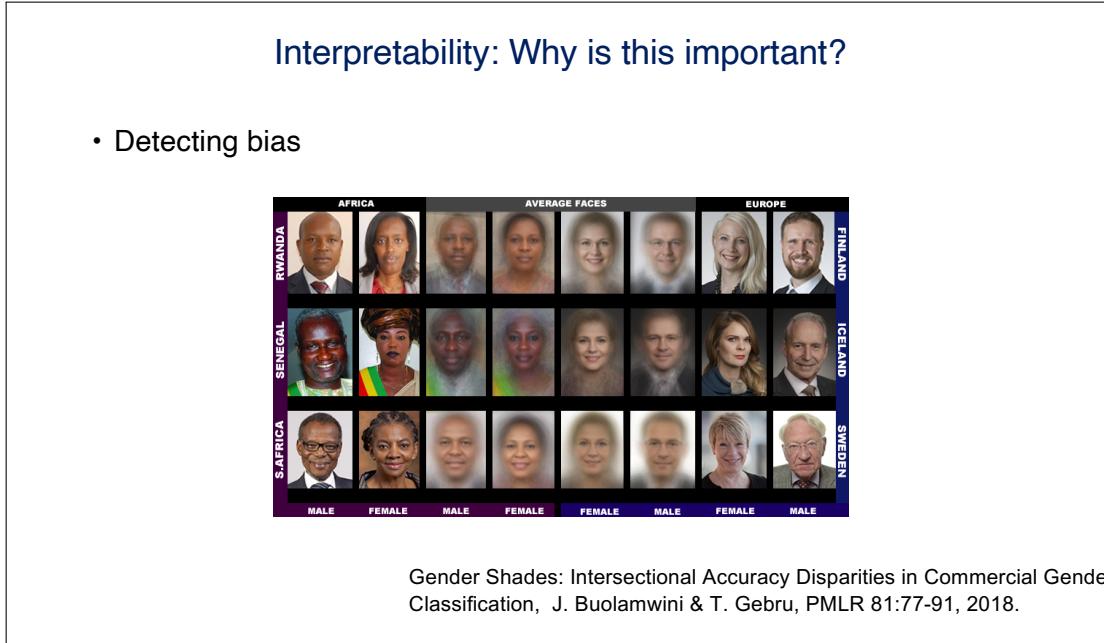


Data during deployment

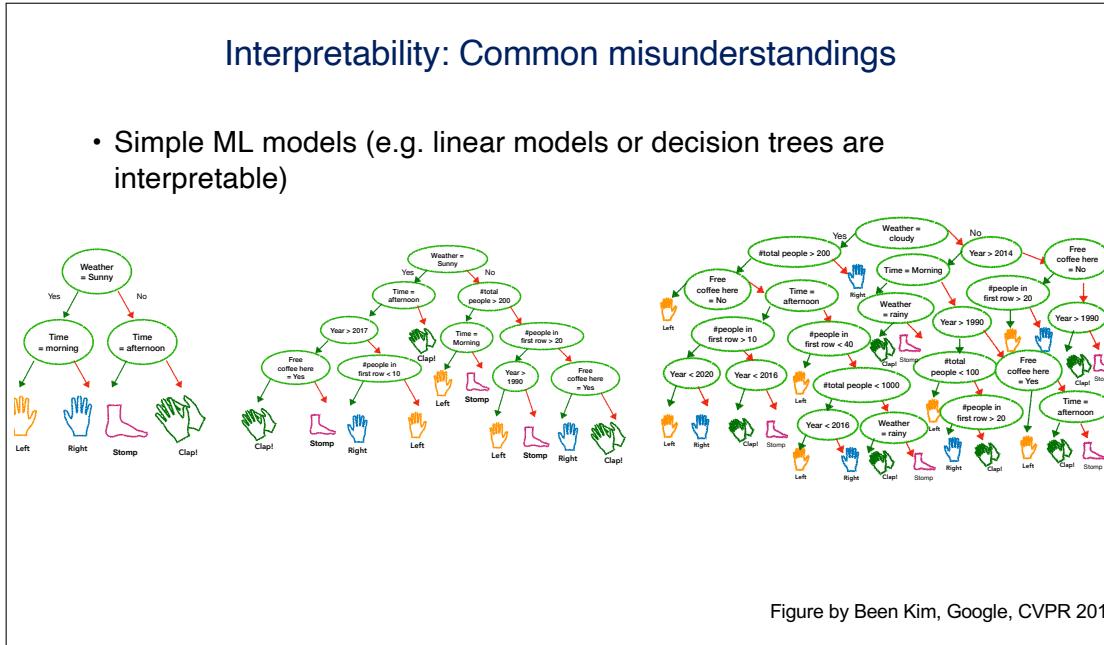
Figure 52: caption

**Figure 53:** caption**Figure 54:** caption

**Figure 55:** caption**Figure 56:** caption

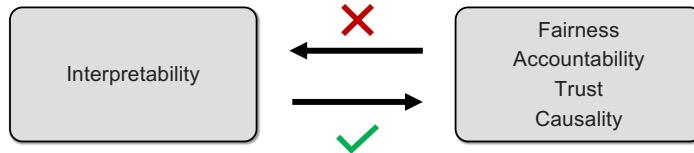
**Figure 57:** caption

## 2.2 Common misunderstandings

**Figure 58:** caption

## Interpretability: Common misunderstandings

- Trust, fairness and interpretability are all the same thing



- Interpretability can help to formalize concepts such as fairness or trust
- Once formalized it may not be needed anymore

Adapted from slides by Been Kim, Google, CVPR 201

**Figure 59:** caption

## Interpretability: Common misunderstandings

- We don't always care about interpretability:
  - No significant consequences or when predictions are all you need
  - Sufficiently well-studied problem
  - Prevent gaming the system



**Figure 60:** caption

### 2.3 Types of methods

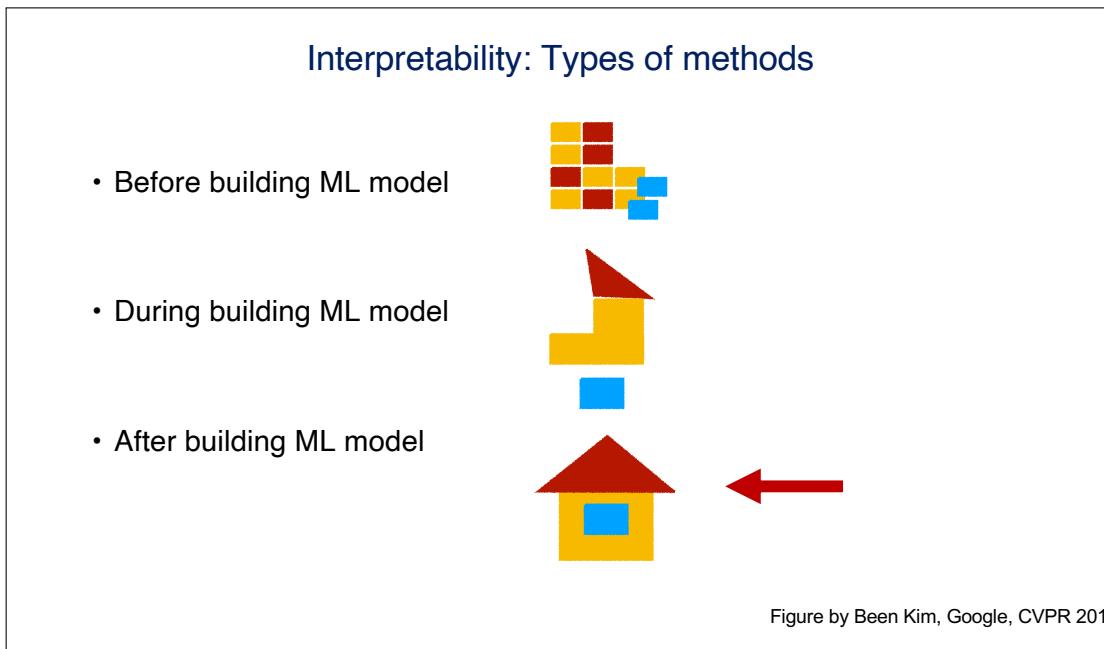


Figure 61: caption

### 2.4 How can we interpret an existing ML model?

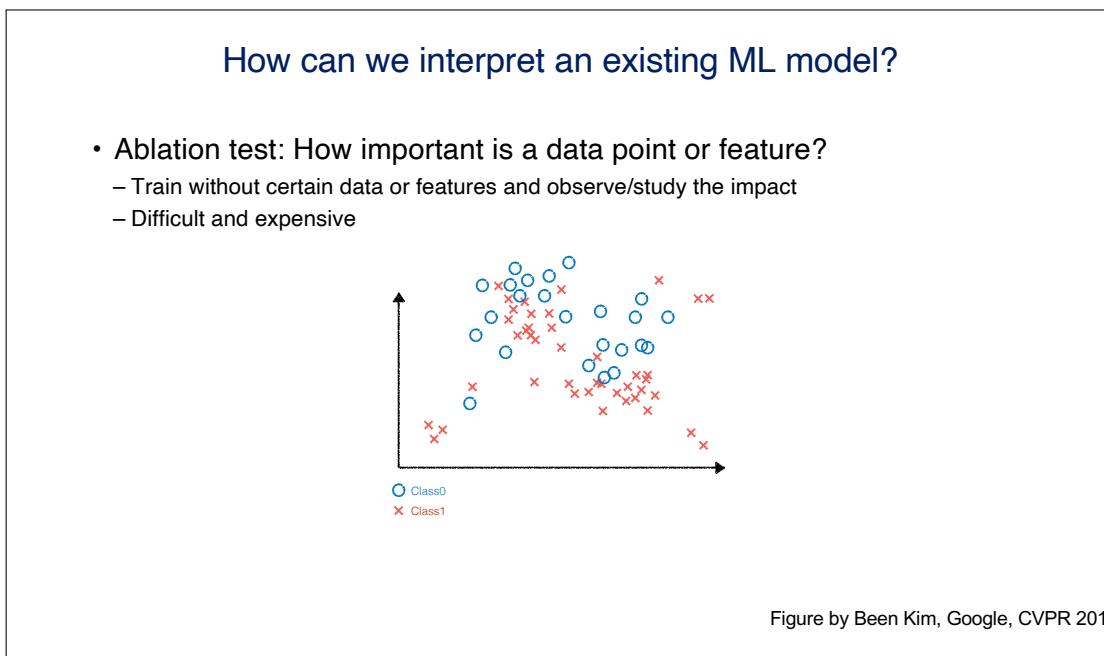


Figure 62: caption

## How can we interpret an existing ML model?

- Ablation test: How important is a data point or feature?
  - Train without certain data or features and observe/study the impact
  - Difficult and expensive

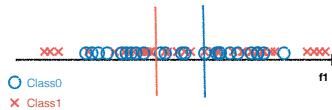


Figure by Been Kim, Google, CVPR 201

**Figure 63:** caption

## How can we interpret an existing ML model?

- Fit functions (use first derivatives)
  - Sensitivity analysis
  - Saliency maps

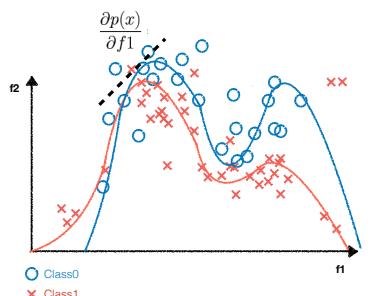


Figure by Been Kim, Google, CVPR 201

**Figure 64:** caption

## How can we interpret an existing ML model?

- Fit functions (use first derivatives)

- Sensitivity analysis
- Saliency maps

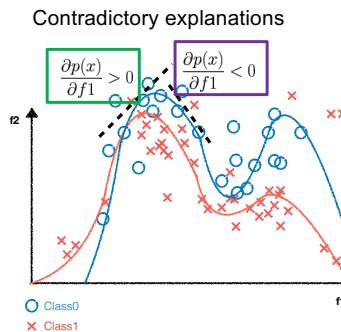


Figure by Been Kim, Google, CVPR 201

Figure 65: caption

## How can we interpret an existing ML model?



A trained  
machine learning model  
(e.g. neural network)



$p(z)$   
“dogness”

Why is this a  
dog?

Figure 66: caption

How can we interpret an existing ML model?

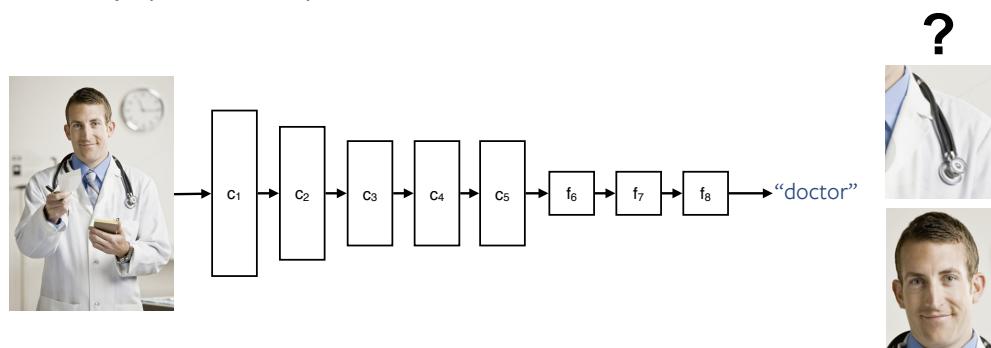


$$\frac{\partial p(z)}{\partial x_{i,j}}$$

**Figure 67:** caption

How can we interpret an existing ML model?

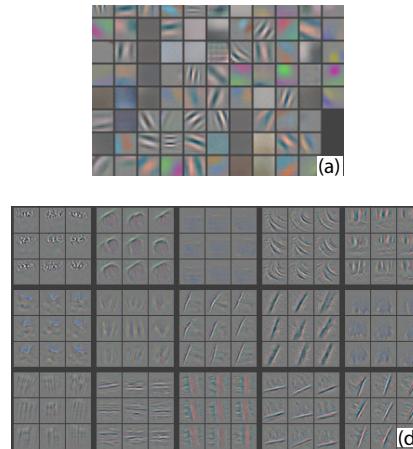
- Visualization and attribution:
  - Identify input features responsible for model decision



**Figure 68:** caption

## How can we interpret an existing ML model?

- Direct visualization of filters
- Easy to implement
- Limited practical value
  - First layers are easy to interpret (mostly low-level features)
  - Higher layers are more difficult to interpret (non-interpretable features)



<https://arxiv.org/abs/1311.2901>

Figure 69: caption

## How can we interpret an existing ML model?

- Problem: Visualization of filters has limited value
- Solution: Instead visualize activations generated by kernels
  - Strong response: Feature is present
  - Weak response: Feature is not present
  - Easy to implement
  - Easy to interpret for early layers

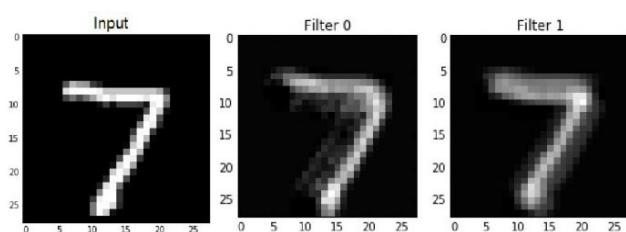


Figure 70: caption

## How can we interpret an existing ML model?

- Problem: Visualization of filters has limited value
- Solution: Instead visualize activations generated by kernels
  - Strong response: Feature is present
  - Weak response: Feature is not present
  - Easy to implement
  - Easy to interpret for early layers
  - Higher layers are more sparse
  - Channels may correspond to specific features



<https://arxiv.org/abs/1506.06579>

Figure 71: caption

### 2.4.1 Occlusions

## How can we interpret an existing ML model? Occlusions

- Idea: Mask out region in the input image and observe network output
- If masked out region causes a significant drop in confidence, the masked-out region is important



Figure 72: caption

### 2.4.2 Saliency maps

**How can we interpret an existing ML model?  
Saliency maps**

- **DeconvNet**
  - Given a trained network and an image
  - Choose activation at one layer (set all others to zero)
  - Invert network

The diagram illustrates the DeconvNet architecture. It shows the forward pass from left to right and the inverse pass from right to left. The forward pass consists of:

- Layer Above Reconstruction
- Max Unpooling
- Unpooled Maps
- Rectified Linear Function
- Rectified Unpooled Maps
- Convolutional Filtering ( $F^*$ )
- Reconstruction

The inverse pass (backward pass) consists of:

- Pooled Maps
- Max Pooling
- Rectified Feature Maps
- Rectified Linear Function
- Feature Maps
- Convolutional Filtering ( $F$ )
- Layer Below Pooled Maps

Switches connect the forward and inverse paths between the Max Unpooling and Rectified Unpooled Maps stages. Arrows indicate the flow of data from left to right and right to left.

- No training involved
- Backward pass in network is almost identical to backpropagation (apart from ReLUs)

Visualizing and Understanding Convolutional Networks, M. D. Zeiler and R. Fergus, ECCV 2011

Figure 73: caption

**How can we interpret an existing ML model?  
Saliency maps**

- **DeconvNet**
  - Given a trained network and an image
  - Choose activation at one layer (set all others to zero)
  - Invert network

The diagram shows the DeconvNet architecture as 3D volumes. The forward pass (left to right) consists of:

- Layer Above Reconstruction
- Unpooling
- Unpooled Maps
- Max Locations "Switches"
- Pooling
- Pooled Maps

The inverse pass (right to left) consists of:

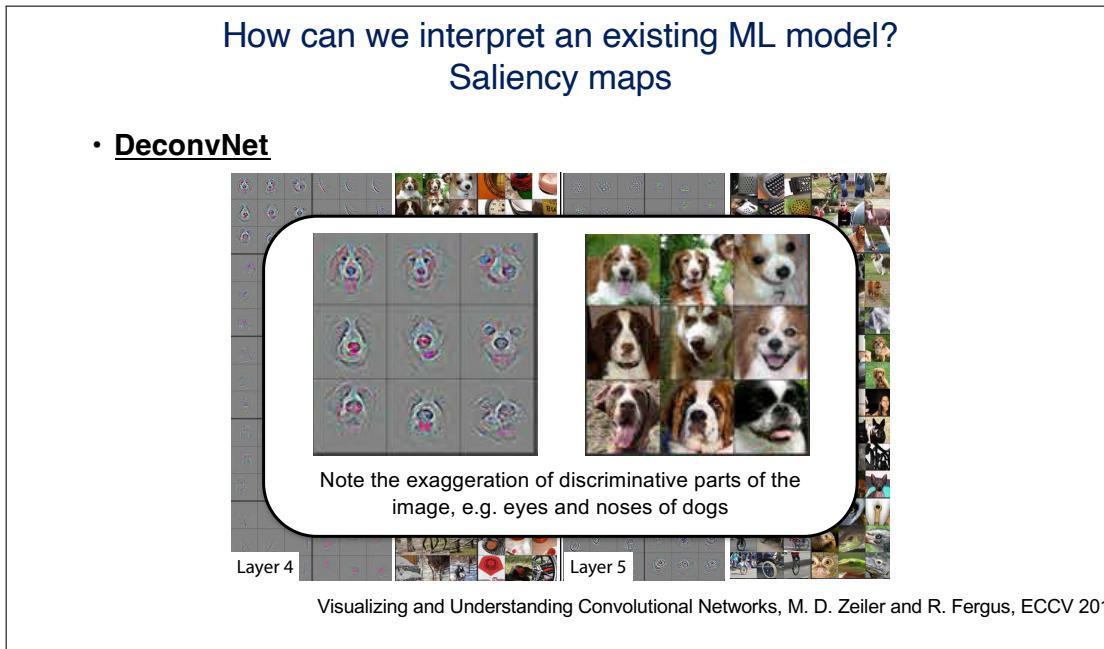
- Rectified Feature Maps
- Rectified Linear Function
- Feature Maps
- Convolutional Filtering ( $F$ )
- Layer Below Pooled Maps

Arrows indicate the flow of data between corresponding layers in the forward and inverse paths. The 3D nature highlights the spatial dimensions of the feature maps.

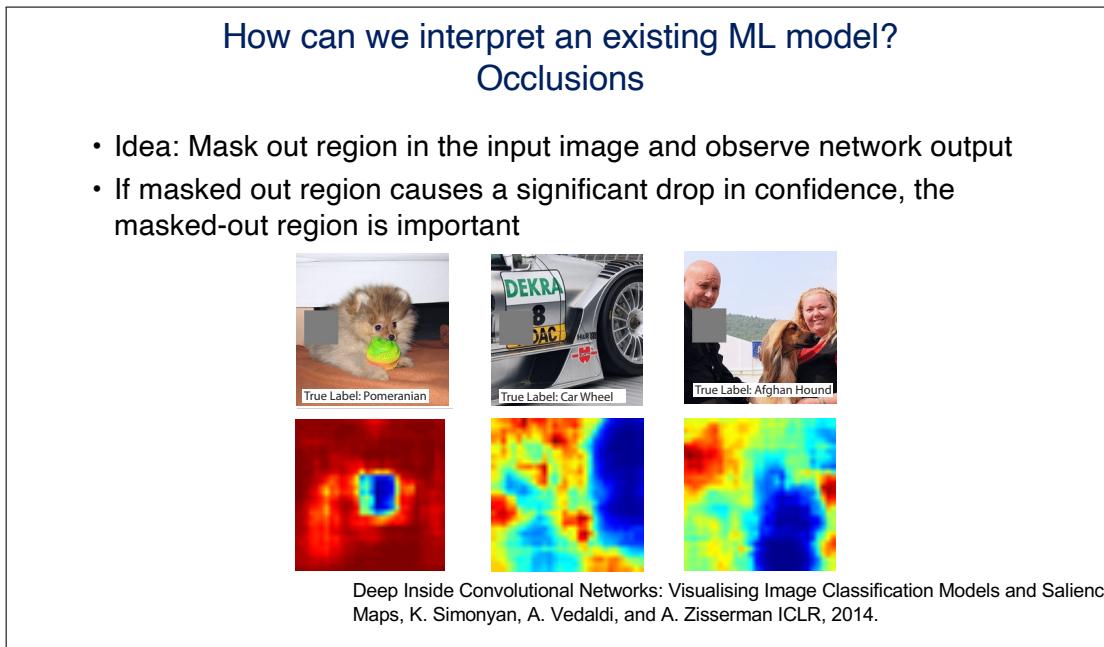
- No training involved
- Backward pass in network is almost identical to backpropagation (apart from ReLUs)

Visualizing and Understanding Convolutional Networks, M. D. Zeiler and R. Fergus, ECCV 2011

Figure 74: caption

**Figure 75:** caption

### 2.4.3 Occlusions

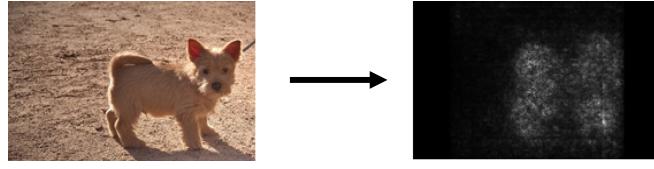
**Figure 76:** caption

#### 2.4.4 Saliency maps

**How can we interpret an existing ML model?  
Saliency maps**

- **Question:**
  - Which pixels are most significant to a neuron?
  - How would they need to change to most affect the activation of the neuron?
- **Solution:**
  - Use back propagation but differentiate activation with respect to **input pixels, not weights**

weights of the network are fixed  $\frac{\partial h}{\partial x_i}$  

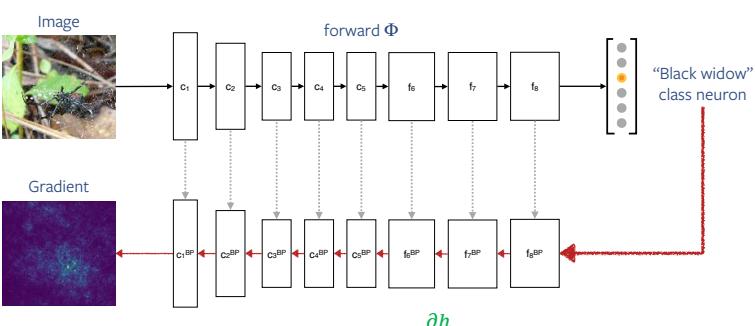


Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, K. Simonyan, A. Vedaldi, and A. Zisserman ICLR, 2014.

**Figure 77:** caption

**How can we interpret an existing ML model?  
Saliency maps**

- **Gradient (backpropagation)**
  - Define loss as activation of arbitrary neuron in any layer (last layer is most interesting)



$\text{backward pass} = \frac{\partial h}{\partial x_i}$  Salient pixels light up

Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, K. Simonyan, A. Vedaldi, and A. Zisserman ICLR, 2014.

**Figure 78:** caption

**How can we interpret an existing ML model?**  
**Saliency maps**

- **Guided backpropagation**
  - Improve results by “guiding” the backpropagation process
  - Idea:
    - Positive gradients = features the neuron is interested in
    - Negative gradients = features the neuron is not interested in
  - Set all negative gradients in the backpropagation to zero
  - Propagating through the ReLU:

Forward                      Backprop  
Deconvnet                      Guided backprop

**Figure 79:** caption

**How can we interpret an existing ML model?**  
**Saliency maps**

**Deconvolution**

Visualizing and understanding convolutional neural networks  
Zeiler & Fergus, ECCV, 2014

**Gradient (backpropagation)**

Deep inside convolutional networks:  
Visualising image classification models and saliency maps  
Simonyan, Vedaldi, Zisserman, ICLR 2014

**Guided backpropagation**

Striving for simplicity: The all convolutional net  
Springenberg, Dosovitsky, Brox, Riedmiller, ICLR, 2015

**Figure 80:** caption

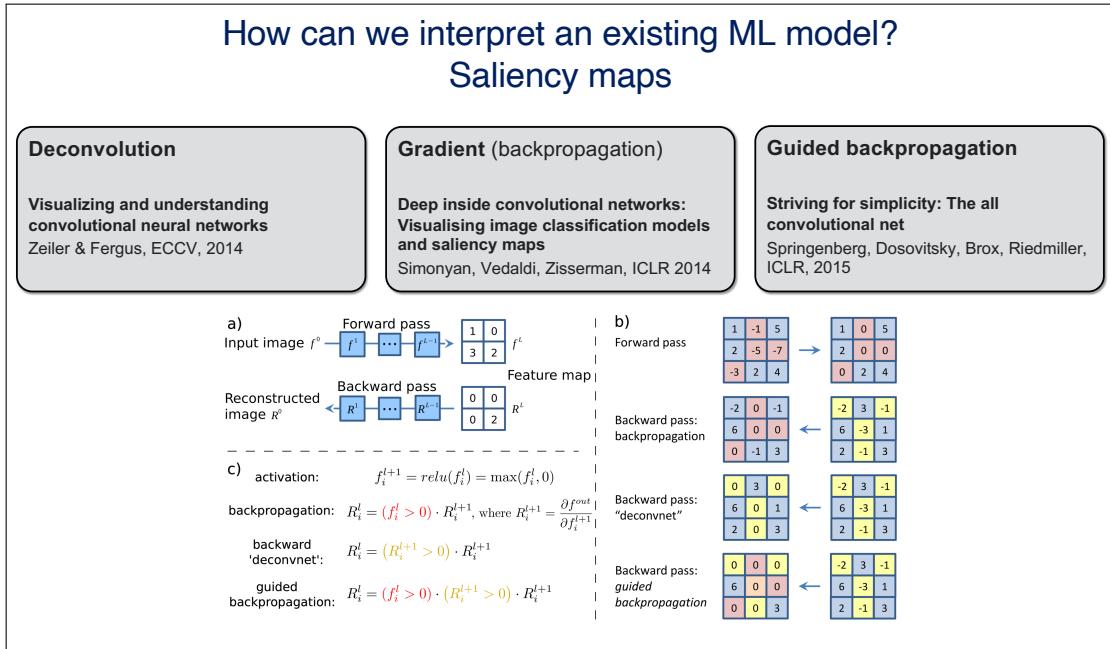


Figure 81: caption

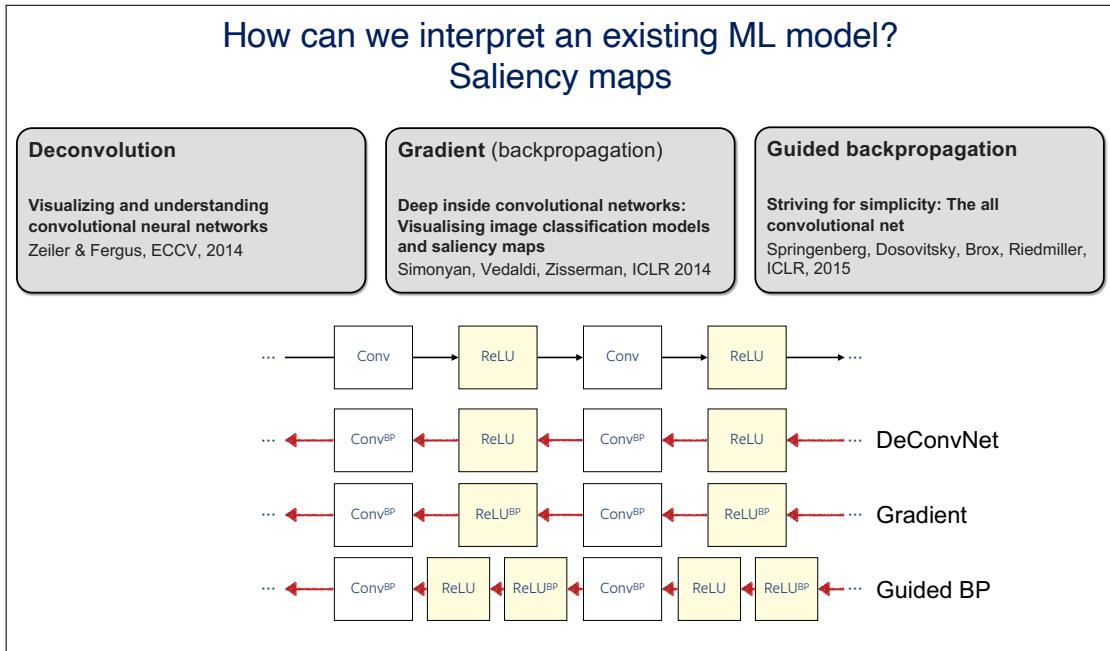
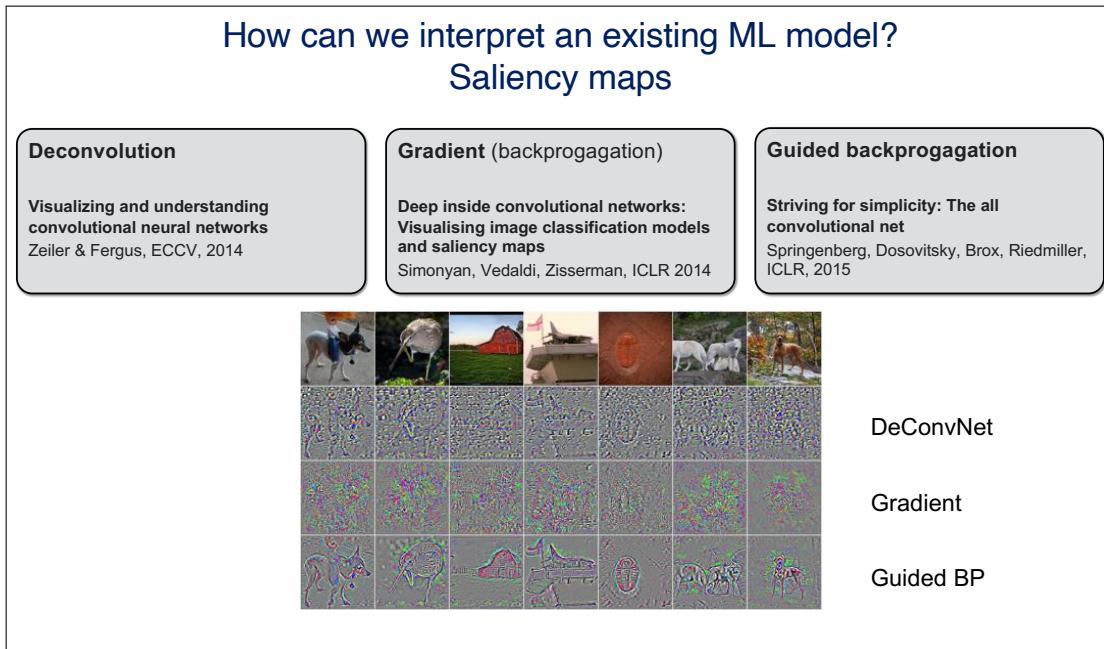
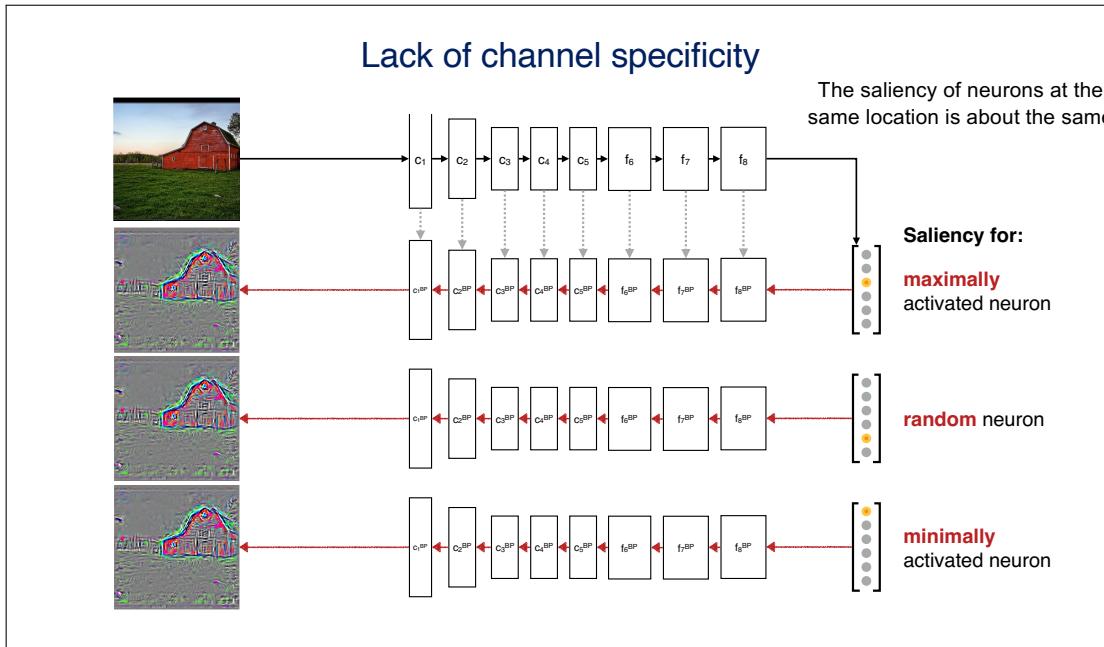


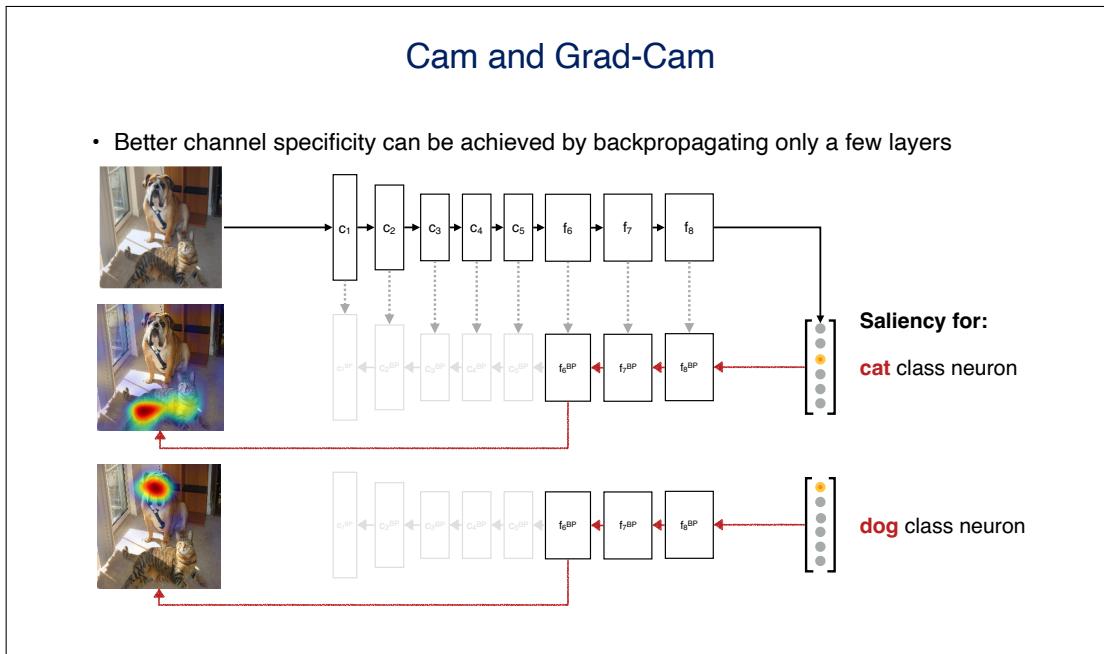
Figure 82: caption

**Figure 83:** caption

## 2.5 Lack of channel specificity

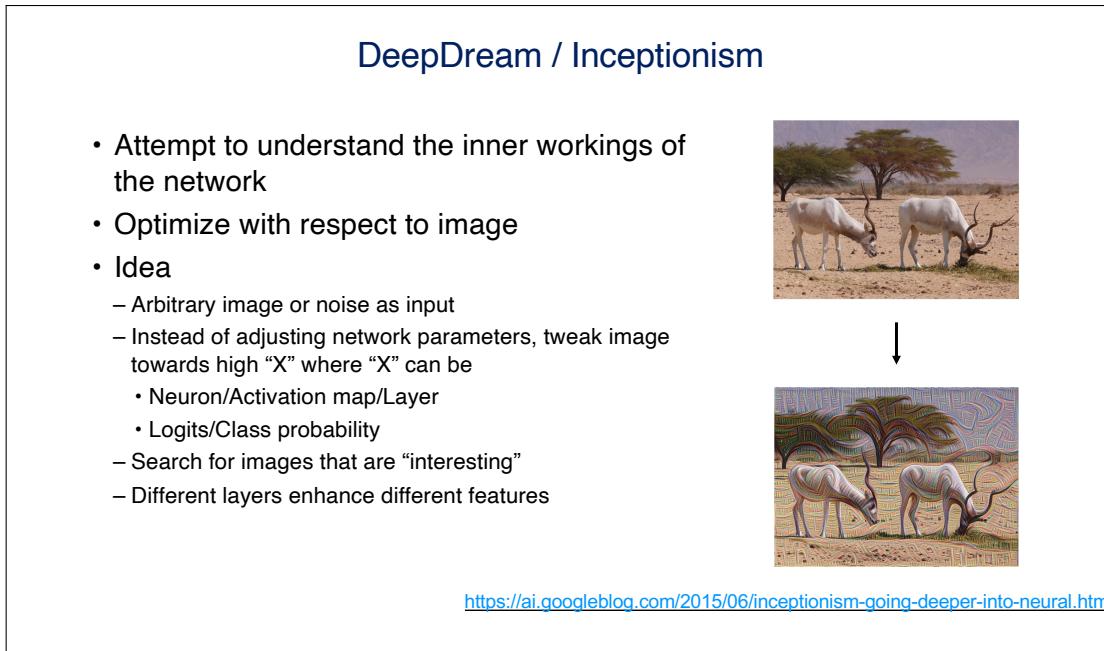
**Figure 84:** caption

## 2.6 Cam and Grad-Cam



**Figure 85:** caption

## 2.7 DeepDream / Inceptionism



**Figure 86:** caption

### DeepDream / Inceptionism

- Find an image  $x$  such that the activation  $\phi_n(x)$  at layer  $n$  is high

$$\max_x \phi_n(x) - \lambda \mathcal{R}(x)$$

Regularizer: e.g. L1 or L2 norm of image



<https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.htm>

**Figure 87:** caption

## 2.8 Inversion

### Inversion

- Inversion attempts to construct an image from a layer activation  $\hat{y}$ :

$$\hat{x} = \min_x (\|\phi(x) - \hat{y}\|_2^2 + \lambda \mathcal{R}(x))$$

- $\hat{x}$  is the reconstructed image
- $\phi(x)$  is the network output for input image  $x$
- $\hat{y}$  is the desired activation
- $\mathcal{R}$  is the regularizer

**Figure 88:** caption

### 3 Robustness: Adversarial Methods

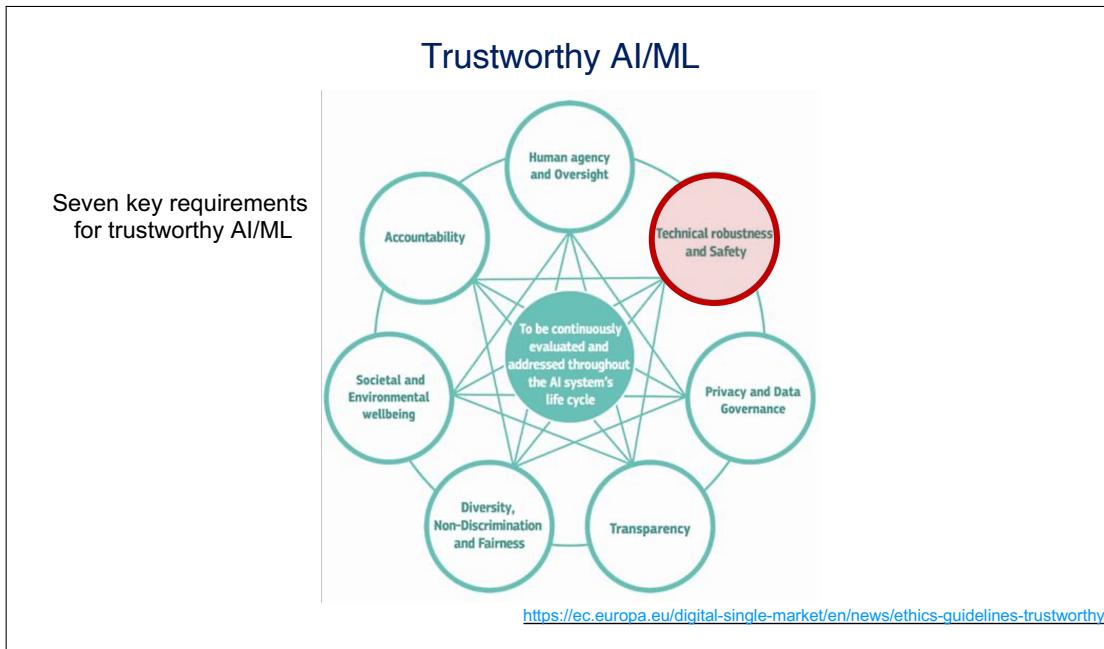


Figure 89: caption

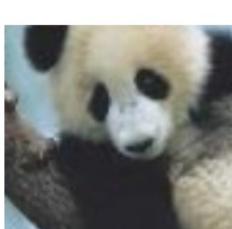
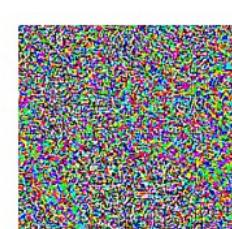
#### 3.1 Adversarial Attacks



Figure 90: caption

## Adversarial attacks

- Unintuitive behaviour of NNs

 $+ .007 \times$  $=$ 

“panda”  
57.7% confidence

“gibbon”  
99.3 % confidence

**Figure 91:** caption

### 3.1.1 Perturbation

## Adversarial attacks - Perturbation

- Assume a linear classifier:

$$\theta^T x$$

- We can think of an adversarial example that contains a small, non-perceivable perturbation to the input. Let's denote the perturbation as  $\eta$ :

$$\tilde{x} = x + \eta$$

- Then, the logits of the classifier would be

$$\begin{aligned}\theta^T \tilde{x} &= \theta^T(x + \eta) \\ &= \theta^T x + \theta^T \eta\end{aligned}$$

**Figure 92:** caption

### Adversarial attacks - Perturbation

- Given a small perturbation  $\eta$ , the effect of the perturbation on the logits of the classifier is given by  $\theta^T \eta$ .
- Idea:
  - Find a  $\eta$  that causes a change that is non-perceivable and ostensibly innocuous to the human eye, yet destructive and adverse enough for the classifier to the extent that its predictions are no longer accurate.
  - An adversarial example is one that which maximizes the value of  $\theta^T \eta$  to sway the model into making a wrong prediction

**Figure 93:** caption

### Adversarial attacks - Perturbation

- Problem:
  - Need a constraint on  $\eta$ ; otherwise, one could just apply a large perturbation to the input
- Solution:
  - Apply a constraint such that

$$\|\eta\|_\infty \leq \epsilon \quad \|x\|_\infty = \max_i \{|x_i|\}$$

- Assume a perturbation:

$$\eta = \epsilon \cdot \text{sign}(\theta)$$

- What are the bounds of this perturbation?

I. Goodfellow et al. ICLR 2015

**Figure 94:** caption

### Adversarial attacks - Perturbation

- What are the bounds of this perturbation?

$$\eta = \epsilon \cdot \text{sign}(\theta)$$

$$\theta^T \eta = \epsilon \cdot \theta^T \text{sign}(\theta)$$

$$= \epsilon \|\theta\|_1$$

$$= \epsilon m n$$

Here the average magnitude of an element of  $\theta$  is given by  $m$

I. Goodfellow et al. ICLR 2015

**Figure 95:** caption

### Adversarial attacks - Perturbation

- This means that the change in activation given by the perturbation increases linearly with respect to  $n$  (or the dimensionality).
- If  $n$  is large, one can expect even a small perturbation capped at  $\epsilon$  to produce a perturbation big enough to render the model susceptible to an adversarial attack.
- Remember that for images  $n = \text{no. of pixels}$
- Such perturbed examples are referred to as **adversarial examples**

I. Goodfellow et al. ICLR 2015

**Figure 96:** caption

### 3.1.2 Fast Gradient Sign Method

#### Adversarial attacks - Fast Gradient Sign Method

Key idea:

- Perform gradient descent in order to maximize the loss (the goal of adversarial attack).
- Consider the input image  $x$  to be a trainable parameter and compute the gradient with respect to the input image to create a perturbation.

$$\eta = \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\theta, x, y))$$

- An adversarial example can be created as:

$$\tilde{x} = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\theta, x, y))$$

I. Goodfellow et al. ICLR 2015

Figure 97: caption

## 3.2 Adversarial attacks

#### Adversarial attacks

How can you use adversarial attacks?

1. Generate adversarial examples
2. Add the generated adversarial examples to the training set
3. Retrain model using training set

Adversarial data augmentation

Figure 98: caption

## A References

### References

- Trustworthy ML
  - <https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai>
- Federated Learning: Challenges, Methods, and Future Directions
  - <https://arxiv.org/pdf/1908.07873.pdf>
- Privacy-Preserving Deep Learning
  - [https://www.cs.cornell.edu/~shmat/shmat\\_ccs15.pdf](https://www.cs.cornell.edu/~shmat/shmat_ccs15.pdf)
- Communication-Efficient Learning of Deep Networks from Decentralized Data
  - <http://proceedings.mlr.press/v54/mcmahan17a/mcmahan17a.pdf>

Figure 99: caption

### References

- The Future of Digital Health with Federated Learning
  - <https://arxiv.org/abs/2003.08119>
- Secure, privacy-preserving and federated machine learning in medical imaging
  - <https://www.nature.com/articles/s42256-020-0186-1>

Figure 100: caption

## References

- Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)
  - <https://arxiv.org/abs/1711.11279>
- The building blocks of interpretability
  - <https://distill.pub/2018/building-blocks/>
- Understanding deep neural networks through deep visualization
  - <https://arxiv.org/abs/1506.06579>
- Visualizing and Understanding Convolutional Networks
  - <https://arxiv.org/abs/1311.2901>
- Understanding Neural Networks Through Deep Visualization
  - <https://arxiv.org/abs/1506.06579>

**Figure 101:** caption