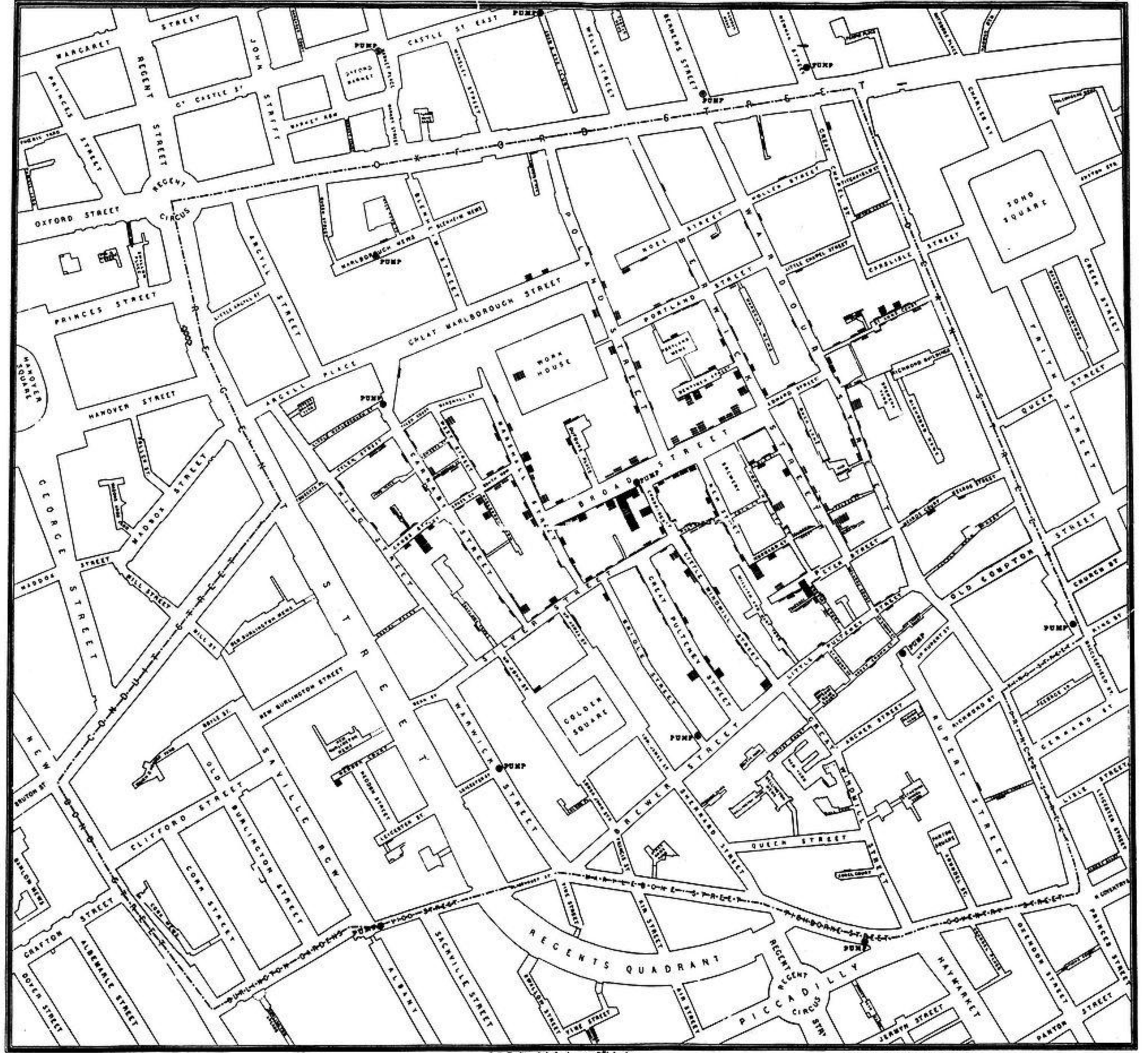
DEEP LEARNING WITH DIFFERENTIAL PRIVACY

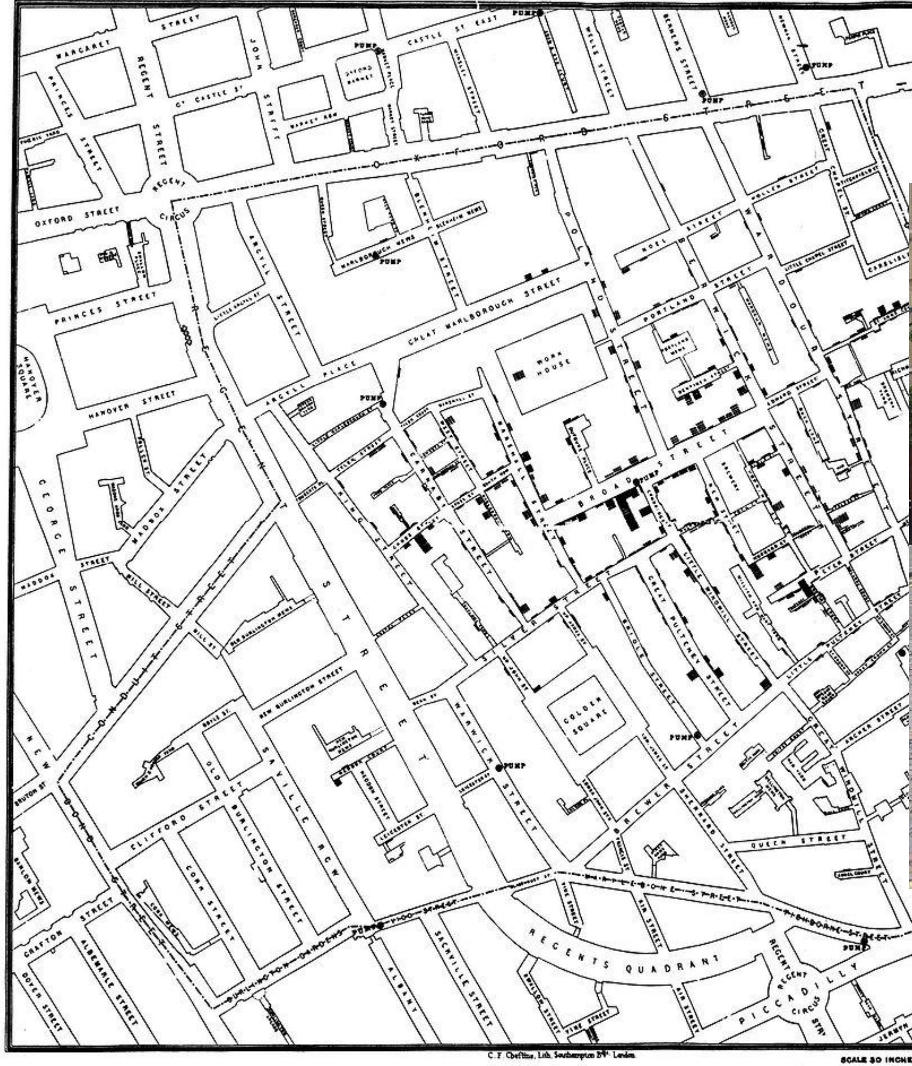
Martin Abadi, Andy Chu, Ian Goodfellow*,
Brendan McMahan, Ilya Mironov, Kunal Talwar, Li Zhang
Google

* Open AI



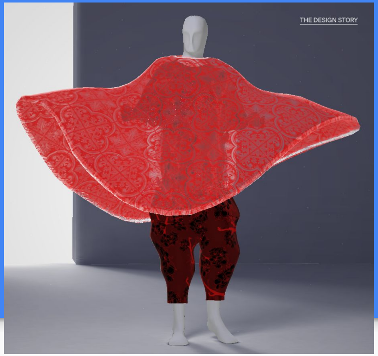
C.F. Cheffins, Lith, Southampton Bldgs London.

SCALE 30 INCHES TO A MILE.



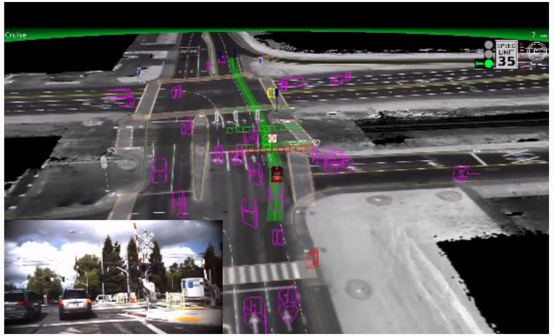
OXFORD STREET
REGENT CIRCUS
GREAT MARLBOROUGH STREET
PRINCES STREET
HANOVER STREET
HANOVER SQUARE
GEORGE STREET
CONDUIT STREET
NEW BOND STREET
MADDOX STREET
MILL STREET
REGENT STREET
ARGYLL PLACE
MARLBOROUGH MEWS
POLAND STREET
BERWICK STREET
WARDOUR STREET
WORK HOUSE
BROAD STREET
PUMP
GOLDEN SQUARE
WARWICK STREET
BREWER STREET
GREAT PULTENEY STREET
QUEEN STREET
CLIFFORD STREET
OLD BURLINGTON STREET
NEW BURLINGTON STREET
SAVILLE ROW
CORK STREET
GRAFTON STREET
ALBEMARLE STREET
DOVER STREET
ALBANY
SACKVILLE STREET
REGENTS QUADRANT
PICCADILLY
HAYMARKET
WELLS STREET
BERNERS STREET
CASTLE ST EAST
MARGARET STREET
CHARLES ST
C.F. Cheffins, Lith, Southampton Bldgs London
SCALE 30 INCHES TO A MILE.

Deep Learning



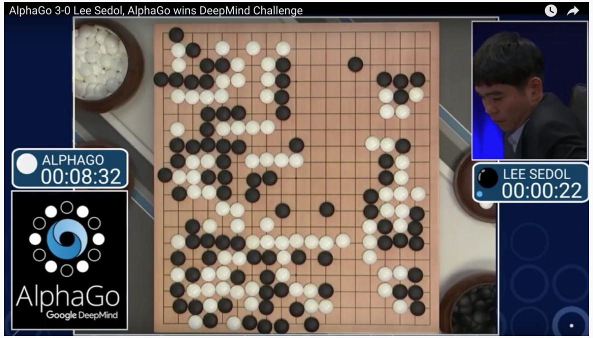
Fashion

- Cognitive tasks: speech, text, image recognition
- Natural language processing: sentiment analysis, translation
- Planning: games, autonomous driving

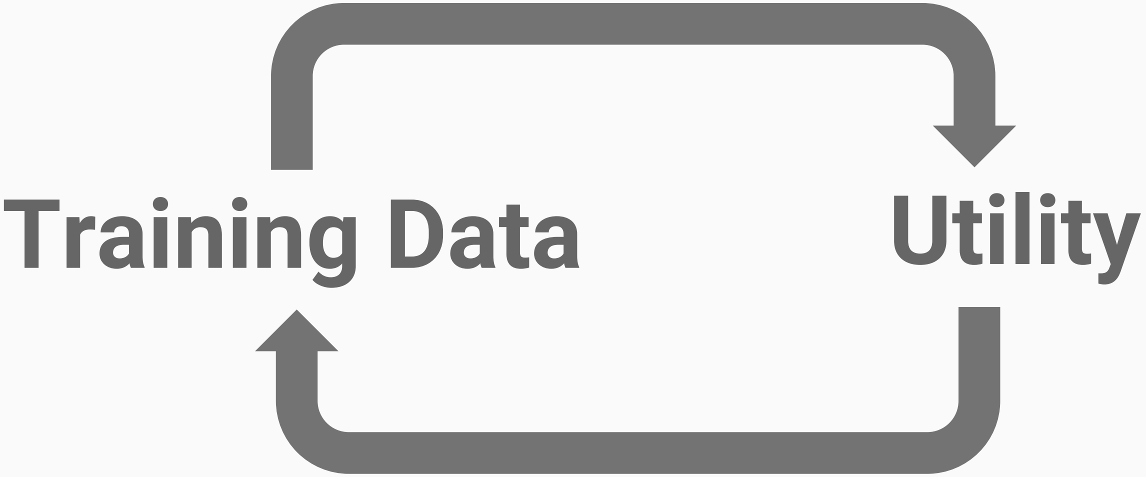


Self-driving cars

Translation



Gaming

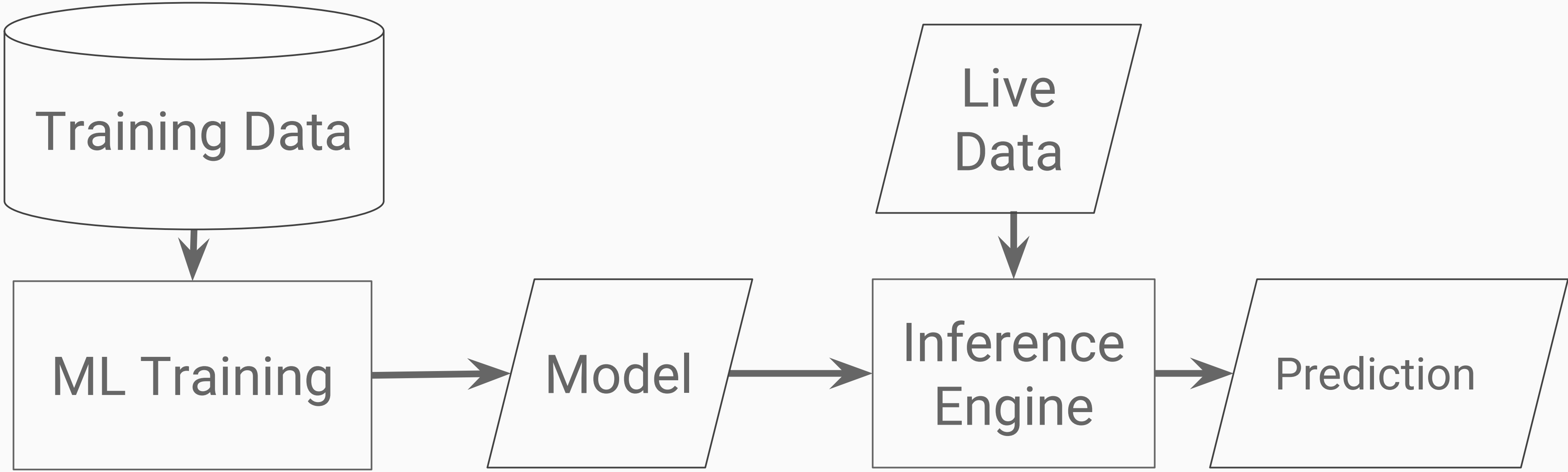

Training Data
Utility

Privacy of Training Data

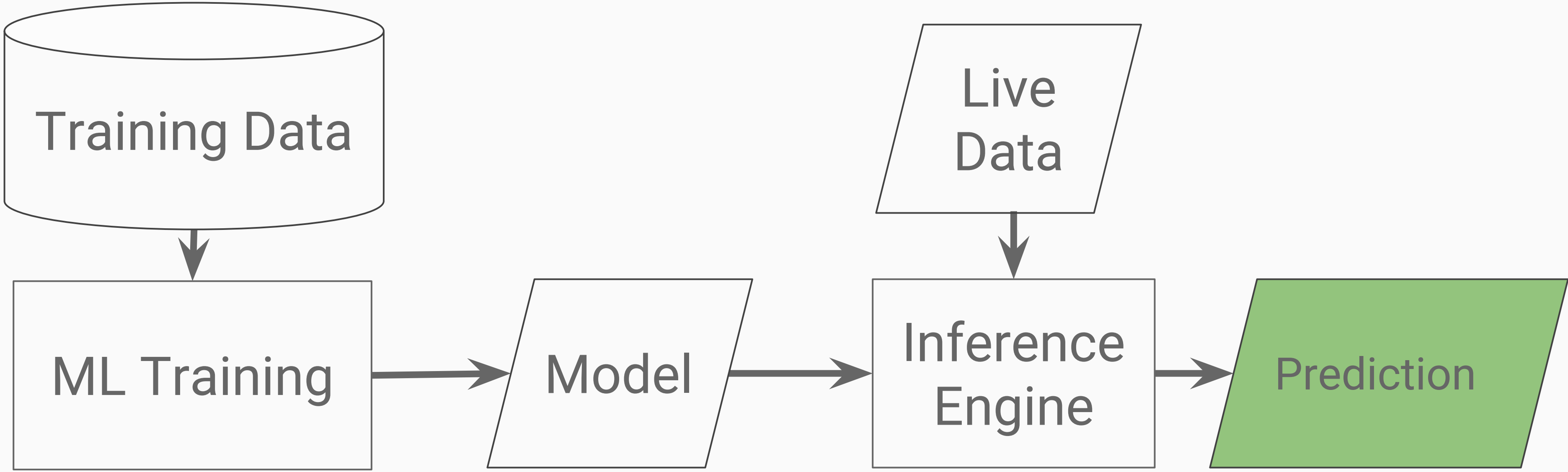
- ✔ Data encryption in transit and at rest
- ✔ Data retention and deletion policies
- ✔ ACLs, monitoring, auditing

What do models reveal about training data?

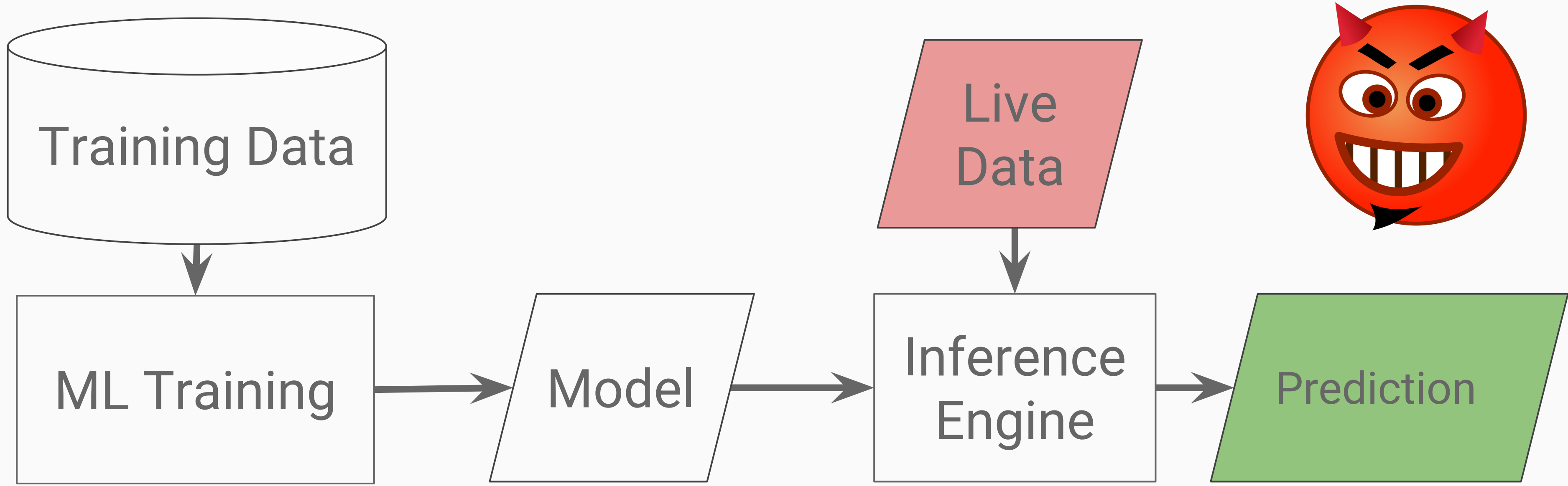
ML Pipeline and Threat Model



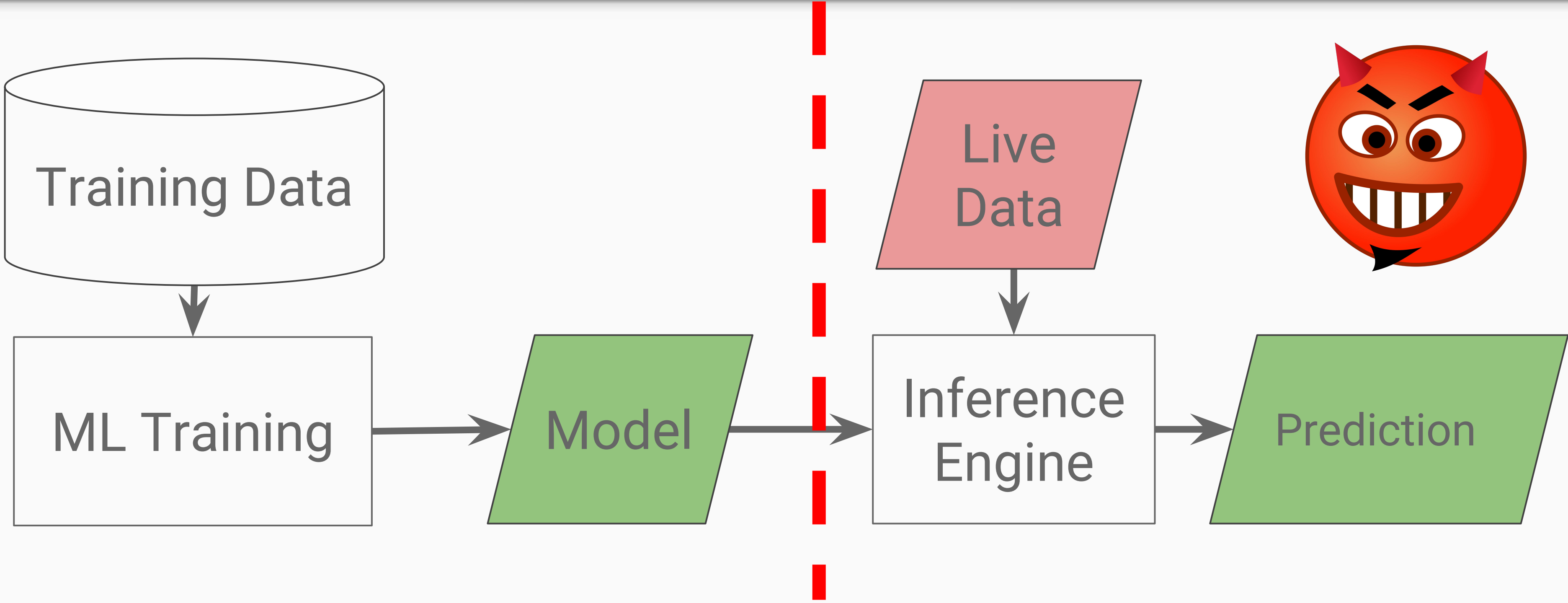
ML Pipeline and Threat Model



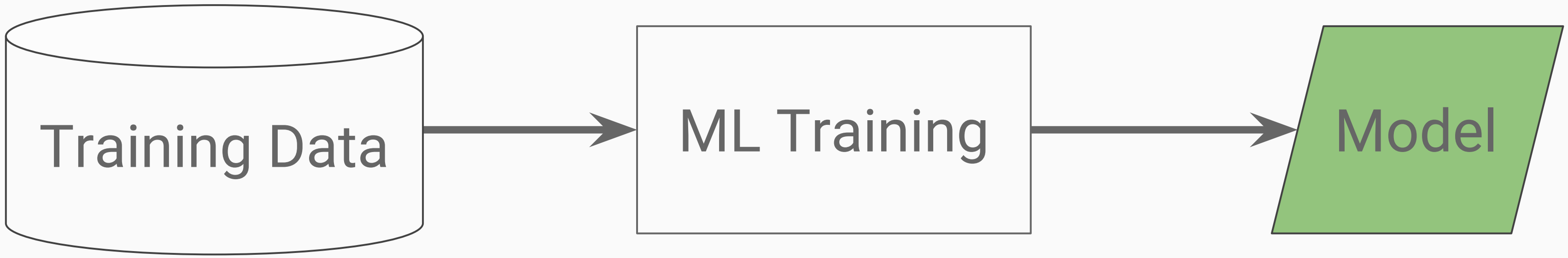
ML Pipeline and Threat Model



ML Pipeline and Threat Model



ML Pipeline and Threat Model



Machine Learning Privacy Fallacy

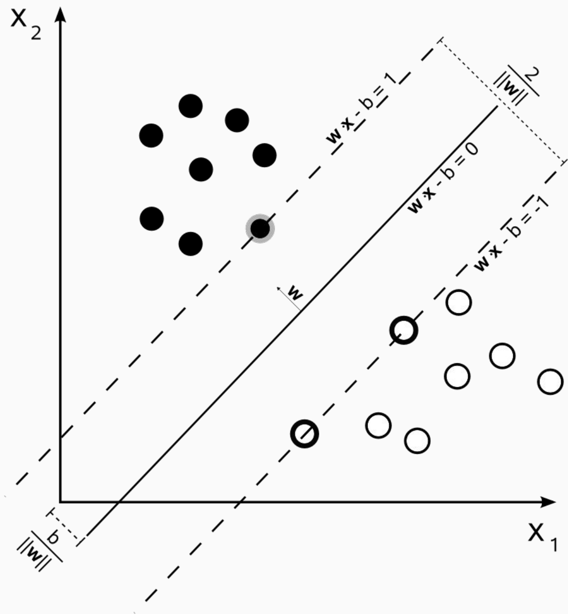
Since our ML system is good, it automatically protects privacy of training data.

Machine Learning Privacy Fallacy

- Examples when it just ain’t so:
  - Person-to-person similarities
  - Support Vector Machines
- Models can be very large
  - Millions of parameters
- Empirical evidence to the contrary:
  - M. Fredrikson, S. Jha, T. Ristenpart, “Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures”, CCS 2015
  - R. Shokri, M. Stronati, V. Shmatikov, “Membership Inference Attacks against Machine Learning Models”, https://arxiv.org/abs/1610.05820

Machine Learning Privacy Fallacy

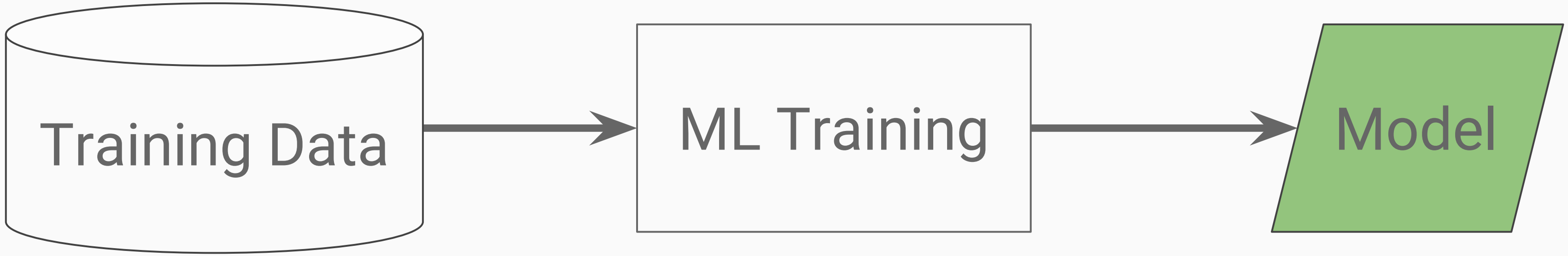
- Examples when it just ain’t so:
  - Person-to-person similarities
  - Support Vector Machines
- Models can be very large
  - Millions of parameters



Model Inversion Attack

- M. Fredrikson, S. Jha, T. Ristenpart, “Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures”, CCS 2015

- R. Shokri, M. Stronati, V. Shmatikov, “Membership Inference Attacks against Machine Learning Models”, https://arxiv.org/abs/1610.05820


Training Data
ML Training
Model

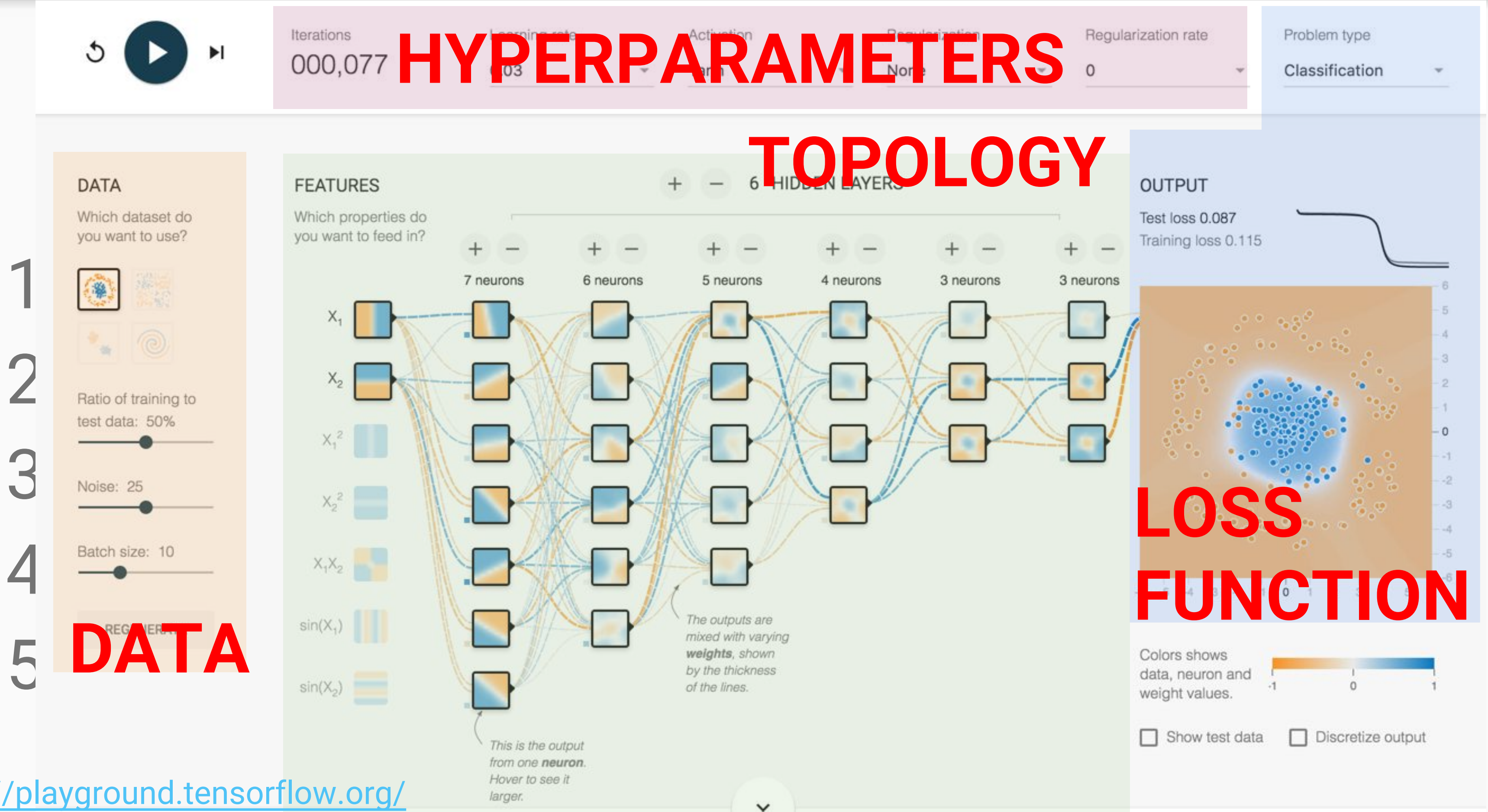
Deep Learning Recipe

1. Loss function
2. Training / Test data
3. Topology
4. Training algorithm
5. Hyperparameters

Deep Learning Recipe

1. Loss function	softmax loss
2. Training / Test data	MNIST and CIFAR-10
3. Topology	
4. Training algorithm	
5. Hyperparameters	

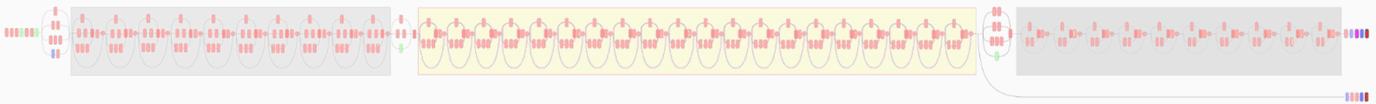
Deep Learning Recipe



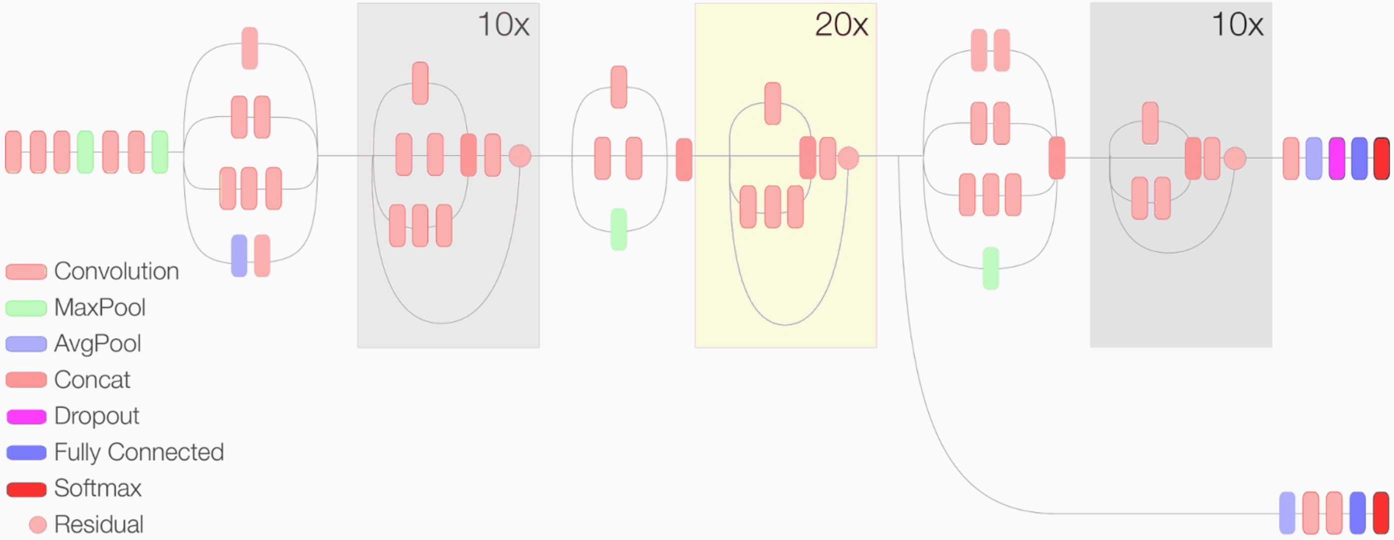
http://playground.tensorflow.org/

Layered Neural Network

Inception Resnet V2 Network



Compressed View



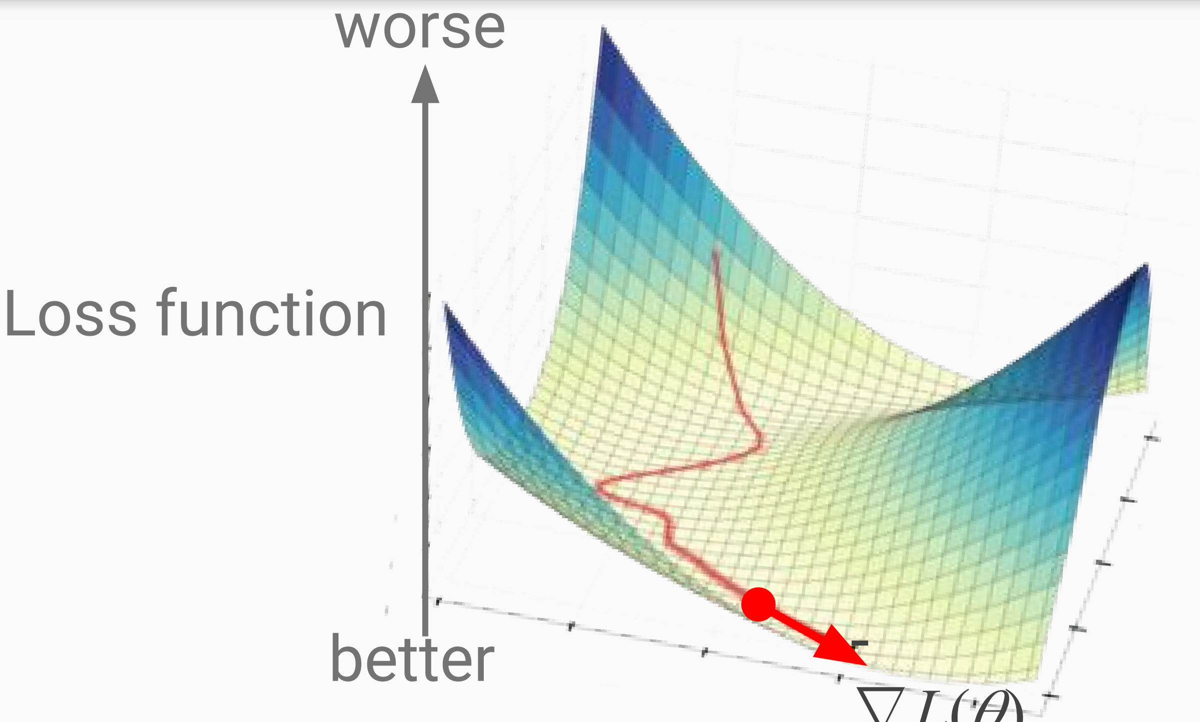
Deep Learning Recipe

1. Loss function	softmax loss
2. Training / Test data	MNIST and CIFAR-10
3. Topology	neural network
4. Training algorithm	
5. Hyperparameters	

Deep Learning Recipe

1. Loss function	softmax loss
2. Training / Test data	MNIST and CIFAR-10
3. Topology	neural network
4. Training algorithm	SGD
5. Hyperparameters	

Gradient Descent



Gradient Descent

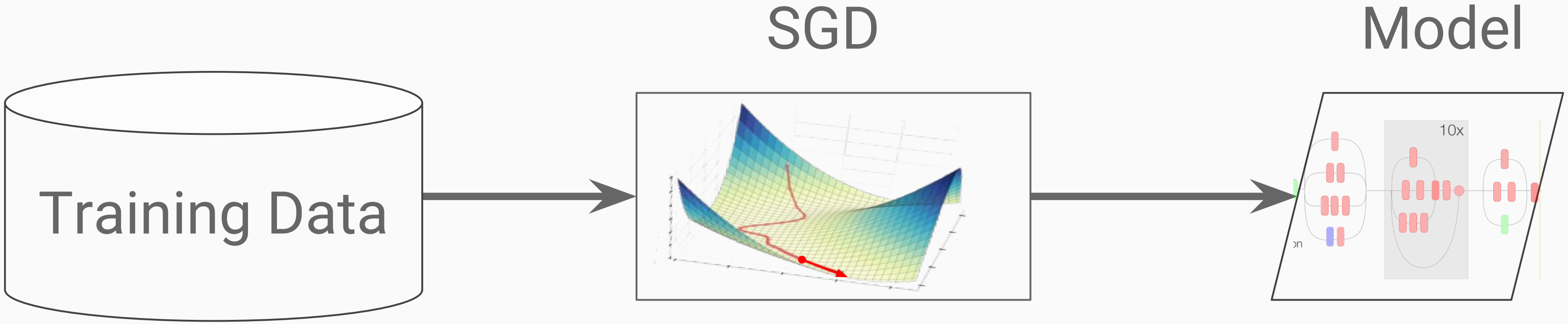
Compute $\nabla L(\theta_1)$ → $\theta_2 := \theta_1 - \eta \nabla L(\theta_1)$ → Compute $\nabla L(\theta_2)$ → $\theta_3 := \theta_2 - \eta \nabla L(\theta_2)$ →

Stochastic Gradient Descent

Compute $\nabla L(\theta_1)$ on random sample → $\theta_2 := \theta_1 - \eta \nabla L(\theta_1)$ → Compute $\nabla L(\theta_2)$ on random sample → $\theta_3 := \theta_2 - \eta \nabla L(\theta_2)$ →

Deep Learning Recipe

1.	Loss function	softmax loss
2.	Training / Test data	MNIST and CIFAR-10
3.	Topology	neural network
4.	Training algorithm	SGD
5.	Hyperparameters	tune experimentally


Training Data
SGD
Model
10x

Differential Privacy

Differential Privacy

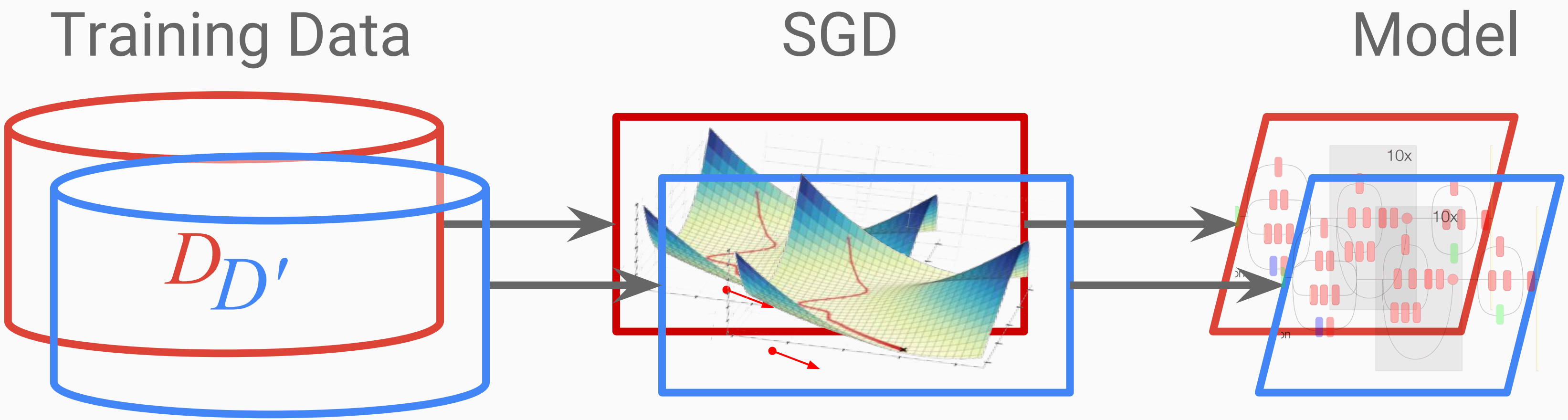
(ε, δ)-Differential Privacy: The distribution of the output $M(D)$ on database D is (nearly) the same as $M(D')$:

$$\forall S: \qquad \Pr[M(D)\in S] \leq \exp(\varepsilon) \cdot \Pr[M(D')\in S]+\delta.$$

quantifies information leakage

allows for a small probability of failure

Interpreting Differential Privacy



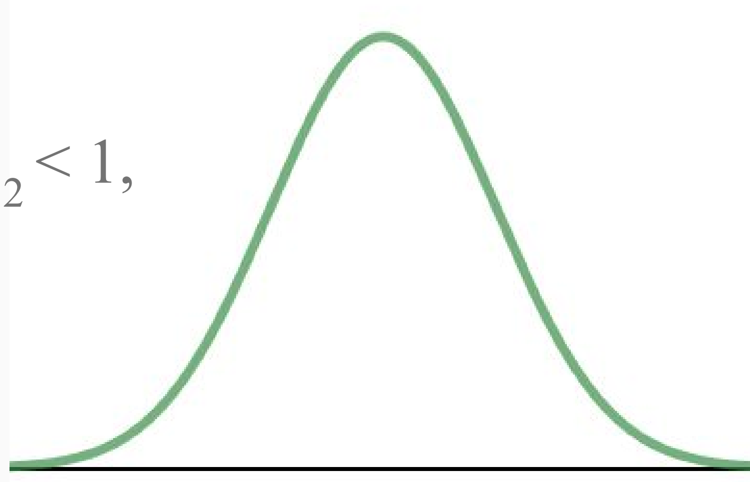
Differential Privacy: Gaussian Mechanism

If ℓ_2-sensitivity of $f : \mathcal{D} \to \mathbb{R}^n$:

$$\max_{D,D'} \|f(D) - f(D')\|_2 < 1,$$

then the Gaussian mechanism

$$f(D) + N^n(0, \sigma^2)$$



offers (ε, δ)-differential privacy, where $\delta \approx \exp(-(\varepsilon\sigma)^2/2)$.

Dwork, Kenthapadi, McSherry, Mironov, Naor, "Our Data, Ourselves", Eurocrypt 2006

Simple Recipe

To compute f with differential privacy

1. Bound sensitivity of f
2. Apply the Gaussian mechanism

Basic Composition Theorem

If f is $(\varepsilon_1, \delta_1)$-DP and g is $(\varepsilon_2, \delta_2)$-DP, then

$$f(D), g(D) \text{ is } (\varepsilon_1+\varepsilon_2, \delta_1+\delta_2)\text{-DP}$$

Simple Recipe for Composite Functions

To compute composite f with differential privacy

1. Bound sensitivity of f's components
2. Apply the Gaussian mechanism to each component
3. Compute total privacy via the composition theorem

Deep Learning with Differential Privacy

Deep Learning

1.	Loss function	softmax loss
2.	Training / Test data	MNIST and CIFAR-10
3.	Topology	neural network
4.	Training algorithm	SGD
5.	Hyperparameters	tune experimentally

Our Datasets: "Fruit Flies of Machine Learning"

MNIST dataset:

70,000 images

28×28 pixels each

CIFAR-10 dataset:

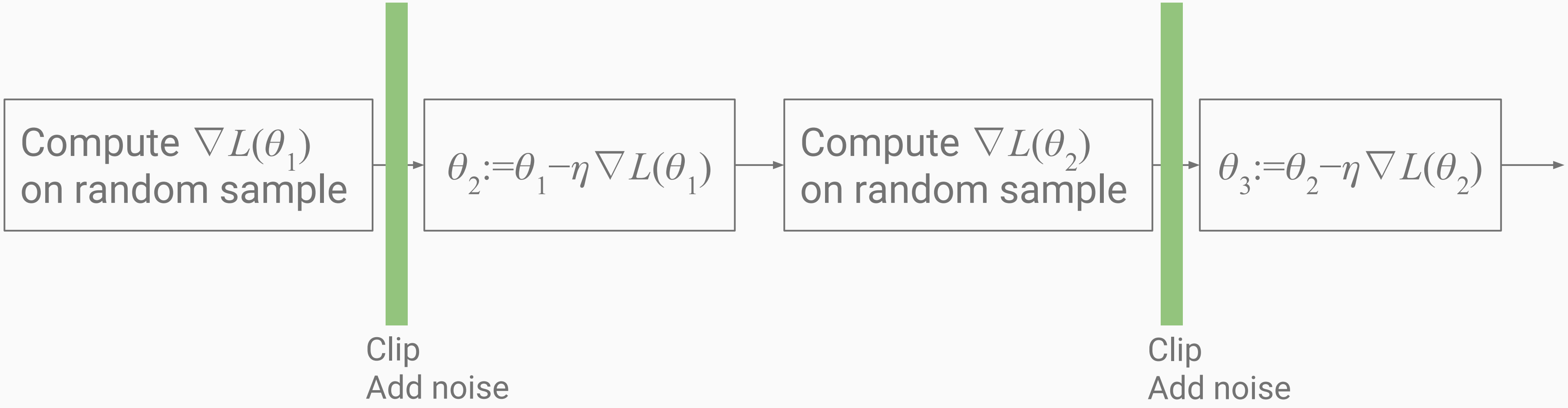
60,000 color images

32×32 pixels each

Differentially Private Deep Learning

1. Loss function	softmax loss
2. Training / Test data	MNIST and CIFAR-10
3. Topology	PCA + neural network
4. Training algorithm	SGD
5. Hyperparameters	tune experimentally

Stochastic Gradient Descent with Differential Privacy



Differentially Private Deep Learning

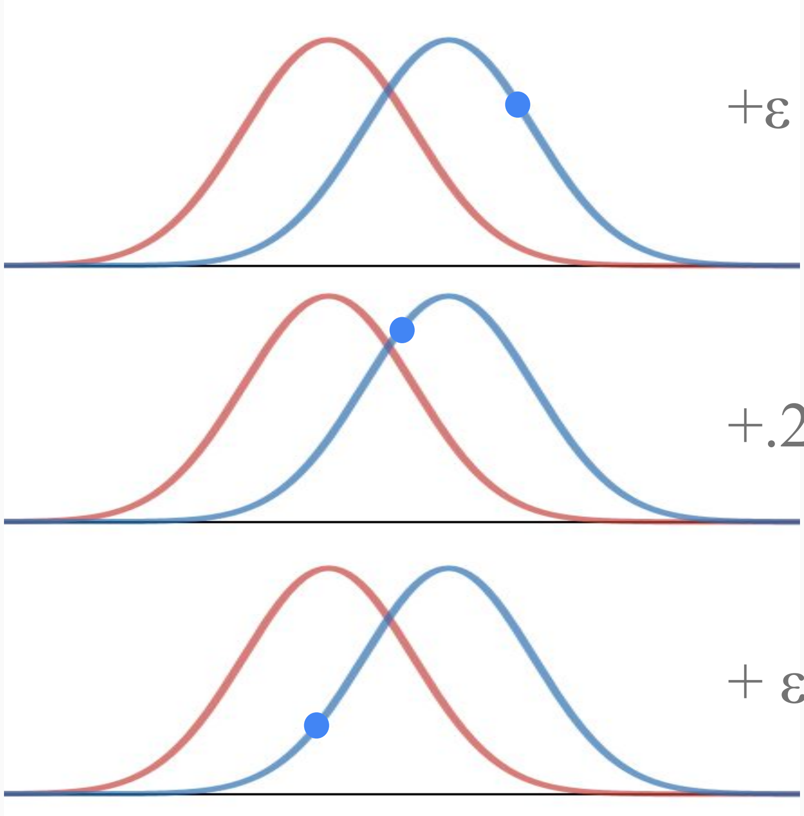
1.	Loss function	softmax loss
2.	Training / Test data	MNIST and CIFAR-10
3.	Topology	PCA + neural network
4.	Training algorithm	Differentially private SGD
5.	Hyperparameters	tune experimentally

Naïve Privacy Analysis

1. Choose $\sigma = \frac{\sqrt{2\log 1/\delta}}{\varepsilon}$	$= 4$
2. Each step is (ε, δ)-DP	$(1.2, 10^{-5})$-DP
3. Number of steps T	$10{,}000$
4. Composition: $(T\varepsilon, T\delta)$-DP	$\boxed{(12{,}000, .1)\text{-DP}}$

Advanced Composition Theorems

Composition theorem



$+\varepsilon$ for Blue

$+.2\varepsilon$ for Blue

$+\ \varepsilon$ for Red

“Heads, heads, heads”

Rosenkrantz: 78 in a row. A new record, I imagine.

Strong Composition Theorem

1. Choose $\sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon}$ $= 4$
2. Each step is (ε, δ)-DP (1.2, 10^{-5})-DP
3. Number of steps T 10,000
4. Strong comp: $(\varepsilon\sqrt{T \log 1/\delta}$, $T\delta)$-DP $\boxed{(360, .1)\text{-DP}}$

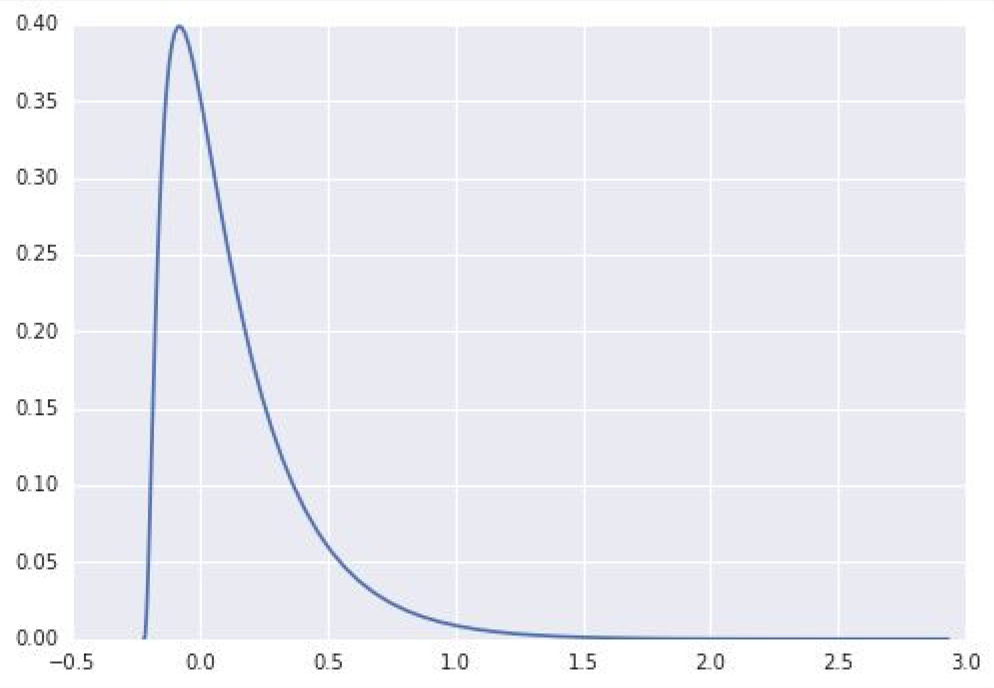
Dwork, Rothblum, Vadhan, "Boosting and Differential Privacy", FOCS 2010
Dwork, Rothblum, "Concentrated Differential Privacy", https://arxiv.org/abs/1603.0188

Amplification by Sampling

1. Choose $\sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon}$ $= 4$
2. Each batch is q fraction of data 1%
3. Each step is $(2q\varepsilon, q\delta)$-DP $(.024, 10^{-7})$-DP
4. Number of steps T 10,000
5. Strong comp: $(2q\varepsilon\sqrt{T \log 1/\delta}, qT\delta)$-DP $\boxed{(10, .001)\text{-DP}}$

S. Kasiviswanathan, H. Lee, K. Nissim, S. Raskhodnikova, A. Smith, "What Can We Learn Privately?", SIAM J. Comp, 2011

Privacy Loss Random Variable



log(privacy loss)

Moments Accountant

1. Choose $\sigma = \frac{\sqrt{2 \log 1/\delta}}{\varepsilon}$ $= 4$
2. Each batch is q fraction of data 1%
3. Keeping track of privacy loss's **moments**
4. Number of steps T 10,000
5. Moments: $(2q\varepsilon\sqrt{T}, \delta)$-DP $\boxed{(1.25, 10^{-5})\text{-DP}}$

Results

Summary of Results

	Baseline
	no privacy
MNIST	98.3%
CIFAR-10	80%

Summary of Results

	Baseline	[SS15]	[WKC+16]
	no privacy	reports ε per parameter	ε = 2
MNIST	98.3%	98%	80%
CIFAR-10	80%		

Summary of Results

	Baseline	[SS15]	[WKC+16]	this work		
	no privacy	reports ε per parameter	ε = 2	ε = 8 δ = 10^{-5}	ε = 2 δ = 10^{-5}	ε = 0.5 δ = 10^{-5}
MNIST	98.3%	98%	80%	97%	95%	90%
CIFAR-10	80%			73%	67%	

Contributions

- Differentially private deep learning applied to publicly available datasets and implemented in TensorFlow
  - https://github.com/tensorflow/models
- Innovations
  - Bounding sensitivity of updates
  - Moments accountant to keep tracking of privacy loss
- Lessons
  - Recommendations for selection of hyperparameters
- Full version: https://arxiv.org/abs/1607.00133