Updating ML Models

Ananth Mahadevan

November 8, 2020

Overview

Potential Applications

- **GDPR**: Deletion of private information from public datasets

Potential Applications

- **GDPR**: Deletion of private information from public datasets
- **Continuous Model Updating**: Handle additions, deletions and changes of training samples

Potential Applications

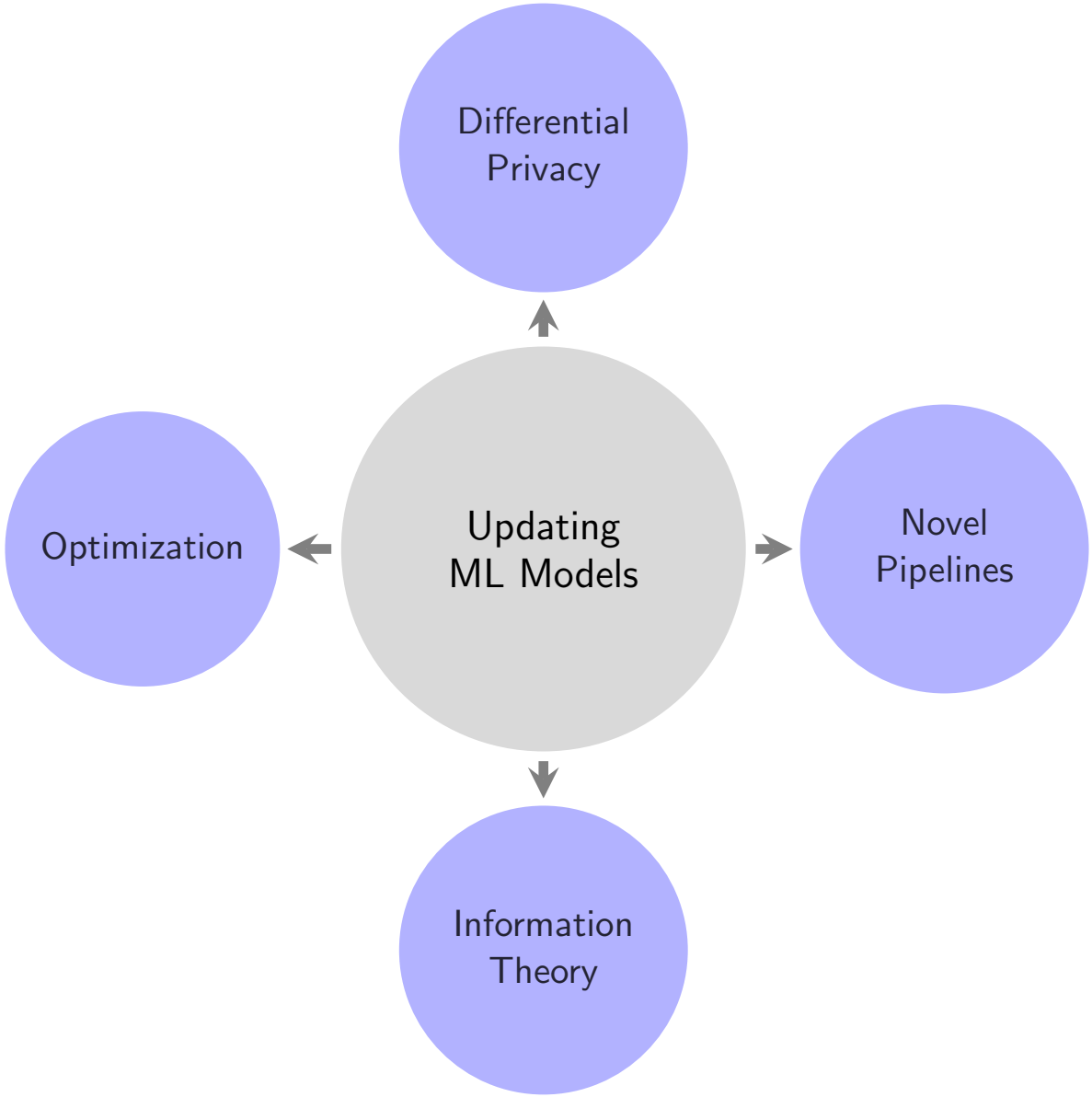
- **GDPR**: Deletion of private information from public datasets
- **Continuous Model Updating**: Handle additions, deletions and changes of training samples
- **Data Valuation**: *Leave One Out* tests to find important training samples

Potential Applications

- **GDPR**: Deletion of private information from public datasets
- **Continuous Model Updating**: Handle additions, deletions and changes of training samples
- **Data Valuation**: *Leave One Out* tests to find important training samples
- **Bias Reduction**: Jackknife resampling that requires retrained model parameters

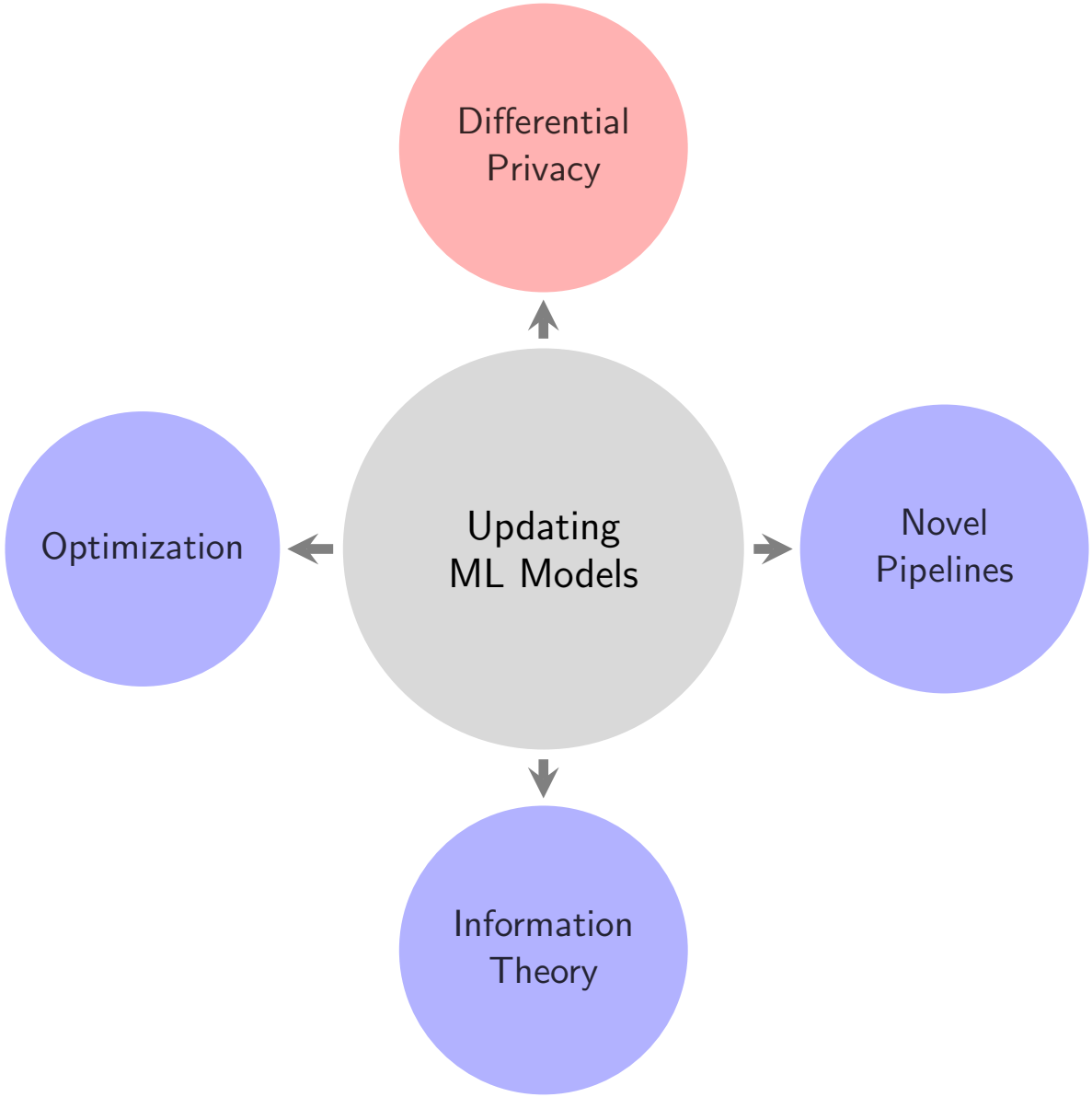
Challenges [Bourtoule et al., 2020]

1. Limited understanding of impact of each data point on the model
2. Stochasticity in training
3. Incremental training
4. Stochasticity in learning


Differential Privacy
Optimization
Updating ML Models
Novel Pipelines
Information Theory

Common Terminology

- Fixed training Dataset $\mathcal{D}$
- Learning Algorithm A (can be randomized)
- Datapoints to be remove $\mathcal{D}_{\mathcal{R}}$, where $|\mathcal{D}_{\mathcal{R}}| = r$, remaining dataset $\mathcal{D}' = \mathcal{D} - \mathcal{D}_{\mathcal{R}}$
- Naive approach is retraining from scratch, i.e, $A(\mathcal{D}')$
- Mechanism M which offers an efficient way to update the model


Differential Privacy
Optimization
Updating ML Models
Novel Pipelines
Information Theory

Certified Data Removal [Guo et al., 2020]

- A outputs a model in hypothesis space $\mathcal{H}$
- Defines ϵ-*certified removal*, $\forall \mathcal{T} \subseteq \mathcal{H}$

$$e^{-\epsilon} \leq \frac{P(M(A(\mathcal{D}), \mathcal{D}_{\mathcal{R}}) \in \mathcal{T})}{P(A(\mathcal{D}') \in \mathcal{T})} \leq e^{\epsilon}$$

- Insufficiency of Parametric indistinguishability
  - Approximate removal processes leaves a gradient residual
  - Residuals can reveal the prior presence of that training sample

Removal Mechanism for Linear Classfiers

- *A* empirical risk $L(\mathbf{w}; \mathcal{D})$ with a convex loss function $\ell(\mathbf{w}^T\mathbf{x}, y)$
- $\mathbf{w}^* = A(\mathcal{D}) = \text{argmin}_w\, L(\mathbf{w}; \mathcal{D})$
- To remove a single point $\mathcal{D}_\mathcal{R} = \{(\mathbf{x}_n, y_n)\}$
- Newton Update Step: $\mathbf{w}^- = M(\mathbf{w}^*, (\mathbf{x}_n, y_n)) = \mathbf{w}^* - H_{\mathbf{w}^*}^{-1}\nabla$
- Where $H_{\mathbf{w}^*} = \nabla^2 L(\mathbf{w}^*, \mathcal{D}')$ and $\nabla = \lambda\mathbf{w}^* + \nabla\ell((\mathbf{w}^*)^T\mathbf{x}_n, y_n)$
- $H_{\mathbf{w}^*}^{-1}\nabla$ is from *influence function* literature

Influence Function

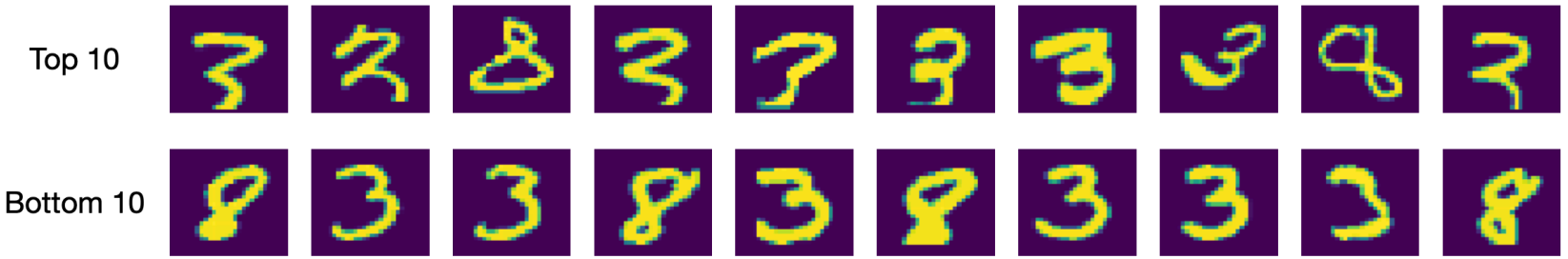


Figure 3. **MNIST training digits sorted by norm of the removal update $\|\mathbf{H}_{\mathbf{w}_*}^{-1}\boldsymbol{\Delta}\|_2$.** The samples with the highest norm (**top**) appear to be atypical, making it harder to undo their effect on the model. The samples with the lowest norm (**bottom**) are prototypical 3s and 8s, and hence are much easier to remove.

Certifing Removal

- $\mathbf{w}^-$ is approximate close to minimizer of $L(\mathbf{w}; \mathcal{D}')$
- $\nabla L(\mathbf{w}^-; \mathcal{D}')$ is gradient residual and if non-zero, reveals Information
- Even a small $\|\nabla L(\mathbf{w}^-; \mathcal{D}')\|_2$ doesn't guarantee certifiable removal
- Therefore, perturb loss at training time

$$L_b(\mathbf{w}; \mathcal{D}) = \sum_{i=1}^{n} \ell(\mathbf{w}^T \mathbf{x}_i, y_i) + \frac{\lambda n}{2} \|\mathbf{w}\|_2^2 + \mathbf{b}^T \mathbf{w}$$

Where $\mathbf{b} \in \mathbb{R}^d$ drawn randomly from some distribution

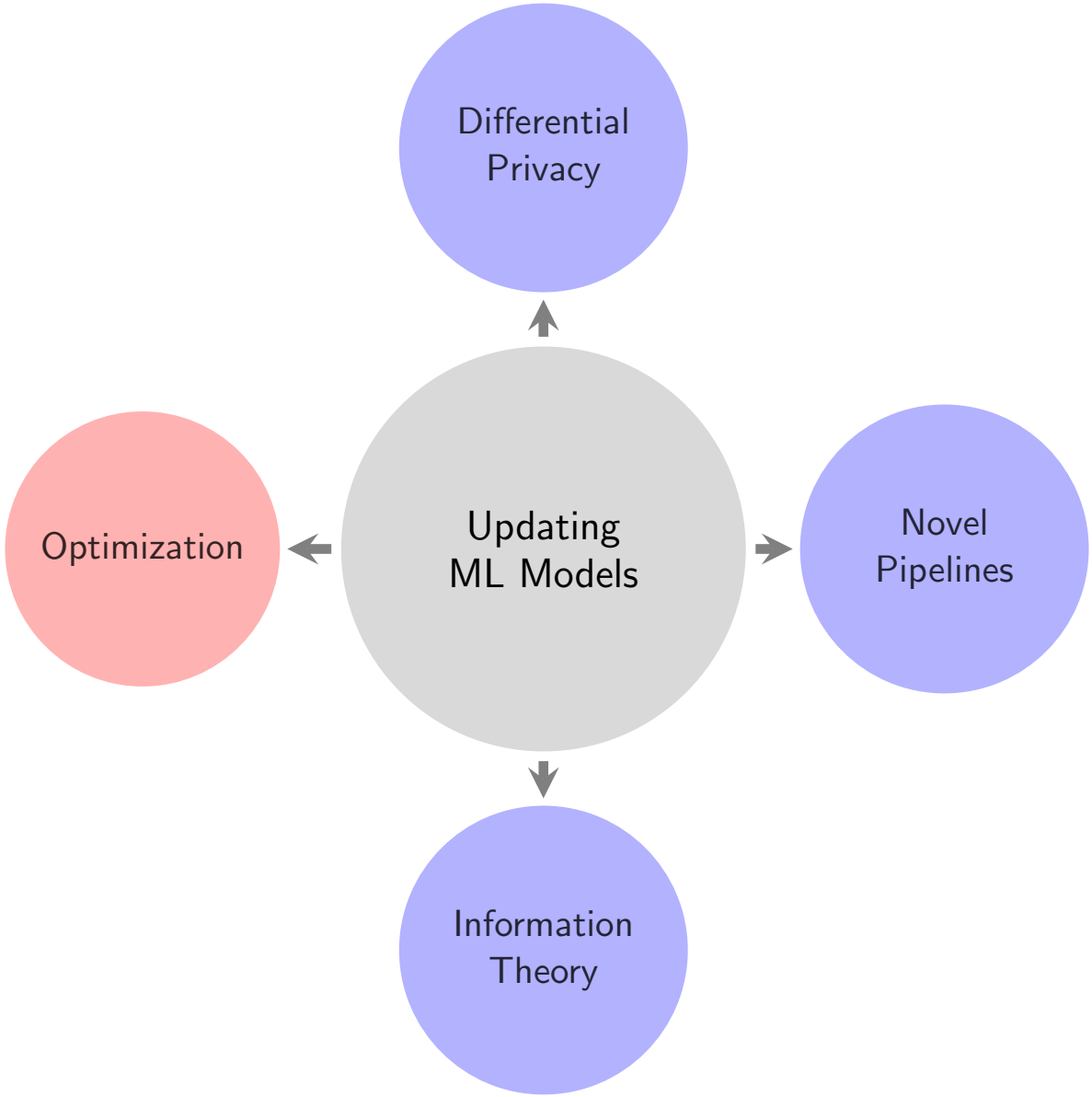
Benefits and Drawbacks

Benefits

- Provides formal guarantee of statistical indistinguishability
- Works well with Differentially Private trained networks
- Uses influence functions to approximate data removal

Limitations

- Requires inverting a Hessian matrix
- Non-convex loss functions not supported
- Adding noise during training hurts model performance
- Very strict notion of removal


Differential Privacy
Optimization
Updating ML Models
Novel Pipelines
Information Theory

DeltaGrad [Wu et al., 2020]

- M targets the Gradient Descent (GD) algorithm
- Naive retraining $A(\mathcal{D}')$ recomputed gradients over all remaining points

$$\mathbf{w}_{t+1}^U \leftarrow \mathbf{w}_t^U - \frac{\eta_t}{n-r} \sum_{i \in \mathcal{D}'} \nabla L_i(\mathbf{w}_t^U)$$

- Instead rewrite it as a *leave-r-out* formula

$$\mathbf{w}_{t+1}^I = \mathbf{w}_t^I - \frac{\eta_t}{n-r} \left[\sum_{i \in \mathcal{D}} \nabla L_i(\mathbf{w}_t^I) - \sum_{i \in \mathcal{D}_\mathcal{R}} \nabla L_i(\mathbf{w}_t^I) \right]$$

- Much cheaper to compute r gradients, when $r \ll n$

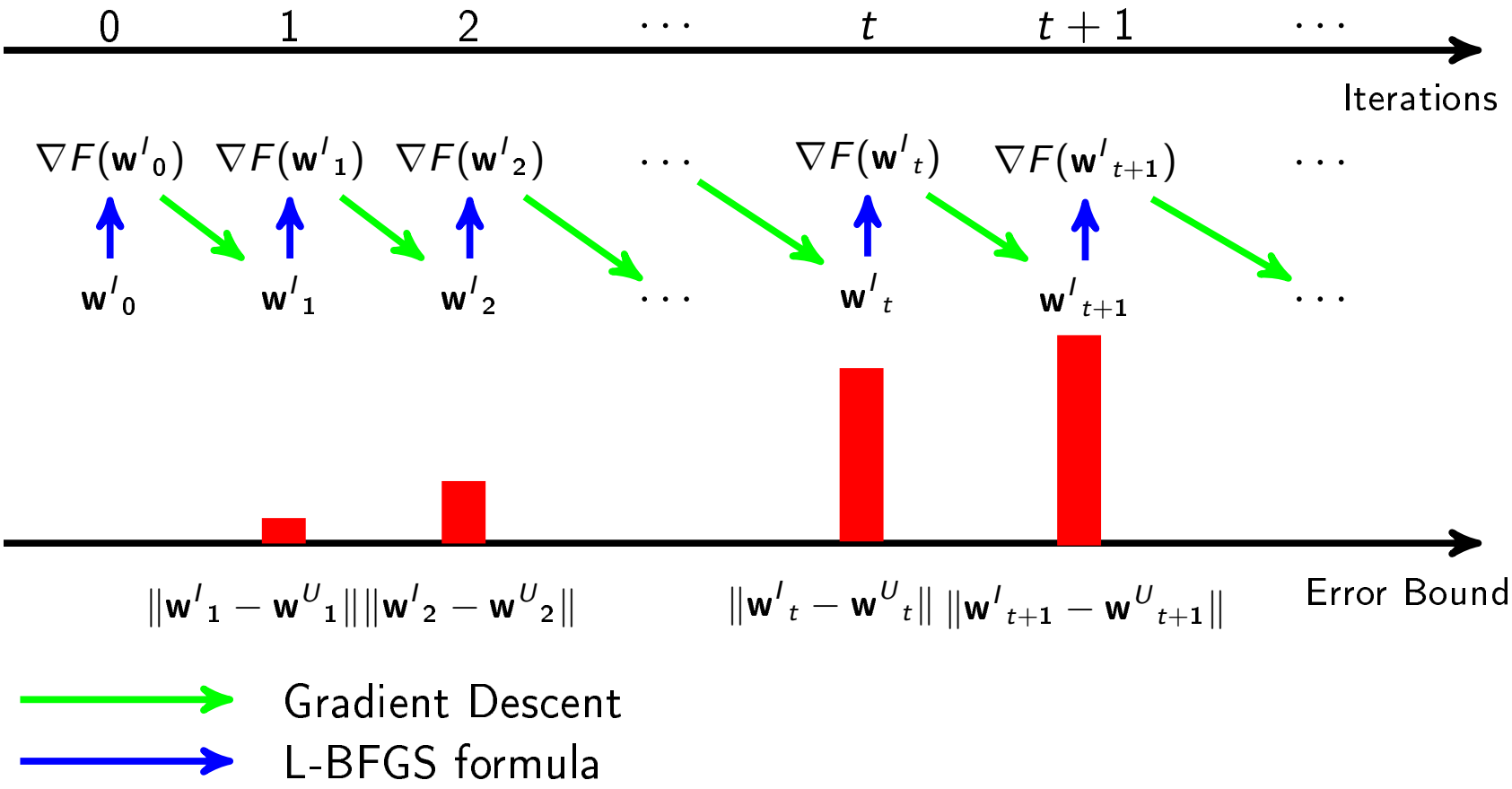
Approximating $\sum_{i \in \mathcal{D}} \nabla L_i(\mathbf{w}_t^I)$

- Need to use historical $\nabla L(\mathbf{w}_t)$ to approximate $\nabla L(\mathbf{w}_t^I)$
- Taylor expansion around $\mathbf{w}_t^I$ gives the following

$$\nabla L(\mathbf{w}_t^I) = \nabla L(\mathbf{w}_t) + \mathbf{H}_t \cdot (\mathbf{w}_t^I - \mathbf{w}_t)$$

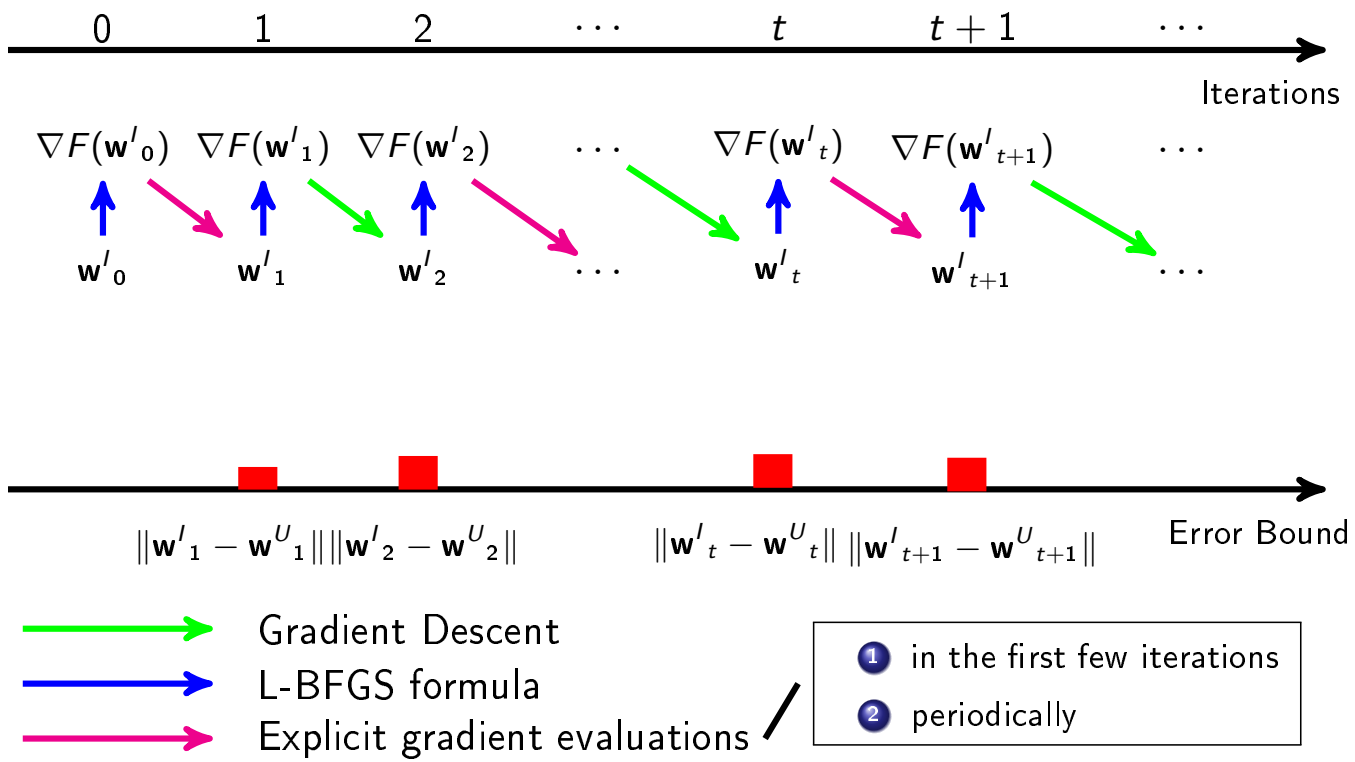
  Where $\mathbf{H}_t = \int_0^1 \mathbf{H}(\mathbf{w}_t + x(\mathbf{w}_t^I - \mathbf{w}))dx$
- Maintaining a Hessian matrix is expensive, so leverage the L-BFGS algorithm to compute a Hessian-vector product
- This leads to issues in error bounds of the approximation

Problem with Error Bound



Controlling the Errors

- Do explicit evaluations for j_0 "burn-in" iterations and then periodically every T_0 iterations



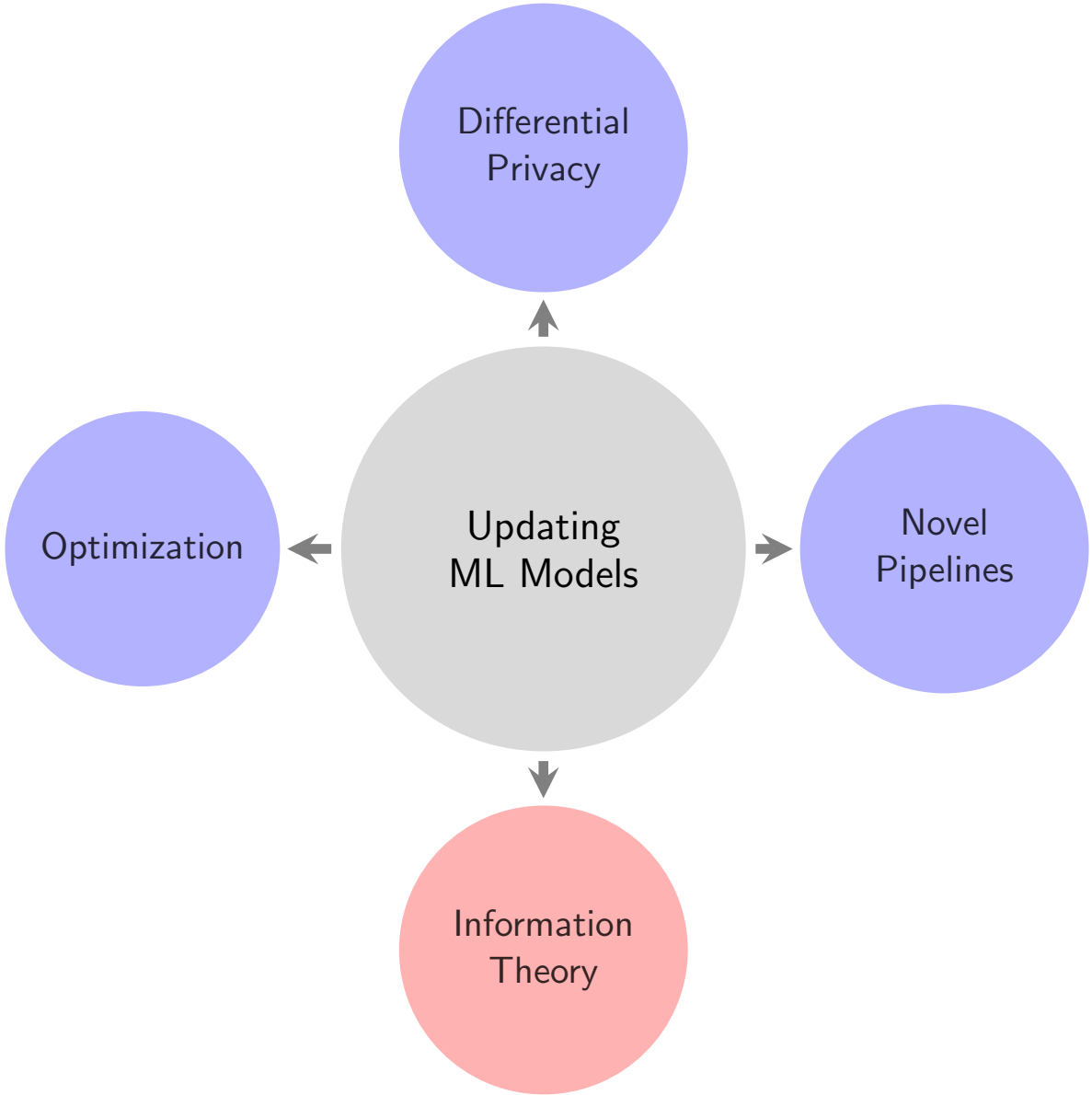
Benefits and Limitations

Benefits

- Handles both additions nad deletions of datapoints
- Can be applied to any ML model trained using Stochastic Gradient Descent
- Approximation guarantees and empirical results on

Limitaitons

- Needs to cache all weights $\mathbf{w}_t$ and gradients $\nabla L(\mathbf{w}_t)$ during training
- Requires tuning of T_0 and j_0 based on dataset
- For SGD, only works with large batch sizes (> 10000), which hurts model performance


Differential Privacy
Optimization
Updating ML Models
Novel Pipelines
Information Theory

Eternal Sunshine of the Spotless Net [Golatkar et al., 2020]

- $P(\mathbf{w}|\mathcal{D})$ distribution of algorithm A
- M is called *scrubbing function* applied to $\mathbf{w}$
- $P(M(\mathbf{w})|\mathcal{D})$ is distribution of possible weights after scrubbing
- Motivation: disallow attacker to use *read-out function* $f(\mathbf{w})$ to gain information about $\mathcal{D}_{\mathcal{R}}$
- Therefore, optimal scrubbing function must have

$$\mathrm{KL}(P(f(M(\mathbf{w}))|\mathcal{D}) \parallel P(f(S_0(\mathbf{w}))|\mathcal{D}')) = 0$$

  Where S_0 is a *certificate* of forgetting
- To be agnostic of $f(\cdot)$, minimize

$$\mathrm{KL}(P(M(\mathbf{w})|\mathcal{D}) \parallel P(S_0(\mathbf{w})|\mathcal{D}'))$$

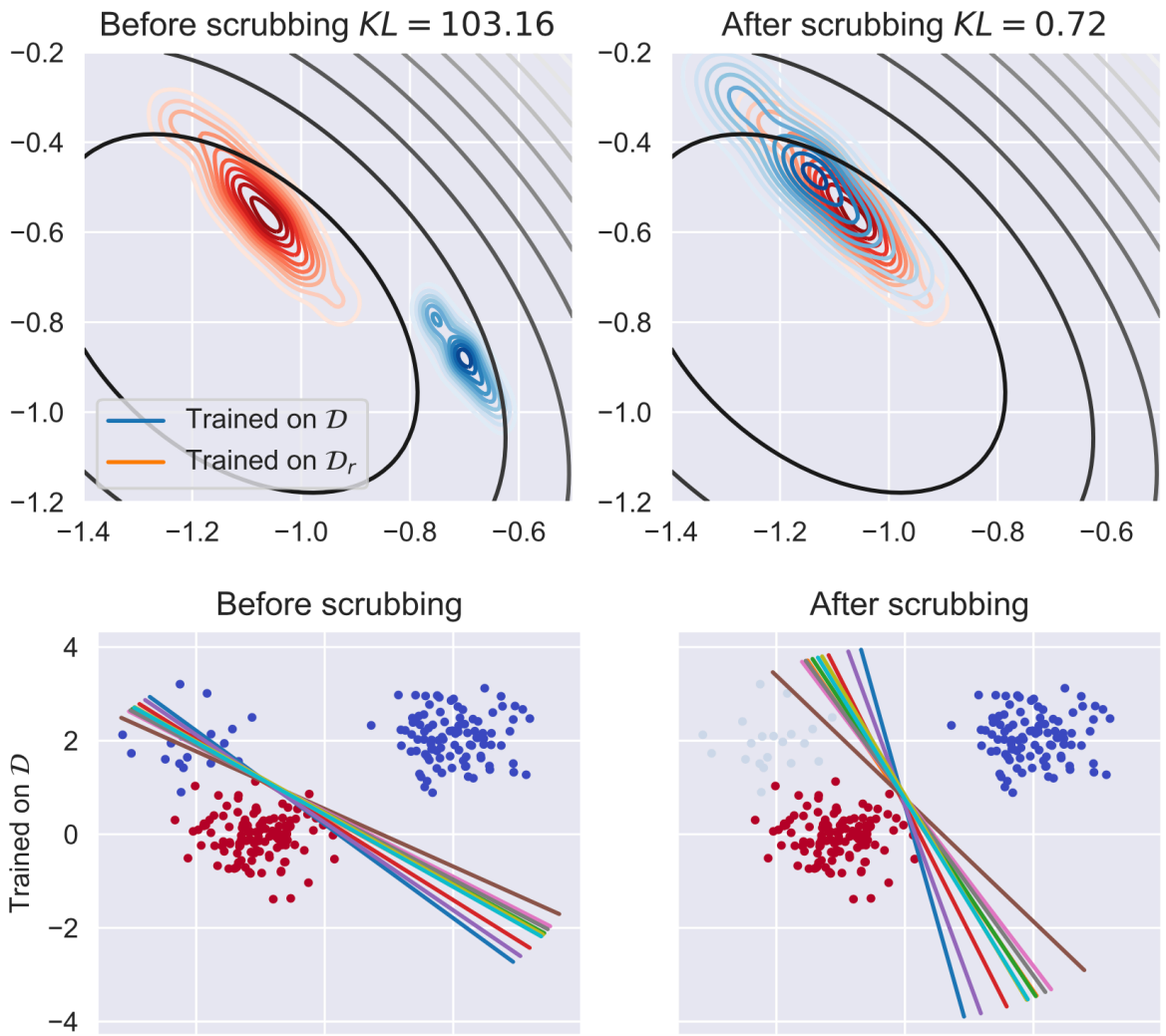
Forgetting Lagrangian

- A trivial noise scrubbing is $M(\mathbf{w}) = S_0(\mathbf{w}) = w + \sigma n$, where $n \sim \mathcal{N}(0, I)$
- As $\sigma \to \infty$, $\mathrm{KL}(p\|q) \to 0$, which invalidates the model
- Define the *Forgetting Lagrangian*:

$$\mathcal{L} = \mathbb{E}_{M(\mathbf{w})}[L_{\mathcal{D}'}(\mathbf{w})] + \lambda \mathrm{KL}(P(M(\mathbf{w})|\mathcal{D}) \parallel P(S_0(\mathbf{w})|\mathcal{D}'))$$

- Use quadratic approximation and noise to scrub weights
- $M(\mathbf{w}) = h(\mathbf{w}) + n$ and $S_0 = w + n'$ where $h(\mathbf{w})$ is deterministic and $n, n' \sim \mathcal{N}(, \Sigma)$

Scrubbing Example



Robust Quadratic Scrubbing

- Noisy Newton update, as $t \to \infty$ is defined as

$$M_t(\mathbf{w}) = \mathbf{w} - \mathbf{H}^{-1}\nabla L_{\mathcal{D}'}(\mathbf{w}) + (\lambda\sigma_h^2)^{1/4}\mathbf{H}^{-1/4}$$

  Where $\mathbf{H} = \nabla^2(L_{\mathcal{D}'}(\mathbf{w}))$, σ_h represents error in approximating SGD with a continuous gradient flow and λ hyperparameter
- For Deep Neural Networks, Hessian matrix is expensive to compute and store
- Simplified scrubbing to only adding noise

$$M(\mathbf{w}) = \mathbf{w} + (\lambda\sigma_h^2)^{1/4}F^{-1/4}$$

- F is the Fisher Information Matrix, computed using the Levenberg- Marquardt semi-positive-definite approximation of $\nabla^2 L_{\mathcal{D}}(\mathbf{w})$

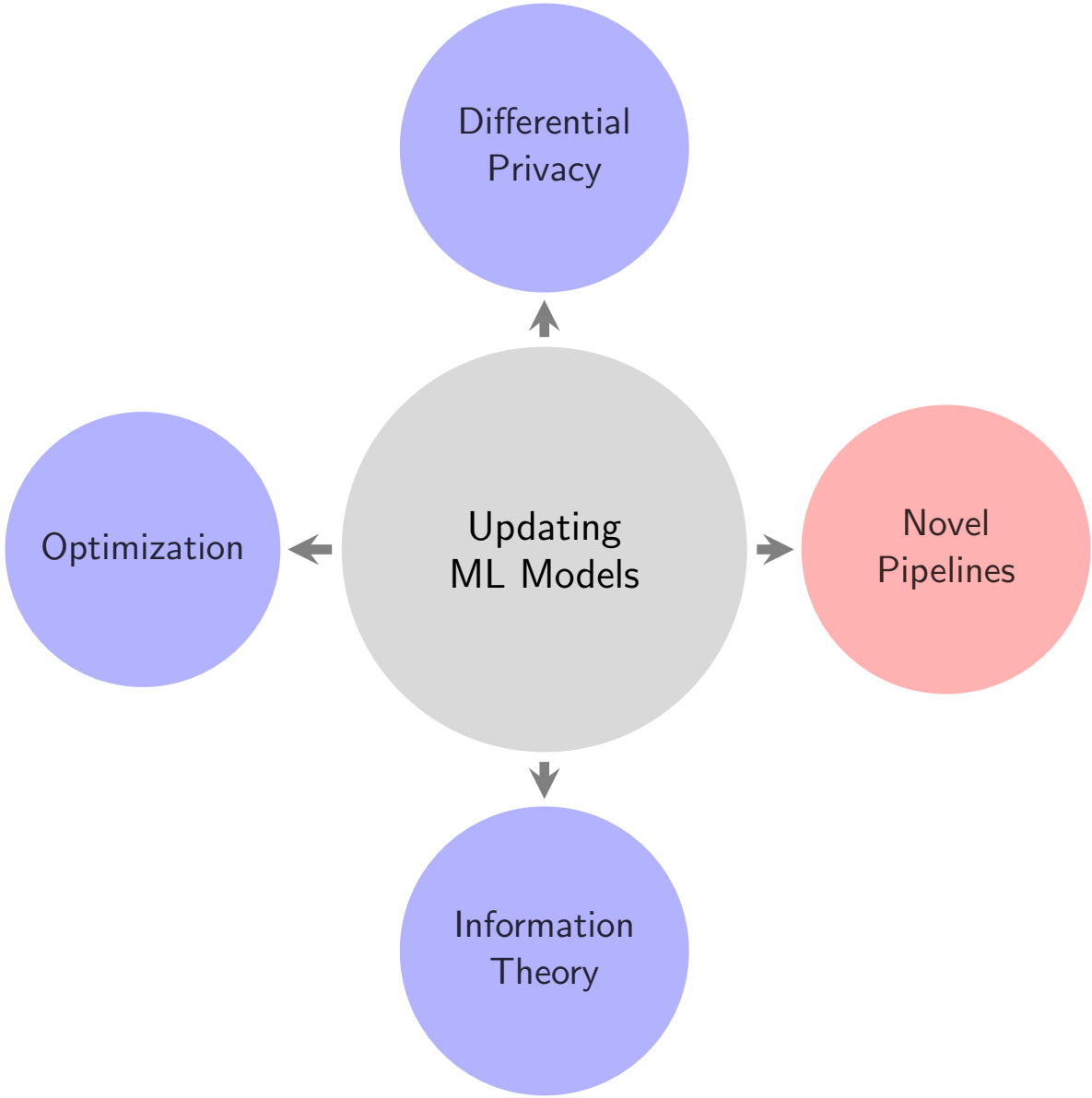
Benefits and Limitations

Benefits

- Works for Deep Neural Networks
- Allows to remove a entire class, multiple classes, or a subset of a class of the training dataset
- Process is optimal if quadratic assumptions hold true

Limitations

- Space and time complexity of approach unknown
- Considers worst case of attacker using any read-out function $f(\cdot)$
- Results based on stability of SGD after pre-training networks

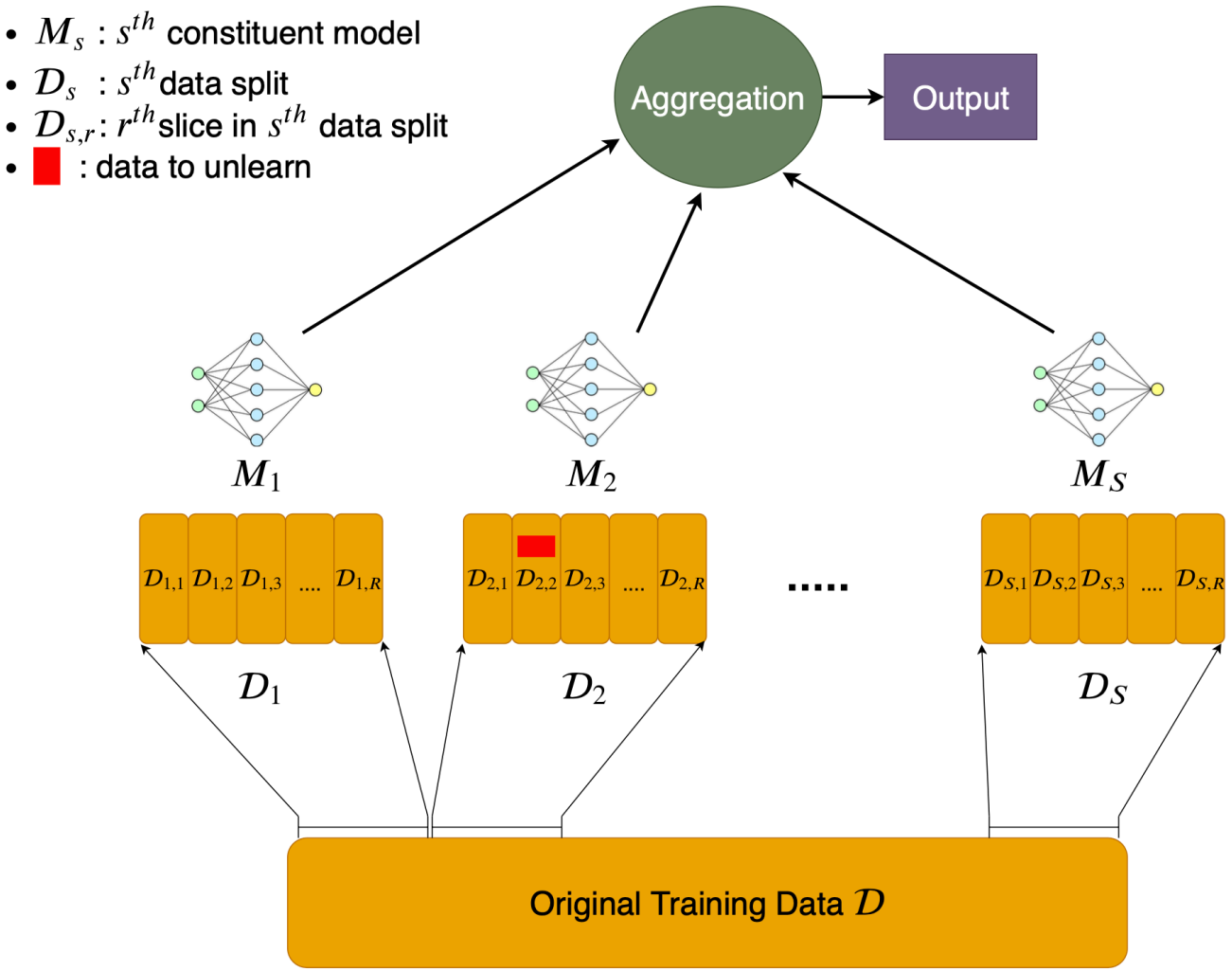

Differential Privacy
Optimization
Updating ML Models
Novel Pipelines
Information Theory

Machine Unlearning: SISA [Bourtoule et al., 2020]

SISA Framework

- **Sharded**:
  Dataset is divided into S disjoint shards, $\cap_{k \in [S]} \mathcal{D}_k = \emptyset$ and $\cup_{k \in [S]} \mathcal{D}_k = \mathcal{D}$
- **Isolated**:
  ML models are trained on each data shard $\mathcal{D}_k$ in isolation to other data shards
- **Sliced**:
  Further, each shard $\mathcal{D}_k$ is sliced into R disjoint slices.
  Incrementally present shards during training
- **Aggregated**:
  During inference aggregate predictions from individual models

SISA Framework



Benefits and Drawbacks

Benefits

- Service providers have *plausible removal* of data
- Practical approach reduces the overhead of unlearning
- Can utilize distributional of remove requests for sharding

Drawbacks

- Training of isolated shards might lead to weak learners for especially complex tasks
- Sharding and slicing increase the overall space complexity of the approach
- Distribution aware sharding might lead to biased models

References I


Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. (2020).
Machine Unlearning.
arXiv:1912.03817 [cs].


Golatkar, A., Achille, A., and Soatto, S. (2020).
Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks.
arXiv:1911.04933 [cs, stat].

References II


Guo, C., Goldstein, T., Hannun, A., and van der Maaten, L. (2020).
Certified Data Removal from Machine Learning Models.
arXiv:1911.03030 [cs, stat].


Wu, Y., Dobriban, E., and Davidson, S. B. (2020).
DeltaGrad: Rapid retraining of machine learning models.
arXiv:2006.14755 [cs, stat].