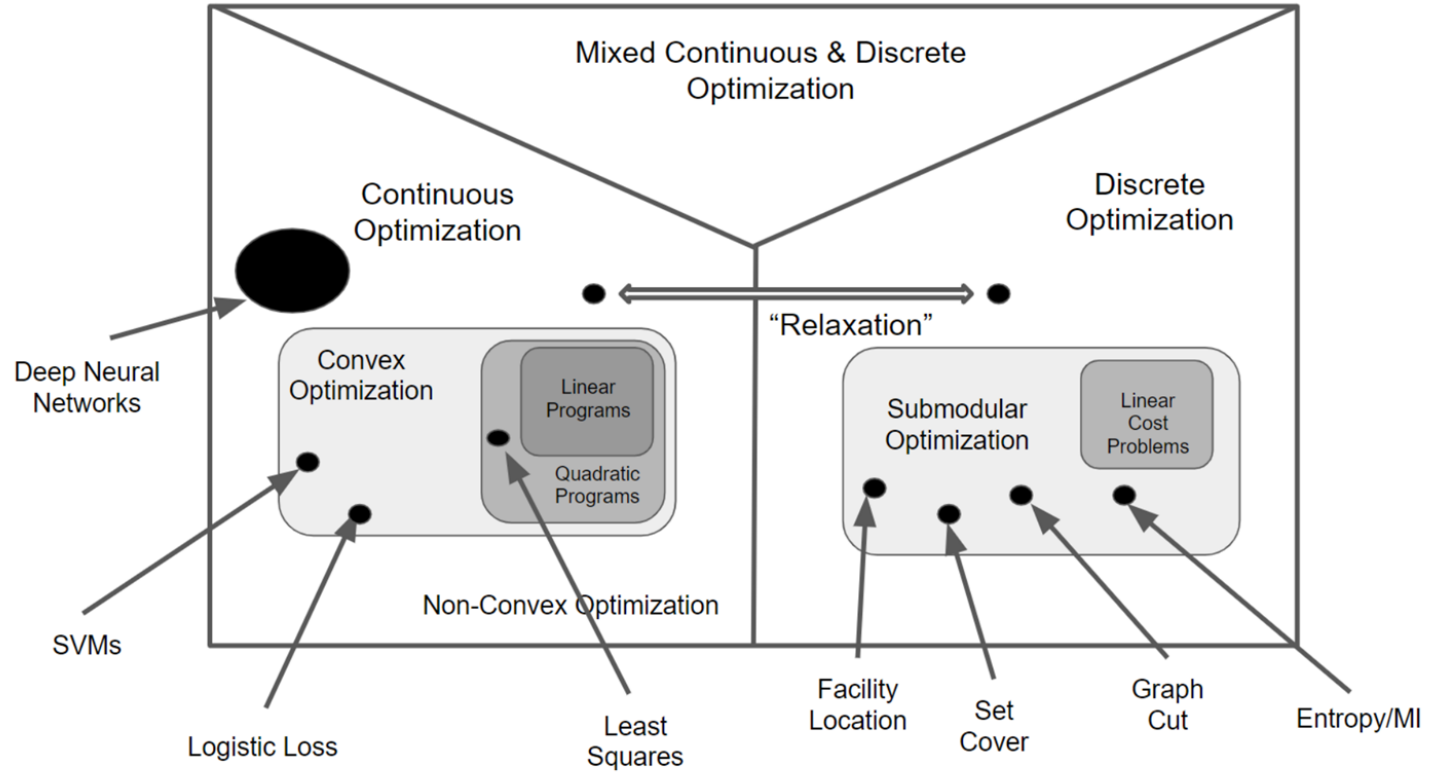
Applications of Continuous and Discrete Optimization

Lecture 11

Advanced Topics in Optimization For Machine Learning (CS 7301)

Instructor: Rishabh Iyer

Big Picture: Continuous and Discrete Optimization



Applications we will cover today

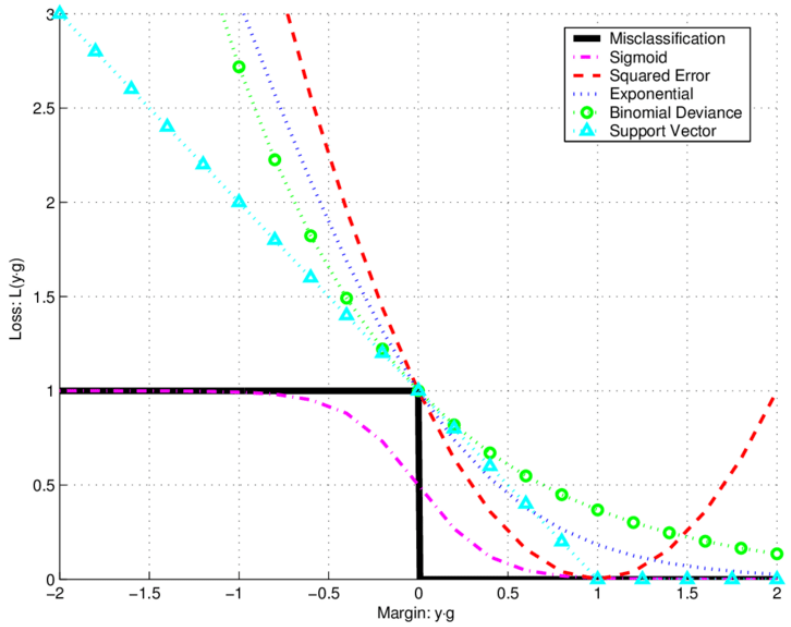
- **Continuous Optimization**
  - General ML Loss
  - Semi-supervised learning
  - Meta-learning
  - Min-Max Optimization: Generative Adversarial Networks
  - Robust and Adverserial Optimization
  - Bi-level optimization
- Discrete Optimization
  - Data Subset Selection
  - Active Learning
  - Feature Selection
  - Data summarization
  - Model Compression

Loss Functions in ML: Setup

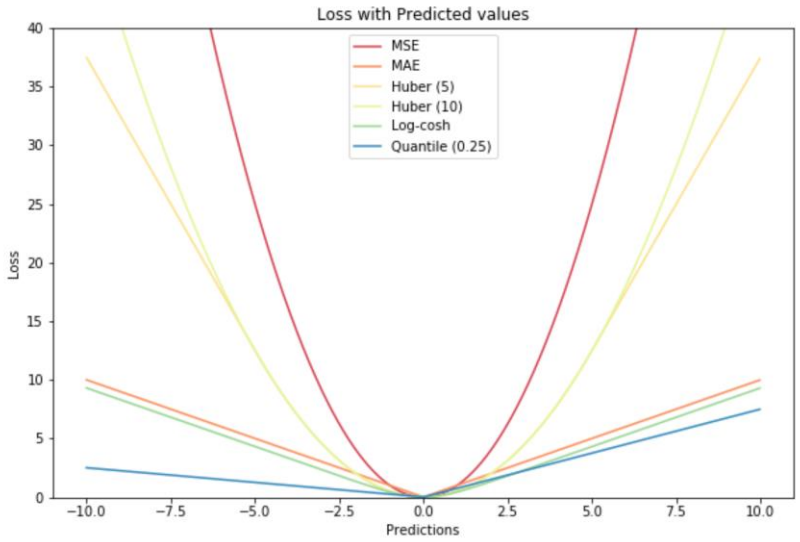
Supervised Learning

Loss Functions in ML: Examples

- **Classification**
  - Log Loss
  - Focal Loss
  - KL Divergence/ Relative Entropy
  - Exponential Loss
  - Hinge Loss
- **Regression**
  - Mean Square Error/ Quadratic Loss
  - Mean Absolute Error
  - Huber Loss/ Smooth Mean Absolute Error
  - Log cosh Loss
  - Quantile Loss



Classification



Regression

Source: https://phuctrt.medium.com/loss-functions-why-what-where-or-when-189815343d3f

Unsupervised Learning: Clustering

Our goal is to find the *best* or *optimal* clustering (i.e. the one with the lowest value of F). We call the optimal clustering C_{opt} , and the lowest/best value of the objective function F_{opt}.

$$\mathcal{C}_{\text{opt}} = \arg\min_{\mathcal{C}_i} F(\mathcal{C}_i)$$

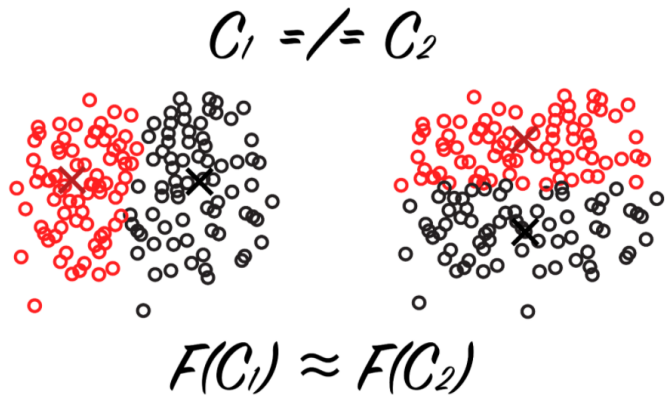
$$F_{\text{opt}} = \min_{\mathcal{C}_i} F(\mathcal{C}_i)$$

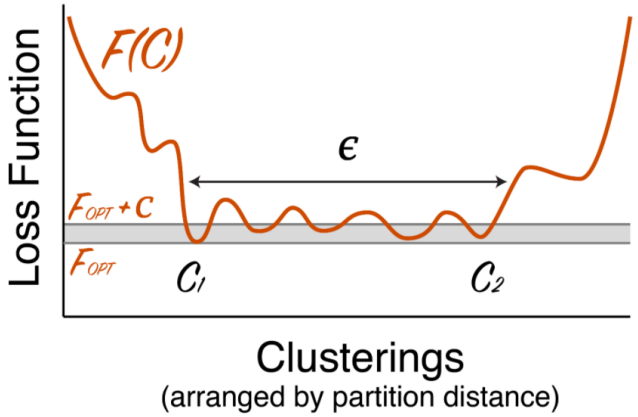
$$F(\mathcal{C}) = \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in \mathcal{K}_i} \left\| \bar{\mathbf{x}}_i - \mathbf{x}_j \right\|_2^2$$

k-means clustering

$$F(\mathcal{C}) = \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in \mathcal{K}_i} \left| \tilde{\mathbf{x}}_i - \mathbf{x}_j \right|$$

k-medians clustering

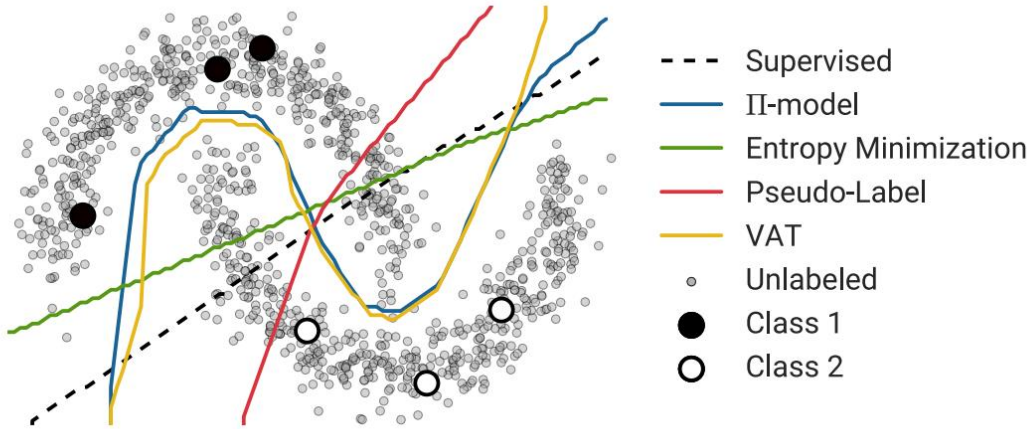




Source: http://alexhwilliams.info/itsneuronalblog/2015/11/18/clustering-is-easy/

Semi-Supervised Learning

SSL Loss Function: $$\sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} l(f(\mathbf{x}_i, \theta), y_i) + \sum_{x_j \in \mathcal{U}} r(f(\mathbf{x}_j, \theta))$$



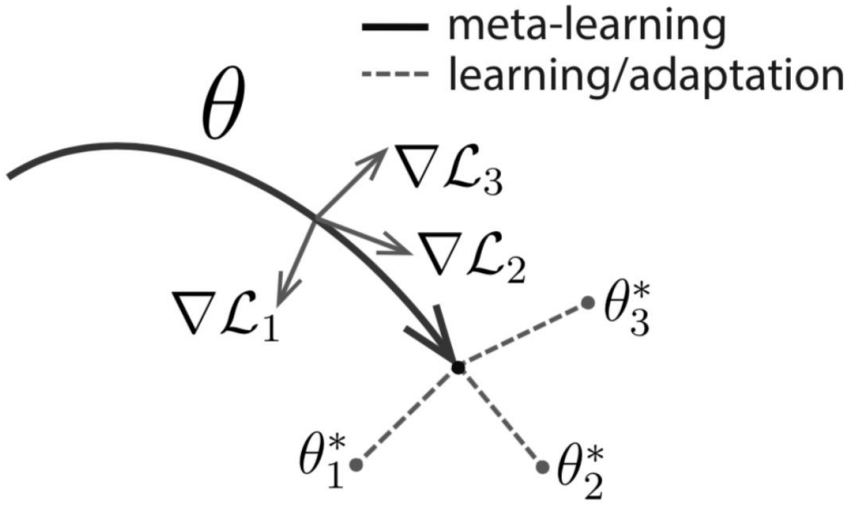
Example SSL Algorithms

- Entropy Min
- Pseudo Label
- Consistency Regularization
- PI Model
- Mean Teacher
- Virtual Adversarial Training
- Mix-Match
- Fix-Match

Additional Reading:

1) https://akyrillidis.github.io/2020/03/25/ssl.html, 2) https://papers.nips.cc/paper/2018/file/c1fea270c48e8079d8ddf7d06d26ab52-Paper.pdf

Meta-Learning (e.g., Model Agnostic Meta Learning)



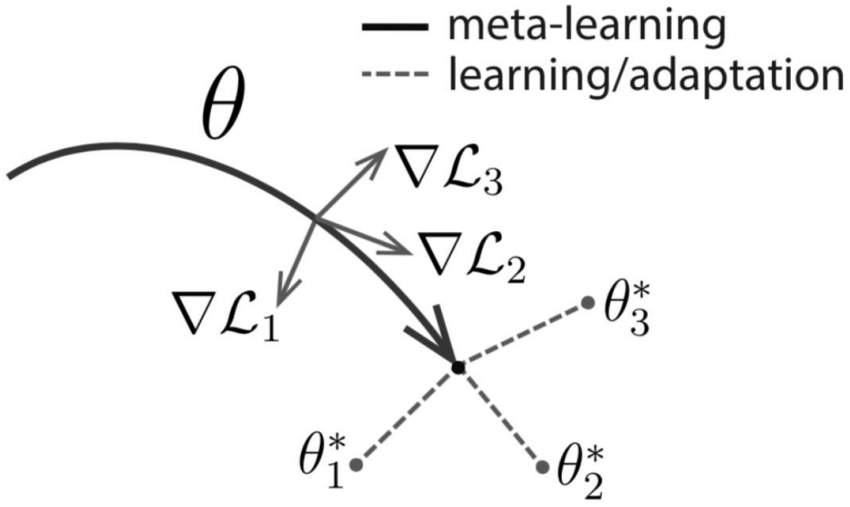
$$\theta^* = \arg\min_\theta \sum_{\tau_i \sim p(\tau)} \mathcal{L}_{\tau_i}^{(1)}(f_{\theta_i'}) = \arg\min_\theta \sum_{\tau_i \sim p(\tau)} \mathcal{L}_{\tau_i}^{(1)}(f_{\theta - \alpha\nabla_\theta \mathcal{L}_{\tau_i}^{(0)}(f_\theta)})$$

$$\theta \leftarrow \theta - \beta\nabla_\theta \sum_{\tau_i \sim p(\tau)} \mathcal{L}_{\tau_i}^{(1)}(f_{\theta - \alpha\nabla_\theta \mathcal{L}_{\tau_i}^{(0)}(f_\theta)})$$

Here $\theta_i' = \theta - \alpha\nabla_\theta \mathcal{L}_{\tau_i}^{(0)}(f_\theta)$

Paper: https://arxiv.org/abs/1703.03400

Meta-Learning (e.g., Model Agnostic Meta Learning)



Paper: https://arxiv.org/abs/1703.03400

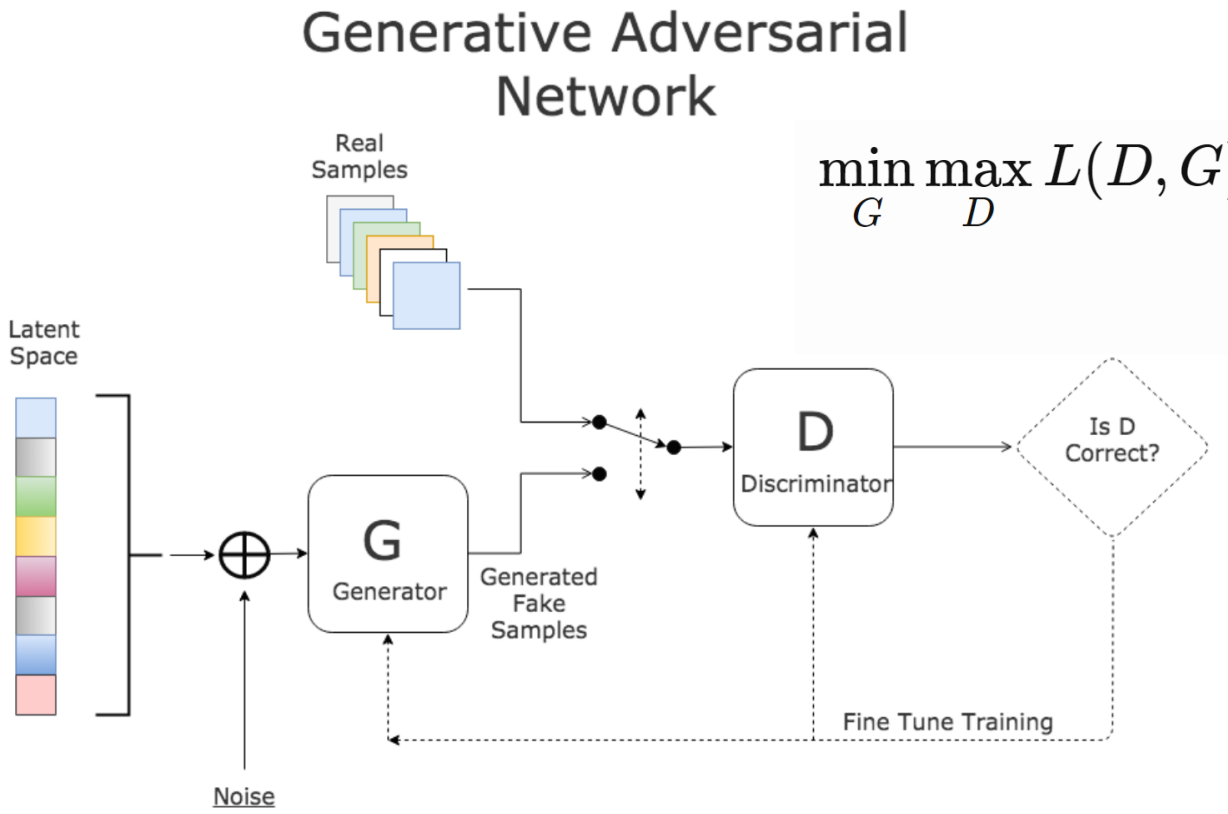
Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks

Require: α, β: step size hyperparameters

1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for** Note: the meta-update is using different set of data.
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Min-Max Optimization (e.g. GANs)

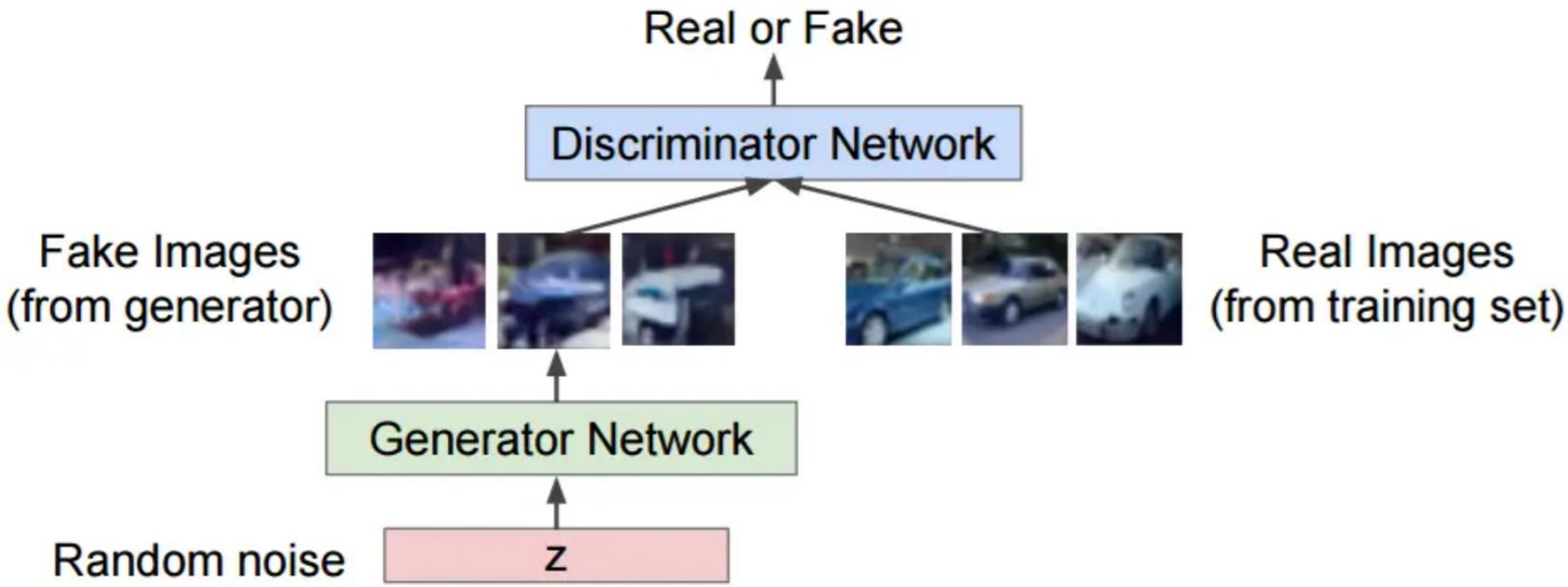


$$\min_G \max_D L(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$
$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1-D(x)]$$

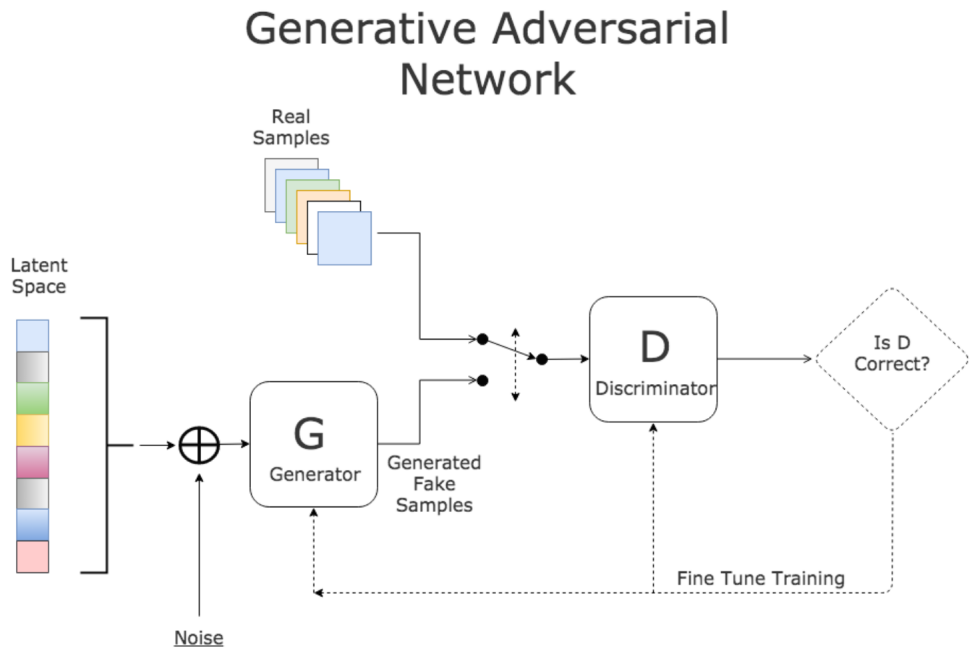
Furthermore, $L(G,D^*) = 2D_{JS}(p_r \| p_g) - 2\log 2$

https://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html

Min-Max Optimization (e.g. GANs)



Min-Max Optimization (e.g. GANs)



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

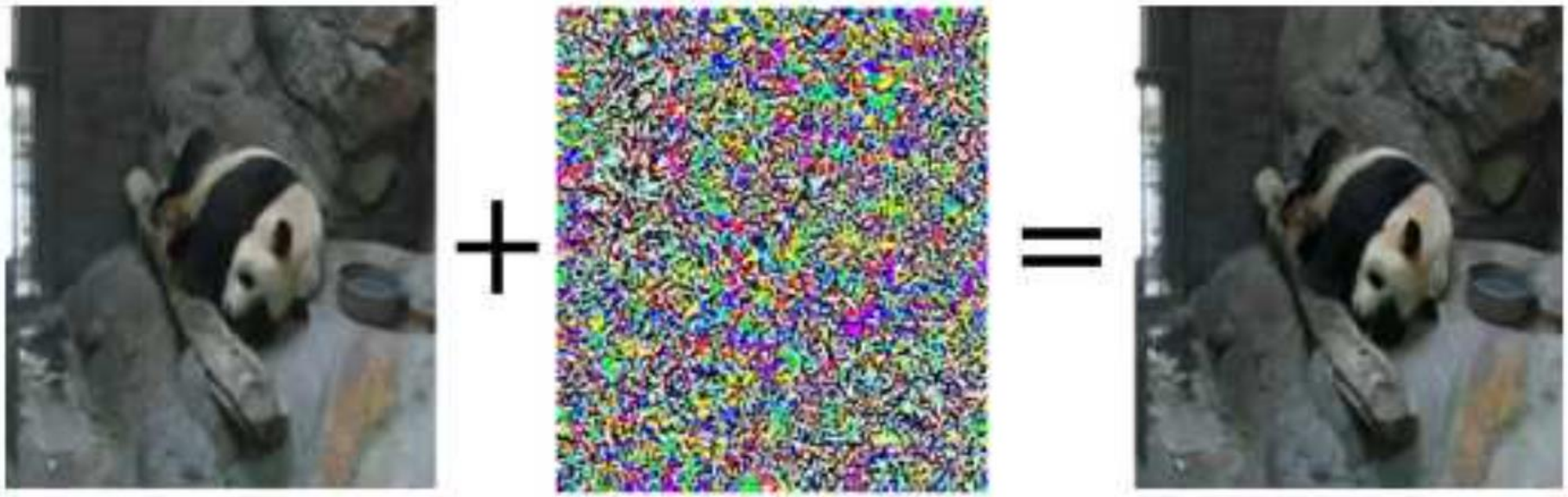
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Original GAN paper: https://arxiv.org/abs/1406.2661

Min-Max (Adverserial) Robustness



Min-Max (Adverserial) Robustness

$$\underset{\theta}{\text{minimize}}\, \hat{R}_{\text{adv}}(h_\theta, D_{\text{train}}) \equiv \underset{\theta}{\text{minimize}}\, \frac{1}{|D_{\text{train}}|} \sum_{(x,y)\in D_{\text{train}}} \max_{\delta\in\Delta(x)} \ell(h_\theta(x+\delta)), y).$$

$$\theta := \theta - \frac{\alpha}{|B|} \sum_{(x,y)\in B} \nabla_\theta \max_{\delta\in\Delta(x)} \ell(h_\theta(x+\delta)), y)$$

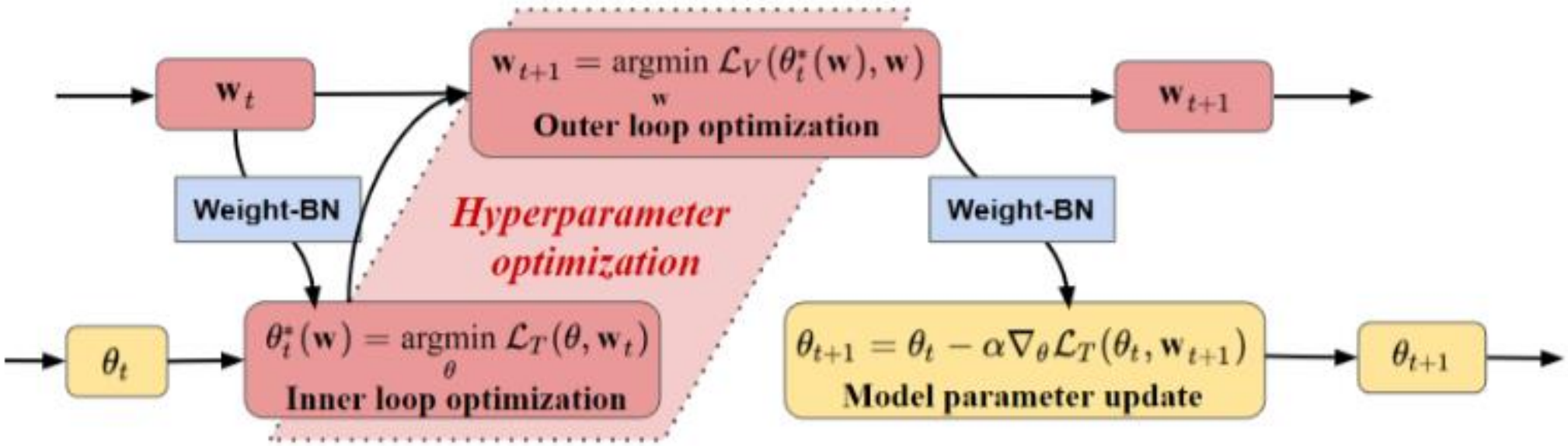
the gradient we require is simply given by

$$\nabla_\theta \max_{\delta\in\Delta(x)} \ell(h_\theta(x+\delta)), y) = \nabla_\theta \ell(h_\theta(x+\delta^\star)), y)$$

Where: $\delta^\star = \underset{\delta\in\Delta(x)}{\text{argmax}}\, \ell(h_\theta(x+\delta)), y)$

Bi-Level Optimization: General Formulation

Bi-Level Optimization



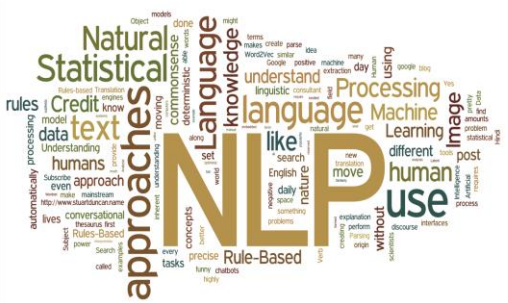
Applications we will cover today

- Continuous Optimization
  - General ML Loss
  - Semi-supervised learning
  - Meta-learning
  - Min-Max Optimization: Generative Adversarial Networks
  - Robust and Adverserial Optimization
  - Bi-level optimization
- **Discrete Optimization**
  - Data Subset Selection
  - Active Learning
  - Feature Selection
  - Data summarization
  - Model Compression

Staggering Costs/Energy/Time for Deep Learning

Example: Training a BERT model on Wikipedia and Book Corpora Dataset[1]

Cost - $200k (340 million parameter model)

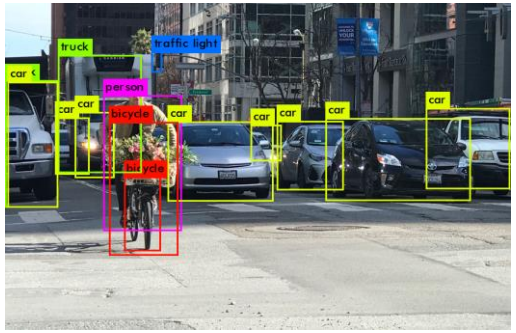


Example: Training a single Deep model for NLP (with NAS)[2]

Cost - $3.2M

Example: Training an ELMO model on a P-100 GPU [2]

Time - 14 Days



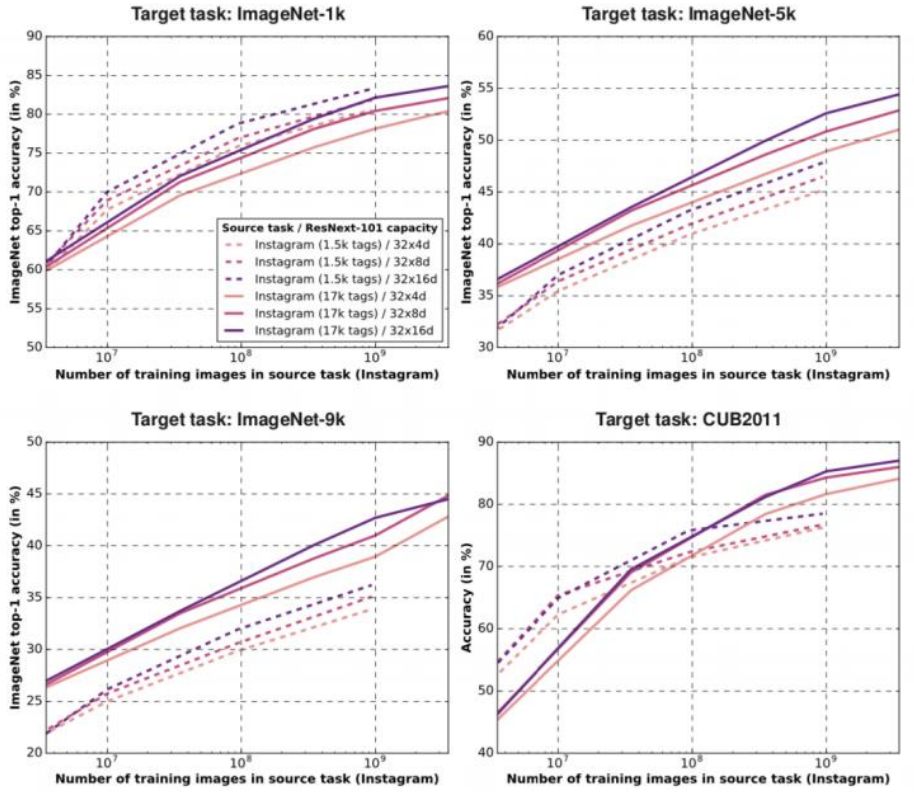
Example: Training a YOLO V5x model on a V100 GPU[3]

Time - 8 Days

1. https://arxiv.org/pdf/2004.08900.pdf 2. https://arxiv.org/pdf/1906.02243.pdf 3. https://github.com/ultralytics/yolov5

Energy Consumption and Costs

Consumption	CO_2e (lbs)
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000
Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155



STRUBEL ET AL 2019, Energy and Policy Considerations for Deep Learning in NLP

SCHARTZ ET AL 2020, GREEN AI

Energy Consumption and Costs

Model	Hardware	Power (W)	Hours	kWh·PUE	CO_2e	Cloud compute cost
$\text{Transformer}_{base}$	P100x8	1415.78	12	27	26	\$41–\$140
Transformer_{big}	P100x8	1515.43	84	201	192	\$289–\$981
ELMo	P100x3	517.66	336	275	262	\$433–\$1472
BERT_{base}	V100x64	12,041.51	79	1507	1438	\$3751–\$12,571
BERT_{base}	TPUv2x16	—	96	—	—	\$2074–\$6912
NAS	P100x8	1515.43	274,120	656,347	626,155	\$942,973–\$3,201,722
NAS	TPUv2x1	—	32,623	—	—	\$44,055–\$146,848
GPT-2	TPUv3x32	—	168	—	—	\$12,902–\$43,008

STRUBEL ET AL 2019, Energy and Policy Considerations for Deep Learning in NLP

SCHARTZ ET AL 2020, GREEN AI

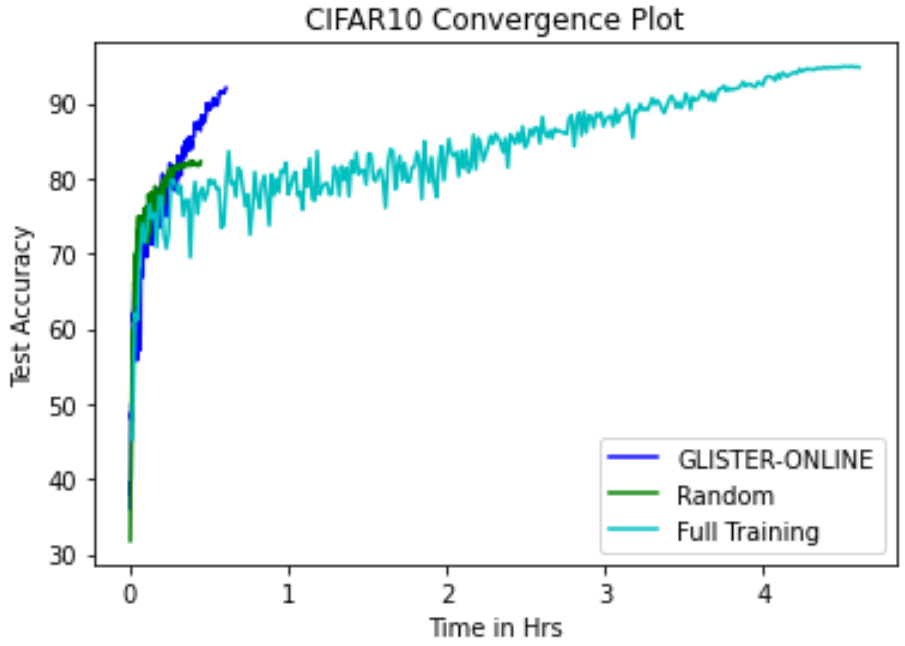
Can we train a model just using a very small subset of the entire data?

Approaches

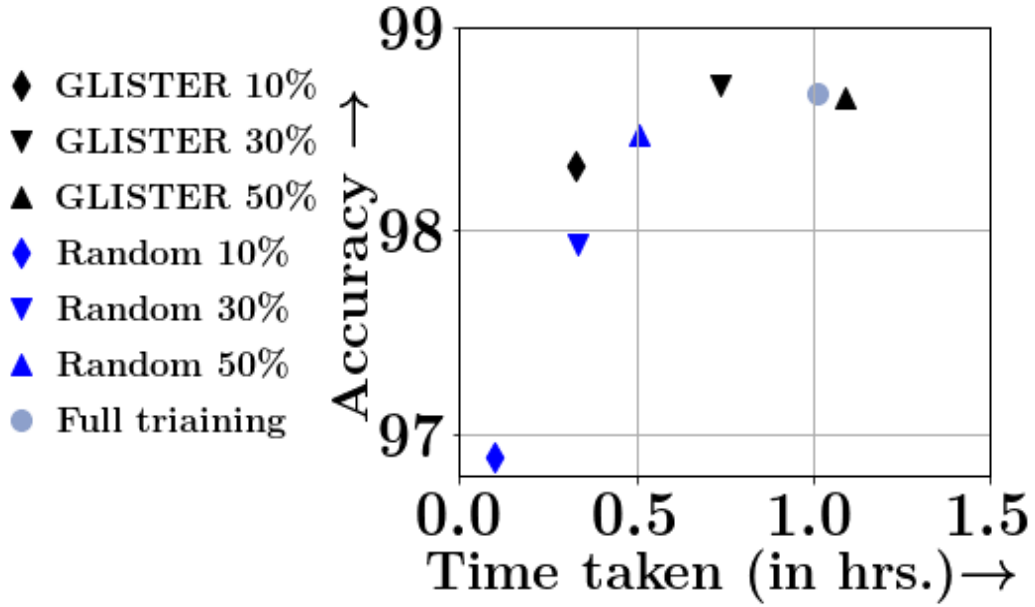
- Gradient Matching (Grad-Match)
- Discrete Bi-Level (GLISTER)
- Gradient Coreset (CRAIG)

1. Grad-Match: https://arxiv.org/pdf/2103.00123.pdf
2. GLISTER: https://arxiv.org/pdf/2012.10630.pdf
3. CRAIG: https://arxiv.org/abs/1906.01827

Empirical Results I



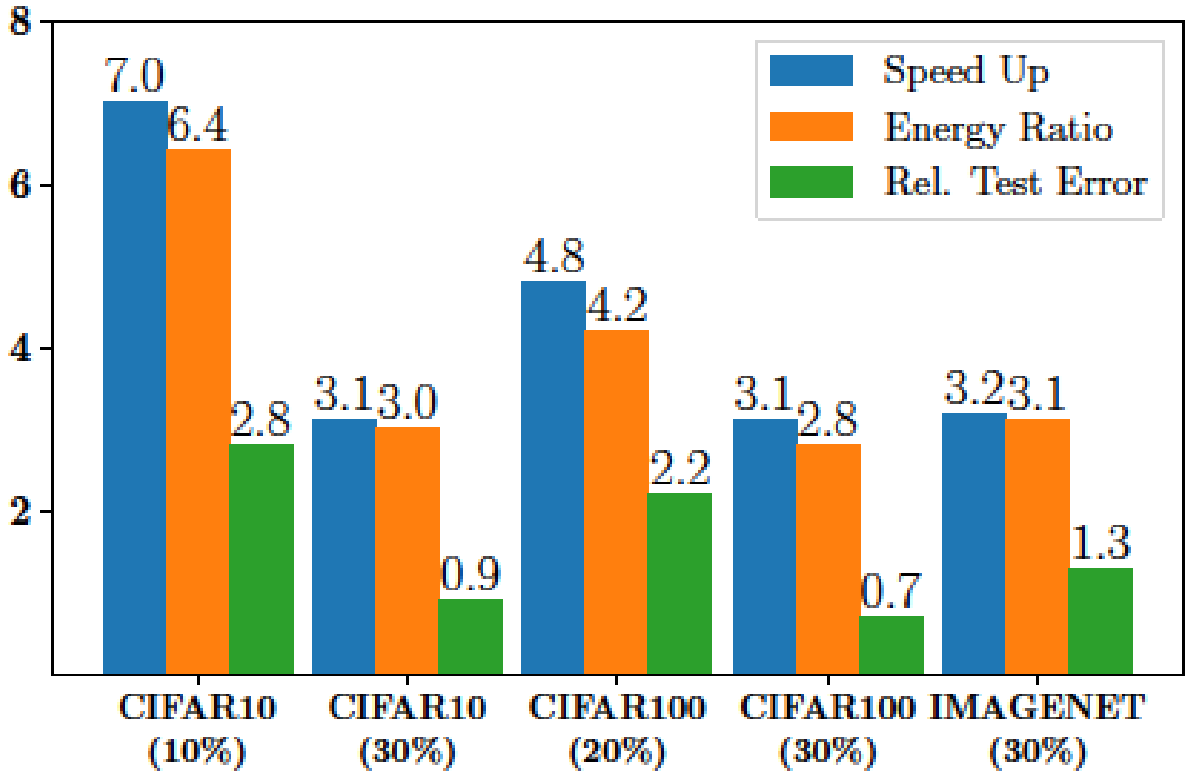
RESNET18 Model on CIFAR10 Dataset



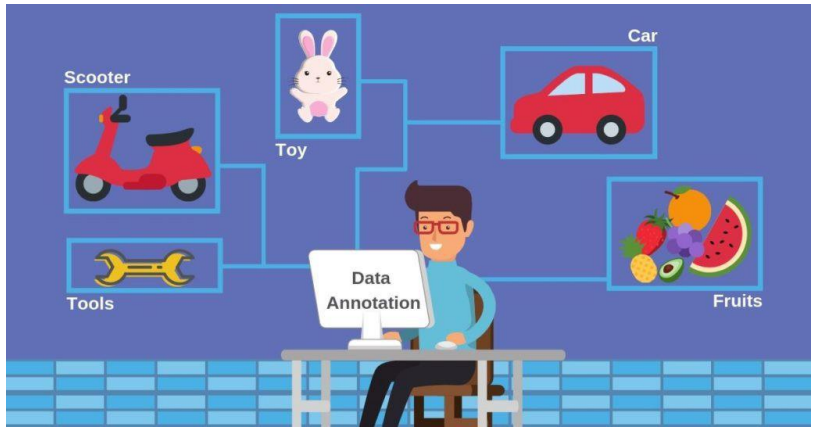
LeNet Model on MNIST Dataset

GLISTER achieves a 7.2x speedup on CIFAR10 dataset using RESNET 18 and 3x speedup on MNIST at 10% subset, while losing 3% and 0.2% in terms of accuracy, respectively.

Empirical Results II (Grad-Match)



Labeling Large Datasets is Expensive



Example: Labeling an 50K Image Dataset for Classification

$5K - $32K

Example: Labeling an 50K Image Dataset for Detection

$10K - $45K

Example: Labeling an 50K 3D Point Clouds

$120K - $250K

Example: Labeling an 50K Text classification Dataset

$80K - $150K

Labeling Sources: Google Labeling, Amazon Mechanical Turk, Hive.AI

Diversified Active Learning

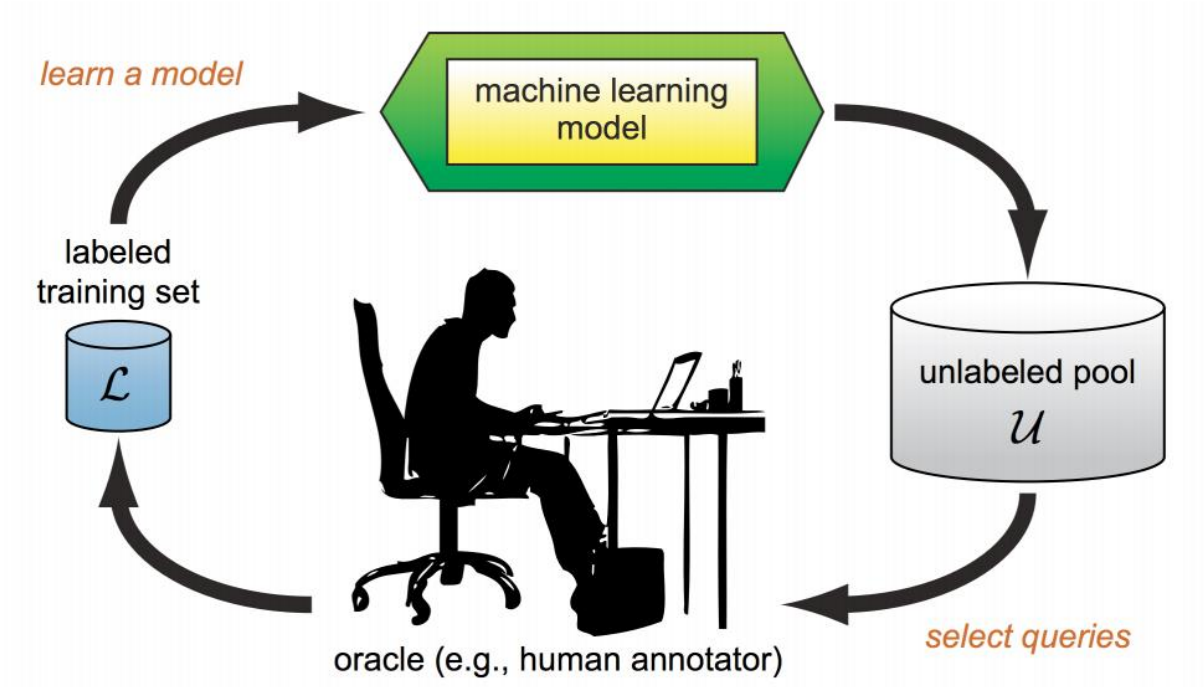


Figure: Source: Settle, 2010

- Basic idea of Active Learning: Pick the Most uncertain samples to label
- In practice, the human labeler labels data (e.g. images) in batches
- Does it make sense to pick diverse and uncertain instances to label?

Wei-Iyer-Bilmes, 2015, Sener and Savarese 2018, Kaushal et al 2019, Ash et al 2020

Approach 1: Filtered Active Submodular Selection (FASS)

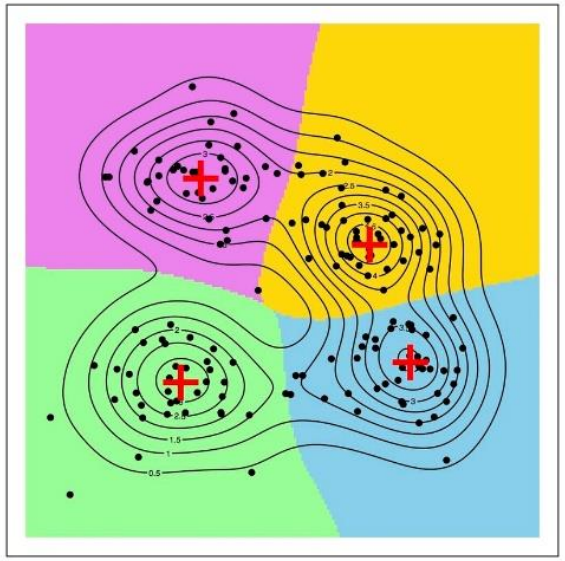
- (a) Uncertainty filtering:
  - Remove data that system is already certain about.
- (b) Subset selection:
  - Formulate as $\max_{|S| \leq k} f(S)$;
  - f is instantiated by hypothesized labels.

1. https://www.cse.iitb.ac.in/~ganesh/papers/wacv2019b.pdf
2. http://proceedings.mlr.press/v37/wei15.pdf

Filtered Active Submodular Selection

Approach 2: Active Learning by Picking Diverse Gradients

- Idea: Pick a set of examples with diverse gradients
- Very closely related to CRAIG: If we use Facility Location as the diversity function!
- Also, related to FASS, but does not require the Uncertainty Filtering: picking diverse gradients ensures uncertainty and diversity together!
- Extend Grad-Match to active learning setting

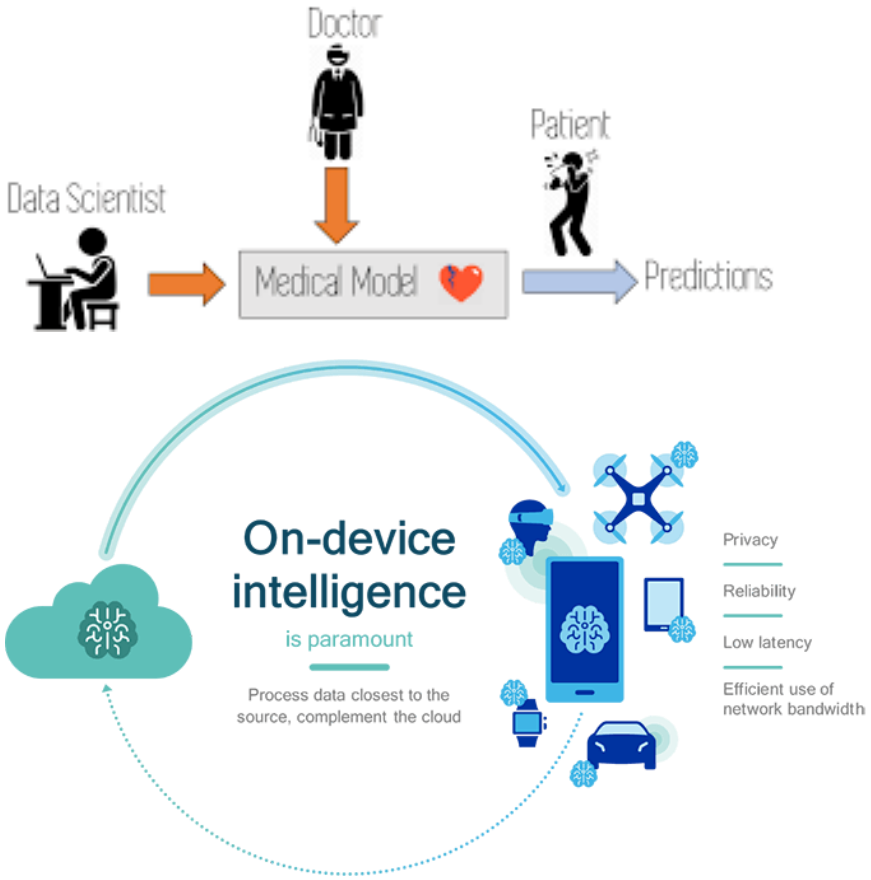


Paper: https://arxiv.org/abs/1906.03671

Ash et al 2020

Approach 3: Submodular Mutual Information Based Active Learning

Resource Constraints and Feature Costs



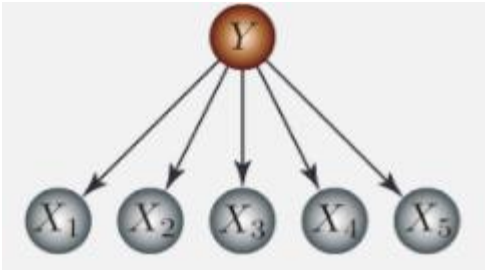
1. Eliciting Features are costly (e.g. lab tests, genetic tests etc.)
2. AI/ML Models need to run on low resource devices (edge devices)
    1. Privacy
    2. Low Latency
    3. Bandwidth and costs

Mutual Information Based Feature Selection

- Denote Y as the class variable and $X_1, \dots, X_n$ as features.
- MI Based Feature Selection:

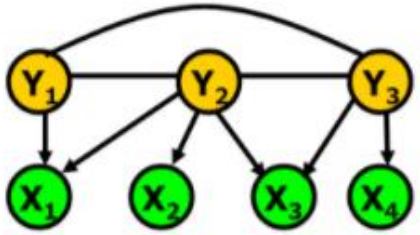
$$\max_{X:|X|\leq k} I(Y; X_A) \text{ where } \mathrm{F(A)} = I(Y; X_A) = H(X_A) - H(X_A|Y)$$

- Unfortunately $I(Y; X_A)$ is not always submodular!

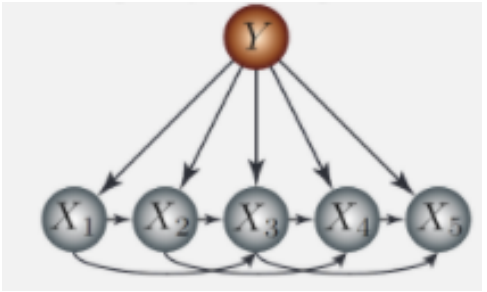


NB Assumption Holds

If X_i are all conditionally independent given Y, then F(A) is submodular! [Krause & Guestrin `05]



Proof:
"information never hurts"



NB Assumption Fails

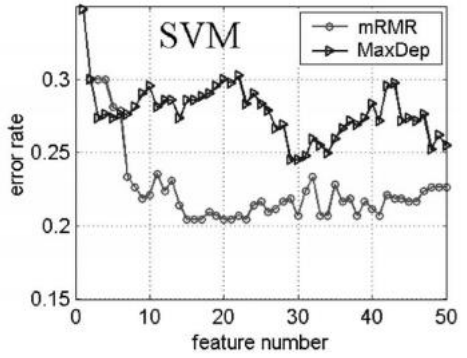
MI Based Feature Selection (Contd)

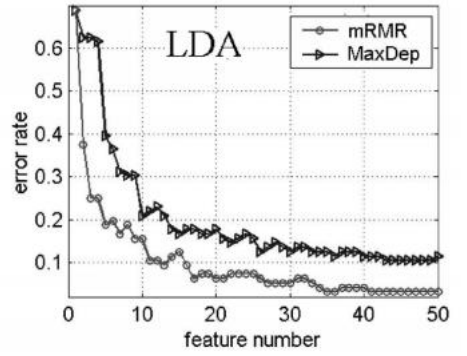
- MI Based Feature Selection:

$$\max_{A:|A|\leq k} I(Y; X_A) \text{ where } I(Y; X_A) = \mathrm{F(A)} = H(X_A) - H(X_A|Y)$$

- Even when $I(Y; X_A)$ is not submodular, it is a difference of submodular functions [Iyer & Bilmes 2012]
- However, estimating the MI can be difficult (particularly with large feature sets!)
- Many MI based FS frameworks approximate the MI (by breaking it into relevance and redundancy terms)

$$\max_{A:|A|\leq k} \sum_{i\in A} I(X_i, Y) - \sum_{i,j\in A} I(X_i; X_j)$$





http://ranger.uta.edu/~chqding/papers/mRMR_PAMI.pdf

This is submodular!!

[Peng et al 2005, Brown et al 2012, …]

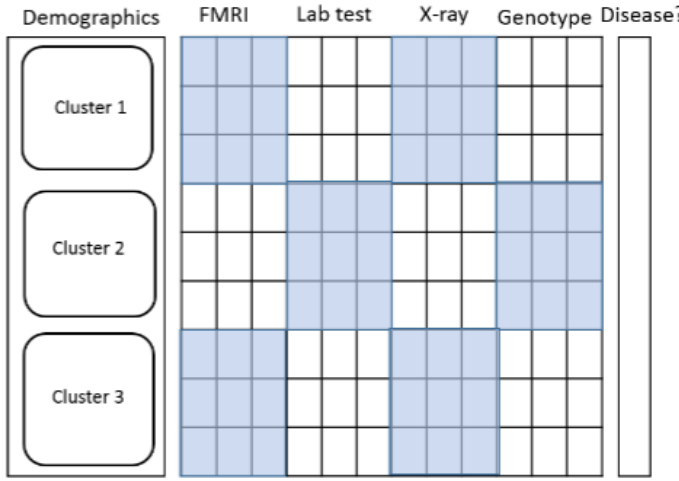
Cost Aware Feature Selection

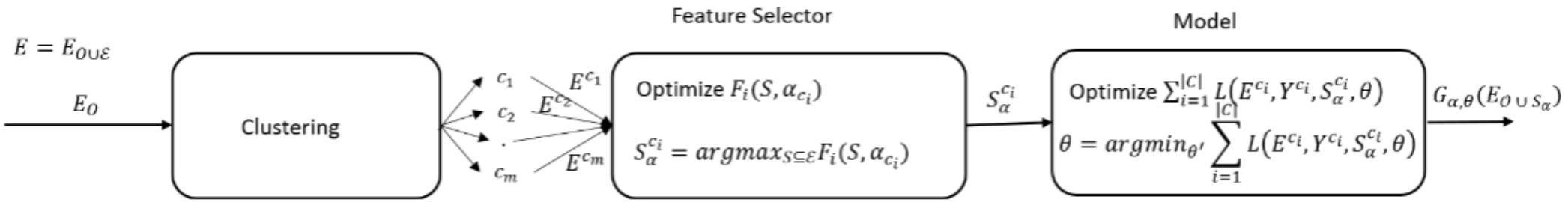
- Can we do feature selection in the presence of feature costs?

(e.g. medical diagnosis of a patient)

- Possibly different features may be relevant to different people
- Idea: Cluster patients based on observed features (e.g. demographics)
- Use the Mutual Information FS Module and perform joint training

$$\max_{A:|A|\leq k} \sum_{i\in A} I(X_i, Y) - \sum_{i,j\in A} I(X_i; X_j)$$

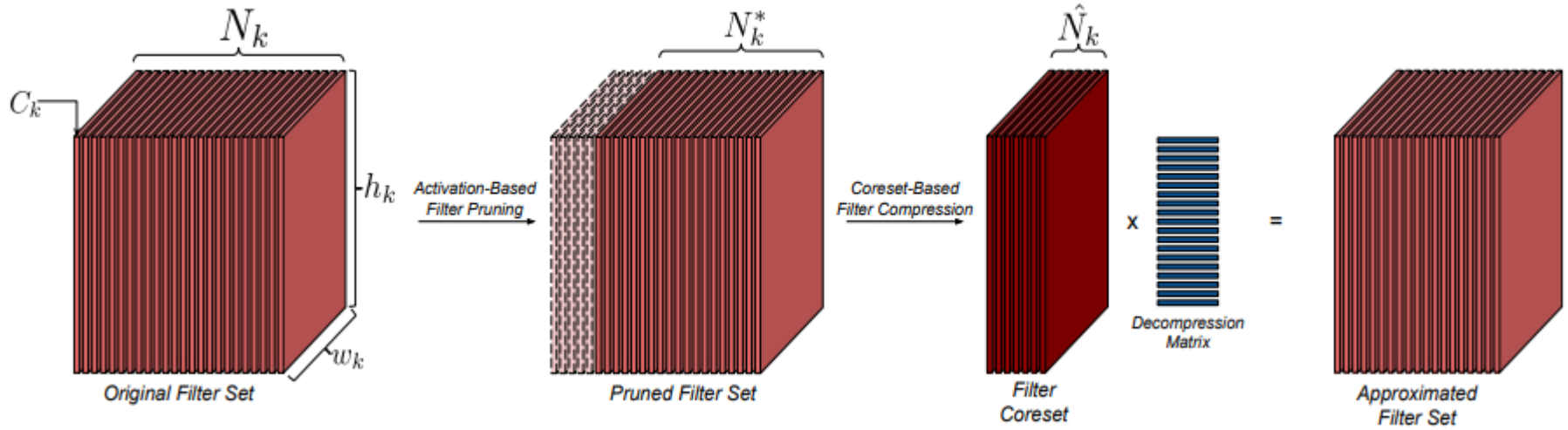




https://starling.utdallas.edu/assets/pdfs/CAFE_CODS2021.pdf

[Das et al 2020, …]

Coresets for Model Compression



Compress a Neural Network at each layer by removing redundancies in the weights and neurons

Dubey et al ECCV 2018 (https://arxiv.org/abs/1807.09810), Mussay et al ICLR 2020 (https://arxiv.org/abs/1907.04018), …