# Model-Free Subset Selection

- Introduction
- Active Learning
  - Distance-based
  - Ensemble-based
- Dataset Distillation
  - Synthesize Summaries
  - Gradient-Analysis

# Introduction

- Big data presents new challenges as gathering, storing, and analyzing them becomes expensive.
- Select or generate small summaries of large data sets.
- Aim to
  - Train a model with less data, e.g. active learning
  - Get a small dataset, e.g. video summarization

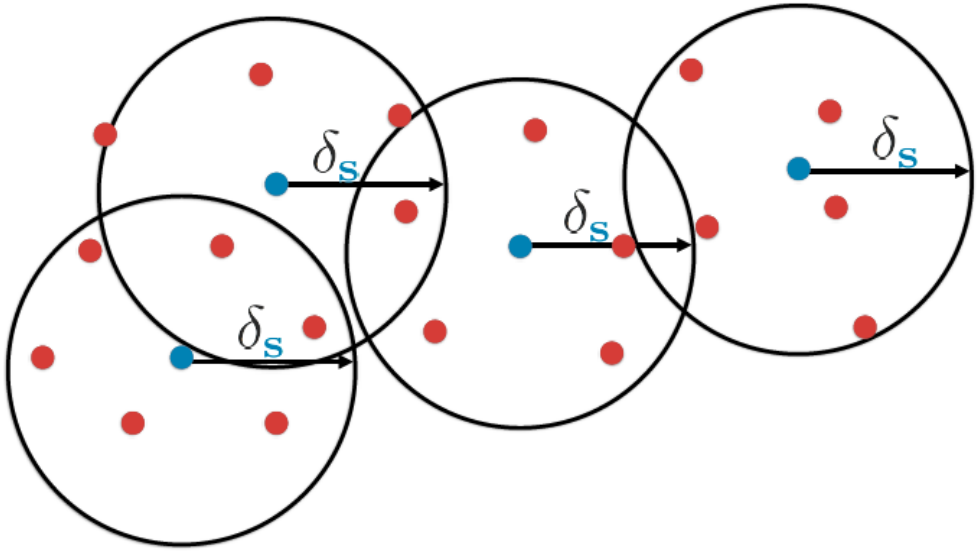
# Content

- Active Learning
  - Distance-based
  - Ensemble-based
- Dataset Distillation
  - Synthesize Summaries
  - Gradient-Analysis

# Core-Set [ICLR, 2018]

$$\min_{\mathbf{s}^1:|\mathbf{s}^1|\leq b} \max_i \min_{j\in\mathbf{s}^1\cup\mathbf{s}^0} \Delta(\mathbf{x}_i, \mathbf{x}_j)$$

choose b center points such that the largest distance between a data point and its nearest center is minimized




**Algorithm 1** k-Center-Greedy

**Input:** data $\mathbf{x}_i$, existing pool $\mathbf{s}^0$ and a budget $b$
Initialize $\mathbf{s} = \mathbf{s}^0$
**repeat**
  $u = \arg\max_{i\in[n]\setminus\mathbf{s}} \min_{j\in\mathbf{s}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
  $\mathbf{s} = \mathbf{s} \cup \{u\}$
**until** $|\mathbf{s}| = b + |\mathbf{s}^0|$
**return** $\mathbf{s} \setminus \mathbf{s}^0$

# Core-Set

$$Feasible(b, \mathbf{s}^0, \delta, \Xi) \longrightarrow \min_{\mathbf{s}^1} \max_i \min_{j \in \mathbf{s}^1 \cup \mathbf{s}^0} \Delta(\mathbf{x}_i, \mathbf{x}_j) \leq \delta.$$

Assume an upper limit on the number of outliers $\Xi$ such that our algorithm can choose not to cover at most $\Xi$ unsupervised data points.

**Algorithm 2** Robust k-Center

**Input:** data $\mathbf{x}_i$, existing pool $\mathbf{s}^0$, budget $b$ and outlier bound $\Xi$
**Initialize** $\mathbf{s}_g$ = k-Center-Greedy$(\mathbf{x}_i, \mathbf{s}^0, b)$
$\delta_{2-OPT} = \max_j \min_{i \in \mathbf{s}_g} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
$lb = \frac{\delta_{2-OPT}}{2}$, $ub = \delta_{2-OPT}$
**repeat**
  **if** $Feasible(b, \mathbf{s}^0, \frac{lb+ub}{2}, \Xi)$ **then**
    $ub = \max_{i,j|\Delta(\mathbf{x}_i,\mathbf{x}_j) \leq \frac{lb+ub}{2}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
  **else**
    $lb = \min_{i,j|\Delta(\mathbf{x}_i,\mathbf{x}_j) \geq \frac{lb+ub}{2}} \Delta(\mathbf{x}_i, \mathbf{x}_j)$
  **end if**
**until** $ub = lb$
**return** $\{i \; s.t. \; u_i = 1\}$

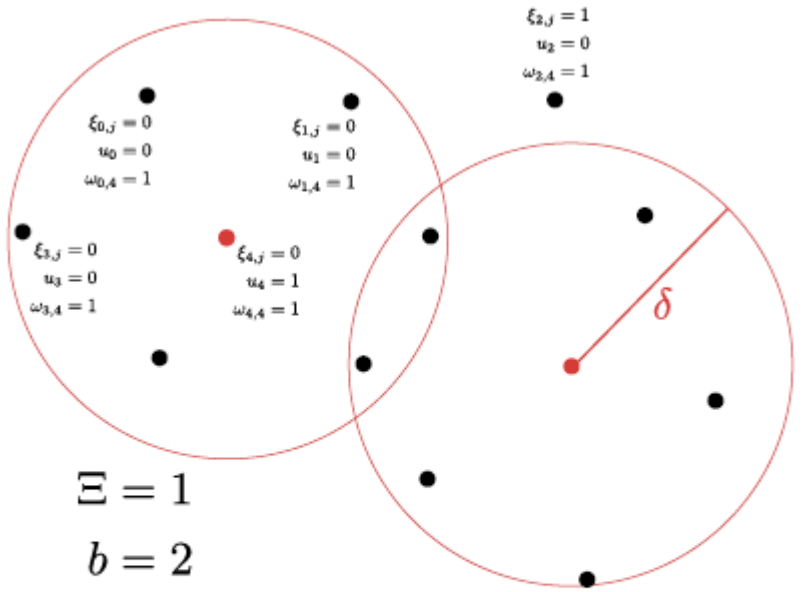



Figure 2: Visualizations of the variables. In this solution, the $4^{th}$ node is chosen as a center and nodes $0, 1, 3$ are in a $\delta$ ball around it. The $2^{nd}$ node is marked as an outlier.

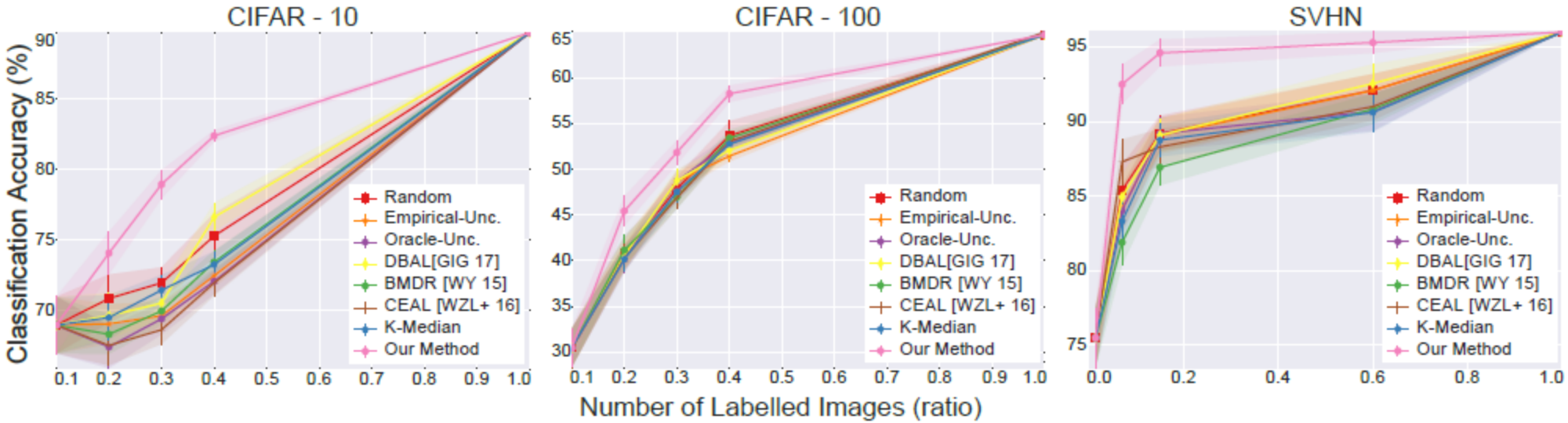



Figure 3: Results on Active Learning for Weakly-Supervised Model (error bars are std-dev)

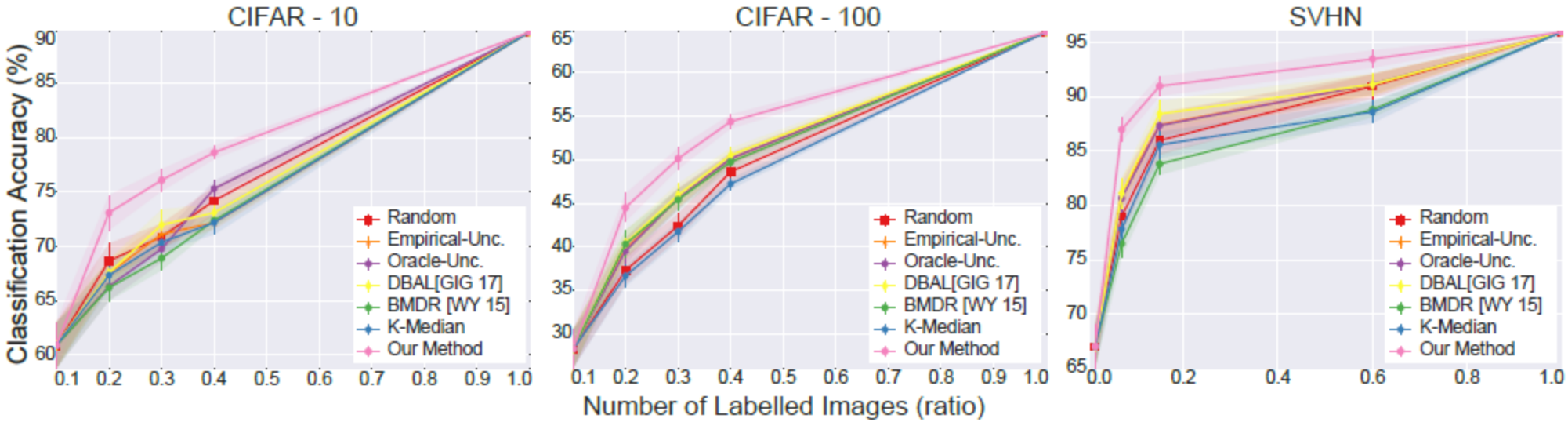



Figure 4: Results on Active Learning for Fully-Supervised Model (error bars are std-dev)

# Diverse mini-batch Active Learning

$$f(\mathcal{S}) = \sum_{x_i \in \mathcal{X}^U} \min_{x_j \in \mathcal{S}} d(x_i, x_j) \qquad \mathcal{S} \subseteq \mathcal{X}^U$$

Facility Location

*K*-Means: $$\sum_{x_i \in \mathcal{X}^U} \sum_k z_{i,k} \|x_i - \mu_k\|^2$$

Assume we are also given informativeness scores $s_i$ for every example

Weighted *K*-Means: $$\sum_{x_i \in \mathcal{X}^U} \sum_k z_{i,k} s_i \|x_i - \mu_k\|^2$$

[Zhdanov (Amazon). Diverse mini-batch Active Learning. ArXiv, 2019.]

**Algorithm 1** Diverse mini-Batch Active Learning (DBAL)

**Input:** dataset of examples $x_i$, budget $B$, batch-size $k$, pre-filter factor $\beta$

Select first $k$ examples randomly, obtain labels for these examples

**repeat**

Train classifier on all the examples selected so far

Get informativeness for every unlabeled example

Prefilter to top $\beta k$ informative examples

Cluster $\beta k$ examples to $k$ clusters with (weighted) K-means

Select $k$ different examples closest to the cluster centers, obtain labels for these examples

**until** Budget $B$ is exhausted

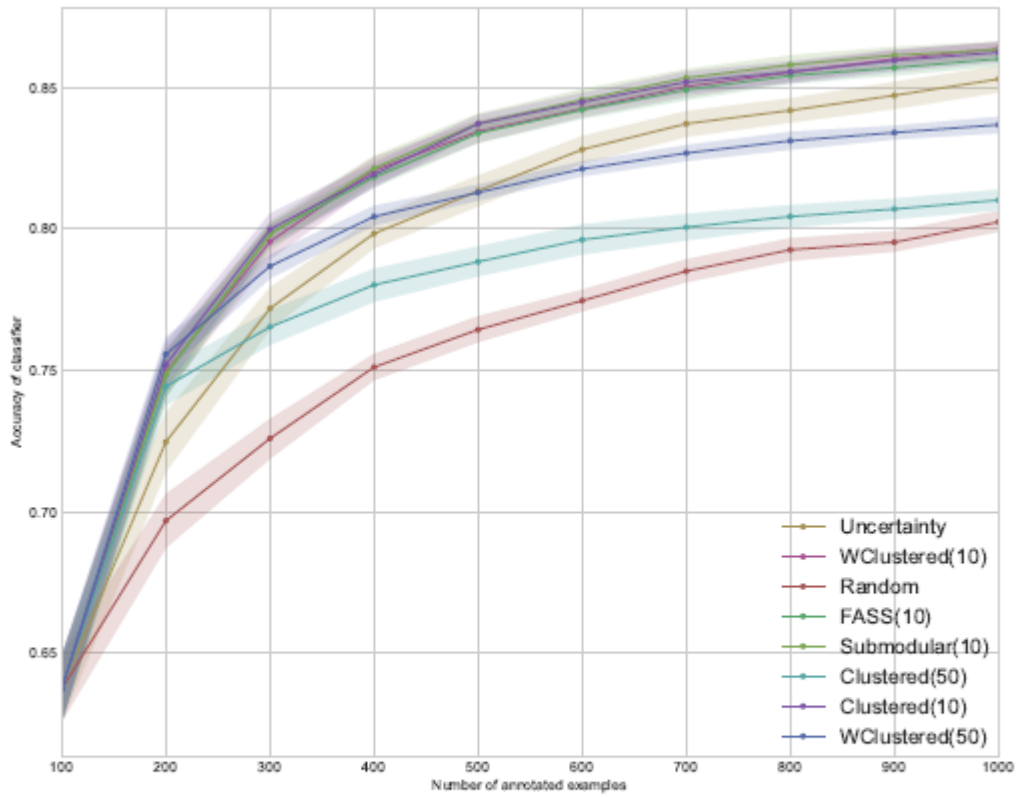



Figure 1: Accuracy on Browse Nodes UK dataset

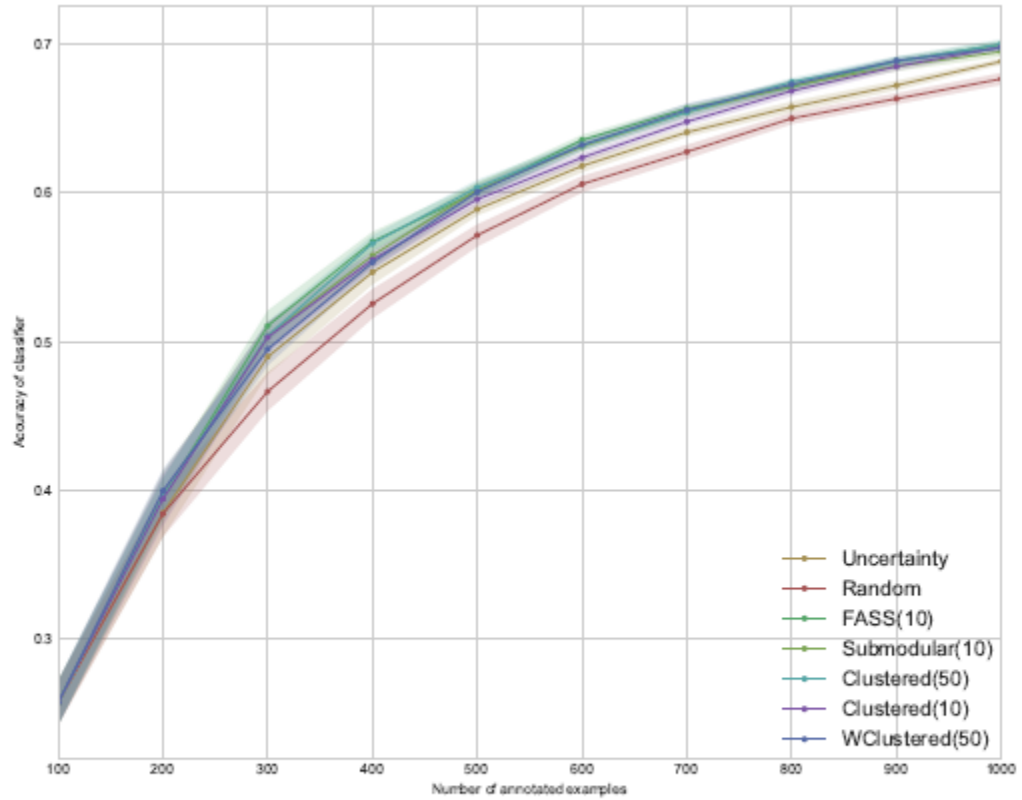



Figure 2: Accuracy on 20 Newsgroups Dataset

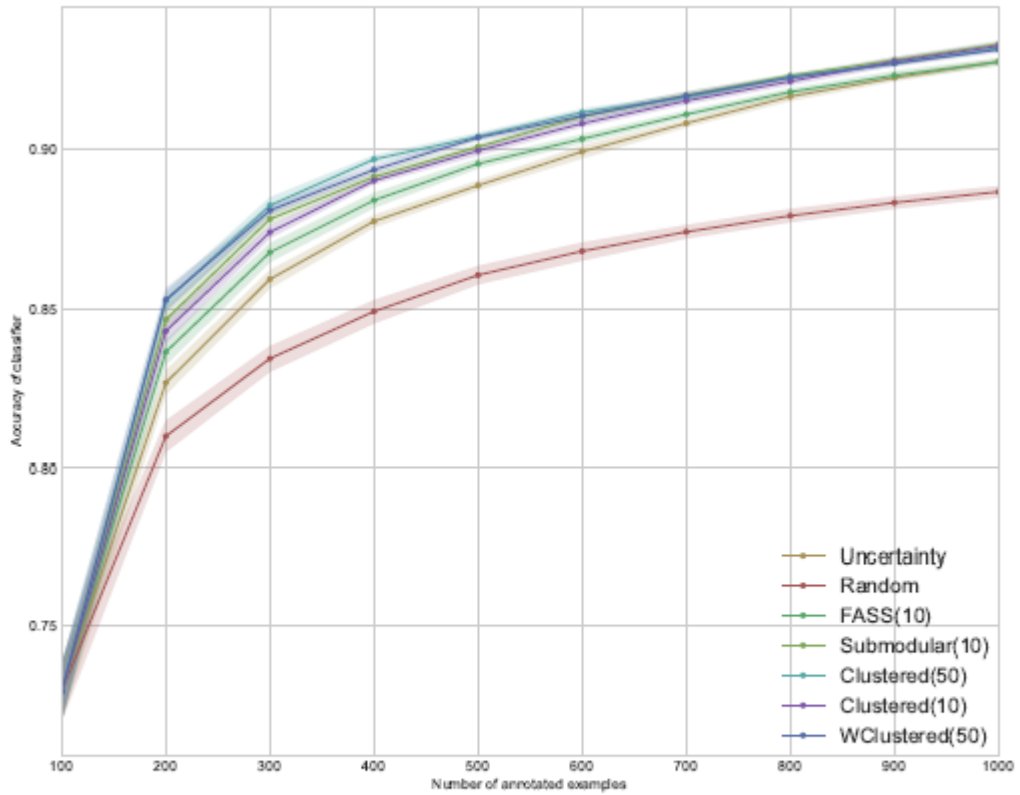



Figure 3: Accuracy on MNIST dataset

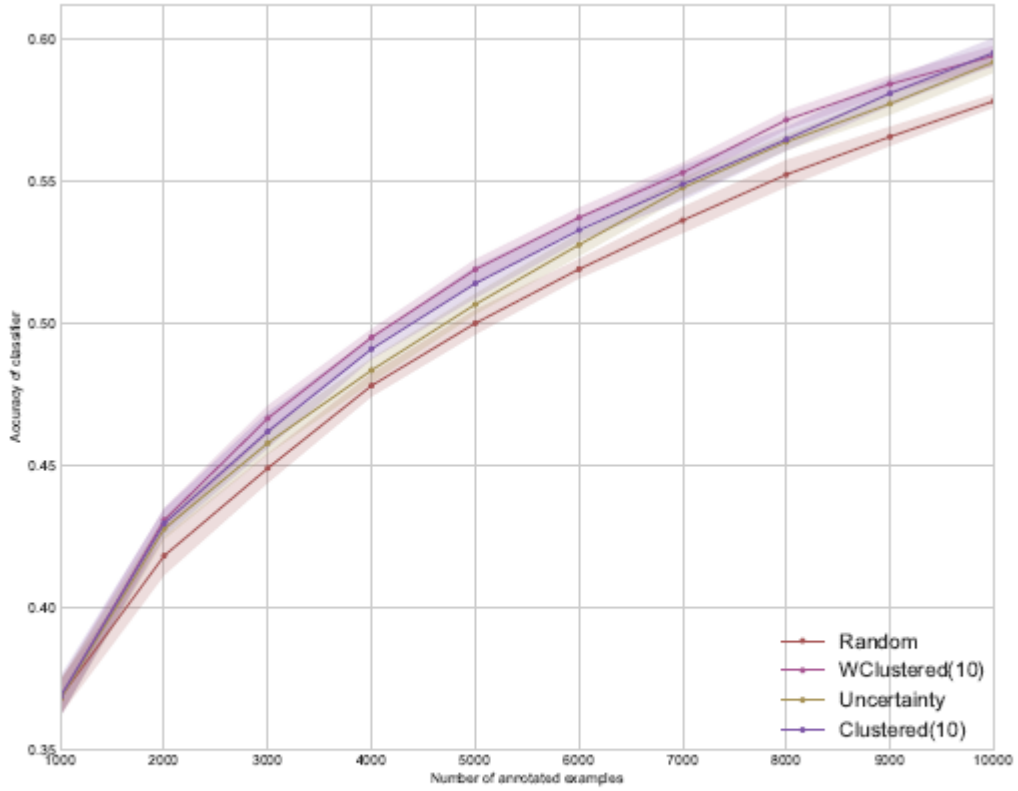



Figure 5: Accuracy on CIFAR-10 dataset

# Content

- Active Learning
  - Distance-based
  - Ensemble-based

- Dataset Distillation
  - Synthesize Summaries
  - Gradient-Analysis

# Active Dataset Subsampling

Training data

$$\mathsf{P}(y = k|\mathbf{x}^*, L) \approx \frac{1}{E}\sum_{e\in E}\mathsf{P}(y = k|\mathbf{x}^*, \theta_e),\ \theta_e \sim \mathsf{q}(\theta)$$

$$\mathcal{H}(\mathbf{p}) = -\mathbf{p}^T \log \mathbf{p}.$$

- Acquisition Functions

- Mutual information

$$\underbrace{\mathcal{I}(\mathbf{p})}_{Model} = \underbrace{\mathcal{H}(\mathbf{p})}_{Predictive} - \underbrace{\frac{1}{E}\sum_{e\in E}\mathcal{H}(\mathbf{p}^{(e)})}_{Data}$$

when the models make similar predictions $\mathcal{I}(\mathbf{p}) \to 0$

- Variation ratios

$$\mathcal{V}(\mathbf{p}) = 1 - \frac{1}{E}\sum_{e\in E}(\arg\max_{k\in K}\mathbf{p}_k^{(e)} = M)$$

$M$ → the majority vote

the fraction of members in the ensemble that do not agree with the majority vote

[Chitta, *et.al.* (NVIDIA). Less is More: An Exploration of Data Redundancy with Active Dataset Subsampling. ArXiv, 2019.]

# Active Dataset Subsampling

1. A **labeled dataset**, consisting of $N_l$ labeled pairs, $L = \{(\mathbf{x}_l^j, y_l^j)\}_{j=1}^{N_l}$, where each $\mathbf{x}^j \in X$ is a data point and each $y^j \in Y$ is its corresponding label.
2. An **acquisition model**, $\mathcal{M}_a : X \to Y$. For our ensemble based uncertainty estimation technique, the acquisition model $\mathcal{M}_a$ takes the form of a set of $E$ different DNNs with parameters $\{\theta_a^{(e)}\}_{e=1}^{E}$.
3. A **subsampled dataset**, $S = \{(\mathbf{x}_s^j, y_s^j)\}_{j=1}^{N_s}$, where $S$ is a subset of $L$ selected using an acquisition function $\alpha(\mathbf{x}, \mathcal{M}_a)$.
4. A **subset model**, $\mathcal{M}_s : X \to Y$, with parameters $\theta_s$, trained on $S$.

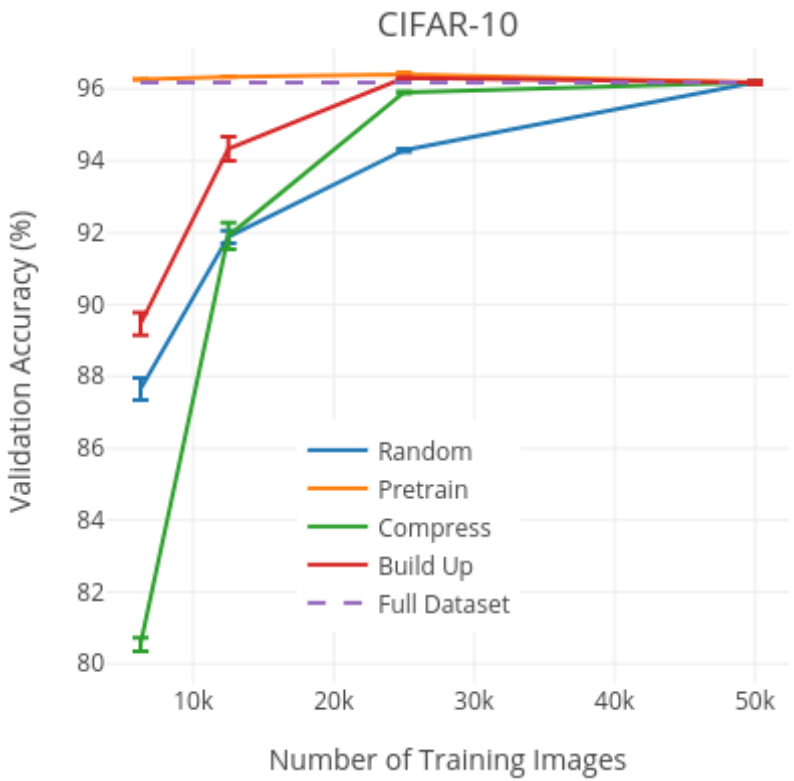
# Initialization

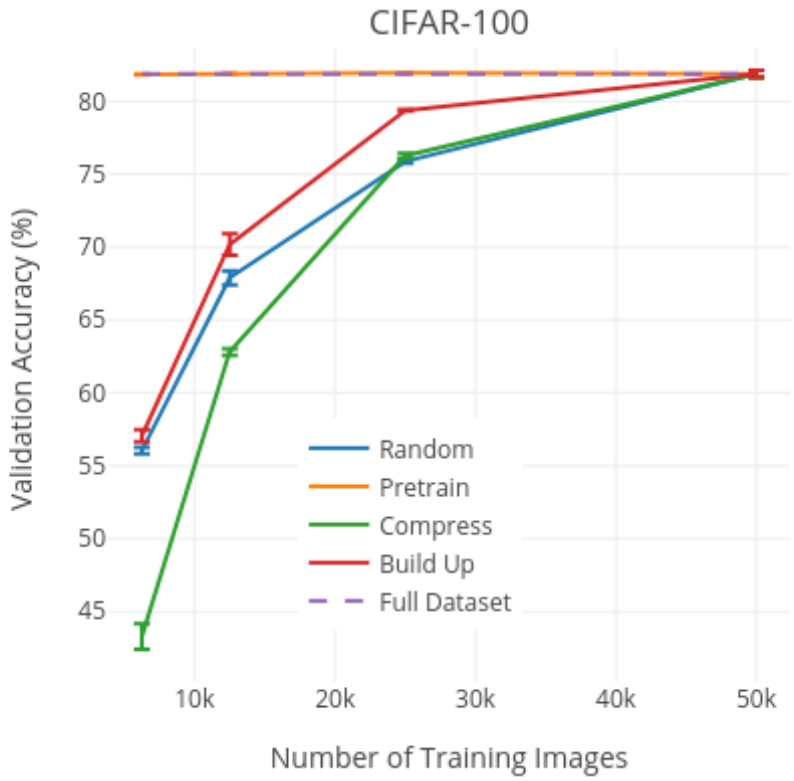
- The **pre-train scheme** uses the entire dataset L for pre-training both the acquisition and subset models. During optimization, the subset model is then fine tuned on the subsampled dataset S.

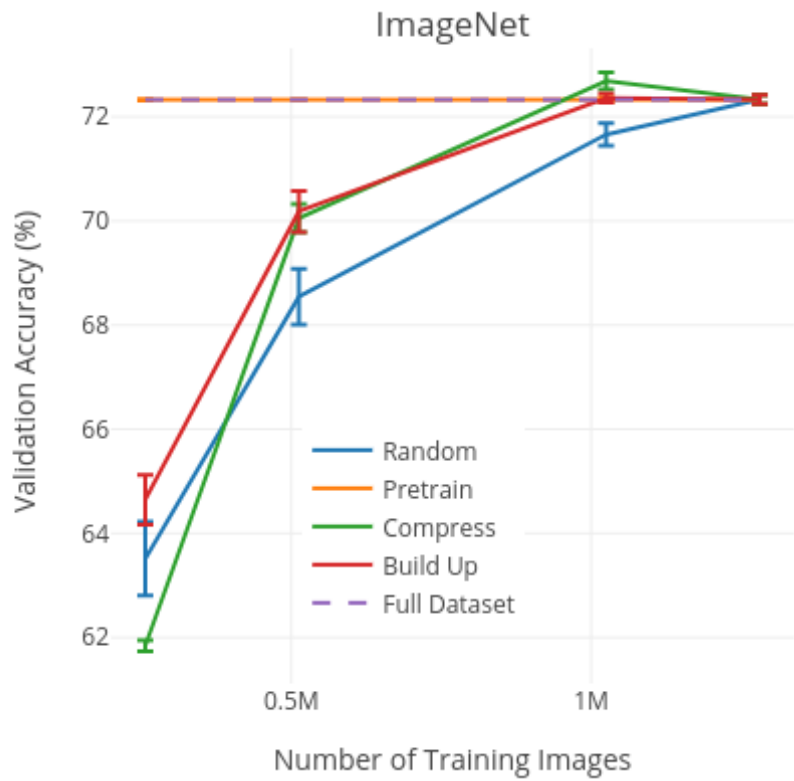
- In the **compress scheme**, the acquisition model is pre-trained on L but the subset model is randomly initialized and trained from scratch on S. The acquisition model therefore accesses all the data and then ’compresses’ the dataset for the subset model.

- In the **build-up scheme**, we aim to emulate an iteration in a typical active learning loop. A set of existing subset models are used as an acquisition model, in an approach with multiple iterations of ADS.

# Experiment


CIFAR-10
Validation Accuracy (%)
96
94
92
90
88
86
84
82
80
10k
20k
30k
40k
50k
Number of Training Images
Random
Pretrain
Compress
Build Up
Full Dataset


CIFAR-100
Validation Accuracy (%)
80
75
70
65
60
55
50
45
10k
20k
30k
40k
50k
Number of Training Images
Random
Pretrain
Compress
Build Up
Full Dataset


ImageNet
Validation Accuracy (%)
72
70
68
66
64
62
0.5M
1M
Number of Training Images
Random
Pretrain
Compress
Build Up
Full Dataset

# Minimum-Margin Active Learning

$$margin(h, z) = h(z; \hat{y}_1(z)) - h(z; \hat{y}_2(z))$$

**Algorithm 1** Min-Margin Active Sampling

**Inputs**: Initial sample $\mathcal{T}_0$, candidate sample set $\mathcal{Z}$, number of bootstrapped models $K$, bootstrap sample size fraction $\beta$, number of candidate examples to select $B$, learning procedure $H$

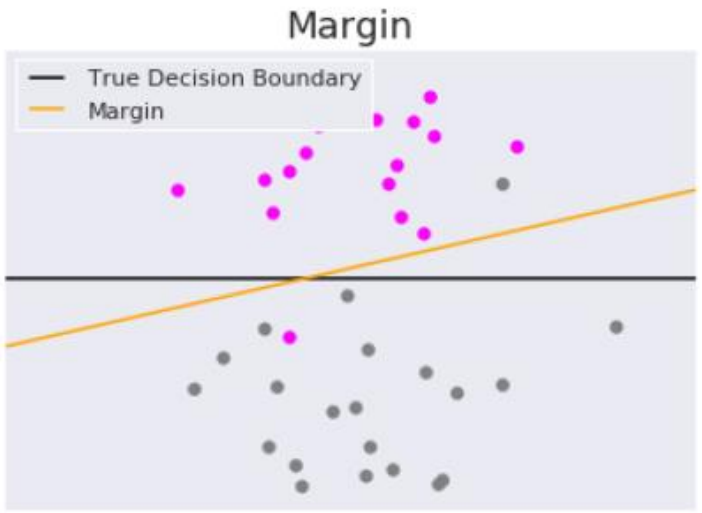
**Bootstrap**: For each $k = 1, \ldots, K$, let $\mathcal{T}_k$ be a random sample with replacement from $\mathcal{T}_0$ of $\lfloor \beta N_g \rfloor$ examples from each class $g$, and $h_k := H(\mathcal{T}_k)$

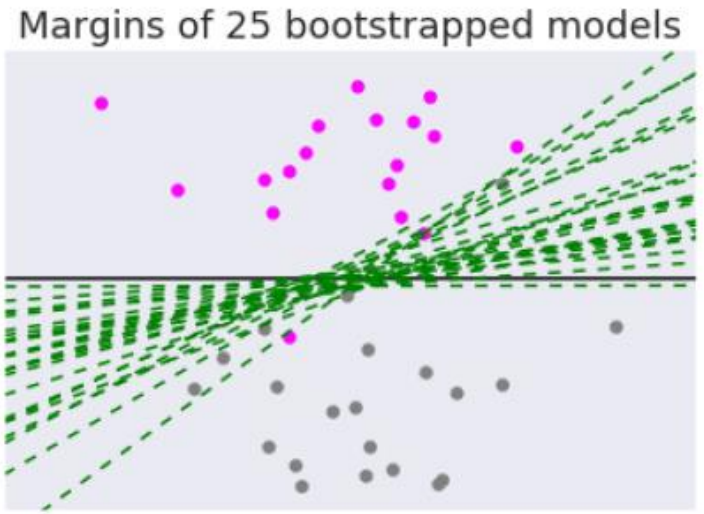
**Score**: For each candidate $z \in \mathcal{Z}$, let $score(z) := \min_{k \in [K]} \text{margin}(h_k, z)$.

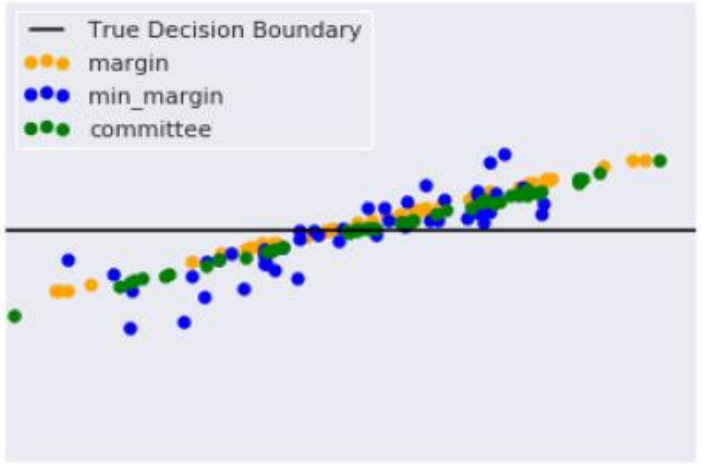
**Return** the $B$ candidates from $\mathcal{Z}$ with lowest $score$.

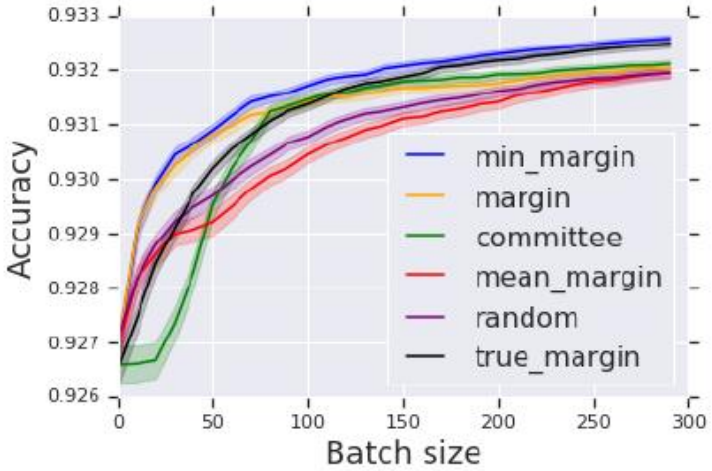
[Jiang and Gupta (Google). ArXiv, 2019.]

# Illustration

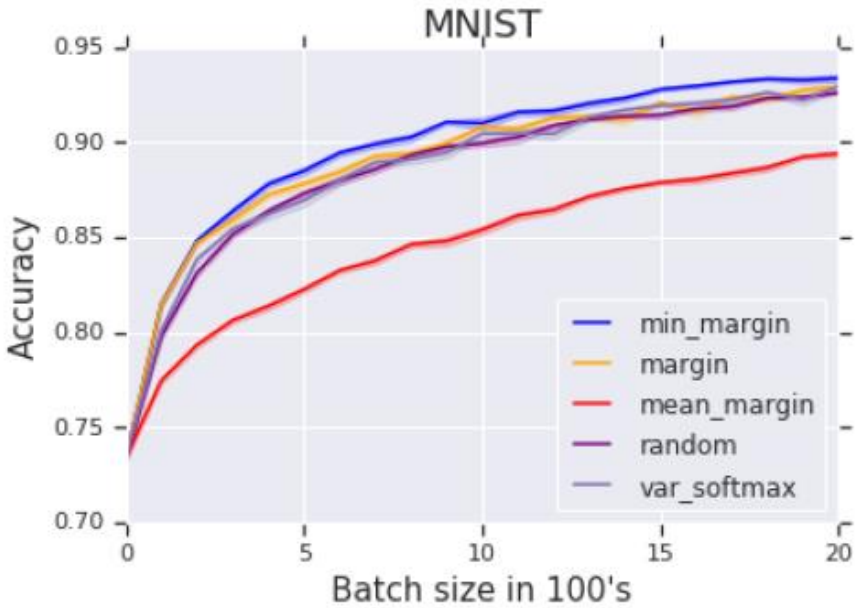


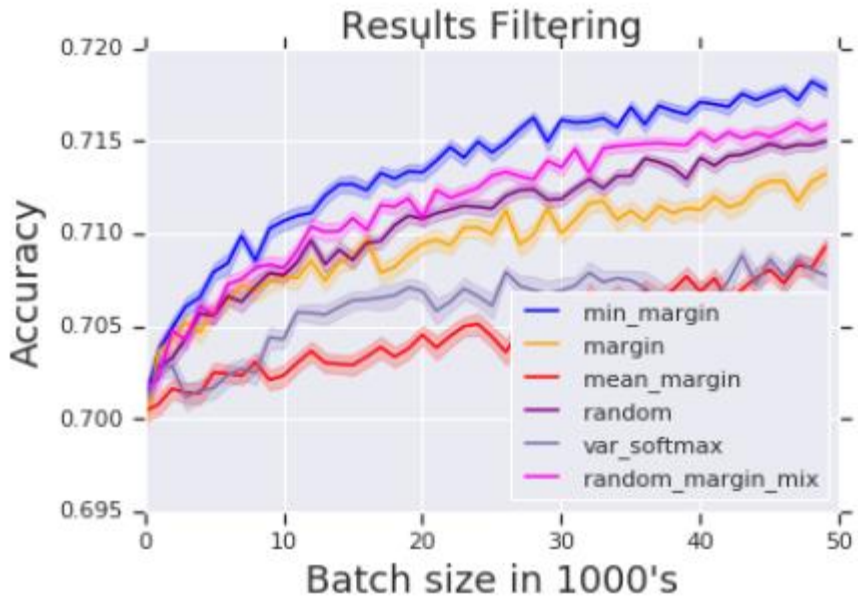



one-shot setting







Results Filtering
Accuracy
Batch size in 1000's
0.720
0.715
0.710
0.705
0.700
0.695
0
10
20
30
40
50
min_margin
margin
mean_margin
random
var_softmax
random_margin_mix


Query Intent
Accuracy
Batch size in 1000's
0.685
0.680
0.675
0.670
0.665
0.660
0.655
0.650
0
10
20
30
40
50
min_margin
margin
mean_margin
random
var_softmax
random_margin_mix

# Content

- Active Learning
  - Distance-based
  - Ensemble-based

- Dataset Distillation
  - Synthesize Summaries
  - Gradient-Analysis

# Synthesize Summaries

Standard training: $\theta^* = \arg\min_\theta \frac{1}{N}\sum_{i=1}^{N} \ell(x_i, \theta)$ $\theta_{t+1} = \theta_t - \eta \nabla_{\theta_t} \ell(\mathbf{x}_t, \theta_t)$ **Slow!**

of update steps to converge. Instead, we aim to learn a tiny set of synthetic distilled training data $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^{M}$ with $M \ll N$ and a corresponding learning rate $\eta$ so that a single GD step such as

$$\theta_1 = \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{\mathbf{x}}, \theta_0) \quad (2)$$

using these learned synthetic data $\tilde{\mathbf{x}}$ can greatly boost the performance on the real test set. Given an

$$\tilde{\mathbf{x}}^*, \tilde{\eta}^* = \arg\min_{\tilde{\mathbf{x}}, \tilde{\eta}} \mathcal{L}(\tilde{\mathbf{x}}, \tilde{\eta}; \theta_0) = \arg\min_{\tilde{\mathbf{x}}, \tilde{\eta}} \ell(\mathbf{x}, \theta_1) = \arg\min_{\tilde{\mathbf{x}}, \tilde{\eta}} \ell(\mathbf{x}, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell(\tilde{\mathbf{x}}, \theta_0))$$

The above distilled data optimized for a given initialization do not generalize well to other initializations.

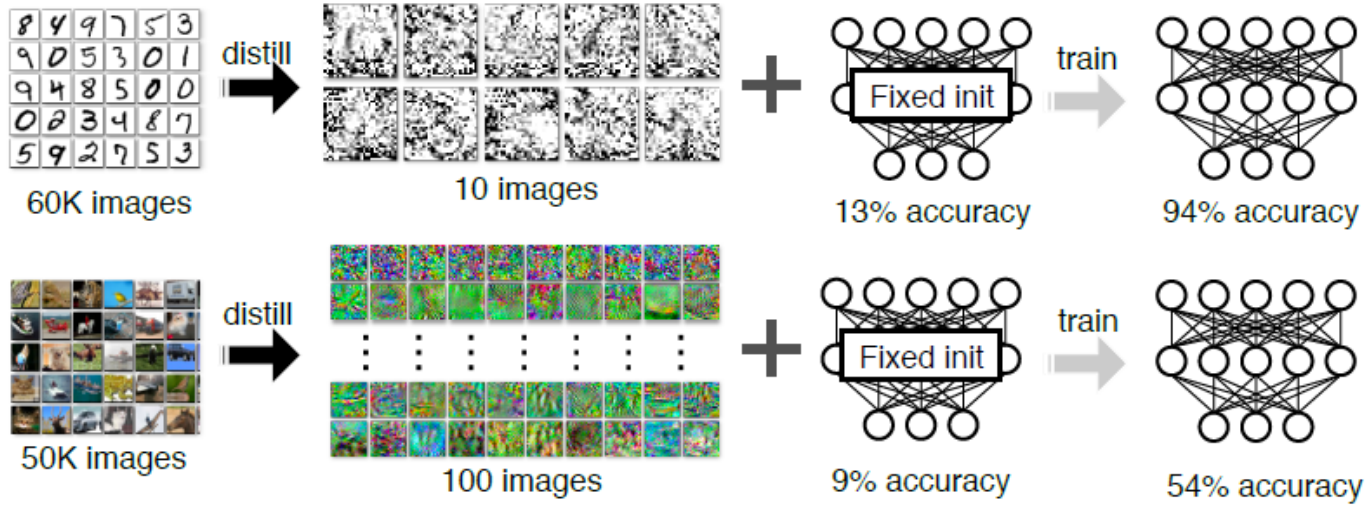
$$\tilde{\mathbf{x}}^*, \tilde{\eta}^* = \arg\min_{\tilde{\mathbf{x}}, \tilde{\eta}} \mathbb{E}_{\theta_0 \sim p(\theta_0)} \mathcal{L}(\tilde{\mathbf{x}}, \tilde{\eta}; \theta_0)$$

# Synthesize Summaries

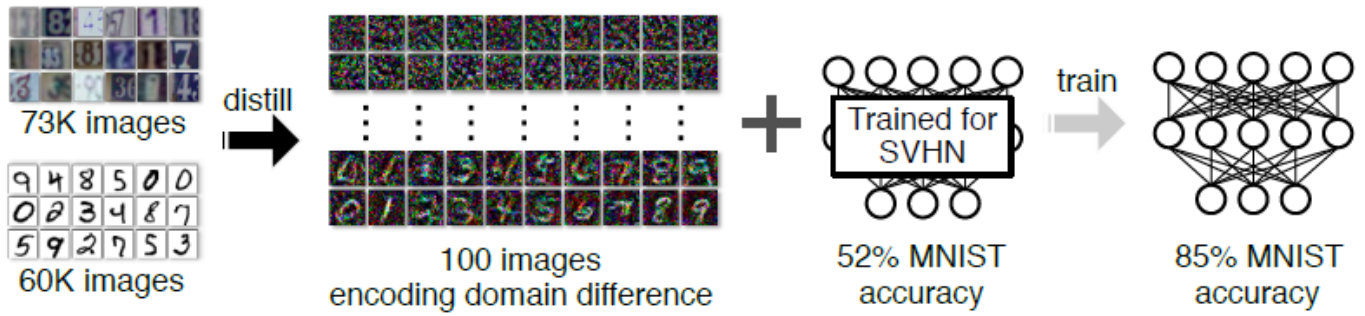
**Algorithm 1** Dataset Distillation

**Input:** $p(\theta_0)$: distribution of initial weights; $M$: the number of distilled data
**Input:** $\alpha$: step size; $n$: batch size; $T$: the number of optimization iterations; $\tilde{\eta}_0$: initial value for $\tilde{\eta}$

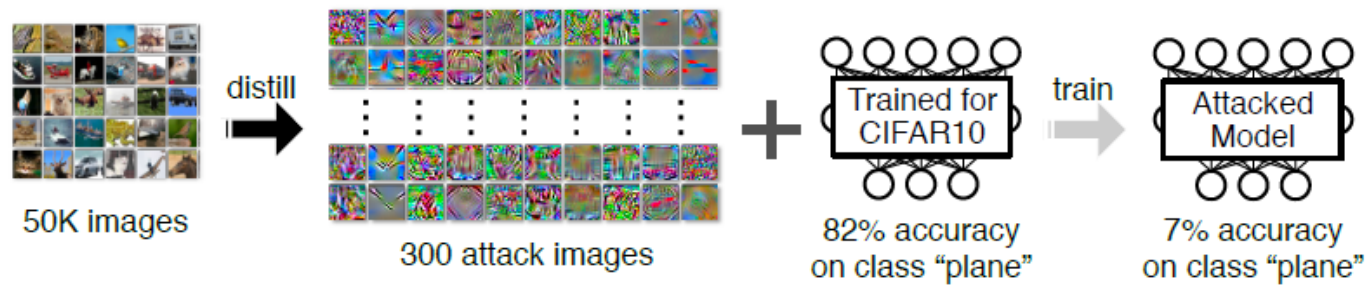
Initialize $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^{M}$ randomly, $\tilde{\eta} \leftarrow \tilde{\eta}_0$
**for each** training step $t = 1$ to $T$ **do**
    Get a minibatch of real training data $\mathbf{x}_t = \{x_{t,j}\}_{j=1}^{n}$
    Sample a batch of initial weights $\theta_0^{(j)} \sim p(\theta_0)$
    **for each** sampled $\theta_0^{(j)}$ **do**
        Compute updated parameter with GD: $\theta_1^{(j)} = \theta_0^{(j)} - \tilde{\eta}\nabla_{\theta_0^{(j)}}\ell(\tilde{\mathbf{x}}, \theta_0^{(j)})$
        Evaluate the objective function on real training data: $\mathcal{L}^{(j)} = \ell(\mathbf{x}_t, \theta_1^{(j)})$
    **end for**
    Update $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha\nabla_{\tilde{\mathbf{x}}}\sum_j \mathcal{L}^{(j)}$, and $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha\nabla_{\tilde{\eta}}\sum_j \mathcal{L}^{(j)}$
**end for**
**Output:** distilled data $\tilde{\mathbf{x}}$ and optimized learning rate $\tilde{\eta}$




(a) Dataset distillation on MNIST and CIFAR10




(b) Dataset distillation can quickly fine-tune pre-trained networks on new datasets




(c) Dataset distillation can maliciously attack classifier networks

# Content

- Active Learning
  - Distance-based
  - Ensemble-based

- Dataset Distillation
  - Synthesize Summaries
  - Gradient Analysis

# Gradient Analysis

- Objective:

$$f_\theta^* = \arg\min_{f_\theta \in \mathcal{F}_\theta} \mathcal{L}(f_\theta)$$

where

$$\mathcal{L}(f_\theta) = \left(\frac{1}{N}\sum_{i=1}^{N} l(f_\theta(\mathbf{x_i}), y_i)\right) + \mathcal{R}(f_\theta)$$

$$= \left(\frac{1}{N}\sum_{i=1}^{N} L_{i,\theta}\right) + \mathcal{R}(f_\theta).$$

- Batch Gradient Descent

$$\theta_{t+1} \leftarrow \theta_t - \left(\frac{\eta_t}{|\mathcal{B}_t|}\sum_{i\in\mathcal{B}_t} \nabla_\theta L_{i,\theta_t}\right) - \eta_t \nabla_\theta \mathcal{R}(f_{\theta_t})$$

- The magnitude of change in parameters from one iteration to the next:

$$\|\theta_{t+1} - \theta_t\| = \left\| \left(\frac{\eta_t}{|\mathcal{B}_t|}\sum_{i\in\mathcal{B}_t} \nabla_\theta L_{i,\theta_t}\right) - \eta_t \nabla_\theta \mathcal{R}(f_{\theta_t}) \right\|$$

- Upper bound:

$$\left\| \sum_{i\in\mathcal{B}_t} \nabla_\theta L_{i,\theta} \right\| \leq \sum_{i\in\mathcal{B}_t} \|\nabla_\theta L_{i,\theta}\|$$

- Top-k images with largest gradient magnitude

$$\mathcal{B}^* = \max_{\mathcal{B}:|\mathcal{B}|=k} \sum_{i=1}^{N} ||\nabla_\theta L_{i,\theta}||$$

# Gradient Analysis

**Algorithm 1** Gradient Analysis

**procedure** ANALYSIS($f_\theta$)
    Train network $f_\theta$ on all data
    Compute test accuracy
    **for** $i = 1, ..., N$ **do**
    **return** SUBSAMPLE_ANALYSIS($f_\theta$)

**procedure** SUBSAMPLE_ANALYSIS($f_\theta$)
    Subsample data using $\nabla_i$ ▷ See Section 3.3
    Retrain network $f_\theta$ on subsampled data only
    **return** data subsample, test accuracy

# Experiments

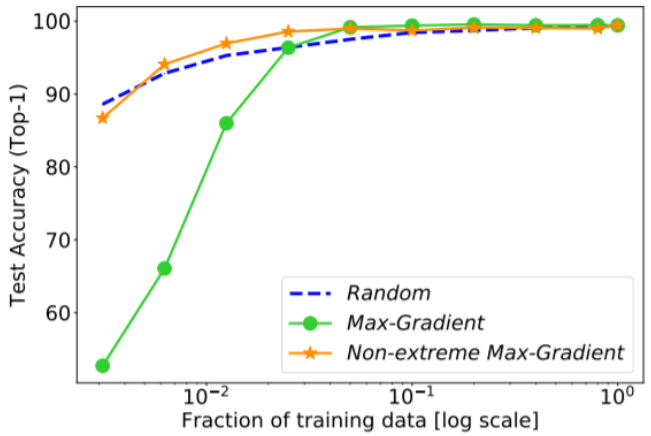
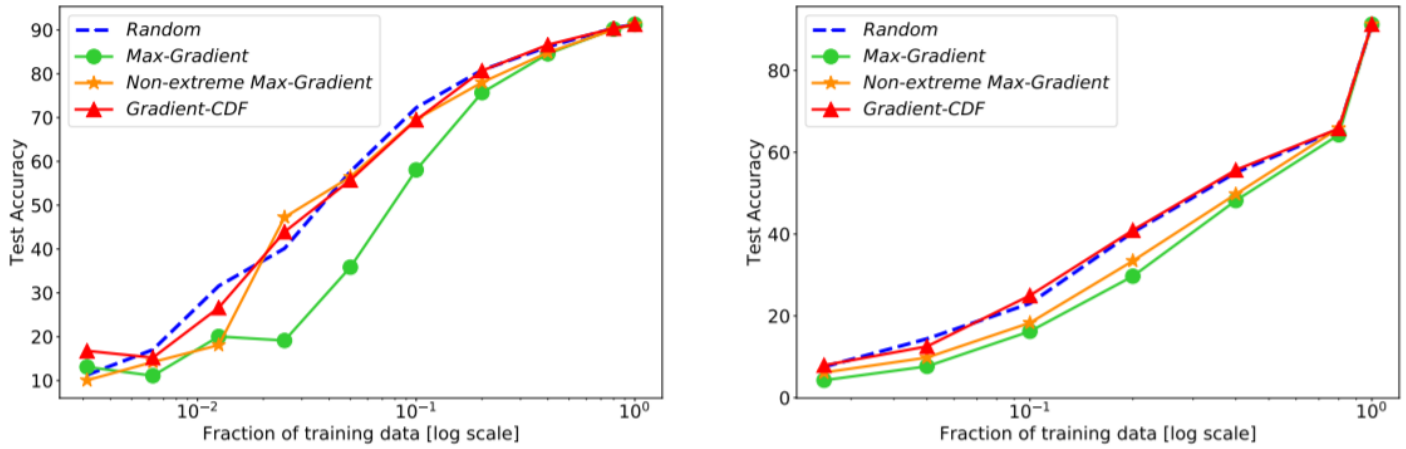


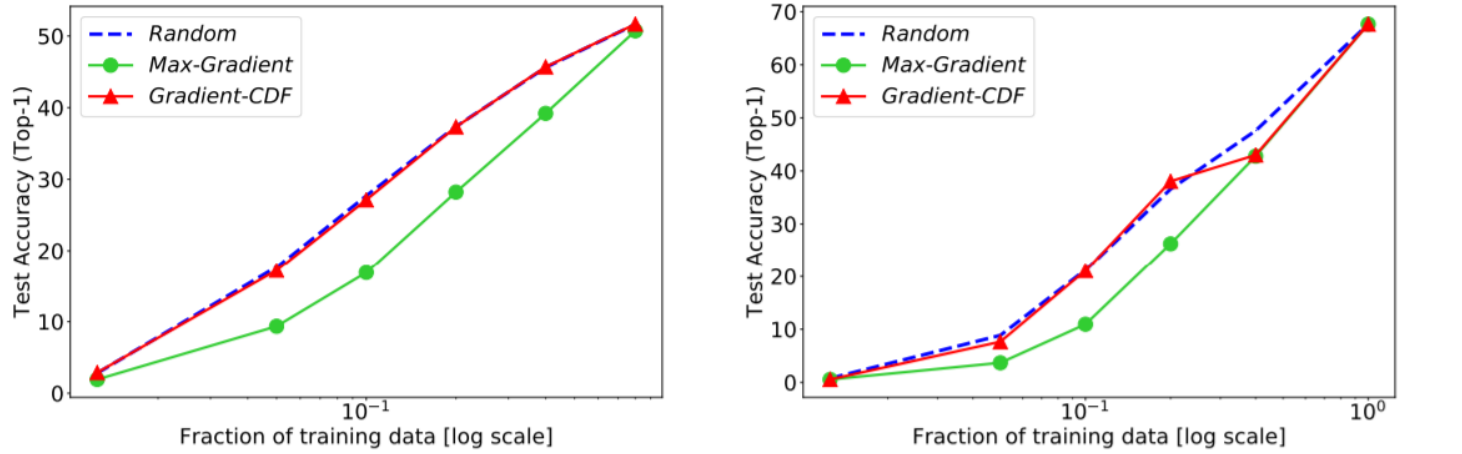

Figure 3: Top-1 test accuracy for MNIST. *Non-extreme Max-Gradient* overtakes *Random* when using 0.6% of training data. *Max-Gradient* overtakes *Random* when using 3% of training data.




(a) CIFAR-10 for VGG16 model

(b) CIFAR-100 for VGG16 model

Figure 8: Top-1 test accuracy for CIFAR-10/100 run on a VGG16 network.




(a) AlexNet; last data point uses 80% of the dataset

(b) VGG16; last data point computed with pretrained model

Figure 9: Top-1 test accuracy for ImageNet. Gradients used from sampling computed using a VGG16 network.

# Summary

- Data-Based
  - Representative
  - Diversity
- Model-Based
  - Influence for model: uncertainty, gradient
  - Committee: inconsistency