Geometric Median (GM) Matching for Robust k-Subset Selection from Noisy Data

Anish Acharya and Sujay Sanghavi and Alexandros G. Dimakis and Inderjit S Dhillon

ICML 2025

Department of Statistics, Seoul National University
Presented by Sangmoon Han

2025-11-25

Table of Contents

Overview

- **Background:** In large-scale datasets, existing data pruning - the combinatorial task of selecting a small and representative subset from a large dataset - approaches become unreliable when the data are corrupted because they depend on **empirical mean** estimation, which is extremely sensitive to outliers.
- **Objective:** To develop a data pruning method that remains robust under severe corruption while still selecting a informative subset.
- **Problem:** Existing pruning approaches rely on centroids or decision boundaries to select prototypical samples, causing them to discard non-prototypical but uncorrupted and informative examples near the decision boundary.
- **Solution:** This paper proposed method of selecting a k-subset such that the mean of the subset approximates the **geometric median** of the (potentially) noisy dataset over some appropriate **embedding space**, ensuring robustness even under arbitrary corruption.

Notation

- $p(x)$: Clean data distribution.
- $q(x)$: Adversarially chosen arbitrary distribution.
- $\psi \in [0, 1/2)$: Corruption fraction.
- $p_{\text{noisy}}(x) = (1 - \psi)p(x) + \psi q(x)$: Mixture distribution under corruption.

- D_G : Clean sample set, i.e. for $\mathbf{x}_i \in \mathbb{R}^d \overset{i.i.d}{\sim} p(x)$, $x \in D_G$.
- D_B : Corrupted sample set, i.e. for $\mathbf{x}_i \in \mathbb{R}^d \overset{i.i.d}{\sim} p_{\text{noisy}}(x)$, $x \in D_B$.
- $D = D_G \cup D_B$: the full dataset.

- $\phi(x)$: Encoder mapping raw input $x \in \mathbb{R}^d$ to an embedding space H
- $\omega_i = \phi(x_i)$: embedded feature vector

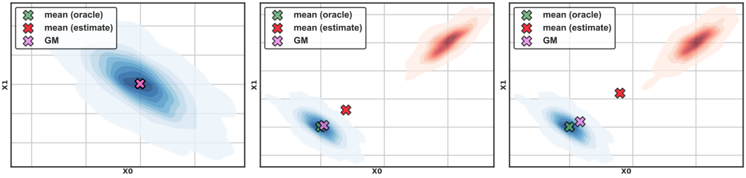
k-Subset Selection via Moment Matching

- In the uncorrupted setting i.e. when $\psi = 0$, a natural approach for data pruning is to formulate it as a combinatorial moment matching objective:

$$\underset{\substack{\mathcal{D}_{\mathcal{S}} \subseteq \mathcal{D} \\ |\mathcal{D}_{\mathcal{S}}| = k}}{\arg\min} \left[\Delta^2 (\mathcal{D}_{\mathcal{S}}, \mathcal{D}) := \| \boldsymbol{\mu}(\mathcal{D}) - \boldsymbol{\mu}(\mathcal{D}_{\mathcal{S}}) \|^2 \right] \quad (1)$$

  where $\boldsymbol{\mu}(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_i \in \mathcal{D}} \phi(\mathbf{x}_i)$, $\boldsymbol{\mu}(\mathcal{D}_{\mathcal{S}}) = \frac{1}{k} \sum_{\mathbf{x}_j \in \mathcal{D}_{\mathcal{S}}} \phi(\mathbf{x}_j)$.

- However, in the corrupted setting, the moment matching objective can result in arbitrarily poor solutions.
- The vulnerability can be attributed to the estimation of target moment via **empirical mean** - notorious for its sensitivity to outliers.



(a) **No Corruption** ($\psi = 0$) (b) **20% Corruption**($\psi = 0.2$) (c) **40% Corruption**($\psi = 0.4$)

Geometric Median and it's approximation

- Given a finite collection of observations $\{\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \ldots \phi(\mathbf{x}_n)\}$ defined over Hilbert space $\mathcal{H} \subset \mathbb{R}^d$, the Geometric Median(GM) μ^{GM} is defined as:

$$\boldsymbol{\mu}^{\mathrm{GM}} = \operatorname*{arg\,min}_{\mathbf{z} \in \mathcal{H}} \left[\rho(\mathbf{z}) := \sum_{i=1}^{n} \|\mathbf{z} - \phi(\mathbf{x}_i)\| \right] \quad (2)$$

- The gradient-based optimization of (2) is difficult since the objective involves the non-smooth term $\|\phi(\mathbf{x}_i) - \mathbf{z}\|$. And for dimensions $d \geq 2$, in general, the geometric median does not admit a closed-form solution.
- However, since the problem is **convex**, iterative algorithms can be used to approximate the geometric median efficiently. As a result, several algorithms have been proposed to compute $\boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}} \in \mathcal{H}$, which is called ϵ-accurate GM, i.e

$$\sum_{i=1}^{n} \left\| \boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}} - \phi(\mathbf{x}_i) \right\| \leq (1+\epsilon) \sum_{i=1}^{n} \left\| \boldsymbol{\mu}^{\mathrm{GM}} - \phi(\mathbf{x}_i) \right\|$$

Robust Moment Matching

- Recall, the goal of solution is to find a k-subset $\mathcal{D}_{\mathcal{S}}$ such that the empirical mean of the subset $\boldsymbol{\mu}(\mathcal{D}_{\mathcal{S}})$ approximately matches $\boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}}(\mathcal{D})$.
- Utilizing $\boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}}$, the authors aim to solve for the following objective:

$$\underset{\substack{\mathcal{D}_{\mathcal{S}} \subseteq \mathcal{D} \\ |\mathcal{D}_{\mathcal{S}}| = k}}{\arg\min} \left(\Delta_{\mathrm{GM}}^2 := \left\| \boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}}(\mathcal{D}) - \boldsymbol{\mu}(\mathcal{D}_{\mathcal{S}}) \right\|^2 \right) \quad (3)$$

  where $\boldsymbol{\mu}(\mathcal{D}_{\mathcal{S}}) = \frac{1}{k} \sum_{\mathbf{x}_j \in \mathcal{D}_{\mathcal{S}}} \phi(\mathbf{x}_j)$.
- Since the optimization problem above is combinatorial in nature, a herding-style greedy minimization procedure [3, Yutian Chen], which iteratively builds the subset by adding one sample at a time.
- This herding procedure is closely related to the Frank–Wolfe algorithm [1, Francis Bach], since it iteratively selects points that best align with the current descent direction over the convex hull of $\{\phi(x) \mid x \in \mathcal{D}\}$.
- In addition, the GM is guaranteed to lie in the relative interior of the convex hull of the majority (good) points i.e. $\boldsymbol{\mu}^{\mathrm{GM}} \in \mathrm{conv}\left(\{\phi(x) \mid x \in \mathcal{D}_{\mathcal{G}}\}\right)$ [2, Boyd], making it an attractive choice for estimating the target mean.

Proposed Algorithm

- Starting with a suitably chosen $\boldsymbol{\theta}_0 \in \mathcal{H}$; their method repeatedly performs the following updates, adding one sample at a time. For $t = 1, \cdots, k$:

$$\mathbf{x}_{t+1} := \underset{\mathbf{x} \in \mathcal{D}/\{x_1, \cdots, x_t\}}{\arg\max} \langle \boldsymbol{\theta}_t, \phi(\mathbf{x}) \rangle$$

$$\boldsymbol{\theta}_{t+1} := \boldsymbol{\theta}_t + \left(\boldsymbol{\mu}_\epsilon^{\mathrm{GM}}(\mathcal{D}) - \phi\left(\mathbf{x}_{t+1}\right) \right)$$

- If $\boldsymbol{\theta}_0 = \boldsymbol{\mu}_\epsilon^{\mathrm{GM}}$, then $\boldsymbol{\theta}_T = (T+1)\boldsymbol{\mu}_\epsilon^{\mathrm{GM}}(\mathcal{D}) - \sum_{t=1}^{T} \phi\left(\mathbf{x}_t\right)$ and

$$\mathbf{x}_{T+1} = \underset{\mathbf{x} \in \mathcal{D}}{\arg\max} \left[\left\langle \boldsymbol{\mu}_\epsilon^{\mathrm{GM}}(\mathcal{D}), \phi(\mathbf{x}) \right\rangle - \frac{1}{T+1} \sum_{t=1}^{T} \omega\left(\mathbf{x}, \mathbf{x}_t\right) \right]$$

  where $\omega\left(\mathbf{x}, \mathbf{x}'\right) = \langle \phi(\mathbf{x}), \phi\left(\mathbf{x}'\right) \rangle_{\mathcal{H}}$.

- The first term $\langle \mu_\epsilon^{\mathrm{GM}}(\mathcal{D}),\ \phi(x) \rangle$ encourages selecting a point whose feature embedding is well aligned with the target robust center $\mu_\epsilon^{\mathrm{GM}}(\mathcal{D})$.

- The second term $-\frac{1}{T+1} \sum_{t=1}^{T} \omega(x, x_t)$ penalizes points that are too similar to the already selected samples, preventing redundant selections.

Proposed Algorithm

Algorithm 1 GEOMETRIC MEDIAN (GM) MATCHING

(initialization)
A finite collection of grossly corrupted (Definition 1) observations $\mathcal{D} = \{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^n$; pretrained encoder $\phi(\cdot) : \mathbb{R}^d \to \mathbb{R}^s$ e.g. CLIP (Radford et al., 2021b); initial weight vector $\boldsymbol{\theta}_0 \in \mathbb{R}^s$; number of sampling batches B, population fraction for GM computation $0 < \gamma_{\text{GM}} \leq 1$.

(compute embeddings)
$\Phi = \{\omega_i = \phi(\mathbf{x}_i) \in \mathbb{R}^s : \forall \mathbf{x}_i \in \mathcal{D}\}$
(pick random n_{GM}-subset for GM computation)
$\Phi_{\text{GM}} \overset{i.i.d}{\sim} \Phi$, where, $n_{\text{GM}} = |\Phi_{\text{GM}}| = \gamma_{\text{GM}}|\Phi| \leq n$
(compute ϵ-approximate geometric median via Algorithm 2)
$\boldsymbol{\mu}_\epsilon^{\text{GM}}(\Phi_{\text{GM}}) = \arg\min_{\mathbf{z}\in\mathbb{R}^s} \sum_{\omega_i \in \Phi_{\text{GM}}} \|\mathbf{z} - \omega_i\|$
(partition data into batches)
$\mathcal{D} = \bigcup_{b=1}^{B} \mathcal{D}_b$
(initialize subset)
$\mathcal{D}_\mathcal{S} \leftarrow \emptyset$
for *batch index* $b = 1, \dots B$ **do**
  (load batch embeddings)
  $\Phi_b = \{\omega_i \in \Phi : \mathbf{x}_i \in \mathcal{D}_b\}$
  for *iterations* $t = 0, 1, \dots, k/B$ **do**
    (find embedding closest to $\boldsymbol{\theta}_t$)
    $\omega := \arg\max_{\omega_i \in \Phi_b} \langle \boldsymbol{\theta}_t, \omega_i \rangle$
    (update direction parameter)
    $$\boldsymbol{\theta}_{t+1} := \boldsymbol{\theta}_t + \left[\boldsymbol{\mu}_\epsilon^{\text{GM}}(\Phi_b) - \omega\right]$$
    (update selected subset)
    $\mathcal{D}_\mathcal{S} := \mathcal{D}_\mathcal{S} \cup \mathbf{x}$ where, $\omega = \phi(\mathbf{x})$
    (update the batch embedding set)
    $\Phi_b := \Phi_b \setminus \omega$
  end
end
return: $\mathcal{D}_\mathcal{S}$

Experiments

- Experiments are divided into two fundamental learning paradigms:
  - Image Classification
  - Image Generation

Method / Ratio	20%	30%	40%	60%	80%	100%	Mean ↑
CIFAR-100							
Random	50.26±3.24	53.61±2.73	64.32±1.77	71.03±0.75	74.12±0.56	78.14±0.55	62.67
Herding	48.39±1.42	50.89±0.97	62.99±0.61	70.61±0.44	74.21±0.49	78.14±0.55	61.42
Forgetting	35.57±1.40	49.83±0.91	59.65±2.50	**73.34±0.39**	**77.50±0.53**	78.14±0.55	59.18
GraNd-score	42.65±1.39	53.14±1.28	60.52±0.79	69.70±0.68	74.67±0.79	78.14±0.55	60.14
EL2N-score	27.32±1.16	41.98±0.54	50.47±1.20	69.23±1.00	75.96±0.88	78.14±0.55	52.99
Optimization-based	42.16±3.30	53.19±2.14	58.93±0.98	68.93±0.70	75.62±0.33	78.14±0.55	59.77
Self-sup.-selection	44.45±2.51	54.63±2.10	62.91±1.20	70.70±0.82	75.29±0.45	78.14±0.55	61.60
Moderate-DS	51.83±0.52	57.79±1.61	64.92±0.93	71.87±0.91	75.44±0.40	78.14±0.55	64.37
GM Matching	**55.93± 0.48**	**63.08± 0.57**	**66.59± 1.18**	70.82± 0.59	74.63± 0.86	78.14± 0.55	**66.01**
Tiny ImageNet							
Random	24.02±0.41	29.79±0.27	34.41±0.46	40.96±0.47	45.74±0.61	49.36±0.25	34.98
Herding	24.09±0.45	29.39±0.53	34.13±0.37	40.86±0.61	45.45±0.33	49.36±0.25	34.78
Forgetting	22.37±0.71	28.67±0.54	33.64±0.32	41.14±0.43	**46.77±0.31**	49.36±0.25	34.52
GraNd-score	23.56±0.52	29.66±0.37	34.33±0.50	40.77±0.42	45.96±0.56	49.36±0.25	34.86
EL2N-score	19.74±0.26	26.58±0.40	31.93±0.28	39.12±0.46	45.32±0.27	49.36±0.25	32.54
Optimization-based	13.88±2.17	23.75±1.62	29.77±0.94	37.05±2.81	43.76±1.50	49.36±0.25	29.64
Self-sup.-selection	20.89±0.42	27.66±0.50	32.50±0.30	39.64±0.39	44.94±0.34	49.36±0.25	33.13
Moderate-DS	25.29±0.38	30.57±0.20	34.81±0.51	41.45±0.44	46.06±0.33	49.36±0.25	35.64
GM Matching	**27.88±0.19**	**33.15±0.26**	**36.92±0.40**	**42.48±0.12**	46.75±0.51	49.36±0.25	**37.44**

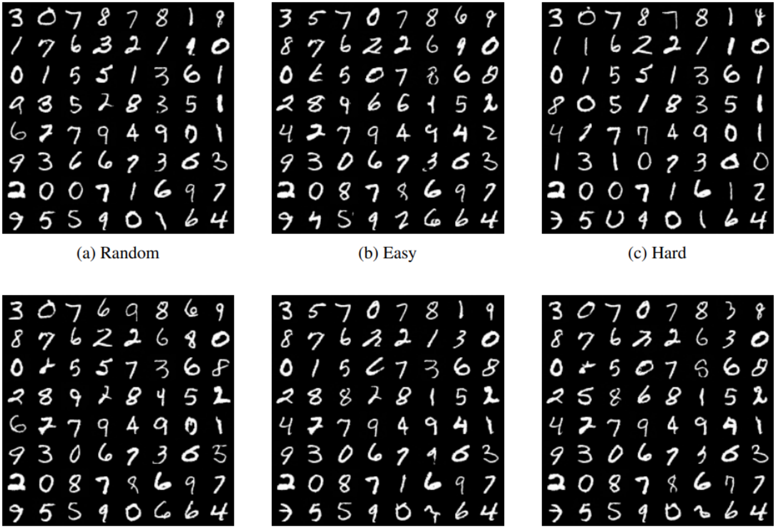
Table 1: (CLEAN) IMAGE CLASSIFICATION: Comparing Downstream Test Accuracy (Top-1) (%) of several pruning algorithms (Section 6.1) on CIFAR-100 and Tiny-ImageNet in the uncorrupted setting. ResNet-50 is used both as proxy (pretrained) and for downstream classification.

Experiments

CIFAR-100							
Method / Ratio	**20%**	**30%**	**40%**	**60%**	**80%**	**100%**	**Mean ↑**
			5% Feature Corruption				
Random	43.14±3.04	54.19±2.92	64.21±2.39	69.50±1.06	72.90±0.52	77.26±0.39	60.79
Herding	42.50±1.27	53.88±3.07	60.54±0.94	69.15±0.55	73.47±0.89	77.26±0.39	59.81
Forgetting	32.42±0.74	49.72±1.64	54.84±2.20	70.22±2.00	75.19±0.40	77.26±0.39	56.48
GraNd-score	42.24±0.57	53.48±0.76	60.17±1.66	69.16±0.81	73.35±0.81	77.26±0.39	59.68
EL2N-score	26.13±1.75	39.01±1.42	49.89±1.87	68.36±1.41	73.10±0.36	77.26±0.39	51.30
Optimization-based	38.25±3.04	50.88±6.07	57.26±0.93	68.02±0.39	73.77±0.56	77.26±0.39	57.64
Self-sup.-selection	44.24±0.48	55.99±1.21	61.03±0.59	69.96±1.07	74.56±1.17	77.26±0.39	61.16
Moderate-DS	46.78±1.90	57.36±1.22	65.40±1.19	71.46±0.19	**75.64±0.61**	77.26±0.39	63.33
GM Matching	**49.50±0.72**	**60.23±0.88**	**66.25±0.51**	**72.91±0.26**	75.10±0.29	77.26±0.39	**64.80**
			10% Feature Corruption				
Random	43.27±3.01	53.94±2.78	62.17±1.29	68.41±1.21	73.50±0.73	76.50±0.63	60.26
Herding	44.34±1.07	53.31±1.49	60.13±0.38	68.20±0.74	74.34±1.07	76.50±0.63	60.06
Forgetting	30.43±0.70	47.50±1.43	53.16±0.44	70.36±0.82	75.11±0.71	76.50±0.63	55.31
GraNd-score	36.36±1.06	52.26±0.66	60.22±1.39	68.96±0.62	72.78±0.51	76.50±0.63	58.12
EL2N-score	21.75±1.56	30.80±2.23	41.06±1.23	64.82±1.48	73.47±1.30	76.50±0.63	46.38
Optimization-based	37.22±0.39	48.92±1.38	56.88±1.48	67.33±2.15	72.94±1.90	76.50±0.63	56.68
Self-sup.-selection	42.01±1.31	54.47±1.19	61.37±0.68	68.52±1.24	74.73±0.36	76.50±0.63	60.22
Moderate-DS	47.02±0.66	55.60±1.67	62.18±1.86	71.83±0.78	**75.66±0.66**	76.50±0.63	62.46
GM Matching	**48.86±1.02**	**60.15±0.43**	**66.92±0.28**	**72.03±0.38**	73.71±0.19	76.50±0.63	**64.33**
			20% Feature Corruption				
Random	40.99±1.46	50.38±1.39	57.24±0.65	65.21±1.31	71.74±0.28	74.92±0.88	57.11
Herding	44.42±0.46	53.57±0.31	60.72±1.78	69.09±1.73	73.08±0.98	74.92±0.88	60.18
Forgetting	26.39±0.17	40.78±2.02	49.95±2.31	65.71±1.12	73.67±1.12	74.92±0.88	51.30
GraNd-score	36.33±2.66	46.21±1.48	55.51±0.76	64.59±2.40	70.14±1.36	74.92±0.88	54.56
EL2N-score	21.64±2.03	23.78±1.66	35.71±1.17	56.32±0.86	69.66±0.43	74.92±0.88	41.42
Optimization-based	33.42±1.60	45.37±2.81	54.06±1.74	65.19±1.27	70.06±0.83	74.92±0.88	54.42
Self-sup.-selection	42.61±2.44	54.04±1.90	59.51±1.22	68.97±0.96	72.33±0.20	74.92±0.88	60.01
Moderate-DS	42.98±0.87	55.80±0.95	61.84±1.96	70.05±1.29	73.67±0.30	74.92±0.88	60.87
GM Matching	**47.12±0.64**	**59.17±0.92**	**63.45±0.34**	**71.70±0.60**	**74.60±1.03**	74.92±0.88	**63.21**

Table 3: **(FEATURE CORRUPTION) IMAGE CLASSIFICATION (CIFAR 100):** Comparing the downstream test accuracy of various pruning methods when 5%, 10%, and 20% of images are corrupted. Results are reported across different selection ratios (20%-100%) using **ResNet-50** as both the proxy and downstream classifier. GM Matching consistently outperforms all baselines, demonstrating superior robustness to corrupted data, with increasing performance gains at higher corruption levels.

Experiments

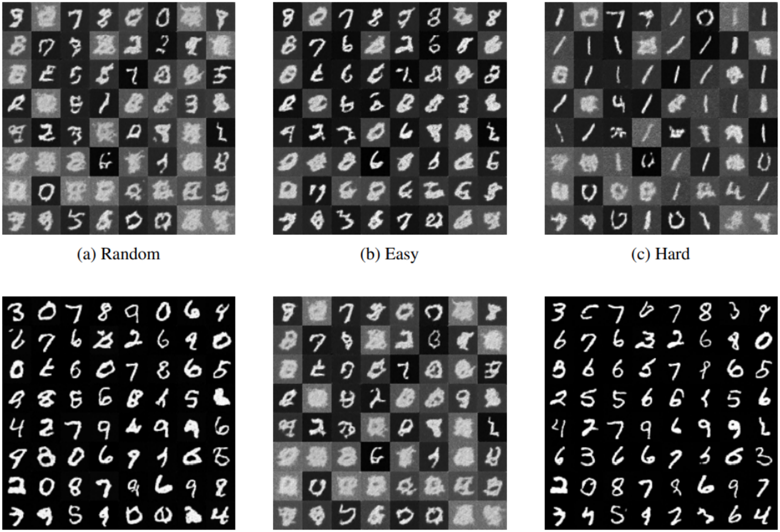


(a) Random (b) Easy (c) Hard

(d) Moderate (e) Herding (f) GM Matching

Figure 15: (No Corruption) Visualization of Generated Samples 40% sampling

Experiments



(a) Random (b) Easy (c) Hard

(d) Moderate (e) Herding (f) GM Matching

Figure 16: (30% Corruption) Visualization of Generated Samples 40% sampling

Appendix

- **Corruption method list**

  1. Gaussian Noise: Adding random perturbations sampled from a standard normal distribution.
  2. Random Occlusion: Mimics missing or occluded regions in images by replacing random patches with black or noisy pixels.
  3. Resolution Reduction: Simulates low-quality images by applying aggressive down-sampling and up-sampling, introducing pixelation artifacts.
  4. Fog: Emulates atmospheric distortions by overlaying a simulated fog effect.
  5. Motion Blur: Models dynamic distortions caused by camera motion or moving objects during exposure.

- **Pruning method list**

  1. Easy: This strategy selects samples that are closest to the centroid of the dataset. These "easy" samples are presumed to be representative of the core data distribution.
  2. Hard: This approach selects samples that are farthest from the centroid.
  3. Moderate: This strategy selects samples that are closest to the median distance from the centroid.
  4. Kernel Herding: kernel herding employs a greedy algorithm to select samples that minimize the discrepancy between the empirical distribution and the target distribution.

References I

Francis Bach, Simon Lacoste-Julien, and Guillaume Obozinski. *On the Equivalence between Herding and Conditional Gradient Algorithms*. 2012. arXiv: 1203.4523 [cs.LG]. URL: https://arxiv.org/abs/1203.4523.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Yutian Chen, Max Welling, and Alex Smola. *Super-Samples from Kernel Herding*. 2012. arXiv: 1203.3472 [cs.LG]. URL: https://arxiv.org/abs/1203.3472.