Geometric Median (GM) Matching for Robust Data Pruning

Anish Acharya [1] **Inderjit S Dhillon** [1 2] **Sujay Sanghavi** [1 3]

Abstract

Large-scale data collections in the wild, are invariably noisy. Thus developing data pruning strategies that remain robust even in the presence of corruption is critical in practice. In this work, we propose Geometric Median (GM) Matching – a herding style greedy algorithm that yields a k-subset such that the mean of the subset approximates the geometric median of the (potentially) noisy dataset. Theoretically, we show that GM Matching enjoys an improved $\mathcal{O}(1/k)$ scaling over $\mathcal{O}(1/\sqrt{k})$ scaling of uniform sampling; while achieving **optimal breakdown point** of **1/2** even under **arbitrary** corruption. Extensive experiments across several popular deep learning benchmarks indicate that GM Matching consistently improves over prior state-of-the-art; the gains become more profound at high rates of corruption and aggressive pruning rates; making GM Matching a strong baseline for future research in robust data pruning.

1. Background

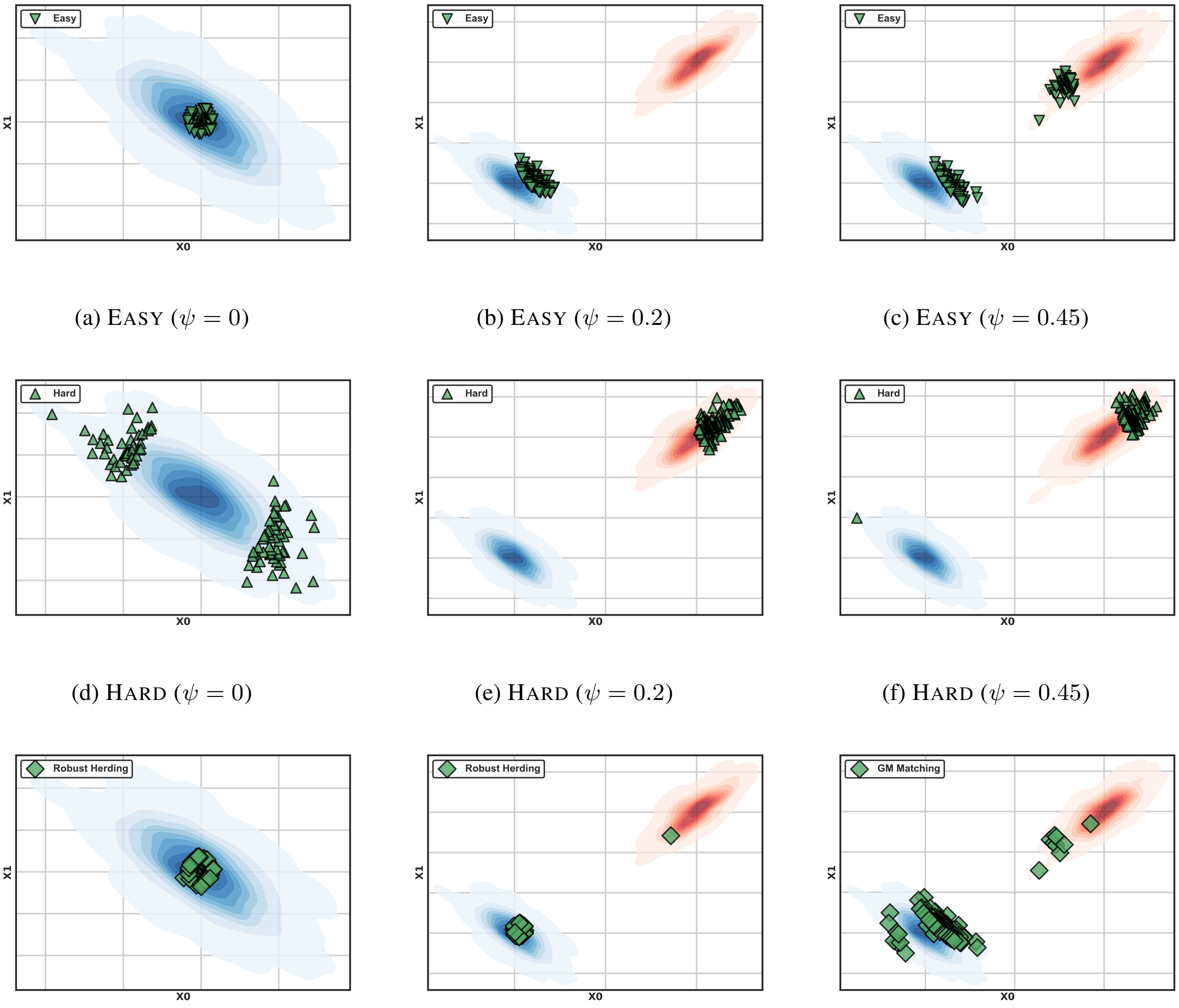
Data pruning, the (combinatorial) task of downsizing a large training set into a small informative subset (Feldman, 2020; Agarwal et al., 2005; Muthukrishnan et al., 2005; Har-Peled, 2011; Feldman and Langberg, 2011), is a promising approach for reducing the enormous computational and storage costs of modern deep learning. Consequently, a large body of recent works have been proposed to solve the data selection problem. At a high level, data pruning approaches rely on some carefully designed **pruning metrics** and rank the training samples based on the scores and retain a fraction of them as representative samples (super samples), used for training the downstream model. For example, (Xia et al., 2022; Joshi and Mirzasoleiman, 2023; Sorscher et al., 2022) calculate the importance score of a sample in terms of the distance from the centroid of its corresponding class marginal. Samples closer to the centroid are considered most prototypical (easy) and those far from the centroid are treated as least prototypical (hard). Canonically, similar scoring criterion have been developed in terms of gradients (Paul et al., 2021), uncertainty (Pleiss et al., 2020), forgetfulness (Toneva et al., 2018). It is worth noting that the distance-based score is closely related to the uncertainty / gradient forgetting based score. Samples close (far away) to the class centroid are often associated with smaller (larger) gradient norm ; lower (higher) uncertainty; harder (easier) to forget (Paul et al., 2021; Sorscher et al., 2022; Xia et al., 2022).

Robustness vs Diversity : In the **ideal scenario** (i.e. in absence of any corruption), hard examples are known to contribute the most in downstream generalization performance (Katharopoulos and Fleuret, 2018; Joshi et al., 2009; Huang et al., 2010; Balcan et al., 2007) as they often capture most of the usable information in the dataset (Xu et al., 2020). On the other hand, in **realistic noisy scenarios** involving outliers, this strategy often fails since the noisy examples are wrongly deemed informative for training (Zhang and Sabuncu, 2018; Park et al., 2024). Pruning methods specifically designed for such noisy scenarios thus propose to retain the most representative (easy) samples (Pleiss et al., 2020; Jiang et al., 2018; Har-Peled et al., 2006; Shah et al., 2020; Shen and Sanghavi, 2019). However, by only choosing samples far from the decision boundary, these methods ignore the more informative uncorrupted less prototypical samples. This can often result in sub-optimal downstream performance and in fact can also lead to degenerate solutions due to a covariance-shift problem (Sugiyama and Kawanabe, 2012); giving rise to a *robustness vs diversity trade off* (Xia et al., 2022). This issue restricts the applicability of these methods, as realistic scenarios often deviate from expected conditions, making it challenging or impractical to adjust the criteria and methods accordingly. To go beyond these limitations, in this work, we consider the setting where a fraction of the samples can be **arbitrarily corrupted**.

Definition 1 (α**-corrupted generation process).** *Given $\psi \in [0, \frac{1}{2})$ and a set of observations from the original distribution of interest, an adversary is allowed to* ***inspect*** *all the samples and* ***arbitrarily*** *perturb up to ψ fraction of them. We refer to a set of samples $\mathcal{D} = \mathcal{D}_{\mathcal{G}} \cup \mathcal{D}_{\mathcal{B}}$ generated through this process as α-corrupted where, $\alpha :=$*

[1] UT Austin [2] Google [3] Amazon. Correspondence to: Anish Acharya <anishacharya@utexas.edu>.

Published at ICML 2024 Workshop on Foundation Models in the Wild. Copyright 2024 by the author(s).



(a) Easy ($\psi = 0$)

(b) Easy ($\psi = 0.2$)

(c) Easy ($\psi = 0.45$)

(d) Hard ($\psi = 0$)

(e) Hard ($\psi = 0.2$)

(f) Hard ($\psi = 0.45$)

(g) GM Matching ($\psi = 0$)

(h) GM Matching ($\psi = 0.2$)

(i) GM Matching ($\psi = 0.45$)

Figure 1: **Toy Example: 0/20/45% of the samples are corrupted** i.e. drawn from an adversary chosen distribution (red). We compare several baselines for choosing 10% samples: (Uniform) random sampling, (Easy) selects of samples closest to the centroid. (Hard) Selection of samples farthest from the centroid. (Moderate) selects samples closest to the median distance from the centroid. (Herding) moment matching, (GM Matching) robust moment (GM) matching. Clearly GM Matching is significantly more robust and diverse than the other approaches even at such high corruption rates.

$|\mathcal{D}_{\mathcal{B}}|/|\mathcal{D}_{\mathcal{G}}| = \frac{\psi}{1-\psi} < 1$. *$\mathcal{D}_{\mathcal{B}}$ and $\mathcal{D}_{\mathcal{G}}$ denote the sets of corrupt and clean samples.*

Given such an α-corrupted [1] set of observations $\mathcal{D} = \mathcal{D}_{\mathcal{G}} \cup \mathcal{D}_{\mathcal{B}} = \{(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathcal{Y}\}_{i=1}^{N}$, the goal of **robust data pruning** is thus to judiciously select a subset $\mathcal{D}_{\mathcal{S}} \subseteq \mathcal{D}$; that *encapsulates the comprehensive statistical characteristics of the underlying clean (uncorrupted) distribution induced by subset $\mathcal{D}_{\mathcal{G}}$ without any a-priori knowledge about the corrupted samples*. Note that, allowing the noisy samples to be **arbitrarily corrupted** enables us to generalize many important robustness scenarios; including **corrupt features, label noise** and **adversarial attacks**.

In response, we develop a selection strategy that aims to find subset such that the discrepancy between the mean of subset and the geometric median (GM)(Definition 2) of the (potentially noisy) dataset is minimized. We call this algorithm Geometric Median Matching and describe it in Section 2. We show that GM Matching is guaranteed to converge to a neighborhood of the underlying true (uncorrupted) mean of the (corrupted) dataset even when $1/2$ fraction of the samples are arbitrarily corrupted.

Theoretically, we show that, GM Matching converges to a neighborhood of original underlying mean, at an impressive $\mathcal{O}(1/k)$ rate while being robust.

2. Geometric Median (GM) Matching

We make a **key** observation that in presence of arbitrary corruption, the class center itself can be arbitrarily shifted and in fact need not even lie in the convex hull of the underlying clean samples. That is to say, in the arbitrary corruption scenario, the entire notion of easy (robust) / hard based on heuristic falls apart. We exploit the breakdown and translation invariance property of Geometric Median (GM) (Definition 2) – a well studied spatial estimator, known for several nice properties like **rotation and translation invariance** and **optimal breakdown point of 1/2 under gross corruption** (Minsker et al., 2015; Kemperman, 1987). to perform subset selection while being resilient to arbitrary corruption.

Algorithm 1 GM MATCHING

```
initialize : A finite collection of $\alpha$ corrupted ( Definition 1) observations $\mathcal{D}$ defined over Hilbert space $\mathcal{H} \in \mathbb{R}^d$, equipped with norm $\|\cdot\|$ and inner $\langle\cdot\rangle$ operators; $\theta_0, \mathcal{S} \leftarrow \emptyset$.

$\boldsymbol{\mu}^{\text{GM}} = \arg\min_{\mathbf{z}\in\mathcal{H}} \sum_{\mathbf{x}_i\in\mathcal{D}} \|\mathbf{z} - \mathbf{x}_i\|$
for iterations t = 0, 1, . . . , k-1 do
    $\mathbf{x}_{t+1} := \arg\max_{\mathbf{x}\in\mathcal{D}} \langle\theta_t, \mathbf{x}\rangle$
    $\theta_{t+1} := \theta_t + \boldsymbol{\mu}_{\epsilon}^{\text{GM}} - \mathbf{x}_{t+1}$
    $\mathcal{S} := \mathcal{S} \cup \mathbf{x}_{t+1}$
    $\mathcal{D} := \mathcal{D} \setminus \mathbf{x}_{t+1}$
end
return: $\mathcal{S}$
```

Definition 2 (Geometric Median). *Given a finite collection of observations $\{\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_n\}$ defined over Hilbert space $\mathcal{H} \in \mathbb{R}^d$, equipped with norm $\|\cdot\|$ and inner $\langle\cdot\rangle$ operators, the geometric median(or Fermat-Weber point) (Weber et al., 1929) is defined as:*

$$\mathbf{x}_* = \arg\min_{\mathbf{z}\in\mathcal{H}} \left[\rho(\mathbf{z}) := \sum_{i=1}^{n} \left\|\mathbf{z} - \mathbf{x}_i\right\|\right] \tag{1}$$

Computing the GM exactly is known to be hard and no linear time algorithm exists (Bajaj, 1988). Consequently, it is necessary to rely on approximation methods to estimate the geometric median (Weiszfeld, 1937; Vardi and Zhang, 2000; Cohen et al., 2016). We call a point $\mathbf{x}_\epsilon \in \mathcal{H}$ an **ϵ-accurate geometric median** if:

$$\sum_{i=1}^{n} \left\|\mathbf{x}_\epsilon - \mathbf{x}_i\right\| \leq (1+\epsilon) \sum_{i=1}^{n} \left\|\mathbf{x}_* - \mathbf{x}_i\right\| \tag{2}$$

Given a random batch of samples $\mathcal{D}$ from an α corrupted dataset (Definition 1), access to an ϵ accurate GM$(\cdot)$ oracle; we aim to find a k-subset of samples such that the mean of the selected subset approximately matches the geometric median of the training dataset, i.e. we aim solve:

$$\arg\min_{\mathcal{D}_{\mathcal{S}}\subseteq\mathcal{D}, |\mathcal{D}_{\mathcal{S}}|=k} \left\|\boldsymbol{\mu}_{\epsilon}^{\text{GM}} - \frac{1}{k}\sum_{\mathbf{x}_i\in\mathcal{S}} \mathbf{x}_i\right\|^2 \tag{3}$$

where, $\boldsymbol{\mu}_{\epsilon}^{\text{GM}}$ denotes the ϵ-approximate GM (2) of training dataset $\mathcal{D}$. Intuitively, if such a subset can be found, the expected cost (e.g. loss / average gradient) over $\mathcal{S}$ is guaranteed to be close (as characterized in Theorem 1) to the expected cost over the uncorrupted examples $\mathcal{D}_{\mathcal{G}}$; enabling robust data pruning. This is because, even in presence of grossly corrupted samples (unbounded), GM remains bounded w.r.t the underlying true mean (Acharya et al., 2022; Minsker et al., 2015; Cohen et al., 2016; Wu et al., 2020; Chen et al., 2017).

Noting that, the proposed **robust moment matching objective** (3) is an instance of the famous set function maximization problem – known to be NP hard via a reduction from k-set cover (Feige, 1998; Nemhauser et al., 1978); we adopt the iterative, herding (Welling, 2009; Chen et al., 2010) style greedy approach to approximately solve (3):

[1] This strong corruption model is also referred to as the Gross Contamination Framework (Diakonikolas and Kane, 2019), that generalizes the popular Huber Contamination Model (Huber, 1992) and Byzantine Corruption Framework (Lamport et al., 1982).

	ResNet-50→ VGG-16		**ResNet-50→ ShuffleNet**		
Method / Ratio	20%	30%	20%	30%	**Mean ↑**
		No Corruption			
Random	29.63±0.43	35.38±0.83	32.40±1.06	39.13±0.81	34.96
Herding	31.05±0.22	36.27±0.57	33.10±0.39	38.65±0.22	35.06
Forgetting	27.53±0.36	35.61±0.39	27.82±0.56	36.26±0.51	32.35
GraNd-score	29.93±0.95	35.61±0.39	29.56±0.46	37.40±0.38	33.34
EL2N-score	26.47±0.31	33.19±0.51	28.18±0.27	35.81±0.29	31.13
Optimization-based	25.92±0.64	34.82±1.29	31.37±1.14	38.22±0.78	32.55
Self-sup.-selection	25.16±1.10	33.30±0.94	29.47±0.56	36.68±0.36	31.45
Moderate-DS	31.45±0.32	37.89±0.36	33.32±0.41	39.68±0.34	35.62
GM Matching	**35.86±0.41**	**40.56±0.22**	**35.51±0.32**	**40.30±0.58**	**38.47**
		20% Label Corruption			
Random	23.29±1.12	28.18±1.84	25.08±1.32	31.44±1.21	27.00
Herding	23.99±0.36	28.57±0.40	26.25±0.47	30.73±0.28	27.39
Forgetting	14.52±0.66	21.75±0.23	15.70±0.29	22.31±0.35	18.57
GraNd-score	22.44±0.46	27.95±0.29	23.64±0.10	30.85±0.21	26.22
EL2N-score	15.15±1.25	23.36±0.30	18.01±0.44	24.68±0.34	20.30
Optimization-based	22.93±0.58	24.92±2.50	25.82±1.70	30.19±0.48	25.97
Self-sup.-selection	18.39±1.30	25.77±0.87	22.87±0.54	29.80±0.36	24.21
Moderate-DS	23.68±0.19	28.93±0.19	28.82±0.33	32.39±0.21	28.46
GM Matching	**28.77±0.77**	**34.87±0.23**	**32.05±0.93**	**37.43±0.25**	**33.28**
		20% Feature Corruption			
Random	26.33±0.88	31.57±1.31	29.15±0.83	34.72±1.00	30.44
Herding	18.03±0.33	25.77±0.34	23.33±0.43	31.73±0.38	24.72
Forgetting	19.41±0.57	28.35±0.16	18.44±0.57	31.09±0.61	24.32
GraNd-score	23.59±0.19	30.69±0.13	23.15±0.56	31.58±0.95	27.25
EL2N-score	24.60±0.81	31.49±0.33	26.62±0.34	33.91±0.56	29.16
Optimization-based	25.12±0.34	30.52±0.89	28.87±1.25	34.08±1.92	29.65
Self-sup.-selection	26.33±0.21	33.23±0.26	26.48±0.37	33.54±0.46	29.90
Moderate-DS	29.65±0.68	35.89±0.53	32.30±0.38	38.66±0.29	34.13
GM Matching	**33.45±1.02**	**39.46±0.44**	**35.14±0.21**	**39.89±0.98**	**36.99**
		PGD Attack			
Random	26.12±1.09	31.98±0.78	28.28±0.90	34.59±1.18	30.24
Herding	26.76±0.59	32.56±0.35	28.87±0.48	35.43±0.22	30.91
Forgetting	24.55±0.57	31.83±0.36	23.32±0.37	31.82±0.15	27.88
GraNd-score	25.19±0.33	31.46±0.54	26.03±0.66	33.22±0.24	28.98
EL2N-score	21.73±0.47	27.66±0.32	22.66±0.35	29.89±0.64	25.49
Optimization-based	26.02±0.36	31.64±1.75	27.93±0.47	34.82±0.96	30.10
Self-sup.-selection	22.36±0.30	28.56±0.50	25.35±0.27	32.57±0.13	27.21
Moderate-DS	27.24±0.36	32.90±0.31	29.06±0.28	35.89±0.53	31.27
GM Matching	**27.96±1.60**	**35.76±0.82**	**34.11±0.65**	**40.91±0.84**	**34.69**

Table 1: **Tiny ImageNet** : A ResNet-50 proxy (pretrained on TinyImageNet) is used to find important samples from Tiny-ImageNet; which is then used to train a VGGNet-16 and ShuffleNet. We repeat the experiment across multiple corruption settings - clean; 20% Feature / Label Corruption and PGD attack when 20% and 30% samples are selected.

We start with a suitably chosen $\theta_0 \in \mathbb{R}^d$; and repeatedly perform the following updates, adding one sample at a time, k times:

$$\mathbf{x}_{t+1} := \arg\max_{\mathbf{x}\in\mathcal{D}} \langle \theta_t, \mathbf{x} \rangle \tag{4}$$

$$\theta_{t+1} := \theta_t + \left(\boldsymbol{\mu}_{\epsilon}^{\text{GM}} - \mathbf{x}_{t+1} \right) \tag{5}$$

Clearly, GM MATCHING is performing greedy minimization of the error (3). Intuitively, at each iteration, it accumulates the discrepancies between the GM of dataset and empirical mean of the chosen samples. By pointing towards the direction that reduces the accumulated error, θ_t guides the algorithm to explore underrepresented regions of the feature space, thus **promoting diversity**. Moreover, by matching **GM (robust moment)**, the algorithm ensures that far-away points (outliers) suffer larger penalty – consequently, encouraging GM MATCHING to choose more samples near the convex hull of uncorrupted points $\text{conv}\{\phi_{\mathcal{B}}(\mathbf{x}) | \mathbf{x} \in \mathcal{D}_{\mathcal{G}}\}$. Theoretically, we can establish the following convergence guarantee for GM MATCHING :

Theorem 1. *Suppose that, we are given, a set of α-corrupted samples $\mathcal{D}$ (Definition 1), pretrained proxy model $\phi_{\mathbf{B}}$, and an ϵ approx.* GM$(\cdot)$ *oracle* (1). *Then,* GM MATCHING *guarantees that the mean of the selected k-subset $\boldsymbol{\mu}^{\mathcal{S}} = \frac{1}{k}\sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{S}}} \mathbf{x}_i$ converges to the neighborhood of $\boldsymbol{\mu}^{\mathcal{G}} = \mathbb{E}_{\mathbf{x}\in\mathcal{D}_{\mathcal{G}}}(\mathbf{x})$ at the rate $\mathcal{O}(\frac{1}{k})$ such that:*

$$\mathbb{E}\left\|\boldsymbol{\mu}^{\mathcal{S}} - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 \leq \frac{8|\mathcal{D}_{\mathcal{G}}|}{(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|)^2} \sum_{\mathbf{x}\in\mathcal{D}_{\mathcal{G}}} \mathbb{E}\left\|\mathbf{x} - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 + \frac{2\epsilon^2}{(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|)^2} \tag{6}$$

This result suggest that, even in presence of α corruption, the proposed algorithm GM Matching converges to a neighborhood of the true mean, where the neighborhood radius depends on two terms – the first term depend on the variance of the uncorrupted samples and the second term depends on how accurately the GM is calculated. Consequently, we say that GM Matching achieves the optimal breakdown point.

2.1. Empirical Evidence

To ensure reproducibility, our experimental setup is identical to (Xia et al., 2022). Spanning across three popular image classification datasets - CIFAR10, CIFAR100 and Tiny-ImageNet - and popular deep nets including ResNet-18/50 (He et al., 2016), VGG-16 (Simonyan and Zisserman, 2014), ShuffleNet (Ma et al., 2018a), SENet (Hu et al., 2018), EfficientNet-B0(Tan and Le, 2019); we compare GM MATCHING against several popular data pruning strategies as baselines: (1) Random; (2) Herding (Welling, 2009); (3) Forgetting (Toneva et al., 2018); (4) GraNd-score (Paul et al., 2021); (5) EL2N-score (Paul et al., 2021); (6) Optimization-based (Yang et al., 2022); (7) Self-sup.-selection (Sorscher et al., 2022) and (8) Moderate (Xia et al., 2022). We do not run these baselines for be these baselines are borrowed from (Xia et al., 2020).

We consider three corruption scenarios: **(1) Image Corruption** : a popular robustness setting, often encountered when training models on real-world data (Hendrycks and Dietterich, 2019; Szegedy et al., 2013). (2)**Label Noise :** data in the wild always contains noisy annotations (Li et al., 2022; Patrini et al., 2017; Xia et al., 2020). **(3) Adversarial Attack :** Imperceptible but adversarial noise on natural examples e.g. PGD attack (Madry et al., 2017) and GS Attacks (Goodfellow et al., 2014).

Overall, we improve over prior work almost in all settings, the gains are especially more profound in presence of corruption and at aggressive pruning rates. Thus, making GM Matching a strong baseline for future research in robust data pruning. Due to space constraints we defer the experiments and discussion to Appendix while providing a glimpse in Figure 1, Table 9.

3. Conclusion

In this work, we formalized and studied the problem of robust data pruning. We show that existing data pruning strategies suffer significant degradation in performance in presence of corruption. Orthogonal to existing works, we propose GM MATCHING where our goal is to find a k-subset from the noisy data such that the mean of the subset approximates the GM of the noisy dataset. We solve this meta problem using a herding style greedy approach. We theoretically justify our approach and empirically show its efficacy by comparing GM MATCHING against several popular benchmarks across multiple datasets. Our results indicate that GM MATCHING consistently outperforms existing pruning strategies in both clean and noisy settings. While in this work we have only explored greedy herding style approach, it is possible to investigate other combinatorial approaches to solve the meta problem. Further, while we only studied the problem under gross corruption framework, it remains open to improve the results by incorporating certain structural assumptions.

References

Acharya, A., Hashemi, A., Jain, P., Sanghavi, S., Dhillon, I. S., and Topcu, U. (2022). Robust training in high dimensions via block coordinate geometric median descent. In Camps-Valls, G., Ruiz, F. J. R., and Valera, I., editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 11145–11168. PMLR.

Agarwal, P. K., Har-Peled, S., Varadarajan, K. R., et al. (2005). Geometric approximation via coresets. *Combinatorial and computational geometry*, 52(1).

Bajaj, C. (1988). The algebraic degree of geometric optimization problems. *Discrete & Computational Geometry*, 3:177–191.

Balcan, M.-F., Broder, A., and Zhang, T. (2007). Margin based active learning. In *International Conference on Computational Learning Theory*, pages 35–50. Springer.

Chen, Y., Su, L., and Xu, J. (2017). Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):1–25.

Chen, Y., Welling, M., and Smola, A. (2010). Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 109–116.

Cohen, M. B., Lee, Y. T., Miller, G., Pachocki, J., and Sidford, A. (2016). Geometric median in nearly linear time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 9–21.

Diakonikolas, I. and Kane, D. M. (2019). Recent advances in algorithmic high-dimensional robust statistics. *arXiv preprint arXiv:1911.05911*.

Feige, U. (1998). A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652.

Feldman, D. (2020). Core-sets: Updated survey. In *Sampling techniques for supervised or unsupervised tasks*, pages 23–44. Springer.

Feldman, D. and Langberg, M. (2011). A unified framework for approximating and clustering data. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 569–578.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Har-Peled, S. (2011). *Geometric approximation algorithms*. Number 173. American Mathematical Soc.

Har-Peled, S., Roth, D., and Zimak, D. A. (2006). Maximum margin coresets for active and noise tolerant learning.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hendrycks, D. and Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

Hessel, J., Holtzman, A., Forbes, M., Bras, R. L., and Choi, Y. (2021). Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

Huang, S.-J., Jin, R., and Zhou, Z.-H. (2010). Active learning by querying informative and representative examples. *Advances in neural information processing systems*, 23.

Huber, P. J. (1992). Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., and Fei-Fei, L. (2018). Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pages 2304–2313. PMLR.

Joshi, A. J., Porikli, F., and Papanikolopoulos, N. (2009). Multi-class active learning for image classification. In *2009 ieee conference on computer vision and pattern recognition*, pages 2372–2379. IEEE.

Joshi, S. and Mirzasoleiman, B. (2023). Data-efficient contrastive self-supervised learning: Most beneficial examples for supervised learning contribute the least. In *International conference on machine learning*, pages 15356–15370. PMLR.

Katharopoulos, A. and Fleuret, F. (2018). Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR.

Kemperman, J. (1987). The median of a finite measure on a banach space. *Statistical data analysis based on the L1-norm and related methods (Neuchâtel, 1987)*, pages 217–230.

Lamport, L., SHOSTAK, R., and PEASE, M. (1982). The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401.

Li, L., Xu, W., Chen, T., Giannakis, G. B., and Ling, Q. (2019). Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1544–1551.

Li, S., Xia, X., Ge, S., and Liu, T. (2022). Selective-supervised contrastive learning with noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 316–325.

Lopuhaa, H. P., Rousseeuw, P. J., et al. (1991). Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, 19(1):229–248.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. (2018a). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131.

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S., Schoenebeck, G., Song, D., Houle, M. E., and Bailey, J. (2018b). Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Minsker, S. et al. (2015). Geometric median and robust estimation in banach spaces. *Bernoulli*, 21(4):2308–2335.

Muthukrishnan, S. et al. (2005). Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294.

Park, D., Choi, S., Kim, D., Song, H., and Lee, J.-G. (2024). Robust data pruning under label noise via maximizing re-labeling accuracy. *Advances in Neural Information Processing Systems*, 36.

Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. (2017). Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952.

Paul, M., Ganguli, S., and Dziugaite, G. K. (2021). Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34:20596–20607.

Pleiss, G., Zhang, T., Elenberg, E., and Weinberger, K. Q. (2020). Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056.

Shah, V., Wu, X., and Sanghavi, S. (2020). Choosing the sample with lowest loss makes sgd robust. In *International Conference on Artificial Intelligence and Statistics*, pages 2120–2130. PMLR.

Shen, Y. and Sanghavi, S. (2019). Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, pages 5739–5748. PMLR.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sorscher, B., Geirhos, R., Shekhar, S., Ganguli, S., and Morcos, A. (2022). Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

Sugiyama, M. and Kawanabe, M. (2012). *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR.

Toneva, M., Sordoni, A., Combes, R. T. d., Trischler, A., Bengio, Y., and Gordon, G. J. (2018). An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

Vardi, Y. and Zhang, C.-H. (2000). The multivariate l1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4):1423–1426.

Weber, A., Friedrich, C. J., et al. (1929). *Alfred Weber's theory of the location of industries*. The University of Chicago Press.

Weiszfeld, E. (1937). Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386.

Welling, M. (2009). Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128.

Wu, Z., Ling, Q., Chen, T., and Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. *IEEE Transactions on Signal Processing*, 68:4583–4596.

Xia, X., Liu, J., Yu, J., Shen, X., Han, B., and Liu, T. (2022). Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*.

Xia, X., Liu, T., Han, B., Gong, C., Wang, N., Ge, Z., and Chang, Y. (2020). Robust early-learning: Hindering the memorization of noisy labels. In *International conference on learning representations*.

Xu, Y., Zhao, S., Song, J., Stewart, R., and Ermon, S. (2020). A theory of usable information under computational constraints. *arXiv preprint arXiv:2002.10689*.

Yang, S., Xie, Z., Peng, H., Xu, M., Sun, M., and Li, P. (2022). Dataset pruning: Reducing training data by examining generalization influence. *arXiv preprint arXiv:2205.09329*.

Zhang, Z. and Sabuncu, M. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31.

Supplementary Material

4. Experiments

In this section, we outline our experimental setup, present our key empirical findings, and discuss deeper insights into the performance of GM Matching.

4.1. Experimental Setup

A. Baselines:

To ensure reproducibility, our experimental setup is identical to (Xia et al., 2022). We compare the proposed GM Matching selection strategy against the following popular data pruning strategies as baselines for comparison: (1) Random; (2) Herding (Welling, 2009); (3) Forgetting (Toneva et al., 2018); (4) GraNd-score (Paul et al., 2021); (5) EL2N-score (Paul et al., 2021); (6) Optimization-based (Yang et al., 2022); (7) Self-sup.-selection (Sorscher et al., 2022) and (8) Moderate (Xia et al., 2022). We do not run these baselines for be these baselines are borrowed from (Xia et al., 2020).

Additionally, for further ablations we compare GM Matching with many (natural) distance based geometric pruning strategies: (**UNIFORM**) Random Sampling, (**EASY**) Selection of samples closest to the centroid; (**HARD**) Selection of samples farthest from the centroid; (**MODERATE**) (Xia et al., 2022) Selection of samples closest to the median distance from the centroid; (**HERDING**) Moment Matching (Chen et al., 2010), (**GM MATCHING**) Robust Moment (GM) Matching (3).

B. Datasets and Networks :

We perform extensive experiments across three popular image classification datasets - CIFAR10, CIFAR100 and Tiny-ImageNet. Our experiments span popular deep nets including ResNet-18/50 (He et al., 2016), VGG-16 (Simonyan and Zisserman, 2014), ShuffleNet (Ma et al., 2018a), SENet (Hu et al., 2018), EfficientNet-B0(Tan and Le, 2019).

C. Implementation Details :

For the CIFAR-10/100 experiments, we utilize a batch size of 128 and employ SGD optimizer with a momentum of 0.9, weight decay of 5e-4, and an initial learning rate of 0.1. The learning rate is reduced by a factor of 5 after the 60th, 120th, and 160th epochs, with a total of 200 epochs. Data augmentation techniques include random cropping and random horizontal flipping. In the Tiny-ImageNet experiments, a batch size of 256 is used with an SGD optimizer, momentum of 0.9, weight decay of 1e-4, and an initial learning rate of 0.1. The learning rate is decreased by a factor of 10 after the 30th and 60th epochs, with a total of 90 epochs. Random horizontal flips are applied for data augmentation. Each experiment is repeated over 5 random seeds and the variances are noted.

D. Proxy Model :

Needless to say, identifying sample importance is an ill-posed problem without some notion of similarity among the samples. Thus, it is common to assume access to a proxy encoder $\phi_{\mathbf{B}}(\cdot) : \mathbb{R}^d \to \mathcal{H} \in \mathbb{R}^p$ that maps the features to a separable Hilbert space equiped with inner product. Intuitively, this simply means that the embedding space of the encoder fosters proximity of semantically similar examples, while enforcing the separation of dissimilar ones – a property often satisfied by even off-the-shelf pretrained foundation models (Hessel et al., 2021; Sorscher et al., 2022). We perform experiments across multiple choices of such proxy encoder scenarios: In Table 2- 7, we show results in the **standard setting:** when the proxy model shares the same architecture as the model e.g. ResNet50 to be trained downstream, and is pretrained (supervised) on the clean target dataset e.g. TinyimageNet. However, we also experiment with (a) **distribution shift:** proxy model pretrained on a different (distribution shifted) dataset e.g. ImageNet and used to sample from Mini ImageNet. (b) Network Transfer: Where, the proxy has a different network compared to the classifier. We describe these experiments in more detail in Section 4.4.

4.2. Ideal (No Corruption) Scenario

Our first sets of experiments involve performing data pruning across selection ratio ranging from 20% - 80% in the uncorrupted setting. The corresponding results, presented in Table 2, indicate that while GM Matching is developed with robustness scenarios in mind, it outperforms the existing strong baselines even in the clean setting. Overall, on both CIFAR-100 and Tiny ImageNet GM Matching improves over the prior methods > 2% on an average. In particular, we note that GM Matching enjoys larger gains in the low data selection regime, while staying competitive at low pruning rates.

CIFAR-100							
Method / Ratio	**20%**	**30%**	**40%**	**60%**	**80%**	**100%**	**Mean ↑**
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 2: **No Corruption :** Comparing (Test Accuracy) pruning algorithms on CIFAR-100 and Tiny-ImageNet in the uncorrupted setting. ResNet-50 is used both as proxy and for downstream classification.

4.3. Corruption Scenarios

To understand the performance of data pruning strategies in presence of corruption, we experiment with three different sources of corruption – image corruption, label noise and adversarial attacks.

4.3.1. Robustness to Image Corruption

In this set of experiments, we investigate the robustness of data pruning strategies when the input images are corrupted – a popular robustness setting, often encountered when training models on real-world data (Hendrycks and Dietterich, 2019; Szegedy et al., 2013). To corrupt images, we apply five types of realistic noise: Gaussian noise, random occlusion, resolution reduction, fog, and motion blur to parts of the corrupt samples i.e. to say if m samples are corrupted, each type of noise is added to one a random $m/5$ of them, while the other partitions are corrupted with a different noise. The results are presented in Table 3, 4. We observe that GM Matching outperforms all the baselines across all pruning rates improving ≈3% across both datasets on an average. We note that, the gains are more consistent and profound in this setting over the clean setting. Additionally, similar to our prior observations in the clean setting, the gains of GM Matching are more significant at high pruning rates (i.e. low selection ratio).

4.3.2. Robustness to Label Corruption

Next, we consider another important corruption scenario where a fraction of the training examples are mislabeled. Since acquiring manually labeled data is often impractical and data in the wild always contains noisy annotations – it is important to ensure the pruning method is robust to label noise. We conduct experiments with synthetically injected symmetric label noise (Li et al., 2022; Patrini et al., 2017; Xia et al., 2020). The results are summarized in Table 5,6. Encouragingly, GM Matching **outperforms the baselines by ≈ 12%**. Since, mislabeled samples come from different class - they are spatially quite dissimilar than the correctly labeled ones thus, less likely to be picked by GM matching, explaining the superior performance.

4.3.3. Robustness to Adversarial Attacks

Finally, we investigate the robustness properties of data pruning strategies in presence of adversarial attacks that add imperceptible but adversarial noise on natural examples (Szegedy et al., 2013; Ma et al., 2018b; Huang et al., 2010).

CIFAR-100							
Method / Selection ratio	**20%**	**30%**	**40%**	**60%**	**80%**	**100%**	**Mean ↑**
			No Corruption				
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**
			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: **Image Corruption (CIFAR 100):** Comparing (Test Accuracy) pruning methods when 20% of the images are corrupted. ResNet-50 is used both as proxy and for downstream classification.

Tiny ImageNet							
Method / Ratio	**20%**	**30%**	**40%**	**60%**	**80%**	**100%**	**Mean ↑**
No Corruption							
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**
5% Feature Corruption							
Random	23.51±0.22	28.82±0.72	32.61±0.68	39.77±0.35	44.37±0.34	49.02±0.35	33.82
Herding	23.09±0.53	28.67±0.37	33.09±0.32	39.71±0.31	45.04±0.15	49.02±0.35	33.92
Forgetting	21.36±0.28	27.72±0.43	33.45±0.21	40.92±0.45	45.99±0.51	49.02±0.35	33.89
GraNd-score	22.47±0.23	28.85±0.83	33.81±0.24	40.40±0.15	44.86±0.49	49.02±0.35	34.08
EL2N-score	18.98±0.72	25.96±0.28	31.07±0.63	38.65±0.36	44.21±0.68	49.02±0.35	31.77
Optimization-based	13.65±1.26	24.02±1.35	29.65±1.86	36.55±1.84	43.64±0.71	49.02±0.35	29.50
Self-sup.-selection	19.35±0.57	26.11±0.31	31.90±0.37	38.91±0.29	44.43±0.42	49.02±0.35	32.14
Moderate-DS	24.63±0.78	30.27±0.16	34.84±0.24	40.86±0.42	45.60±0.31	49.02±0.35	35.24
GM Matching	**27.46±1.22**	**33.14±0.61**	**35.76±1.14**	**41.62±0.71**	**46.83±0.56**	49.02±0.35	**36.96**
10% Feature Corruption							
Random	22.67±0.27	28.67±0.52	31.88±0.30	38.63±0.36	43.46±0.20	48.40±0.32	33.06
Herding	22.01±0.18	27.82±0.11	31.82±0.26	39.37±0.18	44.18±0.27	48.40±0.32	33.04
Forgetting	20.06±0.48	27.17±0.36	32.31±0.22	40.19±0.29	45.51±0.48	48.40±0.32	33.05
GraNd-score	21.52±0.48	26.98±0.43	32.70±0.19	40.03±0.26	44.87±0.35	48.40±0.32	33.22
EL2N-score	18.59±0.13	25.23±0.18	30.37±0.22	38.44±0.32	44.32±1.07	48.40±0.32	31.39
Optimization-based	14.05±1.74	29.18±1.77	29.12±0.61	36.28±1.88	43.52±0.31	48.40±0.32	29.03
Self-sup.-selection	19.47±0.26	26.51±0.55	31.78±0.14	38.87±0.54	44.69±0.29	48.40±0.32	32.26
Moderate-DS	23.79±0.16	29.56±0.16	34.60±0.12	40.36±0.27	45.10±0.23	48.40±0.32	34.68
GM Matching	**27.41±0.23**	**32.84±0.98**	**36.27±0.68**	**41.85±0.29**	**46.35±0.44**	48.40±0.32	**36.94**
20% Feature Corruption							
Random	19.99±0.42	25.93±0.53	30.83±0.44	37.98±0.31	42.96±0.62	46.68±0.43	31.54
Herding	19.46±0.14	24.47±0.33	29.72±0.39	37.50±0.59	42.28±0.30	46.68±0.43	30.86
Forgetting	18.47±0.46	25.53±0.23	31.17±0.24	39.35±0.44	44.55±0.67	46.68±0.43	31.81
GraNd-score	20.07±0.49	26.68±0.40	31.25±0.40	38.21±0.49	42.84±0.72	46.68±0.43	30.53
EL2N-score	18.57±0.30	24.42±0.44	30.04±0.15	37.62±0.44	42.43±0.61	46.68±0.43	30.53
Optimization-based	13.71±0.26	23.33±1.84	29.15±2.84	36.12±1.86	42.94±0.52	46.88±0.43	29.06
Self-sup.-selection	20.22±0.23	26.90±0.50	31.93±0.49	39.74±0.52	44.27±0.10	46.68±0.43	32.61
Moderate-DS	23.27±0.33	29.06±0.36	33.48±0.11	40.07±0.36	44.73±0.39	46.68±0.43	34.12
GM Matching	**27.19±0.92**	**31.70±0.78**	**35.14±0.19**	**42.04±0.31**	**45.12±0.28**	46.68±0.43	**36.24**

Table 4: **Image Corruption (Tiny ImageNet):** Comparing (Test Accuracy) pruning methods under feature (image) corruption. ResNet-50 is used both as proxy and for downstream classification.

	CIFAR-100 (Label noise)		Tiny ImageNet (Label noise)		
Method / Ratio	**20%**	**30%**	**20%**	**30%**	**Mean ↑**
		20% Label Noise			
Random	34.47±0.64	43.26±1.21	17.78±0.44	23.88±0.42	29.85
Herding	42.29±1.75	50.52±3.38	18.98±0.44	24.23±0.29	34.01
Forgetting	36.53±1.11	45.78±1.04	13.20±0.38	21.79±0.43	29.33
GraNd-score	31.72±0.67	42.80±0.30	18.28±0.32	23.72±0.18	28.05
EL2N-score	29.82±1.19	33.62±2.35	13.93±0.69	18.57±0.31	23.99
Optimization-based	32.79±0.62	41.80±1.14	14.77±0.95	22.52±0.77	27.57
Self-sup.-selection	31.08±0.78	41.87±0.63	15.10±0.73	21.01±0.36	27.27
Moderate-DS	40.25±0.12	48.53±1.60	19.64±0.40	24.96±0.30	31.33
GM Matching	**52.64±0.72**	**61.01±0.47**	**25.80±0.37**	**31.71±0.24**	**42.79**
		35% Label Noise			
Random	24.51±1.34	32.26±0.81	14.64±0.29	19.41±0.45	22.71
Herding	29.42±1.54	37.50±2.12	15.14±0.45	20.19±0.45	25.56
Forgetting	29.48±1.98	38.01±2.21	11.25±0.90	17.07±0.66	23.14
GraNd-score	23.03±1.05	34.83±2.01	13.68±0.46	19.51±0.45	22.76
EL2N-score	21.95±1.08	31.63±2.84	10.11±0.25	13.69±0.32	19.39
Optimization-based	26.77±0.15	35.63±0.92	12.37±0.68	18.52±0.90	23.32
Self-sup.-selection	23.12±1.47	34.85±0.68	11.23±0.32	17.76±0.69	22.64
Moderate-DS	28.45±0.53	36.55±1.26	15.27±0.31	20.33±0.28	25.15
GM Matching	**43.33± 1.02**	**58.41± 0.68**	**23.14± 0.92**	**27.76± 0.40**	**38.16**

Table 5: **Robustness to Label Noise:** Comparing (Test Accuracy) pruning methods on CIFAR-100 and TinyImageNet datasets, under 20% and 35% Symmetric Label Corruption, at 20% and 30% selection ratio. ResNet-50 is used both as proxy and for downstream classification.

			Tiny ImageNet (Label Noise)				
Method / Ratio	**20%**	**30%**	**40%**	**60%**	**80%**	**100%**	**Mean ↑**
Random	17.78±0.44	23.88±0.42	27.97±0.39	34.88±0.51	38.47±0.40	44.42±0.47	28.60
Herding	18.98±0.44	24.23±0.29	27.28±0.31	34.36±0.29	39.00±0.49	44.42±0.47	28.87
Forgetting	13.20±0.38	21.79±0.43	27.89±0.22	**36.03±0.24**	**40.60±0.31**	44.42±0.47	27.50
GraNd-score	18.28±0.32	23.72±0.18	27.34±0.33	34.91±0.19	39.45±0.45	44.42±0.47	28.34
EL2N-score	13.93±0.69	18.57±0.31	24.56±0.34	32.14±0.49	37.64±0.41	44.42±0.47	25.37
Optimization-based	14.77±0.95	22.52±0.77	25.62±0.90	34.18±0.79	38.49±0.69	44.42±0.47	27.12
Self-sup.-selection	15.10±0.73	21.01±0.36	26.62±0.22	33.93±0.36	39.22±0.12	44.42±0.47	27.18
Moderate-DS	**19.64±0.40**	**24.96±0.30**	**29.56±0.21**	35.79±0.36	39.93±0.23	44.42±0.47	30.18
GM Matching	**25.80±0.37**	**31.71±0.24**	**34.87±0.21**	**39.76±0.71**	**41.94±0.23**	44.42±0.47	**34.82**

Table 6: **Pruning with Label Noise (TinyImageNet):** Comparing (Test Accuracy) pruning methods under 20% Symmetric Label Corruption across wide array of selection ratio. ResNet-50 is used both as proxy and for downstream classification.

	CIFAR-100 (PGD Attack)		CIFAR-100 (GS Attack)		
Method / Ratio	20%	30%	20%	30%	**Mean ↑**
Random	43.23±0.31	52.86±0.34	44.23±0.41	53.44±0.44	48.44
Herding	40.21±0.72	49.62±0.65	39.92±1.03	50.14±0.15	44.97
Forgetting	35.90±1.30	47.37±0.99	37.55±0.53	46.88±1.91	41.93
GraNd-score	40.87±0.84	50.13±0.30	40.77±1.11	49.88±0.83	45.41
EL2N-score	26.61±0.58	34.50±1.02	26.72±0.66	35.55±1.30	30.85
Optimization-based	38.29±1.77	46.25±1.82	41.36±0.92	49.10±0.81	43.75
Self-sup.-selection	40.53±1.15	49.95±0.50	40.74±1.66	51.23±0.25	45.61
Moderate-DS	43.60±0.97	51.66±0.39	44.69±0.68	53.71±0.37	48.42
GM Matching	**45.41 ±0.86**	**51.80 ±1.01**	**49.78 ±0.27**	**55.50 ±0.31**	**50.62**
	Tiny ImageNet (PGD Attack)		**Tiny ImageNet (GS Attack)**		
Method / Ratio	20%	30%	20%	30%	**Mean ↑**
Random	20.93±0.30	26.60±0.98	22.43±0.31	26.89±0.31	24.21
Herding	21.61±0.36	25.95±0.19	23.04±0.28	27.39±0.14	24.50
Forgetting	20.38±0.47	26.12±0.19	22.06±0.31	27.21±0.21	23.94
GraNd-score	20.76±0.21	26.34±0.32	22.56±0.30	27.52±0.40	24.30
EL2N-score	16.67±0.62	22.36±0.42	19.93±0.57	24.65±0.32	20.93
Optimization-based	19.26±0.77	24.55±0.92	21.26±0.24	25.88±0.37	22.74
Self-sup.-selection	19.23±0.46	23.92±0.51	19.70±0.20	24.73±0.39	21.90
Moderate-DS	21.81±0.37	27.11±0.20	23.20±0.13	28.89±0.27	25.25
GM Matching	**25.98 ±1.12**	**30.77 ±0.25**	**29.71 ±0.45**	**32.88 ±0.73**	**29.84**

Table 7: **Robustness to Adversarial Attacks**. Comparing (Test Accuracy) pruning methods under PGD and GS attacks. ResNet-50 is used both as proxy and for downstream classification.

Specifically, we experiment with two popular adversarial attack algorithms – PGD attack (Madry et al., 2017) and GS Attacks (Goodfellow et al., 2014). We employ adversarial attacks on models trained with CIFAR-100 and Tiny-ImageNet to generate adversarial examples. Following this, various methods are applied to these adversarial examples, and the models are retrained on the curated subset of data. The results are summarized in Table 7. Similar to other corruption scenarios, even in this setting, GM Matching outperforms the baselines ≈ and remains effective across different pruning rates yielding ≈ 3% average gain over the best performing baseline.

4.4. Generalization to Unseen Network / Domain

One crucial component of data pruning is the proxy network. Since, the input features (e.g. images) often reside on a non-separable manifold, data pruning strategies rely on a proxy model to map the samples into a separable manifold (embedding space), wherein the data pruning strategies can now assign importance scores. However, it is important for the data pruning strategies to be robust to architecture changes i.e. to say that samples selected via a proxy network should generalize well when trained on unseen (during sample selection) networks / domains. We perform experiments on two such

	ResNet-50→SENet		ResNet-50→EfficientNet-B0		
Method / Ratio	**20%**	**30%**	**20%**	**30%**	**Mean ↑**
Random	34.13±0.71	39.57±0.53	32.88±1.52	39.11±0.94	36.42
Herding	34.86±0.55	38.60±0.68	32.21±1.54	37.53±0.22	35.80
Forgetting	33.40±0.64	39.79±0.78	31.12±0.21	38.38±0.65	35.67
GraNd-score	35.12±0.54	41.14±0.42	33.20±0.67	40.02±0.35	37.37
EL2N-score	31.08±1.11	38.26±0.45	31.34±0.49	36.88±0.32	34.39
Optimization-based	33.18±0.52	39.42±0.77	32.16±0.90	38.52±0.50	35.82
Self-sup.-selection	31.74±0.71	38.45±0.39	30.99±1.03	37.96±0.77	34.79
Moderate-DS	36.04±0.15	41.40±0.20	34.26±0.48	39.57±0.29	37.82
GM Matching	**37.93±0.23**	**42.59±0.29**	**36.31±0.67**	**41.03±0.41**	**39.47**

Table 8: **Network Transfer (Clean)** : Tiny-ImageNet Model Transfer Results. A ResNet-50 proxy is used to find important samples which are then used to train SENet and EfficientNet.

	ResNet-50→ VGG-16		**ResNet-50→ ShuffleNet**		
Method / Ratio	20%	30%	20%	30%	**Mean ↑**
		No Corruption			
Random	29.63±0.43	35.38±0.83	32.40±1.06	39.13±0.81	34.96
Herding	31.05±0.22	36.27±0.57	33.10±0.39	38.65±0.22	35.06
Forgetting	27.53±0.36	35.61±0.39	27.82±0.56	36.26±0.51	32.35
GraNd-score	29.93±0.95	35.61±0.39	29.56±0.46	37.40±0.38	33.34
EL2N-score	26.47±0.31	33.19±0.51	28.18±0.27	35.81±0.29	31.13
Optimization-based	25.92±0.64	34.82±1.29	31.37±1.14	38.22±0.78	32.55
Self-sup.-selection	25.16±1.10	33.30±0.94	29.47±0.56	36.68±0.36	31.45
Moderate-DS	31.45±0.32	37.89±0.36	33.32±0.41	39.68±0.34	35.62
GM Matching	**35.86±0.41**	**40.56±0.22**	**35.51±0.32**	**40.30±0.58**	**38.47**
		20% Label Corruption			
Random	23.29±1.12	28.18±1.84	25.08±1.32	31.44±1.21	27.00
Herding	23.99±0.36	28.57±0.40	26.25±0.47	30.73±0.28	27.39
Forgetting	14.52±0.66	21.75±0.23	15.70±0.29	22.31±0.35	18.57
GraNd-score	22.44±0.46	27.95±0.29	23.64±0.10	30.85±0.21	26.22
EL2N-score	15.15±1.25	23.36±0.30	18.01±0.44	24.68±0.34	20.30
Optimization-based	22.93±0.58	24.92±2.50	25.82±1.70	30.19±0.48	25.97
Self-sup.-selection	18.39±1.30	25.77±0.87	22.87±0.54	29.80±0.36	24.21
Moderate-DS	23.68±0.19	28.93±0.19	28.82±0.33	32.39±0.21	28.46
GM Matching	**28.77±0.77**	**34.87±0.23**	**32.05±0.93**	**37.43±0.25**	**33.28**
		20% Feature Corruption			
Random	26.33±0.88	31.57±1.31	29.15±0.83	34.72±1.00	30.44
Herding	18.03±0.33	25.77±0.34	23.33±0.43	31.73±0.38	24.72
Forgetting	19.41±0.57	28.35±0.16	18.44±0.57	31.09±0.61	24.32
GraNd-score	23.59±0.19	30.69±0.13	23.15±0.56	31.58±0.95	27.25
EL2N-score	24.60±0.81	31.49±0.33	26.62±0.34	33.91±0.56	29.16
Optimization-based	25.12±0.34	30.52±0.89	28.87±1.25	34.08±1.92	29.65
Self-sup.-selection	26.33±0.21	33.23±0.26	26.48±0.37	33.54±0.46	29.90
Moderate-DS	29.65±0.68	35.89±0.53	32.30±0.38	38.66±0.29	34.13
GM Matching	**33.45±1.02**	**39.46±0.44**	**35.14±0.21**	**39.89±0.98**	**36.99**
		PGD Attack			
Random	26.12±1.09	31.98±0.78	28.28±0.90	34.59±1.18	30.24
Herding	26.76±0.59	32.56±0.35	28.87±0.48	35.43±0.22	30.91
Forgetting	24.55±0.57	31.83±0.36	23.32±0.37	31.82±0.15	27.88
GraNd-score	25.19±0.33	31.46±0.54	26.03±0.66	33.22±0.24	28.98
EL2N-score	21.73±0.47	27.66±0.32	22.66±0.35	29.89±0.64	25.49
Optimization-based	26.02±0.36	31.64±1.75	27.93±0.47	34.82±0.96	30.10
Self-sup.-selection	22.36±0.30	28.56±0.50	25.35±0.27	32.57±0.13	27.21
Moderate-DS	27.24±0.36	32.90±0.31	29.06±0.28	35.89±0.53	31.27
GM Matching	**27.96±1.60**	**35.76±0.82**	**34.11±0.65**	**40.91±0.84**	**34.69**

Table 9: **Network Transfer** : A ResNet-50 proxy (pretrained on TinyImageNet) is used to find important samples from Tiny-ImageNet; which is then used to train a VGGNet-16 and ShuffleNet. We repeat the experiment across multiple corruption settings - clean; 20% Feature / Label Corruption and PGD attack when 20% and 30% samples are selected.

scenarios:

A. Network Transfer: In this setting, the proxy model is trained on the target dataset (no distribution shift). However, the proxy architecture is different than the downstream network. In Table 8, we use a ResNet-50 proxy trained on MiniImageNet to sample the data. However, then we train a downstram SENet and EfficientNet-B0 on the sampled data. In Table 9, we use a ResNet-50 pretrained on Mini ImageNet as proxy, whereas we train a VGG-16 and ShuffleNet over the selected samples.

B. Domain Transfer: Next, we consider the setting where the proxy shares the same architecture with the downstream model. However, the proxy used to select the samples is pretrained on a different dataset (distribution shift) than target dataset. In **??** we use a proxy ResNet-18 pretrained on ImageNet to select samples from CIFAT-10. The selected samples are used to train a subsequent ResNet-18. In **??**, we additionally freeze the pretrained encoder i.e. we use ResNet-18 encoder pretrained on ImageNet as proxy. Further, we freeze the encoder and train a downstream linear classifier on top over CIFAR-10 (linear probe).

5. Complete Proofs

In this section we will describe the proof techniques involved in establishing the theoretical claims in the paper.

5.0.1. Intermediate Lemmas

In order to prove Theorem 1, we will first establish the following result which follows from the definition of GM; see also (Lopuhaa et al., 1991; Minsker et al., 2015; Cohen et al., 2016; Chen et al., 2017; Li et al., 2019; Wu et al., 2020; Acharya et al., 2022) for similar adaptations.

Lemma 1. *Given a set of α-corrupted samples $\mathcal{D} = \mathcal{D}_\mathcal{G} \cup \mathcal{D}_\mathcal{B}$ (Definition 1), and an ϵ-approx.* GM$(\cdot)$ *oracle* (1)*, then we have:*

$$\mathbb{E}\Big\|\boldsymbol{\mu}^{\text{GM}} - \boldsymbol{\mu}^{\mathcal{G}}\Big\|^2 \le \frac{8|\mathcal{D}_\mathcal{G}|}{(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|)^2}\sum_{\mathbf{x}\in\mathcal{D}_\mathcal{G}}\mathbb{E}\Big\|\mathbf{x} - \boldsymbol{\mu}^{\mathcal{G}}\Big\|^2 + \frac{2\epsilon^2}{(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|)^2} \tag{7}$$

where, $\boldsymbol{\mu}^{\text{GM}} = \text{GM}(\{\mathbf{x}_i \in \mathcal{D}\})$ is the ϵ-approximate GM *over the entire (α-corrupted) dataset; and $\boldsymbol{\mu}^{\mathcal{G}} = \frac{1}{|\mathcal{D}_\mathcal{G}|}\sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\mathbf{x}_i$ denotes the mean of the (underlying) uncorrupted set.*

Proof. Note that, by using triangle inequality, we can write:

$$\begin{aligned}\sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\boldsymbol{\mu}^{\text{GM}} - \mathbf{x}_i\Big\| &\ge \sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{B}}\Big(\Big\|\mathbf{x}_i\Big\| - \Big\|\boldsymbol{\mu}^{\text{GM}}\Big\|\Big) + \sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\Big(\Big\|\boldsymbol{\mu}^{\text{GM}}\Big\| - \Big\|\mathbf{x}_i\Big\|\Big)\\ &= \Big(\sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}} - \sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{B}}\Big)\Big\|\boldsymbol{\mu}^{\text{GM}}\Big\| + \sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{B}}\Big\|\mathbf{x}_i\Big\| - \sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\Big\|\mathbf{x}_i\Big\|\\ &= \Big(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|\Big)\Big\|\boldsymbol{\mu}^{\text{GM}}\Big\| + \sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\mathbf{x}_i\Big\| - 2\sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\Big\|\mathbf{x}_i\Big\|.\end{aligned} \tag{8}$$

Now, by definition (2); we have that:

$$\sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\boldsymbol{\mu}^{\text{GM}} - \mathbf{x}_i\Big\| \le \inf_{\mathbf{z}\in\mathcal{H}}\sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\mathbf{z} - \mathbf{x}_i\Big\| + \epsilon \le \sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\mathbf{x}_i\Big\| + \epsilon \tag{9}$$

Combining these two inequalities, we get:

$$\Big(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|\Big)\Big\|\boldsymbol{\mu}^{\text{GM}}\Big\| \le \sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\mathbf{x}_i\Big\| - \sum_{\mathbf{x}_i\in\mathcal{D}}\Big\|\mathbf{x}_i\Big\| + 2\sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\Big\|\mathbf{x}_i\Big\| + \epsilon \tag{10}$$

This implies:

$$\Big\|\boldsymbol{\mu}^{\text{GM}}\Big\| \le \frac{2}{\Big(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|\Big)}\sum_{\mathbf{x}_i\in\mathcal{D}_\mathcal{G}}\Big\|\mathbf{x}_i\Big\| + \frac{\epsilon}{\Big(|\mathcal{D}_\mathcal{G}| - |\mathcal{D}_\mathcal{B}|\Big)} \tag{11}$$

Squaring both sides,

$$\left\|\boldsymbol{\mu}^{\mathrm{GM}}\right\|^2 \leq \left[\frac{2}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)} \sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{G}}} \left\|\mathbf{x}_i\right\| + \frac{\epsilon}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)}\right]^2 \tag{12}$$

$$\leq 2\left[\frac{2}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)} \sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{G}}} \left\|\mathbf{x}_i\right\|\right]^2 + 2\left[\frac{\epsilon}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)}\right]^2 \tag{13}$$

Where, the last step is a well-known consequence of triangle inequality and AM-GM inequality. Taking expectation on both sides, we have:

$$\mathbb{E}\left\|\boldsymbol{\mu}^{\mathrm{GM}}\right\|^2 \leq \frac{8}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)^2} \sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{G}}} \mathbb{E}\left\|\mathbf{x}_i\right\|^2 + \frac{2\epsilon^2}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)^2} \tag{14}$$

Since, GM is **translation equivariant**, we can write:

$$\mathbb{E}\left[\mathrm{GM}\left(\left\{\mathbf{x}_i - \boldsymbol{\mu}^{\mathcal{G}} | \mathbf{x}_i \in \mathcal{D}\right\}\right)\right] = \mathbb{E}\left[\mathrm{GM}\left(\left\{\mathbf{x}_i | \mathbf{x}_i \in \mathcal{D}\right\}\right) - \boldsymbol{\mu}^{\mathcal{G}}\right] \tag{15}$$

Consequently we have that :

$$\mathbb{E}\left\|\boldsymbol{\mu}^{\mathrm{GM}} - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 \leq \frac{8}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)^2} \sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{G}}} \mathbb{E}\left\|\mathbf{x}_i - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 + \frac{2\epsilon^2}{\Big(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|\Big)^2}$$

This concludes the proof. ■

5.0.2. PROOF OF THEOREM 1

We restate the theorem for convenience:

Theorem 1 Suppose that, we are given, a set of α-corrupted samples $\mathcal{D}$ (Definition 1), pretrained proxy model $\phi_{\mathbf{B}}$, and an ϵ approx. $\mathrm{GM}(\cdot)$ oracle (1). Then, GM MATCHING guarantees that the mean of the selected k-subset $\boldsymbol{\mu}^{\mathcal{S}} = \frac{1}{k}\sum_{\mathbf{x}_i \in \mathcal{D}_{\mathcal{S}}} \mathbf{x}_i$ converges to the neighborhood of $\boldsymbol{\mu}^{\mathcal{G}} = \mathbb{E}_{\mathbf{x} \in \mathcal{D}_{\mathcal{G}}}(\mathbf{x})$ at the rate $\mathcal{O}(\frac{1}{k})$ such that:

$$\mathbb{E}\left\|\boldsymbol{\mu}^{\mathcal{S}} - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 \leq \frac{8|\mathcal{D}_{\mathcal{G}}|}{(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|)^2} \sum_{\mathbf{x} \in \mathcal{D}_{\mathcal{G}}} \mathbb{E}\left\|\mathbf{x} - \boldsymbol{\mu}^{\mathcal{G}}\right\|^2 + \frac{2\epsilon^2}{(|\mathcal{D}_{\mathcal{G}}| - |\mathcal{D}_{\mathcal{B}}|)^2} \tag{16}$$

Proof. To prove this result, we first show that GM MATCHING converges to $\boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}}$ at $\mathcal{O}(\frac{1}{k})$. It suffices to show that the error $\delta = \left\|\boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}} - \frac{1}{k}\sum_{\mathbf{x}_i \in \mathcal{S}} \mathbf{x}_i\right\| \to 0$ asymptotically. We will follow the proof technique in (Chen et al., 2010) mutatis mutandis to prove this result. We also assume that $\mathcal{D}$ contains the support of the resulting noisy distribution.

We start by defining a GM-centered marginal polytope as the convex hull –

$$\mathcal{M}_{\epsilon} := \mathrm{conv}\left\{\mathbf{x} - \boldsymbol{\mu}_{\epsilon}^{\mathrm{GM}} \; | \mathbf{x} \in \mathcal{D}\right\} \tag{17}$$

Then, we can rewrite the update equation (**??**) as:

$$\theta_{t+1} = \theta_t + \boldsymbol{\mu}_\epsilon^{\text{GM}} - \mathbf{x}_{t+1} \tag{18}$$
$$= \theta_t - (\mathbf{x}_{t+1} - \boldsymbol{\mu}_\epsilon^{\text{GM}}) \tag{19}$$
$$= \theta_t - \left(\arg\max_{\mathbf{x}\in\mathcal{D}} \langle \theta_t, \mathbf{x} \rangle - \boldsymbol{\mu}_\epsilon^{\text{GM}} \right) \tag{20}$$
$$= \theta_t - \arg\max_{\mathbf{m}\in\mathcal{M}_\epsilon} \langle \theta_t, \mathbf{m} \rangle \tag{21}$$
$$= \theta_t - \mathbf{m}_t \tag{22}$$

Now, squaring both sides we get :

$$\|\theta_{t+1}\|^2 = \|\theta_t\|^2 + \|\mathbf{m}_t\|^2 - 2\langle \theta_t, \mathbf{m}_t \rangle \tag{23}$$

rearranging the terms we get:

$$\|\theta_{t+1}\|^2 - \|\theta_t\|^2 = \|\mathbf{m}_t\|^2 - 2\langle \theta_t, \mathbf{m}_t \rangle \tag{24}$$
$$= \|\mathbf{m}_t\|^2 - 2\|\mathbf{m}_t\|\|\theta_t\| \langle \frac{\theta_t}{\|\theta_t\|}, \frac{\mathbf{m}_t}{\|\mathbf{m}_t\|} \rangle \tag{25}$$
$$= 2\|\mathbf{m}_t\| \left(\frac{1}{2}\|\mathbf{m}_t\| - \|\theta_t\| \langle \frac{\theta_t}{\|\theta_t\|}, \frac{\mathbf{m}_t}{\|\mathbf{m}_t\|} \rangle \right) \tag{26}$$

Assume that $\|\mathbf{x}_i\| \leq r \; \forall \mathbf{x}_i \in \mathcal{D}$. , Then we note that,

$$\|\mathbf{x}_i - \boldsymbol{\mu}_\epsilon^{\text{GM}}\| \leq \|\mathbf{x}_i\| + \|\boldsymbol{\mu}_\epsilon^{\text{GM}}\| \leq 2r$$

Plugging this in, we get:

$$\|\theta_{t+1}\|^2 - \|\theta_t\|^2 \leq 2\|\mathbf{m}_t\| \left(r - \|\theta_t\| \langle \frac{\theta_t}{\|\theta_t\|}, \frac{\mathbf{m}_t}{\|\mathbf{m}_t\|} \rangle \right) \tag{27}$$

Recall that, $\boldsymbol{\mu}_\epsilon^{\text{GM}}$ is guaranteed to be in the relative interior of conv$\{\mathbf{x} \mid \mathbf{x} \in \mathcal{D}\}$ (Lopuhaa et al., 1991; Minsker et al., 2015). Consequently, $\exists\kappa$-ball around $\boldsymbol{\mu}_\epsilon^{\text{GM}}$ contained inside $\mathcal{M}$ and we have $\forall t > 0$

$$\langle \frac{\theta_t}{\|\theta_t\|}, \frac{\mathbf{m}_t}{\|\mathbf{m}_t\|} \rangle \geq \kappa > 0 \tag{28}$$

This implies, $\forall t > 0$

$$\|\theta_t\| \leq \frac{r}{\kappa} \tag{29}$$

Expanding the value of θ_t we have:

$$\left\|\theta_k\right\| = \left\| \theta_0 + k\boldsymbol{\mu}_\epsilon^{\text{GM}} - \sum_{i=1}^{k} \mathbf{x}_k \right\| \leq \frac{r}{\kappa} \tag{30}$$

Apply Cauchy Schwartz inequality:

$$\left\| k\boldsymbol{\mu}_\epsilon^{\text{GM}} - \sum_{i=1}^{k} \mathbf{x}_k \right\| \leq \left\|\theta_0\right\| + \frac{r}{\kappa} \tag{31}$$

normalizing both sides by number of iterations k

$$\left\|\boldsymbol{\mu}_{\epsilon}^{\textsc{Gm}} - \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}_k\right\| \leq \frac{1}{k}\left(\left\|\theta_0\right\| + \frac{r}{\kappa}\right) \tag{32}$$

Thus, we have that GM MATCHING converges to $\boldsymbol{\mu}_{\epsilon}^{\textsc{Gm}}$ at the rate $\mathcal{O}(\frac{1}{k})$.

Combining this with Lemma 1, completes the proof. ■