Mycroft: Towards Effective and Efficient External Data Augmentation

Zain Sarwar*, **Van Tran***, **Arjun Nitin Bhagoji***, **Nick Feamster, Ben Y. Zhao**
University of Chicago
{zsarwar, tranv, abhagoji, feamster, ravenben}@uchicago.edu

Supriyo Chakraborty
Capital One
supriyo.chakraborty@capitalone.com

Abstract

In data-scarce domains like networked systems, external data augmentation may often be necessary to improve training data quality, as model trainers usually only have visibility into limited portions of the underlying data distribution. However, relevant data is often privately owned, making it both difficult and expensive for trainers to identify and acquire the needed training data. In this study, we introduce Mycroft, a data-efficient approach that allows model trainers to evaluate the utility of private data from various owners while operating under a limited data-sharing budget. Mycroft leverages feature space distances to identify small, high-utility data subsets from each data owner, which serve as indicators of the overall dataset's utility. In domains with differentiable models, Mycroft can effectively apply gradient matching techniques to identify these valuable data subsets. Our experiments, including novel threat detection in IoT networks and image classification in the vision domain, show that Mycroft quickly reaches performance levels comparable to the baseline where all the data is shared.

1 Introduction

In data-scarce domains like networked systems, model trainers often face limited access to diverse, representative training data. Public datasets are typically narrow in scope [1], and network data owned by private entities is rarely shared due to privacy concerns. Additionally, acquiring data from private sources generally requires formal agreements and compensation, making external data both costly and challenging to obtain. Therefore, model trainers must carefully select data providers whose data aligns with their needs, while data owners should share only the data necessary to optimize utility and minimize privacy risks. The key question our paper answers is: *How can a model trainer determine if the data owned by a third party is useful to them without acquiring all of it?* We propose Mycroft, a data sharing framework to address this challenge. Our framework operates in three broad steps: (1) the model trainer transmits some information about the task their model is not performing well on, typically in the form of data samples; (2) the data owners then use an algorithm to identify, using (1), a small but meaningful subset of their data that provides evidence of its utility (or lack thereof); and (3) the model trainer then evaluates the utility of the provided subset on their task, and decides, which, if any, of the data owners to acquire data from. The *key technical challenge*(§3) we solve in this paper lies in Step (2), where each data owner must find the most "relevant" training data with respect to the model trainer's transmitted data samples that may not be i.i.d. with their training data.

*These authors contributed equally to this work

Machine Learning for Systems Workshop at (NeurIPS 2024).

Our experimental results(§5) show that Mycroft outperforms random sampling methods and rapidly converges to the performance of the full data sharing setting while using a much smaller data budget, especially for IoT data. This means that model trainers can acquire useful network attack data with minimal overheads.

2 Problem Formulation and Notation

We consider a setting where there is a model trainer (MT) and m data owners (DOs). MT has trained a model M_{MT} on its own dataset D^{MT} and is aiming to improve their performance on test data D^{test} via external data augmentation. The performance of MT's model on D^{test} is measured with respect to some task. In this paper, we assume that the task is supervised learning. We then posit that there exists a subset D^{hard} of D^{test} on which MT is aiming to improve their performance, leading them to use external data augmentation. Each DO has a dataset D_i which could aid MT in improving their performance on D^{hard}. However, the DOs do not share all of their data with MT. Rather, they share a small subset D^{useful} of up to size k, which we call the *budget*. Moreover, the MT does not share M_{MT} with the DO. If D^{useful} is able to improve the performance of MT's model, then MT and DO could potentially enter into a data-sharing agreement for additional data acquisition. This paper focuses on how each DO can identify D^{useful} and subsequently, how MT can utilize this data or rank several DOs. Each DO's task is then:

Definition 2.1 (Task for each DO)**.** Find $D_i^{\text{useful}} \subseteq D_i$ such that $|D_i^{\text{useful}}| \leq k$ and $L(D^{\text{hard}}, M'_{\text{MT}}) \leq L(D^{\text{hard}}, M_{\text{MT}})$, where $M_{\text{MT}} = \texttt{TRAIN}(D^{\text{MT}})$ and $M'_{\text{MT}} = \texttt{TRAIN}(D^{\text{MT}} \cup D_i^{\text{useful}})$.

3 Mycroft: Identifying and sharing useful data

Algorithm 1 Mycroft

Require: M_{MT}, D^{test}, DO's loss function L_{DO}, DO_i's dataset D_i, Budget k,

1: $D^{\text{hard}} \leftarrow \texttt{TEST}(D^{\text{test}}, M_{\text{MT}})$
2: Send D^{hard} to the DOs
3: $D_i^{\text{useful}} \leftarrow \text{DO}_i$ runs DataSelect ▷ DataSelect calls **FeatureSimilarity** 2 or **OMP** 3 depending on whether L_{DO} is differentiable
4: $M'_{\text{MT}} = \texttt{TRAIN}(D^{\text{MT}} \cup D_i^{\text{useful}})$.

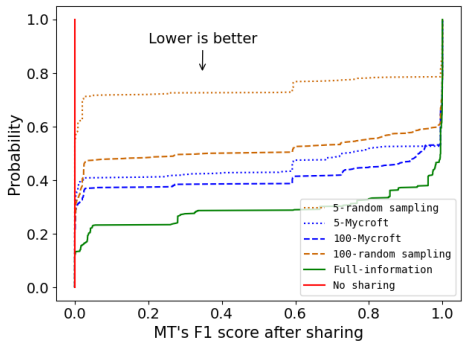
Overview of approach: The overall approach is described in Algorithm 1. After the D^{hard} is identified, a small subset is sent to DOs while the rest is reserved for testing. The key technical challenge we address in this section is that of designing the subroutine DataSelect. Our first approach uses feature similarity in the raw data space or feature space of a deep neural network (DNN) whereas our second approach uses gradient similarity in the model space of a DNN to find functionally similar samples to D^{hard}.

3.1 Approach 1 : Feature Similarity

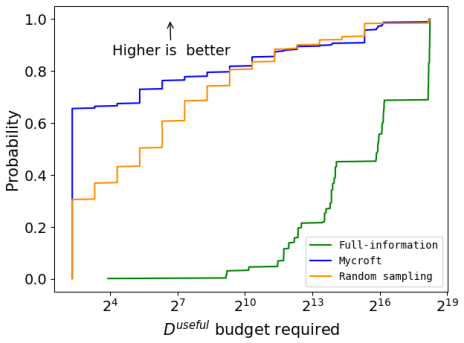
Intuitively, data samples that are similar to samples from D^{hard} would be useful for the MT. Given a good feature extractor $\phi(\cdot)$ that maps a sample x_j to its feature representation $\phi(x_j)$, we compute the distances of each sample in D^{hard} to each sample in D_i and construct D_i^{useful} by using a greedy heuristic which selects the top-k samples having the minimum distance to D^{hard} samples. (see Algorithm 2). The choice of feature representations $\phi(\cdot)$ and distance function $d(\cdot, \cdot)$ is contingent on the domain of the data. For the IoT dataset, we use our **ExtractBinningFeatures** algorithm (Appendix 4) whereas for the image datasets, we use L2 distances in the feature space of Unicom [3].

3.2 Approach 2 : Loss Gradient Similarity

We consider the approach of selecting data from D_i that is similar in a loss gradient space to D^{hard}. For models trained with a differentiable loss function, the gradient of the loss with respect to a model's parameters at each sample indicates how it impacted the model during training. Samples with similar gradients are functionally similar. We formulate the problem of finding D_i^{useful} as that of obtaining a k-sparse weight vector $\mathbf{w}$ over D_i, with the weight assigned to each sample corresponding



(a) CDF of MT's F1 score after data sharing. Mycroft outperforms random-sampling in a budget matched setting.



(b) CDF of D_i^{useful} required for random-sampling and Mycroft to match full-information N = 474 cases where sharing data is helpful.

Figure 1: Performance of Mycroft, random-sampling and full-information on the IoT dataset.

to its utility. This formulation is inspired by [20], who use it for dataset compression. To find the optimal $1 - e^{\gamma\ *}\ k$-sparse $\mathbf{w}$, we: (1) find the averaged gradient of the loss L computed on D^{hard} with respect to the parameters θ_i of M_i (denoted $\nabla_{\theta_i} L(D^{\text{hard}})$); (2) compute the gradient of the loss L computed on each sample $z_j \in D_i$ with respect to the parameters θ_i of M_i; (3) solve the following regularized optimization problem:

$$\min_{\|\mathbf{w}\|_0 \leq k} e_\lambda(\mathbf{w}) = \min_{\|\mathbf{w}\|_0 \leq k} \left\| \sum_{z_j \in D_i} \mathbf{w}_j \nabla_{\theta_i} L(z_j) - \nabla_{\theta_i} L(D^{\text{hard}}) \right\| + \lambda \|\mathbf{w}\|_2^2 . \tag{1}$$

The ℓ_0-"norm" constraint on $\mathbf{w}$ enforces sparsity but leads to an NP-hard problem. To tackle this issue, we can use a greedy algorithm, Orthogonal Matching Pursuit (OMP) [28], to find a close approximation due to the sub-modularity of $e_\lambda(\mathbf{w})$ [9] [*]. We detail OMP in Algorithm 3.

We also develop an approach which unifies feature similarity and loss gradient similarity. We explain it further along with its proof of optimality in Appendix B.

4 Experimental Setup

We evaluate Mycroft on classification tasks over two domains: network traffic classification and computer vision. We use 5 datasets for the latter and a tabular dataset for the former representing flow features of network traffic. Further details are in Appendix D.

Tabular dataset: The networking dataset is sourced from the IoT-23 dataset [14]. To address privacy concerns, MTs and DOs only share derived tabular features of their network traffic flows, rather than raw data. The MT selects DOs to improve its detection of malicious traffic. We evaluate 665 combinations of MTs, DOs, and attack types for a thorough analysis.

Image datasets:

- **Food datasets:** We use three datasets from the food computing domain: Food-101 [4], UPMC Food-101 [32] and ISIA Food-500 [32]. We assign Food-101 to be the MT's dataset and UPMC Food-101 and ISIA Food-500 as datasets of two DOs.
- **Dog datasets:** We use two datasets for dog breed classification: Imagenet-Dogs [8] (MT) and Tsinghua-Dogs [36] (DO).
- **Dogs & Wolves:** We curate a dataset of Dogs & Wolves which contains spurious correlations called Dogs & Wolves - Spurious (MT). However, the MT's model performs poorly on data which does not contain the spurious correlations called Dogs & Wolves - Natural (DO). An illustration of this dataset is provided in Appendix 4.

[*] γ is the regularized maximum norm of the gradient of the loss function and the proof of the optimality is in [20]

[*] This use of OMP is inspired by [20], who use it for dataset compression during training.

Budget k	Food-101 - UPMC		Food-101 - ISIA500		Imagenet - Tsinghua		Dogs & Wolves - Spurious - Dogs & Wolves - Natural	
	`Mycroft`	`random-sampling`	`Mycroft`	`random-sampling`	`Mycroft`	`random-sampling`	`Mycroft`	`random-sampling`
8	0.42	0.36	0.33	0.18	0.43	0.31	0.44	0.13
16	0.50	0.38	0.48	0.33	0.62	0.48	0.50	0.13
32	0.61	0.47	0.54	0.40	0.70	0.56	0.75	0.25
64	0.75	0.64	0.72	0.61	NA	NA	0.75	0.25
128	0.86	0.72	0.83	0.70	NA	NA	0.88	0.38

Table 1: Accuracy of M'_{MT} with varying budgets of D_i^{useful}. `full-information` results in 100% accuracy on D^{hard}. Column headers indicate the `MT` and `DO` datasets separated by hyphens.

Models: For the IoT dataset, we use Decision Trees, XGBoost and Random Forests. For the image datasets, all our experiments use ResNet50 [16] models pre-trained on Imagenet. For feature similarity matching for image data, we use the feature space of Unicom [3]. Appendix D.2 contains details for the training procedure.

Metrics, Baselines & Evaluation setup: We use F1 score for the IoT dataset (due to class imbalance) for D^{hard} and report classification accuracy for the image datasets. Our baseline techniques are `random-sampling`, which uses class label knowledge, and `full-information`. Random sampling has proven effective for dataset compression [25, 15, 26, 21], while `full-information` often provides the upper bound on data utility.

5 Results

For the IoT data, we use only feature similarity since loss gradient similarity cannot be applied, while for image data, we apply gradient similarity with a minor contribution from feature similarity.

IoT Dataset: From Fig 1a, we see that D_i^{useful} budget of 5 samples retrieved using `Mycroft` outperforms D_i^{useful} of 100 samples selected by `random-sampling`. Furthermore, if the D_i^{useful} budget doubles until it reaches `full-information` performance (Fig 1b), `Mycroft` reaches the `full-information` performance using a smaller D_i^{useful} budget compared to `random-sampling`. This shows `Mycroft` can help the data owner to efficiently identify relevant data sources for intrusion detection. More results are in Appendix E.

Image Datasets: We present results for the image datasets Table 1 and make three key observations. First, we can see that `Mycroft` outperforms `random-sampling` at all budgets of D_i^{useful}. Secondly, we note that `Mycroft` can rapidly converge to the `full-information` setting using only a fraction of the dataset. Finally, the performance gap between `random-sampling` and `Mycroft` for the Dogs & Wolves dataset highlights that when a `DO` has a very small subset of data useful for the `MT`, `Mycroft` can retrieve that subset very efficiently. We also present results for the situation where the `DO` has corrupted data, noisy labels or when multiple `DO`s need to be ranked in Appendix F.

6 Discussion and Related Work

Related Work: Several studies have explored improving training data quality by using existing datasets through techniques like image overlay, random erasure, and generative models to generate new samples [18, 34, 7, 31, 17, 11, 18, 19]. However, these methods can fall short if the available training data is insufficient for training a generative model, or if certain distributions are significantly underrepresented in the data. To address this, research has focused on sourcing data from publicly available data lakes [27, 5, 33, 10, 30, 35, 6, 12, 13]. However, these studies do not address the challenge of acquiring data from private entities, where data-sharing restrictions are much stricter. Another related area of research [22, 23, 15] focuses on creating "coresets"—subsets of data that can approximate the cost function for the entire dataset. However, our study is not concerned with approximating the entire dataset but rather finding a small subset of the data which will be useful for some specific tasks. Therefore, these methods are not applicable to our research.

Limitations: In certain cases, model trainers may not be willing to share even small subsets of their data with the data owners, in which case more privacy-preserving methods such as noise addition or feature sharing can be explored.

References

[1] URL https://www.unb.ca/cic/datasets/ids-2018.html.

[2] Pytorch transforms. *arXiv preprint arXiv:2304.02643*. URL https://pytorch.org/vision/0.9/transforms.html.

[3] Xiang An, Jiankang Deng, Kaicheng Yang, Jaiwei Li, Ziyong Feng, Jia Guo, Jing Yang, and Tongliang Liu. Unicom: Universal and compact representation learning for image retrieval. *arXiv preprint arXiv:2304.05884*, 2023.

[4] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014.

[5] Sonia Castelo, Rémi Rampin, Aécio Santos, Aline Bessa, Fernando Chirigati, and Juliana Freire. Auctus: a dataset search engine for data discovery and augmentation. *Proc. VLDB Endow.*, 14(12):2791–2794, jul 2021. ISSN 2150-8097. doi: 10.14778/3476311.3476346. URL https://doi.org/10.14778/3476311.3476346.

[6] Raul Castro Fernandez, Jisoo Min, Demitri Nava, and Samuel Madden. Lazo: A cardinality-based method for coupled estimation of jaccard similarity and containment. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1190–1201, 2019. doi: 10.1109/ICDE.2019.00109.

[7] Phillip Chlap, Hang Min, Nym Vandenberg, Jason Dowling, Lois Holloway, and Annette Haworth. A review of medical image data augmentation techniques for deep learning applications. *Journal of Medical Imaging and Radiation Oncology*, 65(5):545–563, 2021.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. Restricted strong convexity implies weak submodularity. *arXiv preprint arXiv:1612.00804*, 2016.

[10] Mahdi Esmailoghli, Jorge-Arnulfo Quiané-Ruiz, and Ziawasch Abedjan. Cocoa: Correlation coefficient-aware data augmentation. In *International Conference on Extending Database Technology*, 2021. URL https://api.semanticscholar.org/CorpusID:232283631.

[11] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.

[12] Raul Castro Fernandez. *Aurum: a story about research taste*, page 387–391. Association for Computing Machinery and Morgan & Claypool, 2018. ISBN 9781947487192. URL https://doi.org/10.1145/3226595.3226631.

[13] Sainyam Galhotra, Yue Gong, and Raul Castro Fernandez. Metam: Goal-oriented data discovery, 2023.

[14] Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga. IoT-23: A labeled dataset with malicious and benign IoT network traffic. 2020. doi: 10.5281/zenodo.4743746. URL http://doi.org/10.5281/zenodo.4743746.

[15] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pages 181–195. Springer, 2022.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Zeshan Hussain, Francisco Gimenez, Darvin Yi, and Daniel Rubin. Differential data augmentation techniques for medical imaging classification tasks. In *AMIA annual symposium proceedings*, volume 2017, page 979. American Medical Informatics Association, 2017.

[18] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[19] Xi Jiang, Shinan Liu, Aaron Gember-Jacobson, Arjun Nitin Bhagoji, Paul Schmitt, Francesco Bronzino, and Nick Feamster. Netdiffusion: Network data augmentation through protocol-constrained traffic generation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 8(1):1–32, 2024.

[20] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR, 2021.

[21] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8110–8118, 2021.

[22] Krishnateja Killamsetty, Xujiang Zhao, Feng Chen, and Rishabh Iyer. Retrieve: Coreset selection for efficient and robust semi-supervised learning. *Advances in Neural Information Processing Systems*, 34:14488–14501, 2021.

[23] Yeachan Kim and Bonggun Shin. In defense of core-set: A density-aware core-set selection for active learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 804–812, 2022.

[24] Not Listed. Nfstream: Flexible network data analysis framework. `https://www.nfstream.org/`, 2024.

[25] Mohammad Sultan Mahmud, Joshua Zhexue Huang, Salman Salloum, Tamer Z Emara, and Kuanishbay Sadatdiynov. A survey of data partitioning and sampling methods to support big data analysis. *Big Data Mining and Analytics*, 3(2):85–101, 2020.

[26] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020.

[27] Fatemeh Nargesian, Ken Pu, Bahar Ghadiri-Bashardoost, Erkang Zhu, and Renée J. Miller. Data lake organization. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):237–250, 2023. doi: 10.1109/TKDE.2021.3091101.

[28] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pages 40–44. IEEE, 1993.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[30] Aécio Santos, Aline Bessa, Fernando Chirigati, Christopher Musco, and Juliana Freire. Correlation sketches for approximate join-correlation queries. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 1531–1544, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383431. doi: 10.1145/3448016.3458456. URL `https://doi.org/10.1145/3448016.3458456`.

[31] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[32] Xin Wang, Devinder Kumar, Nicolas Thome, Matthieu Cord, and Frederic Precioso. Recipe recognition with large multimodal food dataset. In *2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2015.

[33] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. Infogather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, page 97–108, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450312479. doi: 10.1145/2213836.2213848. URL https://doi.org/10.1145/2213836.2213848.

[34] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

[35] Erkang Zhu, Fatemeh Nargesian, Ken Q. Pu, and Renée J. Miller. Lsh ensemble: internet-scale domain search. *Proc. VLDB Endow.*, 9(12):1185–1196, aug 2016. ISSN 2150-8097. doi: 10.14778/2994509.2994534. URL https://doi.org/10.14778/2994509.2994534.

[36] Ding-Nan Zou, Song-Hai Zhang, Tai-Jiang Mu, and Min Zhang. A new dataset of dog breed images and a benchmark for finegrained classification. *Computational Visual Media*, 6:477–487, 2020.

In this Appendix, we aim to (i) provide proofs for the optimality of `Mycroft` (ii) provide more details about the datasets, algorithms and models used to obtain the results in the main body of the paper (iii) present additional experiments to further validate the discoveries in the main body of the paper. The Appendix is organized as follows:

1. Summary of symbols and notations used (Appendix A)
2. Description and Proof of the optimality of our joint optimization objective`Mycroft` (Appendix B)
3. Further details about the experiment setup including the datasets and models used (Appendix D)
4. Additional results and ablation studies (Appendix E)
5. Details and results for case studies (Appendix F)
6. Runtime analysis, total compute and algorithms for `Mycroft` subroutines (Appendix C)

A Symbols and Notations

Symbol	Description
`DO`	Data Owner
`MT`	Model Trainer
D^{MT}	`MT`'s dataset
M_{MT}	`MT`'s model
D^{test}	`MT`'s test dataset
D_i	i^{th} `DO`'s dataset
D_i^{useful}	Subset of D_i retrieved by `Mycroft` or `random-sampling`
D^{hard}	Subset of D^{test} which is incorrectly classified and is shared with the `DO`s

Table 2: Table of notations used in the paper.

B Proof for the optimality of our joint optimization objective

B.1 Combining Feature Similarity and Gradient Matching

In the case where the `DO` has access to a well-trained model as well as a good feature extractor, both notions of similarity can be used. We do this by adding a regularization term in terms of a

composite norm that incorporates the feature similarity between samples from D^{hard} and D_i to the approximation error in Eq. 1:

$$e'_{\lambda_1,\lambda_2}(\mathbf{w}) = e_{\lambda_1}(\mathbf{w}) + \lambda_2 \|\Psi\mathbf{w}\|_2^2, \tag{2}$$

where Ψ is a matrix of distances. The second term functions as a regularizer that penalizes samples that are far from feature representations of D^{hard}. In the following theorem, we show sub-modularity:

Theorem B.1. *If the loss function $L(\cdot)$ is bounded above by L_{max} and $\forall j,\ \|\nabla_{\theta_i} L(z_j)\| \leq \nabla_{max}$, then $f_{\lambda_1,\lambda_2}(\mathbf{w}) = L_{max} - e'_{\lambda}(\mathbf{w})$ is weakly submodular with parameter $\gamma' \geq \frac{\lambda_1+\lambda_2\|\Psi\|_2^2}{\lambda_1+\lambda_2\|\Psi\|_2^2+k\nabla_{\max}^2}$,*

where $\|\cdot\|_2$ is the spectral norm for matrices. Again, from [9], we get that OMP returns a $1-e^{\gamma'}$-close approximation of the maximum value of $f_{\lambda_1,\lambda_2}(\mathbf{w})$ and we present the proof for this below. In essence, we need to prove that the following function is weakly submodular with paramter γ':

$$f_{\lambda_1,\lambda_2}(\mathbf{w}) = L_{\max} - \min_{\|\mathbf{w}\|_0 \leq k} \left\| \sum_{z_j \in D_i} \mathbf{w}_j \nabla_{\theta_i} L(z_j) - \nabla_{\theta_i} L(D^{\text{hard}}) \right\| + \lambda_1 \|\mathbf{w}\|_2^2 + \lambda_2 \|\Psi\mathbf{w}\|_2^2, \tag{3}$$

under the conditions specified in Theorem B.1. Given that this function is submodular, then the use of the Orthogonal Matching Pursuit (OMP) algorithm from [9] will return a k-sparse subset with performance that is a $1-e^{\lambda'}$ approximation of the maximum value.

Proof of Theorem B.1. From Elenberg et al. [9], a function is γ' weakly submodular with $\gamma' \geq \frac{m}{M}$ where m is the restricted strong concavity parameter and M is the restricted smoothness parameter.

To prove that $f_{\lambda_1,\lambda_2}(\mathbf{w})$ is strongly concave with parameter m, we need to show that

$$-\frac{m}{2}\|\mathbf{v}-\mathbf{w}\|_2^2 \geq f_{\lambda_1,\lambda_2}(\mathbf{v}) - f_{\lambda_1,\lambda_2}(\mathbf{w}) - \langle \nabla f_{\lambda_1,\lambda_2}(\mathbf{w}), \mathbf{v}-\mathbf{w}\rangle \tag{4}$$

Plugging in $f_{\lambda_1,\lambda_2}(\cdot)$ from Eq. 1, we get

$$\begin{aligned} -\frac{m}{2}\|\mathbf{v}-\mathbf{w}\|_2^2 &\geq -\lambda_1\|\mathbf{v}-\mathbf{w}\|_2^2 - \lambda_2\|\Psi\mathbf{v}-\Psi\mathbf{w}\|_2^2 \\ &\geq -\lambda_1\|\mathbf{v}-\mathbf{w}\|_2^2 - \lambda_2\|\Psi\|_2^2\|\mathbf{v}-\mathbf{w}\|_2^2, \end{aligned}$$

where the final inequality arises from the property of the induced norm with respect to a matrix and $\|\Psi\|$ is the spectral norm of the distance matrix Ψ. This implies $m \leq 2(\lambda_1 + \lambda_2\|\Psi\|_2^2)$.

To prove that $f_{\lambda_1,\lambda_2}(\mathbf{w})$ is restricted smooth with parameter M, we need to show that

$$f_{\lambda_1,\lambda_2}(\mathbf{v}) - f_{\lambda_1,\lambda_2}(\mathbf{w}) - \langle \nabla f_{\lambda_1,\lambda_2}(\mathbf{w}), \mathbf{v}-\mathbf{w}\rangle \geq -\frac{M}{2}\|\mathbf{v}-\mathbf{w}\|_2^2 \tag{5}$$

Expanding the term on the L.H.S. again, we get,

$$\begin{aligned} &-\lambda_1\|\mathbf{v}-\mathbf{w}\|_2^2 - \lambda_2\|\Psi\|_2^2\|\mathbf{v}-\mathbf{w}\|_2^2 - \sum_j \mathbf{v}_j(\sum_k (\mathbf{w}_k - \mathbf{v}_j)\nabla_{\theta_i}(z_j)^\intercal \nabla_{\theta_i}(z_k)) \\ &\geq -\lambda_1\|\mathbf{v}-\mathbf{w}\|_2^2 - \lambda_2\|\Psi\|_2^2\|\mathbf{v}-\mathbf{w}\|_2^2 - k\nabla_{\max}^2\|\mathbf{v}-\mathbf{w}\|_2^2, \end{aligned}$$

where the final inequality arises from the $k-$ sparse condition on the weight vectors and the bound on the gradients of the loss function. This gives $M \geq 2(\lambda_1 + \lambda_2\|\Psi\|_2^2 + k\nabla_{\max}^2)$.

Together, this gives $\gamma' \geq \frac{\lambda_1+\lambda_2\|\Psi\|_2^2}{\lambda_1+\lambda_2\|\Psi\|_2^2+k\nabla_{\max}^2}$.

□

C Pseudo code and runtime analysis for `Mycroft`

C.1 Pseudo code for subroutines which `Mycroft` uses

[AB: There is no text here! Need to refer to the subroutines and how they relate to each other.]

Algorithm 2 FeatureSimilarity

Require: D^{hard}, D_i, Budget k,
$\phi(D^{\text{hard}}), \phi(D_i^{\text{useful}}) \leftarrow$ DO runs **FeatureExtractor**(D^{hard}, D_i) ▷ **Unicom** or **Binnning**
$\Psi \leftarrow$ **ComputeDistances**$(\phi(D^{\text{hard}}), \phi(D_i^{\text{useful}}))$
$D_i^{\text{useful}} \leftarrow$ **RetrieveTopK**(Ψ, k)
return D_i^{useful}

Algorithm 3 OMP

Require: D^{hard}, DO's loss function : L, D_i, M_i's parameteres θ, regularization coefficients: λ_1,λ_2 , subset size: k, tolerance: ϵ
$\mathcal{X} \leftarrow \emptyset$
$r \leftarrow \nabla_{\theta_i} L(D^{\text{hard}})$
while $\mathcal{X} \leq k$ and $r \geq \epsilon$ **do**
  $m \leftarrow \arg\max_j |Proj(\nabla_{\theta_i} L(D_i), r)|$
  $\mathcal{X} \leftarrow \mathcal{X} \cup \{m\}$
  $w^* \leftarrow \arg\min_w e'_{\lambda_1,\lambda_2}(w, \mathcal{X})$
  $r \leftarrow r - Proj(\mathcal{X}, w^*)$
end while
return $\mathcal{X}$, w

C.2 Complexity & Runtime of Mycroft

Image datasets: Mycroft for the image domain consists of two techniques: Unicom and GradMatch. Here, we discuss the computation and memory complexity of both these techniques in order to give a sense of their efficiency and practicality. Unicom operates by projecting all data points in the representation space of the Unicom model, which is based on CLIP [29], and computing distances between those data points. Thus, its compute and memory requirement scale in proportion to the number of data points to be projected as each sample requires a forward pass through the model to acquire its feature representation which needs to be held in stored for computing distances with other data points. Empricially, we find that this procedure takes less than 5 minutes and requires less than 4 GB of GPU memory for each experiment we present in this paper. GradMatch requires computing gradients for each data point in D^{hard} and DO's using the DO's model once. In practice, we only use the gradients of the last two layers, which significantly reduces the compute and memory requirements. The gradients are then used to run the OMP algorithm which has a complexity of $\mathcal{O}(NM + Mk + k^3)$ for each of the k iterations where k is $|D_i^{\text{useful}}|$, M is the dimension of the gradients and N is $|D_i|$. For experiments in this paper, each experiment ran in under 10 minutes and required approximately 6 GigaBytes of memory.

Algorithm 4 BinningDistance

Require: D^{hard}, DO, percentage to sample from DO: r, minimum number of non-empty bins for each feature: b, binning candidates list (in increasing order): $candidates$, features used for binning: $features$
$\mathcal{X}^{\text{DO}}, \mathcal{X}^{D^{\text{hard}}} \leftarrow$ ExtractBinningFeatures$(D^{\text{hard}}, \text{DO}, r, b, candidates, features)$
$D \leftarrow \emptyset$
for p^{DO} in $\mathcal{X}^{\text{DO}}$ **do**
  $d\_l \leftarrow \emptyset$
  for $p^{D^{\text{hard}}}$ in $\mathcal{X}^{D^{\text{hard}}}$ **do**
    $d \leftarrow$ GetDistance$(p^{\text{DO}}, p^{D^{\text{hard}}})$
    $d\_l \leftarrow d\_l \cup \{d\}$
  end for
  $D \leftarrow D \cup \{d\_l\}$
end for
return D

Algorithm 5 ExtractBinningFeatures

Require: D^{hard}, DO, percentage to sample from DO: r, minimum number of non-empty bins for each feature: b, binning candidates list (in increasing order): $candidates$, features used for binning: $features$

DO$^{\text{samples}} \leftarrow$ sample(DO, r)
UnionSamples $\leftarrow D^{\text{hard}} \cup$ DO$^{\text{samples}}$
$\mathcal{X}^{\text{DO}} \leftarrow \emptyset$
$\mathcal{X}^{D^{\text{hard}}} \leftarrow \emptyset$
for f in $features$ **do**
    NumBin $\leftarrow \max(candidates)$
    for n in $candidates$ **do**
        NumFilled $\leftarrow$ CountNonEmptyBins(UnionSamples$[f], n)$
        if NumFilled $\geq b$ **then**
            NumBin $\leftarrow b$
            break
        end if
    end for
    edge$^f \leftarrow$ GetEdge(UnionSamples$[f]$, NumBin)
    $\mathcal{X}^{\text{DO}} \leftarrow \mathcal{X}^{\text{DO}} \cup$ GetBinningCoordinates(DO$[f]$, edge^f)
    $\mathcal{X}^{D^{\text{hard}}} \leftarrow \mathcal{X}^{D^{\text{hard}}} \cup$ GetBinningCoordinates($D^{\text{hard}}[f]$, edge^f)
end for
return $\mathcal{X}^{\text{DO}}$, $\mathcal{X}^{D^{\text{hard}}}$

Tabular dataset: The runtime for tabular dataset as described in 4 is $\mathcal{O}(MNF)$ where M is the number of samples in DO, N is the number of samples in D^{hard}, F is the number of features used for *ExtractBinningFeatures*. Empirically, each experiment takes less than 5 minutes and requires less than 3 GB of memory.

D Further setup details

MT will evaluate the utility of DO's data based on the framework found in Figure 2. Further details about the datasets and training process used to obtain the results in the main body of the paper are provided in this section.

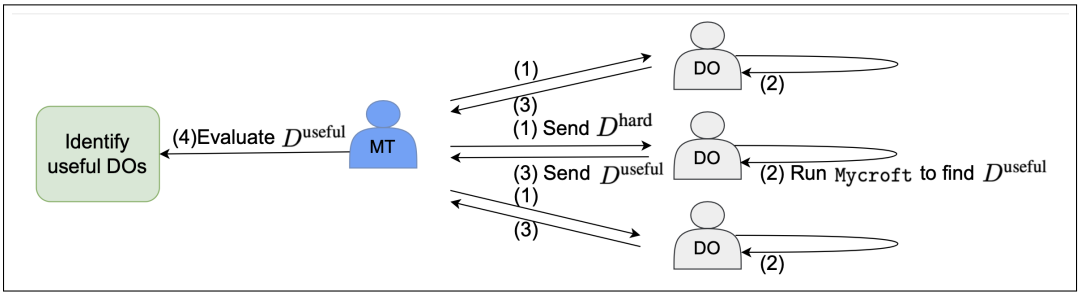


Figure 2: Framework for MT to evaluate the utility of DO's data.

D.1 Datasets

D.1.1 Dogs & Wolves dataset

Neural networks are known to learn spurious correlations in supervised settings. While test data containing the correlations learnt during training often gets classified correctly, data which does not contain such correlations is prone to misclassification. We exploit this phenomenon to create a dataset which helps us simulate a controlled MT-DO interaction. In our case, the MT has a training and validation dataset which contains spurious correlations but a test dataset which does not contain them

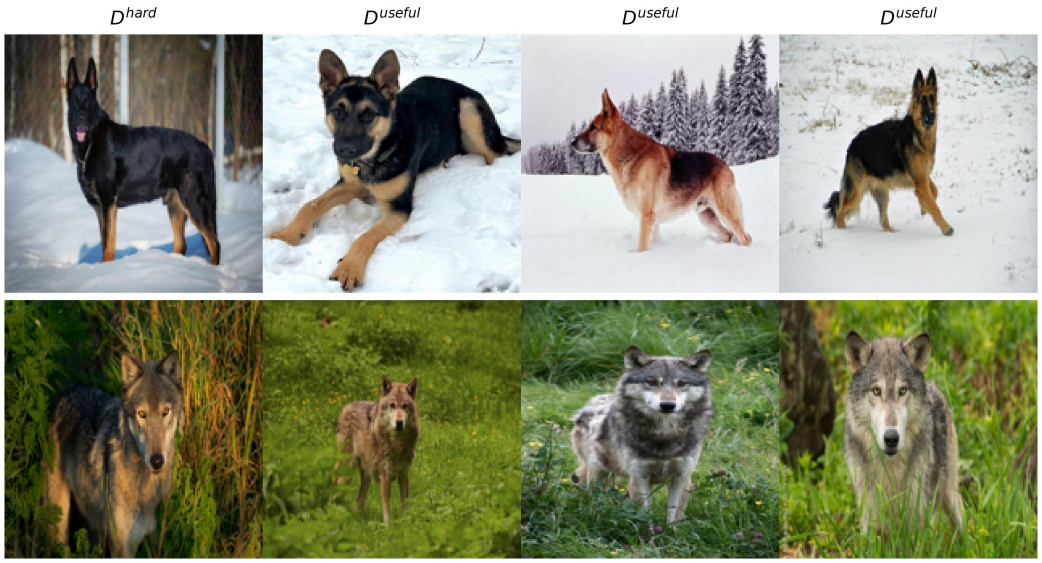


Figure 3: Top-k retrieved D_i^{useful} samples using Unicom for D^{hard} from the Dogs & Wolves dataset.

Figure 4: Subsets in the Dogs & Wolves dataset. The first column shows Dogs & Wolves - Spurious where the dogs are on a grass background and the wolves are on snow. The second column shows Dogs & Wolves - Natural where the dogs are on snow and the wolves are on grass.

and thus their model suffers on the test dataset.
Concretely, we curate a dataset which consists of two classes: Dogs and Wolves. Spurious correlations are introduced in it by controlling the background of each image which can either be snow or grass. The MT has data from both animals being on one type of background. In particular, the dogs are on grass and the wolves are on snow. In the absense of negative examples, the model takes a shortcut by associating the true label with the background and not the animal. We refer to MT's training and validation subset as Dogs & Wolves - Spurious. However, the model performs poorly when the test samples do not contain the spurious correlations i.e., dogs on snow and wolves on grass. We refer to this subset as Dogs & Wolves - Natural. We simulate a DO which has a dataset containing both Dogs & Wolves - Spurious and Dogs & Wolves - Natural and thus their model does not learn background related spurious correlations. An illustration of this dataset is provided in Figure 4.
Now, if the MT wants to perform well on data from Dogs & Wolves - Natural, they must acquire data from that distribution and retrain their model in order to break the spurious correlations. This motivates the MT to acquire new data form the DO. It should be noted that the MT is oblivious as to why their model performs poorly on Dogs & Wolves - Natural.

D.1.2 Food dataset

Food-101 contains 101,000 images of 101 food classes with 750 and 250 images for each category for training and testing. UPMC Food-101 is a twin dataset to Food-101 and thus has the same number of categories and size as Food-101. ISIA Food-500 is another food recognition dataset with approximately 400,000 images for 500 food classes and has many classes which intersect with the set of classes in UPMC Food-101. In our experiments, we use Food-101 as the MT's dataset and UPMC Food-101 and ISIA Food-500 as datasets of two different DO's.

D.1.3 Tabular dataset

Data source:
The dataset we use consists of five captures (scenarios) of different IoT network traffic [14]. Each network consists of traffic of two types : benign traffic (when the IoT devices are not not under attack) and malicious traffic (when the devices are under attack). The attacks are executed in a Raspberry Pi and each capture can suffer from different attacks. Details about the types of benign/attack present in each capture can be found in Table 3.

Table 3: Attacks present in each capture, Benign stands for Benign traffic. PHP stands for Part Of A Horizontal PortScan attack, CC stands for C&C attack, CCT stands for C&C Torii.

Capture	Benign/Attacks present
1	Benign, PHP,
3	Benign, CC, PHP
20	Benign, CCT
21	Benign, CCT
34	Benign, CC, PHP

Data Labeling:
From the raw pcap files (which contain the raw network traffic data), we use the Python library NFStream [24] to extract feature flows (in tabular format) from pcap files. We then match the timing of the flow with the timing that the attack was executed as mentioned in the data source [14] to label the data. After labelling the flows, we split these flows based on whether they are benign or malicious based on their attack type.

Model Trainers: We define an MT to be the IoT device in the captures that want to improve its model's ability to predict some particular attack. In this study, we have 7 different MTs. In addition, because the attack data in this dataset has quite uniform distribution (most likely because they are conducted in a lab-setting) such that if the model has been trained on the attack, they are very likely to predict a future attack of the same type with high accuracy, we assume that these MTs are only trained on benign data. In reality, this scenario is possible because if the network is new, the chances of them being trained on attack data for this new network is low. The MTs see a very small number of attack data which its model fail to predict and would like to get more data from DOs to improve their models.

Data Owners: We artificially inflate the number and complexity of DOs by mixing data from different captures to generate 95 new DOs. This will increase the difficulty of finding relevant samples in DOs and simulate the real scenarios where DO are often quite complex.

Details for D^{hard}: The quantity of D^{hard} which each MT possesses is very small (2% of the malicious data) and is chosen randomly from the malicious data of the MT's dataset.

D.2 Training details

Here, we provide more details about the training procedure we used for obtaining the neural networks we use for our computer vision tasks.

For the public image datasets we use, we observe that the model performs well on most classes perform and thus, there is little to gain from external data augmentation. Therefore, to simulate a more realistic setting, we reduce performance on certain classes of the MT's model by training them

with limited training data. On average, we use 10% of the original training data for the classes we choose to augment using external datasets and attain an average accuracy of 65% for them.

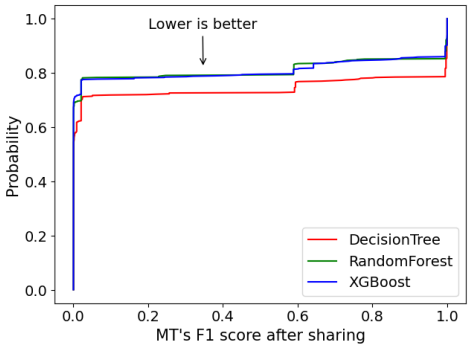
The MT's and DO's models are ResNet50 models pretrained on Imagenet and finetuned on their respective datasets. The MT's and DO's base model is trained for 120 epochs using a learning rate of 0.03 with a cosine annealing weight decay. The MT's augmented models, M'_{MT}, are obtained by finetuning their base models for 25 epochs on D_i^{useful} and their original training dataset.

E Additional results & Ablation studies

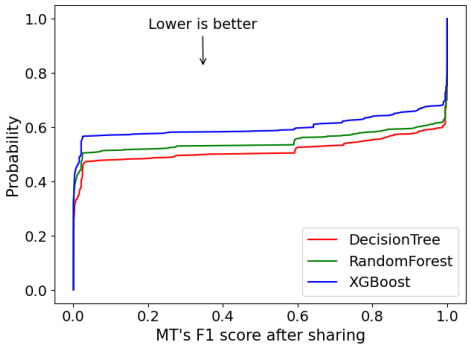
E.1 Tabular-data

For this dataset, as noted in §5, the results presented only use feature similarity. We discuss the reasons here. First, the models that perform the best on this dataset are tree-based classifiers for which gradient matching does not apply. In addition, to resemble the effect of gradient matching for tabular data, we have also attempted to retrieve D_i^{useful} based on DO's model confidence score and DecisionTree's decision path when trained on DO's data. We find that the performance of these approaches are not as good as Mycroft. Potential reasons for why these approaches do not work are (i) features that differentiate benign and malicious traffic for DO might not be features that are important for MT's model (as can be seen in the fact that for some DOs, MT's model does not improve after data sharing) and (ii) the DecisionTree's decision path when trained on DO's data might be over-reliant on only one or very few features, hence, do not provide useful signals to select D_i^{useful}.

Performance with different classifiers:
In this section, we present MT's F1 score after data sharing using `random-sampling` and Mycroft for different classifiers. We find that DecisionTree seems to be the best classifier for this dataset. Refer to Figure 5, 6 and 7 for the results.



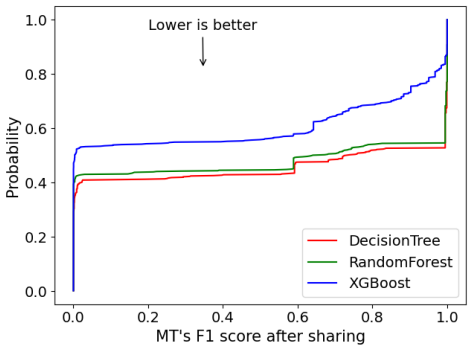
(a) CDF of MT's F1 score after data sharing using `random-sampling` with D_i^{useful} budget of 5 samples for different classifiers. DecisionTree seems to be the best classifier for this dataset.



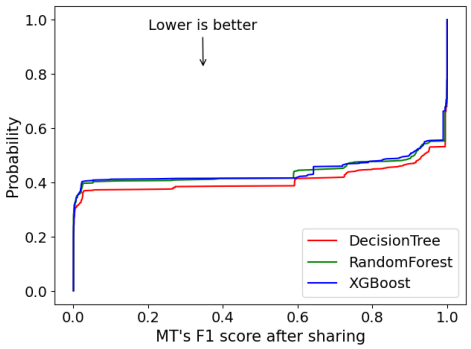
(b) CDF of MT's F1 score after data sharing using `random-sampling` with D_i^{useful} budget of 100 samples for different classifiers. DecisionTree seems to be the best classifier for this dataset.

Figure 5: Performance of `random-sampling` for different classifiers for the tabular dataset.

Performance when D_i^{useful} is selected based on different data selection algorithm:
To explore whether DecisionTree's decision path can be used to select D_i^{useful}, we use D_i^{useful} budget of 5 samples and compare MT's F1 score after data sharing when D_i^{useful} is retrieved from samples of the same decision path as D^{hard}, of different decision paths as D^{hard}, retrieved from `random-sampling` and Mycroft. Note that to make this study comparable, we only consider cases where samples of the same decision path as D^{hard} and samples of different decision path as D^{hard} can be found. This total up to 466 cases. We find that although sharing samples of same decision paths as D^{hard} can be slightly better than `random-sampling`, Mycroft still outperforms this approach significantly. Refer to Figure 8 for the results.



(a) CDF of MT's F1 score after data sharing using Mycroft with D_i^{useful} budget of 5 samples for different classifiers. DecisionTree seems to be the best classifier for this dataset.



(b) CDF of MT's F1 score after data sharing using Mycroft with D_i^{useful} budget of 100 samples for different classifiers. DecisionTree seems to be the best classifier for this dataset.

Figure 6: Performance of Mycroft for different classifiers for the tabular dataset.

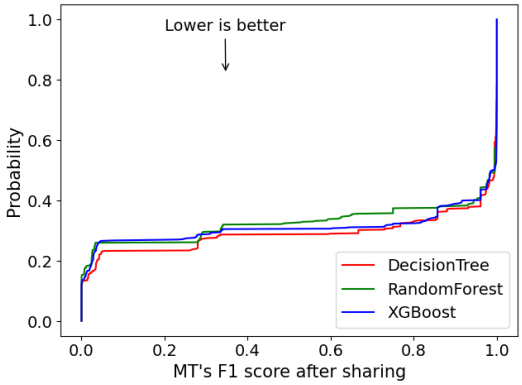


Figure 7: CDF of MT's F1 score after data sharing for full-information for different classifiers. DecisionTree seems to be the best classifier for this dataset.

Performance of combining Mycroft and other data selection methods: To explore the effects of combining Mycroft and other data selection methods such as DecisionTree's decision path and random-sampling, we let D_i^{useful} to be made up of samples selected by Mycroft and samples selected by these other data selection methods. We find that MT's F1 score after data sharing is not significantly improved when combining Mycroft with other data selection methods compared to using Mycroft alone. Refer to Figure 10 for an example of the results.

F Case Studies

Having evaluated Mycroft in our main setup, we now explore its use in three other realistic scenarios. The first two are evaluated using vision datasets only whereas the third scenario includes evaluation on both vision and tabular datasets.

Scenario 1 - Corrupted data features: In a real world setting, data features can often be corrupted for various reasons such as hardware failures, transmission errors, etc. Since Mycroft is designed to operate on large scale real-world datasets, we are interested in understanding its ability to retrieve useful subsets from a dataset with corrupted features.

For evaluation, we corrupt the DO's dataset using PyTorch's *transforms* module which includes random masking, color jitters etc. and display the results in Figure 9. We see that only deteriorates the

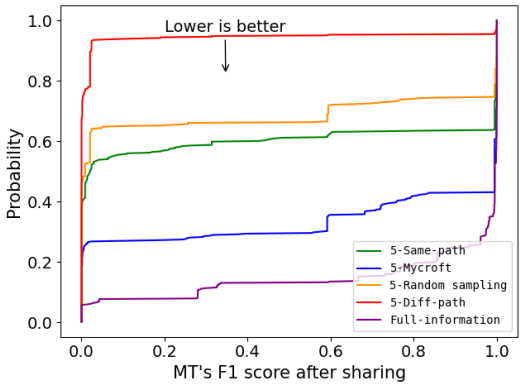


Figure 8: CDF of MT's F1 score after data sharing using different data selection methods with D_i^{useful} budget of 5 samples and using `full-information`. Classifier is DecisionTree. N = 466 cases where samples of the same decision path as D^{hard} and samples of different decision path as D^{hard} can be found.

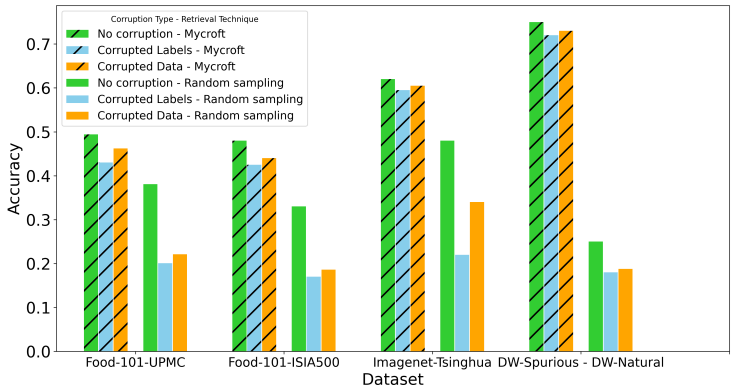


Figure 9: Accuracy of M'_{MT} when trained on D_i^{useful} retrieved using `Mycroft` and `random-sampling` under the scenario where approximately 70% of the data or labels are corrupted.

performance of `Mycroft` by an average of 2.7% over its baseline setting with no corruption whereas the performance of `random-sampling` deteriorates, on average, by 13.7%. This further validates our finding that `Mycroft` is most useful when the DO's dataset is heterogeneous and only a subset of it is useful for the MT.

Scenario 2 - Corrupted labels: Large scale datasets for supervised learning often have incorrect labels. Therefore, we also perform data sharing experiments where we introduce label corruption in the DO's datasets to evaluate how well `Mycroft` performs in this setting. We implement label corruption by randomly permuting 70% of the labels in each dataset we evaluate on. We show the results in Figure 9. We notice that `Mycroft` is fairly robust to label distortions, with an average performance reduction of 4.4% over the baseline performance whereas `random-sampling` suffers from an average reduction in accuracy of 16.9%.

Scenario 3 - Preference ordering for several DOs: In data markets, often there are multiple data sellers with datasets of varying utilities. In such a scenario, `Mycroft` should be able to rank the utility of different datasets in order to facilitate more informed data sharing agreements. We experiment with this scenario by constructing several DO datasets with varying amounts of utility and evaluating whether `Mycroft` can reconstruct this ordering. The details for the DOs and MT's used for this scenario are as follows:
`DO-1`: This DO contains the highest quantity of data from the same distribution as D^{hard} and is the same as the DO we use in other experiments involving the Dogs & Wolves dataset. This DO should provide the highest utility to the MT.

DO	Mycroft	random-sampling	full-information
DO-1	0.81	0.18	0.88
DO-2	0.63	0.31	0.69
DO-3	0.44	0.25	0.63
DO-4	0.56	0.25	0.50
DO-5	0.19	0.25	0.13

Table 4: Preference ordering (based on accuracy) generated from Mycroft and random-sampling for selecting from among several DO candidates with different levels of utility (where each DO's number corresponds to their utillity) from **Scenario 4**. Mycroft is mostly able to retrieve the ground-truth preference ordering whereas random-sampling fails to do so.

MT	Mycroft	random-sampling	full-information
MT-1	49	34	80
MT-2	16	5	40
MT-3	76	58	92
MT-4	87	23	90
MT-5	60	37	60
MT-6	0	0	24
MT-7	92	25	88

Table 5: Number of useful DOs (F1 score of M'_{MT}>=0.5) retrieved by Mycroft and random-sampling for budget of 5 samples and by full-information for different MTs for tabular data. Number of DO candidates = 95.

DO-2: DO-2 is a noisy version of DO-1 where we introduce noise by randomly transforming the images using PyTorch transforms [2]. The transforms we apply include Random crops, resizing, flipping, changing contrast and perspective. We expect the utility of training on such images to be lower as compared to the clean images.
DO-3: DO-3 has randomly sampled data from dog and wolf classes in the ImageNet dataset. We empirically verify that it contains a subset of data from the distribution required by the MT and will thus be useful to the MT to some degree.
DO-4: This DO contains a small subset of the useful samples contained in DO-1. While useful in nature, this DO's ability to signal its utility should be limited.
DO-5: DO-5 contains no data from the required training distribution and only consists of the data from MT's training distribution. This type of data should have the least utility.

The results for this experiment for image datasets are in Table 4 and show that Mycroft can indeed help select the most promising DO from a set of candidates while random-sampling is unable to do so. For the tabular data, because there is no clear ranking amongst the DOs for this dataset (useful DOs tend to have very similar performance after data sharing, same as non-useful DOs), as such, we plot the number of useful DOs (defined as those that give an F1 score of at least 0.5 after data sharing) retrieved by Mycroft and random-sampling for different MTs in Table 5. We see that Mycroft is able to retrieve more useful DOs compared to random-sampling for all MTs. This shows that Mycroft is mostly able to retrieve the ground true utility rankings of the various DO datasets indicating it is well suited for establishing a preference for a set of datasets.

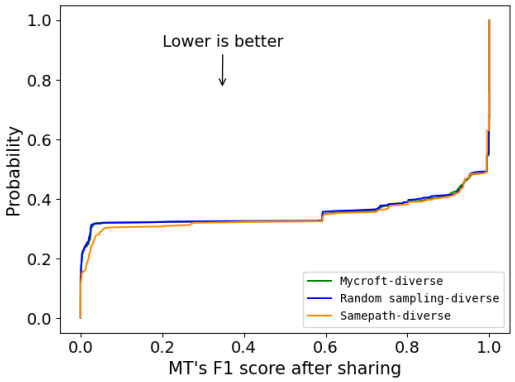


Figure 10: CDF of MT's F1 score after data sharing when D_i^{useful} is made up of 100 samples retrieved from Mycroft and 5 samples retrieved by different data selection methods. Classifier is DecisionTree. N = 574 cases where samples of the same decision path as D^{hard}can be found. Note that the 5 samples later selected by each data selection method must be different from 100 samples already selected by Mycroft.