Distributed Upload and Active Labeling for Resource-Constrained Fleet Learning

Oguzhan Akcin, Harsh Goel, Ruihan Zhao and Sandeep P. Chinchali
The University of Texas at Austin
{oguzhanakcin, harshg99, ruihan.zhao, sandeepc}@utexas.edu

Abstract: In multi-robot systems, fleets are often deployed to collect data that improves the performance of machine learning models for downstream perception and planning. However, real-world robotic deployments generate vast amounts of data across diverse conditions, while only a small portion can be transmitted or labeled due to limited bandwidth, constrained onboard storage, and high annotation costs. To address these challenges, we propose Distributed Upload and Active Labeling (DUAL), a decentralized, two-stage data collection framework for resource-constrained robotic fleets. In the first stage, each robot independently selects a subset of its local observations to upload under storage and communication constraints. In the second stage, the cloud selects a subset of uploaded data to label, subject to a global annotation budget. We evaluate DUAL on classification tasks spanning multiple sensing modalities, as well as on RoadNet—a real-world dataset we collected from vehicle-mounted cameras for time and weather classification. We further validate our approach in a physical experiment using a Franka Emika Panda robot arm, where it learns to move a red cube to a green bowl. Finally, we test DUAL on trajectory prediction using the nuScenes autonomous driving dataset to assess generalization to complex prediction tasks. Across all settings, DUAL consistently outperforms state-of-the-art baselines, achieving up to 31.1% gain in classification accuracy and a 13% improvement in real-world robotics task completion rates.

Keywords: Fleet Learning, Distributed Data Collection, Active Learning

1 Introduction

Modern robotic systems, such as autonomous vehicles, aerial drones, and mobile manipulators, increasingly rely on data collected by large-scale, distributed fleets. These fleets generate several terabytes of data per day [1–3] with diverse sensory modalities such as images, LiDAR scans, and control trajectories to train robust models for perception, prediction, and control. Industrial efforts such as Waymo [4] and Tesla [5], as well as collaborative academic initiatives like DROID [6], Open X-Embodiment [7], have demonstrated the effectiveness of aggregating data from heterogeneous, geographically dispersed platforms to enable scalable, generalizable robot learning.

However, real-world robotics data collection faces two critical bottlenecks: local bandwidth/storage constraints and annotation requirements. Robots often operate in network-constrained environments where uploading all collected data is impractical. For instance, communication bandwidth for data uploads is typically limited to 10 Gbps [8] across multiple devices, making it infeasible to transfer the terabytes of raw sensor data [9]. These constraints necessitate onboard data selection strategies to prioritize informative data while uploading to the cloud for annotation and training.

On the global side, large-scale annotation of the uploaded data remains prohibitively expensive and time-consuming. For instance, robotics benchmarks like RT-1 [10] and Open X-Embodiment [7] required months of manual effort to collect with teleoperation. This issue is further compounded by the need for high-quality annotations across diverse tasks, such as object detection, semantic segmentation, and trajectory prediction, each requiring either manual supervision or querying expensive foundation models. Recent findings further underscore that even semi-autonomous data collection

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

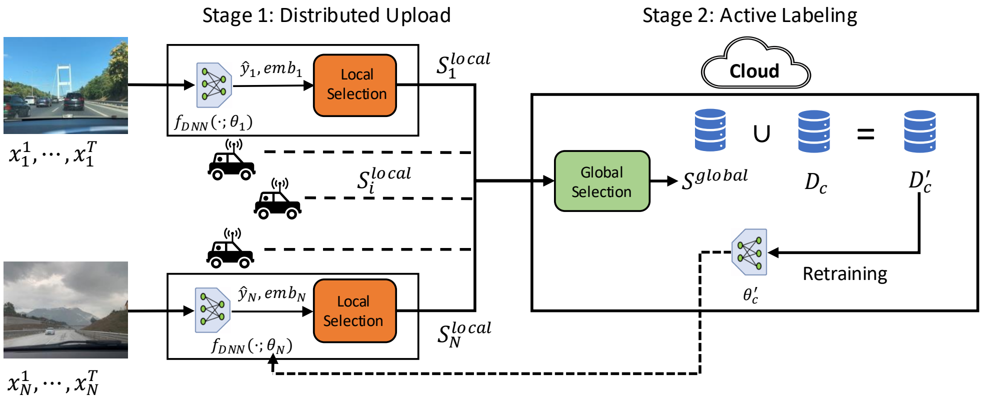


Figure 1: **Overview of Distributed Upload and Active Labeling (DUAL).** DUAL operates in two stages: (1) *Distributed Upload* and (2) *Active Labeling*. In the first stage, each robot i independently observes a stream of local data points $x_i^1, \ldots, x_i^T$, and processes them through a local model $f_{\text{DNN}}(\cdot;\theta_i)$ to generate predictions $\hat{y}_i$ and embeddings emb_i. Then, based on a dataset utility function, each robot selects a subset of informative data samples S_i^{local} to upload under local bandwidth and storage constraints. In the second stage, the cloud aggregates all uploaded data and performs a centralized submodular maximization to select a globally informative subset S^{global} to annotate, subject to a global labeling budget. The labeled data is added to the training set $\mathcal{D}_c$, used to update model weights θ_c in the cloud, which are then transmitted back to the robots.

pipelines remain difficult to scale reliably due to system fragility, and therefore require extensive human supervision [11]. As a result, only a small portion of the uploaded data can be labeled and used to retrain models effectively. These constraints collectively shift the challenge from passive data collection to active data curation: robotic fleets must make intelligent decisions about both *what to upload* and *what to annotate*.

To curate datasets for such scenarios, we introduce **Distributed Upload and Active Labeling (DUAL)**. DUAL addresses two key challenges introduced above: 1) selecting the most valuable data to upload under storage and communication constraints, and 2) selecting the most informative subset of uploaded data to label under a global annotation budget. An overview of DUAL is shown in Fig. 1. In the first stage, each robot independently selects and uploads a subset of its local observations. In the second stage, the cloud labels a subset of all uploaded data. DUAL leverages submodular maximization at both stages to select diverse and informative samples. We provide theoretical guarantees showing that DUAL approximates the globally optimal labeled dataset despite bandwidth and labeling constraints.

DUAL is a scalable, communication-efficient data curation algorithm that is broadly applicable across various robotics domains. It outperforms state-of-the-art baselines that curate data for downstream tasks for autonomous driving and robotic manipulation, even under strict bandwidth and annotation constraints. To assess the performance of DUAL under realistic network limitations, we also introduce the RoadNet dataset. Collected from autonomous vehicles in Turkish cities, RoadNet features raw urban image sequences, providing a testbed to evaluate data selection strategies with restricted upload capacity. This dataset benchmarks the robustness of DUAL and its practical utility to scale curation of robotics data.

We summarize our main contributions as follows:

1. We propose Distributed Upload and Active Labeling (DUAL), a scalable two-stage framework for decentralized data curation in robotic fleets based on sub-modular maximization, which combines local upload decisions with centralized active labeling.
2. We empirically validate DUAL to curate data across a range of domains, including trajectory prediction on the **nuScenes** autonomous driving dataset, classification tasks on diverse modalities, and real-world physical robot experiments using a **Franka Emika Panda robot**. Models trained on curated data through DUAL achieve up to **31.1%** improvement in classification accuracy and **13%** gain in real-world task success rate over baselines.
3. We introduce **RoadNet**, a new dataset of vehicle-mounted camera recordings collected across multiple cities, designed to benchmark decentralized fleet data collection under realistic network limitations.

2 Related Work

Data collection is a critical problem in robotics, directly impacting the performance of perception and decision-making systems. Our work builds on broad literature in data-efficient learning, particularly active learning [12–20], where the goal is to select the most informative samples to label. Traditional pool-based active learning methods assume access to a centralized pool of unlabeled data, whereas in robotic fleets, data is inherently decentralized across agents with limited observability and communication. Recent work on distributed active learning [21–24] addresses decentralized scenarios but often assumes either centralized coordination or unrestricted data upload capabilities. In contrast, our framework explicitly tackles decentralized upload decisions under both bandwidth and annotation constraints.

Multi-robot systems introduce additional challenges due to heterogeneous data distributions, constrained communication channels, and costly labeling pipelines. Prior work in decentralized cloud robotics [25–29] and collaborative learning across fleets [30, 31] has explored data sharing and distributed training. However, these approaches often assume robots can transmit all collected data to the cloud or treat all observations as equally valuable. More recent works like FLEET-MERGE [32] and Sirius-Fleet [33] propose methods for merging policies or adapting visual world models across fleets, but still largely rely on centralizing either model weights or environmental feedback rather than addressing selective data upload. DUAL differentiates itself by enabling each robot to independently select a limited, informative subset of observations to upload, which are then selectively labeled through centralized active learning.

A distinct line of work focuses on data aggregation and supervision strategies to improve task performance in robotics. For instance, imitation learning methods like DAgger [34] and its extensions [35–38] address distribution shift by incorporating human feedback during training. These approaches typically operate on data collected by a single robot and focus on action supervision rather than general-purpose data curation. While Fleet-DAgger and its derivatives [39, 40] extend supervision to multi-robot settings, it does not address upload or annotation constraints. In contrast, our work tackles the broader problem of distributed dataset construction for general supervised learning tasks—classification, detection, and prediction—under practical bandwidth and labeling limitations. Unlike policy-centric aggregation methods, DUAL explicitly selects high-quality data for labeling, making it complementary to such downstream training pipelines.

3 Problem Formulation

We study decentralized data collection in multi-robot systems, where fleets of robots collect data in a distributed manner. Each robot's ability to upload data is limited by factors such as its available cache memory, upload bandwidth, and communication costs. Additionally, we model global limitations on the labelling of the uploaded data that arise due to expensive annotations by humans or foundation models. The objective is to choose a subset of observations that maximizes a dataset utility function to evaluate informative samples for training, while respecting local data upload constraints and the global labelling budget.

Robotic Fleet and Local Observations. Consider a fleet of N_{robot} robots, each operating in distinct environments. Each robot i makes observations $X_i = \{x_i^1, \ldots, x_i^T\}$, consisting of T samples collected over a fixed time interval. Each data point $x \in X_i$ is processed locally via a neural network $f_{\text{DNN}}(x; \theta_i)$ to produce a prediction $\hat{y}_i$ and a task-relevant embedding emb_i. These embeddings can be obtained via foundation models such as CLIP [41] or active learning methods [18, 42], and serve as compact representations to evaluate sample similarity and diversity. Each robot selects a subset $S_i \subseteq X_i$ to upload, constrained by a robot-specific cache or communication limit N_i^{cache}, which bounds the number of data points the robot can share.

Global Labeling Budget. Even after data is uploaded to the cloud, labeling remains a key bottleneck due to the limited human annotation resources and the inference throughput of foundation models. We model this constraint with a global labeling budget N^{label}, which limits the total number of data points that can be labeled. This budget reflects real-world limitations in centralized learning systems and introduces a second decision stage: selecting the most valuable samples to label from the uploaded pool.

Dataset Utility Function. To guide which samples to retain and label, we define a dataset utility function $f(\mathcal{D};\mathcal{T})$ that measures the informativeness and diversity of a candidate set $\mathcal{D}$ with respect to a target dataset $\mathcal{T}$. Since true model performance is unknown, we use a proxy objective based on a monotone submodular function, which is standard in data selection [43, 44]. Submodular functions capture diminishing returns: the gain from adding a new data point decreases as the selected set grows. We adopt the facility location function:

$$f(\mathcal{D};\mathcal{T}) = \sum_{t\in\mathcal{T}} \max_{a\in\mathcal{D}} \frac{1}{1+\alpha\|emb_t - emb_a\|_2}, \tag{1}$$

where emb_t and emb_a denote the task-relevant embeddings of points t and a, respectively. The parameter α controls the influence of embedding distance on similarity. This formulation captures data utility based on proximity in embedding space, encouraging the selected subset to effectively cover the data distribution and include diverse samples. While we use the facility location function in our experiments, other submodular objectives, such as mutual information or set cover, can also be applied depending on the application.

Data Collection Problem. Given per-robot upload limits N_i^{cache} and a global labeling budget N^{label}, the data collection problem aims to select a subset of data points S_i from each robot's local observations X_i that maximizes the dataset utility function f. The problem can be formulated as:

Problem 1 (Data Collection Problem)**.**

$$\begin{aligned} \max_{S_1,\dots,S_{N_{\text{robot}}}} \quad & f(\mathcal{D}_c \cup \bigcup_{i=1}^{N_{\text{robot}}} S_i;\mathcal{T}), \\ \text{subject to:} \quad & S_i \subseteq X_i, \qquad \forall i = 1,\dots,N_{\text{robot}}, \\ & |S_i| \le N_i^{\text{cache}} \qquad \forall i = 1,\dots,N_{\text{robot}}, \\ & \sum_{i=1}^{N_{\text{robot}}} |S_i| \le N^{\text{label}}. \end{aligned} \tag{2}$$

Here, $\mathcal{D}_c$ denotes the dataset already available at the cloud server, which is updated after each round of data collection. The first constraint ensures that each robot only selects data points from its local observations, while the second constraint enforces the upload limit for each robot. The third constraint is the labeling budget that ensures the number of labeled samples does not exceed N^{label}.

Dataset Update and Model Retraining. The cloud dataset is updated with the selected samples to form a new dataset: $\mathcal{D}_c^{'} = \mathcal{D}_c \cup \bigcup_{i=1}^{N_{\text{robot}}} S_i$. This updated dataset is used to retrain a centralized model, resulting in new model weights $\theta_c^{'}$. The updated model can be shared with robots, which can use the updated cloud model or fine-tune it on their local data sets to produce models $\theta_i^{'}$.

4 Distributed Upload and Active Labeling (DUAL)

We now present **Distributed Upload and Active Labeling (DUAL)**, a two-stage data selection framework designed to solve the data curation problem introduced in Eq. 2. DUAL operates in two stages: (1) *Distributed Upload*, in which each robot locally selects data samples to transmit under cache and bandwidth limits, and (2) *Active Labeling* in which the cloud selects the most informative samples from the union of all uploaded data points to annotate. This hierarchical structure is motivated by tractability: solving the global labeling problem in a fully decentralized manner would require robots to repeatedly exchange intermediate selections and marginal gains, leading to significant communication overhead [30, 45]. By decoupling local uploads from centralized labeling, DUAL achieves both scalability and high dataset utility. The full procedure is detailed in Alg. 1.

Stage 1: Distributed Upload (Robot-Side). Each robot $i \in \{1,\dots,N_{\text{robot}}\}$ initializes an empty upload set (line 2) and greedily selects data points from its local observations X_i (lines 3-6). At each iteration, each robot adds the sample that provides the maximum marginal gain to the dataset utility function $f(\cdot;\mathcal{T})$ (line 4). Importantly, robots do not require full access to the raw target dataset $\mathcal{T}$ in this process. Since f operates only over task-relevant embeddings, it is sufficient for robots to use embeddings of the target dataset rather than raw data. After N_i^{cache} points are selected, each robot uploads local selections S_i^{local} to the cloud (line 8). This stage is fully decentralized, allowing each robot to independently select data without communication with others.

Stage 2: Active Labeling (Cloud-Side). After local selections, the cloud aggregates all uploaded data points into a unified candidate set S^{local} (line 9). It then initializes an empty global selection set (line 10) and greedily selects N^{label} points to label (lines 11-14). At each iteration, the point maximizing the marginal utility function $f(\cdot;\mathcal{T})$ is added to the global selection (line 12). Finally, the points in the global selection are added to the cloud dataset (line 15).

Computational Efficiency and Optimality Guarantee. The greedy selection algorithm employed at both stages of DUAL is a well-established heuristic for submodular maximization under cardinality and partition matroid constraints [46]. In our setup, the per-robot cache limit N_i^{cache} corresponds to a partition matroid constraint, while the global label budget N^{label} corresponds to a cardinality constraint.

The computational complexity of DUAL measures the number of function evaluations of the dataset utility function $f(\cdot;\mathcal{T})$. The local upload selection performed independently by each robot i requires $O(|X_i| \cdot N_i^{\text{cache}})$ function evaluations, where $|X_i|$ denotes the number of data points observed by robot i. Thus, the total complexity of the local upload stage across all robots is $O\left(\sum_{i=1}^{N_{\text{robot}}} |X_i| \cdot N_i^{\text{cache}}\right)$. The global label selection performed by the cloud requires $O(\sum_{i=1}^{N_{\text{robot}}} N_i^{\text{cache}} \cdot N^{\text{label}})$ function evaluations, where N^{label} is the number of data points to be labeled. This is because the cloud must evaluate the dataset utility function for each of the N^{label} selected points against the union of all uploaded samples. Thus, the total complexity of DUAL is $O\left(\sum_{i=1}^{N_{\text{robot}}} |X_i| \cdot N_i^{\text{cache}} + \sum_{i=1}^{N_{\text{robot}}} N_i^{\text{cache}} \cdot N^{\text{label}}\right)$. Overall, DUAL achieves linear scalability with respect to the fleet size, making it suitable for large-scale robotic deployments where centralized data gathering would be impractical.

Algorithm 1 Distributed Upload and Active Labeling (DUAL)

Input: observed data points X_i, cloud dataset $\mathcal{D}_c$, target dataset $\mathcal{T}$, dataset utility function f
Output: updated cloud dataset $\mathcal{D}_c^{'}$

```
                Stage 1: Distributed Upload
 1: for $i = 1$ to $N_{\text{robot}}$ do
 2:     $S_i^{\text{local}} \leftarrow \emptyset$
 3:     for $j = 1$ to $N_i^{\text{cache}}$ do
 4:         $x^* \leftarrow \text{argmax}_{x \in X_i \setminus S_i^{\text{local}}} f(\mathcal{D}_c \cup S_i^{\text{local}} \cup \{x\};\mathcal{T})$
 5:         $S_i^{\text{local}} \leftarrow S_i^{\text{local}} \cup \{x^*\}$
 6:     end for
 7: end for
 8: Upload $S_i^{\text{local}}$ to the cloud
                Stage 2: Active Labeling
 9: $S^{\text{local}} \leftarrow \bigcup_{i=1}^{N_{\text{robot}}} S_i^{\text{local}}$
10: $S^{\text{global}} \leftarrow \emptyset$
11: for $k = 1$ to $N^{\text{label}}$ do
12:     $x^* \leftarrow \text{argmax}_{x \in S^{\text{local}} \setminus S^{\text{global}}} f(\mathcal{D}_c \cup S^{\text{global}} \cup \{x\};\mathcal{T})$
13:     $S^{\text{global}} \leftarrow S^{\text{global}} \cup x^*$
14: end for
15: $\mathcal{D}_c^{'} \leftarrow \mathcal{D}_c \cup S^{\text{global}}$
16: return $\mathcal{D}_c^{'}$
```

Moreover, DUAL provides theoretical performance guarantees. Following results from distributed submodular maximization [45], DUAL achieves the following approximation ratio relative to the optimal solution S^* of the data collection problem in Eq. 2:

$$f_{\mathcal{D}_c}(S^{\text{global}}) \geq \frac{1}{2\min(N_{\text{robot}}, N_{\text{max}}^{\text{cache}})} \cdot f_{\mathcal{D}_c}(S^*), \tag{3}$$

where $N_{\text{max}}^{\text{cache}} = \max_{i=1}^{N_{\text{robot}}} N_i^{\text{cache}}$ is the maximum local upload size across all robots, and $f_{\mathcal{D}_c}(A) = f(\mathcal{D}_c \cup A;\mathcal{T}) - f(\mathcal{D}_c;\mathcal{T})$ is the marginal dataset utility function. This approximation guarantee indicates that DUAL maintains strong data curation performance even under decentralized upload and global labeling constraints.

5 Experiments

We simulate a real-world data collection setting in which, during each round (e.g., each day), every robot observes a set of data samples drawn from skewed (non-i.i.d.) class distributions. After each round, robots select and upload a subset of their observed samples, subject to cache and network constraints, for centralized labeling and model retraining in the cloud. For classification tasks, we report the accuracy of the retrained model after each round. For autonomous driving and physical robot control tasks, we report the final model performance after the last round.

Classification Experiments. We evaluate DUAL on the ESC-50 environmental sound dataset [47] and the ModelNet10 3D point cloud dataset [48]. For audio classification, we use the CLAP model as a backbone [49], while for 3D point clouds, we use PointNet++ [50] as a backbone model. Models are fine-tuned after each data collection round on newly labeled samples, and performance is reported on a held-out test set.

RoadNet: Real-World Dataset. We evaluate DUAL on RoadNet, a real-world autonomous driving dataset collected from vehicle video streams across multiple cities in Turkey. The dataset captures diverse environmental conditions, including weather, lighting, and geographic variability. To simulate decentralized fleet-scale data collection, each video stream represents an individual robot's local observation buffer, modeling realistic environmental heterogeneity. In the RoadNet classification task, the model predicts a label comprising the time of day, weather condition, and location type from each road scene image. For example, a typical label might be "sunny, highway, afternoon," capturing the combined contextual attributes of the scene. Example frames are shown in Fig. 2, with full dataset details available in the Appendix.

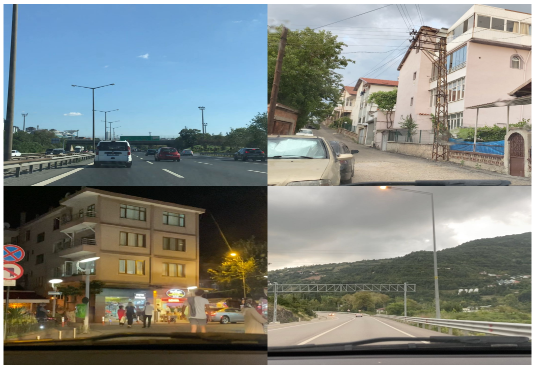


Figure 2: **Examples from the RoadNet dataset.** RoadNet includes vehicle recordings captured across diverse environments (e.g., urban areas, highways, rural roads), under various times of day (e.g., day and night), and weather conditions (e.g., sunny, rainy, overcast).

Network Configurations. Similar to previous work [51], we simulate different cache limits based on network configurations. We consider four network configurations: (1) **Always**: All robots upload equal amounts of data. (2) **Mixed-Scarce**: Some robots have high cache limits while others have low cache limits. (3) **Ookla** [52]: Robots have cache limits that are proportional to the throughput of network traces from Ookla's 5G measurements. (4) **5G** [51]: Robots have cache limits determined by real-world upload throughput data collected from robotics labs.

Baselines. We compare DUAL against several baselines, including (1) **Random**: Robots upload random samples. (2) **Margin** [53]: Robots upload samples where the difference between the top two prediction probabilities is the smallest. (3) **Entropy** [54]: Robots upload samples with the highest prediction entropy. (4) **Data Games** [31]: Robots select samples based on known labels to achieve class balance via a game-theoretic approach (only applicable to classification tasks). (5) **Fleet Active Learning (FAL)** [30]: Robots iteratively select samples to upload and label. (6) **Upper Bound**: All robots upload all their data, and the cloud selects the best samples to label. This is a centralized greedy solution to the data collection problem and represents an idealized, centralized upper bound.

Autonomous Driving Experiments. In addition to the RoadNet dataset, we further evaluate DUAL on the nuScenes dataset [55], a real-world, multi-modal AV benchmark. We use this dataset to test whether DUAL generalizes to large-scale urban driving conditions. In this experiment, we use the PGP model [56] to extract features from the RGB images, LiDAR point clouds, and depth images. The model is fine-tuned on the labeled subset selected by each data selection policy. For evaluation, we report the minimum Average Displacement Error (MinADE) and Miss Rate, which respectively quantify overall trajectory accuracy and the proportion of failed final predictions after the final round of retraining. Full results are shown in Table 1.

Physical Robot Experiments. To evaluate the real-world deployability of DUAL in an embodied robotic setting, we implement it on a tabletop manipulation task, Place-Red-in-Green. In this task,

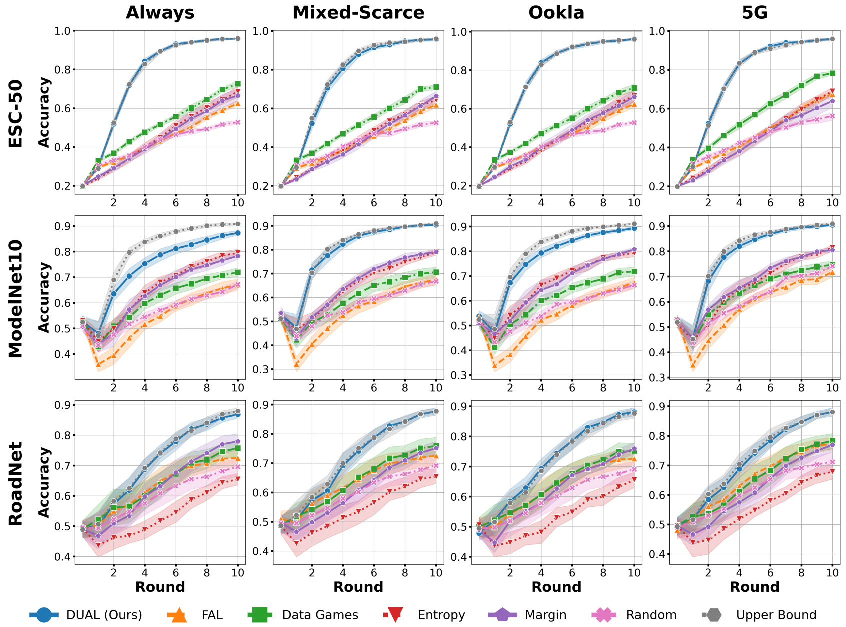


Figure 3: **DUAL outperforms baselines across all datasets and network configurations.** We report the accuracy of the retrained model after each data collection round, with shaded regions denoting one standard deviation over multiple runs. Each row corresponds to a dataset, and each column to a network configuration. By combining decentralized upload with centralized active labeling, DUAL achieves consistently higher accuracy and closely approximates the centralized upper bound. On RoadNet, DUAL improves accuracy by 14.64% over the strongest baseline.

Selection Method	MinADE$_5$	MinADE$_{10}$	MissRate$_{5,2}$	MissRate$_{10,2}$
Random	1.62 ± 0.09	1.22 ± 0.04	0.67 ± 0.02	0.51 ± 0.02
FAL	1.58 ± 0.11	1.19 ± 0.05	0.67 ± 0.02	0.51 ± 0.02
Entropy	1.64 ± 0.07	1.25 ± 0.05	0.69 ± 0.02	0.53 ± 0.03
Margin	1.61 ± 0.11	1.22 ± 0.06	0.66 ± 0.01	0.49 ± 0.02
DUAL (Ours)	**1.43** ± **0.01**	**1.09** ± **0.01**	**0.65** ± **0.01**	**0.48** ± **0.01**
Upper Bound	**1.42** ± **0.01**	**1.09** ± **0.01**	**0.64** ± **0.01**	**0.47** ± **0.01**

Table 1: **Trajectory Forecasting Results on nuScenes.** We report MinADE$_K$ (minimum Average Displacement Error over K predicted trajectories) and MissRate$_{K,d}$ (proportion of predictions deviating more than d meters) for $K = \{5, 10\}$ and $d = 2$. DUAL consistently outperforms baselines and closely matches the centralized upper bound under upload and labeling constraints.

an Intel RealSense camera collects RGB-D images in cluttered tabletop scenes with various colored bowls and blocks. The images are transformed into colored point clouds and further projected into an isotropic 2D RGB image depicting the (x, y) manipulation plane. The robot observes the images and must learn to place red objects into green bowls. In the experiments, we run our selection algorithm and train models in simulated environments, then evaluate model performance after the final round of retraining in the physical setup. We report the normalized mean squared error for predicted red block and green bowl locations in a simulation environment, and the task success rate, defined as the proportion of correct placements across trials on the physical robot. The robot uses a Transporter Network [57] as the perception module for the task. We compare DUAL against several baselines, including Random, Margin, and Entropy, as well as FAL. The physical robot setup used for evaluation is shown in Fig. 4, and the corresponding quantitative results are reported in Table 2.

Results and Discussion. The experimental results, presented in Fig. 3, demonstrate that DUAL consistently outperforms all baseline methods across classification tasks spanning diverse modalities—audio (ESC-50), 3D point clouds (ModelNet10)—as well as our real-world RoadNet dataset. Compared to the best-performing baseline method, DUAL improves classification accuracy by 31.1% on ESC-50 and 12.0% on ModelNet10. On the RoadNet dataset, where agents observe correlated and region-specific driving data, DUAL achieves a significant gain of 14.64% over the best-performing benchmark, highlighting its ability to select diverse and informative samples even

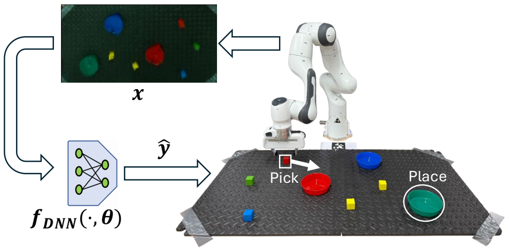


Figure 4: **Physical Robot Setup.** A Franka Emika Panda robot performs a place-red-in-green task, where it must insert a red box into a green bowl based solely on visual input.

Selection Method	Prediction Error ↓	Success Rate ↑
Random	3.18 ± 1.54	0.82 ± 0.11
Margin	19.4 ± 31.6	0.34 ± 0.05
Entropy	13.2 ± 10.7	0.37 ± 0.12
FAL	8.07 ± 5.76	0.78 ± 0.08
DUAL (Ours)	**1.20 ± 0.32**	**0.95 ± 0.02**
Upper Bound	**1.23 ± 0.21**	**0.95 ± 0.04**

Table 2: **Performance on the Place-Red-In-Green Task.** DUAL achieves the lowest prediction error and highest task success rate, matching the centralized upper bound and outperforming all baselines in both metrics.

under decentralized data distributions. These gains arise from DUAL's two-stage design, which separates local data selection from global labeling. By jointly optimizing both cache constraints and the global labeling budget, DUAL ensures that robots with rare or complementary samples also contribute, maximizing diversity and coverage. In contrast, most baselines focus only on local selection, leading to redundant uploads and inefficient use of the label budget. This coordination across stages explains DUAL's consistent advantage over both uncertainty heuristics and fleet learning baselines.

In the nuScenes trajectory prediction task (Table 1), DUAL achieves the lowest error across all metrics. It attains a MinADE$_5$ of 1.43 and a MinADE$_{10}$ of 1.09, outperforming the best baseline by margins of 0.15 and 0.10 meters, respectively. In terms of final prediction reliability, DUAL achieves a MissRate$_{10,2}$ of 0.48, compared to 0.51 from the strongest baseline, reducing the miss rate by 3 percentage points. These results demonstrate that DUAL enables more precise and reliable trajectory forecasting in complex urban environments and closely matches the centralized upper bound despite operating under tight data selection constraints.

In the Place-Red-In-Green robotic manipulation task (Table 2), DUAL leads to substantial improvements in both perception quality and task execution. In simulation, DUAL achieves a prediction error of 1.20 compared to 3.18 for Random and 8.07 for FAL, corresponding to a 62.3% reduction relative to FAL. When deployed on the real robot, DUAL achieves a task success rate of 0.95, significantly outperforming FAL (0.78) and Random (0.82), and matching the centralized upper bound. These results show that DUAL is not only effective in simulation but also deployable in real-world robotic systems, enabling improved sample efficiency and policy generalization.

Across all domains—classification on various modalities, urban driving, and physical robot learning—DUAL consistently provides performance improvements under tight upload and labeling constraints. These results confirm that our method generalizes across input modalities, system scales, and task settings, while remaining both computationally practical and deployment-ready.

6 Conclusion and Future Work

In this work, we introduced **Distributed Upload and Active Labeling (DUAL)**, a scalable two-stage framework for decentralized data curation in multi-robot systems. DUAL addresses practical constraints by jointly optimizing which data to upload under limited communication budgets and which uploaded samples to label under annotation constraints. Our method leverages submodular maximization at both the robot and cloud levels, providing strong theoretical approximation guarantees while remaining computationally efficient. Across diverse experiments spanning audio, 3D point cloud, autonomous driving, and robotic manipulation tasks, DUAL consistently outperforms strong baselines under realistic network conditions. Furthermore, we introduced **RoadNet**, a new dataset collected from real-world vehicle recording to benchmark decentralized data curation methods under heterogeneous and bandwidth-limited settings.

In future work, we aim to incorporate dynamic network topologies, modeling intermittent connectivity and mobile robot fleets. Additionally, we are interested in exploring learning-based approaches, such as reinforcement learning, to adaptively improve data selection strategies over time. Finally, we envision applying DUAL to broader multi-robot tasks, including decentralized exploration, collaborative mapping, and active perception for object detection and tracking.

7 Limitations

While DUAL is designed to operate under practical bandwidth and labeling constraints, it has a few limitations. First, DUAL assumes that each robot has access to reliable local computation for performing greedy upload selection. In extremely resource-constrained settings, this local selection step could introduce non-trivial computational overhead. Second, DUAL relies on fixed feature extractors (e.g., pretrained models) for selecting informative samples. If the feature representations are misaligned with the downstream task, the effectiveness of data selection may degrade. Finally, although DUAL performs decentralized upload decisions, it requires centralized aggregation and labeling at the cloud, which may not be suitable for fully decentralized systems without any infrastructure support.

Acknowledgments

This work was supported in part by the National Science Foundation under Grant No. 2148186 and by the National Science Foundation grant No. 2133481. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Sagan Rossi. Autonomous and adas test cars generate hundreds of tb of data per day, Dec 2021. URL https://www.tuxera.com/blog/autonomous-and-adas-test-cars-produce-over-11-tb-of-data-per-day/.

[2] Chris Mellor. Autonomous vehicle data storage: We grill self-driving car experts about sensors, clouds ... and robo taxis, Feb 2020. URL https://blocksandfiles.com/2020/02/03/autonomous-vehicle-data-storage-is-a-game-of-guesses/.

[3] Adam Zewe. A technique for more effective multipurpose robots, Jun 2024. URL https://news.mit.edu/2024/technique-for-more-effective-multipurpose-robots-0603.

[4] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, et al. Scalability in perception for autonomous driving: Waymo open dataset. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 2443–2451. Computer Vision Foundation / IEEE, 2020. doi:10.1109/CVPR42600.2020.00252. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Sun_Scalability_in_Perception_for_Autonomous_Driving_Waymo_Open_Dataset_CVPR_2020_paper.html.

[5] Joey Klender. Tesla reveals semi fleet data, shows off new feature and infrastructure plans, Apr 2025. URL https://www.teslarati.com/tesla-reveals-semi-fleet-data-shows-off-new-feature-infrastructure-plans/.

[6] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado, editors, Robotics: Science and Systems XX, Delft, The Netherlands, July 15-19, 2024, 2024. doi:10.15607/RSS.2024.XX.120. URL https://doi.org/10.15607/RSS.2024.XX.120.

[7] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and RT-X models : Open x-embodiment collaboration. In IEEE International Conference on Robotics and Automation, ICRA 2024, Yokohama, Japan, May 13-17, 2024, pages 6892–6903. IEEE, 2024. doi:10.1109/ICRA57147.2024.10611477. URL https://doi.org/10.1109/ICRA57147.2024.10611477.

[8] 10gbps 5g data speeds - qualcomm and keysight achieve industry-first milestone. https://datacenternews.asia/story/10gbps-5g-data-speeds-qualcomm-and-keysight-achieve-industry-first-milestone, 2021. [Online; accessed 14-June-2022].

[9] Data is the new oil in the future of automated driving. https://newsroom.intel.com/editorials/krzanich-the-future-of-automated-driving/#gs.LoDUaZ4b, 2016. [Online; accessed 10-June-2021].

[10] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. RT-1: robotics transformer for real-world control at scale. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu, editors, Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023, 2023. doi:10.15607/RSS.2023.XIX.025. URL https://doi.org/10.15607/RSS.2023.XIX.025.

[11] Suvir Mirchandani, Suneel Belkhale, Joey Hejna, Evelyn Choi, Md Sazzad Islam, and Dorsa Sadigh. So you think you can scale up autonomous robot data collection? In 8th Annual Conference on Robot Learning, 2024. URL https://openreview.net/forum?id=XrxLGzF0lJ.

[12] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH, 4:129–145, 1996.

[13] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[14] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 1183–1192. PMLR, 06–11 Aug 2017. URL http://proceedings.mlr.press/v70/gal17a.html.

[15] Simon Tong. Active learning: theory and applications, volume 1. Stanford University USA, 2001.

[16] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/95323660ed2124450caaac2c46b5ed90-Paper.pdf.

[17] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In International Conference on Machine Learning, 2017.

[18] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In International Conference on Learning Representations, 2018. URL https://openreview.net/forum?id=H1aIuk-RW.

[19] Jiaxi Wu, Jiaxin Chen, and Di Huang. Entropy-based active learning for object detection with progressive diversity constraint. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9387–9396, 2022.

[20] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. ICML, Williamstown, pages 441–448, 2001.

[21] Jingyun Yang, Junwu Zhang, Connor Settle, Akshara Rai, Rika Antonova, and Jeannette Bohg. Learning periodic tasks from human demonstrations. In 2022 International Conference on Robotics and Automation (ICRA), pages 8658–8665, 2022. doi:10.1109/ICRA46639.2022.9812402.

[22] Annalisa T. Taylor, Thomas A. Berrueta, and Todd D. Murphey. Active learning in robotics: A review of control principles. Mechatronics, 77:102576, 2021. ISSN 0957-4158. doi:https://doi.org/10.1016/j.mechatronics.2021.102576. URL https://www.sciencedirect.com/science/article/pii/S0957415821000659.

[23] Maya Cakmak, Crystal Chao, and Andrea L. Thomaz. Designing interactions for robot active learners. IEEE Transactions on Autonomous Mental Development, 2(2):108–118, 2010. doi:10.1109/TAMD.2010.2051030.

[24] Crystal Chao, Maya Cakmak, and Andrea L. Thomaz. Transparent active learning for robots. In 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 317–324, 2010. doi:10.1109/HRI.2010.5453178.

[25] Ben Kehoe, Sachin Patil, Pieter Abbeel, and Ken Goldberg. A survey of research on cloud robotics and automation. IEEE Trans. Automation Science and Engineering, 12(2):398–409, 2015.

[26] Sandeep Chinchali, Apoorva Sharma, James Harrison, Amine Elhafsi, Daniel Kang, Evgenya Pergament, Eyal Cidon, Sachin Katti, and Marco Pavone. Network offloading policies for cloud robotics: a learning-based approach. Autonomous Robots, 45(7):997–1012, 2021. doi: 10.1007/s10514-021-09987-4. URL https://doi.org/10.1007/s10514-021-09987-4.

[27] Yuchong Geng, Dongyue Zhang, Po-han Li, Oguzhan Akcin, Ao Tang, and Sandeep P Chinchali. Decentralized sharing and valuation of fleet robotic data. In 5th Annual Conference on Robot Learning, Blue Sky Submission Track, 2021.

[28] Ajay Kumar Tanwani, Nitesh Mor, John D. Kubiatowicz, Joseph E. Gonzalez, and Ken Goldberg. A fog robotics approach to deep robot learning: Application to object recognition and grasp planning in surface decluttering. 2019 International Conference on Robotics and Automation (ICRA), pages 4559–4566, 2019.

[29] Po-han Li, Sravan Kumar Ankireddy, Ruihan Zhao, Hossein Nourkhiz Mahjoub, Ehsan Moradi Pari, ufuk topcu, Sandeep P. Chinchali, and Hyeji Kim. Task-aware distributed source coding under dynamic bandwidth. In Thirty-seventh Conference on Neural Information Processing Systems, 2023. URL https://openreview.net/forum?id=1A4ZqTmnye.

[30] Oguzhan Akcin, Orhan Unuvar, Onat Ure, and Sandeep P. Chinchali. Fleet active learning: A submodular maximization approach. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, Proceedings of The 7th Conference on Robot Learning, volume 229 of Proceedings of Machine Learning Research, pages 1378–1399. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/akcin23a.html.

[31] Oguzhan Akcin, Po-han Li, Shubhankar Agarwal, and Sandeep P. Chinchali. Decentralized data collection for robotic fleet learning: A game-theoretic approach. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, Proceedings of The 6th Conference on Robot Learning, volume 205 of Proceedings of Machine Learning Research, pages 978–988. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/akcin23a.html.

[32] Lirui Wang, Kaiqing Zhang, Allan Zhou, Max Simchowitz, and Russ Tedrake. Robot fleet learning via policy merging. In The Twelfth International Conference on Learning Representations, 2024. URL https://openreview.net/forum?id=IL71c1z7et.

[33] Huihan Liu, Yu Zhang, Vaarij Betala, Evan Zhang, James Liu, Crystal Ding, and Yuke Zhu. Multi-task interactive robot fleet learning with visual world models. In 8th Annual Conference on Robot Learning, 2024. URL https://openreview.net/forum?id=DDIoRSh8ID.

[34] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, volume 15 of Proceedings of Machine Learning Research, pages 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

[35] Kunal Menda, Katherine Rose Driggs-Campbell, and Mykel J. Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019, Macau, SAR, China, November 3-8, 2019, pages 5041–5048. IEEE, 2019. doi:10.1109/IROS40897.2019.8968287. URL https://doi.org/10.1109/IROS40897.2019.8968287.

[36] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S. Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, Proceedings of the 5th Conference on Robot Learning, volume 164 of Proceedings of Machine Learning Research, pages 598–608. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/hoque22a.html.

[37] Sonia Chernova and Manuela Veloso. Interactive policy learning through confidence-based autonomy. J. Artif. Int. Res., 34(1):1–25, jan 2009. ISSN 1076-9757.

[38] Snehal Jauhri, Carlos Celemin, and Jens Kober. Interactive imitation learning in state-space. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, Proceedings of the 2020 Conference on

Robot Learning, volume 155 of Proceedings of Machine Learning Research, pages 682–692. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/jauhri21a.html.

[39] Ryan Hoque, Lawrence Yunliang Chen, Satvik Sharma, Karthik Dharmarajan, Brijen Thananjeyan, Pieter Abbeel, and Ken Goldberg. Fleet-dagger: Interactive robot fleet learning with scalable human supervision. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, Proceedings of The 6th Conference on Robot Learning, volume 205 of Proceedings of Machine Learning Research, pages 368–380. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/hoque23a.html.

[40] Gaurav Datta, Ryan Hoque, Anrui Gu, Eugen Solowjow, and Ken Goldberg. Iifl: Implicit interactive fleet learning from heterogeneous human supervisors. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, Proceedings of The 7th Conference on Robot Learning, volume 229 of Proceedings of Machine Learning Research, pages 2340–2356. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/datta23a.html.

[41] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning, 2021.

[42] Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In International Conference on Learning Representations, 2020. URL https://openreview.net/forum?id=ryghZJBKPS.

[43] Andreas Krause and Daniel Golovin. Submodular function maximization. In Tractability, 2014.

[44] Jeffrey Bilmes. Submodularity In Machine Learning and Artificial Intelligence. Arxiv, abs/2202.00132, Oct 2022. URL https://arxiv.org/abs/2202.00132.

[45] Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization. Journal of Machine Learning Research, 17(235):1–44, 2016. URL http://jmlr.org/papers/v17/mirzasoleiman16a.html.

[46] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions—II, pages 73–87. Springer Berlin Heidelberg, Berlin, Heidelberg, 1978. ISBN 978-3-642-00790-3. doi:10.1007/BFb0121195. URL https://doi.org/10.1007/BFb0121195.

[47] Karol J. Piczak. Esc: Dataset for environmental sound classification. In Proceedings of the 23rd ACM International Conference on Multimedia, MM '15, page 1015–1018, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334594. doi:10.1145/2733373.2806390. URL https://doi.org/10.1145/2733373.2806390.

[48] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes, 2015. URL https://arxiv.org/abs/1406.5670.

[49] Benjamin Elizalde, Soham Deshmukh, and Huaming Wang. Natural language supervision for general-purpose audio representations, 2023. URL https://arxiv.org/abs/2309.05767.

[50] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017. URL https://arxiv.org/abs/1706.02413.

[51] Oguzhan Akcin, Ahmet Ege Tanriverdi, Kaan Kale, and Sandeep P. Chinchali. Fleet supervisor allocation: A submodular maximization approach. In 8th Annual Conference on Robot Learning, 2024. URL https://openreview.net/forum?id=9dsBQhoqVr.

[52] Ookla. Internet speed dataset. https://www.kaggle.com/datasets/dhruvildave/ookla-internet-speed-dataset, 2022. [Online; accessed 15-February-2024].

[53] Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, Machine Learning: ECML 2006, pages 413–424, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-46056-5.

[54] Dan Wang and Yi Shang. A new active labeling method for deep learning. In 2014 International Joint Conference on Neural Networks (IJCNN), pages 112–119, 2014. doi:10.1109/IJCNN.2014.6889457.

[55] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving, 2020. URL https://arxiv.org/abs/1903.11027.

[56] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, Proceedings of the 5th Conference on Robot Learning, volume 164 of Proceedings of Machine Learning Research, pages 203–212. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/deo22a.html.

[57] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, Proceedings of the 2020 Conference on Robot Learning, volume 155 of Proceedings of Machine Learning Research, pages 726–747. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/zeng21a.html.

[58] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions–i. Math. Program., 14(1):265–294, dec 1978. ISSN 0025-5610. doi:10.1007/BF01588971. URL https://doi.org/10.1007/BF01588971.

[59] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016. doi:10.1109/CVPR.2016.90.

Appendix

Code and Data Availability: The code and related materials can be found in the following code repository:

`https://github.com/UTAustin-SwarmLab/DUAL.git`

The organization of the appendix is as follows:

- Section A provides the details of the theoretical bounds presented in the main paper.
- Section B provides additional details on the experimental setup and hyperparameters used in the experiments.
- Section C provides additional details on the network configurations used in the experiments.
- Section D provides numerical results of the experiments presented in the main paper for the classification experiments.
- Section E provides additional details on the RoadNet dataset.

A Theoretical Bounds

This section elaborates on the theoretical foundations of the **Distributed Upload and Active Labeling (DUAL)** framework introduced in Section 4 of the main paper. We begin by defining key concepts such as submodularity, monotonicity, and matroid constraints. We then outline the greedy policy under these constraints and derive the approximation guarantees for DUAL by leveraging results from distributed submodular maximization.

Preliminaries.

Definition 1 (Submodular Function)**.** *A set function $f : 2^{\mathcal{X}} \to \mathbb{R}$ is submodular if for all $A \subseteq B \subseteq \mathcal{X}$ and $x \in \mathcal{X} \setminus B$, the marginal gain of adding x to A is at least as large as the marginal gain of adding x to B:*

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B).$$

Definition 2 (Monotone Function)**.** *A set function $f : 2^{\mathcal{X}} \to \mathbb{R}$ is monotone if for all $A \subseteq B \subseteq \mathcal{X}$, we have:*

$$f(A) \leq f(B).$$

Matroid Constraints.

Definition 3 (Uniform Matroid)**.** *Let E be a finite set and $k \in \mathbb{Z}^+$ be a positive integer. A uniform matroid $M = (E, \mathcal{I})$ is defined as follows:*

- *The ground set E is a finite set of elements.*
- *The independent sets $\mathcal{I}$ are all subsets of E with at most k elements.*

Definition 4 (Partition Matroid)**.** *Let E be a finite set and $\mathcal{P} = \{E_1, E_2, \ldots, E_k\}$ be a partition of E. A partition matroid $M = (E, \mathcal{I})$ is defined as follows:*

- *The ground set E is a finite set of elements.*
- *The independent sets $\mathcal{I}$ are all subsets of E such that for each i, the number of elements in the independent set from E_i is at most k_i, where k_i is the capacity of partition E_i.*

Greedy Policy.

Definition 5 (Greedy Policy under Matroid Constraints)**.** *Let $f : 2^E \to \mathbb{R}_{\geq 0}$ be a monotone submodular function, and let $M = (E, \mathcal{I})$ be a matroid over the ground set E with independent sets $\mathcal{I}$. The greedy policy is a sequential selection process that starts with the empty set $S = \emptyset$ and iteratively adds the element $x \in E \setminus S$ that provides the largest marginal gain, subject to the constraint that $S \cup \{x\} \in \mathcal{I}$. Formally, the algorithm is defined as:*

Theorem 1 (Approximation under Uniform Matroid [58])**.** *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function, and let $M = (E, \mathcal{I})$ be a uniform matroid. The greedy policy (Algorithm 2) achieves an approximation ratio of at least $1 - \frac{1}{e}$. For the detailed proof, see [58].*

Theorem 2 (Approximation under Partition Matroid [46])**.** *Let $f : 2^E \to \mathbb{R}$ be a monotone submodular function, and let $M = (E, \mathcal{I})$ be a partition matroid. The greedy policy (Algorithm 2) achieves an approximation ratio of at least $\frac{1}{2}$. For the detailed proof, see [46].*

Algorithm 2 Greedy Policy for Submodular Maximization under Matroid Constraints

```
$S \leftarrow \emptyset$
while $\exists x \in E \setminus S$ such that $S \cup \{x\} \in \mathcal{I}$ do
    $x^* \leftarrow \text{argmax}_{x \in E \setminus S, S \cup \{x\} \in \mathcal{I}} f(S \cup \{x\}) - f(S)$
    $S \leftarrow S \cup \{x^*\}$
end while
return $S$
```

Distributed Setting.

Definition 6 (Distributed Submodular Maximization under Matroid Constraints)**.** *Consider a finite ground set E partitioned across m agents, where agent i has access to a local subset $E_i \subseteq E$. Let $f : 2^E \to \mathbb{R}_{\geq 0}$ be a monotone submodular function, and let $\mathcal{I}$ be the collection of independent sets of a matroid $M = (E, \mathcal{I})$.*

Each agent i selects a local subset $S_i \subseteq E_i$ such that $S_i \in \mathcal{I}_i$, where $\mathcal{I}_i$ is the local restriction of $\mathcal{I}$. A centralized algorithm then selects a global subset $S \subseteq \bigcup_{i=1}^{m} S_i$ such that $S \in \mathcal{I}$. The goal is to maximize $f(S)$ subject to the matroid constraint.

Definition 7 (GreeDi Algorithm [45])**.** *The GreeDi algorithm operates in two stages:*

- *Each agent i runs the greedy policy locally on E_i under its matroid constraint $\mathcal{I}_i$ to select a subset S_i.*
- *A centralized greedy policy merges the local selections and chooses a global subset $S \subseteq \bigcup_i S_i$ under $\mathcal{I}$.*

Theorem 3 (GreeDi Approximation Guarantee [45])**.** *Let f be a monotone submodular function and $M = (E, \mathcal{I})$ a matroid with rank r. Suppose the local greedy algorithm achieves a τ-approximation. Then, the GreeDi algorithm over m agents guarantees:*

$$f(S) \geq \frac{\tau}{\min(m, r)} \cdot f(S^*),$$

where S^* is the optimal centralized solution.

Connection to DUAL. Our **DUAL** framework instantiates the GreeDi algorithm:

- In the **Distributed Upload** stage, each robot greedily selects up to N_i^{cache} samples, subject to a local cache constraint.
- In the **Active Labeling** phase, the cloud greedily selects N^{label} samples from the uploaded pool.

Approximation Guarantee for DUAL. Applying Theorem 3, the DUAL framework satisfies the following guarantees:

- Under **uniform matroid constraints**, DUAL achieves:

$$f(S_{\text{DUAL}}) \geq \frac{1 - \frac{1}{e}}{\min(N_{\text{robot}}, N_{\text{max}}^{\text{cache}})} \cdot f(S^*),$$

- Under **partition matroid constraints**, DUAL guarantees a more conservative bound:

$$f(S_{\text{DUAL}}) \geq \frac{1}{2\min(N_{\text{robot}}, N_{\text{max}}^{\text{cache}})} \cdot f(S^*).$$

Here, S_{DUAL} is the subset selected by the DUAL algorithm, S^* is the optimal solution, N_{robot} is the number of robots, and $N_{\text{max}}^{\text{cache}} = \max_{i=1}^{N_{\text{robot}}} N_i^{\text{cache}}$ denotes the maximum local cache size across all robots, corresponding to the rank of the matroid.

While the uniform matroid setting aligns with the implementation of DUAL in practice, we conservatively report the partition matroid bound ($\frac{1}{2\min(N_{\text{robot}}, N_{\text{max}}^{\text{cache}})}$) in the main paper. This choice ensures the robustness of our theoretical guarantees under stricter assumptions and avoids overstating the algorithm's performance.

B Experimental Setup

This section provides detailed implementation and evaluation settings for our experiments across three domains: (1) classification, (2) autonomous driving, and (3) physical robot manipulation. Each scenario involves decentralized data collection by multiple robots under varying bandwidth and annotation constraints, in line with the DUAL framework.

Across all experiments, we simulate a multi-robot system in which each robot collects data from a unique local distribution. Robots are grouped into *environments*, and within each environment, all robots observe the same set of samples, representing local homogeneity. However, environments themselves differ in class distribution, simulating realistic deployment scenarios with cross-region or task-specific heterogeneity.

B.1 Classification Experiments

We conduct classification experiments on ESC-50 [47] (audio), ModelNet10 [48] (3D point cloud), and RoadNet (autonomous driving imagery). For each dataset, robots begin with an initial labeled set $\mathcal{D}_c$ sampled uniformly across classes, and model training proceeds in rounds. At each round, robots upload selected data points (subject to bandwidth constraints) and the cloud selects a subset to annotate under a global labeling budget. The central model is retrained on the newly labeled dataset and distributed back to robots.

B.1.1 Embedding Functions.

We adopt the BADGE [42] approach for obtaining data embeddings in our classification experiments. BADGE constructs embeddings by combining predictive uncertainty with feature diversity, using gradient information from the final (output) layer of a neural network. Specifically, the embedding for input x under model $f_{\text{DNN}}(x;\theta_i)$ with current parameters θ_i is defined as:

$$\phi_i(x) = \nabla_{\theta_{\text{out}}} \ell_{\text{CE}}(f_{\text{DNN}}(x;\theta_i), \hat{y}), \tag{4}$$

where $\hat{y} = \arg\max f_{\text{DNN}}(x;\theta_i)$ is the model-predicted class and θ_{out} denotes the parameters of the final layer. These embeddings reflect both the model's uncertainty (through gradient magnitude) and data diversity (through direction), enabling the selection of samples that induce large and diverse updates during training.

B.1.2 Model Training and Evaluation

Across all datasets, models are trained using the Adam optimizer with a learning rate of 0.001 and a batch size of 100. A scheduler with a decay rate of 0.99 is applied, and training spans 400 epochs per round. No data augmentation is used. Each experiment is repeated across 12 random seeds.

B.1.3 ESC-50

The ESC-50 dataset consists of 2,000 environmental audio recordings, each 5 seconds long, categorized into 50 classes. Each class contains 40 samples, and the dataset is divided into 2,000 training samples and 1,000 test samples. The audio files are sampled at 44.1 kHz and converted to spectrograms for model training.

Simulation Parameters. For ESC-50, we simulate 4 heterogeneous environments, each with 5 robots, totaling 20 robots. Three environments observe a subset of 10 classes; one observes all classes. Classes to be observed by each environment are randomly selected for each round of the simulation. After the classes are selected, the class distributions within each environment are generated using a Dirichlet distribution with $\alpha = 5$. Each round, robots observe 100 samples to select samples for upload. The total cache size is set to 400 samples, and total labeling budget is set to 18 samples per round, and the initial dataset contains 10 samples. Data collection is simulated for 10 rounds, resulting in a final dataset of 190 samples.

DNN and Embedding Function. We employ a CLAP [49] as the backbone for feature extraction. Then we use a multi-layer perceptron (MLP) with two fully connected layers, with a hidden dimension of 128, and ReLU activation with dropout (0.3) to mitigate overfitting. The final layer is a softmax layer for classification. We use BADGE embeddings for sample selection.

B.1.4 ModelNet10

The ModelNet10 dataset consists of 4,899 3D CAD models categorized into 10 classes, such as airplane, car, and chair. The dataset is divided into a training set with 4,888 samples and a test set with 1,000 samples. Each model is represented as a point cloud.

Simulation Parameters. For ModelNet10, we simulate 4 heterogeneous environments, each with 5 robots, totaling 20 robots. Three environments are restricted to observing samples from only 2 classes, while one environment has access to all 10 classes. The specific class subsets for each environment are randomly selected at the beginning of each simulation round. After class assignment, class distributions within each environment are generated using a Dirichlet distribution with concentration parameter $\alpha = 5$. Each round, robots observe

100 samples to select samples for upload. The total cache size is set to 400 samples, and total labeling budget is set to 35 samples per round, and the initial dataset contains 10 samples. Data collection is simulated for 10 rounds, resulting in a final dataset of 360 samples.

DNN and Embedding Function. We employ a PointNet++ [50] model as the backbone for 3D point cloud feature extraction. The final classification head consists of two fully connected layers with 128 hidden units, ReLU activations, and a dropout rate of 0.3 to mitigate overfitting. A softmax layer is applied to produce class probabilities. BADGE embeddings are computed from this architecture and used for sample selection.

B.1.5 RoadNet

The RoadNet dataset comprises 8,223 high-resolution RGB images (1920 × 1080 × 3) captured from vehicle-mounted cameras. The dataset spans various locations, weather conditions, and times of day, ensuring a diverse set of driving scenarios. More details about this dataset can be found in Section E.

Simulation Parameters. The RoadNet experiment simulates 10 heterogeneous environments, each containing 2 robots, for a total of 20 robots. Videos in the dataset are divided into two groups based on the number of class types they contain: Group 1 videos contain only 3 common classes, while Group 2 videos contain all classes. Eight environments are randomly assigned videos from Group 1, and the remaining two environments receive videos from Group 2. Each environment is assigned at least 200 samples, which are extracted from the assigned videos. If a video contains fewer than 200 frames, an additional video from the same group is appended. The total cache size is set to 400 samples, and total labeling budget is set to 18 samples per round, and the initial dataset contains 12 samples. Data collection is simulated for 10 rounds, resulting in a final dataset of 192 samples.

DNN and Embedding Function. We use a pre-trained ResNet-50 [59] model with transfer learning. The final layers are replaced with a multi-layer perceptron (MLP) with two fully connected layers, with a hidden dimension of 128, and ReLU activation with dropout (0.3) to mitigate overfitting. The final layer is a softmax layer for classification. We use BADGE embeddings for sample selection.

B.2 Autonomous Driving Experiments

We also evaluate our proposed method in the context of autonomous driving, where the goal is to improve the model's performance in the trajectory prediction task. In this task, the model is trained to predict the future trajectory of a vehicle based on its current state and the states of other vehicles in the environment. The model is evaluated based on its ability to accurately predict the future trajectory of the vehicle. We use the nuScenes dataset [55] for this task, which is a large-scale dataset for autonomous driving that includes a variety of driving scenarios and environments.

nuScenes. The nuScenes dataset contains 1,000 driving scenes, each 20 seconds long and sampled at 2 Hz. Each scene includes rich annotations and multiple sensor modalities: 6 cameras (surround-view), a 32-beam LiDAR, a 5-beam radar, GPS/IMU, and detailed map priors. The dataset covers complex scenarios including intersections, merges, and roundabouts under varied weather and lighting conditions. The dataset is split into 700 training and 300 test scenes.

Simulation Parameters We simulate 4 heterogeneous environments, each with 5 robots (20 total). Two environments are constrained to observing data from a fixed subset of 10 scenes (drawn randomly per round), while the remaining two environments access data from the full scene pool. Each round, every robot observes 1000 unlabeled samples and selects a subset for upload. We use the Always network configuration, where each robot has an equal cache size. The total cache size is fixed at 1000 samples per round, evenly divided among robots. The centralized annotator is limited to labeling 50 samples per round. The experiment starts with an initial labeled dataset of 50 samples and runs for 10 rounds, yielding a final curated dataset of 550 labeled samples.

DNN and Embedding Function We use PGP [56], a trajectory prediction model that represents the driving scene as a directed lane graph and predicts future trajectories via a learned graph-based policy. The encoder integrates information from HD maps, the ego vehicle's motion, and surrounding agents using graph neural networks and attention mechanisms. A policy module samples plausible routes along the graph, and a decoder generates multimodal trajectory distributions conditioned on route context and latent intent variables. For sample selection, we use the encoder's context-aware node embeddings aggregated along sampled graph traversals, which capture rich spatial and behavioral context. The encoder is initialized with pre-trained weights, while the decoder is trained from scratch on selected samples.

B.3 Physical Robot Experiments

We conduct physical robot experiments to evaluate the feasibility of implementing our decentralized data selection approach in a real-world robotic manipulation setting. These experiments assess how well a model trained on simulation data using the Transporter Network architecture [57] transfers to physical hardware, focusing on both spatial accuracy and task success rates.

We now describe the physical robot experiments. In these experiments, we first train our model on the simulation data generated by Transporters [57] and then deploy and test on the real robots. The goal of these experiments is to evaluate the performance of our method in a real-world scenario, where the model is trained on simulated data and then deployed on physical robots. We report the prediction error of the model on the simulated data and the task success rate on the real robots. The prediction error is defined as the average distance between the predicted pick and place locations and the actual pick and place locations. The task success rate is defined as the percentage of successful pick and place tasks completed by the robots. We use this simulation to test whether our method can also be implemented in robotic applications.

Task Description: Place-Red-In-Green. We adopt the Ravens benchmark's *place-red-in-green* task, which requires a robot to identify red blocks and place them into green bowls on a cluttered tabletop containing distractor objects of varying shapes and colors. This task poses challenges of multimodal perception, precise action generation, and generalization to new object configurations.

Simulation Setup. A synthetic dataset of 1,000 episodes is generated using the Ravens simulation environment, with randomized object positions and orientations. An additional 100 test episodes are reserved for evaluation. In each episode, the robot is presented with an RGB-D observation and must complete a single pick-and-place operation. We simulate 4 heterogeneous environments, each containing 5 robots (20 robots in total). Two environments receive data sampled from 10 scenes, while the remaining two environments observe all scenes. At each round, robots observe 100 candidate samples. We use the Always network configuration, meaning each robot has equal upload bandwidth. Each robot has a cache of 300 samples, and we simulate 10 rounds with a global labeling budget of 10 samples per round, starting with 2 labeled samples. This yields a final labeled dataset of 102 samples.

Model Architecture and Embeddings. We use the Transporter Network [57], a spatially structured architecture that estimates pick and place poses by learning dense pixelwise action-value maps from top-down RGB-D inputs. The network consists of two branches: a pick model and a place model. The pick model learns a fully convolutional action-value function over pixels to identify effective grasp points, while the place model performs cross-correlation (template matching) between the picked region and the rest of the image to identify suitable placement locations. This formulation is inherently equivariant to translation and rotation, allowing the model to learn effective manipulation policies with limited data. We train the model from scratch on selected samples. For data selection, we compute BADGE embeddings over the predicted pick and place logits and use them for scoring and upload prioritization.

Real Robot Deployment. The learned model is deployed on a physical Panda Franka Emika robot. The robot is equipped with an Intel RealSense RGB-D camera, and the system is controlled via MoveIt in ROS. RGB-D observations are processed in real time to predict pick and place actions using the trained Transporter Network. The Franka Hand gripper is used for manipulation. The robot executes pick-and-place actions autonomously, attempting to place red blocks into green bowls across multiple physical configurations. Each model is evaluated across 5 seeds, and each seed is tested on 4 unique scene setups.

Metrics and Evaluation. We report two main metrics: (1) *prediction error*, defined as the mean Euclidean distance between predicted and ground-truth pick/place positions in simulation, and (2) *task success rate*, defined as the fraction of correctly executed pick-and-place episodes on real hardware. This two-stage evaluation setup allows us to gauge both sample efficiency and sim-to-real transfer capability.

Limitations of Comparisons. Due to the reliance of the Data Games method on class distributions, we exclude it from autonomous driving and real-world robot experiments, as such distributions do not exist. All other baselines are evaluated where applicable.

C Network Configurations

In the classification experiments, we simulate four network configurations—**Always**, **Mixed-Scarce**, **Ookla**, and **5G**—to evaluate the performance of our method under varying communication constraints. These configurations determine the per-robot *cache size*, i.e., the number of data samples each robot is allowed to upload per round. We fix the total cache budget per round to 400 samples and vary how it is distributed across robots to reflect diverse deployment conditions.

Dataset	Network	Selection Policy							Percentage Improvement
		Random	Margin	Entropy	Data Games	FAL	DUAL (Ours)	Upper Bound	
ESC-50	Always	0.53±0.02	0.67±0.06	0.69±0.05	0.73±0.03	0.62±0.03	**0.96**±0.01	0.96±0.01	32.05
	Mixed-Scarce	0.53±0.02	0.66±0.04	0.64±0.03	0.71±0.02	0.62±0.02	**0.96**±0.01	0.96±0.01	34.60
	Ookla	0.53±0.02	0.66±0.05	0.67±0.05	0.71±0.03	0.62±0.03	**0.96**±0.01	0.96±0.01	35.98
	5G	0.56±0.03	0.64±0.06	0.69±0.03	0.78±0.02	0.67±0.03	**0.96**±0.01	0.96±0.01	22.53
	Average	0.54±0.03	0.66±0.05	0.67±0.05	0.73±0.03	0.63±0.03	**0.96**±0.01	0.96±0.01	31.07
ModelNet10	Always	0.67±0.02	0.78±0.03	0.79±0.03	0.72±0.03	0.67±0.04	**0.87**±0.02	0.91±0.01	9.83
	Mixed-Scarce	0.67±0.03	0.79±0.02	0.79±0.02	0.75±0.03	0.68±0.04	**0.90**±0.01	0.91±0.01	14.33
	Ookla	0.66±0.03	0.80±0.02	0.79±0.03	0.74±0.03	0.67±0.04	**0.90**±0.01	0.91±0.01	10.58
	5G	0.67±0.03	0.80±0.02	0.81±0.02	0.75±0.03	0.71±0.03	**0.90**±0.01	0.91±0.01	11.12
	Average	0.69±0.04	0.80±0.02	0.80±0.03	0.75±0.03	0.68±0.04	**0.89**±0.02	0.91±0.01	12.00
RoadNet	Always	0.70±0.05	0.78±0.05	0.65±0.04	0.76±0.05	0.72±0.05	**0.88**±0.02	0.88±0.02	11.42
	Mixed-Scarce	0.69±0.06	0.75±0.04	0.65±0.05	0.76±0.06	0.73±0.06	**0.88**±0.02	0.88±0.02	15.51
	Ookla	0.69±0.05	0.76±0.04	0.66±0.05	0.75±0.05	0.73±0.05	**0.88**±0.02	0.88±0.03	16.08
	5G	0.71±0.06	0.77±0.04	0.68±0.05	0.78±0.05	0.77±0.04	**0.88**±0.03	0.88±0.02	12.54
	Average	0.70±0.05	0.77±0.04	0.66±0.05	0.76±0.05	0.74±0.05	**0.88**±0.02	0.88±0.02	14.64

Table 3: **Comparison of Selection Policies Across Networks and Datasets.** This table presents the final-round classification accuracy (mean ± standard deviation) for different selection policies across three datasets (ESC-50, ModelNet10, and RoadNet) and four network configurations (Always, Mixed-Scarce, Ookla, 5G). **Bold** values indicate the performance of DUAL (Ours) and the best-performing baseline in each row. Shaded cells highlight DUAL and the oracle-style Upper Bound, which assumes access to full global information. The final column reports DUAL's percentage improvement over the strongest baseline in that configuration. The last row of each dataset block reports the average accuracy and average percentage improvement. These results correspond to Fig. 3.

Always. The Always configuration assumes uniform network conditions across the fleet. Each robot receives an equal share of the total cache, with cache size set to $400/n_{\text{robot}}$, where n_{robot} denotes the number of robots. This serves as a baseline to evaluate our method in the absence of any heterogeneity.

Mixed-Scarce. In the Mixed-Scarce configuration, robots are split into two groups with different cache allocations to simulate heterogeneous connectivity. Specifically, 70% of the robots belong to a low-cache group, and the remaining 30% belong to a high-cache group. The cache size of robots in the low-cache group is set to $2/9$ of that of robots in the high-cache group. This configuration allows us to study how unequal upload opportunities affect data selection and downstream performance.

Ookla. In this configuration, robot cache sizes are assigned based on real-world cellular network performance data from the Ookla Speedtest dataset [52]. We divide the coverage area into a 10×10 spatial grid and compute the average upload speed within each cell. We then randomly assign robots to cells and allocate their cache sizes proportional to the normalized upload speeds of the corresponding cells. This setting reflects realistic geographic variability in mobile connectivity.

5G. The 5G configuration uses real-world 5G network performance data collected from a floor of a robotics lab [51]. The dataset includes upload throughput measurements across 100 locations. Each robot is assigned a cache size proportional to the average upload speed of one of these regions. This configuration evaluates DUAL's performance under practical 5G deployment conditions where cache sizes are influenced by location-dependent connectivity.

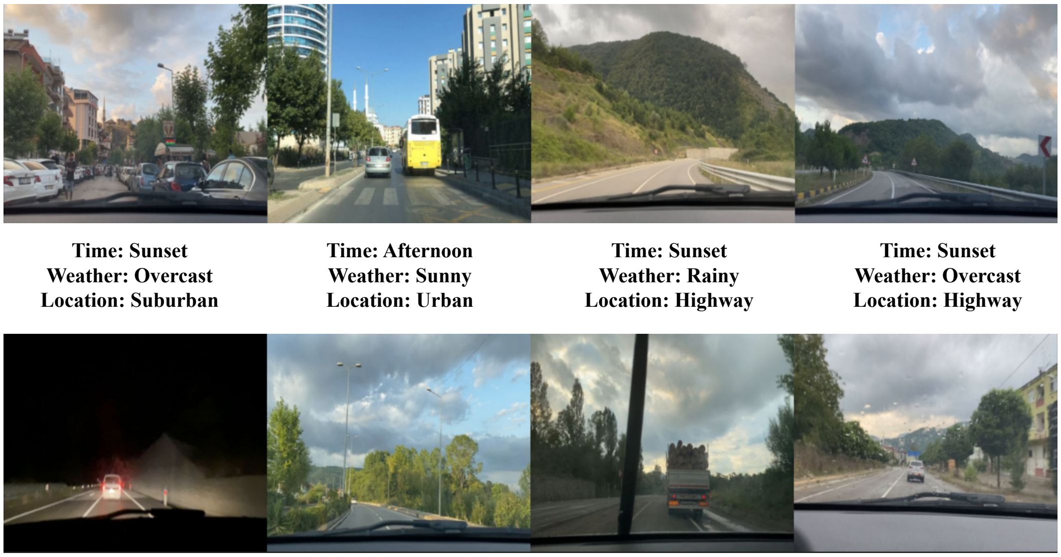
D Numerical Results

In this section, we present detailed numerical results corresponding to the classification experiments shown in Fig. 3 of the main paper. Table 3 reports the final-round classification accuracy (mean ± standard deviation) across three datasets (ESC-50, ModelNet10, and RoadNet) and four network configurations (Always, Mixed-Scarce, Ookla, and 5G). We compare our method, DUAL (Ours), against several baselines: Random, Margin, Entropy, Data Games, and FAL, along with an oracle-style Upper Bound that has access to full information.

Across all datasets and network configurations, DUAL consistently achieves the highest accuracy, matching the performance of the Upper Bound in every case. These results demonstrate that DUAL is highly effective at selecting informative and diverse samples for upload, even under realistic and heterogeneous cache constraints.

The last column reports the percentage improvement of DUAL over the best-performing baseline in each configuration. Notably, on ESC-50, DUAL shows dramatic gains—up to 31.07%—highlighting its ability to handle high-entropy audio classification tasks. Performance gains on ModelNet10 and RoadNet are also substantial, especially under more challenging network configurations like Mixed-Scarce and 5G. These findings confirm the adaptability and robustness of DUAL in decentralized, resource-constrained data curation scenarios.

In Table 3, we use **boldface** to highlight both DUAL (Ours) and the best-performing baseline in each row. Shaded cells further indicate DUAL and the oracle-style Upper Bound. This allows a clear comparison between our method and the most competitive alternatives.



Time: Sunset
Weather: Overcast
Location: Suburban

Time: Afternoon
Weather: Sunny
Location: Urban

Time: Sunset
Weather: Rainy
Location: Highway

Time: Sunset
Weather: Overcast
Location: Highway

Time: Dusk
Weather: -
Location: Urban

Time: Sunset
Weather: Overcast
Location: Highway

Time: Sunset
Weather: Rainy
Location: Highway

Time: Afternoon
Weather: Rainy
Location: Rural

Figure 5: **Examples from the RoadNet Dataset.** This figure shows eight sample images from the RoadNet dataset, each annotated with its corresponding time of day, weather condition, and location. The examples reflect a variety of environments, including urban areas, highways, and rural roads, captured under different weather conditions such as sunny, rainy, and foggy scenes.

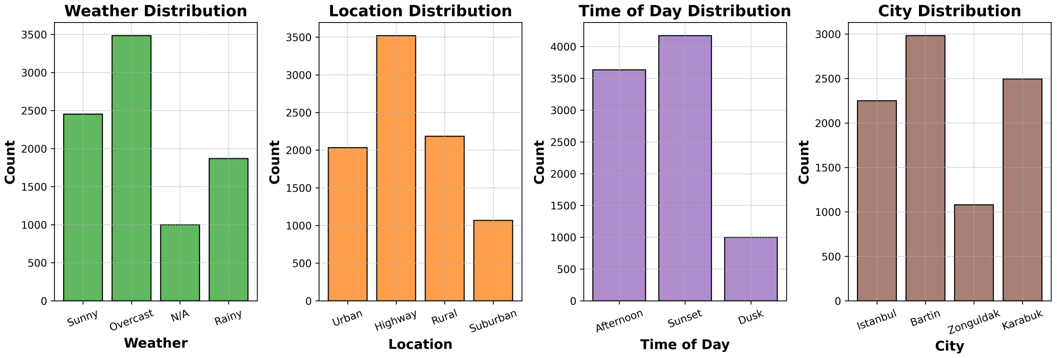


Figure 6: **RoadNet Dataset Statistics.** Distribution of RoadNet images across weather conditions, locations, times of day, and cities. The dataset exhibits substantial diversity, supporting robust training and evaluation of autonomous driving models.

E RoadNet Dataset

The RoadNet dataset consists of 25 video sequences, each ranging from 1 to 5 minutes in duration and recorded at a resolution of 1920 × 1080 × 3 pixels. We sample frames at 2 frames per second from each video, resulting in a total of 8,223 images. The recordings are captured from the front-facing view of a vehicle and cover a wide range of geographic locations, weather conditions, and times of day, examples from dataset is shown in Fig. 5. The dataset includes three weather conditions—clear, overcast, and rainy—as well as four scene types: urban, suburban, rural, and highway. Temporal diversity is also reflected by including recordings taken during the morning, afternoon, sunset, and dusk.

RoadNet is primarily collected from provinces in the Black Sea Region of Turkey, including Karabük, Bartin, and Zonguldak, along with samples from Istanbul. This geographic and environmental diversity provides a rich, realistic set of driving conditions, making RoadNet a valuable resource for training and evaluating vision models in autonomous driving applications. The distribution of data points across weather, location, time of day, and city is shown in Fig. 6.