SINGAPORE MANAGEMENT UNIVERSITY

Learning De-Biased Representations for Remote-Sensing Imagery

Conference on Neural Information Processing Systems 2024 (NeurIPS'24)

Background & Motivation

Challenges in RS domain • Current Solutions & Limits • Our Key Observations

What is Remote Sensing, and why research in this field is crucial.

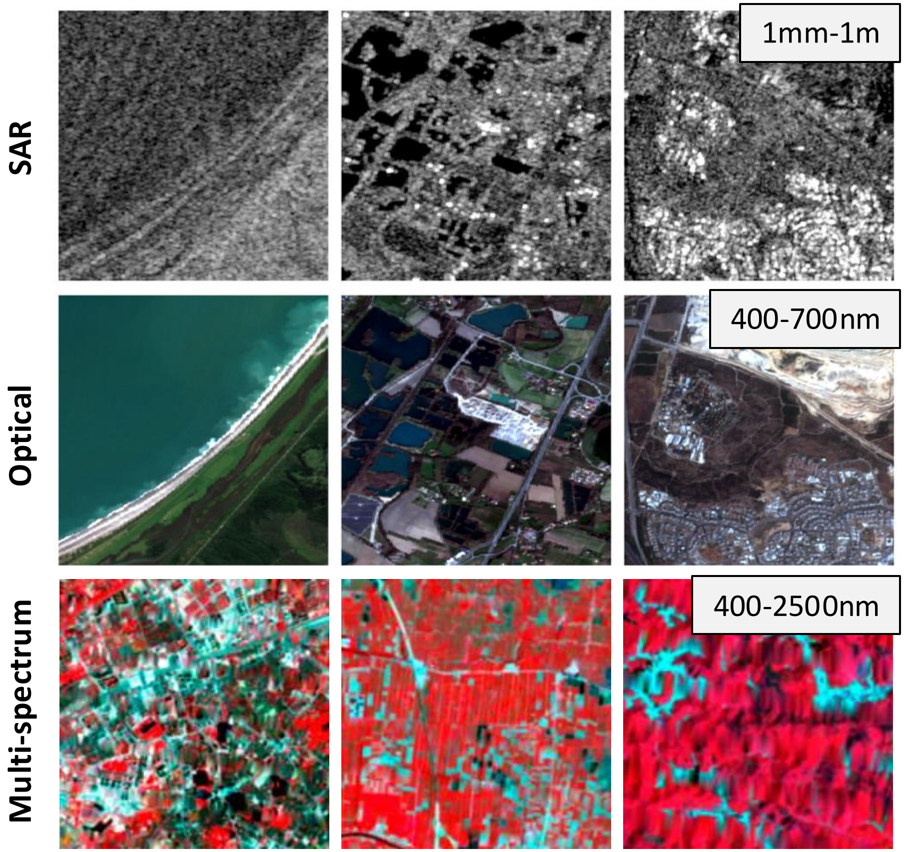
Remote Sensing Domain

- **Definition**

  Remote sensing images are captured from an overhead perspective by spaceborne or airborne sensors, which present unique viewpoints compared to natural images.

- **Multiple Spectrums**
  - Optical RS (ORS): 400-700nm
  - Multi-spectral RS (MSRS): 400-2500nm
  - Synthetic Aperture Radar (SAR): 1mm-1m

- **Key Applications**
  - Environmental monitoring
  - Resource management
  - Disaster response

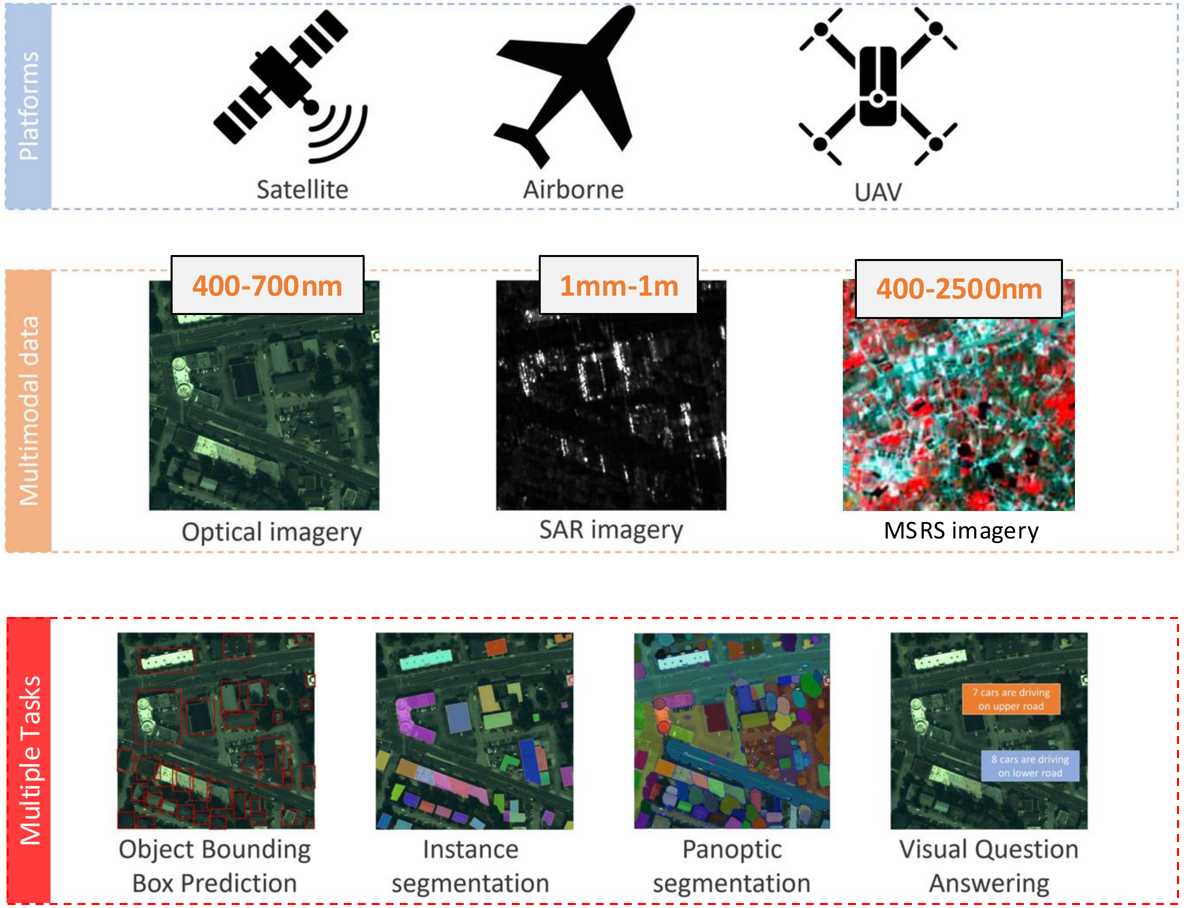


Source: EUSI Database

Remote Sensing data are diverse and complex, requiring heavy processing costs.

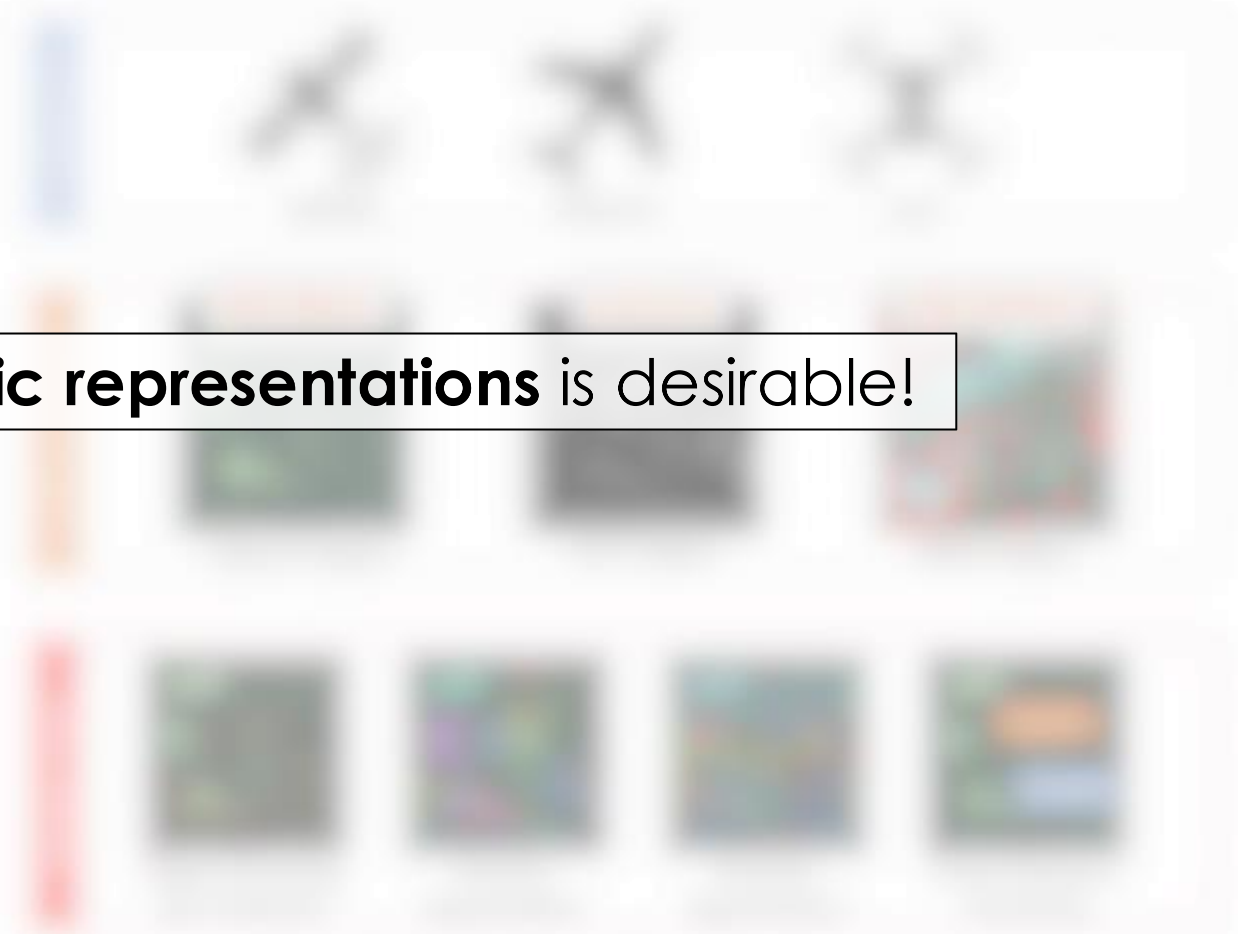
Challenges in RS Data

- **RS Data Diversity and Complexity**
  - Various data **source & processing tech**
  - Various **spectrums**
  - Various downstream **tasks**



Remote Sensing data are diverse and complex, requiring heavy processing costs.

Challenges in RS Data



Learning **robust and generic representations** is desirable!

Why not training from scratch?

Parameter Efficient **Transfer Learning**

- **Self-supervised Training from Scratch**
  - Data scarcity in certain spectrums (*e.g.*, **SAR** imagery)
  - Constraints in **model scale** and **data scale**
  - Constraints in **training GPU time**

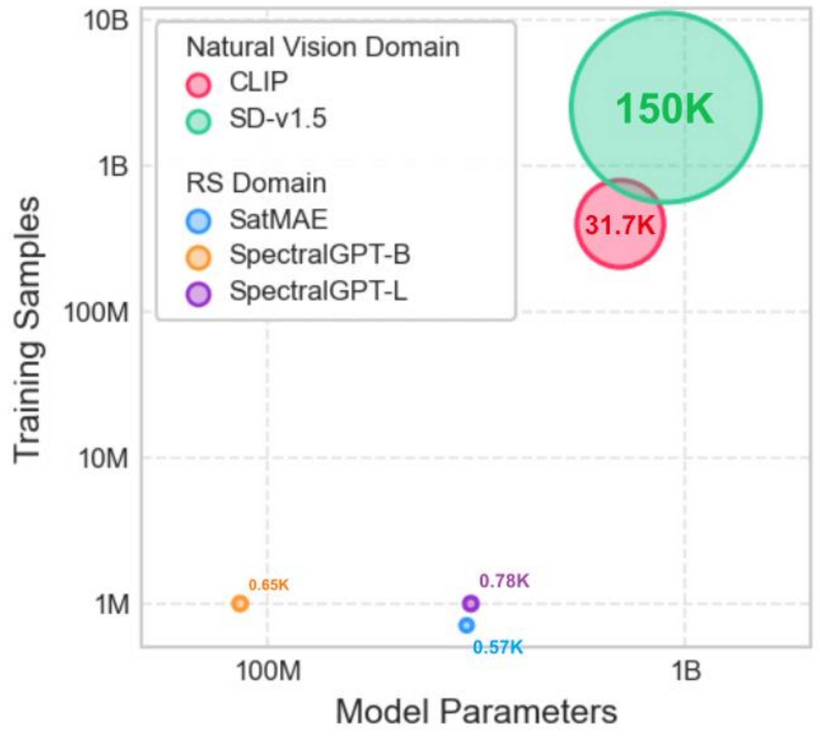
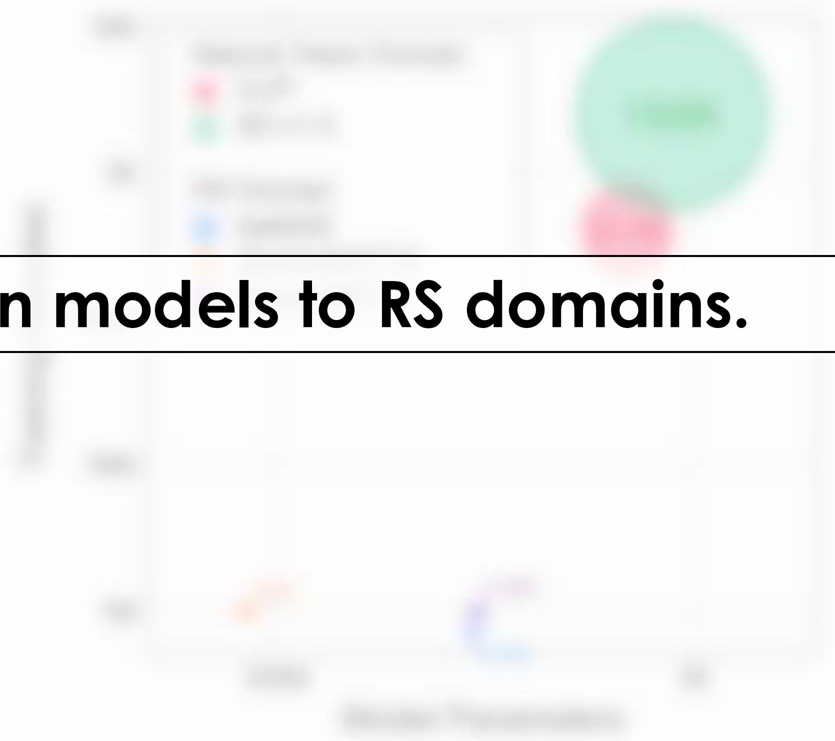


Figure: **Compare foundation models.** The bubble figure shows model scale, data scale and training time of five representative foundation models. Numbers near to bubbles are training GPU-hour. Models from RS domain uses less training GPU-hours compared with natural vision domain.

Why not training from scratch?

Parameter Efficient **Transfer Learning**



We propose to **transfer existing foundation models to RS domains.**

Why do we need parameter efficient?

Parameter Efficient Transfer Learning

- **Transfer Learning Setups**
  - Adaptation **from natural vision domain to RS domain**
  - Adaptation **between RS spectrums**
- **Zero-Shot and Fine-tuning**
  - Fine-tuning suffers from 1) catastrophic forgetting, 2) long training time, and 3) high VRAM usage.
  - Even zero-shot outperforms fine-tuning.
- **Parameter Efficient Transfer Learning (PEFT)**
  - **LoRA - Low Rank Adaptation**
  - Both fine-tuning, zero-shot and PEFT suffers from **long-tailed** distribution issue.

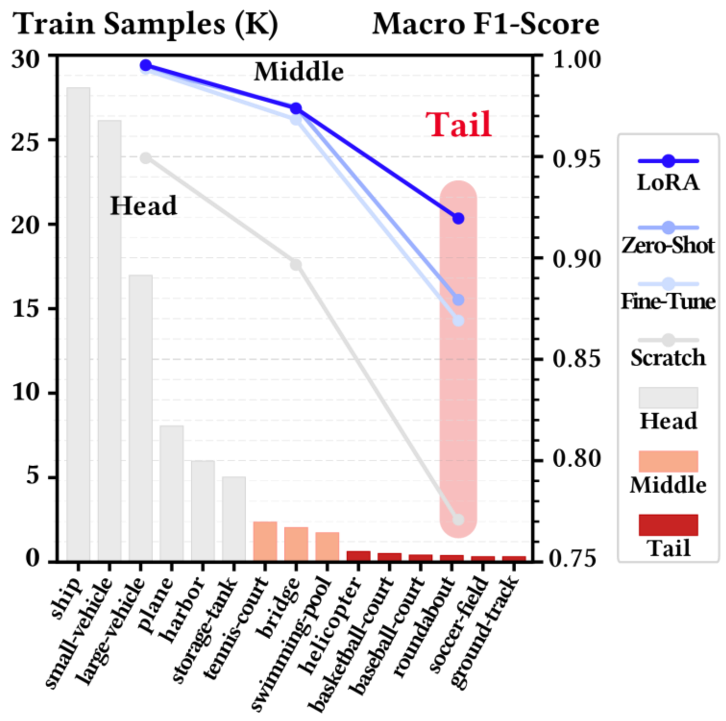


Figure: **Performance of Natural to ORS adaptation setting.** The debLoRA achieves highest performance, especially for tail class.

Insights & Design

Key Observations • Framework • Core Components • Algorithm Explanation

We observe that representation space learnt by PEFT methods are biased.

Key Observation – Biased Representation Space

- **Biased Representation Space**
  - When learnt on long-tailed data, LoRA's adapted **feature space** of LoRA **is biased[2]**.
  - Validation samples of **head class** are mostly **correctly** classified.
  - Validation samples of **tail class** are **wrongly** classified as head class.
  - Key Challenge: **Train/Val distribution mismatch** for tail classes.

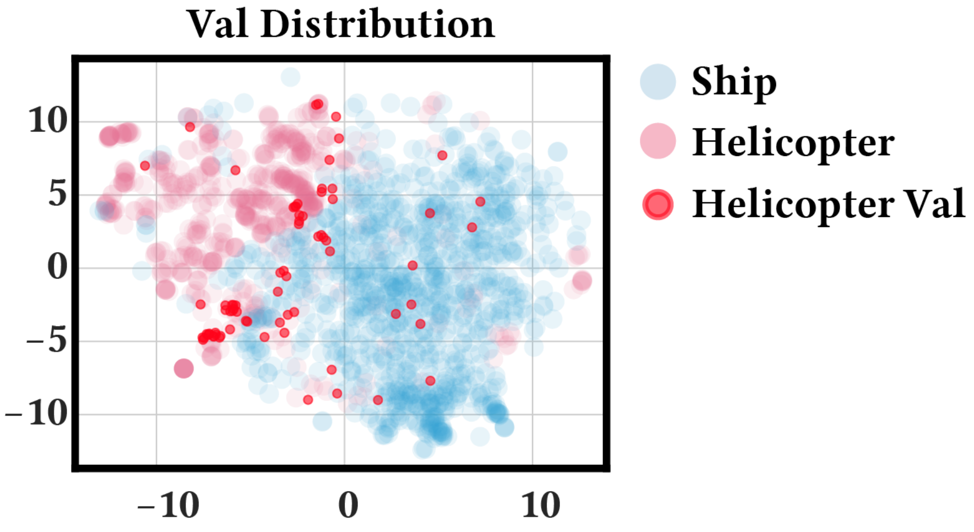
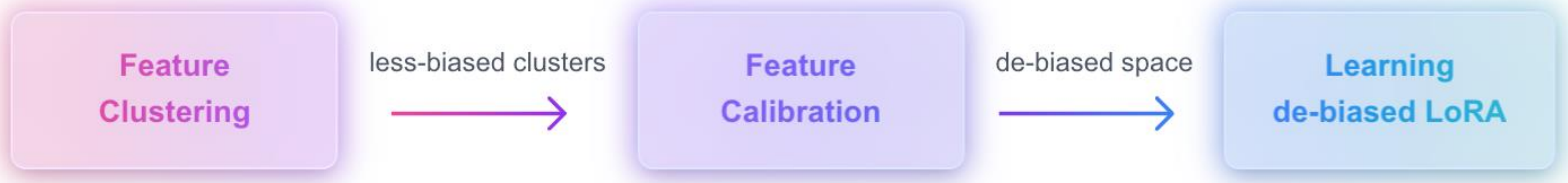


Figure: **Feature distribution of training samples.** For clearer visualization, we pick representative head class "Helicopter" and tail class "Ship" from DOTA v1 dataset as an example.

[2] We define feature space Z as biased if $Vol(Z_h) \gg Vol(Z_t)$, and $\exists\, z_t \in Z_t: P(z_t \in Z_h) > P(z_t \in Z_t)$, where Z_h and Z_t denotes the feature spaces of head and tail classes respectively, $Vol(\cdot)$ denotes feature space volume, and $P(\cdot)$ denotes the probability distribution predicted by the model.

Framework of Our Approach

- **Three key components**
  - **Feature clustering** – Unsupervised clustering to find less biased prototypes.
  - **Feature calibration** – Use less-biased prototypes to calibrate tail class features.
  - **debLoRA learning** – Learn a LoRA module to capture this de-bias mapping.



We first found balanced prototypes within feature space.

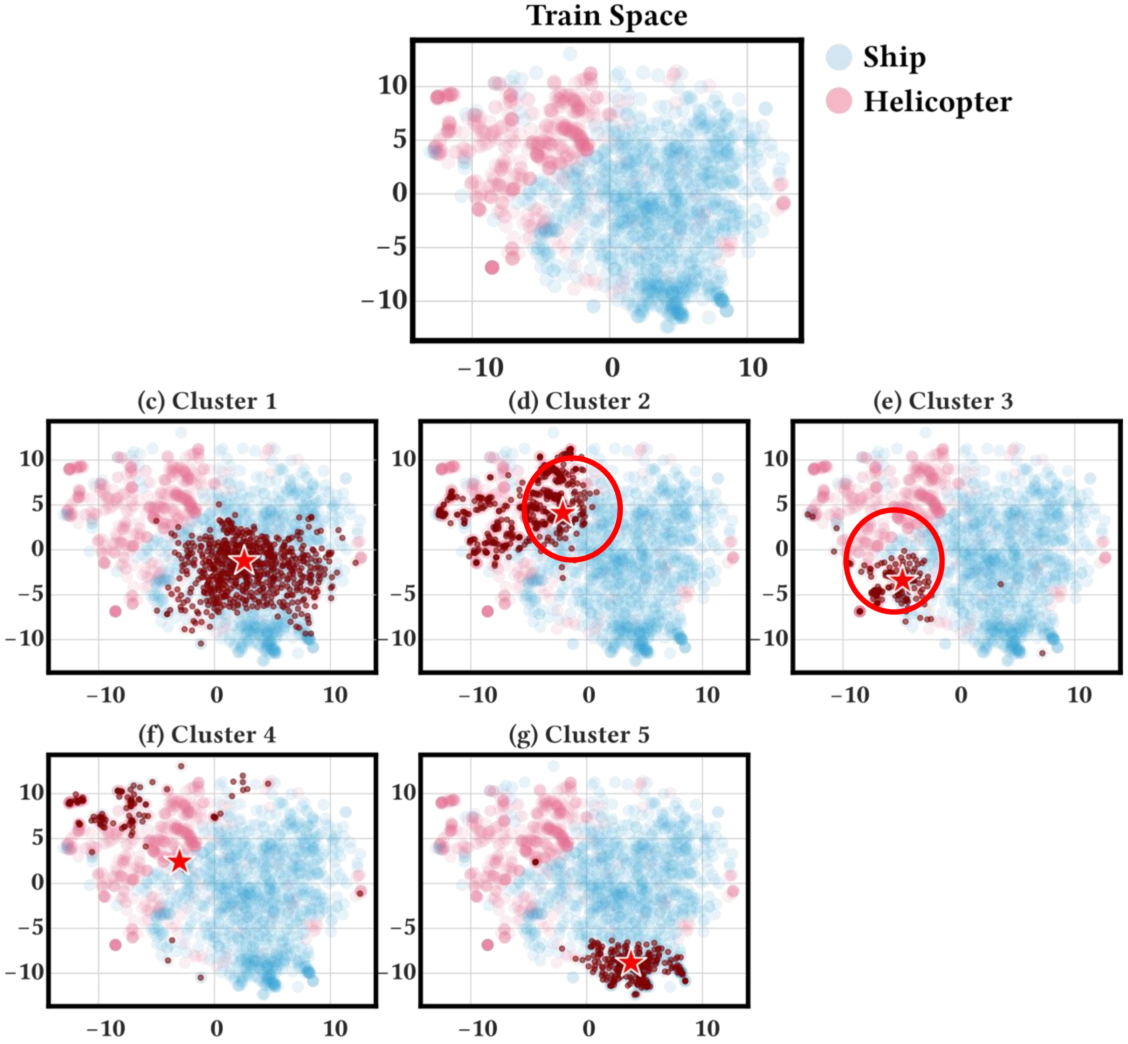
Feature Clustering

- **Feature clustering**
  - We conduct K-Means clustering over training samples' feature space.

$$\min_{\mu_k} \sum_{i=1}^{N} \min_{\mathrm{k}} \|z_i - \mu_k\|^2, s.t. \forall k, n_k \geq \frac{N}{K \cdot \rho},$$

where μ_k and n_k denote the center and size of the k-th cluster, respectively.

  - Some cluster centers are contributed by both head and tail classes, and hence is less biased (*e.g.*, clusters 2 and 3).



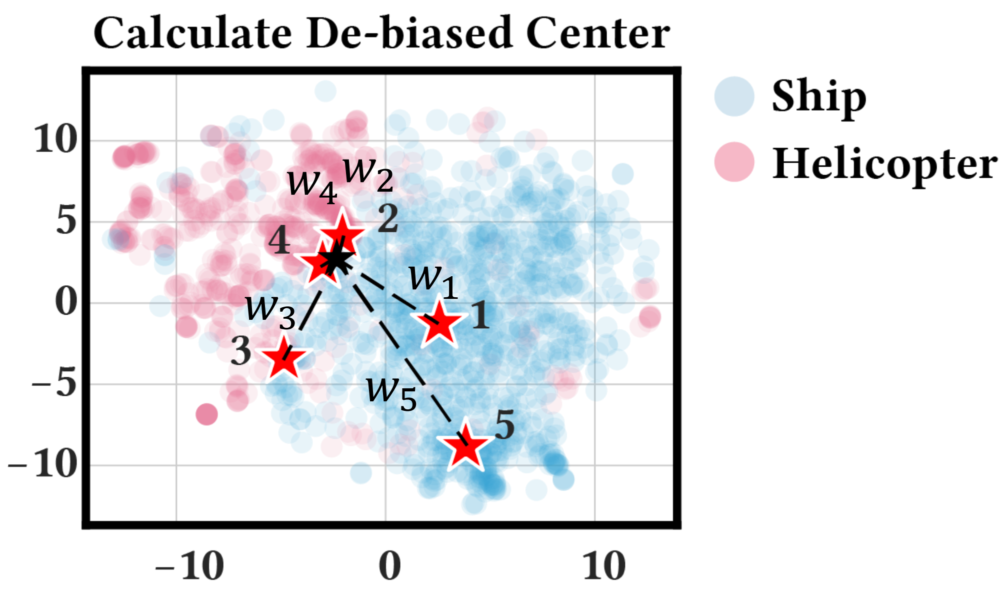
We secondly construct less biased centers and calibrate features.

Construct De-Biased Center

- **De-Biased Center**
  - We calculate de-biased representation center for each tail class:

$$\hat{\mu}_c = \sum_k w_k \cdot \mu_k \, , w_k = \frac{n_k}{n_c},$$

    here weight w_k proportion to the fraction of class c samples in k-th cluster.
  - This ensures that the de-biased center $\hat{\mu}_c$ is not dominated by head classes



We utilize LoRA to capture the de-bias mapping.

Feature Calibration

- **Tail Class Calibration**
  - De-Biased Center are **closer to validation** samples.
  - We calibrate tail class features z by moving them close to de-biased center $\hat{\mu}$:

$$\tilde{z} = \alpha z + (1 - \alpha)\hat{\mu},$$

where $\alpha = \min(1, \frac{10}{ir})$ empirically.

- **Learning debLoRA**
  - We learn an LoRA module with training objective

$$\min_{\phi} \frac{1}{D_t} \sum_{x \in D_t} \left\| g_\phi(f_\theta(x)) - \tilde{z} \right\|^2$$