ON THE JOINT INTERACTION OF MODELS, DATA, AND FEATURES

Yiding Jiang
Carnegie Mellon University
yidingji@cs.cmu.edu

Christina Baek
Carnegie Mellon University
kbaek@cs.cmu.edu

J. Zico Kolter
Carnegie Mellon University
Bosch Center for AI
zkolter@cs.cmu.edu

Goal

- Gaining a better understanding of the types of features learned by different models
- Understanding how ‘features are distributed in the data and how models with different seeds learn different features’

Background

- Training models with different seeds leads to different predictions
- The ensemble is usually better calibrated and has higher performance
- You can use the disagreement between models to estimate accuracy:
    - Generalised Disagreement Equality
        - The accuracy of model m1 is equally to the average number of points where it agrees with model m2 trained with a different seeds.
        - **In short, the model is likely to be correct when two models agree, likely to be wrong when they disagree**
        - **The authors try to understand this from a feature point of view**

Q1: how to define ‘features’ in DL models?

- One can *think of defining features as quantifying the “unit” of information in data that models use to make predictions*
- However this is not a very precise definition

- Here the author define two concepts

Features

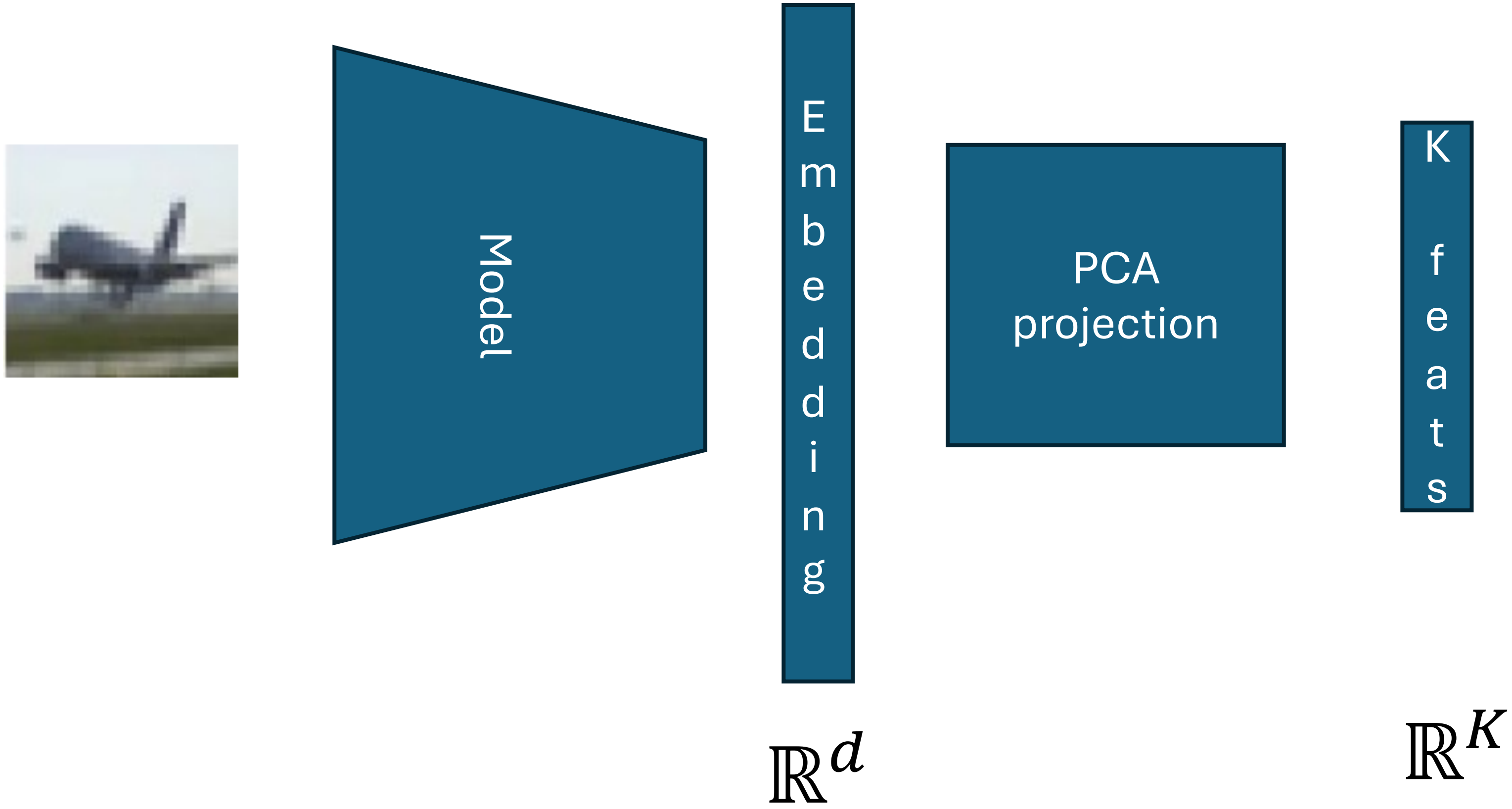
- PCA projection of last layer embeddings

Interaction tensor

- Jointly models features learned across models

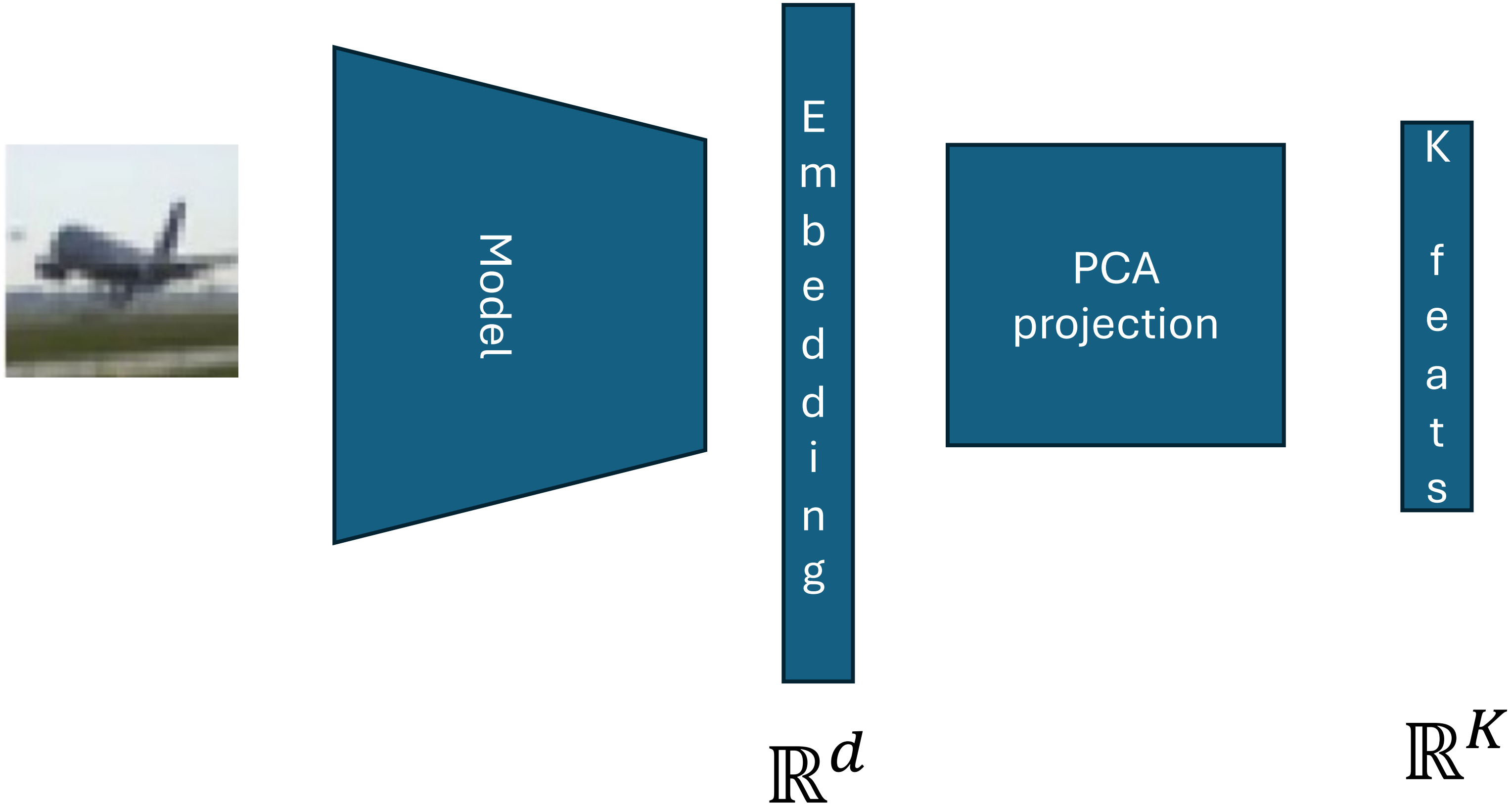
Definitions of features and interaction tensor

- Features



Definitions of features and interaction tensor

- Features



They denote the **'a-th feature of the model'** as a-th dimension of the projected embeddings.

In total **each model has K features.**

How to match ‘features’ to data points

- For every data point we want to know **which ‘features’ are present**.
- Process is simple:
  - Compute the PCA projection $v_m(x) \in R^K$
  - Normalise by l-inf norm to have it in [0,1]
  - **If the k-th dimension of projected embedding i.e. $v_m(x)_k$ is higher than a certain threshold consider that datapoint x contain feature k of model m.**

Interaction tensor

- We are interested in knowing if different models learn the same ‘features’
- So we need **to match them across models** (cause even if they would learn exactly the same features they could be permuted)
- The author propose a way to do this matching, they call ‘feature clustering’.

Interaction tensor

- For each model, for every point x we collect all the K PCA features.
- Then we can compute the empirical covariance between any feature a of model f_1 and feature b of model f_2 across all data points.
- The idea is simple **if two features correlate perfectly between two models across the dataset they can be considered as the same feature**.

Interaction tensor

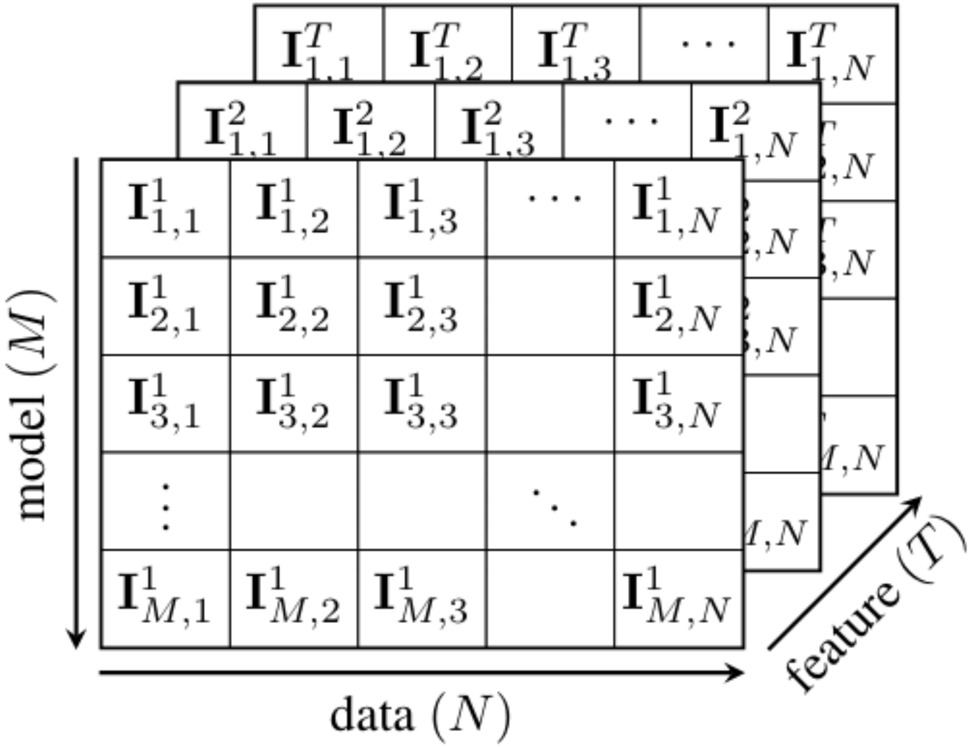
- To compute the interaction tensor
  - 1. Get all projected PCA features for every model and every datapoint
  - 2. Compute the covariance between every pair of features for any model pairs. We get $\Lambda \in [-1,1]^{M \times M \times K \times K}$ the collection of all the correlation matrices between every pair of models where $\Lambda_{i,j,a,b}$ is the correlation between the a-th feature of model i and the b-th feature feature a for model
  - 3. Apply greedy clustering algorithm on the correlation matrix to determine which feature are the same across model. **In short if correlation is very high -> same feature.**

Interaction tensor

- After doing this feature clustering, we end up with T features learned across the models collection.
  - Some will be common across all models
  - Some will be model specific

- We can now compute which feature are present in the learned representation of each data point for every model.

Interaction tensor



$\mathbf{I}^T_{1,1}$	$\mathbf{I}^T_{1,2}$	$\mathbf{I}^T_{1,3}$	$\cdots$	$\mathbf{I}^T_{1,N}$

$\mathbf{I}^2_{1,1}$	$\mathbf{I}^2_{1,2}$	$\mathbf{I}^2_{1,3}$	$\cdots$	$\mathbf{I}^2_{1,N}$

$\mathbf{I}^1_{1,1}$	$\mathbf{I}^1_{1,2}$	$\mathbf{I}^1_{1,3}$	$\cdots$	$\mathbf{I}^1_{1,N}$
$\mathbf{I}^1_{2,1}$	$\mathbf{I}^1_{2,2}$	$\mathbf{I}^1_{2,3}$		$\mathbf{I}^1_{2,N}$
$\mathbf{I}^1_{3,1}$	$\mathbf{I}^1_{3,2}$	$\mathbf{I}^1_{3,3}$		$\mathbf{I}^1_{3,N}$
$\vdots$			$\ddots$	
$\mathbf{I}^1_{M,1}$	$\mathbf{I}^1_{M,2}$	$\mathbf{I}^1_{M,3}$		$\mathbf{I}^1_{M,N}$

Figure 6: An illustration of the interaction tensor, $\mathbf{\Omega}$. The three axes correspond to *model*, *data*, anc *features*. An entry $\mathbf{I}^t_{m,n}$ is 1 if both data n and model m contains feature t and is 0 otherwise.

O0: Images with least features are closer to class prototypes. Weird cases have more features.

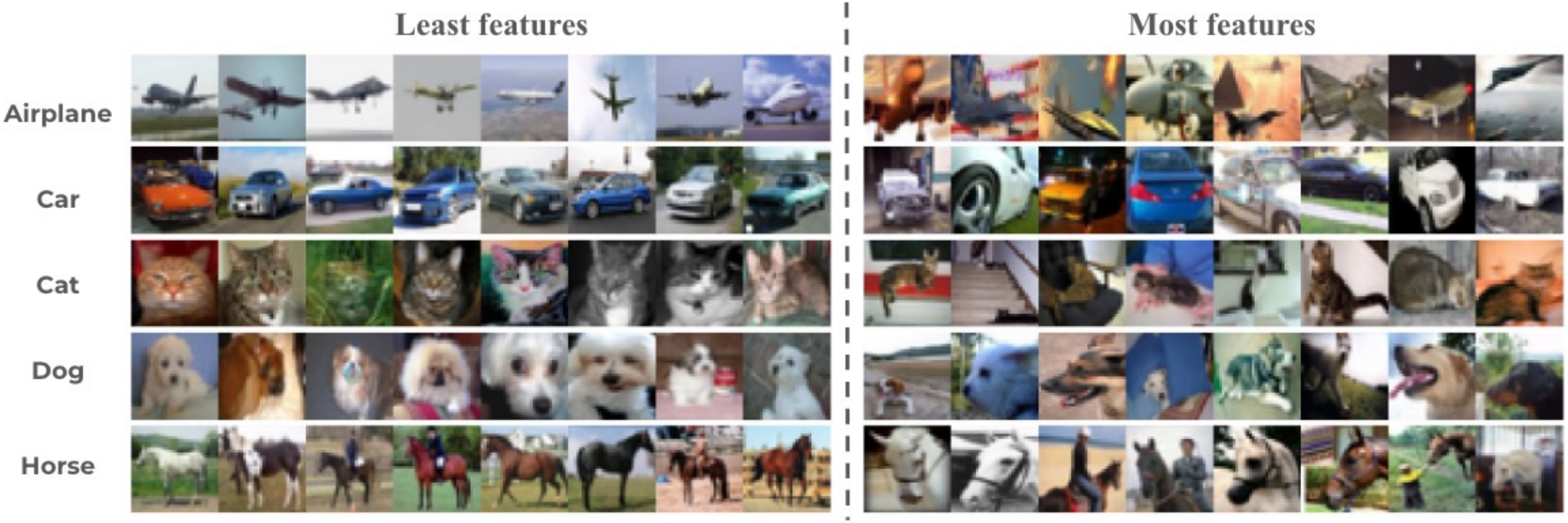
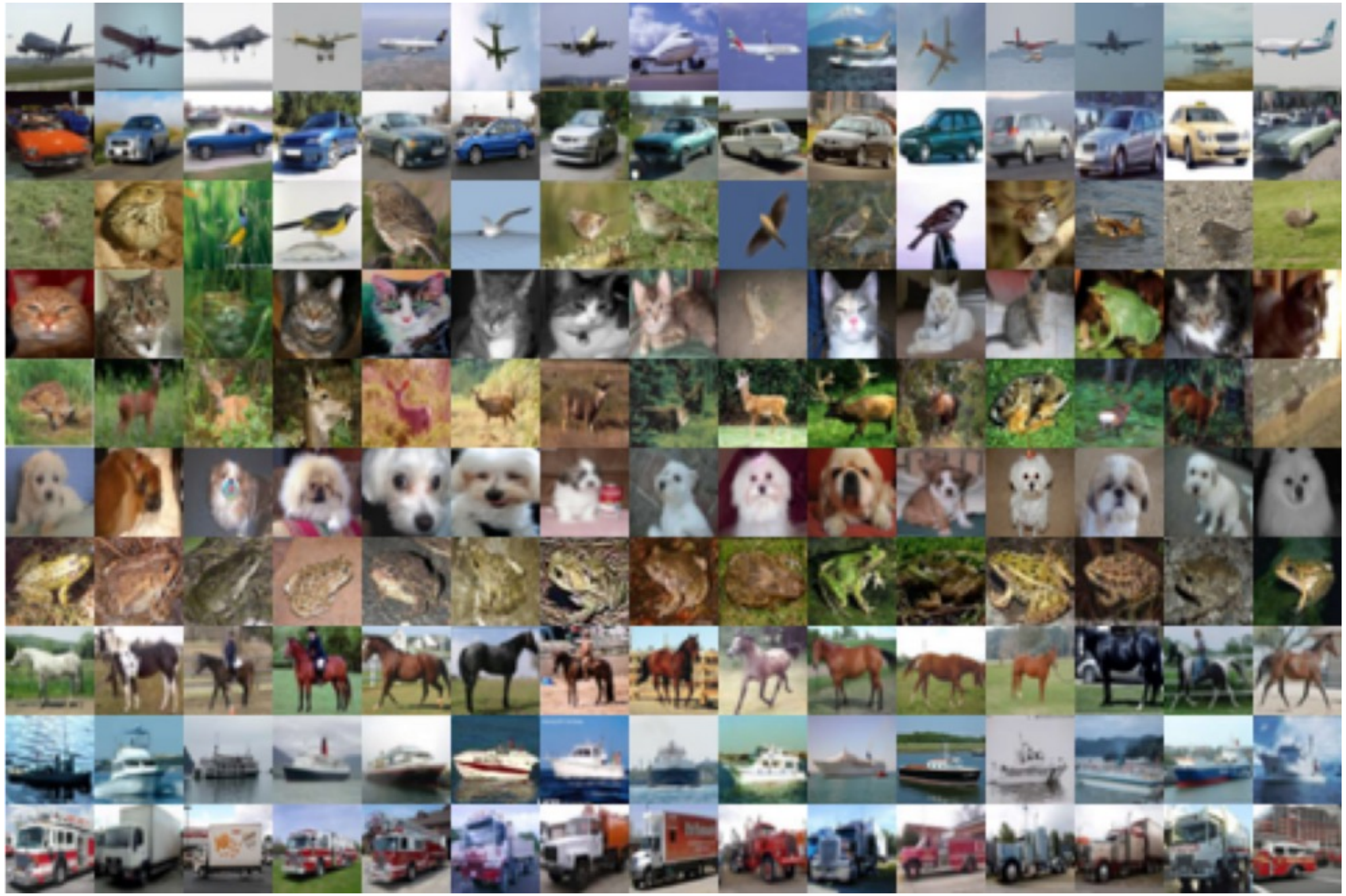


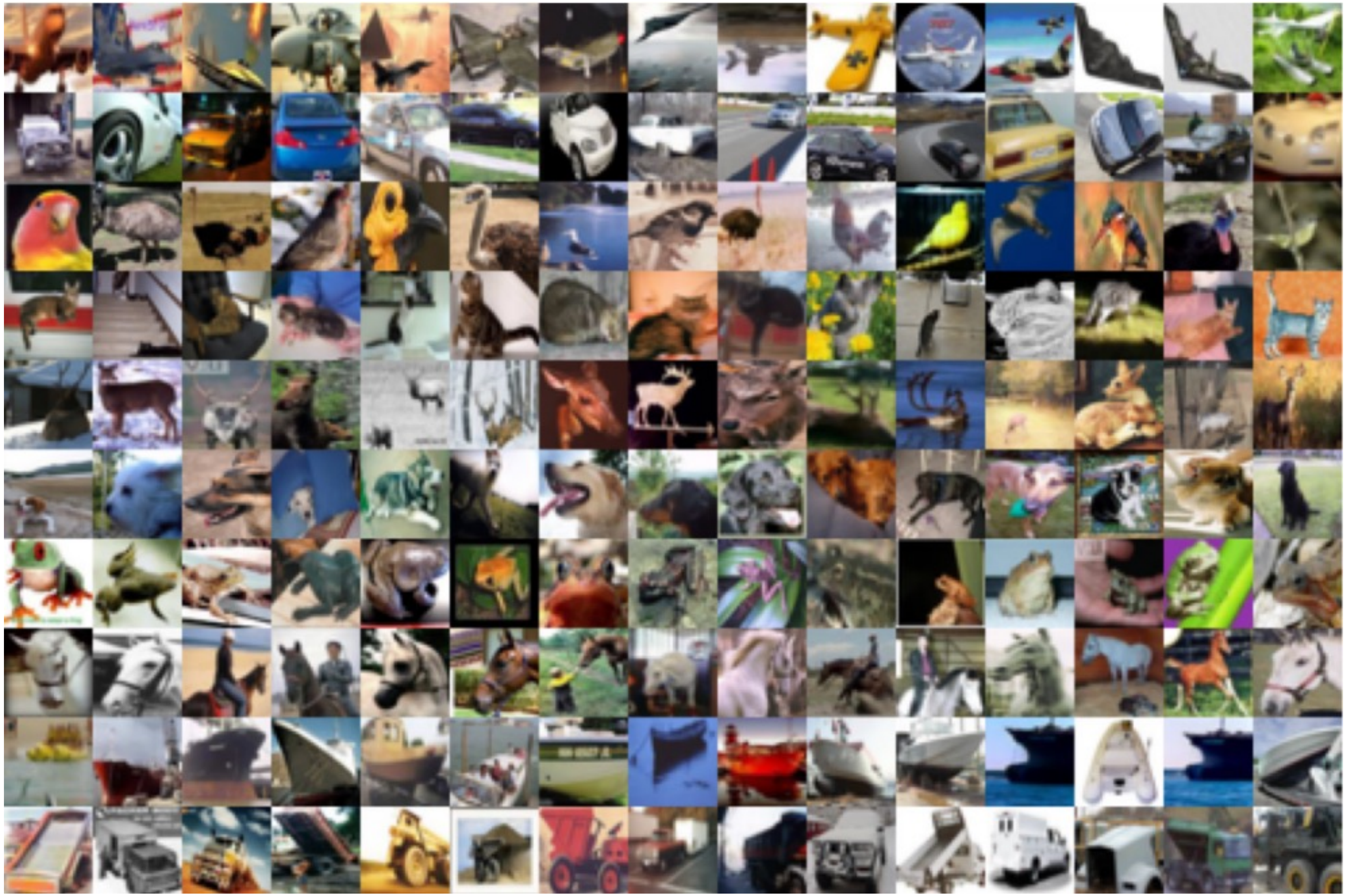
Figure 1: Visualization of images with *the least features* (left) and *the most features* (right) for classes of CIFAR 10 under our feature definition (defined in Section 3.1). Each row corresponds to one class of CIFAR 10 (zoom in for better viewing quality). We can see that the images with the least features are semantically similar to each other whereas the images with the most features are much more diverse and contain unusual instances of the class or objects in rare viewpoints. More examples for all classes can be found in Figure 12 of the appendix.

O0: Images with least features are closer to class prototypes. Weird cases have more features.



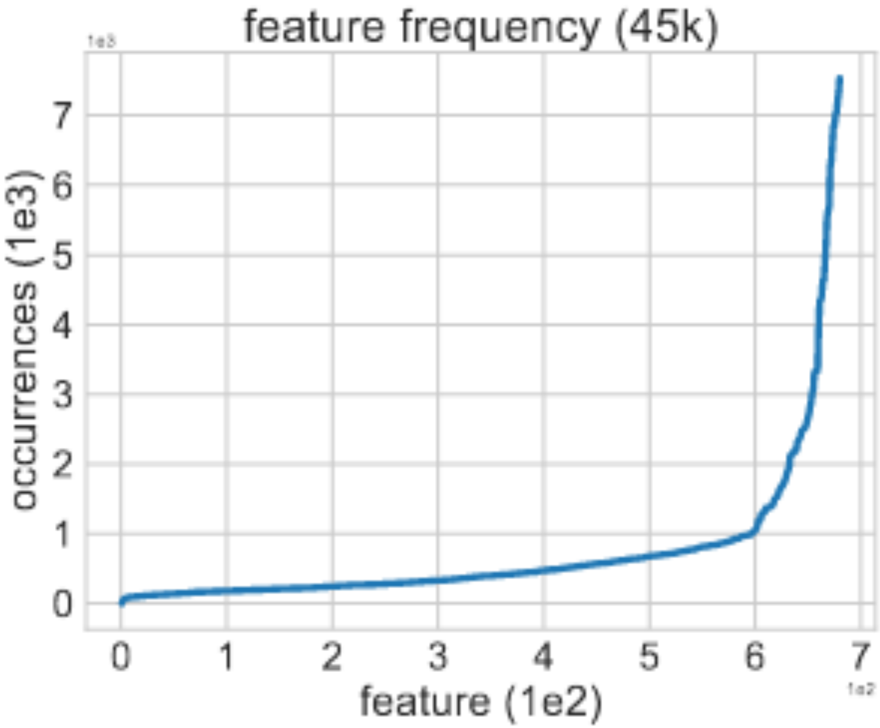
Points with **least number of features.** Very uniform.

O0: Images with least features are closer to class prototypes. Weird cases have more features.



Points with **most number of features**. Very diverse, loads of weird images.

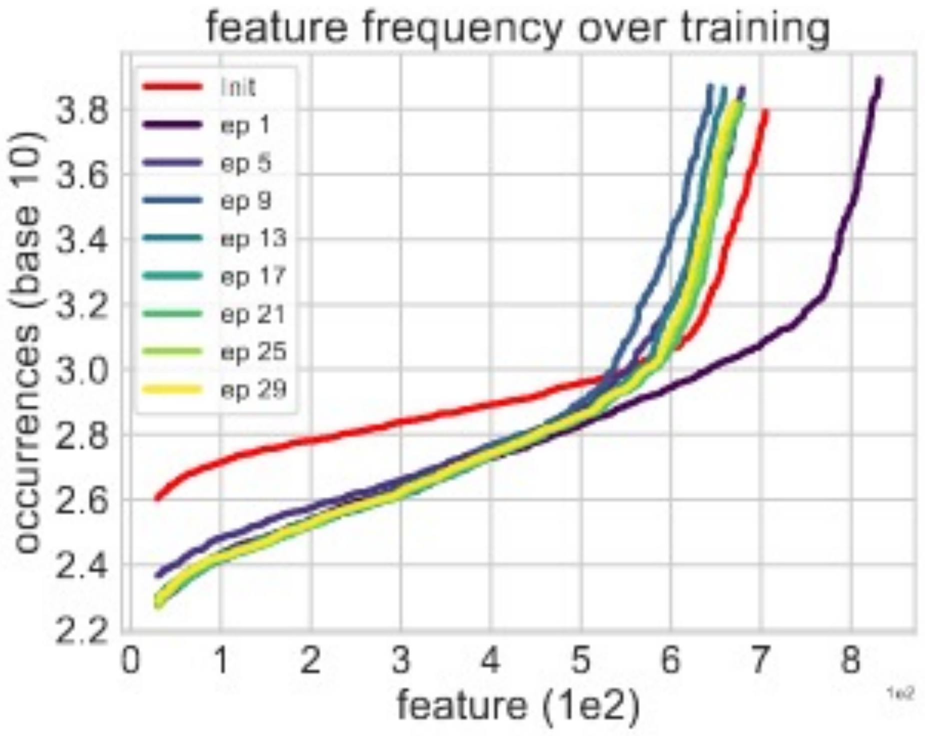
O1: Feature frequency is long-tailed.



Features vs how often they appear in the dataset. The features are sorted by frequency and the distribution appears to be long-tailed.

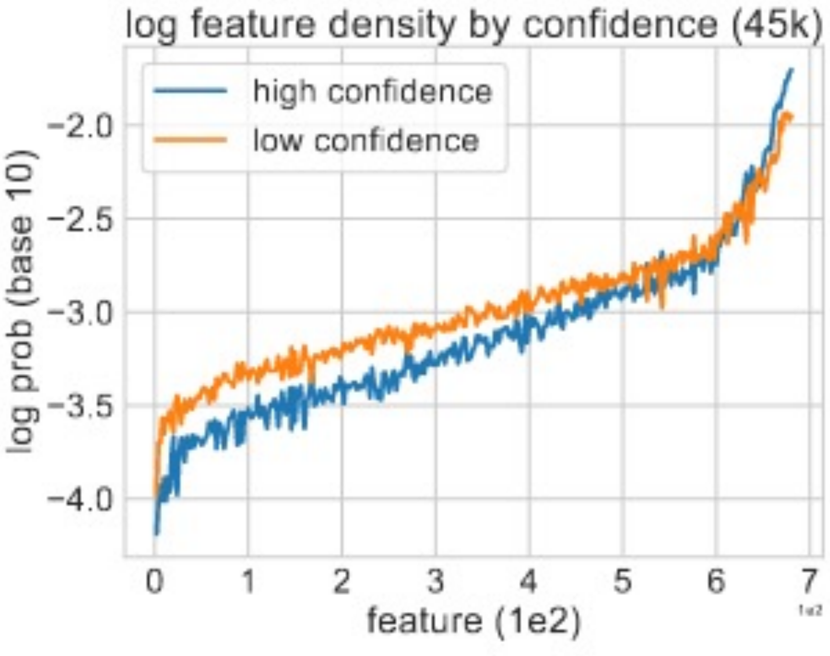
> Most data point are concentrated on a few features

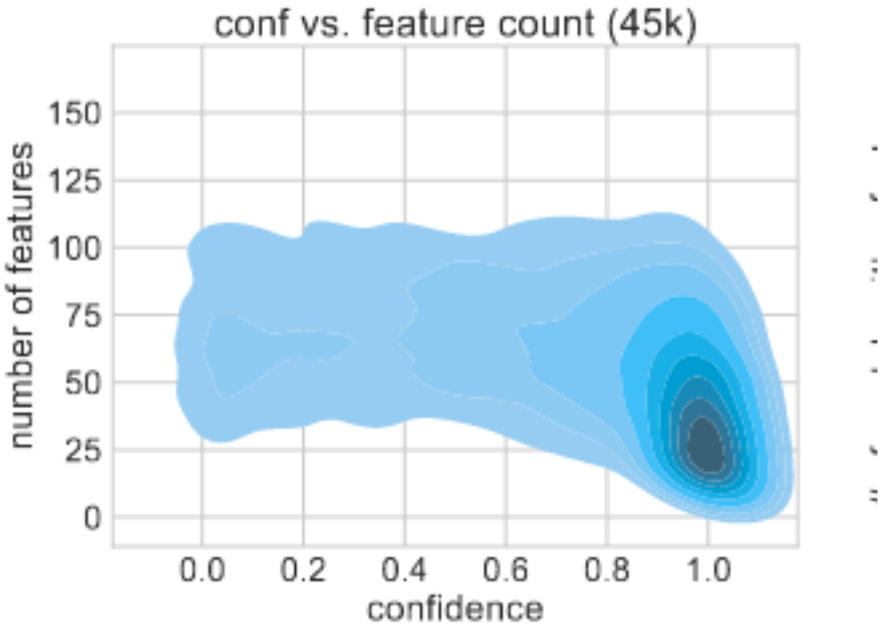
Evolution of features during training



At the head of the distribution, **the models first learn a large number of features and then prune the features as training continues**, eventually converging to a fixed distribution with a small number of features.

O2: The ensemble tends to be more confident on data points with fewer features

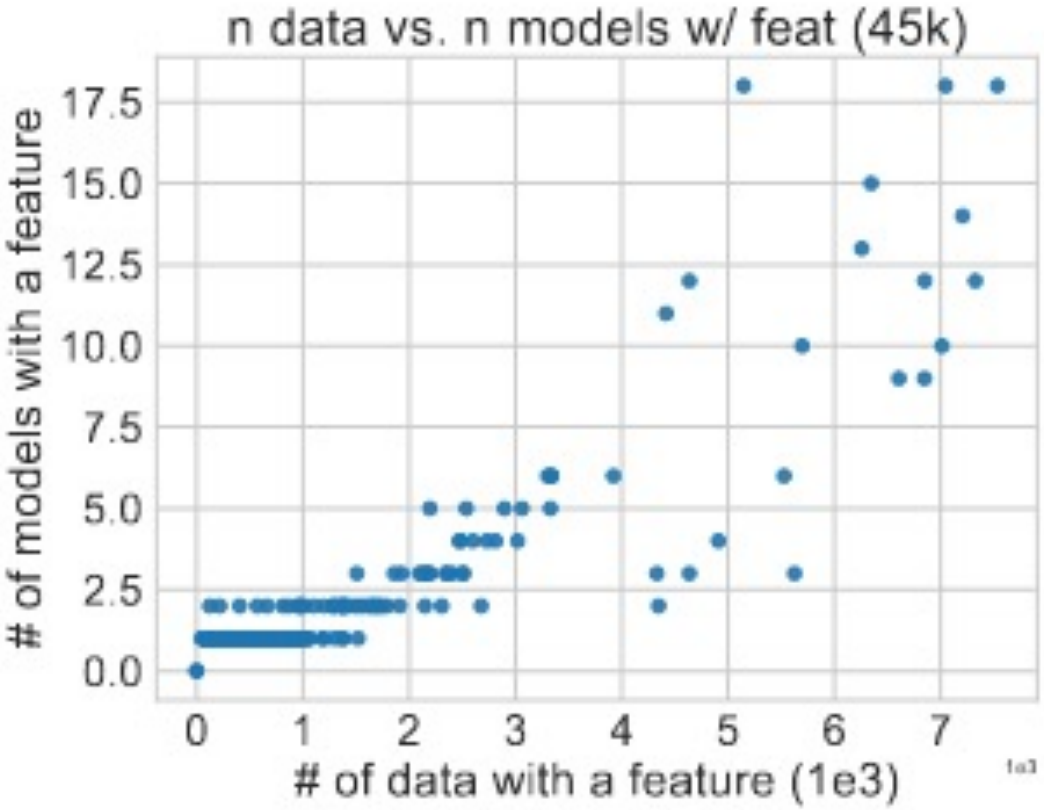




Feature frequency by different confidence levels. Low-confidence data tend to have more low-frequency features.

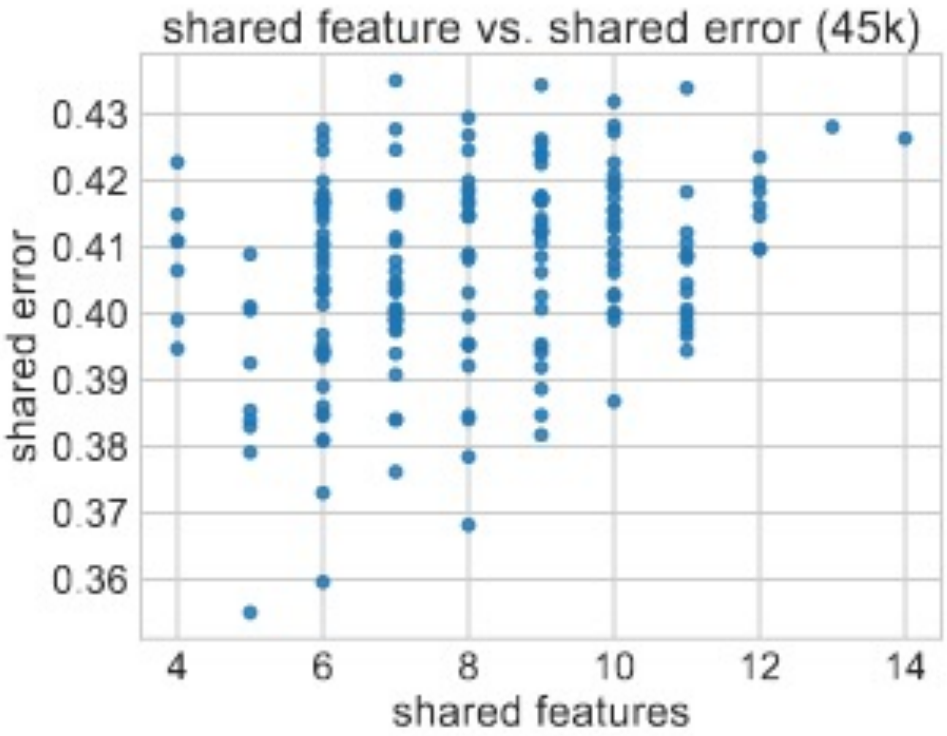
> Data points that have rarer features are more difficult to classify

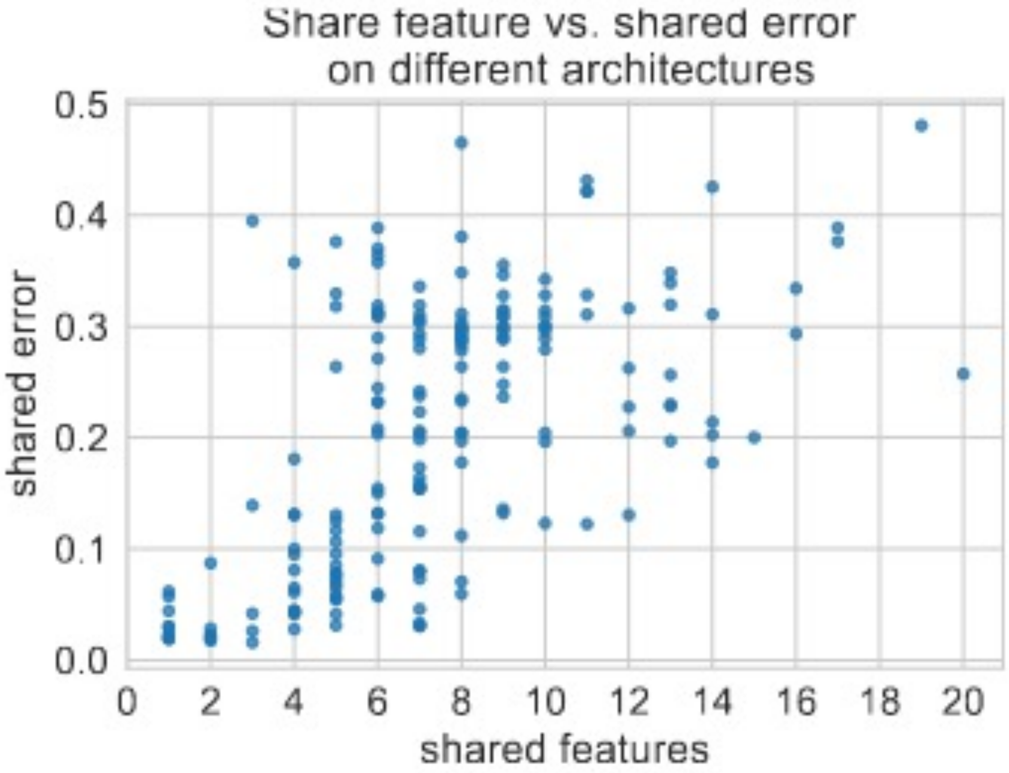
O3: Number of models with a certain feature is positively correlated with the feature's frequency



> The more data has a certain feature, the more likely the model will learn that feature

04: Models with shared features make similar mistakes !





- ➢ The lower bound of shared error monotonically increases with the number of shared features.

- ➢ Even more true for different architecture

What are the implication for disagreement equality?

- GDE:
  - Estimate model accuracy by average of data points were two models trained with different seeds agree.
  - It works most of the time but for some dataset shifts it breaks.
- The authors hypothesise that if the test set has the same ratio of data points with dominant/rare features as in the training set agreement will work. **If your shifted test set has more data points with rare features it will fail.**

What are the implication for disagreement equality?

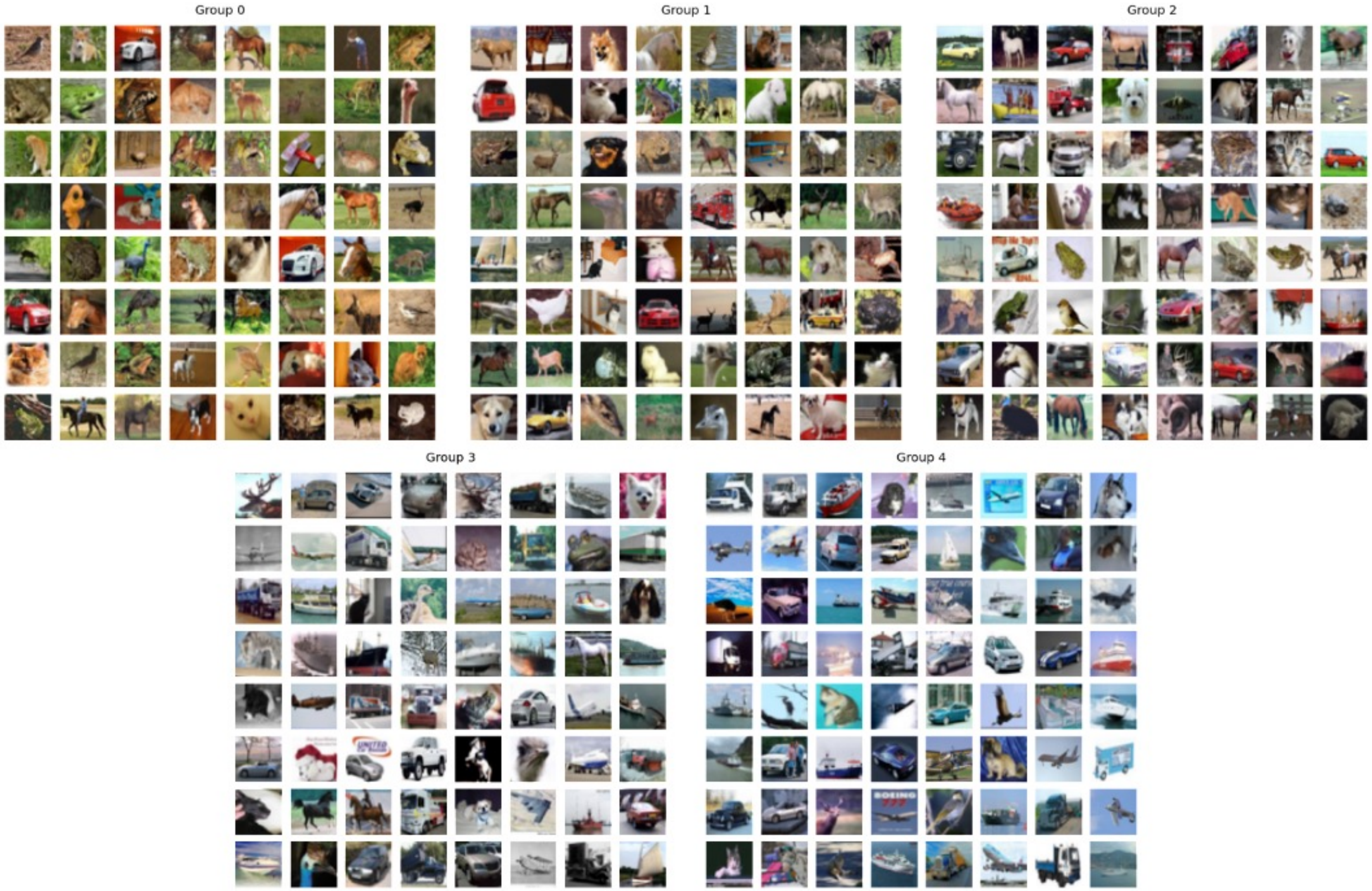
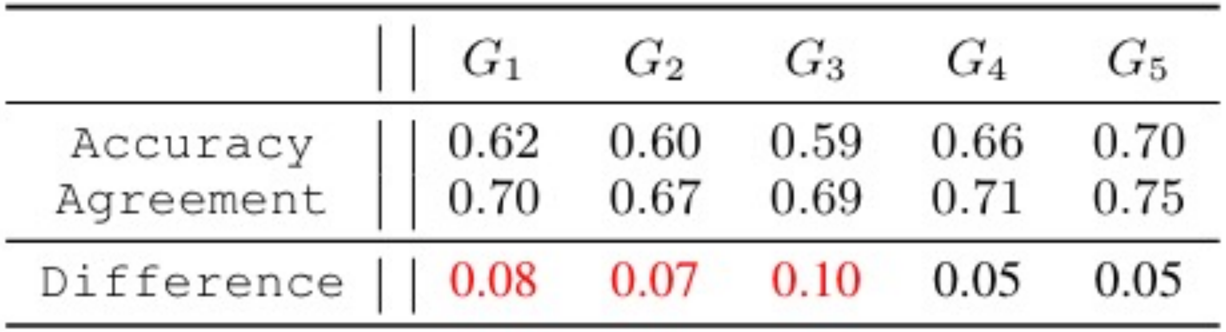


Figure 17: Visualization based on data partitioning based on blue intensity.

What are the implication for disagreement equality?



	G_1	G_2	G_3	G_4	G_5
Accuracy	0.62	0.60	0.59	0.66	0.70
Agreement	0.70	0.67	0.69	0.71	0.75
Difference	0.08	0.07	0.10	0.05	0.05

The more diverse the set is (i.e. the more rare features) the more error the GDE makes in estimating accuracy. That is the worse the ensemble is calibrated.

Note here the data is not OOD, these are all CIFAR10 but G1 has more difficult examples.

Conclusions

- Framework to inspect features learned by different models
- Insights about features versus ensemble learning:
  - **Most data points have only a few features that are shared across models.**
  - **Feature distribution is long tailed, data points with the most number of features also have the rarest features and are ‘weird’**
  - **Data points with rarer features are classified with less confidence**
  - **The more features overlap across models the more similar is the error i.e. the better GDE can estimate model performance.**

What’s your opinion?