Last Layer Retraining of Selectively Sampled Wild Data Improves Performance

by

Hao Bang Yang

S.B., Computer Science and Engineering and Physics, Massachusetts Institute of Technology (2022)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

Authored By:	Hao Bang Yang Department of Electrical Engineering and Computer Science May 12, 2023
Certified by:	Justin Solomon Associate Professor Thesis Supervisor
Certified by:	Mikhail Yurochkin Research Staff at MIT-IBM Watson AI Thesis Supervisor
Accepted by:	Katrina LaCurts Chair, Master of Engineering Thesis Committee

Last Layer Retraining of Selectively Sampled Wild Data Improves Performance

by

Hao Bang Yang

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 2023, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

While AI models perform well in labs where training and testing data are in a similar domain, they experience significant drops in performance in the wild where the data can lie in domains outside the training distribution. Out-of-distribution (OOD) generalization is difficult because these domains are underrepresented or non-existent in training data. The pursuit of a solution to bridging the performance gap between in-distribution and out-of-distribution data has led to the development of various generalization algorithms that target finding invariant/"good" features. Recent results have highlighted the possibility of poorly generalized classification layers as the main contributor to the performance difference while the featurizer is already able to produce sufficiently good features.

This thesis will verify this possibility over a combination of datasets, generalization algorithms, and training methods for the classifier. We show that we can improve the OOD performance significantly compared to the original models when evaluated in natural OOD domains by simply retraining a new classification layer using a small number of labeled examples. We further study methods for efficient selection of labeled OOD examples to train the classifier by utilizing clustering techniques on featurized unlabeled OOD data.

Thesis Supervisor: Justin Solomon
Title: Associate Professor

Thesis Supervisor: Mikhail Yurochkin
Title: Research Staff at MIT-IBM Watson AI

Acknowledgments

First and foremost, I would like to thank my advisor at MIT-IBM Watson who oversaw my time as a 6A Intern: Mikhail Yurochkin. Mikhail has been extremely supportive of me throughout my journey from me joining IBM, in the summer of 2022, without a single idea of what my thesis is going to be to now as I am finishing up the last parts of my writing. He has provided me with invaluable guidance in whatever area I needed help in, especially with suggesting the topic of research that I am currently working on. As my mentor for the good part of half a year, Mikhail was always patient with my mistakes and helped me become a better researcher and collaborator. I am extremely grateful for all the time Mikhail had spent helping me because my thesis wouldn't exist otherwise, and all the academic papers he seemed to have ready on hand.

I would also like to thank my MIT thesis advisor Justin Solomon. While I had not been the greatest of advisees, Justin was always patient with my mistakes and was always there to provide feedback. The time he spent, helping break down every part of my project from motivation to results has been very helpful in writing my thesis and adding information that I lacked in my project.

Finally, I would like to thank my friends and family for all the support and joy they gave me over the past year. Thank you to my mom who didn't really understand my thesis but always supported my choices and actions. Thank you to my girlfriend, Julie, who is always there to listen to my complaints during our 10PM calls, help me solve any issue that I'm dealing with through my time writing my thesis, and never afraid to tell me if I'm doing something wrong. Thank you to my brothers at the Pi Lambda Phi fraternity who I knew I could always depend on when I needed to take my mind off my thesis and a place to sleep if I'm in the area; my family line - Tony, Andrew, Josh, Mason, and Kevin - who always put up with my antics and a great source of laughs when I needed it. Thankful for my friends at Site 4 and Meng Fam who shared my misery in writing a thesis and always were a source of information, help, and general happiness.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

A major challenge facing machine learning is the disconnect in performance between state-of-the-art models testing on data in the distribution (ID) that it was trained on and those models testing on out-of-distribution (OOD) data that is underrepresented in the training data. [15, 14]. Neural networks classically train a featurizer that produces rich generalized features and a classifier that processes those features to a certain output; the featurizer tends to be complex but the classifier tends to be a linear classifier such as logistic regression. While this setup works well when both training and testing data lie within the same distribution, the default featurizer may not be able to remove all spurious correlations and may even enhance some "shortcuts" in the features [3] which cause the classifier to fail to generalize. Algorithms such as Empirical Risk Minimization (ERM) [23], deepCORAL [21], and Domain-Adversarial Neural Network (DANN) [2] seek to address this issue by building a more robust featurizer during the training that learns to disregard spurious correlations and in turn, learning a generalized classifier.

Recent advancements in these algorithms' implementations [1, 19, 20] focus on improving the entire network, such as a new objective function [16], but have resulted in diminishing returns in the pursuit of better performance. We remember that in a classical neural network, the featurizer and classifier are separable and generalization algorithms try to build good featurizers that remove all spurious correlations. Thus, we hypothesize that the issue of poor OOD performance lies in the algorithms

learning a good featurizer with a poorly generalizing classifier so we are able to use the featurizer to train a new classifier using a subset of the OOD data set to significantly improve the performance on the OOD domains. However, the OOD data set is unlabeled, and training a new classifier requires labeled data. We accept the trade-off that we are able to query a certain data point for its label but labeling is a very expensive process. Consequently, training a new classifier must use as little of labeled examples, also known as shots [24, 13], as possible to minimize the expense of labeling points (low budget scenario) [5, 11].

This work seeks to verify the hypothesis that the algorithms learn a good ID featurizer/bad classifier by freezing the original ID featurizer and training a new classifier using the OOD data set, demonstrating that training a new classifier using the ID featurizer requires very few shots, and proposing various sampling algorithms to further improve the performance and fairness of the algorithm on OOD data. The crucial point to address is the high expense of labeling OOD data points in order to train the new classifier so the number of shots must be minimized; we still have access to the full unlabeled OOD data set.

1.1 Background and Related Work

1.1.1 WILDS

The WILDS benchmark contains 10 diverse data sets that demonstrate two kinds of distribution shifts seen in the wild [8]: domain generalization and subpopulation shift. Domain generalization occurs when the train and test data distributions are similar but come from unique domains; in practice, it is impossible to find labeled training data from all possible domains so some domains are not represented in training but may be seen when the model is deployed. Subpopulation shift occurs when the proportions of subpopulations in the training distribution are drastically different from that of the test distribution. In this work, we will be focusing on two domain generalization data sets - iWildCam and Camelyon-17 - to generate our ID/OOD

data sets. Both these data sets include a set of prelabeled data and a disjoint set of unlabeled data.

iWildCam

Heat/motion-activated cameras placed in various locations in the wild are used to take burst photographs of different animals. Between cameras, there are massive shifts in illumination, color, angle, background, vegetation, and exact species/frequencies of animals present which causes poor generalization in models trained on existing cameras when used on new cameras.

Each data point in the iWildCam data set is composed of a photo taken by a camera, the label of the animal/lack of animal out of 182 possible options, and a domain value specifying which camera it is from. In total, there are $203,029$ images from 323 cameras. About 35% of images are of empty space with the next top two classes being 8% and 6%. In addition to the large class imbalance, some of the images may be noisy because ecologists may label bursts of images as a single animal so empty space may be falsely labeled. The number of images in each domain is highly variable where some domains have more than a thousand images but others may have less than 50.

Camelyon-17

Tissue slides collected from various hospitals are examined under a microscope to determine if they are tumorous or not. Between hospitals, there are massive differences in data collection, slide staining, patient demographics, and image quality which causes models trained on a certain set of hospitals to generalize poorly to new hospitals.

Each data point in the Camelyon-17 data set is composed of a 96×96 photo, a binary label of whether the central 32×32 area contains tumor tissue, and domain information specifying which hospitals each slide is from. In total, there are $450,000$ patches from 5 hospitals and each patch is marked by a pathologist whether it contains tumor cells. Most importantly, there are an equal number of positive and negative

cases of tumor tissue within both the training and testing data sets.

1.1.2 Breeds

BREEDS is a suite of subpopulation shift benchmarks created by leveraging the class structure of ImageNet [18]. For a specific benchmark, a class hierarchy is designed that groups semantically similar classes into superclasses. Subpopulation shifts can be generated by altering which subpopulation is present in the training and testing sets. For most benchmarks, the BREEDS subpopulation shift differs from that of WILDS. While subpopulation shifts in WILDS divide a certain class into unequal proportions, the shift in BREEDS divides the superclass so each ImageNet class part of the subclass remains undivided. This difference is important in transforming this data set into a more domain generalization problem.

Specifically, this paper will primarily work with the `ENTITY-13` data set. This data set has a total of 13 superclasses: garment, bird, reptile, arthropod, mammal, craft, equipment, furniture, instrument, man-made structure, wheeled vehicle, and produce. There is a total of 20 subclasses which are further divided in half for the training and testing set.

1.1.3 CIFAR100

CIFAR-10/100 [9] is a labeled subset of the tiny images data set. CIFAR-10 has a total of 10 classes with no distinction of smaller class within each class and CIFAR-100 has a total of 100 classes that is grouped into 20 superclasses. The superclass/class structure of CIFAR-100 is similar to `ENTITY-13` which makes it much more easily adaptable to domain generalization as compared to CIFAR-10.

This paper will focus on CIFAR-100 with 20 superclasses: aquatic mammals, fish, flowers, food containers, fruit and vegetables, household electronic devices, household furniture, insects, large carnivores, large man-made outdoor things, large natural outdoor scenes, large omnivores and herbivores, medium-sized mammals, non-insect invertebrates, people, reptiles, small mammals, trees, and two types of vehicles. Each

super class has 5 distinct subclasses of 600 images each; these images are divided by a 5/1 split into the training and testing sets.

1.1.4 Clustering

Clustering is a type of unsupervised learning method that discovers correlations and hidden patterns within unlabeled data. This technique is often used when the data has some inherent underlying geometry or groupings. Specifically, this work will focus on K-Means clustering.

K-Means clustering [10] relies on an iterative process to group the data into k distinct clusters, and converges to the state where the summed square distance between all points and its nearest cluster center (inertia) is minimized. The algorithm begins by specifying the number of clusters k required and the data set to be clustered X. k points are randomly initialized to be the cluster centers M. All the points are then clustered based on their nearest cluster center $\mu_i \in M$. A new μ_i is calculated for each cluster and this process repeats until the inertia $\mathrm{E} = \sum_{i=1}^{k} \sum_{x \in C_i} |x - x_i|^2$ [12] of the data set is minimized.

1.1.5 Optimal Transport

The optimal transport problem is finding the optimal path of moving one distribution of mass into another distribution [22]; the movement of each "unit" of the distribution between two locations is given a cost and the sum cost is minimized. Also known as the earth mover's problem, a realistic application is calculating the optimal method of moving a large mound of dirt to fill some hole given the various costs of transporting the dirt. Optimal transport can also be used to compare two probability distributions by calculating the minimum cost of transforming one distribution to the other through the Wasserstein Metric. An issue with OT is the dimensionality problem since the number of paths grows exponentially with the number of dimensions.

1.1.6 Related Work

Domain-adjusted Regression

[17] utilizes the DomainBed [4] data sets to demonstrate the potential of learning new predictors on existing features and introduces a new convex objective for learning a robust classifier. The proposed experiment is much simpler in nature where an ERM model is pretrained on some DomainBed data set and then it is allowed to "cheat" to retrain the prediction layer. Cheating is defined as retraining the model by also giving it access to OOD data points not in the test domain. There is $> 10\%$ improvement after cheating and cheating on the layer last outperforms retraining the entire model.

More importantly, the Domain-Adjusted Regression (DARE) objective is introduced as a convex objective with strong theoretical guarantees to learn a better linear featurizer on the frozen features. It is proven that DARE is the min-max-optimal predictor under the adversarial perturbation set. While typical training of a classifier attempts to encapsulate all OOD domains into one function, DARE makes domain-specific adjustments to predict under a canonical latent space instead.

Last Layer Re-Training is Sufficient for Robustness to Spurious Correlations

[7] attempts to resolve the performance gap between ID and OOD data by using a small *reweighting* data set where spurious correlations are already removed to retrain a new classifier using the old featurizer. The reweighting data set or deep feature reweighting (DFR) data set, is a reweighting of the features of the existing trained model and requires manual decisions to determine if spurious correlations are removed. The paper has produced good results when testing on the Waterbirds data set where groups of differing class distributions and sizes are created to generate a specific spurious correlation - the background.

Notably, this approach is similar to the hypothesis of this thesis but has a few key differences. First, the paper retrains the data using a subset of the training data set while we are using the OOD/wild data set. Second, the DFR data set requires

more manual intervention to remove spurious correlations and strongly maintain class balance. We test and analyze a more algorithmic approach with very minor manual interactions to label already sampled points while not strongly requiring class balance in the sample set. Lastly, we wish to minimize the number of sampled points since labeling wild data is expensive.

Typiclust

[6] attempts to build an algorithm that relies on typically - a value of how dense the region around that point is - rather than on probability to select a point in the sample set. The paper asserts that typical deep active learning algorithms perform well in high-budget settings but fail in low-budget settings; antithetically, random sampling would perform better at low-budget settings but fails in high-budget settings. The proposed algorithm, Typiclust, seeks to sample the most diverse set of typical data points in order to improve performance in low-budget settings by using self-supervised representation learning and calculating point density in the learned representation. Based on the empirical results of the paper, we see that Typiclust performs extremely well on CIFAR-10, CIFAR-100, and ImageNet-100 which indicates the potential of typicality/density as a metric of selection in low-budget regimes.

Chapter 2

General Methodology

2.1 ID Featurizer / OOD Classifier Validation

The experiment to verify the hypothesis that training on ID data yields a good featurizer and a poor classifier follows three stages. First, a model is pretrained until it has good training and validation performance within a subset of the total domains available S_{ID} and the complementary domains S_{OOD} are hidden from the model; due to the nature of our data set, the S_{OOD} domains serve as the wild domains. The featurizer of the trained model is separated from the classifier; there is evidence [7] that it is sufficient to only remove the last layer of the model. Next, the featurizer's weights are then frozen to prevent them from being updated while the classifier is retrained. The classifier is replaced with a linear layer/logistic regression and recombined with the now-frozen featurizer. Finally, the hidden domains S_{OOD} is split into a train and test set. The modified model is then trained on the S_{OOD} training subset in order to build a better classifier; since the featurizer is frozen, we are only training the last layer classifier. Both the modified model and the original model will be tested against the S_{OOD} testing subset and their performances compared.

There are two expected outcomes to this experiment. If the performance of the model trained on S_{OOD} is significantly greater than that of the original model, it is safe to conclude the ID featurizer is sufficiently robust and the ID classifier is poor compared to the OOD version. On the other hand, if the performance of the

modified model is not significantly greater than that of the original model, the results are inconclusive so we must perform further checks in order to try to differentiate between a bad ID featurizer and a poorly trained OOD classifier. One such way to do so is to perform a sweep over various generalization algorithms.

The hypothesis that only a small amount of data is needed to train a good classifier can also be verified through this experiment. If the performance of the OOD classifier is good for a small number of shots and does not increase significantly as the number of shots increases then it is safe to conclude only a small sample size is necessary. Another excellent result is if only a small number of shots are necessary to train the OOD.

2.2 Effective Sampling

While it is easy to generate ID/OOD data sets in the lab by partitioning labeled data by their domains, it is much harder to source properly labeled data from the wild. Compared to the difficulty of obtaining unlabeled wild data, each wild data point that is included in the wild OOD set is extremely expensive since it requires correct labeling by a third party. To avoid the high costs of training a new OOD classifier, this paper will design and demonstrate various sampling techniques that select the minimal subset of wild data that trains the OOD classifier with the greatest performance.

In order to reduce computation complexity and avoid extraneous information, all proposed sampling algorithms accept the featurized version of the wild data points produced by the frozen ID featurizer as input. This also allows certain sampling techniques to take advantage of geometric properties that lie within the feature space if such clusterings exist. Optimally, each sampling technique should select an equal number and most representative data points for each class in the wild data. Sampling techniques are compared by their respective performances over multiple models for the same data set.

In total, this paper will experiment with 7 different sampling algorithms: random

sampling Sec. 2.2.1, balanced sampling Sec. 2.2.2, K-Means sampling Sec 2.2.3, iterative K-Means sampling Sec. 2.2.4, weighted K-Means sampling Sec. 2.2.5, density-based K-means sampling Sec. 2.2.6, and Typiclust [6].

For the rest of this thesis, we define the terminology $N-$shot as a set with N labeled points for each class [24]. Let C be the number of classes present in the wild data. For subsequent algorithms, $\oplus$ denotes appending to a list.

2.2.1 Random Sampling

The random sampling algorithm (Alg. 1) is the simplest of the proposed sampling techniques and serves as a baseline of comparison against the other techniques. The random sampling algorithm randomly selects $N \times C$ samples from the wild data without any additional knowledge of the data, features, or class labels; these points are then labeled to be used in the training set. This method is heavily influenced by class imbalances that may exist within the wild data and do not consistently contain an equal number of points per class.

Algorithm 1 Random Sampling Algorithm

$X \leftarrow$ features
$N \leftarrow$ number of shots per class
$S \leftarrow [\,]$ ▷ Sampled Points
for $i \in [0, N)$ **do**
  $S \leftarrow S \oplus G_i[j]$ ▷ Random j
end for

2.2.2 Balanced Sampling

The balanced sampling algorithm (Alg. 2) is similar to the random sampling algorithm except it ensures there is an equal number of points per class by "cheating". "Cheating" [6] is defined as the foreknowledge of the labels of the wild data. In reality, it would be extremely costly and impossible for the algorithm to know these labels. To randomly select an equal number of points per class, the algorithm uses the label of each point in the data to group points with the same label and randomly select a

certain number of points per labeled group.

A slightly less costly implementation of this algorithm is to sample and label points until there is the required amount of points per class; extraneous points that were sampled for certain classes are discarded. While this does reduce the cost of the algorithm, a large class imbalance as seen in the WILDS data sets would result in the selection of nearly every point.

This algorithm will serve as an approximate upper bound for comparison against other sampling techniques. it is important to note that it is not the absolute upper point because randomly selected points could be outliers or may form spurious correlations.

Algorithm 2 Balanced Random Sampling Algorithm

$X \leftarrow$ features
$Y \leftarrow$ labels
$N \leftarrow$ number of shots per class
$G \leftarrow \{G_l = [\,] \; \forall \; l \in Y\}$ ▷ points per class
$S \leftarrow [\,]$ ▷ Sampled Points
for $c \in [0, |X|)$ **do** ▷ Builds sets of points with same label
  $G_{Y[c]} \leftarrow G_{Y[c]} \oplus X[c]$
end for
for $G_i \in G$ **do**
  for $n \in [0, N]$ **do** ▷ Grabs N random points from G_i
    $S \leftarrow S \oplus G_i[j]$ ▷ Random j
  end for
end for

2.2.3 K-Means Sampling

The K-Means sampling algorithm samples N shots using the K-Means clustering algorithm and selects a certain number of points per cluster to reach the desired amount of data points. With sufficiently good features, the feature's labels should correlate with clusters in the representation space since those features should be geometrically similar. By leveraging the clustering of similarly labeled points, the algorithm is able to build a close-to-balanced sample without relying on the exact labels of the points.

Specifically, two ways of clustering are analyzed:

1. Run K-Means with $N \times C$ clusters, select the point closest to each cluster center and query the labels for those points. This will be referred to this as Full K-Means (Alg. 3) for the rest of this paper since the number of clusters is equal to the full size of our sample.

2. Run K-Means with C clusters, select the closest N points per cluster to the cluster center and query the labels for those points. This shall be referred to as Class K-Means (Alg. 4) for the rest of the paper since the number of clusters is equal to the number of classes.

It is expected that Full K-means is more affected by class imbalances than Class K-Means since more points in a class mean increased opportunities for clusters to form. However, Full K-means is better at capturing the spread of features in a class since it samples from multiple clusters for a certain class.

Algorithm 3 Full K-Means

$X \leftarrow$ data points
$N \leftarrow$ number of shots per class
$C \leftarrow$ number of classes
$S \leftarrow [\,]$
$M_{Full} = \{\mu_i\} \leftarrow \texttt{K-Means}(X, N \times C)$
for $\mu_i \in M_{Full}$ **do** ▷ $\oplus$ is appending to array, d is just Euclidean distance
  $S \leftarrow S \oplus arg\,min_{x_j \in X} d(x_j, \mu_i)$ ▷ Full K-Means
end for

Algorithm 4 Class K-Means

$X \leftarrow$ data points
$N \leftarrow$ number of shots per class
$C \leftarrow$ number of classes
$S \leftarrow [\,]$
$M_{Class} = \{\mu_i\} \leftarrow \texttt{K-Means}(X, C)$
for $\mu_i \in M_{Class}$ **do** ▷ $\oplus$ is appending to array, d is just Euclidean distance
  $S \leftarrow S \oplus arg\,min_{X' \in X, |X|=N} \sum_{x_j \in X'} d(x_j, \mu_i)$ ▷ Class K-Means
end for

2.2.4 Iterative K-Means Sampling

The iterative K-Means sampling algorithm (Alg. 5) uses a modified K-Means algorithm that iteratively selected the farthest (most dissimilar) point from the set of points already selected in order to solve the issues of Full/Class K-Means. Specifically, the K-Means clustering algorithm is run using a value $K >> N \times C$ to reduce the number of points the iterative process has to check and to reduce noise in the data; for larger N it is fine to only satisfy $K > N \times C$ since we want to reduce computation size. Next, K-Means is run with C clusters on the K cluster centers of the first K-Means; the C points closest to the second K-Mean's cluster centers will serve as the starting list of sampled points S. Until S has reached the same size as $N-$shot, the point closest to the cluster center with the greatest geometric distance sum to the points in S is selected. The geometric distance sum to the points in S for point p is the sum of the distances from p to $s_i \forall s_i \in S$.

This algorithm is an improvement over Full K-Means since selecting the point with the great geometric distance prevents class imbalances from causing too many points with the same label to be selected since similarly labeled points would be close together. In addition, it is an improvement over Class K-Means since the initial large-size K-Means cluster would enforce the diversity of features sampled and lessen the impact of outliers on the outcome.

Algorithm 5 Iterative K-Means

$X \leftarrow$ datapoints
$N \leftarrow$ number of shots per class
$C \leftarrow$ number of classes
$S \leftarrow [\,]$
$M_K = \{\mu_k\} \leftarrow \texttt{K-Means}(X, K)$ ▷ Where $K >> C$
$S \leftarrow S \oplus \texttt{K-Means}(M_K, C)$ ▷ $\oplus$ is appending to array
while $|S| < N \times C$ **do** ▷ d is just Euclidean distance
  $S \leftarrow S \oplus arg\,max_{\mu_k \in M_k} \sum_{\mu_i \in S} d(\mu_k, \mu_i)$
end while

2.2.5 Weighted K-Means Sampling

The weighted iterative K-Means algorithm (Alg. 6), shortened to weighted K-Means, builds upon the iterative K-Means algorithm through the use of class weights when calculating geometric distance for a new point to be sampled. The algorithm follows the iterative version exactly except when calculating the geometric distance sum, each distance to a point in S is weighted by how often that point's label appears in S. This weight for a point $s_i \in S$ is more precisely $w_i = 1 - \frac{c_i}{|S|}$ where c_i is the count of s_i's label in S.

By weighting the geometric distance sum, the algorithm is able to better control how evenly spread the classes are in the sampled set. if a certain class dominates the sample during the iterative process, then that class's weight will be small which causes the algorithm to choose points farther away which should be of another class.

Algorithm 6 Weighted K-Means

$X \leftarrow$ data points
$N \leftarrow$ number of shots per class
$C \leftarrow$ number of classes
$S \leftarrow [\,]$
$M_K = \{\mu_k\} \leftarrow \texttt{K-Means}(X, K)$ ▷ Where $K >> C$
$S \leftarrow S \oplus \texttt{K-Means}(M_K, C)$ ▷ $\oplus$ is appending to array
while $|S| < N \times C$ **do** ▷ d is just Euclidean distance
  $S \leftarrow S \oplus arg\,max_{\mu_k \in M_k} \sum_{\mu_i \in S} w_i * d(\mu_k, \mu_i)$
end while

2.2.6 Density-based K-Means Sampling

The density-based K-Means sampling algorithm (Alg. 7), shortened to dense K-Means, builds upon weighted K-Means by leveraging the density [6] of the clusters in order to select the optimal sample set. Instead of selecting the point closest to the cluster center after performing the K-Means clustering algorithm, the point that has the highest density within the cluster is selected. The density of a point is valued by the inverse of the average Euclidean distance to its K nearest neighbors.

The point with the highest density may be a better representative of a certain

label compared to the mean of that label's cluster since it is the point most similar to points of the same label. High-density points ensure geometric distance sums are maximized for the majority of points in a cluster. This maximizes the representation power of a single selected point since it maximizes the number of points the iterative process will skip over. In addition, there is strong evidence in algorithms such as Typiclust that a density-based approach has value.

Algorithm 7 Density-based K-Means

$X \leftarrow$ data points
$N \leftarrow$ number of shots per class
$C \leftarrow$ number of classes
$S \leftarrow [\,]$
$M_K = \{\mu_k\} \leftarrow \texttt{K-Means}(X, K)$ ▷ Where $K >> C$
$S \leftarrow S \oplus \texttt{K-Means}(M_K, C)$ ▷ $\oplus$ is appending to array
while $|S| < N \times C$ **do** ▷ d is just Euclidean distance
  $S \leftarrow S \oplus arg\,max_{\mu_k \in M_k} \sum_{\mu_i \in S} w_i * d(\mu_k, \mu_i) - \delta_j$ ▷ $w_i = 1$ if not weighted
end while

where δ_j is the inverse of the summed distance between a point μ_k to its $K = 20$ nearest neighbors: $(\sum_{p_i \in NN}^{K} d(\mu_k, p_i))^{-1}$.

2.3 Altered Generalization Algorithm

Generalization algorithms that use domain alignment attempt to solve a problem that is similar to the optimal transport problem: a source distribution (predicted labels with their respective probabilities) needs to be transformed into a target distribution (the true labels). This parallel provides inspiration to explore the potential of optimal transport-based models as a form of generalization algorithms. Specifically, the Wasserstein metric is used due to ease of implementation and effectiveness.

Specifically, the Wasserstein generalization algorithm modifies the deepCORAL algorithm [21] by redefining the CORAL loss from the distance between the second-order statistics of the source and target features to the Wasserstein loss between the source and target features. By using Wasserstein loss, the new algorithm is able to leverage optimal transport techniques to find the most optimal path of transforming

the source(trained) features to the target(oracle) features.

2.4 Data sets

There are a total of 4 data sets that span both domain generalization and subpopulation shift that will be used to test the combination of ID featurizer/OOD classifier, the effectiveness of the sampling algorithms Sec. 2.2 and the performance of the Wasserstein generalization algorithm Sec. 2.3. Specific design choices and experiments will be described in the following sections. The performance will be compared using three different metrics: pure accuracy (percentage of correct), balanced accuracy (average of label accuracy of all classes), and F1 metric (a metric used by the WILDS benchmark). For most data sets, the balanced accuracy is lower than pure accuracy but follow similar trends in performance so we will primarily use pure accuracy for analysis unless there is unique features in the balanced accuracy.

2.4.1 Domain Generalization

The domain generalization data sets are easy to adapt to the problem design since the training and testing datasets are already split into disjoint domains. Both data sets come from the WILDS benchmark which has an existing Python package built for training/testing and includes a hyper-parameter sweep for various generalization algorithms. To test the variety of techniques and experiments proposed previously, this paper will select the highest-performing model for each generalization algorithm based on the sweep, remove the classifier layer, freeze the remaining part of the model, attach a new classifier layer in the form of logistic regression and train the new combined model. It is uncertain that the hyper-parameters of the original model are also the optimal set for the modified model so a new hyper-parameter sweep must be executed.

iWildCam

Since the domains of iWildCam are associated with cameras in the wild, the images are very noisy and some domains do not have a significant amount of non-empty images. Domains also do not include the full gamut of potential animals; some domains may include tens of animals while others may only include one. When domains with insufficient non-empty images are sampled from, problems such as an insufficient number of points to properly cluster labels can arise. To alleviate these issues, each domain is preprocessed such that only domains that have more than a predetermined number of points are used when testing. While preprocessing does not ensure a minimum amount of points per label, even animals with proportionally small representation should be able to be sampled given a sufficiently large data set.

Camelyon17

No special action was taken to enable the experiments for Camelyon17 due to the data set only having 1 domain, no perceivable noise and well-organized images.

2.4.2 Hierarchical Subpopulation Shift

Compared to the domain generalization data sets, the subpopulation shift data sets require more setup to transform the hierarchical class structure into a domain generalization problem. Starting from the root of the class structure, each superclass becomes the class in the domain generalization analog. Next, 1 subclass from each superclass is selected, grouped into a domain, and given a random label. The grouping process is then repeated, with no subclass selected twice, until every subclass is within a domain. This way, synthetic domains can be generated and an experiment identical to that of the domain generalization data sets can be performed. A major difference is that there are no pretrained models to select from so a hyper-parameter sweep must be performed for each data set.

Superclass	Domain 1	Domain 2	Domain 3	Domain 4	Domain 5	Domain 6	Domain 7	Domain 8	Domain 9	Domain 10
garment	lab coat	brassiere	hoopskirt	cardigan	pajama	academic gown	apron	diaper	sweatshirt	sarong
bird	prairie chicken	red-breasted merganser	albatross	water ouzel	goose	oystercatcher	American egret	hen	lorikeet	ruffed grouse
reptile	sidewinder	leatherback turtle	boa constrictor	garter snake	terrapin	box turtle	ringneck snake	rock python	American chameleon	green lizard
arthopod	cabbage butterfly	admiral	lacewing	trilobite	sulphur butterfly	cicada	garden spider	leaf beetle	long-horned beetle	fly
mammal	English setter	llama	lesser panda	armadillo	indri	giant schnauzer	pug	Doberman	American Staffordshire terrier	beagle
accessory	hair slide	umbrella	pickelhaube	mitten	sombrero	shower cap	sock	running shoe	mortarboard	handkerchief
craft	schooner	gondola	canoe	wreck	warplane	balloon	submarine	pirate	lifeboat	airship
equipment	cassette player	snorkel	horizontal bar	soccer ball	racket	baseball	joystick	microphone	tape player	reflex camera
furniture	studio couch	throne	crib	rocking chair	dining table	park bench	chest	window screen	medicine chest	barber chair
instrument	maraca	saltshaker	magnetic compass	accordion	digital clock	screw	can opener	odometer	organ	screwdriver
man-made structure	suspension bridge	worm fence	turnstile	tile roof	beacon	street sign	maze	chainlink fence	bakery	drilling platform
wheeled vehicle	jinrikisha	golfcart	tow truck	ambulance	bullet train	fire engine	horse cart	streetcar	tank	Model T
produce	pomegranate	mushroom	strawberry	lemon	head cabbage	Granny Smith	hip	ear	banana	artichoke

Table 2.1: Class Domain Distribution for `ENTITY-13`

Breeds

Out of the 13 superclasses of `ENTITY-13`, the domains are trivially divided into domains by the index a certain class appears in the class structure as seen in Tab. 2.1. The source data set is left unaltered and used to train the featurizer through a hyperparameter sweep. Although the classes in the test never appear in the training data set, the superclasses are set up with a semantic/visual connection between subclasses so there should be the same underlying set of features for all the domains. Further work is planned to test the impact of random groupings of classes into domains.

CIFAR-100

Compared to the single class domain divisions of `ENTITY-13`, the 20 superclasses of CIFAR-100 are divided such that there is one class per domain; again this division is done trivially by the index of a class in the class structure as seen in Tab. 2.2 where the first three classes are the training set and the last two are the testing domains. Compared to the Breeds data, fewer classes were used to train the generalization model. The train data is again left unaltered from the original data set and a hyperparameter sweep was completed to obtain the best models for each algorithm. One interesting aspect to note is the fact that some class-to-superclass groupings are not exact. Although mushrooms are neither fruits nor vegetables and bears are not purely carnivores, those classes are the closest visual match.

Superclass	Training	Domain 3	Domain 4
aquatic mammals	beaver, dolphin , otter	seal	whale
fish	aquarium fish, flatfish , ray	shark	trout
flowers	orchids, poppies, roses	sunflowers	tulips
food containers	bottles, bowls, cans	cups	plates
fruit and vegetables	apples, mushrooms, oranges	pears	sweet peppers
household electrical devices	clock, computer keyboard, lamp	telephone	television
household furniture	bed, chair, couch	table	wardrobe
insects	bee, beetle, butterfly	caterpillar	cockroach
large carnivores	bear, leopard, lion	tiger	wolf
large man-made outdoor things	bridge, castle, house	road	skyscraper
large natural outdoor scenes	cloud, forest, mountain	plain	sea
large omnivores and herbivores	camel, cattle, chimpanzee	elephant	kangaroo
medium-sized mammals	fox, porcupine, possum	raccoon	skunk
non-insect invertebrates	crab, lobster, snail	spider	worm
people	baby, boy, girl	man	woman
reptiles	crocodile, dinosaur, lizard	snake	turtle
small mammals	hamster, mouse, rabbit	shrew	squirrel
trees	maple, oak, palm	pine	willow
vehicles 1	bicycle, bus, motorcycle	pickup truck	train
vehicles 2	lawn-mower, rocket, streetcar	tank	tractor

Table 2.2: Class Domain Distribution for CIFAR-100

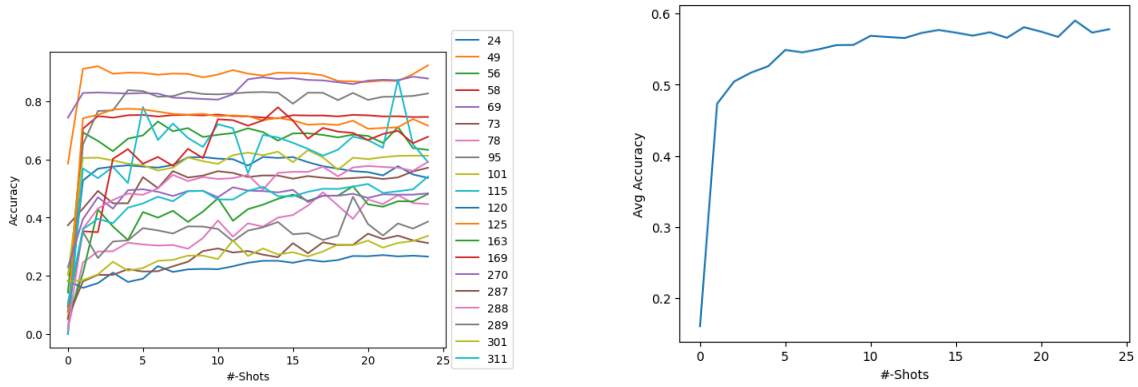
Chapter 3

Results and Analysis

3.1 ID Featurizer with OOD Classifier

On average, the combination of an ID featurizer/OOD classifier performs significantly better than a model completely trained on ID data on the iWildCam, Breeds, and CIFAR-100 data sets. This difference is not as significant on the Camelyon17 data set and this will be further discussed in Section 3.3. Since the improvement is identical across the data sets, models, and sampling algorithms, this paper will focus on analyzing the performance of the OOD classifier on the iWildCam data set with the baseline ERM model and class K-Means sampling. In Fig. 3-1, the y-axis is the accuracy of the model and the x-axis is the number of shots sampled where 0-shots is the accuracy of the ID classifier.

Fig. 3-1a plots the accuracy of 20 different domains over the range of 1-shot to 24-shots. We see a consistent increase across the domains in the performance from the original classifier to the OOD classifier even at 1-shot. The largest improvement in accuracy at 1-shot is seen in the lower orange domain with an original accuracy of 0.000% rounded to an accuracy of 0.706%. While some domains may have more variable performance at higher shots, the average over all the domains in Fig. 3-1b shows that the accuracy plateaus at less than 10 shots. This supports the theory that a very little number of shots/samples is necessary to train a good classifier. Overall, the significant improvement in performance validates the theory that training an OOD



(a) Various Pruned Domains

(b) Average of Pruned Domains

Figure 3-1: ERM, Class Sampling on iWildCam

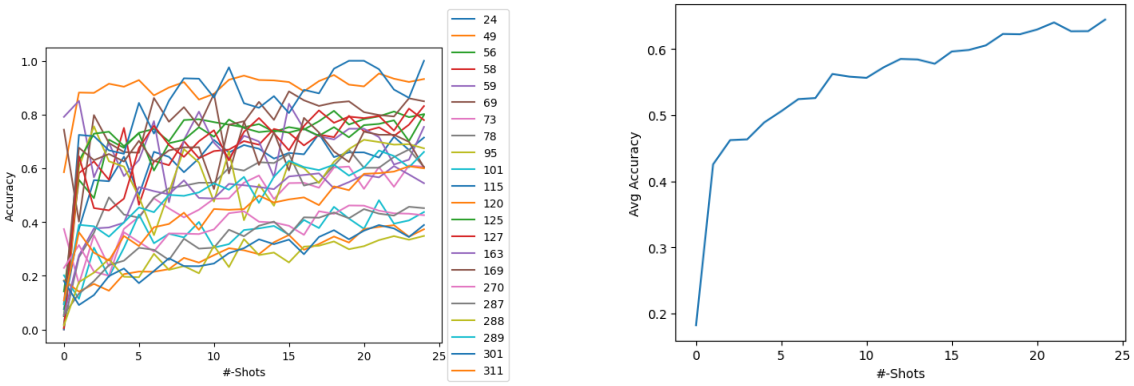
classifier using the ID featurizer performs better than the original model combination.

3.2 iWildCam

The iWildCam data set tests 6 different sampling algorithms - balanced, class K-Means, full K-Means, iterative K-Means, weighted K-Means, and Typiclust - on 7 models trained using different generalization algorithms - ERM, deepCORAL, Wasserstein, PseudoLabel, DANN, AFN, and FixMatch. Domains are preprocessed such that there are at least 50 valid/ data points in each domain. Each combination of model and sampling algorithm is evaluated over multiple metrics such as normal accuracy, balanced accuracy, and the F1 macro.

ERM with Balanced Sampling

Fig. 3-2 shows the performances of various domains using the ERM model with the balanced sampling algorithm on the iWildCam data set. At first glance, the performances in Fig. 3-2a are extremely noisy but there is a general upward trend as the number of shots increases. This trend is better shown in the averaged performance plot Fig. 3-2b where there is a significant jump from the original performance to 1-shot and the increase plateaus as the number of samples increases. It is also important to note that for some domains at low shots, the performance is worse than the



(a) Various Pruned Domains

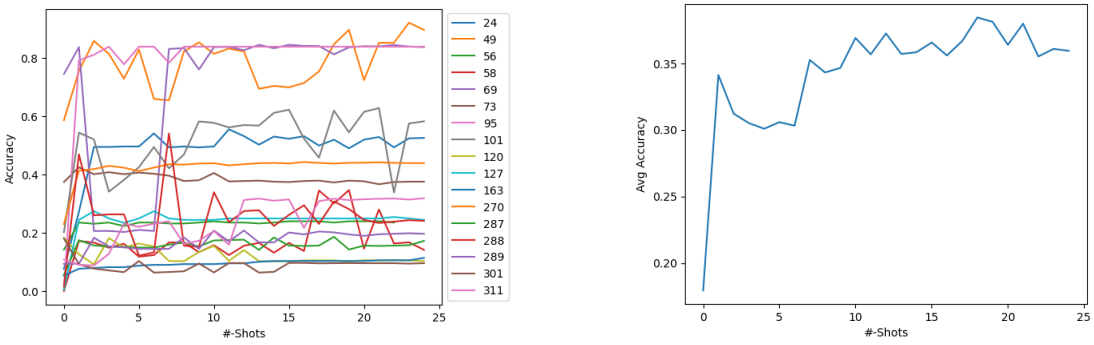
(b) Average of Pruned Domains

Figure 3-2: ERM, Balanced Sampling on iWildCam

original classifier. We believe this deviation from the average and the messy performance across domains can be attributed to noise in the data. Within the iWildCam domains, images are sometimes labeled improperly such as empty images that were taken in bursts with images with actual animals in them will be labeled as such animals. Because the balanced sampling algorithm does not take into account how representative a data point is for a class, if there is a lot of noise within the data then the random sampling within a class would yield poor training data. As the number of points per class sampled increases, the chance of selecting good points would increase and thus yield better performance in the long run.

ERM with Class K-Means Sampling

A better-performing sampling algorithm on the data set with the ERM model is class K-Means sampling. Due to the clustering technique, the class K-Means sampling algorithm is able to sample representative points without being strongly influenced by the noise in the domains' data. Fig. 3-1a displays a more consistent increase in performance as the number of shots increases. In addition, general performance on all the domains when compared with the balanced sampling algorithm by at least 5% on average for lower shot counts. While the average performance in Fig. 3-1b plateaus much faster, balanced sampling performs better at higher shot counts. This is in line with the original hypothesis since balanced sampling is "cheating" but due to the



(a) Various Pruned Domains

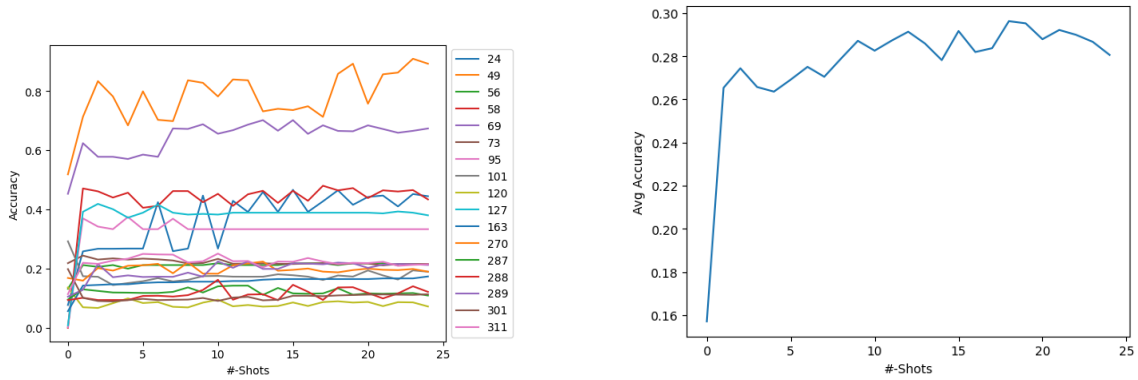
(b) Average of Pruned Domains

Figure 3-3: ERM, Full Sampling on iWildCam

large amount of noise in iWildCam data, performance at lower shot counts presents worse than the performance of the same model using the class K-Means sampling.

ERM with Full K-Means Sampling

On the other hand, the Full K-Means sampling algorithm also utilizes a clustering algorithm but performs worse when compared to both class K-Means sampling and balanced sampling. Aside from a select subset of domains in Fig. 3-3a, most domains have significantly lower performance and high variability over the number of shots. Domain 69 has an excellent original and 1-shot accuracy but the performance drops significantly. The average performance of full K-Means sampling in Fig. 3-3b is more than 20% lower. Like balanced sampling, full K-Means sampling is greatly affected by the noise and label distribution within each domain data. Because the algorithm generates more than 1 cluster per label, noise within a group of points of the same label would be clustered and favored to be sampled. Similarly, labels with more data points will generate more clusters and be favored. For a low number of shots, there is a higher chance of clustering around the noise and high label counts. Unlike balanced sampling, this algorithm does not perform better at higher shot counts. While balanced sampling benefits from having an increased chance of randomly not sampling noise, increased shot count in the full K-Means algorithm would simply divide the existing noise clusters into more data points.



(a) Various Pruned Domains

(b) Average of Pruned Domains

Figure 3-4: ERM, Full Sampling on iWildCam Balanced Accuracy

Switching over to the balanced accuracy, an interesting pattern appears compared to the normal accuracy plot. The performance drop in Domain 69 is not as apparent in balanced accuracy but still exists. This change indicates that the issue with noise in data still exists but a majority of the original decrease in accuracy might be attributed to an imbalance in the sampled set. Looking closer at the data of Domain 69, the proportion of empty space to labeled animals is significantly larger than in other domains. While the OOD classifier is trained properly, the large proportion of empty space would cause the overall accuracy to drop but the balanced accuracy which accounts for the distribution would remain higher.

General Analysis

Overall, there is a significant performance improvement from the ID classifier to the OOD classifier over all the models seen in Fig. 3-5a and over all the various sampling algorithms seen in Fig. 3-5b. Specifically, there are a few similarities between the performances of all the models: there is a sharp jump at the start, a slow plateau at higher shot counts, and there is very little noise over the average. The shape of the plots of average model performance strongly supports the hypothesis that the OOD classifier performs better than the ID classifier and that only a small number of shots is necessary to achieve such performance. An interesting aspect of the plot to notice is that the performances are split into two major groups: the deepCORAL models

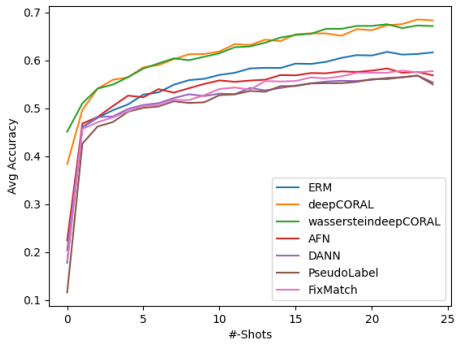
and the others.

We see that the models deepCORAL and Wasserstein deepCORAL (Sec. 2.3) performs significantly better than the other models from 0-shots to 24-shots. We see that Wasserstein deepCORAL has better original performance but plateaus to the same performance as deepCORAL nearly immediately. Since both deepCORAL-related models perform better, there is reason to suspect that there is some aspect of the model design that pairs well with the iWildCam data such as aligning the source and target distributions under a certain metric. The Wasserstein model performs better than every other model on the WILDS benchmark leaderboard under submission conditions. More analysis will be done using the other data sets in later sections to determine if this is unique to iWildCam or ubiquitous to all the image data sets.

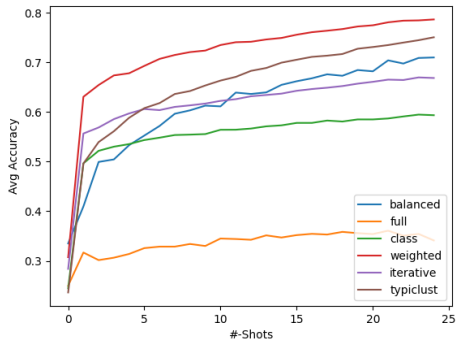
A similar plot to the average performance over models can be seen for various sampling algorithms in Fig. 3-5b. There is an identical jump in performance from the original accuracy to the 1-shot accuracy and a slow steady plateau as the number of shots increases. These results continue to affirm the hypothesis that the OOD classifier performs better than the ID classifier at a low number of shots.

While there is no grouping of certain sampling algorithms, the weighted K-means sampling algorithm performs significantly better than the other algorithms and the full K-Means algorithm performs significantly worse. In fact, there seems to be a distinct ordering in how well each sampling algorithm performs. Initially, it is surprising that both the Typiclust sampling algorithm and the weighted K-Means perform better than the balanced sampling algorithm. But as mentioned in the previous discussion in Sec. 3.2 and Sec 3.2, these deviations from expectation can be attributed to the noise in the domain data.

Ignoring the sampling algorithms impacted by noise, the class K-Means algorithm performs the worst. The iterative K-Means algorithm builds upon the class K-Means algorithm so the increase in performance indicates the hypothesis that iterative selecting clustered points better avoids outliers and selects a more diverse and representative set of points. The subsequent improvement seen in weighted K-Means over iterative K-Means further justifies the importance of a diverse sample and affirms the prop-



(a) Average Performance Over Generalization Models



(b) Average Performance Over Sampling Algorithms

Figure 3-5: iWildCam Performance Comparison

erty of weighted K-Means to spread the distribution of classes better. In addition, the smoothness of the average accuracy gives support to the ability of the weighted K-means algorithm being able to remove noise from the sample. The joint increased performance of the iterative and weighted versions of the algorithm illustrates the fact that iWildCam features are clustered in multiple smaller groups rather than 1 large cluster; a large 1 cluster point group would primarily sample outliers but multiple disjoint clusters would yield good representative points for each cluster. Finally, the Typiclust algorithm performs better than all the algorithms aside from the weighted K-Means algorithm. The algorithm design is similar to that of the iterative K-Means sampling algorithm which explains the increased performance compared to the iterative K-means sampling algorithm and provides incentive in exploring the effect of a density-based metric when sampling points.

3.3 Camelyon17

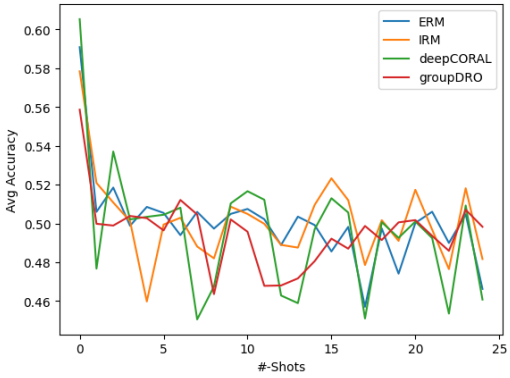
The Camelyon17 data set tests 5 different sampling algorithms - balanced, class K-Means, full K-Means, iterative K-Means, and weighted K-Means - on 4 models trained using different generalization algorithms - ERM, deepCORAL, IRM, and groupDRO. Since there were only data from 5 hospitals, only 1 domain is used in the wild/test set. Each model and sampling algorithm combination is evaluated over the same metrics

as iWildCam such as normal accuracy, balanced accuracy, and the F1 macro.

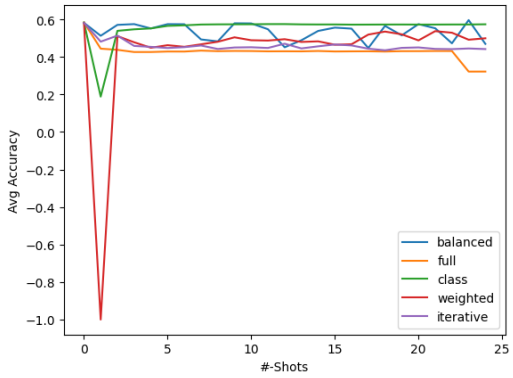
General Analysis

Unlike the significant improvement in performance seen in iWildCam, the OOD classifier does not perform better than the original classifier. In Fig. 3-6a, we can see that all the models perform worse than the original accuracy by around 10%. While the plot is noisy, the accuracy decrease is immediate and plateaus with no change at higher shot counts. Switching to the average performances oversampling algorithms in Fig. 3-6b, we see that aside from the class K-Means sampling algorithm, all the other algorithms perform around 10% worse than the original accuracy; there is no need to compare balanced accuracy since the classes are already balanced in the data set. A very interesting trend that occurred is the drop in performance in all the sampling algorithms at 1-shot; the weighted sampling algorithm accuracy is -1.0 because the OOD classifier was not able to separate the classes. Since the full K-Means and Class K-Means algorithm at 1-shot performs the clustering exactly the same, we can assume that the difference in performance is due to the poorly representative points being sampled; this would also explain why the Class K-Means algorithm recovers in performance while the Full K-Means algorithm remains at low performance.

The performance difference between Class K-Means and iterative/weighted K-Means is indicative of the natural clustering of features in the feature space. Both the iterative and weighted K-Means algorithm seeks the farthest points from the already sampled points because both algorithms assume there are multiple classes with distinct clusters and the farthest point from the sampled classes would discover points in different classes/clusters. If the Camelyon17 data points are closely centered around one cluster in the feature space then the iterative/weighted K-Means algorithms would only serve to select non-representative outlier points. On the other hand, the Class K-Means algorithm would successfully cluster both labels and sample the most representative points of each class.



(a) Average Performance Over Generalization Models



(b) Average Performance Over Sampling Algorithms

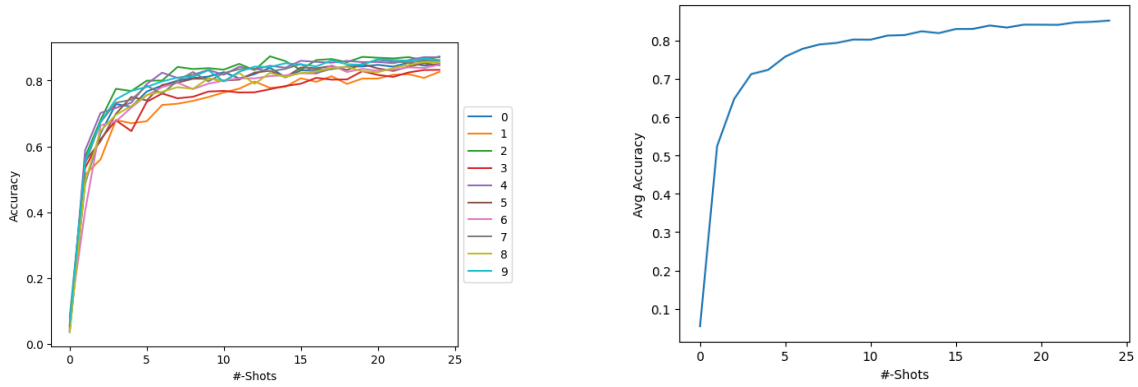
Figure 3-6: Camelyon17 Performance Comparison

3.4 Breeds

The Breeds data set tests 8 different sampling algorithms - balanced, class K-Means, full K-Means, iterative K-Means, weighted K-Means, pure dense K-Means, weighted dense K-Means, and Typiclust - on 4 models trained using different generalization algorithms - ERM, deepCORAL, wassersteindeepCORAL, and DANN. The Dense K-Means algorithm is split into weighted and pure depending on whether additional class weights were used. The ERM, deepCORAL, and DANN models were selected as the best cover over a spread of different types of generalization algorithms while the Wasserstein algorithm performed the best on iWildCam. Each model and sampling algorithm combination is evaluated over the same metrics as before: normal accuracy, balanced accuracy, and F1 macro.

ERM with Balanced Sampling

Fig. 3-7a shows the performances of the ERM with balanced sampling over the 10 domains generated from Breeds. We see that the OOD classifier performs exceptionally better than the original classifier on all of the domains with very little variance between the accuracy on each domain. Averaging the performances over all the domains in Fig. 3-7b, there is a large initial jump at 1-shot and plateaus in a way that is similar in shape to a logarithmic function. The balanced sampling algorithm



(a) Various Pruned Domains

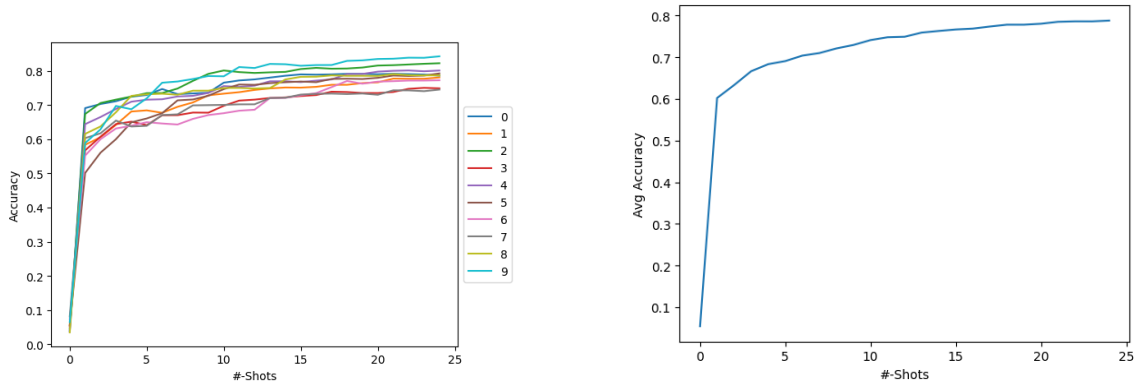
(b) Average of Pruned Domains

Figure 3-7: ERM, Balanced Sampling on Breeds

performs more stable on Breeds compared to iWildCam due to the lack of noise in the domains; this stability allows us to capture a better bound of an optimal sampling algorithm since the balanced sampling algorithm is now able to successfully "cheat".

ERM with Class K-Means Sampling

The ERM model with the Class K-Means sampling algorithm performs worse than the same model with the balanced sampling algorithm because it does not "cheat" but still serves as a significant improvement over the original classifier. Similar to the balanced sampling results, the performances over the 10 domains in Fig. 3-8a do not have significant deviations from the average. In Fig. 3-8b, the 1-shot performance is higher than the 1-shot performance of the balanced sampled algorithm, performance at higher shot counts is lower than balanced sampling performance, and the accuracy plateaus much slower than balanced sampling at higher shot counts. Better performance at low shot counts is caused by the inability of random point selection in the balanced sampling algorithm to sample as representative of a point as the class K-Means cluster algorithm. When the number of shots is increased, the class K-Means sampling algorithm does not necessarily pick better representative points but the balanced K-Means algorithm would be able to randomly sample better points as the number of samples increases.



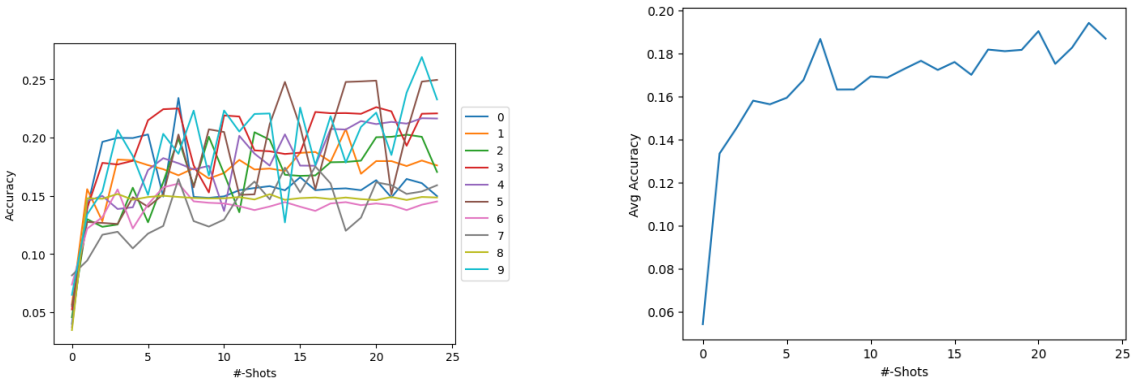
(a) Various Pruned Domains

(b) Average of Pruned Domains

Figure 3-8: ERM, Class Sampling on Breeds

ERM with Full K-Means Sampling

Unsurprisingly, the ERM model with the Full K-Means sampling algorithm performs significantly worse than every other sampling algorithm but still demonstrates an improvement over the original classifier. In addition, there is a high variance in the performances of each domain at any number of shots seen in Fig. 3-9a. This large variability is further demonstrated in Fig. 3-9b where the average performance is coarse but still plateaus at a high shot count. Unlike the iWildCam performance, the Breeds data set does not include noisy image labels so the difference in performance is either caused by poor feature geometry or the inability of the algorithm to capture representative points. The other K-Means-based algorithms perform very well on the same features which indicates that the feature space geometry is well clustered. The design choice of clustering around the total number of expected points rather than the number of labels is a source of poor accuracy. The Full K-means algorithms generally expect the labels to be subdivided into distinct characteristic feature clusters so the non-class-based clustering would be able to distinguish each sub-cluster. If the features are largely clustered as in Breeds domains, then the sub-clusters would unevenly subdivide within existing label clusters and generate a bad spread of samples.



(a) Various Pruned Domains

(b) Average of Pruned Domains

Figure 3-9: ERM, Full Sampling on Breeds

General Analysis

Overall the performance of the 4 models on the Breeds domains is exceptionally well compared to the original accuracy seen in Fig. 3-10a. The models can be divided into two major groups: models that performed well originally and models that did not perform well originally. Both the ERM and DANN models did not have above 50% accuracy on average, but the models demonstrate a large sharp jump at 1-shot accuracy which slowly plateaus at higher shot counts; this indicates the models did not train a well-generalized ID-classifier but the additional OOD training has significant improvement. On the other hand, both CORAL-based models have great original accuracies but show a small drop in performance at 1-shot which slowly plateaus to a higher value at higher shot counts. The initial drop in performance at 1-shot is caused by the inability of a better classifier to be trained using a small sample size. As the number of shots increases, the accuracy of the models rapidly increases until it plateaus at a value higher than the original because the sample size is large enough for good training.

Another interesting grouping is the split between ERM and the other models. Although all the models showed improvement, the CORAL models and DANN have comparable results while the ERM model does more than 10% worse. We believe this split is caused by ERM's lack of consideration of domain alignment which causes it to train a less generalized classifier than the other models. The DANN featurizer is

not as well generalized as the CORAL models as seen in the lack of decrease from the original accuracy to 1-shot accuracy but the featurizer is generalized enough to train good OOD classifiers.

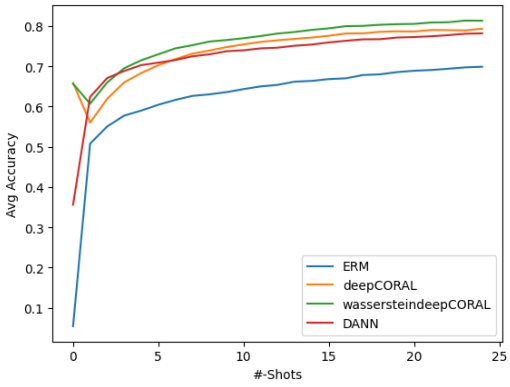
We will focus on the non-full K-Means sampling algorithms because they produce unexpected results. Based on the iWildCam results, it would be expected that the iterative, weighted, and density-based K-Means algorithms would perform better than the class K-Means algorithm because they augment the latter algorithm. However, the class K-Means sampling algorithm performs better than the augmented algorithms which all have similar performance. We believe that this is caused by the innate geometry of the Breeds features which is more centrally clustered than the iWildCam features. For a centrally clustered point group, the class K-Means algorithm would primarily select points near the center of the group which is highly representative but the iterative algorithms would be forced to select more outlier (less representative) points due to the distance constraint. This is supported by the way the Breeds data set is generated which uses clean, stand, and carefully selected images to form domains in the data set while the iWildCam data is noisy, may be partially obscured, and naturally generated.

The Typiclust sampling algorithm performs equally as well as the balanced sampling algorithm which is significantly better than the other sampling algorithm's performances. The elevated accuracy indicates strong potential behind a density-based approach to sampling. While the density-based K-Means algorithms do not perform well, the algorithms are inhibited by the design of the iterative K-Means algorithm.

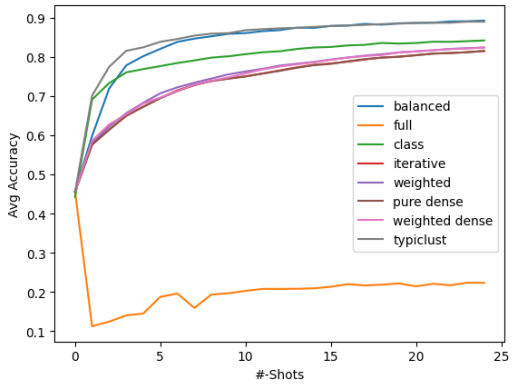
At low shot counts, most of the models and algorithms combinations perform significantly better than the original classifier's accuracy.

3.5 CIFAR-100

The CIFAR-100 data set tests 8 different sampling algorithms - balanced, class K-Means, full K-means, iterative K-Means, weighted K-Means, pure dense K-Means, weighted dense K-Means, and Typiclust - on 4 models trained using different general-



(a) Average Performance Over Generalization Models



(b) Average Performance Over Sampling Algorithms

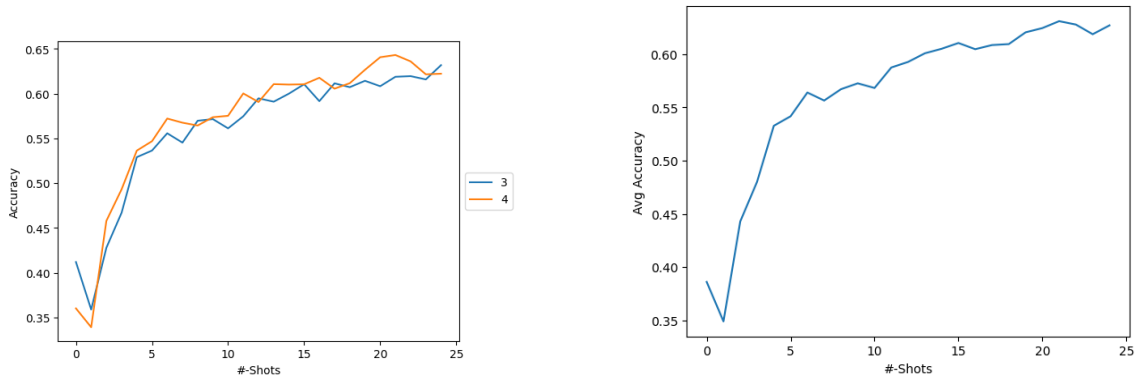
Figure 3-10: Breeds Performance Comparison

ization algorithms - ERM, deepCORAL, wassersteindeepCORAL, and DANN. Each model and sampling algorithm combination is evaluated over the same metrics as before: normal accuracy, balanced accuracy, and F1 macro.

CIFAR-100 is used to further verify the results of Breeds since it is also an artificially generated domain generalization data set with clean, properly justified images. We expect to see a comparable performance in CIFAR-100 as Breeds.

ERM with Balanced Sampling

Both domains with the balanced sampling algorithm demonstrate near identical performance plots which show an eventual improvement from the original accuracy in Fig. 3-11a that plateaus at higher shot counts. The similar performance indicates the features of both domains are of equal complexity to train and the greater accuracy variability in Domain 4 than Domain 3 is tied to the actual subclasses meanings within the domains. As seen in the average domain performance in Fig. 3-11b, there is an initial decline in accuracy at 1-shot but rapidly recovers at higher shot counts. This drop in accuracy is due to the random sampling part of the balanced algorithm being unable to select enough points to build a good sample at 1-shot. We see that at high shot counts which allows the algorithm to randomly select more points per class, the issue is resolved and the accuracy increases.



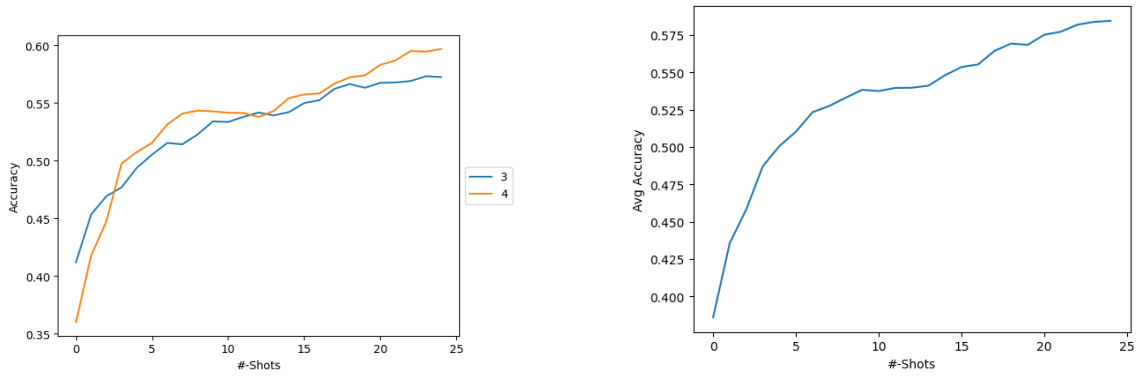
(a) Various Pruned Domains

(b) Average of Pruned Domains

Figure 3-11: ERM, Balanced Sampling on Cifar100

ERM with Class K-Means Sampling

The performance increase in the domains of CIFAR-100 in Fig. 3-12a is much less compared to that in Breeds even though the original accuracy is better. Domain 3 shows a consistently decreased rate of improvement as the number of shots increases while Domain 4's rate of improvement is more variable. Because CIFAR-100's training set has fewer subclasses within a class than Breeds, there lacks the same level of diversity in the trained model which leads to less complex features being sampled. Semantically, Domain 4's subclasses are more different than Domain 3's when compared to the original training data: sweet peppers is a vegetable while all the training examples/domain 3 examples were fruits or a train is a massive vehicle while all the training examples were smaller wheeled vehicles. An interesting trend occurs in Fig. 3-12b where the plot of low shot count accuracies (<= 14 shots) shows great initial improvement which eventually plateaus at higher shot counts, and the plot of high shot count accuracies (> 14 shots) shows the same shape but from the increased accuracy of 14 shots; this effect is largely impacted by Domain 4 but Domain 3 shows indications of the same pattern. If the subclass features are closely situated/overlapping in the feature space then the class K-Means algorithm would not find better points to sample near the cluster centers (low shot counts) but would rapidly discover more representative points when sampling farther away (high shot counts).



(a) Various Pruned Domains

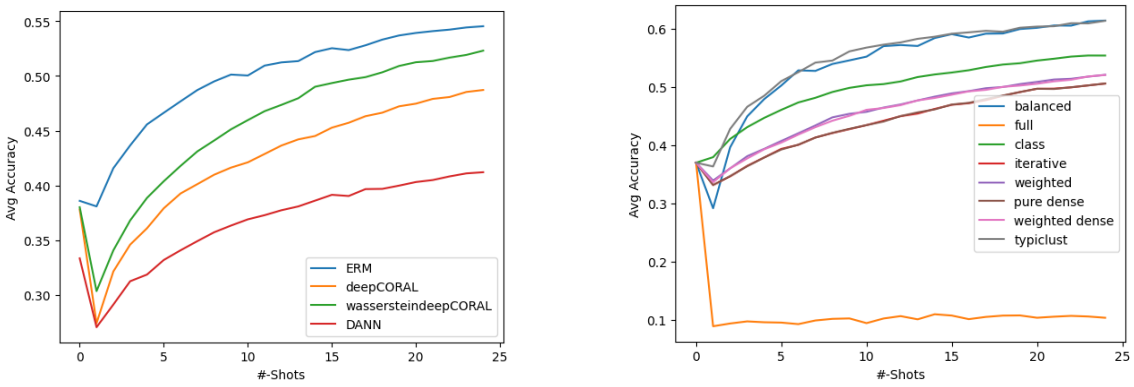
(b) Average of Pruned Domains

Figure 3-12: ERM, Class Sampling on Cifar100

General Analysis

Overall, the average performances of the 4 different models' OOD classifiers show better accuracy than the original classifier at low shot counts. We see in Fig. 3-13a that the performance decreases at 1-shot accuracy but rapidly rises to a plateau of about 10% greater accuracy than the original. The drop in accuracy at 1-shot is caused by the high complexity of the features which at 1-shot would not be able to select enough points to build a good sample. Once the shot count rises, a better and more representative sample is able to be selected and the performance of the OOD classifier rapidly grows; the performance plateaus at high shot counts due to the lack of increasingly representative points to be sampled. Unlike the other data sets seen in previous sections, the ERM model performs better than the other models by a non-trivial margin. The original ID classifier accuracy between ERM, wasser-steindeepCORAL, and deepCORAL is close but the difference becomes more drastic when retraining the OOD classifier. More testing is needed to determine if this issue is specific to the data set design or due to poor features from an insufficiently trained featurizer. The Wasserstein-based CORAL model consistently performs better than the original deepCORAL model which indicates the effectiveness of optimal transport in aligning domains.

Similar to Breeds, the average accuracies of the algorithms can be split into three different groupings in Fig.3-13b: balanced/Typiclust algorithm, the non-full K-Means



(a) Average Performance Over Generalization Models

(b) Average Performance Over Sampling Algorithms

Figure 3-13: Cifar-100 Performance Comparison

algorithms, and the full K-Means algorithm. This grouping is almost identical to that in Breeds so we will focus primarily on the points more unique to CIFAR-100. As seen in the average model performances, 1-shot performance of all algorithms except full K-Means and class K-Means is lower than the original ID classifier accuracy. This lowered performance is tied to the complexity of the data as explained previously. Aside from the full K-Means algorithm, the balanced sampling algorithm has the worst 1-shot performance. This is consistent with the data complexity explanation because the increased complexity lowers the chance of the balanced algorithm randomly selecting a representative point but does not impact the other algorithms as much due to the advantage of the feature geometry.

Aside from the lowered performance at 1-shot, the model and algorithm combinations perform very well above the original accuracy even at a low shot count.

Chapter 4

Conclusion

This research aims to identify the cause of poor out-of-distribution performance by verifying the effectiveness of re-training an out-of-distribution classifier compared to the original in-distribution classifier, to determine the effectiveness of a small sample of the out-of-distribution wild data when retraining, and to distinguish the sampling algorithms that can select the optimal and most representative sample set with no knowledge of the wild data labels. It is crucial to minimize the number of labeled OOD training data because labeling OOD data is very costly. A total of 4 data sets were used: 2 data sets that were natural wild data and 2 data sets that were synthetic data from large-scale image data sets. A total of 8 sampling algorithms were tested: 1 algorithm that had access to the data labels and 7 clustering-based algorithms that were label-blind.

Based on the analysis of the performance of the balanced algorithm on the wild iWildCam data set and synthetic Breeds/CIFAR-100 datasets where retraining the classifier using the OOD data performs significantly better than the ID classifier, we can conclude that the poor performance of the ID model on OOD data is caused by a combination of an exemplary ID featurizer and poorly generalizing ID classifier. We also saw that only a small amount of points per class is needed to provide excellent accuracy after retraining. Camelyon17, a wild data set of tumor cell images, did not have a similar performance as the other data sets and will be discussed in more detail in Sec. 4. Overall, the OOD balanced algorithm accuracies are consistently

multiple times larger than the ID accuracies on the majority of data sets regardless of shot count. While the algorithm is very effective on the majority of the data sets, the balanced sampling algorithm was specifically chosen due to its "cheating" nature where it could easily select the most class-balanced set of OOD data points as possible. In reality, knowing all the labels to the OOD data is extremely expensive so a focus of the research is to discover effective algorithms that don't require any labeled data.

Based on the analysis of the performances of the suite of K-Means algorithms on the combination of data sets and models, we can safely conclude that all the K-Means sampling algorithms (besides full K-Means) are capable of producing a similar improvement in accuracy without the additional information of labels. The K-Means clustering algorithm was specifically selected potentially leverage the clustering of points of each label in the feature space. While the performances of the K-Means algorithms are less than the balanced algorithm, the accuracy is still significantly greater than the original ID classifier accuracy at very low shot counts which supports the idea that, in wild data sets, points of the same label are clustered in the feature space; in some domains, we see a hundred-fold increase since the original accuracy is extremely close to 0. The poor performance of the full K-Means algorithm will be discussed in Sec. 4.

Based on the analysis of the excellent performances of the Wasserstein CORAL model on all the data set and sampling algorithm combinations, we can conclude that optimal transport is extremely effective in domain alignment and helps build well-generalized featurizers. In all cases, the Wasserstein CORAL model performs as well or better than the deepCORAL model. While not the original intent of the research, the superior accuracies demonstrated by domain alignment models and the desire for the featurizer to cluster points of equivalent labels led to the development of a model based on optimal transport.

In conclusion, there is strong evidence that supports all branches of the initial hypothesis seen in the performance analysis of the combination of data sets, models, and sampling algorithms. The issue of domain generalization is an ongoing and

difficult subject in academia with various real-world implications. While this thesis does not claim to provide a state-of-the-art generalization algorithm for training the featurizer, it does provide a low-cost approach using existing algorithms that should help bridge the performance gap between OOD and ID data.

Future Work

It is evident in the analysis that there are some aspects of the research that still requires more work to be done. Firstly, the OOD accuracies in Camelyon17 using any sampling algorithm and model is notably worse than the original accuracy. This is a problem that needs further research to determine if the issue is caused by a feature space geometry unique to Camelyon17 since 3 other data sets perform properly or by poorly trained ID models that do not have well-generalized ID featurizers. Second, the full K-Means sampling algorithm does not perform better than the original accuracy on any data set and model combination. More research is needed to be done to determine the exact faults in the algorithm design. Currently, the leading theory is that the lack of focus on clustering around classes inhibits the algorithm from sampling a class-balanced set of points. Third, the ERM model performs exceptionally well on the CIFAR-100 data set, unlike its performance on the other data sets. The exact reason for the high average ERM accuracy on CIFAR-100 needs to be verified; it is unclear if the ERM model is simply trained better than the other models on the ID data or if the data has a unique geometry that confounds domain alignment algorithms. Lastly, more analysis is needed on the feature space geometry of CIFAR-100 because it is the only data set where the class K-Means algorithm outperforms the augmented K-Means algorithms based on it.

Other work that needs to be done includes: testing the approach designed on other wild and synthetic domain generalization data sets, exploring applications to non-image data sets, and designing better sampling algorithms that perform well on any feature geometry.

Bibliography

[1] Jun-Hyun Bae, Inchul Choi, and Minho Lee. Meta-learned invariant risk minimization, 2021.

[2] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. 2015.

[3] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard S. Zemel, Wieland Brendel, Matthias Bethge, and Felix Wichmann. Shortcut learning in deep neural networks. *ArXiv*, abs/2004.07780, 2020.

[4] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization, 2020.

[5] Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets. 2022.

[6] Guy Hacohen, Avihu Dekel, and Daphna Weinshall. Active learning on a budget: Opposite strategies suit high and low budgets, 2022.

[7] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer retraining is sufficient for robustness to spurious correlations, 2022.

[8] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts, 2020.

[9] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research).

[10] J. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[11] Rafid Mahmood, Sanja Fidler, and Marc T. Law. Low budget active learning via wasserstein distance: An integer programming approach, 2021.

[12] Shi Na, Liu Xumin, and Guan Yong. Research on k-means clustering algorithm: An improved k-means clustering algorithm. In *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*, pages 63–67, 2010.

[13] Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning, 2022.

[14] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *ArXiv*, abs/1806.00451, 2018.

[15] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet?, 2019.

[16] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The risks of invariant risk minimization, 2020.

[17] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization, 2022.

[18] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[19] Steffen Schneider, Shubham Krishna, Luisa Eck, Wieland Brendel, M. Mathis, and Matthias Bethge. Generalized invariant risk minimization: relating adaptation and invariant representation learning. 2020.

[20] Anthony Sicilia, Xingchen Zhao, and Seong Jae Hwang. Domain adversarial neural networks for domain generalization: When it works and how to improve, 2021.

[21] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation, 2016.

[22] Luis Caicedo Torres, Luiz Manella Pereira, and M. Hadi Amini. A survey on optimal transport for machine learning: Theory and applications, 2021.

[23] Vladimir Naumovich Vapnik. Principles of risk minimization for learning theory. In *NIPS*, 1991.

[24] Yaqing Wang, Quanming Yao, James Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning, 2019.