STAT 37793 Final Report on Distribution Shift

Qingyao Sun (ID: 12276357)   Minxuan Duan (ID: 12276392)

Yucong Liu (ID: 12333959)*

May 30, 2022

Contents

*Alphabetical order

1 Introduction

One of the most common assumptions for statistical and machine learning models is that the training and test data are independently and identically sampled (IID) from the same distribution. However, this assumption usually does not hold. For example, models can systematically fail when tested on patients from different hospitals or people from different demographics. Figure 1 also shows some distribution shifts of structure, degree, or color on different datasets of graphs. Such case that a model is deployed on a data distribution $P_{\text{test}}(X, Y)$ related but different from what it was trained on $P_{\text{train}}(X, Y)$ is called **Distribution Shift**.

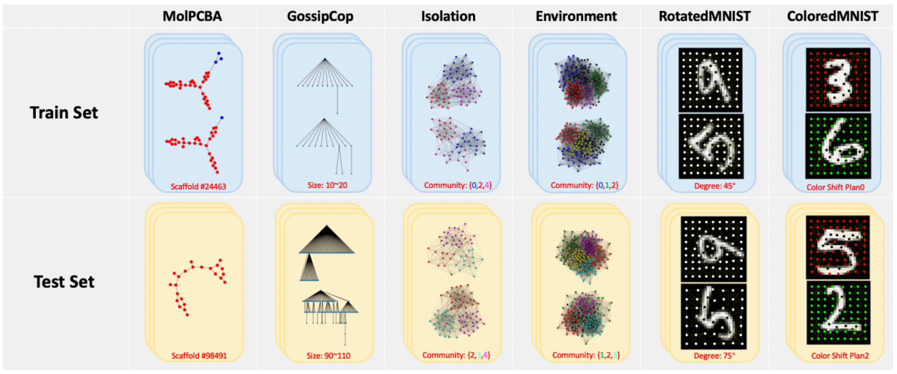


Figure 1: Distribution Shifts on Graphs [3]

Based on different assumptions, we can roughly divide distribution shift into three types:

1. **Covariate Shift** assume the testing distribution differs from the training distribution in covariate shift only, i.e. $P_{\text{test}}(X, Y) = P_{\text{test}}(X)P_{\text{train}}(Y \mid X)$. Since the usual underlying causality mechanism assumes x causes y and would not change, covariate shift is the most widely studied type of distribution shift.

2. **Label Shift** describes the converse causality, under the assumption that the label marginal can change $P_{\text{test}}(Y) \neq P_{\text{training}}(Y)$ but the class-conditional distribution remains fixed $P_{\text{test}}(X \mid Y) = P_{\text{train}}(X \mid Y)$. For example, doctors may want to predict patients' health condition based on the symptom. The possibility of having cancer varies from person to person. However, we note that in some degenerate cases where the label is deterministic, the label shift and covariate shift assumptions can hold simultaneously.

3. **Concept Shift** may sound weird since it assumes $P(Y \mid X)$ changes across domains. But we do encounter many scenarios where the causal mechanism varies over time or location, especially in the study of sociology and economics. These problem can be tricky to spot. We might hope to exploit knowledge that shift only takes place gradually either in a temporal or geographic sense.

No matter what type of distribution shift we face, it would poses significant robustness challenges in real-world ML applications. For instance, [8] builds new test sets for the CIFAR-10 and ImageNet. The authors observe sharp drops in accuracy ranging from 3% to 15% on CIFAR-10 and 11% to 14% on ImageNet (Figure 2). Their result also shows that every percentage point of progress on the original test set translates into more than one percentage point on the new test set.

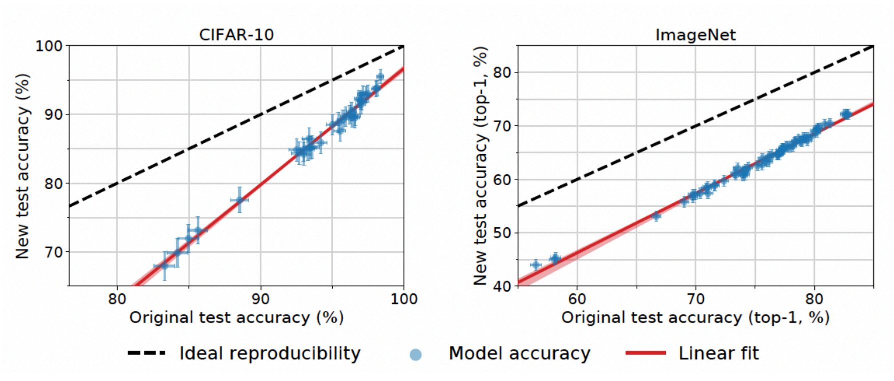


Figure 2: Caption

2 Literature Review of Theoretical Explanation

Why distribution shift harms the generalization accuracy of model has not been resolved but has attracted much research interest. Besides the experimental evidence, [8] also tried to discuss about potential causes of the accuracy drops, with the problem of classification as an example. If we aim to find a model $\hat{f}$ that minimizes the population loss

$$L_{\mathcal{D}}(\hat{f}) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} \mathbb{I}[\hat{f}(x) \neq y],$$

and since we do not know the true distribution $\mathcal{D}$, we instead measure the performance via a test set $\mathcal{S}$ drawn from $\mathcal{D}$:

$$L_S(\hat{f}) = \frac{1}{|\mathcal{S}|} \sum_{(x,y)\in\mathcal{S}} \mathbb{I}[\hat{f}(x) \neq y]$$

If we test by collecting a new test set $\mathcal{S}'$ from a data distribution $\mathcal{D}'$ that we carefully control to resemble the original distribution $\mathcal{D}$.

$$L_{\mathcal{S}} - L_{\mathcal{S}'} = \underbrace{(L_{\mathcal{S}} - L_{\mathcal{D}})}_{\text{Adaptivity gap}} + \underbrace{(L_{\mathcal{D}} - L_{\mathcal{D}'})}_{\text{Distribution Gap}} + \underbrace{(L_{\mathcal{D}'} - L_{\mathcal{S}'})}_{\text{Generalization gap}}$$

The first term $L_S - L_{\mathcal{D}}$ measures how much adapting he model $\hat{f}$ to the test set S causes the test error $L_{\mathcal{S}}$ to underestimate the population loss $L_{\mathcal{D}}$. The second term $L_{\mathcal{D}} - L_{\mathcal{D}'}$ quantifies how much the change from the original distribution $\mathcal{D}$ to our new distribution $\mathcal{D}'$ affects the model $\hat{f}$. Note The third term $L_{\mathcal{D}'} - L_{\mathcal{S}'}$ is the standard generalization gap commonly studied in machine learning. It is determined solely by the random sampling error. After detailed discussion, [8] concludes that the accuracy drops mainly stem from a large distribution gap.

To further take a close look at the theoretical explanation of distribution shift, we read through and present some interesting conclusions and insights here.

2.1 Theory of Concept Shift

[4] explains why accuracy drops by adopting the information theory. Using the chain rule of mutual information, one can express distribution shift as the sum of two separate components:

$$\underbrace{I(XY;t)}_{\text{Distribution shift}} = \underbrace{I(X;t)}_{\text{Covariate shift}} + \underbrace{I(Y;t \mid X)}_{\text{Concept shift}}$$

where $I(XY;t) = D_{\mathrm{KL}}(P(X,Y \mid t) \mid P(X,Y)), t = \text{test or train}$.

For any model $Q(Y \mid X)$ and $\alpha := \min\{P(t = \text{test }), P(t = \text{train })\}$

$$\underbrace{D_{\mathrm{KL}}\left(P_{Y|X}^{t=\text{ train}} \| Q_{Y|X}\right)}_{\text{Train error}} + \underbrace{D_{\mathrm{KL}}\left(P_{Y|X}^{t=\text{ test}} \| Q_{Y|X}\right)}_{\text{Test error}} \geq \frac{1}{1-\alpha} I(Y;t \mid X)$$

Thus, whenever the selection induces concept shift, any sufficiently flexible model must incur in strictly positive test error.

2.2 Theory of Covariate Shift

Besides, [10] focuses on covariate shift generalization. For common loss functions, the covariate shift generalization problem can be tackled by the minimal stable variable set whiclipe satisfies the condition of the following theorem.

Theorem 1. *A subset of variables $S \subseteq X$ that can approximate the target $\mathbb{E}_{P_{test}}[Y \mid X]$ if and only if it satisfies $\mathbb{E}_{P_{train}}[Y \mid S] = \mathbb{E}_{P_{train}}[Y \mid X]$.*

And the existence and uniqueness of such variables are guaranteed.

Theorem 2. *Under ideal conditions (perfectly learned sample weights and infinite samples),*

- *if X_i is not in the minimal stable variable set, stable learning algorithms could filter it out,*
- *otherwise, there exists sample weighting functions with which stable learning algorithms could identify X_i.*

3 Examine on Distribution Shift

Here we design some simple test to demonstrate some properties on distribution shift. We want to see what may happen if there is a distribution shift between train set and test set. Our idea is to add a small distribution shift on the test set.

3.1 Dataset and Experiment Design

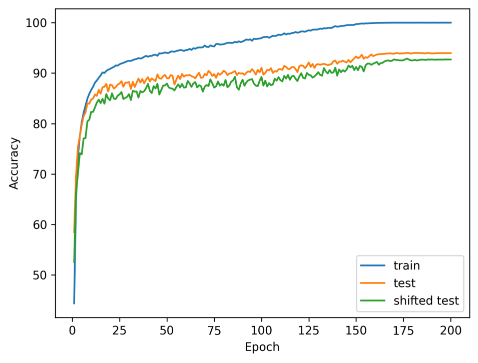
The CIFAR-10 dataset consists of 60000 instance of 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. On CIFAR-10 experiment, we use ResNet-18 as the backbone neural network. We use weight decay of 5×10^{-4}. We set learning rate 0.01, batch size 32, momentum 0.9 and train 200 epochs with Cosine Annealing.

We remain the train set unchanged, and use two kinds of distribution shift on test set.

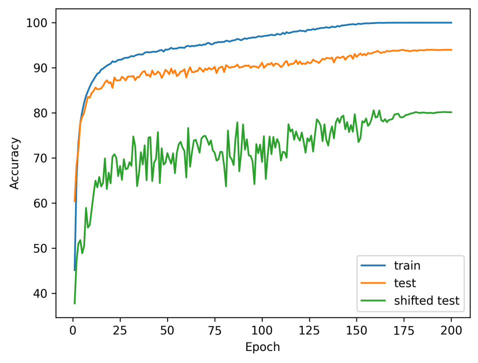
1. Fixed shift
2. Gaussian shift

Fixed shift means we add a fixed translation on test data set. We add 5 and 25 on each input x. Because CIFAR-10 is a dataset of images, this fixed shift can be regarded as a fixed change on color at each pixel in the image.

Gaussian shift means we add a Gaussian noise $\epsilon \sim N(0,\sigma)$ on each test data. We set $\sigma = 0.01, 0.1$ in our experiments.This distribution shift can be regarded as image noise when collecting such image data. We call Gaussian shift 0.1, means Gaussian shift with $\sigma = 0.1$.

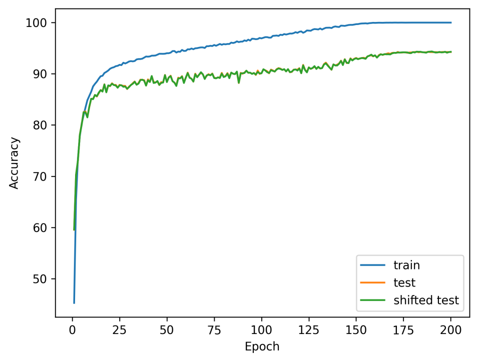


(a) Fixed Shift 5

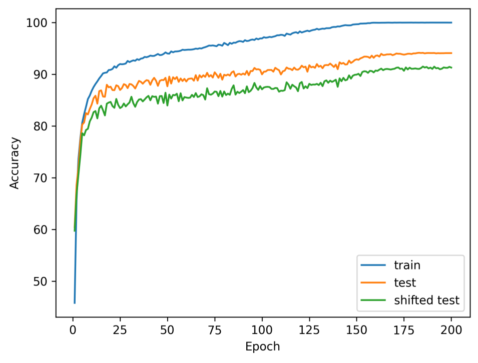


(b) Fixed Shift 25

Figure 3: Performance on CIFAR-10 with Fixed Shift



(a) Gaussian Shift with standard error 0.01



(b) Gaussian Shift with standard error 0.1

Figure 4: Performance on CIFAR-10 with Gaussian Shift

There also many other kind of distribution shift, for example Shot Noise, Impulse Noise, Defocus Blur Frosted, Glass Blur [6]. For simplicity, we only test fixed shift and Gaussian shift in our experiments.

3.2 Results

3 shows the performance with fixed shift and 4 shows the performance with Gaussian shift. The blue line is the train accuracy each epoch. The orange and green lines are test accuracy on original test data and shifted test accuracy.

3.3 Analysis

Firstly, neural network shows robustness toward distribution shift. With fixed shift 5, test accuracy only drops about 1%, and with Gaussian shift 0.01, test accuracy almost the same. With such kind of shift and perturbation, our neural network is steady on test accuracy. Thus we think that robustness toward distribution shift exists.

Here we give a simple explanation about the robustness in our task. The task on CIFAR-10 is image classification. Our model output a score vector for each data. Each element in the score vector means the score of each class, and we choose the largest score as the predicted class. If the data is shifted a little, our network may output different score vector a little, then the largest one may keep the same. So we will have same prediction which means the test results keep similar.

Secondly, models are not able to undertake every distribution shift. With fixed shift 25, the test accuracy drops more than 10% and with Gaussian shift 0.01, the test accuracy drops about 3%. We observe that obvious accuracy drops may occur with some kind of distribution shift. Such heavy distribution shift are unbearable for neural network models. With heavily shifted test data, networks may output totally different score vector and give different but incorrect prediction. Such shifted distribution and data structures are not learned by the model in the training process, then the model makes more mistakes after shift on data.

Thirdly, we observe one interesting trend on the test accuracy. Test accuracy and shifted test accuracy show similar trend even if there is a gap between them. We guess our designed distribution shift didn't destroy the original data structure, and the shifted data distribution can be partly learnable in the training process.

3.4 Open Question and Discussion

In previous observation and discussion, we know that robustness toward distribution exists, but it can not be universal. There is supposed to be some constrain on distribution shift. Intuitively, we can add arbitrary shift on the training distribution as a distribution shift and get an arbitrary test distribution. However that is impossible for models to be robust on arbitrary test distribution. On the other hand, a reasonable distribution shift should not destroy the original data structure or some potential data features. For example, a Gaussian noise $\epsilon \sim N(0, \sigma)$, where $\sigma \to \infty$ can not be a reasonable distribution shift, because the Gaussian noise almost cover the original data distribution. We can only see the noise instead of the original data. While a Gaussian noise $\epsilon \sim N(0, \sigma)$, where $\sigma \to 0$, will make the test distribution almost same as train distribution.

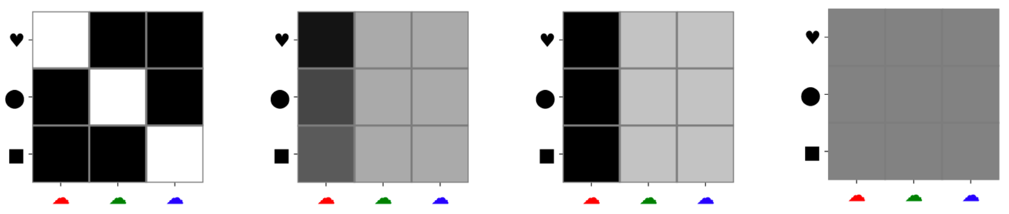
Here we propose an open question.

- How to define/check if one distribution shift is a reasonable shift?
- How to measure a distribution shift which doesn't destroy the original structure of data?

Though we use the test accuracy drop to discuss robustness and performance under distribution shift, we should not use information about model to measure distribution shift. The robustness of models and the measure on distribution shift are two highly relative concepts. We think KL-divergence might be a good idea to measure distribution shift. This can be thought as a relative measure on distribution shift. A absolute measure can be also meaningful in this question like entropy of the shift itself. However, these questions required more exploration in details.

4 Empirical Study of Defenses Against Distribution Shift

In this section we first review the ICLR 2022 Oral paper A Fine-Grained Analysis on Distribution Shift [9]. We think it is valuable in the sense that provides an empirical overview of the robustness of the SoTA methods, and can help practitioners decide which approach to use to defend against a particular kind of distribution shift.



(a) p_{train}: **SC.** (b) p_{train}: **LDD.** (c) p_{train}: **UDS.** (d) p_{test}: y^l, y^a **are IID.**

Figure 5: Visualization of the joint distribution for spurious correlation, low-data drift, and unseen data shift the authors consider on the DSprites example. The lighter the color, the more likely the given sample.

However, we don't quite agree with the authors on some details of the experiment setting, so we conducted our own evaluation as a further exploration. For the record, our experiment does not depend on the framework[1] open sourced by the authors, which seems unusable due to its insufficient dependency specification and some obvious bugs in the code.

4.1 Paper review: *A Fine-Grained Analysis on Distribution Shift*

The paper does an experimental evaluation of the generalization performance of 19 methods across 6 datasets and 3 distribution shifts with varying label noise and dataset size.

4.1.1 Framework

The authors propose a Bayesian structure as the underlying model, i.e.

$$z \sim p(z) \qquad y^i \sim p(y^i|z) \quad i = 1 \dots K \qquad \boldsymbol{x} \sim p(\boldsymbol{x}|z), \tag{1}$$

where z is the latent factor, $\boldsymbol{x}$ is the inputs, $y^1, y^2, \cdots, y^K$ are called "attributes", and one of these K attributes is a label of interest, denoted as y^l. By a simple re-factorization, we can write

$$p(y^{1:K}, \boldsymbol{x}) = p(y^{1:K}) \int p(\boldsymbol{x}|z)p(z|y^{1:K})dz = p(y^{1:K})p(\boldsymbol{x}|y^{1:K}).$$

The distribution shift discussed in this paper is **label shift**. Specifically, we have

$$\begin{aligned} p_{\text{train}}(y^{1:K}) \neq & p_{\text{test}}(y^{1:K}) \\ p_{\text{train}}(\boldsymbol{x}|y^{1:K}) = & p_{\text{test}}(\boldsymbol{x}|y^{1:K}) \end{aligned} \tag{2}$$

For the training set, they consider three kinds of distribution shifts: Spurious correlation, Low-data drift, and Unseen data shift. For the test set, they assume that the attributes are distributed uniformly. See Figure 5 for more details.

[1] https://github.com/deepmind/distribution_shift_framework

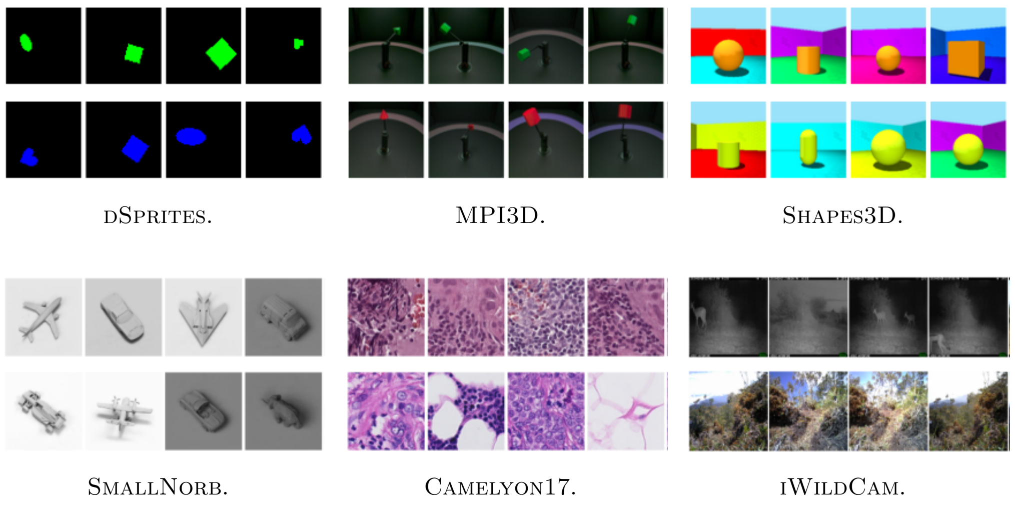


Figure 6: **Dataset samples**. Each row fixes an attribute (e.g. color for dSprites, MPI3D, Shapes3D; azimuth for SmallNorb; hospital for Camelyon17; and location for iWildCam).

4.1.2 Experiment Setup

Datasets The authors evaluated on 6 different datasets. These datasets differ in their properties: dSprites, MPI3D, Shapes3D are all synthetic; SmallNorb consists of black and white images of toys; Camelyon consists of medical imagery and iWildCam of camera trap imagery.

See Figure 6 for some samples from the datasets used in the study. In particular, they make **shape** the label of interest and **color** the non-label attribute in the dSprites dataset, as shown in the top left sub-figure.

Methods The authors evaluated 19 approaches[2] that can be used to improve model robustness, grouped into five categories. In particular, they looked at the following methods

Architecture choice ResNet18, ResNet50, ResNet101, ViT, and an MLP.

Heuristic data augmentation Standard ImageNet augmentation, AugMix without JSD, RandAugment, and AutoAugment.

Learned data augmentation CycleGAN.

Domain generalization IRM, DeepCORAL, domain MixUp, DANN, and SagNet.

Adaptive approaches JTT, and BN-Adapt.

Representation learning β-VAE, and pretraining on ImageNet.

Experiments The authors essentially did a grid search over everything, training over 85K models in total. In particular, for each combination of method, dataset, distribution shifts, label noise, and dataset size, they train multiple models with different seeds and evaluate them on the test set.

[2]or 18 if not counting the baseline ResNet

4.1.3 Main Results

See Figure 7, Figure 8, and Figure 9 for the aggregated results. In particular, Figure 7 and Figure 8 shows the relative change in accuracy with respect the ResNet baseline, e.g. 0.2 means an improvement of 20%. To clarify, higher N means less severe distribution shift, and always leads to better predictive accuracy, but the average improvement over the ResNet baseline may decrease, which is why there are more red in the lower half of the figures.

Figure 8 shows the relative ranking of each method. For example, the columns labelled with Pretrained and ImageNet is thicker in the upper half, which means they consistently get high rank when compared with other methods. The black bar of Pretrained corresponds to ranking 1 in the Y-axis, which mean it is the best method at least half of the time.

According to the result, data augmentation could lead to degraded performance in some cases, which seems to contradict the conventional wisdom. The authors address this question by noting that the performance of augmentation methods depends on how well they approximate the true generative model. They provide a detailed break-up of the performance of the augmentation methods on each dataset in Figure 10.

4.1.4 Take-home Messages

- While we can improve over ERM, no one method always performs best.
- Pretraining is a powerful tool across different shifts and datasets[3].
- Heuristic augmentation improves generalization if the augmentation describes an attribute.
- Learned data augmentation is effective across different conditions and distribution shifts.

4.1.5 Possible Improvements

Joint uniform distribution The authors assume the attributes have a joint uniform distribution on the test set, but we don't find the assumption very realistic. For example, in a medical setting, the types of equipment may not distribute evenly across all hospitals, the proportion of patients with a tumor is not necessarily 50%, and if "sex" and "pregnancy" are two of the attributes then we are forced to consider pregnant men.

However, as unrealistic as it is, we note that the joint uniform assumption can be used to avoid bias and improve fairness by ensuring each sub-population receives the equal amount of attention from the model.

Only two attributes of variation The authors only consider two attributes in each dataset, leaving only one non-label attribute of variation. For example, they make shape the label and object color the non-label attribute in the SHAPES3D dataset, but there are no distribution shift happening in other attributes like floor hue, orientation, scale, shape, and wall hue. Ideally we could explore the effect of shifting the joint distribution of multiple attributes, as this is what typically happens in practice.

4.2 Our Evaluation

We implement the possible improvements described in the last paragraph.

[3]To be fair, pre-training effectively mitigates distribution shift by giving the model data access to additional data, so this is a little cheating.

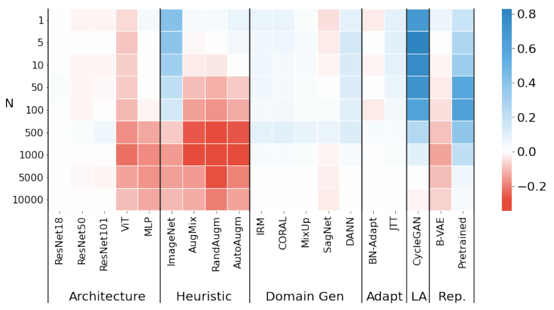


Figure 7: **Spurious Correlation.** The authors use all correlated samples and vary the number of samples N from the true, uncorrelated distribution. They plot the relative change over the baseline ResNet, averaged over all seeds and datasets. Blue is better, red worse.

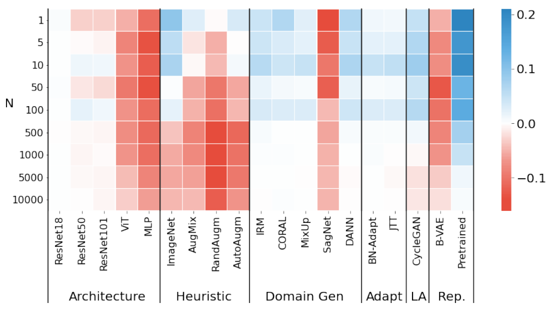


Figure 8: **Low-data drift.** The authors use all samples from the high data regions and vary the number of samples N from the low-data region. They plot the relative change over the baseline ResNet, averaged over all seeds and datasets. Blue is better, red worse.

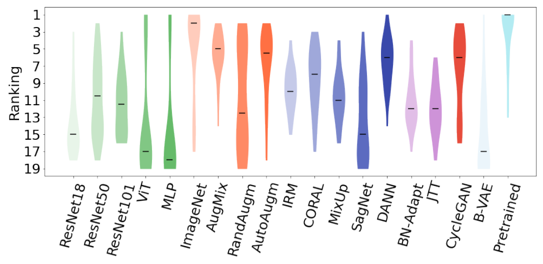


Figure 9: **Unseen data shift.** We rank the methods (where best is 1, worst 19) for each dataset and seed and plot the rankings, with the overall median rank as the black bar.

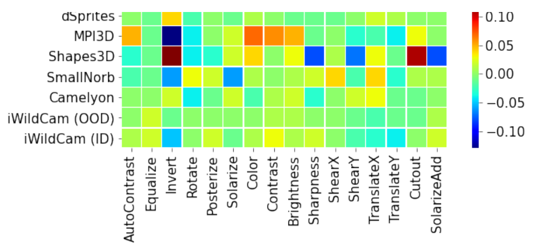


Figure 10: **RandAugment ablation.** Performance of each augmentation across the different datasets. As can be seen, no one augmentation always improves performance. An augmentation that may improve performance on one dataset (e.g. invert on Shapes3D) hurts performance on another (e.g. invert on SmallNorb).

4.2.1 Model

We took inspiration from the paper, but put a distribution shift on z instead of y, i.e.

$$z \sim p(z) \qquad y^i \sim p(y^i|z) \quad i = 1 \dots K \qquad \boldsymbol{x} \sim p(\boldsymbol{x}|z), \tag{3}$$

where $p(z)$ is different on the training and test set, but both $p(y^i|z)$ and $p(\boldsymbol{x}|z)$ stays the same. Note that shifting the distribution of z can give us covariate shift, label shift, and concept shift simultaneously.

In our case, $p(y^l|z)$ is a point-mass distribution and $p(\boldsymbol{x}|z)$ is a normal distribution whose mean is determined by z. To be clear, y^l does not need to be a deterministic function of z, but we did not introduce any noise here for simplicity.

4.2.2 Experiment Setup

Dataset We used the SHAPES3D dataset from Deepmind [2], which contains the following latent factor values, with no noise added. Note that this is the same SHAPES3D used in the paper, and we refer readers to the top right sub-figure of Figure 6 for some samples from it.

floor hue 10 values linearly spaced in [0, 1]

wall hue 10 values linearly spaced in [0, 1]

object hue 10 values linearly spaced in [0, 1]

scale 8 values linearly spaced in [0, 1]

shape 4 values in [0, 1, 2, 3]

orientation 15 values linearly spaced in [-30, 30]

Each combination of the latent factors corresponds to a $64 \times 64 \times 3$ RGB image, which we preprocess by scaling the elements into the range [0., 1.]. This gives us a total of $10 \times 10 \times 10 \times 8 \times 4 \times 15 = 480000$ images.

To make some connection with the model introduced in the previous paragraph, in our experiment z is a six-tuple of the the latent factors, $\boldsymbol{x}$ is the corresponding image with an additive Gaussian noise $\mathrm{N}(0, \sigma^2)$ and clamped in to the range $[0, 1]$, and $\boldsymbol{y}$ is a component of z, e.g. scale or shape.

Distributions We adjust the weight of each sample based on its latent factor. The code snippet below shows the unnormalized weights/distributions we used, where `scale`, `shape`, and `orientation` all vectors with size 480000 which contain indices of the corresponding factor for each sample, scaled to fit evenly between 0 and 1 inclusive. For example, there are 4 distinct shapes in the dataset, so the values of the i-th element of `shape` is one of $\left\{0, \frac{1}{3}, \frac{2}{3}, 1\right\}$ indicating the shape of the i-th image, for $1 \le i \le 480000$.

Each of the following distributions is used as $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$, resulting in $12 \times 12 = 144$ pairs. Note that it characterizes both additive and multiplicative interactions between the attributes.

```
weights = {
    'uniform': np.ones(480000),
    'scale': scale,
    'shape': shape,
    'orientation': orientation,
    'scale+shape': scale + shape,
    'scale+orientation': scale + orientation,
    'shape+orientation': shape + orientation,
    'scale+shape+orientation': scale + shape + orientation,
    'scale*shape': scale * shape,
    'scale*orientation': scale * orientation,
    'shape*orientation': shape * orientation,
    'scale*shape*orientation': scale * shape * orientation,
}
```

Methods Due to the restriction of time and computing power, we only evaluated the robustness of standard ERM. The models we are using include ResNet18, ResNet34, ResNet50, and ResNet101 [5].

Experiment Here is an overview of our experiment.

1. Sample a training set from the training distribution $\mathcal{D}_{\text{train}i}, i = 1, \cdots, 480000$.
2. Train a neural network with training set.
3. Evaluate the accuracy on the population, i.e. each of the 480000 images. Denote it with A_i, for $i \in \{1, \cdots, 480000\}$.
4. Calculate the expected test accuracy on $\mathcal{D}_{\text{test}}$ with $\sum_{i=1}^{480000} A_i \mathcal{D}_{\text{test}i}$.

The training set contains 303360 specimens.[4] Since we are sampling with replacement, on average the training set only contains $0.632 \times 0.632 \approx 40\%$ unique specimens from the population. This limits the neural networks' potential of memorization.

We implemented the experiments with JAX [1] and Haiku [7]. The source code of our experiments can be found on GitHub.[5] We conduct the experiments on five Cloud TPU v3-8 virtual machines provided by Google's TPU Research Cloud program. We use the following hyperparameters in all of our experiments. While we are reporting the seed, note that the result may not be fully reproducible due to the fact that we are doing distributed training across 8 TPU cores, and that floating point addition is not associative.

Seed 42

Noise Scale (σ) 0.01

Device count 8

Local batch size 2048

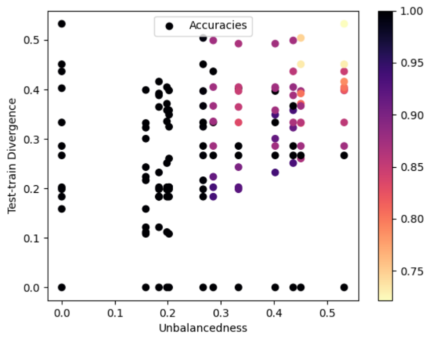
Training batches 512

Learning rate 0.001

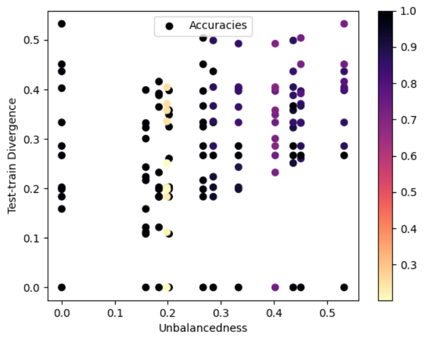
We note that using a σ of 0.1 or 0.001 and different seeds produces roughly the same result, which means the pattern we are reporting is robust.

[4] the size is an arbitrary choice, but it comes from the 0.632 rule of bootstrapping: $0.632 \times 480000 = 303360$.

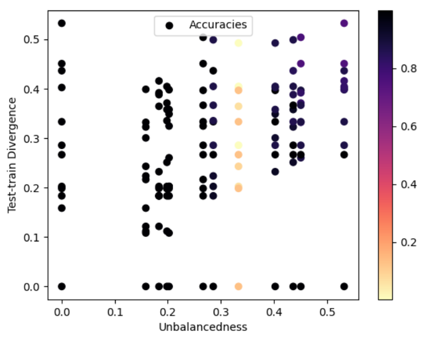
[5] https://github.com/nalzok/distribution_shift



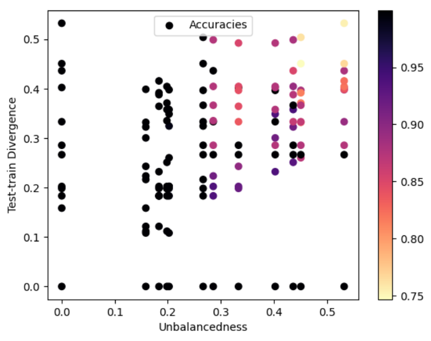
(a) ResNet18



(b) ResNet34



(c) ResNet50



(d) ResNet101

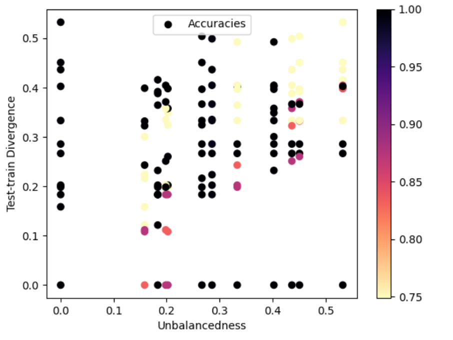
Figure 11: Using **Scale** as the label. Here **Unbalancedness** stands for the total variation distance between a uniform distribution and $\mathcal{D}_{\text{train}}$, and **Test-train Divergence** means the total variation distance between $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$. Darker means a higher test accuracy.

4.2.3 Main Results

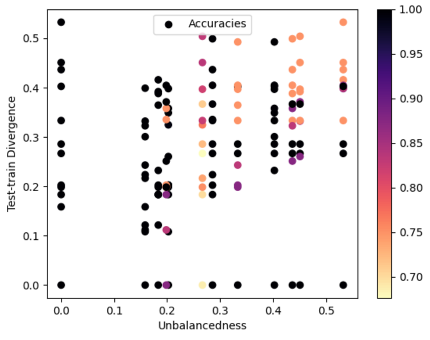
We present the aggregated result of our experiments in Figure 11, Figure 12, and Figure 13. The figures collectively depict two general trends. Firstly, points near the X-axis are typically dark, which means the test accuracy will be high when $\mathcal{D}_{\text{train}} \approx \mathcal{D}_{\text{test}}$. This is unsurprising. Secondly, all points near the Y-axis are black, which means the test accuracy will be high when the training data come from a balanced distribution, no matter how $\mathcal{D}_{\text{test}}$ looks like, which we think is an interesting observation.

4.2.4 Take-home Messages

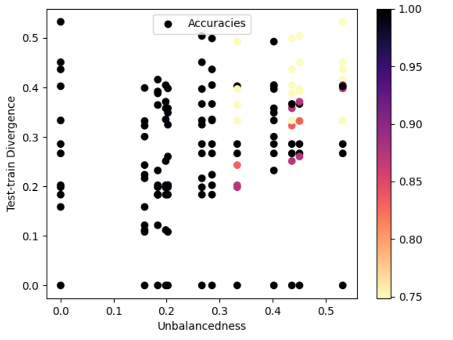
- Neural networks trained with balanced data have high accuracy regardless of $\mathcal{D}_{\text{test}}$, making even ERM immune to distribution shift.
- Neural networks trained with unbalanced data have high accuracy when the $\mathcal{D}_{\text{test}}$ resembles $\mathcal{D}_{\text{train}}$, but works poorly when the $\mathcal{D}_{\text{test}}$ looks very different from $\mathcal{D}_{\text{train}}$.
- When measuring the severeness of distribution shift, we should take the balancedness of the training data/distribution into account.



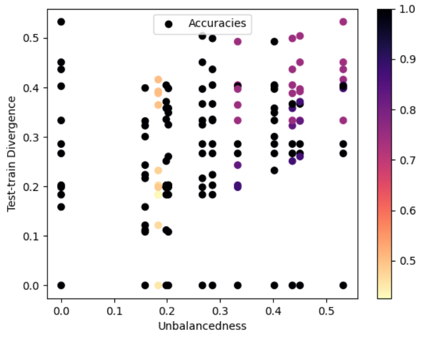
(a) ResNet18



(b) ResNet34

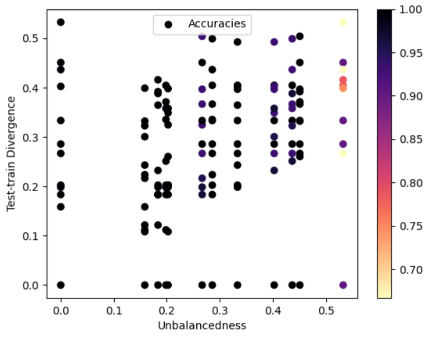


(c) ResNet50

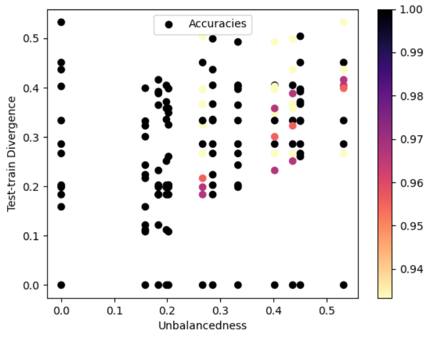


(d) ResNet101

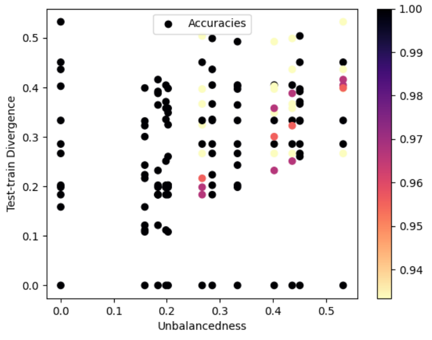
Figure 12: Using **Shape** as the label



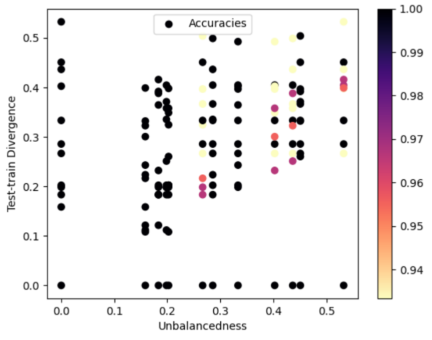
(a) ResNet18



(b) ResNet34



(c) ResNet50



(d) ResNet101

Figure 13: Using **Orientation** as the label

5 Teamwork Contribution

Minxuan Duan is responsible for Section 1 and 2. Yucong Liu is responsible for Section 3. Qingyao Sun is responsible for Section 4.

References

[1] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs.* Version 0.2.5. 2018. URL: http://github.com/google/jax.

[2] Chris Burgess and Hyunjik Kim. *3D Shapes Dataset.* https://github.com/deepmind/3dshapes-dataset/. 2018.

[3] Mucong Ding et al. "A Closer Look at Distribution Shifts and Out-of-Distribution Generalization on Graphs". In: (2021).

[4] Marco Federici, Ryota Tomioka, and Patrick Forré. *An Information-theoretic Approach to Distribution Shifts.* arXiv:2106.03783. type: article. arXiv, Nov. 1, 2021. DOI: 10.48550/arXiv.2106.03783. arXiv: 2106.03783[cs,math]. URL: http://arxiv.org/abs/2106.03783 (visited on 05/24/2022).

[5] Kaiming He et al. *Deep Residual Learning for Image Recognition.* 2015. DOI: 10.48550/ARXIV.1512.03385. URL: https://arxiv.org/abs/1512.03385.

[6] Dan Hendrycks and Thomas Dietterich. "Benchmarking neural network robustness to common corruptions and perturbations". In: *arXiv preprint arXiv:1903.12261* (2019).

[7] Tom Hennigan et al. *Haiku: Sonnet for JAX.* Version 0.0.3. 2020. URL: http://github.com/deepmind/dm-haiku.

[8] Benjamin Recht et al. "Do imagenet classifiers generalize to imagenet?" In: *International Conference on Machine Learning.* PMLR. 2019, pp. 5389–5400.

[9] Olivia Wiles et al. *A Fine-Grained Analysis on Distribution Shift.* arXiv:2110.11328. type: article. arXiv, Nov. 25, 2021. DOI: 10.48550/arXiv.2110.11328. arXiv: 2110.11328[cs]. URL: http://arxiv.org/abs/2110.11328 (visited on 05/24/2022).

[10] Renzhe Xu et al. *Why Stable Learning Works? A Theory of Covariate Shift Generalization.* arXiv:2111.02355. type: article. arXiv, Nov. 3, 2021. DOI: 10.48550/arXiv.2111.02355. arXiv: 2111.02355[cs,stat]. URL: http://arxiv.org/abs/2111.02355 (visited on 05/24/2022).