AutoEval: Are Labels Always Necessary for Classifier Accuracy Evaluation?

Weijian Deng and Liang Zheng, *Senior Member, IEEE*

Abstract—Understanding model decision under novel test scenarios is central to the community. A common practice is evaluating models on labeled test sets. However, many real-world scenarios see unlabeled test data, rendering the common supervised evaluation protocols infeasible. In this paper, we investigate such an important but under-explored problem, named Automatic model Evaluation (AutoEval). Specifically, given a trained classifier, we aim to estimate its accuracy on various unlabeled test datasets. We construct a meta-dataset: a dataset comprised of datasets (sample sets) created from original images via various transformations such as rotation and background substitution. Correlation studies on the meta-dataset show that classifier accuracy exhibits a strong negative linear relationship with distribution shift (Pearson's Correlation $r < -0.88$). This new finding inspires us to formulate AutoEval as a dataset-level regression problem. Specifically, we learn regression models (e.g., a regression neural network) to estimate classifier accuracy from overall feature statistics of a test set. In the experiment, we show that the meta-dataset contains sufficient and diverse sample sets, allowing us to train robust regression models and report reasonable and promising predictions of the classifier accuracy on various test sets. We also provide insights into application scopes, limitations, and potential future directions of AutoEval.

Index Terms—Automatic model evaluation, meta-dataset, accuracy estimation, dataset-level regression

1 Introduction

Model evaluation is an indispensable step in almost every computer vision task. Using a fully labeled test set that is unseen during training, the goal of evaluation is to estimate a model's (hopefully) unbiased accuracy when deployed in real-world scenarios. In most cases, the evaluation process is *supervised*, meaning that a labeled test set is given, so that the model accuracy is calculated by comparing the predicted labels with the ground truth labels. Fig. 1a demonstrates such an example. In the community, there are many well-established benchmarks, such as ImageNet [1] and COCO [2]), which provide hold-out labeled test sets for model evaluation.

Compared with such evaluation on benchmarks, model evaluation in real-world deployment is not that straightforward. In the real world, conditions where models are deployed often differ significantly from the conditions where they are trained. As such, model performance on a benchmark test set may not reflect the same during deployment. This requires us to re-evaluate the model accuracy upon deployment. However, we often face scenarios where annotations of test samples are not provided. What is more, it is very complex and expensive to manually gather test data annotations. Even if acquired, the annotated samples may only represent a very limited set of environments, which adds bias to the evaluated performance. For example, it is costly to annotate test samples for license plate recognition systems; even if plate labels are gathered for every car, they still cannot capture the diversity of real-world circumstances, such as various lighting and weather conditions. This raises an interesting question: *can we estimate the model performance on a test set without test labels?*

To answer this question, this paper studies an *unsupervised* evaluation problem: Automatic model Evaluation (AutoEval). Given a classifier trained on a training set, the goal is to estimate its accuracy on an unlabeled test set. Here, we show an example in Fig. 1b. Given a digit classifier trained on MNIST [3], we want to predict its classification accuracy on test sets *without* ground truths. Being able to evaluate a classifier in such an unsupervised manner is crucial for its application in fields like autonomous driving and medical diagnosis. This problem is yet challenging, because a test set contains many images, and each image has varied and rich visual contents. Existing literature in domain adaptation provides us with important hints. Particularly, a small distribution difference leads to a high target distribution accuracy [4], [5], [6]. This implies that a large domain shift causes a low test accuracy. In Fig. 1b, by visually inspecting the overall differences between test and training sets, we can roughly infer that the accuracy on the test set is low.

The above insight (distribution shift degrades accuracy) inspires us to address AutoEval from the perspective of data distribution. More specifically, we empirically investigate the underlying relationship between distribution shift and accuracy. To find out such a relationship, we need to evaluate a trained classifier on many different test sets, which, however, are difficult to collect. Therefore, we propose to use

• *The authors are with the School of Computing, The Australian National University, Canberra, ACT 0200, Australia. E-mail: {weijian.deng, liang.zheng}@anu.edu.au.*

Manuscript received 1 May 2021; revised 25 September 2021; accepted 7 December 2021. Date of publication 17 December 2021; date of current version 5 February 2024.
This work was supported in part by Australian Research Council (ARC) Discovery Early Career Researcher Award under Grant DE200101283, in part by ARC Centre of Excellence in Robotic Vision under Grant CE140100016, and in part by ARC Discovery Project under Grant DP210102801.
(Corresponding author: Liang Zheng.)
Recommended for acceptance by L. Liu, T. Hospedales, Y. LeCun, M. Long, J. Luo, W. Ouyang, M. Pietikinen, and T. Tuytelaars.
Digital Object Identifier no. 10.1109/TPAMI.2021.3136244

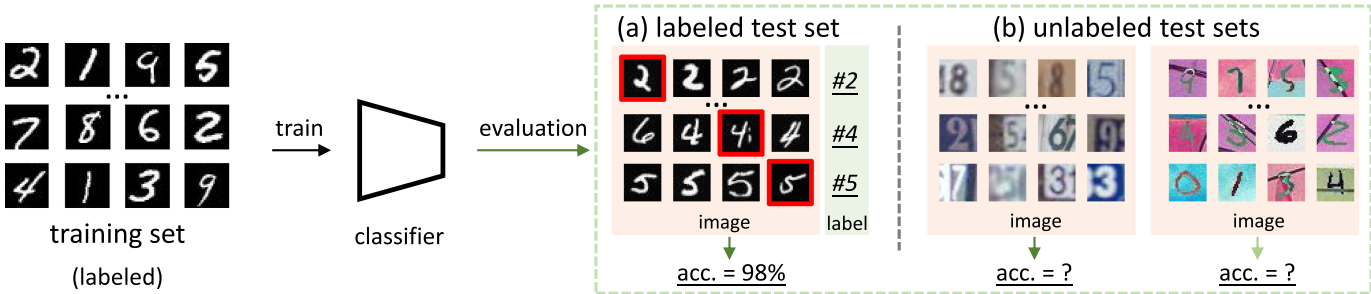

Fig. 1. Problem illustration. Given a classifier trained on a training set, we commonly use a labeled test set to gain an estimate of its out-of-sample performance, as shown in (a). However, real-world data (b) often follow distributions that differ from the original training distribution. As such, the accuracy on the labeled test set (a) might not truly reflect the classifier generalization in real-world deployment scenarios. This motivates us to explore the problem of Automatic model Evaluation, where the goal is to estimate classifier accuracy on various unlabeled test sets.

data synthesis to construct a meta-dataset (dataset of datasets).[1] Unlike most existing works that treat each image as a sample, we focus on the dataset level: in the meta-dataset, each dataset is treated as a sample, which we term the "sample set". Every sample set in the meta set is generated from a seed set that follows the same distribution as the original training set. The generation process is achieved via various geometric and photo-metric transformations on the seed set e.g., blurring, background substitution, foreground rotation, and translation. In the experiment, we show that the generated sample sets exhibit a diverse spread of distributions. Importantly, these synthetic sample sets are fully labeled because they are transformed versions of the seed set. Using these labels, we can calculate the classifier accuracy on each sample set.

We conduct correlation studies on the meta dataset and report that there exists a very strong negative linear correlation between classifier accuracy and distribution difference (the Pearson's Correlation $r < -0.88$). This new finding shows that it is feasible to estimate classifier accuracy from distribution statistics. Therefore, we formulate AutoEval as a *dataset-level regression* problem. In Table 1, we show the analogy between standard image classification and the AutoEval task. Because AutoEval is a dataset-level task, we should find an effective representation for each sample set and learn a desirable prediction function.

We propose to train regression models on the meta-dataset. Our regression takes as input the representation of a dataset and outputs the estimated classifier accuracy on this set. In this work, we use the distribution-related statistics, e.g., mean and covariance, to represent a dataset. Subsequently, a sample set is denoted as $(\boldsymbol{f}_i, a_i)$, where a_i is the classification accuracy on this sample set, and $\boldsymbol{f}_i$ is the sample set representation. Given a meta set denoted as $\{(\boldsymbol{f}_i, a_i)\}, i = 1, \ldots, N$, where N is the number of sample sets, we learn two regression models: linear regression and neural network regression. Both are shown to have decent performance in accuracy prediction.

We summarize the main points of this paper below.

- We introduce AutoEval, an unsupervised evaluation problem where the goal is to estimate the accuracy of a trained classifier on a test set *without* any human annotated labels.
- We construct a meta-dataset (a dataset comprised of many datasets) and empirically report a strong negative linear relationship between classifier accuracy and distribution shift. This new finding demonstrates the feasibility of estimating classifier accuracy from overall feature statistics.
- We formulate AutoEval as a dataset-level regression problem. We learn robust regression models on meta-dataset. The experimental results show our models can obtain promising predictions for real-world test datasets.

This article is extended from our conference paper [7]. The major extensions include the following. (i) We provide a detailed correlation study in Section 3.1, where we empirically measure the relationship between classifier accuracy and distribution shift. (ii) We discuss the potential cause of the negative correlation in Section 3.3, which theoretically supports the feasibility of predicting accuracy from distribution-related statistics. (iii) The newly included CIFAR-10 setup further validates the effectiveness of the proposed regression models (Section 4.2). (iv) In Section 5, we show our regression models are also effective under various train/test combinations. (v) We provide an ablation study of neural network regression in Section 4.2, which indicates that using comprehensive statistics to represent a dataset is important for the dataset-level regression. (vi) We investigate the effect of the meta-dataset construction, study the impact of image transformation, and discuss the generalization ability as well

TABLE 1
Analogies Between Standard Image Classification Terms and Their AutoEval Equivalents

	Image Classification	AutoEval
Sample	Image	Dataset (sample set)
Label	Sample class ground truth	Accuracy of model on sample set
Training Set	Set of labeled images	Set of synthetic labeled sample sets (meta set)
Test Set	Set of unseen labeled images	Set of unseen labeled real-world datasets
Loss	Class cross-entropy	Predicted accuracy RMSE
Task	Classify images	Predict accuracy of model from statistics of dataset

The analogy shows that the image classification is an image-based task, while the AutoEval problem in this work is dataset-based.

1. In what follows, we term meta-dataset and meta set interchangeably.

Fig. 2. Examples of generated sample sets in the meta-dataset (**left** : COCO setup; **right** : MNIST setup). The seed set (labeled) is from the same distribution with the original training set; they share the same classes but do not have image overlap. The sample sets are generated from the seed set by using *background replacement* and *image transformations* . The sample sets exhibit distinct data distributions, and yet inherit the foreground objects from the seed set, and thus they are fully labeled.

as the limitations of the regression models in Section 6.2. We show the diversity of the meta-dataset is a key factor in learning generalizable regression models.

2 Problem Definition

We first define a labeled test set, $D^l = \{(\boldsymbol{x}_i, y_i)\}$ where $i \in [1, \ldots, M]$, $\boldsymbol{x}_i$ is an image, y_i is its class label, and M is the number of images. Consider a training dataset $\mathcal{D}_{ori}$ from an underlying distribution $\mathcal{S}$, we train a classifier $f_{\boldsymbol{\theta}} : \boldsymbol{x_i} \rightarrow \hat{y}_i$, which is parameterized by $\boldsymbol{\theta}$ and maps an image $\boldsymbol{x_i}$ to its predicted class $\hat{y}_i$. Given D^l, a standard way to evaluate the classifier is comparing the class predictions $\hat{y}_i$ with the ground truth y_i to obtain accuracy

$$a_{standard} = \frac{\sum_{i=1}^{M} \mathbb{I}[\hat{y}_i == y_i]}{M}, \tag{1}$$

where $\mathbb{I}[\cdot]$ is an indicator function returning 1 if the argument is true and 0 otherwise.

Automatic Model Evaluation (AutoEval). Given classifier $f_{\boldsymbol{\theta}}$ and an unlabeled test set $\mathcal{D}^u = \{\boldsymbol{x}_i\}$ for $i \in [1, \ldots, M]$, we aim to learn an accuracy predictor $\mathcal{A} : (f_{\boldsymbol{\theta}}, \mathcal{D}^u) \rightarrow a$, which outputs an estimated performance $a \in [0, 1]$ on this test set

$$a_{auto} = \mathcal{A}(f_{\boldsymbol{\theta}}, \mathcal{D}^u). \tag{2}$$

In image classification, $\mathcal{D}_{ori}$ and $\mathcal{D}^u$ share the same label space.

3 Methodology

AutoEval is a dataset-level task (shown in Table 1), where we need to consider the overall dataset characteristics. Under this consideration, we formulate AutoEval as a dataset-level *regression problem*. In the following, we first report a strong negative linear correlation between distribution shift and classifier accuracy. This new finding demonstrates that it is feasible to predict classifier accuracy from distribution-related statistics. We then propose two regression methods. At last, we provide a potential cause of negative linear correlation.

3.1 Correlation Study on the Meta-Dataset

3.1.1 Meta-Dataset Construction

Motivated by the implications in domain adaptation, i.e, distribution shift degrades accuracy [8], we empirically measure the relationship between classifier accuracy and distribution shift on meta-dataset: a dataset comprised of many datasets (sample sets).

The meta-dataset is expected to (a) contain many datasets from diverse data distributions, (b) have category labels for each dataset, and (c) have the same label space with the training set. Using the existing real-world datasets cannot construct a meta-dataset that satisfies these requirements, so we resort to data synthesis.

We synthesize sample sets from a single seed dataset. The seed $\mathcal{D}_s$ is sampled from source domain $\mathcal{S}$, and thus has the same distribution as the training set $\mathcal{D}_{ori}$. Given $\mathcal{D}_s$, we use various photometric and geometric transformations and obtain N different sample sets $\mathcal{D}_j, j = 1, \ldots, N$. It is worthy of noting that the sample set inherits all the image labels from the seed set $\mathcal{D}_s$ and thus they are fully labeled. We build up the following three setups.

MNIST Setup. MNIST contains 50,000 training images and 10,000 test images from 10 classes. We train LeNet-5 on the training set and use the test set as the seed. Since MNIST images are binary, the foreground can be easily separated from the background. We replace the background of each MNIST image with a random patch of an image sampled from COCO training set. This practice generates 2,000 and 1,000 sample sets for the training and the validation, respectively.

CIFAR-10 Setup. We use ResNet-44 to train a 10-way classifier on 50,000 training images. To synthesize sample sets, we apply image transformations on 10,000 test images. When creating each sample set, we randomly select three transformations from {*Autocontrast, Brightness, Color, Color-Solarize, Contrast, Rotation, Sharpness, TranslateX/Y*}. In the experiment, we use another four transformations (*Cutout, ColorTemp, Equalize, and ShearX/Y*) to test the robustness of our regression methods. Then, we apply the selected transformations with random magnitudes on test images. This practice creates 1,000 sample sets, of which we use 800 and 200 for training and validation, respectively.

COCO Setup. We choose 12 classes (i.e., aeroplane, bike, bird, boat, bottle, bus, car, dog, horse, monitor, motorbike, and person) from COCO dataset, and process the images following [9] for classification task. Specifically, the images from original training and validation sets are used for training and testing in our setup. To synthesize sample sets, we first replace the background of objects with a random patch of an image from COCO test set. We then randomly select three transformations to introduce more visual changes. Lastly, we generate 1,000 and 600 sample sets for the training and validation, respectively.

We show some examples of sample sets of MNIST and COCO setups in Fig. 2, where background replacement and

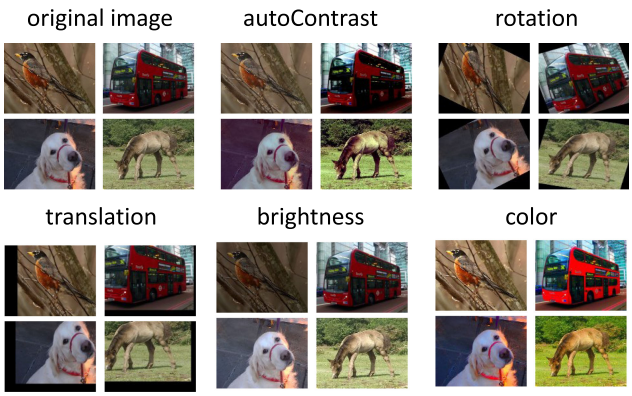

Fig. 3. Visual examples of transformations used for constructing the meta-dataset. Here we show autoContrast, rotation, translation, brightness, and color. For other used transformations, we refer readers to [10].

various visual changes can be observed. The sample sets inherit the foreground objects from the seed set, and thus they are fully labeled. Examples of some transformations are also shown in Fig. 3. In the appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2021.3136244, we present detailed transformation hyper-parameters and more visual examples of the meta-dataset.

3.1.2 Observation: Negative Linear Correlation Between Test Accuracy and Distribution Shift

Given a meta set and a classifier trained on the training dataset $\mathcal{D}_{ori}$, we now measure the statistical relationship between classifier accuracy and distribution shift.

The distribution shift can be represented by first- and second-order statistics of the output image feature representations [11], [12], [13]. In this work, we use the Fréchet distance (FD) to measure the distribution shift. Other measurements can also be used, such as MMD [13]. The Fréchet distance (FD) is defined as

$$\begin{aligned}\mathrm{FD}(\mathcal{D}_{ori}, \mathcal{D}) = &\|\boldsymbol{\mu}_{ori} - \boldsymbol{\mu}\|_2^2 \\ &+ Tr(\boldsymbol{\Sigma}_{ori} + \boldsymbol{\Sigma} - 2(\boldsymbol{\Sigma}_{ori}\boldsymbol{\Sigma})^{\frac{1}{2}}),\end{aligned} \tag{3}$$

where $\boldsymbol{\mu}_{ori}$ and $\boldsymbol{\mu}$ are the mean feature vectors of $\mathcal{D}_{ori}$ and $\mathcal{D}$, respectively. $\boldsymbol{\Sigma}_{ori}$ and $\boldsymbol{\Sigma}$ are the covariance matrices of $\mathcal{D}_{ori}$ and $\mathcal{D}$, respectively. They are calculated from the image features (i.e., activations in the penultimate of the classifier) on $\mathcal{D}_{ori}$ and $\mathcal{D}$, respectively, which are extracted using the classifier $f_{\boldsymbol{\theta}}$ trained on $\mathcal{D}_{ori}$. We use LeNet-5, ResNet-44, and ResNet-50 as the classifiers on MNIST, CIFAR-10, and COCO setup, respectively.

Strong Negative Linear Correlation. In Fig. 4, we plot accuracy against the distribution shift on the three setups (MNIST, COCO, and CIFAR-10). Across these setups, we consistently observe a very strong negative and linear correlation between accuracy and distribution shift: the Pearson's Correlation (r) is less than -0.88. That is, the classifier tends to have a low accuracy on the sample set which has a high distribution shift from the training set $\mathcal{D}_{ori}$. In Section 8, we provide a discussion of potential cause of such strong negative correlation.

3.2 Regression Model and Dataset Representation

3.2.1 Formulation

Motivation. The above correlation study validates the feasibility to estimate classifier accuracy from the distribution shift. In AutoEval, the training set and learned classifier are *given and fixed*, so the distribution shift is only determined by the test set. In other words, we can estimate classifier accuracy from the representation of the test set. Resulting from such inspiration, we formulate AutoEval as a *regression problem*: each sample is a dataset, represented by a feature and labeled by the classifier accuracy on the dataset itself.

Dataset-Level Regression. Given N sample sets in a meta-dataset, we denote the jth sample set $\mathcal{D}_j$ as $(\boldsymbol{f}_j, a_j)$, where $\boldsymbol{f}_j$ is the representation of $\mathcal{D}_j$, and $a_j \in [0, 1]$ is the recognition accuracy of classifier $f_{\boldsymbol{\theta}}$ on $\mathcal{D}_j$. We aim to learn a regression model (accuracy predictor), written as

$$a_j = \mathcal{A}(\boldsymbol{f}_j). \tag{4}$$

We use a standard squared loss function for this model

$$\mathcal{L} = \frac{1}{N}\sum_{j=1}^{N}(\hat{a}_j - a_j)^2, \tag{5}$$

where $\hat{a}_j$ is the predicted accuracy of the jth sample set $\mathcal{D}_j$, and a_j is the ground truth classifier accuracy on $\mathcal{D}_j$.

During testing, we extract the dataset representation $\boldsymbol{f}^u$ of an unlabeled test set $\mathcal{D}^u$, and obtain estimated classification accuracy using $a = \mathcal{A}(\boldsymbol{f}^u)$. To learn regression models defined in Eqs. (4) and (5), we need to specify the design of 1) the regression method $\mathcal{A}$ and 2) the dataset representation $\boldsymbol{f}_j$.

3.2.2 Linear Regression

We first introduce a standard linear regression model

$$a_{linear} = \mathcal{A}_{linear}(\boldsymbol{f}) = w_1 f_{linear} + w_0, \tag{6}$$

where $f_{linear} \in \mathbb{R}$ is the representation of sample set $\mathcal{D}$, and $w_0, w_1 \in \mathbb{R}$ are parameters of this linear regression model. This model is based on the negative linear correlation between classifier accuracy and distribution shift. We define f_{linear} as the quantified domain shift between dataset $\mathcal{D}$ and the original training set $\mathcal{D}_{ori}$. Specifically, we use the Fréchet distance [14] (Eq. (3)) to measure the distribution shift.

3.2.3 Neural Network Regression

Besides linear regression, we introduce a neural network regression model, $a_{neural} = \mathcal{A}_{neural}(\boldsymbol{f}_{neural})$, which has the same formulation as Eq. (4). In practice, we use a simple fully connected neural network for regression. The model $\mathcal{A}_{neural}$ predicts the classifier accuracy a_{neural} from the statistics $\boldsymbol{f}_{neural}$ of a test dataset. In this work, we use both the first-order and second-order moments, i.e., mean vector and covariance matrix, which are commonly used to represent a distribution [12], [13], [15]. Moreover, we include a 1-dim FD score as auxiliary information to the representation. Compared with linear regression, the neural network regression uses a richer dataset representation, written as

$$\boldsymbol{f}_{neural} = [f_{linear}; \boldsymbol{\mu}; \boldsymbol{\sigma}], \tag{7}$$

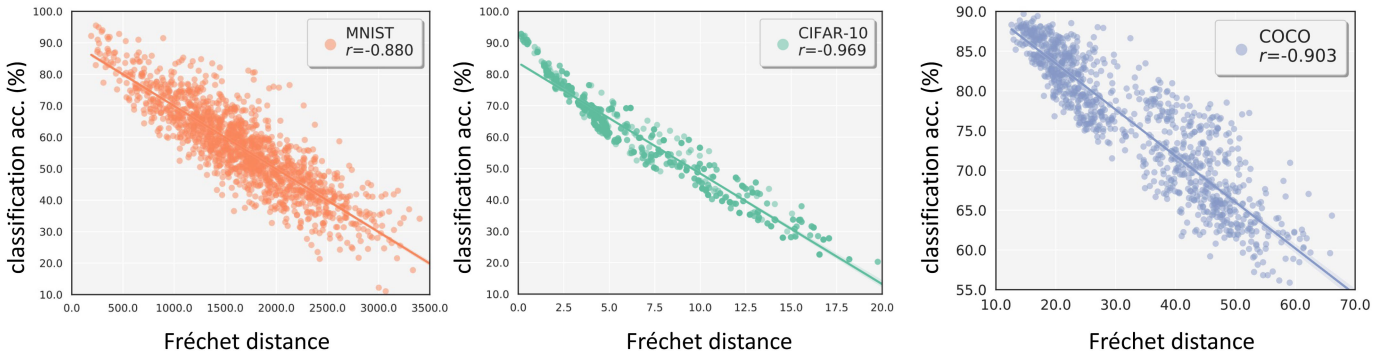

Fig. 4. Relationship between the distribution shift (Fréchet distance) and classifier accuracy on three meta-datasets of three setups. From left to right: MNIST setup with LeNet-5, CIFAR-10 setup with ResNet-44, and COCO setup with ResNet-50, respectively. Each point represents a sample set of the meta-dataset. Across three setups, we consistently observe the strong negative linear correlation (Pearson's Correlation $r < -0.88$) between distribution shift and classifier accuracy. This indicates that the classifier tends to gain a high accuracy on the sample set which has a low distribution shift with training set. The straight lines are calculated by linear regression.

where $f_{linear} \in \mathbb{R}$ is the Fréchet distance between $\mathcal{D}$ and $\mathcal{D}_{ori}$, $\boldsymbol{\sigma}$ is the weighted summation of $\boldsymbol{\Sigma}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are calculated in the same way as Eq. (3). Covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ is very high-dimensional, making training difficult. Dimension reduction is thus necessary. We calculate $\boldsymbol{\sigma}$ by taking a weighted summation of each row of $\boldsymbol{\Sigma}$, i.e., we choose to right multiply $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ with a $d \times 1$-dim vector. This vector is learned along with network parameters by the regression loss. Lastly, if the feature extracted from $f_{\boldsymbol{\theta}}$ is d-dim, the dimensionality of $\boldsymbol{f}_{neural}$ is $1 + 2d$.

3.3 Potential Cause of Negative Linear Correlation

The formulation of dataset-level regression is based on the strong negative linear correlation between distribution shift and classifier accuracy. To understand such correlation, we provide a potential explanation. Given labeled samples $(\boldsymbol{x}, y)$ from an underlying distribution $\mathcal{S}$, the overall objective in classification is to learn a classifier $f_{\boldsymbol{\theta}}$ that minimizes the population loss

$$L_{\mathcal{S}}(f_{\boldsymbol{\theta}}) = \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{S}}[\mathbb{I}[f_{\boldsymbol{\theta}}(\boldsymbol{x}) \neq y]] . \tag{8}$$

Since the distribution $\mathcal{S}$ is unknown, we instead evaluate the classifier on a labeled test set $\mathcal{D}^{\prime}$ drawn from the distribution $\mathcal{S}$

$$L_{\mathcal{D}^{\prime}}(f_{\boldsymbol{\theta}}) = \frac{1}{|\mathcal{D}^{\prime}|}\sum_{(\boldsymbol{x},y)\in\mathcal{D}^{\prime}} \mathbb{I}[f_{\boldsymbol{\theta}}(\boldsymbol{x}) \neq y] . \tag{9}$$

The test error $L_{\mathcal{D}^{\prime}}(f_{\boldsymbol{\theta}})$ is used as a proxy for the population loss $L_{\mathcal{S}}(f_{\boldsymbol{\theta}})$. If a classifier $f_{\boldsymbol{\theta}}$ achieves a low-test error, we assume that it will perform similarly well on future and unseen samples from the same distribution $\mathcal{S}$ [16], [17].

Given a sample set $\mathcal{D}$ from a distribution $\mathcal{T}$, we calculate another test loss $L_{\mathcal{D}}(f_{\boldsymbol{\theta}})$. Classical theory [8], [18] for domain adaptation across different distributions typically relates classifier accuracy drop on a new test set to distribution discrepancy. Following the practice in [17], we decompose the *loss difference* (i.e., the accuracy drop) between $L_{\mathcal{D}}(f_{\boldsymbol{\theta}})$ and $L_{\mathcal{D}^{\prime}}(f_{\boldsymbol{\theta}})$ into three parts (dropping the dependence on $f_{\boldsymbol{\theta}}$ to simplify notation)

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

Generalization Gap. The third term $L_{\mathcal{T}} - L_{\mathcal{D}}$ is the standard generalization gap commonly studied in machine learning. It is determined solely by the random sampling error.

Adaptivity Gap. The first term $L_{\mathcal{D}^{\prime}} - L_{\mathcal{S}}$ is the adaptivity gap. It measures the extent to which using the information of test set $\mathcal{D}^{\prime}$ for training classifier $f_{\boldsymbol{\theta}}$ causes the test error $L_{\mathcal{D}^{\prime}}$ to underestimate the population loss $L_{\mathcal{S}}$. A typical example of leveraging test set information is using the test set to tune hyperparameters of $f_{\boldsymbol{\theta}}$. This may make $f_{\boldsymbol{\theta}}$ perform well only on specific examples in the test set, so that test error $L_{\mathcal{D}^{\prime}}$ cannot truly reflect the generalization error of classifier $f_{\boldsymbol{\theta}}$. If assuming $f_{\boldsymbol{\theta}}$ is independent of the test set $\mathcal{D}^{\prime}$, the adaptivity gap would have the same meaning as the generalization gap [17].

Distribution Gap. The term $L_{\mathcal{S}} - L_{\mathcal{T}}$ the distribution gap. It quantifies how much the change from the original distribution $\mathcal{S}$ to the new distribution $\mathcal{T}$ affects the model $f_{\boldsymbol{\theta}}$.

Based on the above decomposition, we now discuss how the loss difference (i.e., the accuracy drop) varies when the classifier is tested on different samples sets. Specifically, the adaptivity gap is only calculated on the source distribution and is the same for different sample sets, so it does not contribute to the change of the loss difference. Therefore, the generalization gap and distribution gap are the two factors that impact how the loss difference varies. In fact, according to classical works [8], [18] of domain adaptation which have theoretically characterized the accuracy of classifiers under distribution shift, the distribution gap is more likely to dominate the way the loss difference changes, compared with the generalization gap. Particularly, theory suggests that a small distribution difference leads to a high target distribution accuracy [4], [5], [6]. Therefore, the loss difference (i.e., the accuracy drop) increases in proportion to the distribution gap. This is the potential cause of negative linear correlation between the classifier accuracy and distribution shift (Fig. 4).

4 Experiment and Analysis

4.1 Experimental Settings

Real-World Datasets for Testing. This work is an early attempt to the AutoEval problem. To our knowledge, we can find

TABLE 2
Method Comparison in Predicting Classification Accuracy

Train Set	MNIST			CIFAR-10		COCO			
Unseen Test Set	SVHN	USPS	RMSE↓	CIFAR10.1	RMSE↓	Pascal	Caltech	ImageNet	RMSE↓
Ground-truth Accuracy	25.46	64.08	-	87.65	-	86.13	93.40	88.83	-
Predicted Score ($\tau_1 = 0.7$)	10.09	43.60	18.11	91.40	3.75	88.34	93.28	90.17	1.49
Predicted Score ($\tau_1 = 0.8$)	7.97	37.22	22.66	86.90	0.75	84.32	90.78	86.50	2.28
Predicted Score ($\tau_1 = 0.9$)	7.03	32.94	25.59	82.05	5.60	78.61	87.71	81.33	6.96
Entropy Score ($\tau_2 = 0.1$)	4.49	22.52	32.92	76.25	11.40	73.51	84.19	76.17	11.61
Entropy Score ($\tau_2 = 0.2$)	6.42	31.24	26.84	83.50	4.15	80.62	88.57	83.33	5.29
Entropy Score ($\tau_2 = 0.3$)	8.32	39.21	21.36	89.55	1.90	86.45	92.43	88.17	0.70
Energy Score [19]	2.23	26.91	30.99	89.75	1.60	82.30	89.42	85.00	3.88
Linear Regression	26.28	50.14	9.87	81.84	5.81	83.87	79.77	83.19	8.62
Neural Network Regression	27.52	64.11	1.46	86.82	0.83	87.76	89.39	91.82	3.04

We use three classification setups: MNIST, CIFAR-10, and COCO. We compare five methods: two intuitive solutions (predicted score based solution and entropy score based solution), energy score [19], linear regression, and neural network regression (Section 3.2). For each setup, we report the estimated classification accuracy (%) and ground-truth recognition accuracy (%). We use the RMSE (%) to evaluate estimation precision.

only a few real-world datasets that have different distributions while containing the same classes. Specifically, for the MNIST setup, the test sets are USPS [20] and SVHN [21], both with 10 classes; for the COCO setup, we use three datasets: PASCAL [22], Caltech [23], and ImageNet [1], all with the common 12 classes; and for the CIFAR-10 setup, we use CIFAR-10.1 [24] as the test set.

Metrics. AutoEval is to predict classifier accuracy on an unlabeled test set. To evaluate the performance of such predictions, we use root mean squared error (RMSE) and mean absolute error (MAE) as metrics. RMSE measures the average squared difference between the estimated accuracy and ground-truth accuracy. MAE measures the average magnitude of the errors. Small RMSE and MAE values correspond to good predictions and vice versa.

Baselines. We first present two intuitive solutions to the AutoEval problem, which are not learning based. They are motivated by the pseudo labeling strategy applied in many vision tasks [25], [26], [27], [28], [29], [30]. The basic assumption is: if a class prediction is made with a high softmax output value, it is likely to be correct. The intuitive two solutions are described below.

Prediction score based solution. Given an input image, we define its confidence score as the maximum softmax output. If the score is *greater than* a threshold $\tau_1 \in [0, 1]$, this image is regarded as being correctly classified.

Entropy score based solution. A score is calculated as the entropy of softmax outputs, normalized by $log(K)$, where K is the number of classes. Also, an image with a score *less than* a threshold $\tau_2 \in [0, 1]$ is regarded as correctly classified. Note that, it is challenging to choose optimal thresholds for the two solutions because we do not have manual labels of test set. Moreover, as to be shown in the experiment, both solutions are rather sensitive to the thresholds, so arbitrarily setting the thresholds would compromise the performance. Thus, in the experiment, we show prediction results of the two solutions under several manually chosen thresholds.

Furthermore, we introduce *energy score based method* [19] for comparison. It better distinguishes in- and out-of-distribution samples than the softmax score. If a test sample has a lower energy score than a threshold τ_3, it is considered to be correctly classified. Follow the practice in [19], [31], we select the threshold τ_3 on the validate set and applied it to unseen test sets. Specifically, the threshold is computed so that 95% of samples in validation set have energy scores lower than the threshold τ_3.

4.2 Classifier Accuracy Prediction

Two Intuitive Solutions are Threshold-Sensitive. Accuracy prediction results of the two intuitive solutions are presented in Table 2. For the predicted score based solution, under the COCO setup and threshold of $\tau = 0.7$, its estimation error is RMSE $= 1.49\%$ which seems good. However, its prediction quality drops significantly (from 1.49% to 6.96%) when using a high threshold of $\tau_1 = 0.9$. Under the CIFAR-10 setup, its estimation error increases from 0.75% to 5.60% when we increase the threshold τ_1 from 0.8 to 0.9. Moreover, its predictions are very poor under MNIST setup. With three values of τ_1, the RMSE is consistently high, i.e., 18.11%, 22.66% and 25.59%, respectively. For the entropy score-based solution, we also have similar observations with the predicted score-based solution across the three setups.

To further examine their sensitivity to thresholds, we plot the RMSE against various thresolds for the two solutions, and results are shown in Fig. 5. We observe that they can make good predictions when using optimal thresholds and that the optimal thresholds are different for three setups. We also confirm that the two solutions are threshold-sensitive.

It is infeasible to select the optimal threshold for the two intuitive solutions because (1) test labels are unavailable and (2) the test scenarios keep changing. This compromises the practicability of the two solutions. In comparison, our regression models do not depend on such a hyper-parameter and yields more stable estimations. That said, it would be interesting to address this drawback in the context of AutoEval. One possible way might be using metadata sets to learn a strategy to adaptively select appropriate thresholds for various test sets.

Regression Methods Obtain Promising Predictions. In the MNIST setup, the RMSE values of linear regression and neural network regression are 9.87% and 1.46%, respectively.

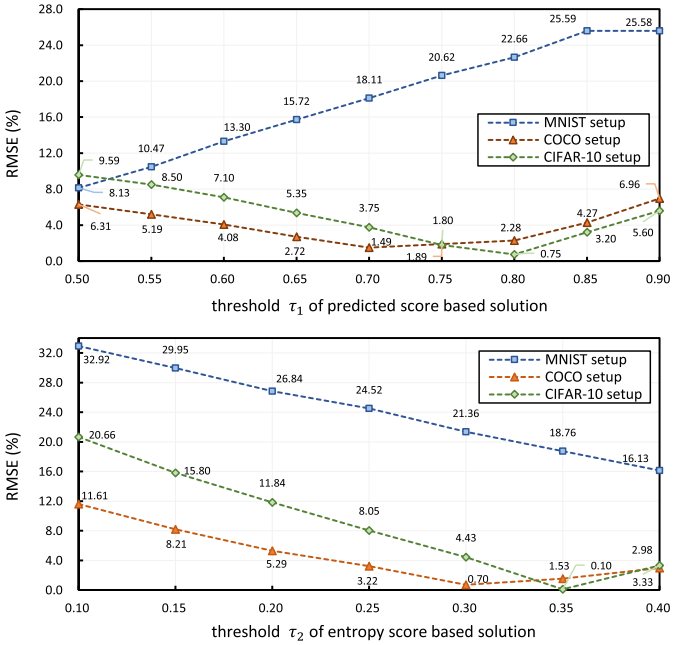

Fig. 5. RMSE (%) of the *predicted score-based solution (top)* and the *entropy score-based solution (bottom)* against various threshold values (τ_1) and (τ_2), respectively. We observe that both solutions can produce accurate estimations when using an optimal threshold value but are threshold-sensitive. Moreover, the optimal threshold values are different for the three setups.

These results are generally lower while being more stable than the two score-based solutions because they are not threshold-dependent. In CIFAR-10 setup, the neural network regression obtains good estimation on unseen test set CIFAR10.1 (0.83% in RMSE). Moreover, we observe that our regression models achieve reasonably good estimations on COCO setup. For example, network regression produces RMSE score of 3.04%.

In addition, we observe that the energy score method can make reasonably accurate predictions: it achieves 1.60% and 3.88% RMSE on CIFAR-10 and COCO setups, respectively. Compared with this method, our network regression is very competitive. For example, neural network regression produces 0.77%, 0.84%, and 29.53% lower RMSE on CIFAR-10, COCO and MNIST setups, respectively. Linear regression outperforms [19] under the MNIST setup, but not the others. These new results further demonstrate that our dataset-level regression methods, especially the neural network regression, are an effective and feasible solution to the AutoEval problem.

5 Analysis of Regression Models

Neural Network Regression is Generally Superior to Linear Regression. In Table 2, we compare the neural network regression with linear regression under the three setups and and observe that the former is more accurate (i.e., its prediction's variance is lower). Specifically, linear regression could fail in some scenarios such as Caltech in COCO setup. The poor performance of linear regression on Caltech is mainly due to three reasons. First, comparing with Pascal and ImageNet, Caltech looks much more different from COCO. In fact, if we plot it in Fig. 4, it lies far from the regressed line. As such, making predictions of such an "outlier" is itself a challenging task. Second, the linear regression model uses a very simple feature, the 1-dimensional FD score, to represent the difference between the training set and sample sets. In comparison, neural network regression uses a more complex feature, including the mean, co-variance and FD. Third, the linear regression model itself is very simple, comprising of only two learnable parameters, while neural network regression has much more parameters and higher capacity. Therefore, neural network regression can better capture the complexity of the sample sets and make more accurate predictions of the challenging Caltech dataset. In fact, the generated sample sets should be complex and diverse, which helps cover as many cases and distributions as possible. These complex and diverse sample sets couple well with the neural network regression model, yielding better performance than the linear regression model.

The above discussions indicate two potential ways to make regression models generalize well on unseen test sets: (1) building up a diverse meta-dataset (e.g., using possibly many various transformations), leveraging more comprehensive statistics to represent a dataset; (2) building stronger learning architectures. In the following, we study the dataset representation used by regression models and discuss the diversity of meta-dataset in Section 6.

Ablation Study on Dataset Representation of Network Regression. As illustrated in Eq. (7), neural network regression uses a richer dataset representation than linear regression. The representation used in network regression includes: FD, Mean, and Covariance. Here, we study the effect of dataset representation and report results in Fig. 7 under the COCO setup. Note that when only using FD as the representation, the neural network regression reduces to linear regression. We show that the estimation is more accurate when using more statistics for dataset representation. For example, only using FD and Mean as the representation achieves 1.84 % higher error than using all statistics. The results indicate that representing a dataset with comprehensive statistics is important for our dataset-level regression. We think it would be interesting to explore other potential dataset representations.

Robustness of Regression Models to Novel Visual Changes. To study whether our models can deal with unseen visual variations, we introduce synthesized datasets for testing. Specifically, we edit the three natural datasets (ImageNet, Pascal and Caltach) with two groups of transformations: *{Cutout, Shear}* and *{Equalize, ColorTemperature}*. These transformations are *not* used in meta-dataset creation. In Fig. 6, we observe that linear regression can make promising predictions on six out of nine datasets. However, it cannot achieve desirable predictions on two edited Caltech sets (e.g., its absolute error is 13.18 % on Caltech-A). Moreover, neural network generalizes well on these synthesized test sets. It consistently produce reasonably good predictions on nine test sets. The above analysis suggests that both regression models have ability to handle unseen variations, while network regression is more robust than linear regression.

Estimations for the Classifiers Trained on Different Datasets. To further validate the effectiveness of our regression methods, we apply them to predict the recognition accuracy of classifiers trained on different datasets. Results are presented

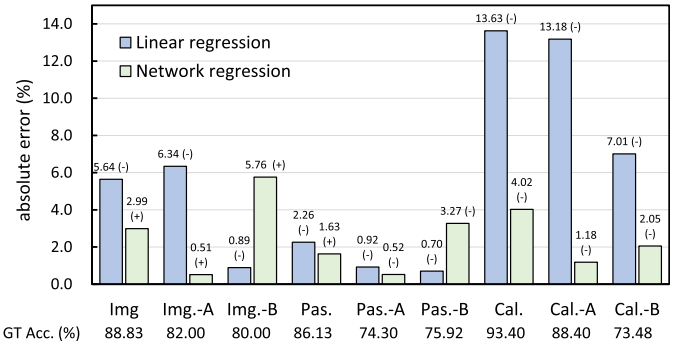

Fig. 6. Comparing linear regression and neural network regression when test data undergo *new* transformations. We use two groups of transformations: *{Cutout, Shear}* and *{Equalize, ColorTemperature}*; the corresponding transformed sets are denoted by "-A" and "-B". We report the absolute error (%) of predictions, and the ground truth accuracy is also shown beneath each dataset. (-) / (+) means the predicted accuracy is lower / higher than the ground-truth accuracy, respectively.

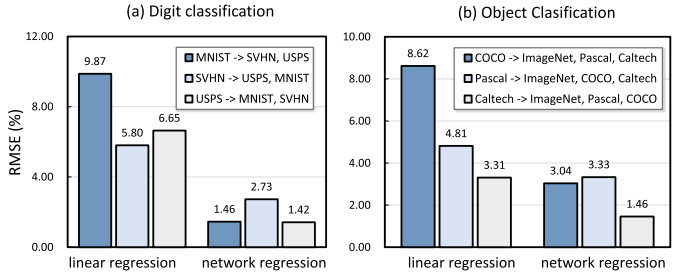

Fig. 8. Accuracy estimation error (RMSE, %) for the classifiers trained on different datasets. In the legends, datasets on the left of "→" are the training set, and those right to "→" are test sets. For example, in "SVHN → USPS, MNIST", we estimate accuracy of a classifier trained on SVHN and tested on MNIST and USPS. We perform experiments on two tasks: (a) digit classification (**left**) and (b) object classification (**right**). These results also validate the effectiveness of the porposed regression models trained for the AutoEval problem.

in Fig. 8. We observe our regression models make promising estimations. For example, for a classifier trained on SVHN and tested on USPS and MNIST (SVHN → USPS, MNIST), linear regression and network regression achieve 5.80% and 2.73% RMSE, respectively. For an object classifier trained on Pascal (Pascal → ImageNet, COCO, Caltech), our methods are also reasonably effective: linear regression and network regression give 4.81% and 3.33% RMSE in the estimated accuracy, respectively. The above experimental results show that dataset-level regression is indeed a feasible way to estimate classifier accuracy.

6 Analysis of Meta-Dataset

The synthesized meta-dataset is a crucial part for learning robust regression models. The underlying assumption is that the meta-dataset can cover most of the variations in terms of distribution in the real-world scenarios. In this section, we first discuss the diversity of the meta-dataset from three aspects, i.e., meta set size and sample set size. We then analyze the impact of meta-dataset construction (i.e., image transformation and background change) on regression models. Lastly, we discuss the generalization ability of regression models.

6.1 Diversity of Meta-Dataset

Meta Set Size. A meta set contains training samples/datasets for regression model training. We first study the impact of meta set size on the regression methods, and results are shown in Fig. 9 (first row). We observe that results of linear regression are relatively stable with different meta set sizes. Good performance is achieved even with only 50 sample sets. This is because linear regression only has two parameters (Eq. (6)), which can be learned with relatively few samples [32]. In comparison, neural work cannot achieve results that are as good when the number of sample sets is small. When provided with adequate sample sets, the neural network learns effectively from rich and diverse sample datasets and tops linear regression.

Sample Set Size. By default, the number of images in each sample set is equal to that of seed $\mathcal{D}_s$. Here, we study the influence of the sample set size on the regression methods. Specifically, we use 1,000 as the size of the meta-dataset and vary the sample set size. We plot the absolute error against different sample size in the second row of Fig. 9. We clearly observe that linear regression is relatively stable under different sample set sizes, while performance of the neural network regression generally exhibits a decreasing trend. For example, on the ImageNet test set, when the sample set size increases from 50 to 6,000, the absolute error of linear regression keeps at around 6.0%. In comparison, absolute error of neural network regression drops by nearly 6.0%. In fact, neural network regression needs more images in each sample set for training. More images in each sample set make the distribute-related representations more accurate and thus are beneficial for learning the network regression model.

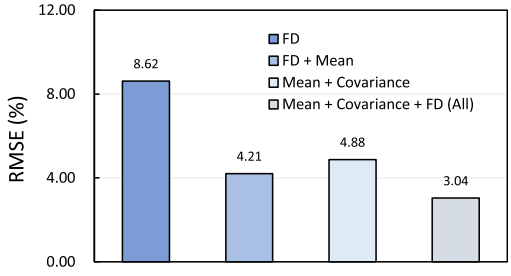

Fig. 7. Ablation study on dataset representation of neural network regression. When only using FD as the representation, the neural network regression reduces to linear regression. We observe that using more statistics to represent a dataset leads to more desirable estimations. The results are obtained under the COCO setup.

6.2 Study on Meta-Dataset Construction

In our work, we use background change and image transformation to construct the meta-dataset. Both techniques can introduce various visual variations. Here, we study the impact of the two techniques on the diversity of meta set. By default, the meta set is constructed by using background changes and three random transformations. We denote it as Meta set-Def. We include another four meta sets. They are: meta set-A, which is constructed by using background change only; Meta set-B, which is constructed by using three random transformations; Meta set-C, which is constructed by using background change and only one random transformation; and Meta set-D, which is constructed by using *six* random transformations.

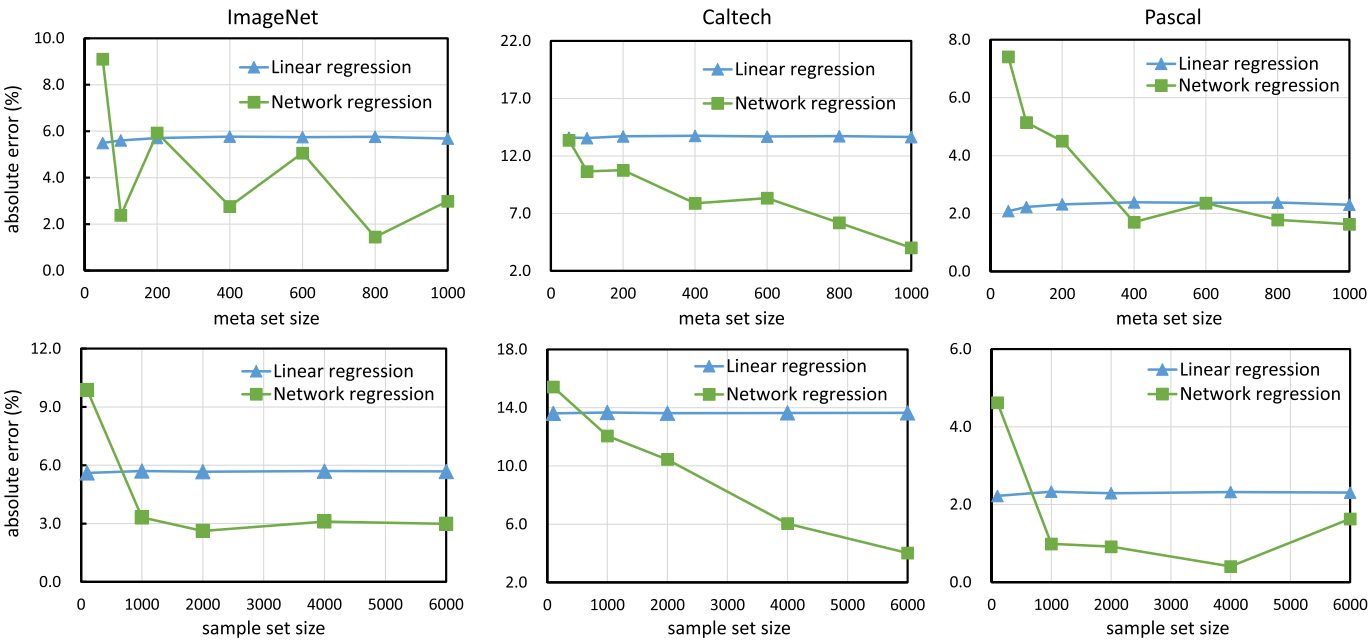

Fig. 9. The impact of meta set size (first row) and sample set size (second row) on the performance of regression methods. We report the absolute errors (%) between estimated classifier accuracy and the ground-truth accuracy. We observe that linear regression is relatively stable with different sample set and meta set size. In comparison, neural network needs more and larger sample sets for training.

We learn regression models on five meta sets and then compare them in Fig. 10. Results are obtained under the COCO setup. First, we observe that linear regression models trained on five meta sets produce similar results on ImageNet and Pascal. Moreover, they fail to make desirable predictions: their absolute errors are higher than 10%. Second, network work regression achieves more accurate estimations when using a more diverse meta set for training. For example, when trained on Meta set-A, its absolute error is higher than 10% on Caltech. However, when trained on meta set-D, its error drops to around 3.59% on Caltech. Third, we observed that the network regression models trained on meta set-D and meta set-Def produce comparable predictions. This shows that using many transformations introduces plenty of visual changes and could avoid using background changes (which requires mask annotations). Furthermore, when more visual variations are introduced in the meta set (e.g., Meta set-D and Meta set-Def), the network regression model learns better, and its advantage over the linear regression model becomes more obvious.

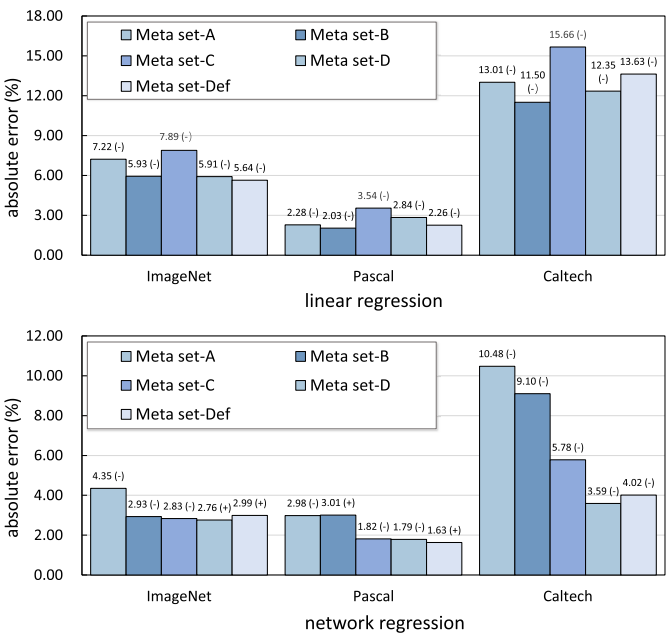

Fig. 10. Absolute errors (%) of linear regression (**top**) and network regression (**bottom**) trained on five meta sets. They are: (1) Meta set A, which is constructed by using background change; (2) Meta set B, which is constructed by conducting three random transformations; (3) Meta set C, which is constructed by using background change and one random transformation; (4) Meta set D, which is constructed by using six random transformations; (5) Meta set-Def, which is constructed by using background change and three random image transformations. By default, we use Meta set-Def. The results are under COCO setup.

Impact of Image Transformations for Constructing Meta Sets . Here, we conduct experiments to study the impact of transformations on the effectiveness of meta-datasets. Specifically, we use two groups of new transformations to construct Meta set-GroupA and Meta set-GroupB. The transformations are provided in [33]. The first group of transformations are: *{Grayscale, ElasticTransformation, PiecewiseAffine, Invert, FilterBlur, EnhanceBrightness, Fog, AdditiveGaussianNoise}* and the second group are: *{LinearContrast, Rain, JpegCompression, FilterDetail, EnhanceSharpness, MultiplyHue, Emboss, AddToSaturation}* . Note that, transformations in both groups introduce uncommon visual changes that are less likely to appear in the testing environments we used in the experiment.

In Fig. 11, we report the results of regression models trained on three different meta-datasets: Meta Set-Def, Meta set-GroupA and Meta set-GroupB. We have two observations. (1) Although the transformations used in Meta set-GroupA and Meta set-GroupB introduce relatively rare visual changes, they are helpful for regression model to learn the underlying relationship between distribution shift and classifier accuracy. More specifically, regression models trained on both meta-datatsets make promising predictions.

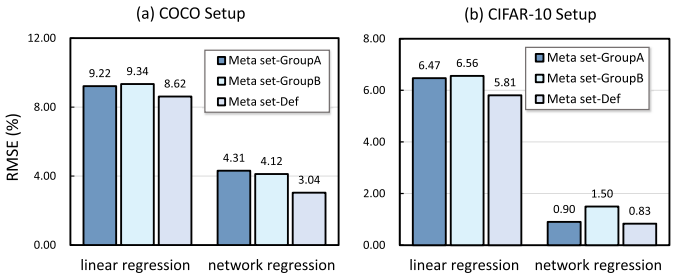

Fig. 11. Effect of different image transformations for meta set generation. We compare linear regression and network regression using the mean squared error (RMSE, %). we construct meta sets using three groups of image transformations. They are: (1) Meta set-Def, which randomly selects three transformations from *{Autocontrast, Brightness, Color, ColorSolarize, Rotation, Sharpness, TranslateX/Y}* ; (2) Meta set-GroupA, which randomly selects three transformations from *{Grayscale, ElasticTransformation, PiecewiseAffine, Invert, FilterBlur, EnhanceBrightness, Fog, AdditiveGaussianNoise}* ; and (3) Meta set-GroupB, which randomly selects three transformations from *{LinearContrast, Rain, JpegCompression, FilterDetail, EnhanceSharpness, MultiplyHue, Emboss, AddToSaturation}* . By default, we use Meta set-Def in this paper. The results are under COCO setup (**left**) and CIFAR-10 setup (**right**). We can observe decent and similar performance produced by using the three meta sets, suggesting that our regression methods are robust under various but reasonable meta sets.

For example, trained on Meta set-GroupA, linear regression and network regression achieve 9.22% and 4.31% RMSE on COCO setup, respectively. (2) We find that regression models trained on Meta Set-Def achieve more accurate estimations. This is because the transformations used in Meta Set-Def introduce more common visual changes such as rotation, translation, and light variation, which are often encountered in the testing environments. This is aligned with our suggestion that the meta-dataset should reflect most of the visual changes (or their equivalences) in the testing environments. For the application scenarios, if we know some prior patterns or conditions in the testing environments, we could customize transformations to be included in the meta-dataset to improve the effectiveness of regression models.

Discussion on the Generalization Ability of the Regression Models. The meta-dataset is expected to contain varied test distributions. However, in real-world situations, we might meet some corner cases and novel distributions that are not contained in the meta-dataset. In these cases, the estimations might not be accurate. To further validate the generalization ability of the network regression model, we newly use three natural image test sets with novel distribution shifts: ImageNet-V2 [17], a reproduction of the ImageNet test set; ImageNet-VID-Robust [34], which is collected from videos; and ImageNet-Adv [35], which is assembled by the images that are misclassified by ResNet-50. The absolute error of network regression is 5.6% and 4.92% on ImageNet-V2 and ImageNet-VID-Robust, respectively. However, on ImageNet-Adv with adversarial filter shifts, the estimation error is 22.5%. The results suggest that the meta-dataset can simulate the distributions of ImageNet-V2 and ImageNet-VID-Robust but fails to cover the adversarial case of ImageNet-Adv. Note that, most classifiers achieve low classification accuracy on ImageNet-Adv. [35]. A possible way to cope with the failure is to specially design samples sets that have such hard samples.

Furthermore, if some unlabeled test images are given, like the settings of domain adaptation [8] and semi-supervised learning [36], we could customize sample sets that are similar to the testing environments. This potentially improve the effectiveness of meta-dataset. For example, we can synthesize sample sets through a graphics engine [37], [38] or style transfer [39], [40] to approximate the style of the test images. Moreover, we could search transformation policies [10] to create sample sets that have similar distributions to the test set.

7 Related Work

Model Generalization Prediction. There are some works develop complexity measurements on training sets and model parameters to predict generalization error [41], [41], [42], [43], [44], [45]. Corneanu *et al.*, [43] use the persistent topology measures to predict the performance gap between training and testing error, even without the need of any testing samples. Jiang *et al.*, [41] introduce a measurement of layer-wise margin distributions for generalization ability. Neyshabur *et al.*, [44] develop bounds on the generalization gap based on the product of norms of the weights across layers. Moreover, the agreement score of several classifies can be used for estimation [46], [47], [48], [49], [50].

Instead of studying the complexity measurements, we use statistics related to test distributions for accuracy estimation. Moreover, we are concerned with various test distributions that have domain gaps with the training one, while the aforementioned works typically assume a fixed test set without a domain gap.

Out-of-Distribution Generalization. Machine learning models need to be able to generalize from training distribution to new environments under dataset shift [51], [52]. To study the problem of out-of-distribution generalization, several benchmarks are proposed [35], [52], [53], [54], [55], [56]. For instance, Hendrycks *et al.* , [53] propose a dataset to evaluate the corruption and perturbation robustness of models. Gulrajani *et al.*, [56] introduce DomainBed to evaluate recent domain generalization methods. To improve out-of-distribution generalization, some propose to learn robust features [40], [57], [58], [59], [60], [61], [62], with examples including adversarial domain augmentation [57], [58], [63] and inter-domain gradient matching [61], [62]. Under the out-of-distribution environments, instead of aiming to improve model generalization ability, our method makes it feasible to estimate model performance without ground truths.

Out-of-Distribution Detection. This task [25], [29], [31], [64], [65], [66], [67], [68], [69] considers the distribution of test samples. Specifically, this task aims to detect abnormal test samples that follow a distribution different from the training distribution (e.g., samples that do not contain any of the classes modeled in the train distribution). This task has been studied from different views, such as anomaly detection [70], [71], [72], open-set recognition [73], uncertainty [68] and rejection [31], [69], [74]. For example, Hendrycks *et al.*, [25] use probabilities output from a softmax classifier as indicator to find out-of-distribution samples. Liu *et al.*, [19] propose to use energy score to detect out-of-distribution samples. The above methods have the potential to improve AutoEval performance in the open world. Specifically, they can detect and reject abnormal and unknown class samples. This ensures that the overall statistics of the

test samples are accurate, which helps the regression model in the open set scenario.

Unsupervised Domain Adaptation. Our work also relates to unsupervised domain adaptation, where the goal is to use labeled source samples and unlabeled target samples to learn a model that can generalize well on the target dataset [6], [75], [76], [77], [78].

Many moment matching schemes have been studied for this task [6], [11], [11], [12], [75], [79], [80]. Long *et al.* [75] and Tzeng *et al.* [6] utilize the maximum mean discrepancy (MMD) metric [13] to learn a shared feature representation. Peng *et al.* [12] propose to address multi-source domain adaptation by matching feature moments. In this work, we empirically quantify the relationship between classifier accuracy and distribution shift. We further validate that dataset-level statistics can be used to estimate classifier performance on unlabeled test sets.

Dataset-Level Analysis. While most computer vision tasks can be described as some forms of image-level analysis, few literature explores the dataset-level analysis. Some work has focused on optimizing dataset construction by selecting a subset of images for pretraining for a specific downstream task [81], [82], [83]. For example, Yan *et al.* [83] introduce a large-scale search engine to find the most useful transfer learning data for the target task. Another avenue of research aims to automatically synthesize a labeled training set for downstream tasks [37], [38], [84]. They propose to learn to edit the parameters of graphic engines to minimize the content gap between synthetic training data and real test data. Our work is related in that we also study inter-dataset relationships, but have a distinct purpose. Specifically, this work aims to estimate classifier performance on an unlabeled dataset from the perspective of the distribution difference.

8 Conclusion and Perspectives

This paper investigates the problem of predicting classifier accuracy on test sets without ground truth labels. It has the potential to yield significant practical value, such as predicting system failure in unseen real-world environments. Importantly, this task requires us to derive similarities and representations on the dataset level, which is significantly different from common image-level problems. We make some tentative attempts by devising two regression models which directly estimate classifier accuracy based on overall distribution statistics. We build a dataset of datasets (meta-dataset) to train the regression models. We show that the synthetic meta-dataset can cover a good range of data distributions and benefit estimations of regression models on real-world test sets. For the remainder of this section, we discuss the limitations, potential, and interesting aspects of AutoEval.

Application Scope and Limitation. Our regression models assume that variations in the real-world cases can be approximated by the image transformations in meta set. Given diverse sample sets, our models learn to make promising predictions for novel test scenarios. However, as discussed in Section 6.2, if the test images exhibit very special patterns or conditions, our models might not be able to work well. On a related extreme case, the test dataset might only contain ambiguous and adversarial samples (e.g., ImageNet-Adv [35]), meaning that the test accuracy could be as poor as random. Such cases are not included in meta-dataset. Potentially, the above this might be alleviated by including such cases into the meta-dataset with a specific dataset design.

Furthermore, in this work, we mainly consider close-set classification in which training and test sets share the same label space. In practice, we might encounter images from unseen classes, i.e., the open-world scenario. These novel images might have a negative impact on the overall statistics of test set and thus compromise the system performance. To this problem, a feasible solution is first using out-of-distribution detection [19], [66] or outlier detection [69], [71] techniques to detect and reject such novel images, and then estimating classifier accuracy using the statistics of the remaining images. In fact, studying the AutoEval under the open-world setting is a very interesting research direction.

Dataset Representation. Our work relates to an interesting research problem: how to represent a dataset? This problem is more challenging than describing a single image because a dataset contains more information. This work uses distribution-related feature statistics (mean and covariance) to characterize a classification dataset. In the light of results in Fig. 7, we think comprehensively representing a dataset is essential to our regression modes. Namely, it is a potential research direction to explore other representations for better representing a dataset so as to improve performance of regression models. Furthermore, it would be interesting to study the dataset representation in other tasks (e.g., object detection and semantic segmentation), where global feature statistics might be unsuitable to represent a dataset.

Acknowledgments

The authors would like to thank the Associate Editor and all anonymous reviewers for their constructive comments in improving this paper.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[2] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[4] Y. Ganin and V. S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.

[5] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2962–2971.

[6] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014, *arXiv:1412.3474*.

[7] W. Deng and L. Zheng, "Are labels always necessary for classifier accuracy evaluation?," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15 069–15 078.

[8] S. Ben-David , J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1, pp. 151–175, 2010.

[9] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "VisDA: The visual domain adaptation challenge," 2017, *arXiv: 1710.06924*.

[10] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 113–123.

[11] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 2058–2065.
[12] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1406–1415.
[13] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Proc. Conf. Neural Inf. Process. Syst.*, 2006, pp. 513–520.
[14] D. Dowson and B. Landau, "The fréchet distance between multivariate normal distributions," *J. Multivariate Anal.*, vol. 12, no. 3, pp. 450–455, 1982.
[15] M. Fréchet, "Sur la distance de deux lois de probabilité," *COmptes Rendus Hebdomadaires des Seances de l Acad. des Sci.*, vol. 244, no. 6, pp. 689–692, 1957.
[16] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.
[17] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5389–5400.
[18] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. Conf. Learn. Theory*, 2009, pp. 1–11.
[19] W. Liu, X. Wang, J. D. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, pp. 21464–21475.
[20] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
[21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," Tech. Rep., 2011. [Online]. Available: https://research.google/pubs/pub37648/
[22] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.
[23] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. 7694, 2007. [Online]. Available: http://authors.library.caltech.edu/7694
[24] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do cifar-10 classifiers generalize to cifar-10?," 2018, *arXiv:1806.00451*.
[25] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=Hkg4TI9xl
[26] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3801–3809.
[27] F. Ma, D. Meng, Q. Xie, Z. Li, and X. Dong, "Self-paced co-training," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2275–2284.
[28] Z. Zheng and Y. Yang, "Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, no. 4, pp. 1106–1120, 2021.
[29] A. Vyas, N. Jammalamadaka, X. Zhu, D. Das, B. Kaul, and T. L. Willke, "Out-of-distribution detection using an ensemble of self supervised leave-out classifiers," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 550–564.
[30] B. Liu, Y. Ding, J. Jiao, X. Ji, and Q. Ye, "Anti-aliasing semantic reconstruction for few-shot semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9747–9756.
[31] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proc. Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1VGkIxRZ
[32] P. J. Huber, *Robust Statistics*. Hoboken, NJ. USA: Wiley, 2004, vol. 523.
[33] A. B. Jung *et al.*, "imgaug," Accessed: Aug. 20, 2021. [Online]. Available: https://github.com/aleju/imgaug, 2020
[34] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5389–5400.
[35] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, "Natural adversarial examples," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15 262–15 271.
[36] Y. Grandvalet, Y. Bengio *et al.*, "Semi-supervised learning by entropy minimization," in *Proc. Annu. Conf. CAP*, 2005, pp. 281–296.
[37] A. Kar *et al.*, "Meta-sim: Learning to generate synthetic datasets," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 4551–4560.
[38] Y. Yao, L. Zheng, X. Yang, M. Naphade, and T. Gedeon, "Simulating content consistent vehicle datasets with attribute descent," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 775–791.
[39] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2414–2423.
[40] K. Zhou, Y. Yang, T. Hospedales, and T. Xiang, "Learning to generate novel domains for domain generalization," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 561–578. [Online]. Available: https://openreview.net/forum?id=HJlQfnCqKX
[41] Y. Jiang, D. Krishnan, H. Mobahi, and S. Bengio, "Predicting the generalization gap in deep networks with margin distributions," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HJlQfnCqKX
[42] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, "Stronger generalization bounds for deep nets via a compression approach," 2018, *arXiv:1802.05296*.
[43] C. A. Corneanu, S. Escalera, and A. M. Martinez, "Computing the testing error without a testing set," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2677–2685.
[44] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, "Exploring generalization in deep learning," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5947–5956.
[45] T. Unterthiner, D. Keysers, S. Gelly, O. Bousquet, and I. Tolstikhin, "Predicting neural network accuracy from weights," 2020, *arXiv:2002.11448*.
[46] O. Madani, D. Pennock, and G. Flake, "Co-validation: Using model disagreement on unlabeled data to validate classification algorithms," in *Proc. Conf. Neural Inf. Process. Syst.*, 2004, vol. 17, pp. 873–880.
[47] E. A. Platanios, A. Dubey, and T. Mitchell, "Estimating accuracy from unlabeled data: A Bayesian approach," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1416–1425.
[48] E. Platanios, H. Poon, T. M. Mitchell, and E. J. Horvitz, "Estimating accuracy from unlabeled data: A probabilistic logic approach," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4361–4370.
[49] P. Donmez, G. Lebanon, and K. Balasubramanian, "Unsupervised supervised learning i: Estimating classification and regression errors without labels," *J. Mach. Learn. Res.*, vol. 11, no. 4, pp. 1323–1351, 2010.
[50] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio, "Fantastic generalization measures and where to find them," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=SJgIPJBFvH
[51] J. Djolonga *et al.*, "On robustness and transferability of convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16458–16468.
[52] P. W. Koh *et al.*, "Wilds: A benchmark of in-the-wild distribution shifts," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 5637–5664.
[53] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HJz6tiCqYm
[54] S. Santurkar, D. Tsipras, and A. Madry, "Breeds: Benchmarks for subpopulation shift," in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=mQPBmvyAuk
[55] A. Barbu *et al.*, "Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models," in *Proc. Conf. Neural Inf. Process. Syst.*, 2019, pp. 9448–9458.
[56] I. Gulrajani and D. Lopez-Paz , "In search of lost domain generalization," in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=lQdXeXDoWtI
[57] R. Volpi, H. Namkoong, O. Sener, J. Duchi, V. Murino, and S. Savarese, "Generalizing to unseen domains via adversarial data augmentation," in *Proc. Conf. Neural Inf. Process. Syst.*, 2018, pp. 5339–5349.
[58] L. Zhao, T. Liu, X. Peng, and D. Metaxas, "Maximum-entropy adversarial data augmentation for improved generalization and robustness," in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, pp. 14435–14447.
[59] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang, "Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization," in *Proc. Int. Conf. Learn. Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=ryxGuJrFvS
[60] E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, and C. Finn, "Just train twice: Improving group robustness without training group information," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 6781–6792.

[61] L. Mansilla, R. Echeveste, D. H. Milone, and E. Ferrante, "Domain generalization via gradient surgery," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 6630–6638.
[62] Y. Shi, J. Seely, P. H. Torr, N. Siddharth, A. Hannun, N. Usunier, and G. Synnaeve, "Gradient matching for domain generalization," 2021, *arXiv:2104.09937*.
[63] F. Qiao and X. Peng, "Uncertainty-guided model generalization to unseen domains," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6790–6800.
[64] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1321–1330.
[65] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," 2018 *arXiv:1802.04865*.
[66] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proc. Conf. Neural Inf. Process. Syst.*, 2018, pp. 7167–7177.
[67] K. Lee, K. Lee, H. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *Proc. Int. Conf. Learn. Representations*, 2018.
[68] Y. Ovadia *et al.*, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Proc. Conf. Neural Inf. Process. Syst.*, 2019, pp. 13 991–14 002.
[69] B. Fu, Z. Cao, M. Long, and J. Wang, "Learning to detect open classes for universal domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 567–583.
[70] J. Andrews, T. Tanay, E. J. Morton, and L. D. Griffin, "Transfer representation-learning for anomaly detection," Tech. Rep., 2016. [Online]. Available: https://research.google/pubs/pub37648/
[71] N. Huyan, D. Quan, X. Zhang, X. Liang, J. Chanussot, and L. Jiao, "Unsupervised outlier detection using memory and contrastive learning," 2021, *arXiv:2107.12642*.
[72] J. Cheng and N. Vasconcelos, "Learning deep classifiers consistent with fine-grained novelty detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1664–1673.
[73] A. Bendale and T. Boult, "Towards open world recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1893–1902.
[74] C. Cortes, G. DeSalvo, and M. Mohri, "Learning with rejection," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2016, pp. 67–82.
[75] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
[76] Y. Zuo, W. Qiu, L. Xie, F. Zhong, Y. Wang, and A. L. Yuille, "Craves: Controlling robotic arm with a vision-based economic system," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4214–4223.
[77] W. Deng, L. Zheng, Q. Ye, Y. Yang, and J. Jiao, "Similarity-preserving image-image domain adaptation for person re-identification," 2018, *arXiv:1811.10551*.
[78] W. Zhang, D. Xu, W. Ouyang, and W. Li, "Self-paced collaborative and adversarial network for unsupervised domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 6, pp. 2047–2061, Jun. 2021.
[79] Z. Zhang, M. Wang, Y. Huang, and A. Nehorai, "Aligning infinite-dimensional covariance matrices in reproducing kernel hilbert spaces for domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3437–3445.
[80] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.
[81] Y. Cui, Y. Song, C. Sun, A. Howard, and S. Belongie, "Large scale fine-grained categorization and domain-specific transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4109–4118.
[82] Y. Zhang, H. Tang, and K. Jia, "Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 233–248.
[83] X. Yan, D. Acuna, and S. Fidler, "Neural data server: A large-scale search engine for transfer learning data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3893–3902.
[84] N. Ruiz, S. Schulter, and M. Chandraker, "Learning to simulate," 2018, *arXiv:1810.02513*.

Weijian Deng received the MS degree from the University of the Chinese Academy of Sciences, Beijing, China, in 2019. He is currently working toward the PhD degree in computer science with Australian National University, Canberra, Australia. His current research interests include visual understanding, domain adaptation, and model generalization.

Liang Zheng (Senior Member, IEEE) received the BE degree in life science and the PhD degree in electronic engineering from Tsinghua University, China, in 2010 and 2015, respectively. He was a posdoctoral researcher with the Centre for Artificial Intelligence, University of Technology Sydney, Australia. He is currently a senior lecturer with the School of Computing, Australian National University. He holds the CS Futures fellowship and ARC DECRA fellowship. His research interests include object re-identification and data-centric vision. He was named Top-40 Early Achievers by The Australian.