Are Labels Necessary for Classifier Accuracy Evaluation?

Weijian Deng and Liang Zheng

Australian National University
{firstname.lastname}@anu.edu.au

Abstract. To calculate the model accuracy on a computer vision task, *e.g.,* object recognition, we usually require a test set composing of test samples and their ground truth labels. Whilst standard usage cases satisfy this requirement, many real world scenarios involve unlabeled test data, rendering common model evaluation methods infeasible. We investigate this important and under-explored problem, *Automatic model Evaluation* (AutoEval). Specifically, given a labeled training set and a model, we aim to estimate the model accuracy on unlabeled test datasets. We construct a *meta-dataset*: a dataset comprised of datasets generated from the original training set via various image transformations such as rotation, background substitution, foreground scaling, etc. As the classification accuracy of the model on each sample (dataset) is known from the original dataset labels, our task can be solved via regression. Using the feature statistics to represent the distribution of a sample dataset, we can train regression techniques (*e.g.,* a regression neural network) to predict model performance. Using synthetic meta-dataset and real-world datasets in training and testing, respectively, we report reasonable and promising estimates of the model accuracy. We also provide insights into the application scope, limitation and future directions of AutoEval.

1 Introduction

Model evaluation is an indispensable step in almost every computer vision task. Using an unseen test set, it aims to estimate a model's (hopefully) unbiased accuracy when deployed in the real world scenarios. In most cases, we are provided with a labeled test set, allowing us to calculate the accuracy of a model by comparing the predicted labels with the ground truth labels Fig. 1(a). The community has many well-established benchmarks that provide various types of evaluation metrics. For example, top-1 error, commonly used in image classification datasets ImageNet [7] and Cifar-10 [25], indicates whether the predicted class with the highest confidence is the same with the ground truth. There are some other metrics such as mean average precision in object detection [29] and panoptic quality [24] in panoptic segmentation. To use these metrics, a common requirement is to have ground truth labels for the test set.

Compared with evaluation on these benchmarks, evaluating model performance for real-world deployment is not that straightforward. Often, real-world

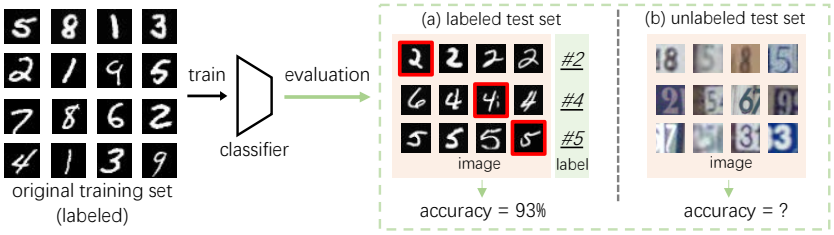

Fig. 1. Problem illustration. Given a classifier trained on an training set, we can gain a hopefully unbiased estimate of its real world performance by evaluating it on an unseen labeled test dataset, as shown in (a). However, in many real-world deployment scenarios we are presented with unlabeled test datasets (b), and as such are unable to evaluate our classifier using common metrics. This inspired us to explore the problem of Automatic model Evaluation

data follows distributions that differ from the original training distribution. In this case, a model's performance on the test set provided in a benchmark may not reflect that achieved during deployment. If we still want to have an estimation of the model's accuracy in this scenario, we have to re-evaluate it on the real-world data. However, we often face scenarios where annotations of test samples are not provided. Furthermore, it can be very complex and expensive to manually gather labels. Even if acquired, these samples may only cover a very limited set of conditions, adding bias to the evaluated performance. For example, it is very expensive to annotate test samples for license plate recognition systems; even label is gathered for every car, it still can not capture the diversity of real-world circumstances such as lighting and weather condition. This raises an interesting question: *can we estimate model performance on a test set without test labels?*

To answer this question, this paper introduces the Automatic model Evaluation (AutoEval) problem. Given a model trained on a training set, the goal is to estimate the accuracy of model predictions on an unlabeled test set. Here, we introduce an example in Fig. 1(b). Given a classifier trained on MNIST [26] with an unlabeled test set, we want to estimate the classification accuracy *without* knowing any test ground truths. This problem is challenging, as a test set contains many images, and each image has varied and rich visual content. However, by visually inspecting the obvious differences between test and training sets, we can infer that the accuracy on the test set is low.

From this observation, we study AutoEval by considering the distribution shift between training and test sets and how it effect model accuracy. Existing literature gives us important hints. Dataset distributions can be represented by first and second-order statistics of the mean vector of output image feature representations [37,33,15]. For example, distribution difference can be estimated via Frchet Distance (FD) [12,10] or maximum mean discrepancy (MMD) metric [15]. Furthermore, domain adaptation literature shows that a smaller distribution difference leads to higher target domain accuracy and implies that a large

	Image Classification	AutoEval
Sample	Image	Dataset (sample set)
Label	Sample class ground truth	Accuracy of model on sample set
Training Dataset	Set of labeled images	Set of synthetic labeled sample sets (meta set)
Testing Dataset	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

Table 1. Analogies between standard image classification terms and their AutoEval equivalents. The analogy shows that the image classification is an image based task, while the AutoEval problem in this work is dataset based

domain gap causes a low test accuracy [13,38,39]. This paper shares a related but distinct purpose from domain adaptation. Instead of improving the accuracy on the target (test) set, we aim to quantitatively estimate the test accuracy without labels, requiring us to study the underlying relationship between dataset distributions and model performance.

We propose to learn this relationship via a meta-dataset (dataset of datasets). We use the terms meta set and meta-dataset interchangeably. Unlike most existing datasets that treat each image as a sample, we work on the dataset level: in the meta-dataset, each dataset is treated as a sample, which we term "sample set". Analogies between standard image classification and AutoEval tasks are shown in Table. 1. The sample sets should possess an appropriate number of images, exhibit a diverse spread of distributions and, in the case of image classification, have the same set of classes. It is difficult to collect sufficient real-world sample sets that meet the three requirements, so we construct the meta set by data synthesis. Every sample set in the meta set is generated from a seed set that follows the same distribution as the original training set. This is achieved via various image transformation operations on the generative set, including blurring, background substitution, foreground rotation, *etc.* Because the synthetic sample sets are transformed versions of the seed set, they are fully labeled. Using these labels, we can obtain the recognition accuracy of the classifier on each sample set. Sample set i can thus be denoted by $(\boldsymbol{f}_i, a_i)$, where a_i is recognition accuracy, and $\boldsymbol{f}_i$ is the vector representation of the dataset, *e.g.,* the mean vector of image features in this dataset. With this meta set denoted as $\{(\boldsymbol{f}_i, a_i)\}, i = 1, ..., N$, where N is the number of sample sets, we can train a regression model that takes $\boldsymbol{f}$ as input and predicts the recognition accuracy of a trained model on a test dataset. To summarize, we make the following contributions.

– We introduce the AutoEval task, aiming to estimate the recognition accuracy of a trained classifier on a test set *without* test labels.

– We propose to learn an accuracy regression model from a synthetic meta-dataset and obtain decent accuracy predictions for real-world test datasets.

2 Related Work

Dataset-level analysis. While most computer vision tasks can be described as some form of image-level analysis, very limited literature exists that explores dataset-level analysis. Some work has focused on optimizing dataset construction by selecting a subset of images for pretraining for a specific downstream task [6,43,41]. Specifically, Cui *et al.* [6] propose a similarity metric to select relevant classes of a given dataset for network pretraining. In a similar line of work, Zhang *et al.* [43] select task-related images from the auxiliary data for training networks. Recently, Yan *et al.* [41] 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 a downstream task [22] [34] by learning 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. Specifically, we propose to estimate the model classification performance on an unlabeled dataset from the perspective of the data distribution.

The out-of-distribution (OoD) detection task [9,18,27,40,28] considers the distribution of test samples. Specifically, this task aims to detect test samples that follow a distribution different from the training distribution. This task has been studied from different views, such as anomaly detection [1], open-set recognition [2], and rejection [4]. Our work considers the all test samples to predict model performance based on the overall distribution of a test dataset.

Model evaluation. Recently, generative adversarial networks (GANs) have achieved outstanding results in generating realistic images [14,23,3]. The objective of GANs is to generate images that follow the real-data distribution. For the evaluation of the performance of GANs at image generation, several metrics have been proposed, such as Inception Score [35] and Frchet Inception Distance [19]. While these evaluation metrics focus only on image quality, our work is concerned with the final model accuracy on a dataset of unlabeled images.

Unsupervised Domain adaptation. Our work also relates to unsupervised domain adaptation. This task aims to use labeled source samples and unlabeled target samples to learn a classifier that can generalize well on the target dataset. Existing approaches attempt to eliminate the shift between the source and target distributions. Many moment matching schemes have been studied for this task [37,30,39,33,37,44]. Long *et al.* [30] and Tzeng *et al.* [39] utilize the maximum mean discrepancy (MMD) metric [15] to learn a shared feature representation. Peng *et al.* [33] aims to address multi-source domain adaptation by matching moments. Sun *et al.* [37] propose to perform domain adaptation by matching the second-order of feature distribution. Zhang *et al.* [44] propose to map infinte-dimensional matrices in RKHS. In this work, we study the underlying relationship between the model performance and the distribution shift.

By leveraging dataset level statistics of we are able to accurately predict model performance on unlabeled test sets.

3 Automatic Model Evaluation

We are interested in predicting the recognition accuracy of a trained classifier on an unlabeled test set.

3.1 Problem Definition

We first define a labeled dataset, $D^l = \{(\boldsymbol{x}_i, y_i)\}$ for $i \in [1, ..., M]$ where $\boldsymbol{x}_i$ is an image, y_i is its class label, and M is the number of images. Consider a source domain $\mathcal{S}$, from which we sample an original training dataset $\mathcal{D}_{ori}$. We use $\mathcal{D}_{ori}$ to 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, we obtain its classification accuracy by comparing the class predictions $\hat{y}_i$ with the ground truths y_i to obtain accuracy,

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

where $[\![\cdot]\!]$ is an indicator function returning 1 if argument is true and 0 otherwise.

In AutoEval, given $f_{\boldsymbol{\theta}}$ and an unlabeled dataset $\mathcal{D}^u = \{\boldsymbol{x}_i\}$ for $i \in [1, ..., M]$, we use an accuracy predictor $A : (f_{\boldsymbol{\theta}}, \mathcal{D}^u) \rightarrow a$, which outputs an estimated classifier accuracy $a \in [0, 1]$ on this test set,

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

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

3.2 An Intuitive Solution

We first present an intuitive solution to the AutoEval problem, which is not learning based. This solution is motivated by the pseudo labeling strategy in many vision tasks [18,42,31]. The basic assumption is: if a class prediction is made with a high confidence (softmax score), this prediction is likely to be correct. Formally, let us consider a K-way classification problem. When feeding a test image x_i to a trained classifier $f_{\boldsymbol{\theta}}$, we obtain $\boldsymbol{s}_i \in \mathbb{R}^K$, which is the output of the softmax layer. The k-th entry in $\boldsymbol{s}_i$ characterizes the probability of x_i belonging to class k. The ℓ_1 norm $\|\boldsymbol{s}_i\|_1 = 1$.

Intuitively, if the maximum entry of $\boldsymbol{s}_i$ is greater than a threshold τ, image x_i is considered to be correctly classified. Our accuracy predictor is written as,

$$a_{max} = A_{max}(f_{\boldsymbol{\theta}}, \mathcal{D}^u) = \frac{\sum_{i=1}^{M} [\![\max(\boldsymbol{s}_i) > \tau]\!]}{M}, \tag{3}$$

where M is the number of images in $\mathcal{D}^u$. We will evaluate A_{max} in the experiment and show that it does not work consistently well.

4 Learning to Predict Classifier Accuracy

4.1 Formulation

Motivated by the implications in domain adaptation, we propose to address AutoEval by measuring the distribution shift between the original training set and the test set, and explicitly learning a mapping function from the distribution shift to the classifier accuracy.

Under this consideration, we formulate AutoEval as a dataset-level regression problem. In this problem, we view a dataset as a sample, and its label is the recognition accuracy on the dataset itself. Suppose we have N sample sets. We denote the j-th sample set $\mathcal{D}_j$ as $(\boldsymbol{f}_j, a_j)$, where $\boldsymbol{f}_j$ is some vector representation for $\mathcal{D}_j$, 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,

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

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 classifier accuracy of the j-th sample set $\mathcal{D}_j$, and a_j is the ground truth classifier accuracy of $\mathcal{D}_j$.

During testing, we extract the dataset representation $\boldsymbol{f}^u$ for the unlabeled test set $\mathcal{D}^u$, and obtain the classification accuracy using $a = A(\boldsymbol{f}^u)$.

To optimize the regression model defined in Eq. 4 and Eq. 5, we need to specify the design of 1) dataset representation $\boldsymbol{f}_i$, 2) regression model A, and 3) the N sample sets (meta-dataset).

4.2 Two Regression Models and Their Dataset Representations

Linear regression. We first introduce a simple linear regression model,

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

where $f_{linear} \in \mathbb{R}$ is the representation of sample set $\mathcal{D}$, and $w_1, w_2 \in \mathbb{R}$ are parameters of this linear regression model. Based on the intuition that the domain gap impacts classifier accuracy, we define f_{linear} as the quantified domain gap between dataset $\mathcal{D}$ and the original training set $\mathcal{D}_{ori}$. Specifically, we use the Frchet distance [10] to measure the domain gap, and thus,

$$f_{linear} = \mathrm{FD}(\mathcal{D}_{ori}, \mathcal{D}) = \|\boldsymbol{\mu}_{ori} - \boldsymbol{\mu}\|_2^2 + Tr(\boldsymbol{\Sigma}_{ori} + \boldsymbol{\Sigma} - 2(\boldsymbol{\Sigma}_{ori}\boldsymbol{\Sigma}))^{\frac{1}{2}}, \tag{7}$$

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 in $\mathcal{D}_{ori}$ and $\mathcal{D}$, which are extracted using the

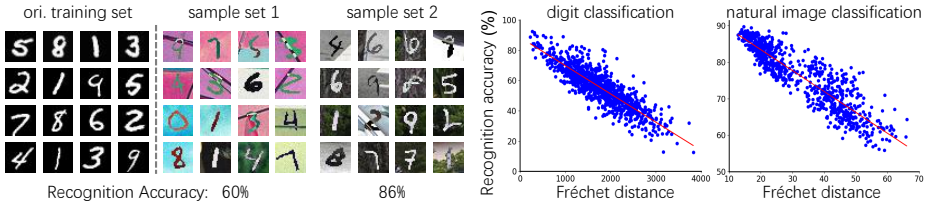

Fig. 2. Relationship between the distribution shift and classifier accuracy on digits and natural image classification. Each point represents a sample set of the meta set. We observe a very strong negative correlation between accuracy and distribution shift, *i.e.*, the test error is in proportion to the distribution shift

classifier $f_{\boldsymbol{\theta}}$ trained on $\mathcal{D}_{ori}$. Other measurements of the domain gap can also be used, such as MMD [15].

Proof of concept. Given a meta set and a classifier trained on the training dataset $\mathcal{D}_{ori}$ from a source domain $\mathcal{S}$, we study the relationship between classifier's accuracy and distribution shift. In Fig 2, we show the accuracy as a function of the distribution shift in the Fig 2. The distribution shift is measured by Frchet distance with the features extracted from the trained classifier. In practice, we use the activations in the penultimate of the classifier as features.

In both digits and natural image classification scenarios, we observe a very strong negative correlation between accuracy and distribution shift in both digits and natural image classification. Namely, the classifier tends to achieve a low accuracy on the sample set which has a high distribution shift with training set $\mathcal{D}_{ori}$. This is consistent with our motivation that distribution shifts impacts the classifier accuracy.

Neural network regression. Besides the linear regression, we also propose a neural network regression model, $a_{neural} = A_{neural}(\boldsymbol{f}_{neural})$, which has the same formulation as Eq. 4. In practice, we use a simple fully connected neural network for regression. The input of the model is the dataset representation $\boldsymbol{f}_{neural}$, and the output is the estimated classifier accuracy a_{neural}.

Inspired by the works about the moments of distributions [33] [44], we propose to represent a dataset by its first-order and second-order feature statistics, *i.e.*, mean vector and covariance matrix. Moreover, we also include a 1-dim FD score to the representation. Compared with linear regression, the neural network regression has a richer dataset representation. The dataset representation is written as,

$$\boldsymbol{f}_{neural} = [f_{linear}; \boldsymbol{\mu}; \boldsymbol{\sigma}], \tag{8}$$

where $f_{linear} \in \mathbb{R}$ is the Frchet distance between $\mathcal{D}$ and $\mathcal{D}_{ori}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are calculate the same way as Eq. 7. We calculate $\boldsymbol{\sigma}$ by taking a weighted summation of each row of $\boldsymbol{\Sigma}$ to produce a single vector, using learned column specific coefficients that are shared across all rows. For example, if the feature extracted from $f_{\boldsymbol{\theta}}$ is d-dim, the dimensionality of $\boldsymbol{f}_{neural}$ is $1 + 2d$.

Fig. 3. Visual examples of transformations. Here we show autoContrast, rotation, coloring, and translation. For other used transformations, we refer readers to [5]

4.3 Constructing Training Meta-dataset

Meta-datasets for training. The regression model (Eq. 4, Eq. 5, Eq. 8) takes the dataset representation as input and outputs a classification accuracy. To train it, we need to prepare a meta-dataset in which each sample is a dataset. In classification, the diversity of the samples in the training set should ideally be sufficient such that test scenario is represented in its distribution. In this work, we seek to create a diverse meta set that (hopefully) contains the test distributions. To construct such a meta set, we should collect sample sets that are 1) large in number, 2) diverse in the data distribution, and 3) have the same label space with the original training set. There are very few real-world datasets that satisfy these requirement, so we resort to data synthesis.

For each classification task (digits or natural images), 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 $\mathcal{D}_{ori}$. Given $\mathcal{D}_s$, we apply various visual transformation and obtain N different sample sets $\mathcal{D}_j, j = 1, ..., N$. Since $\mathcal{D}_s$ is fully labeled, these sample sets inherent the labels from $\mathcal{D}_s$.

To create a sample set $\mathcal{D}_j$, we adopt a two-step procedure: perform background change, and then image transformations. In the first step, we keep the foreground / object unchanged and replace the background. For each sample set, we randomly select an image from the COCO dataset [29], from which we randomly crop a patch and use it as the background. The patch scale and position in that image are both random. In the second step, for the background-replaced images, we use six image transformations defined in [5], including autoContrast, rotation, color, brightness, sharpness, and translation. Examples of some transformations are shown in Fig. 4.2. For each sample set, we randomly select and combine three out of the six transformations, with the magnitude of each transformation being random on per-sample basis. As such, each sample set is generated by background replacement and a combination of three image transformations. Fig. 4 presents examples of sample sets in natural image classification, where background replacement can be observed. In the supplementary materials, we present the detailed transformation parameters and more visual examples of the training meta set. Note that a sample set inherits all the image labels from the seed set and is fully labeled. As such, we can calculate the recognition accuracy of classifier $f_{\boldsymbol{\theta}}$ on each sample set. Sample set $\mathcal{D}_j$ can be

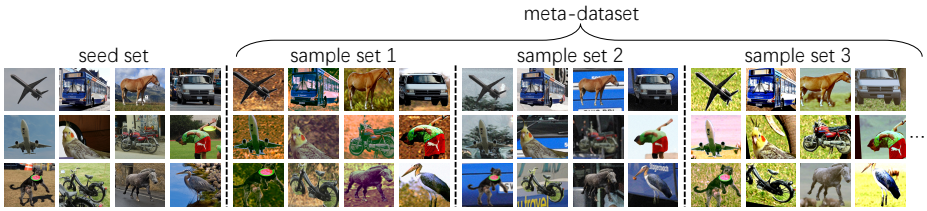

Fig. 4. The seed set and examples of three sample sets. The seed set is from the same distribution with the original training set; they share the same classes but do not have image overlap. The sample sets are generated from the seed by background replacement and image transformations. The sample sets exhibit distinct data distributions, but inherit the foreground objects from the seed, and thus are fully labeled. Many sample sets form a meta-dataset from which an accuracy regression model is trained

denoted as $(\boldsymbol{f}_j, a_j)$, which is used as a training sample to optimize the regression model.

Real-world datasets for testing. This is an early attempt for the AutoEval problem. To our knowledge, we could only find few real-world datasets that have different distributions but contain the same classes. To clarify the AutoEval problem, we conduct extensive analyses with these dataset.

For digits classification, we use USPS [21] and SVHN [32], both with 10 classes. For natural image classification, we use three existing datasets, *i.e.,* PASCAL [11], Caltech [16], and ImageNet [7], all with 12 classes. Details of the test meta sets are provided in Section 5.1.

5 Experiment and Analysis

5.1 Experimental Settings

We study the AutoEval problem on two classification tasks: digit classification and natural image classification.

Digit classification. The original training set contains all the training images of MNIST. We use the testing images of MNIST as the seed to generate the training meta set. Because MNIST images are binary, the foreground can be separated from the background. When generating meta set, we randomly select an image from the COCO training set, and the background of each image is replaced with a random patch of the sampled COCO image. Then, we apply three out of six image transformations to images. We generate $3,000$ sample sets, of which we use 3,000 and 1,000 for the training and the validation meta set, respectively. Moreover, we use two real datasets for testing, *i.e.,* USPS [21] and SVHN [32] datasets.

Natural image classification. We use COCO [29] training set as the original training set, and COCO validation set as the seed set to build the meta set. When generating the meta set for training, we use the instance mask annotation

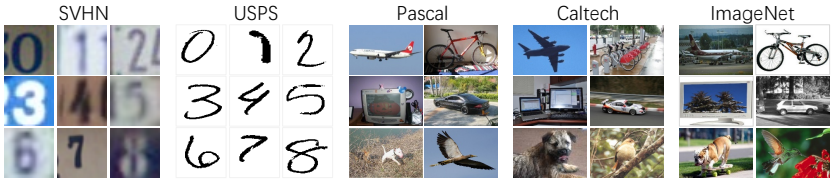

Fig. 5. Sample images from real-world test datasets, including SVNH, USPS, Pascal, Caltech and ImageNet. The former two are for digit classification, and the latter three are for natural image classification. We predict the classifier accuracy on these datasets

of the COCO validation set to get foreground regions. Similar to digit classification, for each sample set, we replace the background with a random patch of an image in the COCO test set. We apply image transformations to introduce more visual difference. We create 1,600 sample sets from the seed set, of which we use 1,000 and 600 for the training and the validation meta set, respectively. In testing, we use PASCAL [11], Caltech [16], and ImageNet [7]. For each dataset, we select images of 12 common classes, *i.e.*, aeroplane, bike, bird, boat, bottle, bus, car, dog, horse, monitor, motorbike, and person. Note that we reduce the "person" class to 600 images to balance the overall number of images per class.

Classifier architecture. For digit classification, we use LeNet-5 [26] as classifier, which is trained on the original MNIST training set. Since the all images are mapped to the RGB space, we modify the number of input channel of LeNet-5 to 3. For natural image classification, we use the ResNet-50 pretrained on ImageNet [7] which is adapted to the 12-way classification.

Metrics. This paper estimates the recognition accuracy of a model on a test set. To evaluate the performance of such estimate, we use root mean squared error (RMSE) and mean absolute error (MAE) as metrics. RMSE measures the average squared difference between the estimated classifier accuracy and ground-truth accuracy. MAE measures the average magnitude of the errors s. Small RMSE and MAE correspond to good predictions and vice versa.

5.2 Evaluating Three Classifier Accuracy Predictors

This paper introduces three possible methods to estimate the recognition accuracy, including the confidence-based method, linear regression and neural network regression. We report and compare the performance of these three methods in Table 2. For the confidence-based method, two thresholds are used, *i.e.*, $\tau = 0.8$ or 0.9 (Eq. 3). We provide the following analysis.

The confidence-based method is very unstable. Under a specific threshold ($\tau = 0.8$), this method makes accuracy prediction on natural image datasets (RMSE=2.28%), but its prediction quality drops significantly (from 2.28% to 6.96%) when we increase value of τ to 0.9. What is more, its performance is very poor when considering the digit classification task. Under two values of τ, the RMSE is consistently high, *i.e.*, 22.66% and 25.59%, respectively. Although

Table 2. Method comparison in predicting classification accuracy. Results on digit classification (SVHN and USPS datasets) and natural image classification (Pascal, Caltech and ImageNet) are shown. We compare three methods, *i.e.,* confidence-based (Section 3.2), linear regression and neural network regression (Section 4.2). For each dataset, we report the estimated classification accuracy (%). For both digit and natural image classification, RMSE (%) is reported. The original training sets are MNIST and COCO, respectively. The ground-truth recognition accuracy (%) is presented

	Digits			Natural images			
Method	SVHN	USPS	RMSE↓	Pascal	Caltech	ImageNet	RMSE↓
Ground-truth accuracy	25.46	64.08	0	86.13	93.40	88.83	0
Confidence ($\tau = 0.8$)	7.97	37.22	22.66	84.32	90.78	86.50	**2.28**
Confidence ($\tau = 0.9$)	7.03	32.94	25.59	78.61	87.71	87.71	6.96
Linear reg.	26.28	50.14	9.87	83.87	79.11	83.19	8.62
Neural network reg.	27.52	64.11	**1.46**	87.76	89.39	91.82	**3.04**

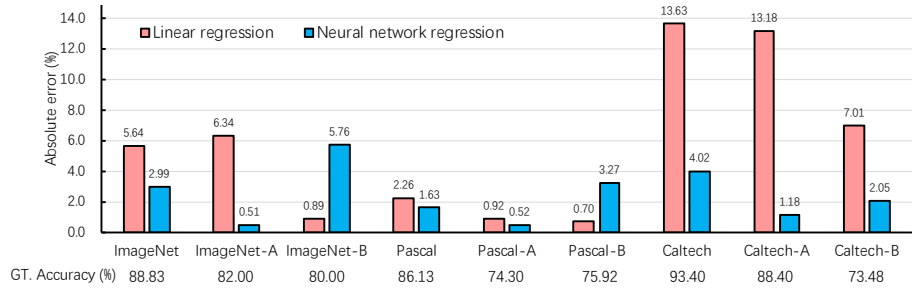

Fig. 6. Comparing linear regression and neural network regression when test data undergo new image transformations such as Cutout [8,45], Shear, Equalize and ColorTemperature [5]. The transformed datasets are denoted by "-A" and "-B". We report the absolute error (%) between estimated classifier accuracy and the ground truth accuracy (also shown below each dataset)

this method is simple to implement, it is sensitive to different thresholds and different tasks. A possible way to improve it is to learn task-specific thresholds that take into account the dataset distributions.

Regression methods make better predictions than confidence-based method. In digit datasets, the RMSE values of linear regression and neural network regression are 9.87% and 1.46%, respectively. A similar trend can be observed in natural image datasets. Their RMSE scores are generally lower and more stable than those of the confidence-based method. This indicates the effectiveness of learning-based methods: the distribution difference between the original training set and the test sets is a critical feature.

Neural network regression is generally better than linear regression. Table 2 demonstrates that neural network regression is more accurate than linear regression in both digit and natural image datasets. For example, RMSE of the former is 8.41% lower than the latter on digit datasets. In fact, the RMSE of

neural network regression is as small as 1.46%: the predicted classifier accuracy is very close to the ground truth accuracy.

To further examine the two regression methods, we report a new group of experiment on natural image classification in Fig. 6. Here, we perform image transformations to the real datasets (ImageNet, Pascal and Caltach) and assess the performance of the two regression methods on these "edited real-world datasets". Note that the image transformations we use here are *not* those applied in meta-dataset generation. Thus, this experiment assesses some generalization ability of the regression methods. From Fig. 6, we first observe that the ground truth recognition accuracy on the edited datasets is lower than that on the real datasets. It suggests that the image transformations are introducing visual differences that hinder the classifier performance. When comparing the two regression methods, we observe that neural network regression gives lower errors in 2 out of 9 datasets. In these two datasets (ImageNet-B and Pascal-A), the absolute error of neural network regression is 5.76% and 3.27%, respectively, which are acceptable. This suggests that neural network regression is generally better.

We note that linear regression is significantly inferior to neural network regression on Caltech datasets, where linear regression gives errors higher than 10%. However, Caltech is an interesting dataset. Its images have relatively simple backgrounds and salient foregrounds, implying that they are "easy" to classify. However, such simple background contrasts significantly with the original training set (COCO), so the FD score between Caltech and COCO is very large. By only looking at the FD score, linear regression tends to predict that the recognition accuracy on Caltech is low. In comparison, neural network regression not only looks at FD, but also the data distribution of Caltech. The meta-dataset might already contain sample sets with such "simple backgrounds" (large FID), and high recognition accuracy. Under such circumstances, the network has learned to overrule the large FD and instead resort to the "simple background" when making predictions.

5.3 Analysis of the Training Meta-Dataset

The synthetic meta-dataset is a key component of our system, allowing us to obtain labeled samples sets in a large scale. We analyze its impact on the system from the following aspects.

Meta set size. We study the impact of meta set size on the regression methods. In Fig. 7 (first row). We observe the results of linear regression are relatively stable with different meta set size. It can achieve acceptable performance even with 50 sample sets. This is because linear regression only has two parameters (Eq. 6), which can be learned with few samples. In comparison, neural work cannot achieve good results when the number of sample sets is small. When provided adequate sample sets, the neural network can learn effectively and surpasses the linear regression.

Sample set size. By default, the number of images in each sample set is equal to that of seed $\mathcal{D}_s$. We study the impact of sample set size on the regression methods. In the experiment, we set the meta set size 1000, and vary

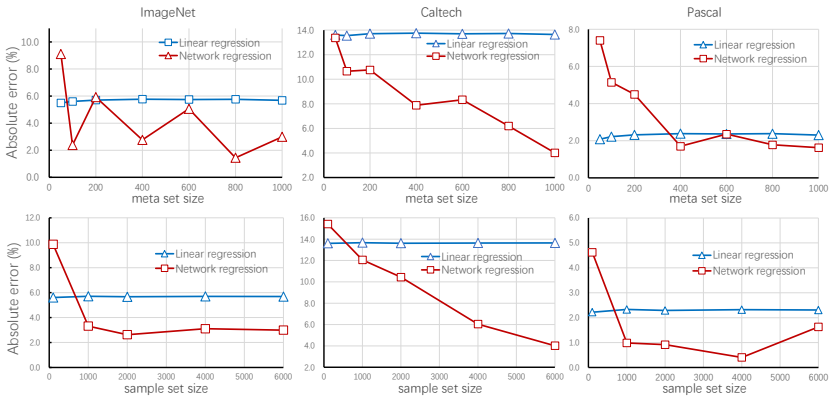

Fig. 7. The impact of meta set size (first row) and sample set size (second row) on the performance of regression methods. We report the absolute errors (%) between estimated classifier accuracy and the ground-truth accuracy. We observe that linear regression is relatively stable with different sample set and meta set size. In comparison, neural network needs more and large sample sets for training

the sample set size. In Fig .7 (second row), we observe linear regression is stable under different sample set sizes. In comparison, the neural network needs more images in each sample set for training. We think more images in each sample set makes the dataset representation more accurate. This is beneficial for regression learning of network.

Classifier architecture. The dataset representation is based the features of classifier trained on the source domain. Different classifier architectures will lead to different dataset representations. In Fig. 8, we report the results with Desenet-121 [20] and VGG-16 [36]. We observe that both neural network and linear regression are still effective with features extracted from these two networks. We believe the dataset distributions in our meta set are diverse regardless of the classifier architecture. This is the key reason why both regression methods can work well under different feature statistics from three classifiers.

6 Conclusions 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 initial attempts by devising two regression models which directly estimate classifier accuracy based on overall dataset statistics. We build a dataset of datasets (meta-dataset) to train the regression model. We show that the synthetic meta-dataset can cover a good range of dataset distributions and

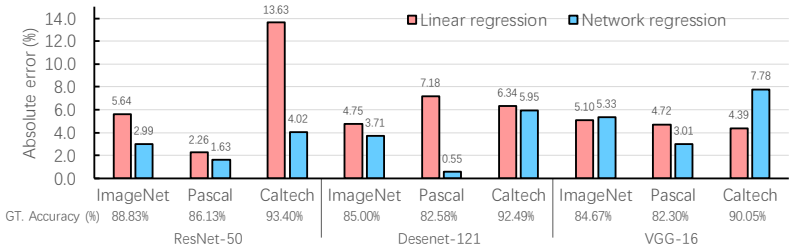

Fig. 8. Estimation accuracy of linear regression and neural network regression under different classifier architectures, *i.e,* ResNet-50 [17], Desenet-121 [20] and VGG-16 [36]. For each classifier, we show the absolute error (%) between grouch-truth accuracy (shown below each dataset) and estimated accuracy. We observe both methods are effective with feature statistics from three classifier architectures

benefit AutoEval on real-world test sets. For the remainder of this section, we discuss the limitations, potential, and interesting aspects of AutoEval.

Application scope. Our system assumes that variations in the real-world test datasets can be approximated by the image transformations in the training meta set. If the test datasets exhibit some very special patterns or conditions, our system might not be able to work. An example is that the test dataset has an entirely different set of classes, and this test distribution cannot be approximated by the meta-dataset in our work. Under this circumstance, our trained model algorithm will still give an estimated accuracy, which is clearly incorrect. On a related extreme case, the test dataset might only contain ambiguous / misclassified samples, meaning that the test accuracy could be as poor as random. Such cases are not included in the construction of meta-dataset, either. Potentially, the above two problems can be addressed by including such cases into the meta-dataset, but it requires specific dataset design. Another option is to explore out-of-distribution detection [9] [18], sample anomaly detection [1] and rejection [4] to explicitly detect such cases.

Dataset Representation. Our work raises an interesting research problem: how to represent dataset? This problem is much more challenging than describing a single image because a dataset contains much more information. This work uses first- and second-order feature statistics and the FD score under the context of object recognition. In recognition, stronger dataset presentations need to be designed so as to better characterize a dataset. This can be manually designed, or end-to-end learned. On the other hand, it would be interesting to study data representation in other tasks such as object detection and semantic segmentation, where global feature statistics might not be a good choice.

Similarities between datasets. In this work, we measure dataset similarity using the FD score. However, this problem is as challenging as dataset representation, especially when we aim to connect the similarity with test accuracy. Moreover, this problem will benefit the domain adaptation field, where more precise domain gap measurement and its connection to target set accuracy will significantly help algorithm design.

7 Supplementary

In this section, we first detail the image transformations used to generate the meta dataset. We then provide more visual examples of sample sets. Last, we show the relationship between distribution shift and accuracy based on three different classifiers.

7.1 Meta Set

Image Transformation The image transformations used in meta set construction are listed in Table. 3. We briefly describe each transformation (the second column), and introduce its magnitude information (the third column).

Table 3. List of all image transformations that we choose from during meta set construction. The magnitude range of each transformation is shown in the third column. Some transformations do not use the magnitude information (*e.g.*, AutoConstrast)

Operation Name	Description	Range of magnitudes
AutoContrast	Maximize the image contrast, by making the darkest pixel black and lightest pixel white.	
Rotate	Rotate the image *magnitude* degrees.	[-30, 30]
Color	Adjust the color balance of the image. A *magnitude*=0 gives a black & white image, whereas *magnitude*=1 gives the original image.	[0.1, 1.9]
Brightness	Adjust the brightness of the image. A *magnitude*=0 gives a black image, whereas *magnitude*=1 gives the original image.	[0.1, 1.9]
Sharpness	Adjust the sharpness of the image. A *magnitude*=0 gives a blurred image, whereas *magnitude*=1 gives the original image.	[0.1, 1.9]
TranslateX(Y)	Translate the image in the horizontal (vertical) direction by *magnitude* number of pixels.	[-150, 150]
Cutout [8,45]	Set a random square patch of side-length *magnitude* pixels to gray.	[0, 60]
ShearX(Y)	Shear the image along the horizontal (vertical) axis with rate *magnitude*.	[-0.3, 0.3]
Equalize	Equalize the image histogram.	
ColorTemperature	Change the temperature of an image to a given *magnitude* in Kelvin.	[1000, 11000]

Sample Set We show more visual examples of sample sets in Fig. 9 and Fig. 10. Each sample set is generated by background change and a combination of three image transformations. Compared with the seed set, each sample set has many visual differences. Thus, each sample set exhibits a distinct data distribution.

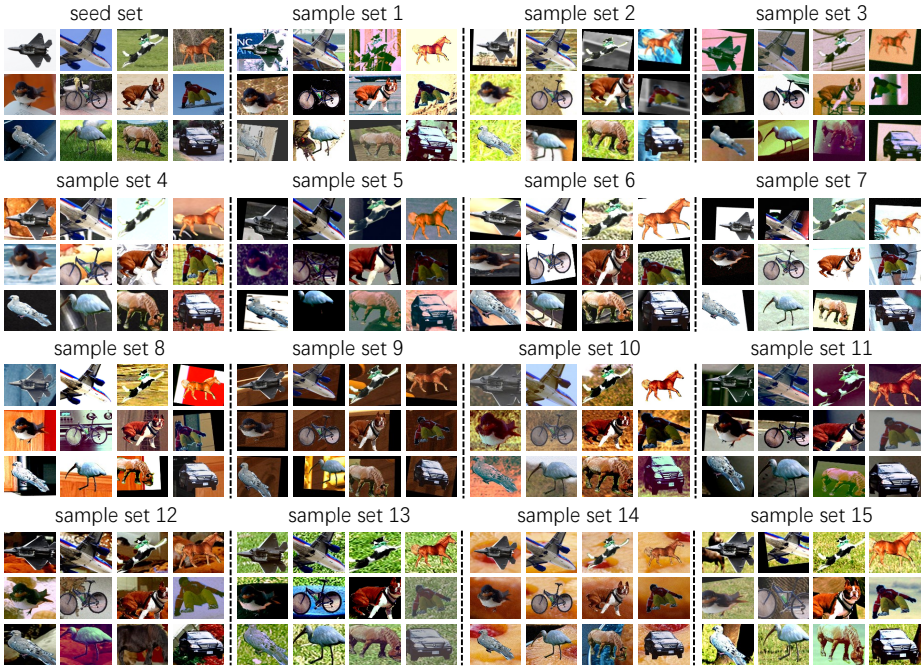

Fig. 9. Seed set and examples of fifteen sample sets for the task of natural image classification. The seed set is sampled from the same distribution as the original training set; they share the same classes but do not have image overlap. The sample sets are generated from the seed by background replacement and image transformations. The sample sets exhibit distinct data distributions, but inherit the foreground objects from the seed, and hence are fully labeled

Moreover, the foreground object is preserved, so the sample set is fully labeled. A meta set consists of many sample sets. With meta set, we can learn a mapping function from the distribution shift to classifier accuracy.

Meta Set Construction Each sample set in the meta set is generated by background replacement and a combination of *three* image transformations. Both techniques can introduce many visual differences, and thus create diverse sample sets. In our work, we use a combination of background change and three random image transformations for the meta set construction. Here, we study the impact of these two techniques on the diversity of the meta set. Specifically, we construct another three meta sets, 1) Meta set A, construction only with background change; 2) Meta set B, construction only with three random image transformations; 3) Meta set C, construction with background change and only one random image transformation. We report the results in Fig. 11.

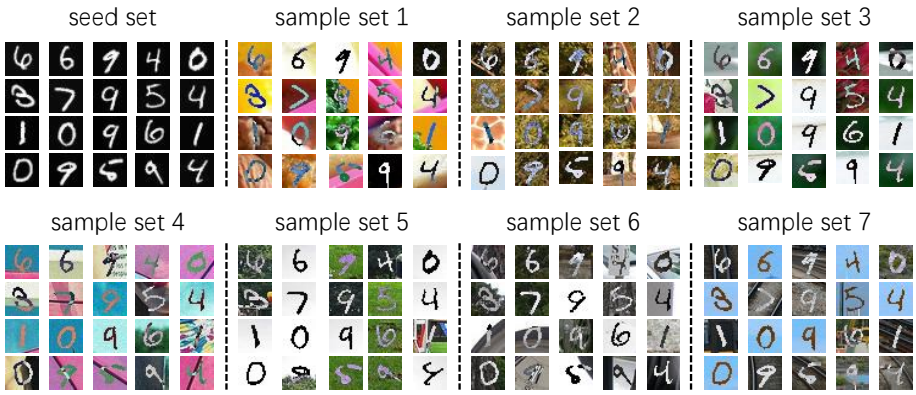

Fig. 10. Seed set and examples of seven sample sets for the task of digit classification

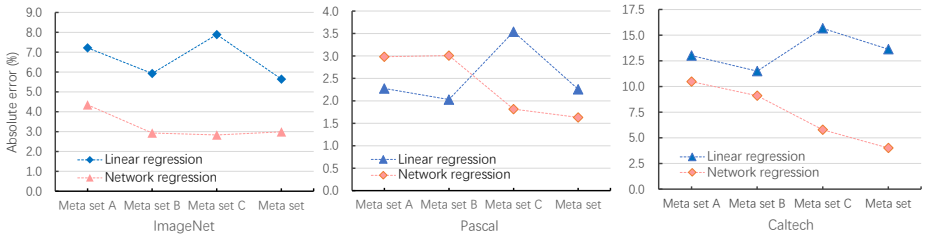

Fig. 11. Absolute errors (%) of the regression method based on different meta sets. The four meta sets are, 1) Meta set A, construction only with background change; 2) Meta set B, construction only with three random image transformations; 3) Meta set C, construction with background change and only one random image transformation; 4) meta set, construction with background change and three random image transformations

We find that regression methods based on meta set A cannot achieve high performance. This indicates only changing background is not enough for meta set construction. Moreover, only using image transformations (meta set B) also insufficient. We note that network regression gains more desirable accuracy when meta set becomes more diverse (using more transformations). This demonstrates that learning a mapping function from the distribution shift to the classifier's accuracy requires a diverse meta set.

7.2 Distribution shift vs. Accuracy

In this section, we show the relationship between distribution and accuracy based on three classifiers, *i.e.*, ResNet-50, Desenet-121, and VGG-16. We show the results on the natural image classification in Fig. 12. Given a classifier, the distribution shift (Frchet distance) is calculated its features. We observe that

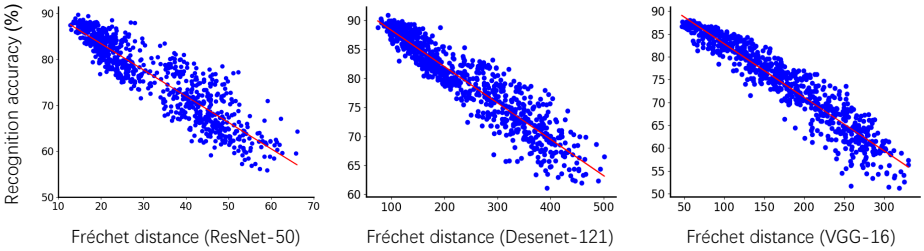

Fig. 12. Relationship between the distribution shift and accuracy based on different classifiers. We show the results on natural image classification. Each point represents a sample set of the meta set. For each classifier, we observe a very strong negative correlation between its accuracy and distribution shift

the range of distribution shift can be varied with different classifiers. However, the overall relationship between the classifier's accuracy and distribution shift is the same. Specifically, they have a very strong negative correlation.

References

1. Andrews, J., Tanay, T., Morton, E.J., Griffin, L.D.: Transfer representation-learning for anomaly detection. JMLR (2016)
2. Bendale, A., Boult, T.: Towards open world recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1893–1902 (2015)
3. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096 (2018)
4. Cortes, C., DeSalvo, G., Mohri, M.: Learning with rejection. In: International Conference on Algorithmic Learning Theory. pp. 67–82. Springer (2016)
5. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 113–123 (2019)
6. Cui, Y., Song, Y., Sun, C., Howard, A., Belongie, S.: Large scale fine-grained categorization and domain-specific transfer learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4109–4118 (2018)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
8. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
9. DeVries, T., Taylor, G.W.: Learning confidence for out-of-distribution detection in neural networks. arXiv preprint arXiv:1802.04865 (2018)
10. Dowson, D., Landau, B.: The fréchet distance between multivariate normal distributions. Journal of multivariate analysis **12**(3), 450–455 (1982)
11. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge 2007 (voc2007) results (2007)

12. Fréchet, M.: Sur la distance de deux lois de probabilité. COMPTES RENDUS HEBDOMADAIRES DES SEANCES DE L ACADEMIE DES SCIENCES **244**(6), 689–692 (1957)
13. Ganin, Y., Lempitsky, V.S.: Unsupervised domain adaptation by backpropagation. In: Proc. ICML. pp. 1180–1189 (2015)
14. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
15. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.J.: A kernel method for the two-sample-problem. In: Proc. NIPS. pp. 513–520 (2006)
16. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Tech. Rep. 7694, California Institute of Technology (2007), http://authors.library.caltech.edu/7694
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
18. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. arXiv preprint arXiv:1610.02136 (2016)
19. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in neural information processing systems. pp. 6626–6637 (2017)
20. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
21. Hull, J.J.: A database for handwritten text recognition research. IEEE Transactions on pattern analysis and machine intelligence **16**(5), 550–554 (1994)
22. Kar, A., Prakash, A., Liu, M.Y., Cameracci, E., Yuan, J., Rusiniak, M., Acuna, D., Torralba, A., Fidler, S.: Meta-sim: Learning to generate synthetic datasets. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4551–4560 (2019)
23. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4401–4410 (2019)
24. Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9404–9413 (2019)
25. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
26. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
27. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In: Advances in Neural Information Processing Systems. pp. 7167–7177 (2018)
28. LEE, K., Lee, K., Lee, H., Shin, J.: Training confidence-calibrated classifiers for detecting out-of-distribution samples. In: ICLR 2018 (2018)
29. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
30. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: Proc. ICML. pp. 97–105 (2015)

31. Ma, F., Meng, D., Xie, Q., Li, Z., Dong, X.: Self-paced co-training. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 2275–2284. JMLR. org (2017)
32. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning (2011)
33. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1406–1415 (2019)
34. Ruiz, N., Schulter, S., Chandraker, M.: Learning to simulate. arXiv preprint arXiv:1810.02513 (2018)
35. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: Advances in neural information processing systems. pp. 2234–2242 (2016)
36. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
37. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
38. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proc. CVPR. pp. 2962–2971 (2017)
39. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance ([Online] Available: https://arxivorg/abs/14123474)
40. Vyas, A., Jammalamadaka, N., Zhu, X., Das, D., Kaul, B., Willke, T.L.: Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 550–564 (2018)
41. Yan, X., Acuna, D., Fidler, S.: Neural data server: A large-scale search engine for transfer learning data. arXiv preprint arXiv:2001.02799 (2020)
42. Zhang, W., Ouyang, W., Li, W., Xu, D.: Collaborative and adversarial network for unsupervised domain adaptation. In: Proc. CVPR. pp. 3801–3809 (2018)
43. Zhang, Y., Tang, H., Jia, K.: Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. In: Proceedings of the european conference on computer vision (ECCV). pp. 233–248 (2018)
44. Zhang, Z., Wang, M., Huang, Y., Nehorai, A.: Aligning infinite-dimensional covariance matrices in reproducing kernel hilbert spaces for domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3437–3445 (2018)
45. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. arXiv preprint arXiv:1708.04896 (2017)