Bias Reduction via End-to-End Shift Learning:
Application to Citizen Science

Di Chen
di@cs.cornell.edu
Cornell University

Carla P. Gomes
gomes@cs.cornell.edu
Cornell University

Abstract

Citizen science projects are successful at gathering rich datasets for various applications. However, the data collected by citizen scientists are often biased — in particular, aligned more with the citizens' preferences than with scientific objectives. We propose the Shift Compensation Network (SCN), an end-to-end learning scheme which learns the shift from the scientific objectives to the biased data while compensating for the shift by re-weighting the training data. Applied to bird observational data from the citizen science project *eBird*, we demonstrate how SCN quantifies the data distribution shift and outperforms supervised learning models that do not address the data bias. Compared with competing models in the context of covariate shift, we further demonstrate the advantage of SCN in both its effectiveness and its capability of handling massive high-dimensional data.

Introduction

Citizen science projects (Sullivan et al. 2014; Larrivée et al. 2014; Seibert et al. 2017) play a critical role in collecting rich datasets for scientific research, especially in computational sustainability (Gomes 2009), because they offer an effective low-cost way to collect large datasets for non-commercial research. The success of these projects depends heavily on the public's intrinsic motivations as well as the enjoyment of the participants, which engages them to volunteer their efforts (Bonney et al. 2009). Therefore, citizen science projects usually have few restrictions, providing as much freedom as possible to engage volunteers, so that they can decide where, when, and how to collect data, based on their interests. As a result, the data collected by volunteers are often biased, and align more with their personal preferences, instead of providing systematic observations across various experimental settings. For example, personal convenience has a significant impact on the data collection process, since the participants contribute their time and effort voluntarily. Consequently, most data are collected in or near urban areas and along major roads. On the other hand, most machine learning algorithms are constructed under the assumption that the training data are governed by the same data distribution as that on which the model will later be tested. As a result, the model trained with biased data would perform poorly when it is
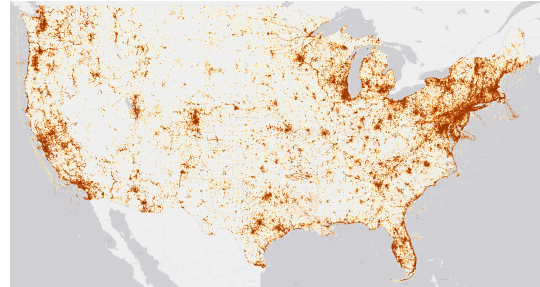
Figure 1: Highly biased distribution of *eBird* observations in the continental U.S. Submissions are concentrated in or near urban areas and along major roads.

evaluated with unbiased test data designed for the scientific objectives.

Incentive mechanisms to shift the efforts of volunteers into the more unexplored areas have been proposed (Xue et al. 2016), in order to improve the scientific quality of the data. However, the scalability of those mechanisms is limited by the budget, and it takes a long time to realize the payback. Furthermore, the type of locality also restricts the distribution of collected data. For example, it is difficult to incentivize volunteers to go to remote places, such as deserts or primal forests, to collect data. Therefore, a tactical learning scheme is needed to bridge the gap between biased data and the desired scientific objectives.

In general, given only the labeled training data (collected by volunteers) and the unlabeled test data (designed for evaluating the scientific objectives), we set out to: (i) learn the shift between the training data distribution P (associated with PDF $p(\mathbf{x}, y)$) and the test data distribution Q (associated with PDF $q(\mathbf{x}, y)$), and (ii) compensate for that shift so that the model will perform well on the test data. To achieve these objectives, we needed to make assumptions on how to bring the training distribution into alignment with the test distribution. Two candidates are *covariate shift* (Bickel, Brückner, and Scheffer 2009), where $p(y|\mathbf{x}) = q(y|\mathbf{x})$, and *label shift* (Lipton, Wang, and Smola 2018), where $p(\mathbf{x}|y) = q(\mathbf{x}|y)$. Motivated by the field observations in the *eBird* project, where the habitat preference $p(y|\mathbf{x})$ of a given species remains the same throughout a season, while the occurrence records $p(\mathbf{x}|y)$

vary significantly because of the volunteers' preferences, we focused on the *covariate shift* setting. Informally, covariate shift captures the change in the distribution of the feature (covariate) vector $\mathbf{x}$. Formally, under covariate shift, we can factor the distributions as follows:

$$p(\mathbf{x}, y) = p(y|\mathbf{x})p(\mathbf{x})$$
$$q(\mathbf{x}, y) = q(y|\mathbf{x})q(\mathbf{x})$$
$$p(y|\mathbf{x}) = q(y|\mathbf{x}) \implies \frac{q(\mathbf{x}, y)}{p(\mathbf{x}, y)} = \frac{q(\mathbf{x})}{p(\mathbf{x})} \quad (1)$$

Thus we were able to learn the shift from P to Q and correct our model by quantifying the test-to-training **shift factor** $q(\mathbf{x})/p(\mathbf{x})$.

Our contribution is an end-to-end learning scheme, which we call the Shift Compensation Network (SCN), that estimates the shift factor while re-weighting the training data to correct the model. Specifically, SCN (i) estimates the shift factor by learning a discriminator that distinguishes between the samples drawn from the training distribution and those drawn from the test distribution, and (ii) aligns the mean of the weighted feature space of the training data with the feature space of the test data, which guides the discriminator to improve the quality of the shift factor. Given the shift factor learned from the discriminator, SCN also compensates for the shift by re-weighting the training samples obtained in the training process to optimize classification loss under the test distribution. We worked with data from *eBird* (Sullivan et al. 2014), which is the world's largest biodiversity-related citizen science project. Applying SCN to the *eBird* observational data, we demonstrate that it significantly improves multi-species distribution modeling by detecting and correcting for the data bias, thereby providing a better approach for monitoring species distribution as well as for inferring migration changes and global climate fluctuation. We further demonstrate the advantage of combining the power of discriminative learning and feature space matching, by showing that SCN outperforms all competing models in our experiments.

Preliminaries

Notation
We use $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y \in \mathcal{Y} = \{0, 1, ..., l\}$ for the feature and label variables. For ease of notation, we use P and Q for the training data and test data distributions, respectively, defined on $\mathcal{X} \times \mathcal{Y}$. We use $p(\mathbf{x}, y)$ and $q(\mathbf{x}, y)$ for the probability density functions (PDFs) associated with P and Q, respectively, and $p(\mathbf{x})$ and $q(\mathbf{x})$ for the marginal PDFs of P and Q.

Problem Setting
We have labeled training data $D_P = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) ..., (\mathbf{x}_n, y_n)\}$ drawn iid from a training distribution P and unlabeled test data $D_Q = \{\mathbf{x}'_1; \mathbf{x}'_2; ...; \mathbf{x}'_n\}$ drawn iid from a test distribution Q, where P denotes the data collected by volunteers and Q denotes the data designed for evaluation of the scientific objectives. Our goal is to yield good predictions for samples drawn from Q. Furthermore, we make the following (realistic) assumptions:
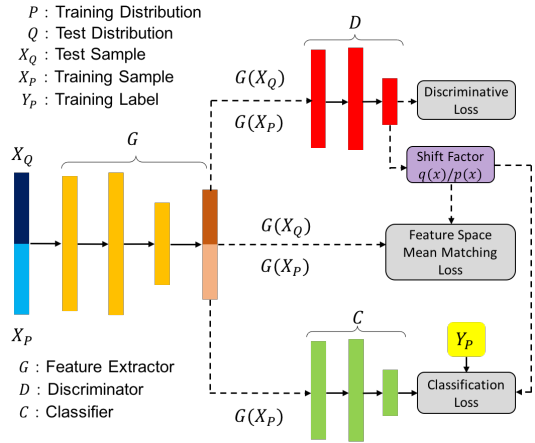

Figure 2: Overview of the Shift Compensation Network

- $p(y|\mathbf{x}) = q(y|\mathbf{x})$
- $p(\mathbf{x}) > 0$ for every $\mathbf{x} \in \mathcal{X}$ with $q(\mathbf{x}) > 0$.

The first assumption expresses the use of *covariate shift*, which is consistent with the field observations in the *eBird* project. The second assumption ensures that the support of P contains the support of Q; without this assumption, this task would not be feasible, as there would be a lack of information on some samples $\mathbf{x}$.

As illustrated in (Shimodaira 2000), in the covariate shift setting the loss $\ell(f(\mathbf{x}), y)$ on the test distribution Q can be minimized by re-weighting the loss on the training distribution P with the shift factor $q(\mathbf{x})/p(\mathbf{x})$, that is,

$$\mathbb{E}_{(\mathbf{x},y) \sim Q}[\ell(f(\mathbf{x}), y)] = \mathbb{E}_{(\mathbf{x},y) \sim P}\left[\ell(f(\mathbf{x}), y)\frac{q(\mathbf{x})}{p(\mathbf{x})}\right] \quad (2)$$

Therefore, our goal is to estimate the shift factor $q(\mathbf{x})/p(\mathbf{x})$ and correct the model so that it performs well on Q.

End-to-End Shift Learning
Shift Compensation Network
Fig. 2 depicts the end-to-end learning framework implemented in the Shift Compensation Network (SCN). A feature extractor G is first applied to encode both the raw training features X_P and the raw test features X_Q into a high-level feature space. Later, we introduce three different losses to estimate the shift factor $q(\mathbf{x})/p(\mathbf{x})$ and optimize the classification task.

We first introduce a discriminative network (with discriminator D), together with a discriminative loss, to distinguish between the samples coming from the training data and those coming from the test data. Specifically, the discriminator D is learned by maximizing the log-likelihood of distinguishing between samples from the training distribution and those from the test distribution, that is,

$$\mathcal{L}_D = \frac{1}{2}\mathbb{E}_{\mathbf{x} \sim P}[\log(D(G(\mathbf{x})))] + \quad (3)$$
$$\frac{1}{2}\mathbb{E}_{\mathbf{x} \sim Q}[\log(1 - D(G(\mathbf{x})))]$$

Proposition 1 *For any fixed feature extractor G, the optimal discriminator D^* for maximizing $\mathcal{L}_D$ is*

$$D^*(G(\mathbf{x})) = \frac{p(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})}.$$

Thus we can estimate the shift factor $\frac{q(\mathbf{x})}{p(\mathbf{x})}$ by $\frac{1 - D(G(\mathbf{x}))}{D(G(\mathbf{x}))}$.
Proof.

$$D^* = \underset{D}{\arg\max} \tfrac{1}{2}\mathbb{E}_{\mathbf{x}\sim P}[\log(D(G(\mathbf{x})))] + \tfrac{1}{2}\mathbb{E}_{\mathbf{x}\sim Q}[\log(1 - D(G(\mathbf{x})))]$$

$$= \underset{D}{\arg\max} \int p(\mathbf{x})\log(D(G(\mathbf{x}))) + q(\mathbf{x})\log(1 - D(G(\mathbf{x})))d\mathbf{x}$$

$\implies$ *(maximizing the integrand)*

$$D^*(G(\mathbf{x})) = \underset{D(G(\mathbf{x}))}{\arg\max} p(\mathbf{x})\log(D(G(\mathbf{x}))) + q(\mathbf{x})\log(1 - D(G(\mathbf{x})))$$

$\implies$ *(the function $d \to p\log(d) + q\log(1-d)$ achieves its maximum in $(0,1)$ at $\frac{p}{p+q}$)*

$$D^*(G(\mathbf{x})) = \frac{p(\mathbf{x})}{p(\mathbf{x})+q(\mathbf{x})}$$

Our use of a discriminative loss is inspired by the generative adversarial nets (GANs) (Goodfellow et al. 2014), which have been applied to many areas. The fundamental idea of GANs is to train a generator and a discriminator in an adversarial way, where the generator is trying to generate data (e.g., an image) that are as similar to the source data as possible, to fool the discriminator, while the discriminator is trying to distinguish between the generated data and the source data. This idea has recently been used in domain adaptation (Tzeng et al. 2017; Hoffman et al. 2017), where two generators are trained to align the source data and the target data into a common feature space so that the discriminator cannot distinguish them. In contrast to those applications, however, SCN does not have an adversarial training process, where the training and test samples share the same extractor G. In our setting, the training and test distributions share the same feature domain, and they differ only in the frequencies of the sample. Therefore, instead of trying to fool the discriminator, we want the discriminator to distinguish the training samples from the test samples to the greatest extent possible, so that we can infer the shift factor between the two distributions reversely as in Proposition 1.

Use of a feature space mean matching (FSMM) loss comes from the notion of kernel mean matching (KMM) (Huang et al. 2007; Gretton et al. 2009), in which the shift factor $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$ is estimated directly by matching the distributions P and Q in a reproducing kernel Hilbert space (RKHS) $\Phi_{\mathcal{H}} : \mathcal{X} \to \mathcal{F}$, that is,

$$\underset{w}{\text{minimize}} \|\mathbb{E}_{\mathbf{x}\sim Q}[\Phi_{\mathcal{H}}(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi_{\mathcal{H}}(\mathbf{x})]\|_2$$

$$\text{subject to } w(\mathbf{x}) \geq 0 \text{ and } \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})] = 1 \quad (4)$$

Though Gretton (2009) proved that the optimization problem (4) is convex and has a unique global optimal solution $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$, the time complexity of KMM is cubic in the

size of the training dataset, which is prohibitive when dealing with very large datasets. We note that even though we do not use the universal kernel (Steinwart 2002) in an RKHS, $w(\mathbf{x}) = q(\mathbf{x})/p(\mathbf{x})$ still implies that

$$\|\mathbb{E}_{\mathbf{x}\sim Q}[\Phi(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi(\mathbf{x})]\|_2 = 0$$

for any mapping $\Phi(\cdot)$. Therefore, our insight is to replace the $\Phi_{\mathcal{H}(\cdot)}$ with a deep feature extractor $G(\cdot)$ and derive the FSMM loss to further guide the discriminator and improve the quality of $w(\mathbf{x})$.

$$\mathcal{L}_{FSMM} = \|\mathbb{E}_{\mathbf{x}\sim Q}[G(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})G(\mathbf{x})]\|_2$$

$$= \left\|\mathbb{E}_{\mathbf{x}\sim Q}[G(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}\left[\frac{1 - D(G(\mathbf{x}))}{D(G(\mathbf{x}))}G(\mathbf{x})\right]\right\|_2 \quad (5)$$

One advantage of combining the power of $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$ is to prevent overfitting. Specifically, if we were learning the shift factor $w(\mathbf{x})$ by using only $\mathcal{L}_{FSMM}$, we could end up getting some ill-posed weights $w(\mathbf{x})$ which potentially would not even be relevant to $q(\mathbf{x})/p(\mathbf{x})$. This is because the dimensionality of $G(\mathbf{x})$ is usually smaller than the number of training data. Therefore, there could be infinitely many solutions of $w(\mathbf{x})$ that achieve zero loss if we consider minimizing $\mathcal{L}_{FSMM}$ as solving a linear equation. However, with the help of the discriminative loss, which constrains the solution space of equation (5), we are able to get good weights which minimize $\mathcal{L}_{FSMM}$ while preventing overfitting. On the other hand, the feature space mean matching loss also plays the role of a regularizer for the discriminative loss, to prevent the discriminator from distinguishing the two distributions by simply memorizing all the samples. Interestingly, in our experiments, we found that the feature space mean matching loss works well empirically even without the discriminative loss. A detailed comparison will be shown in the Experiments section.

Using the shift factor learned from $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$, we derive the weighted classification loss, that is,

$$\mathcal{L}_C = \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\ell(C(G(\mathbf{x})), y)], \quad (6)$$

where $\ell(\cdot, \cdot)$ is typically the cross-entropy loss. The classification loss $\mathcal{L}_C$ not only is used for the classification task but also ensures that the feature space given by the feature extractor G represents the important characteristics of the raw data.

End-to-End Learning for SCN

One straightforward way to train SCN is to use mini-batch stochastic gradient descent (SGD) for optimization. However, the feature space mean matching loss $\mathcal{L}_{FSMM}$ could have a large variance with small batch sizes. For example, if the two sampled batches X_P and X_Q have very few similar features $G(\mathbf{x}_i)$, the $\mathcal{L}_{FSMM}$ could be very large even with the optimal shift factor. Therefore, instead of estimating $\mathcal{L}_{FSMM}$ based on each mini batch, we maintain the moving averages of both the weighted training features M_P and the test features M_Q. Algorithm 1 shows the pseudocode of our two-step learning scheme for SCN. In the first step, we update the moving averages of both the training data and the test data using the features extracted by G with hyperparameter

Algorithm 1 Two-step learning in iteration t for SCN

Input: X_P and X_Q are raw features sampled iid from the training distribution P and the test distribution Q; Y_P is the label corresponding to X_P; M_P^{t-1} and M_Q^{t-1} are the moving averages from the previous iteration.

Step 1
1: $m_Q^t = \sum_{\mathbf{x}_i \in X_Q} G(\mathbf{x}_i)/|X_Q|$
2: $m_P^t = \sum_{\mathbf{x}_j \in X_P} \frac{1-D(G(\mathbf{x}_j))}{D(G(\mathbf{x}_j))} G(\mathbf{x}_j)/|X_P|$
3: $M_Q^t \leftarrow \alpha M_Q^{t-1} + (1-\alpha)m_Q^t$
4: $M_P^t \leftarrow \alpha M_P^{t-1} + (1-\alpha)m_P^t$
5: $\widehat{M_Q^t} \leftarrow M_Q^t/(1-\alpha^t)$
6: $\widehat{M_P^t} \leftarrow M_P^t/(1-\alpha^t)$
7: $\mathcal{L}_{FSMM} \leftarrow \left\| \widehat{M_Q^t} - \widehat{M_P^t} \right\|_2$
8: $\mathcal{L}_D \leftarrow \frac{\sum_{\mathbf{x}_j \in X_P} \log(D(G(\mathbf{x}_j)))}{2|X_P|} + \frac{\sum_{\mathbf{x}_i \in X_Q} \log(1-D(G(\mathbf{x}_i)))}{2|X_Q|}$
9: Update the discriminator D and the feature extractor G by ascending along the gradients:

$\nabla_{\theta_D}(\lambda_1 \mathcal{L}_D - \lambda_2 \mathcal{L}_{FSMM})$ and $\nabla_{\theta_G}(\lambda_1 \mathcal{L}_D - \lambda_2 \mathcal{L}_{FSMM})$

Step 2
10: For $\mathbf{x}_j \in X_P$, $w(\mathbf{x}_j) \leftarrow \frac{1-D(G(\mathbf{x}_j))}{D(G(\mathbf{x}_j))}$
11: $\mathcal{L}_C = \frac{\sum_{\mathbf{x}_j \in X_P} w(\mathbf{x}_j)\ell(C(G(\mathbf{x})),y)}{|X_P|}$
12: Update the classifier C and the feature extractor G by ascending along the gradients:

$\nabla_{\theta_C}\mathcal{L}_C$ and $\nabla_{\theta_G}\mathcal{L}_C$

Here we ignore the gradients coming from the weights $w(\mathbf{x}_j)$, that is, we consider the $w(\mathbf{x}_j)$ as constants.

α. Then we use the losses $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$ to update the parameters in the feature extractor G and the discriminator D with hyperparameters λ_1 and λ_2, respectively, which adjusts the importance of $\mathcal{L}_D$ and $\mathcal{L}_{FSMM}$. (We set $\lambda_1 = 1$ and $\lambda_2 = 0.1$ in our experiments.) In the second step, we update the classifier C and the feature extractor G using the estimated shift factor $w(\mathbf{x})$ from the first step. We initialize the moving averages M_P and M_Q to 0, so that operations (5) and (6) in Algorithm (1) are applied to compensate for the bias caused by initialization to 0, that is,

$$\mathbb{E}[M_Q^t] = \mathbb{E}[\alpha M_Q^{t-1} + (1-\alpha)m_Q^t]$$
$$= \sum_{i=1}^{t} (1-\alpha)\alpha^{i-1}\mathbb{E}[m_Q^i]$$
$$\approx \widetilde{\mathbb{E}}[m_Q^i](1-\alpha^t) \qquad (7)$$

Further, since the mini batches are drawn independently, we show that

$$\text{Var}[M_Q^t] = \text{Var}[\alpha M_Q^{t-1} + (1-\alpha)m_Q^t]$$
$$= \sum_{i=1}^{t} (1-\alpha)^2 \alpha^{2i-2}\text{Var}[m_Q^i]$$
$$\approx \widetilde{\text{Var}}[m_Q^i]\frac{1-\alpha}{1+\alpha}(1-\alpha^{2t}) \qquad (8)$$

That is, by using moving-average estimation, the variance can be reduced by approximately $\frac{1-\alpha}{1+\alpha}$. Consequently, we can apply an α close to 1 to significantly reduce the variance of $\mathcal{L}_{FSMM}$. However, an α close to 1 implies a strong momentum which is too high for the early-stage training. Empirically, we chose $\alpha = 0.9$ in our experiments.

In the second step of the training process, the shift factor $w(\mathbf{x})$ is used to compensate for the bias between training and test. Note that we must consider the shift factor as a constant instead of as a function of the discriminator D. Otherwise, minimizing the classification loss $\mathcal{L}_C$ would end up trivially causing all the $w(\mathbf{x})$ to be reduced to zero.

Related Work

Different approaches for reducing the data bias problem have been proposed. In mechanism design, Xue et al. (2016) proposed a two-stage game for providing incentives to shift the efforts of volunteers to more unexplored tasks in order to improve the scientific quality of the data. In ecology, Phillips et al. (2009) improved the modeling of presence-only data by aligning the biases of both training data and background samples. Fithian et al. (2015) explored the complementary strengths of doing a joint analysis of data coming from different sources to reduce the bias. In domain adaptation, various methods (Jiang and Zhai 2007; Shinnou, Sasaki, and Komiya 2015; Tzeng et al. 2017; Hoffman et al. 2017) have been proposed to reduce the bias between the source domain and the target domain by mapping them to a common feature space while reserving the critical characteristics.

Our work is most closely related to the approaches of (Zadrozny 2004; Huang et al. 2007; Sugiyama et al. 2008; Gretton et al. 2009) developed under the names of *covariate shift* and *sample selection bias*, where the shift factor is learned in order to align the training distribution with the test distribution. The earliest work in this domain came from the statistics and econometrics communities, where they addressed the use of non-random samples to estimate behavior. Heckman (1977) addressed sample selection bias, and Manski and Lerman (1977) investigated estimating parameters under *choice-based bias*, cases that are analogous to a shift in the data distribution. Later, (Shimodaira 2000) proposed correcting models via weighting of samples in empirical risk minimization (ERM) by the shift factor $q(\mathbf{x})/p(\mathbf{x})$.

One straightforward approach to learning the weights is to directly estimate the distributions $p(\mathbf{x})$ and $q(\mathbf{x})$ from the training and test data respectively, using kernel density estimation (Shimodaira 2000; Sugiyama and Müller 2005). However, learning the data distribution $p(\mathbf{x})$ is intrinsically model based and performs poorly with high-dimensional data. Huang et al. (2007) and Gretton et al. (2009) proposed kernel mean matching (KMM), which estimates the shift factor $w(\mathbf{x}) = q(\mathbf{x})/p(\mathbf{x})$ directly via matching the first moment of the covariate distributions of the training and test data in a reproducing kernel Hilbert space (RKHS) using quadratic programming. KLIEP (Sugiyama et al. 2008) estimates $w(\mathbf{x})$ by minimizing the Kullback-Leibler (KL) divergence between the test distribution and the weighted training distribution.

Later, Tsuboi et al. (2009) derived an extension of KLIEP for applications with a large test set and revealed a close relationship of that approach to kernel mean matching. Also, Rosenbaum and Rubin (1983) and Lunceford and Davidian (2004) introduced propensity scoring to design unbiased experiments, which they applied in settings related to sample selection bias.

While the problem of covariate shift has received much attention in the past, it has been used mainly in settings where the size of the dataset is relatively small and the dimensionality of the data is relatively low. Therefore, it has not been adequately addressed in settings with massive high-dimensional data, such as hundreds of thousands of high-resolution images. Among the studies in this area, (Bickel, Brückner, and Scheffer 2009) is the one most closely related to ours. They tackled this task by modeling the sample selection process using Bayesian inference, where the shift factor is learned by modeling the probability that a sample is selected into training data. Though we both use a discriminative model to detect the shift, SCN provides an end-to-end deep learning scheme, where the shift factor and the classification model are learned simultaneously, providing a smoother compensation process, which has considerable advantages for work with massive high-dimensional data and deep learning models. In addition, SCN introduces the feature space mean matching loss, which further improves the quality of the shift factor and leads to a better predictive performance. For the sake of fairness, we adapted the work of (Bickel, Brückner, and Scheffer 2009) to the deep learning context in our experiments.

Experiments

Datasets and Implementation Details

We worked with a crowd-sourced bird observation dataset from the successful citizen science project *eBird* (Sullivan et al. 2014), which is the world's largest biodiversity-related citizen science project, with more than 100 million bird sightings contributed each year by eBirders around the world. Even though *eBird* amasses large amounts of citizen science data, the locations of the collected observation records are highly concentrated in urban areas and along major roads, as shown in Fig. 1. This hinders our understanding of species distribution as well as inference of migration changes and global climate fluctuation. Therefore, we evaluated our SCN [1] approach by measuring how we could improve multi-species distribution modeling given biased observational data.

One record in the *eBird* dataset is referred to as a checklist, in which the bird observer records all the species he/she detects as well as the time and the geographical location of the observation site. Crossed with the National Land Cover Dataset for the U.S. (NLCD) (Homer et al. 2015), we obtained a 16-dimensional feature vector for each observation site, which describes the landscape composition with respect to 16 different land types such as water and forest. We also collected satellite images for each observation site by matching the geographical location of a site to Google Earth, where several preprocesses have been conducted, including cloud

---

[1]Code to reproduce the experiments can be found at https://bitbucket.org/DiChen9412/aaai2019-scn/.

Feature Type	NLCD	Google Earth Image
Dimensionality	16	$256 \times 256 \times 3$
#Training Set	79060	79060
#Validation Set	10959	10959
#Test Set	10959	10959
#Labels	50	50

Table 1: Statistics of the *eBird* dataset

removal. Each satellite image covers an area of 17.8 km^2 near the observation site and has 256×256 pixels. The dataset for our experiments was formed by using all the observation checklists from Bird Conservation Regions (BCRs) 13 and 14 in May from 2002 to 2014, which contains 100,978 observations (Committee and others 2000). May is a migration period for BCR 13 and 14; therefore a lot of non-native birds pass over this region, which gives us excellent opportunities to observe their choice of habitats during the migration. We chose the 50 most frequently observed birds as the target species, which cover over 97.4% of the records in our dataset. Because our goal was to learn and predict multi-species distributions across landscapes, we formed the unbiased test set and the unbiased validation set by overlaying a grid on the map and choosing observation records spatially uniformly. We used the rest of the observations to form the spatially biased training set. Table 1 presents details of the dataset configuration.

In the experiments, we applied two types of neural networks for the feature extractor G: multi-layer fully connected networks (MLPs) and convolutional neural networks (CNNs). For the NLCD features, we used a three-layer fully connected neural network with hidden units of size 512, 1024 and 512, and with ReLU (Nair and Hinton 2010) as the activation function. For the Google Earth images, we used DenseNet (Huang et al. 2017) with minor adjustments to fit the image size. The discriminator D and Classifier C in SCN were all formed by three-layer fully connected networks with hidden units of size 512, 256, and $\#outcome$, and with ReLU as the activation function for the first two layers; there was no activation function for the third layer. For all models in our experiments, the training process was done for 200 epochs, using a batch size of 128, cross-entropy loss, and an Adam optimizer (Kingma and Ba 2014) with a learning rate of 0.0001, and utilized batch normalization (Ioffe and Szegedy 2015), a 0.8 dropout rate (Srivastava et al. 2014), and early stopping to accelerate the training process and prevent overfitting.

Analysis of Performance of the SCN

We compared the performance of SCN with baseline models from two different groups. The first group included *models that ignore the covariate shift* (which we refer to as vanilla models), that is, models are trained directly by using batches sampled uniformly from the training set without correcting for the bias. The second group included *different competitive models for solving the covariate shift problem*: (1) kernel density estimation (KDE) methods (Shimodaira 2000; Sugiyama and Müller 2005), (2) the Kullback-Leibler Importance Estimation Procedure (KLIEP) (Sugiyama et al. 2008),

NLCD Feature			
Test Metrics (%)	AUC	AP	F1 score
vanilla model	77.86	63.31	54.90
SCN	**80.34**	**66.17**	**57.06**
KLIEP	78.87	64.33	55.63
KDE	78.96	64.42	55.27
DFW	79.38	64.98	55.79
Google Earth Image			
vanilla model	80.93	67.33	59.97
SCN	**83.80**	**70.39**	**62.37**
KLIEP	81.17	67.86	60.23
KDE	80.95	67.42	60.01
DFW	81.99	68.44	60.77

Table 2: Comparison of predictive performance of different methods under three different metrics. (The larger, the better.)

and (3) discriminative factor weighting (DFW) (Bickel, Brückner, and Scheffer 2009). The DFW method was implemented initially by using a Bayesian model, which we adapted to the deep learning model in order to use it with the $eBird$ dataset. We did not compare SCN with the kernel mean matching (KMM) methods (Huang et al. 2007; Gretton et al. 2009), because KMM, like many kernel methods, requires the construction and inversion of an $n \times n$ Gram matrix, which has a complexity of $\mathcal{O}(n^3)$. This hinders its application to real-life applications, where the value of n will often be in the hundreds of thousands. In our experiments, we found that the largest n for which we could feasibly run the KMM code is roughly 10,000 (even with SGD), which is only 10% of our dataset. To make a fair comparison, we did a grid search for the hyperparameters of all the baseline models to saturate their performance. Moreover, the structure of the networks for the feature extractor and the classifier used in all the baseline models, were the same as those in our SCN (i.e., G and C), while the shift factors for those models were learned using their methods.

Table 2 shows the average performance of SCN and other baseline models with respect to three different metrics: (1) **AUC**, area under the ROC curve; (2) **AP**, area under the precision–recall curve; (3) **F1 score**, the harmonic mean of precision and recall. Because our task is a multi-label classification problem, these three metrics were averaged over all 50 species in the datasets. In our experiments, the standard error of all the models was less than 0.2% under all three metrics. There are two key results in Table 2: **(1) All bias-correction models outperformed the vanilla models under all metrics, which shows a significant advantage of correcting for the covariate shift. (2) SCN outperformed all the other bias-correcting models, especially on high-dimensional Google Earth images.**

The kernel density estimation (KDE) models had the worst performance, especially on Google Earth images. This is not only because of the difficulty of modeling high-dimensional data distributions, but also because of the sensitivity of the KDE approach. When $p(\mathbf{x}) \ll q(\mathbf{x})$, a tiny perturbation of $p(\mathbf{x})$ could result in a huge fluctuation in the shift factor $q(\mathbf{x})/p(\mathbf{x})$. KLIEP performed slightly better than KDE, by learning the shift factor $w(\mathbf{x}) = \frac{q(\mathbf{x})}{p(\mathbf{x})}$ directly, where it
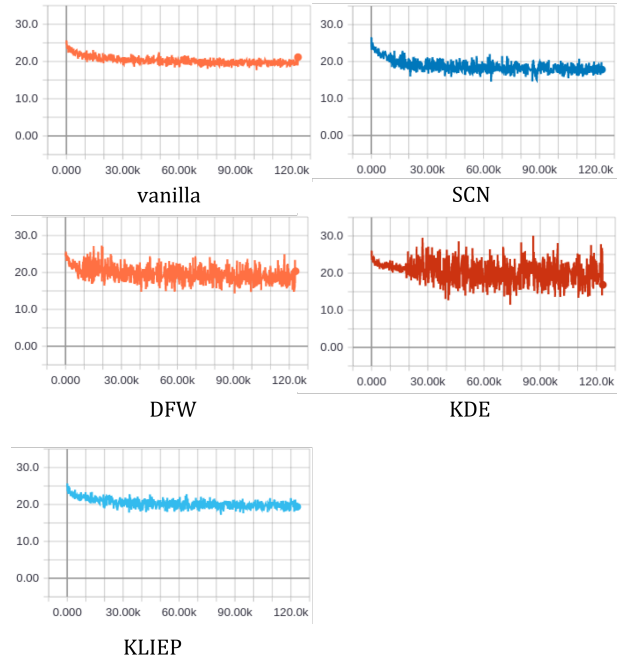

Figure 3: The learning curves of all models. The vertical axis shows the cross-entropy loss, and the horizontal axis shows the number of iterations.

minimized the KL divergence between the weighted training distribution and the test distribution. However, it showed only a slight improvement over the vanilla models on Google Earth images. DFW performed better than the other two baseline models, which is not surprising, given that DFW learns the shift factor by using a discriminator similar to the one in SCN. SCN outperformed DFW not only because it uses an additional loss, the feature space mean matching loss, but also because of its end-to-end training process. DFW first learns the shift factor by optimizing the discriminator, and then it trains the classification model using samples weighted by the shift factor. However, SCN learns the classifier C and the discriminator D simultaneously, where the weighted training distribution approaches the test distribution smoothly through the training process, which performed better empirically than directly adding the optimized weights to the training samples. Wang et al. (2018) also discovered a similar phenomenon in cost-sensitive learning, where pre-training of the neural network with unweighted samples significantly improved the model performance. One possible explanation of this phenomenon is that the training of deep neural networks depends highly on mini-batch SGD, so that the fluctuation of gradients caused by the weights may hinder the stability of the training process, especially during the early stage of the training. Fig. 3 shows the learning curves of all five models, where we used the same batch size and an Adam optimizer with the same learning rate. As seen there, SCN had a milder oscillation curve than DFW, which is consistent with the conjecture we stated earlier. In our experiments, we pre-trained the baseline models with unweighted samples for 20 epochs in order to achieve a stable initialization. Otherwise, some of them

NLCD Feature			
Test Metrics (%)	AUC	AP	F1-score
SCN	**80.34**	**66.17**	**57.06**
SCN_D	79.53	65.11	56.11
SCN_FSMM	79.58	65.17	56.26
SCN$^-$	80.09	65.97	56.83
Google Earth Image			
SCN	**83.80**	**70.39**	**62.37**
SCN_D	82.35	68.96	61.23
SCN_FSMM	82.49	69.05	61.51
SCN$^-$	83.44	69.72	62.01

Table 3: Comparison of predictive performance of the different variants of SCN

would end up falling into a bad local minimum, where they would perform even worse than the vanilla models.

To further explore the functionality of the discriminative loss and the feature mean matching loss in SCN, we implemented several variants of the original SCN model:

- SCN: The original Shift Compensation Network

- SCN_D: The Shift Compensation Network without the feature space mean matching loss ($\lambda_2 = 0$)

- SCN_FSMM: The Shift Compensation Network without the discriminative loss ($\lambda_1 = 0$)

- SCN$^-$: The Shift Compensation Network without using moving-average estimation for the feature space mean matching loss ($\alpha = 0$)

Table 3 compares the performance of the different variants of SCN, where we observe the following: (1) Both the discriminative loss and the feature space mean matching loss play an important role in learning the shift factor. (2) The moving-average estimation for the feature space mean matching loss shows an advantage over the batch-wise estimation (compare SCN to SCN$^-$). (3) Crossed with Table 2, SCN performs better than DFW, even with only the discriminative loss, which shows the benefit of fitting the shift factor gradually through the training process. (4) Surprisingly, even if we use only the feature space mean matching loss, which would potentially lead to ill-posed weights, SCN_FSMM still shows much better performance than the other baselines.

Shift Factor Analysis

We visualized the heatmap of the observation records for the month of May in New York State (Fig. 4), where the left panel shows the distribution of the original samples and the right one shows the distribution weighted with the shift factor. The colors from white to brown indicates the sample popularity from low to high using a logarithmic scale from 1 to 256. As seen there, the original samples are concentrated in the southeastern portion and Long Island, while the weighted one is more balanced over the whole state after applying the shift factor. This illustrates that SCN learns the shift correctly and provides a more balanced sample distribution by compensating for the shift.

We investigated the shift factors learned from the different models (Table 4) by analyzing the ratio of the feature space
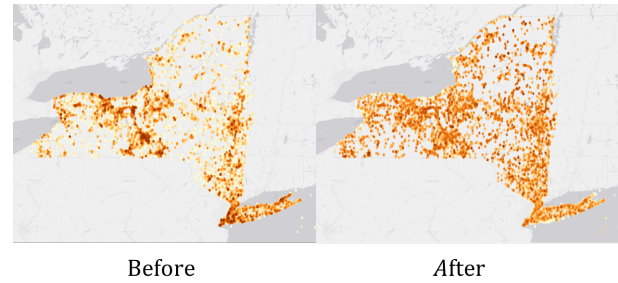

Figure 4: Heatmap of the observation records for the month of May in New York State, where the left panel shows the distribution of the original samples and the right one shows the distribution weighted with the shift factor

Averaged Feature Space Mean Matching Loss	
vanilla model	0.8006
SCN	0.0182
KLIEP	0.0015
KDE	0.0028
DFW	0.0109

Table 4: Feature space discrepancy between the weighted training data and the test data

mean matching loss to the dimensionality of the feature space using equation (9).

$$\|\mathbb{E}_{\mathbf{x}\sim Q}[\Phi(\mathbf{x})] - \mathbb{E}_{\mathbf{x}\sim P}[w(\mathbf{x})\Phi(\mathbf{x})]\|_2 / \dim(\Phi(\mathbf{x}))$$
$$\approx \left\| \frac{1}{|X_Q|}\sum_i \Phi(\mathbf{x}_i) - \frac{1}{|X_P|}\sum_i w(\mathbf{x}_i)\Phi(\mathbf{x}_i) \right\|_2 / \dim(\Phi(\mathbf{x})) \tag{9}$$

Here, we chose the output of the feature extractor in each model (such as $G(\mathbf{x})$ in SCN) as the feature space $\Phi(\mathbf{x})$. Compared to the vanilla models, all the shift correction models significantly reduced the discrepancy between the weighted training distribution and the test distribution. However, crossed with Table 2, it is interesting to see the counterintuitive result that the models with the smaller feature space discrepancies (KDE & KLIEP) did not necessarily perform better.

Conclusion

In this paper, we proposed the Shift Compensation Network (SCN) along with an end-to-end learning scheme for solving the covariate shift problem in citizen science. We incorporated the discriminative loss and the feature space mean matching loss to learn the shift factor. Tested on a real-world biodiversity-related citizen science project, *eBird*, we show how SCN significantly improves multi-species distribution modeling by learning and correcting for the data bias, and that it consistently performs better than previous models. We also discovered the importance of fitting the shift factor gradually through the training process, which raises an interesting question for future research: How do the weights affect the performance of models learned by stochastic gradient descent? Future directions include exploring the best way to learn and apply shift factors in deep learning models.

Acknowledgments

References

Bickel, S.; Brückner, M.; and Scheffer, T. 2009. Discriminative learning under covariate shift. *Journal of Machine Learning Research* 10(Sep):2137–2155.

Bonney, R.; Cooper, C. B.; Dickinson, J.; Kelling, S.; Phillips, T.; Rosenberg, K. V.; and Shirk, J. 2009. Citizen science: a developing tool for expanding science knowledge and scientific literacy. *BioScience* 59(11):977–984.

Committee, U. N., et al. 2000. North american bird conservation initiative: Bird conservation region descriptions, a supplement to the north american bird conservation initiative bird conservation regions map.

Fithian, W.; Elith, J.; Hastie, T.; and Keith, D. A. 2015. Bias correction in species distribution models: pooling survey and collection data for multiple species. *Methods in Ecology and Evolution*.

Gomes, C. P. 2009. Computational sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge* 39(4):5–13.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Gretton, A.; Smola, A. J.; Huang, J.; Schmittfull, M.; Borgwardt, K. M.; and Schölkopf, B. 2009. Covariate shift by kernel mean matching.

Heckman, J. J. 1977. Sample selection bias as a specification error (with an application to the estimation of labor supply functions).

Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.-Y.; Isola, P.; Saenko, K.; Efros, A. A.; and Darrell, T. 2017. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*.

Homer, C.; Dewitz, J.; Yang, L.; Jin, S.; Danielson, P.; Xian, G.; Coulston, J.; Herold, N.; Wickham, J.; and Megown, K. 2015. Completion of the 2011 national land cover database for the conterminous united states–representing a decade of land cover change information. *Photogrammetric Engineering & Remote Sensing*.

Huang, J.; Gretton, A.; Borgwardt, K. M.; Schölkopf, B.; and Smola, A. J. 2007. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, 601–608.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*, volume 1, 3.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 448–456.

Jiang, J., and Zhai, C. 2007. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, 264–271.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Larrivée, M.; Prudic, K. L.; McFarland, K.; and Kerr, J. 2014. ebutterfly: a citizen-based butterfly database in the biological sciences.

Lipton, Z. C.; Wang, Y.-X.; and Smola, A. 2018. Detecting and correcting for label shift with black box predictors. *arXiv preprint*.

Lunceford, J. K., and Davidian, M. 2004. Stratification and weighting via the propensity score in estimation of causal treatment effects: a comparative study. *Statistics in medicine* 23(19):2937–2960.

Manski, C. F., and Lerman, S. R. 1977. The estimation of choice probabilities from choice based samples. *Econometrica: Journal of the Econometric Society* 1977–1988.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814.

Phillips, S. J.; Dudík, M.; Elith, J.; Graham, C. H.; Lehmann, A.; Leathwick, J.; and Ferrier, S. 2009. Sample selection bias and presence-only distribution models: implications for background and pseudo-absence data. *Ecological applications* 19(1):181–197.

Rosenbaum, P. R., and Rubin, D. B. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70(1):41–55.

Seibert, J.; Strobl, B.; Etter, S.; Vis, M.; Ewen, T.; and van Meerveld, H. 2017. Engaging the public in hydrological observations-first experiences from the crowdwater project. In *EGU General Assembly Conference Abstracts*, volume 19, 11592.

Shimodaira, H. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference* 90(2):227–244.

Shinnou, H.; Sasaki, M.; and Komiya, K. 2015. Learning under covariate shift for domain adaptation for word sense disambiguation. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation: Posters*, 215–223.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*.

Steinwart, I. 2002. Support vector machines are universally consistent. *Journal of Complexity* 18(3):768–791.

Sugiyama, M., and Müller, K.-R. 2005. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*.

Sugiyama, M.; Nakajima, S.; Kashima, H.; Buenau, P. V.; and Kawanabe, M. 2008. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, 1433–1440.

Sullivan, B. L.; Aycrigg, J. L.; Barry, J. H.; Bonney, R. E.; Bruns, N.; Cooper, C. B.; Damoulas, T.; Dhondt, A. A.; Dietterich, T.; Farnsworth, A.; et al. 2014. The ebird enterprise: an integrated approach to development and application of citizen science. *Biological Conservation* 169:31–40.

Tsuboi, Y.; Kashima, H.; Hido, S.; Bickel, S.; and Sugiyama, M. 2009. Direct density ratio estimation for large-scale covariate shift adaptation. *Journal of Information Processing* 17:138–155.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, 4.

Wang, L.; Xu, Q.; De Sa, C.; and Joachims, T. 2018. Cost-sensitive learning via deep policy erm.

Xue, Y.; Davies, I.; Fink, D.; Wood, C.; and Gomes, C. P. 2016. Avicaching: A two stage game for bias reduction in citizen science. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 776–785. International Foundation for Autonomous Agents and Multiagent Systems.

Zadrozny, B. 2004. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, 114. ACM.