

Generality-Training of a Classifier for Improved Calibration in Unseen Contexts

Bhawani Shankar Leelar(✉) and Meelis Kull

Institute of Computer Science, University of Tartu, Tartu, Estonia
{bhawani.shankar.leelar,meelis.kull}@ut.ee

Abstract. Artificial neural networks tend to output class probabilities that are miscalibrated, *i.e.*, their reported uncertainty is not a very good indicator of how much we should trust the model. Consequently, methods have been developed to improve the model's predictive uncertainty, both during training and post-hoc. Even if the model is calibrated on the domain used in training, it typically becomes over-confident when applied on slightly different target domains, e.g. due to perturbations or shifts in the data. The model can be recalibrated for a fixed list of target domains, but its performance can still be poor on unseen target domains. To address this issue, we propose a generality-training procedure that learns a modified head for the neural network to achieve better calibration generalization to new domains while retaining calibration performance on the given domains. This generality-head is trained on multiple domains using a new objective function with increased emphasis on the calibration loss compared to cross-entropy. Such training results in a more general model in the sense of not only better calibration but also better accuracy on unseen domains, as we demonstrate experimentally on multiple datasets. The code and supplementary for the paper is available (https://github.com/bsl-traveller/CaliGen.git).

Keywords: classifier calibration · distribution shift · out-of-distribution generalization

1 Introduction

Deep Neural Network (DNNs) typically produce predictions that are not calibrated, which means their predicted probabilities express confidence levels that are not reflected in accuracy. Calibration methods improve the calibration of their predictive uncertainty both during model training [1–5] and post-hoc [6–17]. These calibration methods still tend to be over-confident under distribution shift because of distribution disparity between the source (or pre-shift) and target (or post-shift) domains [18–21]. Standard methods of domain adaptation

This work was supported by the Estonian Research Council grant PRG1604 and the European Social Fund via IT Academy programme.

© The Author(s) 2023
D. Koutra et al. (Eds.): ECML PKDD 2023, LNAI 14173, pp. 374–391, 2023.
https://doi.org/10.1007/978-3-031-43424-2_23

and transfer learning [22] can offer some but limited help because of focusing on prediction accuracy and not on calibration [23].

Table 1. Comparison between calibration generalization and related calibration paradigm

Calibration paradigm	Calibration data	Test data
Single-domain calibration	$\mathcal{D}^{source}$	$\mathcal{D}^{source}$
Multi-domain calibration	$\mathcal{D}^1, \ldots, \mathcal{D}^n$	$\mathcal{D}^1, \ldots, \mathcal{D}^n$
Calibration transfer/adaptation	$\mathcal{D}^{source}, \mathcal{D}^{target}$	$\mathcal{D}^{target}$
Calibration generalization	$\mathcal{D}^1, \ldots, \mathcal{D}^n$	$\mathcal{D}^{n+1}$

This issue is addressed in recent research about calibration across multiple domains [19–21,24–27] where the goal is to obtain calibrated probabilities in the target domain(s) using information from the source domain(s). More precisely, these methods address several different but related tasks, which we propose to categorize as follows, building on the categorization by Wang et al. [28] (see Table 1): (1) *single-domain calibration*, *i.e.*, the classical task of learning calibrated uncertainty estimates in a single domain without any shift involved; (2) *multi-domain calibration* with the goal of learning calibrated predictions for multiple domains by using some labeled data from each of these domains during learning; (3) *calibration transfer or adaptation* where a calibration learned on source domain can be transferred (might lose the calibration on source domain) or adapted (preserves calibration on source domain) to a target domain with the help of some labeled or unlabelled samples from the target domain during learning [19,20,25,27]; and (4) *calibration generalization* where there are no data available from the target domain during the learning phase, and hence the model is faced with test data from a previously unseen domain, typically a variation of the seen domain(s) due to a slight distribution shift, some perturbations to the data or a context change [21,24].

We focus on calibration generalization, and in particular on the same scenario as Gong et al. [21] where the goal is to provide calibrated predictions in an unseen domain under the assumption of having access to the following resources: (a) a model trained on training (source) domains; (b) labeled data in several calibration domains, helping to prepare for the unseen domain; (c) access to the representation layer (*i.e.*, latent space features) of the model, helping to relate test instances of the unseen domain to the training and calibration domains. In other words, the goal is very specific: having access to a model trained on source domains and access to multiple calibration domains, how can we best prepare ourselves for unseen data by using but not modifying the representation layer of the model?

The work of Gong et al. [21] builds on Temperature Scaling (TS) [16], which is a simple and commonly used calibration method where a single temperature parameter is used to adjust the confidence level of the classifier. Instead

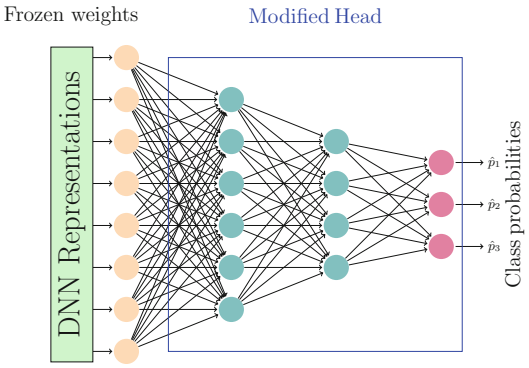


Fig. 1. Modified head of DNN. The network before the Representation layer is fixed, and after the representation layer, 2 dense hidden layers (with dimensions (1024, 512) for DomainNet, and (512, 128) for the other two datasets) and one dense logit layer (equal to the number of classes) are added. We use dropout (0.5), L2 regularizer (0.01), and ReLU activation for all layers with softmax on the last layer

of applying the same temperature on all instances, Gong et al. vary the temperature across instances. The idea is first to cluster the data of all calibration domains jointly and then learn a different temperature for each cluster. Clustering is performed in the representation space, *i.e.*, on the activation vectors of the representation layer instead of the original features. On a test instance, the temperature from the closest cluster is used (a method called *Cluster NN*), or a linear model fitted on cluster centers is used to predict the temperature to be used (a method called *Cluster LR*).

These cluster-based methods have the following limitations. First, they rely on TS to offer a suitable family of calibration transformations, while several newer calibration methods with richer transformation families have been shown to outperform TS [1–3,29]. Second, test-time inference to obtain predictions requires additional computation outside the classifier itself, thus making the solution slightly more complicated technically. While this second point is typically a minor problem, having a single classifier model to provide predictions on test data would still be advantageous.

To address these shortcomings, we propose a novel calibration method CaliGen (Calibration by Generality Training), to improve calibration generalization to unseen domains. We insert additional fully-connected layers to the network on top of the representation layer and train this additional structure on the calibration domains (freezing the representations, *i.e.*, the earlier part of the network, see the illustration in Fig. 1). We propose a custom objective function (see Sect. 4.1) to be used with this training process, which addresses both prediction accuracy and calibration error with a weighted combination of cross-entropy and KL-divergence. We call this process *generality-training* because its goal is to improve the model's generalizability to unseen domains. Our major contributions include the following:

- We propose a novel solution for the simultaneous generalization of a neural network classifier for calibration and accuracy in a domain generalization procedure.
- We propose a novel secondary loss function to be used with cross-entropy to encourage calibration. We have tested it in a calibration generalization setting, but it is open to finding more use cases in similar or different scenarios.
- We provide a theoretical justification for the proposed algorithm and explain its advantage in better generalization.
- We provide experimental results on real-world data to justify its advantage over existing calibration generalization methods. We show that our method generalizes better to unseen target domains with improved accuracy and calibration while maintaining well-calibrated probabilities in the source domains.

The rest of the paper is organized as follows: Sect. 2 discusses related work in calibration and multi-domain calibration. In Sect. 3, we discuss the required background to understand the paper. We propose the method in Sect. 4 and give a theoretical explanation. We discuss datasets and experimental set-up in Sect. 5. In Sect. 6, we discuss the results obtained on 3 datasets with comparison to other state-of-the-art (SOTA) methods along with an ablation study, and in Sect. 7, we give concluding remarks.

2 Related Work

2.1 Calibration

Researchers have proposed many solutions to obtain well-calibrated predictions from neural networks. These solutions can be summarised into three categories. In the first category, the primary training loss is replaced or augmented with a term that explicitly incentives calibration; examples include the AvUC loss [1], MMCE loss [2], Focal loss [3,4] and Cross-entropy loss with Pairwise Constraints [5]. Other examples include Mixup [30], Label smoothing [31], and Label Relaxation [32], which can also be interpreted as modifying the loss function and has been shown to improve calibration.

While the above methods offer to achieve calibrated probabilities during training, solutions using post-hoc calibration methods fall into the second category in which model predictions are transformed after the training by optimizing additional parameters on a held-out validation set [6–17]. One of the most popular techniques in this category is TS [16], however, it is ineffective under distribution shift in certain scenarios [18].

A third category of methods examines model changes such as ensembling multiple predictions [33,34] or multiple priors [29].

2.2 Multi-domain Calibration

All the above works aim to be calibrated in a single domain or context (in-distribution data). Still, studies have shown that existing post-hoc calibration

methods are highly overconfident under domain shift [23,27]. In recent times, researchers have shifted their attention to addressing the issue of calibration under distribution shift and multi-domain calibration [19–21,24–27,35,36]. As calibration on multiple in-distribution domains, multi-domain calibration is also used in fairness [25,35,36]. Recent research on calibration adaptation considers the setting when labels are unavailable from the target domain [19,20,25,27]. However, a more challenging task of calibration generalization, when no samples from target domains are available during training or calibration, is considered by [21,24]. In particular, Wald et al. [24] extended Isotonic Regression to the multi-domain setting, which takes predictions of a trained model on validation data pooled from training domains.

Gong et al. [21] have used latent space features to predict a temperature to use in TS calibration to obtain better calibration. This work is close to our work, where calibration generalization is sought. The core idea behind the work is that instances with similar representations might require a similar temperature to achieve better-calibrated probabilities. They proposed Cluster-level Nearest Neighbour (*Cluster NN*), which uses clustering on features and then calculates the temperature for each cluster, which will be applied on the target domain based on the assigned cluster. They also proposed Cluster-level Regression (*Cluster LR*), where a linear regression model is trained on cluster centers to predict the target temperature. They use multiple domains to learn calibration to better generalize unseen domains while depending on inferred temperature for probability calibration.

3 Background

3.1 Calibration

Consider a DNN $\phi(.)$, parameterized by θ, which for any given input X predicts a probability for each of the K classes as a class probability vector $\hat{P}$. The class with the highest probability is denoted $\hat{Y}$, and the corresponding probability is known as the confidence, denoted $\hat{Q}$. The classifier is defined to be *perfectly confidence-calibrated* [16] if $\mathbb{P}(\hat{Y} = Y|\hat{Q} = q) = q, \forall q \in [0,1]$ where Y is the ground truth label. The calibration methods aim to adjust the confidence during or after the training to achieve calibrated probabilities. Expected Calibration Error (ECE) [13,16] is used to quantify the calibration capabilities of a classifier by grouping test instances into B bins of equal width based on predicted confidence values where B_m is the set of instances such that $B_m = \{i|\hat{q}_i \in (\frac{m-1}{B}, \frac{m}{B}]\}$ and $\hat{q}_i$ is the predicted confidence for the ith instance. ECE is calculated as the weighted absolute difference between accuracy and confidence across bins as:

$$\mathcal{L}_{\text{ECE}} = \sum_{m=1}^{B} \frac{|B_m|}{N} |\mathbb{A}(B_m) - \mathbb{Q}(B_m)|, \tag{1}$$

where N is the total number of instances; for each bin B_m, the accuracy is $\mathbb{A}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} 1(\hat{y}_i = y_i)$ and the confidence is $\mathbb{Q}(B_m) = \frac{1}{|B_m|}\sum_{i\in B_m} \hat{q}_i$; and finally, $\hat{y}_i$ and y_i are the predicted and actual label for the ith instance.

Temperature Scaling Calibration. TS method for calibration is a simple and popular calibration method [16] which gives calibrated probabilities, once an optimal temperature $T^* > 0$ is calculated by minimizing negative log likelihood (NLL) loss as follows:

$$T^* = \arg\min_{T>0} \sum_{(\mathbf{x}_v, \mathbf{y}_v)\in\mathcal{D}_v} \mathcal{L}_{\text{NLL}}(\sigma(\mathbf{z}_v/T), \mathbf{y}_v), \tag{2}$$

where $\mathcal{D}_v$ is the validation set, $\mathbf{z}_v$ are the logit vectors (network outputs before the softmax) obtained from a classifier trained on $\mathcal{D}_{tr}$ (the training set), $\mathbf{y}_v$ are the ground truth labels, and $\sigma(.)$ is the softmax function. Calibrated probabilities are obtained by applying T^* on test logit vectors $\mathbf{z}_{ts}$ of the test set $\mathcal{D}_{ts}$ as $\hat{\mathbf{y}}_{ts} = \sigma(\mathbf{z}_{ts}/T^*)$.

3.2 Standard Calibration-Refinement Decomposition

Following the notations from the previous Section, let $C = (C_1, \cdots, C_K)$ be the perfectly calibrated probability vector, corresponding to the classifier $\phi(.)$, where $C = \mathbb{E}[Y \mid \hat{P}]$. Consider the expected loss of $\hat{P}$ with respect to Y, *i.e.*, $\mathbb{E}[D(\hat{P}, Y)]$ where D is a proper scoring rule. According to the calibration-refinement decomposition [37,38], the expected loss can be decomposed into the sum of the expected divergence of $\hat{P}$ from C and the expected divergence of C from Y with respect to any proper scoring rule D as follows:

$$\mathbb{E}[D(\hat{P}, Y)] = \mathbb{E}[D(\hat{P}, C)] + \mathbb{E}[D(C, Y)] \tag{3}$$

These two terms are known as the *Calibration Loss (CL)* and the *Refinement Loss (RL)*. CL ($\mathbb{E}[D(\hat{P}, C)]$) is the loss due to the difference between the model estimated probability score $\hat{P}$ and the fraction of positive instances with the same output. Better calibrated models have lower CL. The loss RL ($\mathbb{E}[D(C, Y)]$) is due to the multiple class instances with the same score $\hat{P}$.

As NLL decomposes into the sum of CL and RL, training a DNNs with the objective of NLL means putting equal importance to both parts. This motivates our custom modification to the loss function, which we will describe next.

4 Calibration by Generality Training (CaliGen)

We aim to achieve better generalization of calibration and accuracy with generality-training of the classifier. To achieve the best of both, we propose a new loss function, CaliGen loss, to be used with our approach. The primary objective of classifier training with NLL (*i.e.*, the cross-entropy loss) is to increase the classification accuracy. In contrast, we want the network to produce calibrated

probabilities which is hard to achieve by minimizing NLL [16]. To achieve this goal, we need to use an objective function that penalizes the situation when the model produces uncalibrated probabilities.

4.1 CaliGen Loss Function

NLL loss can be expressed as follows [3]:

$$\mathcal{L}_{\text{NLL}}(\hat{P}, Y) = \mathcal{L}_{\text{KL}}(\hat{P}, Y) + \mathbb{H}[Y] \tag{4}$$

where $\mathcal{L}_{\text{KL}}(.)$ is the KL-divergence loss and $\mathbb{H}[Y]$ is entropy which is a constant with respect to the prediction that is being optimized. Following Eq. (3), we can decompose divergence in Eq. (4) as:

$$\mathcal{L}_{\text{NLL}}(\hat{P}, Y) = \mathcal{L}_{\text{KL}}(\hat{P}, C) + \mathcal{L}_{\text{KL}}(C, Y) + \mathbb{H}[Y], \tag{5}$$

Our objective of generalization is to improve emphasis on CL ($\mathcal{L}_{\text{KL}}(\hat{P}, C)$), that is to obtain better calibration. In other words, if we decrease emphasis on RL ($\mathcal{L}_{\text{KL}}(C, Y)$), it will give more importance to CL. Mathematically, we consider a new loss function as follows:

$$\mathcal{L}(\hat{P}, Y) = \mathcal{L}_{\text{KL}}(\hat{P}, C) + (1 - \rho)(\mathcal{L}_{\text{KL}}(C, Y) + \mathbb{H}[Y]), \tag{6}$$

where $\rho \in [0, 1]$ is a hyperparameter, with higher values of ρ putting less emphasis on refinement and thus more emphasis on calibration. The problem in implementing such a loss function is that we cannot know the perfectly calibrated probabilities C. Thus, instead we approximate these with the probability vector $\hat{C}$ obtained by TS calibration, assuming that $\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) \approx \mathcal{L}_{\text{KL}}(\hat{P}, C)$. This assumption is justified as we first fit TS on the validation set data and then use the same data to generate $\hat{C}$. Given the above approximation, we can now add a negligible term $\rho(\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) - \mathcal{L}_{\text{KL}}(\hat{P}, C))$ to the loss function in Eq. (6) as:

$$\begin{aligned}
\mathcal{L}(\hat{P}, \hat{C}, Y) &= \mathcal{L}(\hat{P}, Y) + \rho(\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) - \mathcal{L}_{\text{KL}}(\hat{P}, C)) \\
&= (1 - \rho)\mathcal{L}_{\text{KL}}(C, Y) + \mathcal{L}_{\text{KL}}(\hat{P}, C) \\
&\quad + \rho(\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) - \mathcal{L}_{\text{KL}}(\hat{P}, C)) + (1 - \rho)\mathbb{H}[Y]) \\
&= (1 - \rho)\mathcal{L}_{\text{KL}}(C, Y) + (1 - \rho)\mathcal{L}_{\text{KL}}(\hat{P}, C) \\
&\quad + \rho\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) + (1 - \rho)\mathbb{H}[Y] \\
&= (1 - \rho)(\mathcal{L}_{\text{KL}}(\hat{P}, C) + \mathcal{L}_{\text{KL}}(C, Y) + \mathbb{H}[Y]) + \rho\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) \\
&= (1 - \rho)\mathcal{L}_{\text{NLL}}(\hat{P}, Y) + \rho\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}),
\end{aligned} \tag{7}$$

For $\rho > 0$, the loss function in Eq. (7) decreases the emphasis on $\mathcal{L}_{\text{NLL}}(\hat{P}, Y))$ and adds emphasis on $\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C})$, which is equivalent to reducing the gap between distribution of predicted probabilities $\hat{P}$ and temperature scaled calibrated probabilities $\hat{C}$. We have $\mathcal{L}(\hat{P}, \hat{C}, Y) \approx \mathcal{L}(\hat{P}, Y)$ (using the assumption $\mathcal{L}_{\text{KL}}(\hat{P}, \hat{C}) \approx \mathcal{L}_{\text{KL}}(\hat{P}, C)$), hence by minimizing $\mathcal{L}(\hat{P}, \hat{C}, Y)$, we are minimizing $\mathcal{L}(\hat{P}, Y)$ with more emphasis on CL ($\mathcal{L}_{\text{KL}}(\hat{P}, C)$). We call this custom loss function $\mathcal{L}(\hat{P}, \hat{C}, Y)$ as the CaliGen loss.

4.2 Generality Training

We learn calibration by generality-training of a trained network by modifying the prediction head. We insert two more layers between the representation and logit layers of the trained network, as shown in Fig. 1. The additional two layers improve the ability to realize more complex functions, as demonstrated by the ablation study in Sect. 6.3. This modified head can be considered a separate Multi-Layer Perceptron (MLP) with two hidden layers where input is the representation produced by the trained model. CaliGen loss requires 3 vectors: representation vector, ground truth label vector, and calibrated probabilities. Generality-training of this modified head is a two-stage task: (i) We consider multiple domains for calibration domain set $\mathcal{C}$, and for each domain $c \in \mathcal{C}$, we obtain T_c^* using TS calibration method given in Eq. (2). We then use it to get the calibrated probability vector $\hat{C}$ for each instance in the calibration domains. These calibrated probabilities for each calibration domain are generated once before the generality-training. We have used TS to obtain the calibrated probabilities for its simplicity. However, our method requires only calibrated probabilities, and thus, in principle, TS in generality-training can be replaced by any other post-hoc calibration method. (ii) During generality-training, all the layers are frozen till the representation layer. We use the CaliGen loss function given in Eq. (7) for optimization based on a fixed value of ρ.

Hyper-parameter Tuning. We consider different values of ρ from [0.0, 0.1, ..., 0.9, 1.0] in the CaliGen loss function Eq. (7) for optimization and use early stopping (with 20% of data allocated for validation). We select the best value of ρ using 3-fold cross-validation based on best error while restricting the selection of ρ from 0.2 to 0.8. The selection range is restricted in $[0.2, 0.8]$ to avoid extreme ρ values that do not improve calibration in our observation (see Fig. 2). The best values of ρ are given in the Supplementary material.

5 Experiments

We experiment with three datasets to test our method. These experiments aim to test our proposed method CaliGen on real datasets and compare the results with SOTA methods in the area. In the following, we give a brief description of each dataset.

5.1 Datasets

Office-Home. [39] The dataset contains around 15,500 images of different sizes from 65 categories. It is further divided into four domains: Art, Clipart, Painting, and Real World. We resize the images to 224×224 and split each domain into 80-20 subsets referred to as the *Large* subset and the *Small* subset, respectively. We use the *Large* subset for training and evaluation, whereas the *Small* subset for calibration. We use one domain for training, 2 for calibration, and 1

remaining domain for evaluation along the same lines as used in [21] and perform experiments on all possible 12 combinations of splitting the 4 domains into 1 training, 1 test, and 2 calibration domains.

DomainNet. [40] The dataset contains different size images from 6 domains across 345 categories. The domains are Clipart, Infograph, Painting, Quickdraw, Real, and Sketch. We resize the images to 224×224 and split each domain into 90–10 referred to as the *Large* subset and the *Small* subset, respectively. We use the *Large* subset for training and evaluation and the *Small* subset for calibration. We use 2 domains for training, 3 for calibration, and 1 for evaluation, similar to [21]. We perform experiments on all possible 60 combinations.

CIFAR-10-C. We used the CIFAR-10 dataset [41] and applied 15 corruptions [42,43] from level 1 (less severe) to level 5 (more severe) on it. We consider 4 corruptions (Gaussian Noise, Brightness, Pixelate, and Gaussian blur) of level 1 as well as original images as the source domains, 4 different corruptions (Fog, Contrast, Elastic Transform, and Saturate) of level 1 as the calibration domains, and the remaining 7 corruptions of all levels as the target domain. A more detailed description of this dataset is given in the supplemental material.

5.2 Experimental Setup

Generality-Training. We use ResNet101 [44] and EfficientNet V2 B0 [45] pre-trained on Imagenet and re-train it on each of the datasets listed in Sect. 5.1. For generality-training, we train a Multi-Layer Perceptron (modified head of DNN) with details given in Fig. 1 and Sect. 4.2.

Base Methods. We use TS on the held-out validation set from the source domain as a reference method. This method does not generalize to unseen domains and is called the *source only* calibration (TS). We consider an *Oracle* method where the model is calibrated with TS on the *Large* subset of the target domain. *Oracle* method is closest to the best possible calibration method within the TS family as it has access to the test data for calibration. For other base methods, we consider TS and Top Label Calibration (Histogram Binning Top Label or HB-TL) [17] by fitting these methods on calibration domains. We have also considered learning the weights by focal loss [3] instead of NLL and then applying calibration methods to improve calibration.

Calibration Adaptation Methods. There is very limited work in Calibration generalization settings; however, the work done by Park et al., Calibrated Prediction with Covariate Shift (CPCS) [19] and by Wang et al., Transferable Calibration (TransCal) [20] address the calibration adaptation by estimating the density ratio of the target domain and source domain with unlabelled instances from the target domain. We used calibration domain instances to estimate the

Table 2. Calibration performance (ECE %) evaluated on target domains of Office-Home dataset and averaged by target domains. The weights are learned by minimizing NLL (default) and focal loss (FL)

Method	Art	Clipart	Product	Real World	Average
Uncalibrated	37.61 ± 5.21	40.32 ± 0.28	29.64 ± 4.81	26.05 ± 7.48	33.41 ± 7.75
TS (Source)	18.12 ± 5.42	24.44 ± 5.38	11.64 ± 5.34	11.25 ± 0.74	16.36 ± 7.15
TS (Oracle)	4.77 ± 0.51	6.06 ± 0.88	6.6 ± 0.9	6.59 ± 1.13	6.0 ± 1.16
HB-TL	25.38 ± 1.73	25.99 ± 4.87	18.57 ± 5.92	14.68 ± 3.54	21.16 ± 6.4
TS	8.24 ± 3.15	15.87 ± 1.4	7.73 ± 3.09	8.77 ± 2.91	10.15 ± 4.3
CPCS	8.98 ± 3.08	15.47 ± 1.62	7.26 ± 1.88	9.32 ± 3.6	10.26 ± 4.1
TransCal	28.76 ± 14.78	23.91 ± 15.78	18.2 ± 10.44	15.5 ± 8.38	21.6 ± 13.71
Cluster NN	8.03 ± 1.79	17.08 ± 2.1	8.2 ± 2.9	7.92 ± 1.65	10.31 ± 4.47
Cluster LR	7.4 ± 0.86	17.79 ± 3.33	8.56 ± 2.81	7.48 ± 1.7	10.31 ± 4.95
Cluster En.	7.77 ± 1.81	17.15 ± 2.02	7.84 ± 2.9	7.56 ± 2.07	10.08 ± 4.66
CaliGen	$\mathbf{6.64 \pm 1.84}$	$\mathbf{14.61 \pm 4.87}$	$\mathbf{6.38 \pm 0.93}$	$\mathbf{7.27 \pm 0.54}$	$\mathbf{8.72 \pm 4.32}$
Uncalibrated (FL)	38.97 ± 3.66	40.73 ± 4.11	29.52 ± 2.17	25.54 ± 5.6	33.69 ± 7.55
TS (FL)	8.73 ± 3.18	14.81 ± 1.61	9.02 ± 1.91	9.17 ± 2.36	10.43 ± 3.45

density ratio for a fair comparison with CaliGen, assuming the target domain is unseen.

Cluster-Level Methods. As we propose the calibration generalization method, we compare our method with the current SOTA method cluster level [21]. We use K-means clustering (for *Cluster NN*) with 8 clusters for the Office-Home dataset and 9 clusters for the other two datasets. The number of clusters for Office-Home and DomainNet was selected from the paper [21] while for CIFAR-10-C, we experimented with clusters from 6 to 15 and chose 9, which gave the best results in the test dataset. We train a linear regressor (for *Cluster LR*) on cluster centers, and for the cluster ensemble method, we take the mean of logits of these two along with TS.

6 Results and Discussion

We perform experiments to test the robustness of CaliGen on different challenging tasks: (a) slight distribution shifts due to corruptions (CIFAR-10-C) and (b) major domain shifts (Office-Home, DomainNet). For this, we experimented with different calibration methods considering calibration domains (which include source domains) to compare the performance of CaliGen with other SOTA methods. Gong et al. [21] considered only calibration domain data for calibration learning, while our experiments suggest that it will lose calibration on source data if source domains are not included in calibration domains (see supplementary material). We run 20 iterations of our experiments with 500 samples from

Table 3. Calibration performance (ECE %) evaluated on target domains of DomainNet dataset and averaged by target domains

Method	Clipart	Infograph	Painting	Quickdraw	Real	Sketch
Uncalibrated	16.12 ± 3.2	28.95 ± 6.9	22.74 ± 5.24	38.33 ± 13.88	18.61 ± 3.71	22.1 ± 3.73
TS (Source)	10.98 ± 2.83	22.31 ± 5.58	15.39 ± 3.99	28.43 ± 13.09	11.26 ± 4.01	14.05 ± 2.05
Oracle (TS)	5.93 ± 0.68	4.72 ± 0.73	4.96 ± 0.78	2.44 ± 0.69	5.08 ± 0.6	5.34 ± 0.37
HB-TL	6.99 ± 1.03	17.3 ± 2.92	10.83 ± 2.29	22.34 ± 3.2	7.66 ± 1.72	10.16 ± 2.02
TS	8.02 ± 2.54	10.35 ± 4.34	6.07 ± 1.48	18.23 ± 11.84	9.06 ± 3.66	6.11 ± 1.25
CPCS	7.09 ± 1.54	13.02 ± 5.35	7.44 ± 3.07	21.5 ± 12.52	8.02 ± 1.33	6.35 ± 1.82
TransCal	$\mathbf{6.94 \pm 1.61}$	19.39 ± 7.9	11.05 ± 3.73	23.7 ± 13.04	7.2 ± 2.25	9.07 ± 3.23
Cluster NN	7.27 ± 1.33	11.67 ± 5.03	6.0 ± 1.76	19.6 ± 8.33	$\mathbf{6.24 \pm 1.13}$	6.0 ± 1.04
Cluster LR	8.43 ± 1.84	13.01 ± 7.69	6.97 ± 2.43	21.86 ± 7.04	7.62 ± 2.16	6.08 ± 0.87
Cluster En.	7.2 ± 1.48	11.55 ± 5.34	$\mathbf{5.66 \pm 1.18}$	20.27 ± 7.61	6.3 ± 1.61	$\mathbf{5.23 \pm 0.57}$
CaliGen	9.63 ± 1.3	$\mathbf{6.91 \pm 1.81}$	5.83 ± 0.62	$\mathbf{12.17 \pm 1.56}$	8.4 ± 0.86	5.81 ± 0.6

Large subset or Test subset and report mean ECE % (calculated with bin size 15) with standard deviation and mean Error % with standard deviation for each dataset.

6.1 Performance Measures

Calibration Performance. Our method CaliGen achieves SOTA results for ECE on each dataset as shown in Table 2 (Office-Home) and Table 3 (DomainNet) where we outperform all other methods on average. For the DomainNet dataset, we achieve the best results on average (8.13 ± 2.57, while the second best method is the cluster ensemble achieving ECE of 9.37 ± 6.6). Note that the high standard deviations in the table are due to considering different combinations of domains for source, calibration, and target. E.g., when Art is the target in Office-Home, then Caligen has ECE of 6.64 ± 1.84, where 6.64 and 1.84 are, respectively, the mean and standard deviation of 5.41,8.75,5.76, which are respectively the results when the source is Clipart, Product, or RealWorld. All methods other than CaliGen struggle when the task is more complex (Clipart in Office-Home, and Quickdraw in DomainNet). In contrast, CaliGen improves the uncalibrated ECE significantly. CPCS is either outperforming or comparable to cluster-based methods. In comparison, CaliGen achieves lower ECE on target domains and also on source domains (see supplementary material). DNN trained with focal loss [3] might not give better calibration on the unseen domain when compared to NLL. We have given detailed results considering the focal loss in the supplementary material.

Improvement Ratio. Improvement Ratio (IR) [21] measures the model's calibration transfer score. Given a source-only calibration and target-only (*Oracle*) calibration, one can measure how close to the oracle-level the model is, compared

Table 4. Improvement Ratio based on average ECE % scores of target domains when the classifier is trained using ResNet (R) or EfficientNet (E)

Method	CIFAR-10-C (R)	Office-Home (R)	Office-Home (E)	DomainNet (R)
CaliGen	**0.34**	**0.74**	**0.88**	**0.73**
Second best	0.26 (CPCS)	0.61 (Cl. En.)	0.78 (CPCS)	0.63 (Cl. En.)

Table 5. Error % averaged by target domains for different datasets while classifier is trained using ResNet101 (R) or EfficientNet V2 B0 (E). All TS based methods do not change error and are the same as Uncalibrated

Method	CIFAR-10-C (R)	Office-Home (R)	Office-Home (E)	DomainNet (R)
Uncalibrated	48.41 ± 14.1	68.95 ± 9.68	61.98 ± 8.8	77.93 ± 11.93
CaliGen	47.22 ± 16.13	63.49 ± 9.64	56.74 ± 10.37	76.33 ± 10.86

to the source-only level. It is measured by $IR = \frac{ECE_S - ECE}{ECE_S - ECE_T}$, where ECE_S is the source only ECE and ECE_T is the *Oracle* ECE. CaliGen achieves the best IR across all datasets (see Table 4). The detailed results on the CIFAR-10-C dataset and EfficientNet network are given in the supplementary material.

Accuracy Generalization. The generality training procedure has access to representations, ground truth labels, and calibrated probabilities of calibration domains, but the representations are learned only on source domains. During generality-training, the model tries to minimize NLL loss on ground truth labels along with minimizing divergence to calibrated probabilities which helps in better accuracy generalization (see Table 5) along with calibration generalization, given the representations learned on source domains.

6.2 Effect of ρ on ECE and Error

In our method, ρ is a hyper-parameter for which we select the best value by 3-fold cross-validation based on minimum error. Figure 2 shows the effect of changing ρ on error and ECE for both Office-Home and DomainNet datasets. We observe the best error rate when ρ is 0.2 or 0.3, and the error increases for higher values, while ECE is not monotonous for higher values of ρ. For $\rho = 0$, the objective function is NLL, and the generality-training does not minimize KL-divergence, so higher ECE and lower error are expected. Still, as we increase ρ, the error further minimizes and monotonically increases after $\rho = 0.3$.

6.3 Ablation Study

We perform an ablation study on generality training considering the effects of (i) not modifying the objective function and (ii) not modifying the head.

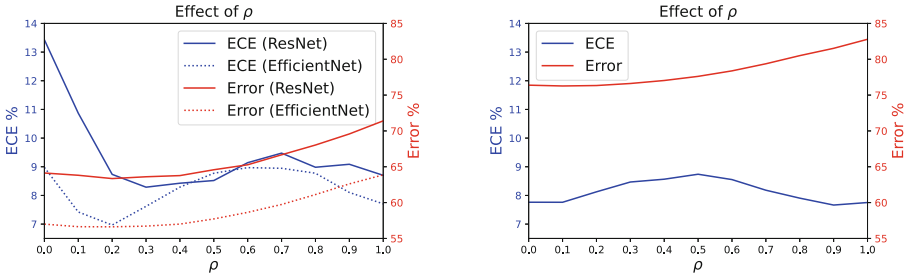


Fig. 2. Effect of ρ on ECE and Error on datasets (a) Office-Home, (b) DomainNet

Unmodified Objective Function. We test the abilities of our proposed objective function given in Eq. (7) by setting $\rho = 0$ (Only the NLL loss). We observe that when only the NLL loss is used, we do not achieve desirable results for ECE as shown in Table 6. This confirms that models trained with NLL give equal importance to CL and RL, thus struggling to produce well-calibrated probabilities.

Table 6. Calibration performance (ECE %) averaged by target domains of Office-Home dataset while fine-tuned with either unmodified loss function or unmodified head

Ablation	Art	Clipart	Product	Real World	Average
Unmodified loss	13.41 ± 4.77	21.23 ± 7.37	10.14 ± 4.83	8.93 ± 0.26	13.43 ± 6.93
Unmodified head	10.57 ± 2.05	19.14 ± 4.67	6.73 ± 0.96	$\mathbf{6.22 \pm 0.32}$	10.66 ± 5.79
CaliGen (no ablation)	$\mathbf{6.64 \pm 1.84}$	$\mathbf{14.61 \pm 4.87}$	$\mathbf{6.38 \pm 0.93}$	7.27 ± 0.54	$\mathbf{8.72 \pm 4.32}$

Unmodified Head. We justify the modification of the network head by testing our CaliGen loss function on the unmodified head. We use the same procedure without modifying the head and select the best ρ by 3-fold cross-validation. As shown in Table 6, modifying the head improves the performance on average. Adding more layers to the head gives it the ability to realize more complex functions while we use dropout (0.5) and L2 regularizer (0.01) to prevent overfitting.

6.4 Limitations

Our method gives better calibration compared to SOTA methods in calibration generalization setting, same as Gong et al. [21], where we set the representations to be fixed. We have made the assumption of fixed representations throughout the whole paper. In contrast, now we investigate what one can do if there are enough time and resources to relearn the representations as well, something that was not considered by Gong et al. [21]. Thus, we use ResNet pre-trained on Imagenet for this experiment and retrained it on source and calibration domains.

Experimental Setup. For training, we use the Office-Home dataset with an additional 50% of the *Small* subsets (10% of the whole domain) from calibration domains along with the source domain (*Large* subset, 80%). This setting redistributes the data for training and calibration such that we have the *Large* subset of source domains and *Small* subset of calibration domains available for training and calibration as discussed in Sect. 5.1. For calibration, the *Small* subset of the source domain (20%) and the remaining 50% of the *Small* subset (10%) of the calibration domains are used.

Table 7. Calibration performance (ECE %) and Error % averaged by target domains of Office-Home dataset while ResNet trained and calibrated on all combinations of 3 domains.

Domain	Trained on source and calib. domains			CaliGen (trained on source domain)		
	ECE (Uncl.)	ECE (TS)	Error	ECE (Uncl.)	ECE	Error
Art	28.67 ± 9.15	9.84 ± 0.42	68.78 ± 3.1	37.61 ± 5.21	6.64 ± 1.84	72.71 ± 2.4
Clipart	40.07 ± 4.13	15.75 ± 1.25	72.34 ± 8.61	40.32 ± 0.28	14.61 ± 4.87	71.94 ± 0.17
Product	19.7 ± 4.03	6.72 ± 1.43	47.78 ± 8.72	29.64 ± 4.81	6.38 ± 0.93	57.97 ± 1.69
RealWorld	21.58 ± 4.6	6.24 ± 0.89	52.55 ± 7.37	26.05 ± 7.48	7.27 ± 0.54	51.34 ± 5.36
Average	27.51 ± 9.92	9.64 ± 3.94	60.36 ± 12.72	33.41 ± 7.75	8.72 ± 4.32	63.49 ± 9.64

Results. Results shown in Table 7 are averaged by target domains. CaliGen obtains surprisingly better ECE than the model trained on calibration data. However, this training procedure aims to give richer representation to source and calibration domains, which helps in better accuracy generalization on unseen domains. However, CaliGen still gives slightly better errors in the Clipart and Real World domains. In a scenario where calibration domains are unavailable during training, it is more sensible to save cost by generality-training with CaliGen instead of retraining the whole network.

7 Conclusion

In this paper, we addressed the problem of calibration generalization for unseen domains. Based on the goal of improving calibration, we derived a new loss function that gives more weightage to Calibration Loss. We proposed a novel generality-training procedure CaliGen which modifies the head of the DNN and uses our proposed CaliGen loss function, which gives more weightage to Calibration Loss. Together, these two changes improve domain generalization, meaning that accuracy and calibration improve on unseen domains while staying comparable to standard training on seen domains. Similarly to several earlier works, the method assumes that all the layers until the representation layer are fixed and cannot be retrained due to limited resources. We also show that if retraining is possible, then results can be improved further.

References

1. Krishnan, R., Tickoo, O.: Improving model calibration with accuracy versus uncertainty optimization. In: Advances in Neural Information Processing Systems, vol. 33, pp. 18237–18248 (2020)
2. Kumar, A., Sarawagi, S., Jain, U.: Trainable calibration measures for neural networks from kernel mean embeddings. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 2805–2814. PMLR (2018). https://proceedings.mlr.press/v80/kumar18a.html
3. Mukhoti, J., et al.: Calibrating deep neural networks using focal loss. In: Larochelle, H., et al. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 15288–15299. Curran Associates Inc. (2020). https://proceedings.neurips.cc/paper/2020/file/aeb7b30ef1d024a76f21a1d40e30c302-Paper.pdf
4. Lin, T.-Y., et al.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
5. Cheng, J., Vasconcelos, N.: Calibrating deep neural networks by pairwise constraints. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13709–13718 (2022)
6. Platt, J., et al.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. Large Margin Classif. **10**(3), 61–74 (1999)
7. Zadrozny, B., Elkan, C.: Transforming classifier scores into accurate multiclass probability estimates. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 694–699 (2002)
8. Kull, M., et al.: Beyond temperature scaling: obtaining well-calibrated multi-class probabilities with dirichlet calibration. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
9. Zadrozny, B., Elkan, C.: Obtaining calibrated probability estimates from decision trees and Naive Bayesian classifiers. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 609–616 (2001)
10. Naeini, M.P., Cooper, G.F.: Binary classifier calibration using an ensemble of near isotonic regression models. In: 2016 IEEE 16th International Conference on Data Mining (ICDM), pp. 360–369. IEEE (2016)
11. Allikivi, M.-L., Kull, M.: Non-parametric Bayesian isotonic calibration: fighting over-confidence in binary classification. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (eds.) ECML PKDD 2019. LNCS (LNAI), vol. 11907, pp. 103–120. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-46147-8_7
12. Kull, M., Filho, T.S., Flach, P.: Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In: Singh, A., Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 54, pp. 623–631. PMLR (2017). https://proceedings.mlr.press/v54/kull17a.html
13. Naeini, M.P., Cooper, G., Hauskrecht, M.: Obtaining well calibrated probabilities using Bayesian binning. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)
14. Wenger, J., Kjellström, H., Triebel, R.: Non-parametric calibration for classification. In: International Conference on Artificial Intelligence and Statistics, pp. 178–190. PMLR (2020)

15. Gupta, K., et al.: Calibration of neural networks using splines. In: International Conference on Learning Representations (ICLR) (2021). https://openreview.net/forum?id=eQe8DEWNN2W
16. Guo, C., et al.: On calibration of modern neural networks. In: International Conference on Machine Learning, pp. 1321–1330. PMLR (2017)
17. Gupta, C., Ramdas, A.: Top-label calibration and multiclass-to-binary reductions. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=WqoBaaPHS-
18. Ovadia, Y., et al.: Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
19. Park, S., et al.: Calibrated prediction with covariate shift via unsupervised domain adaptation. In: Chiappa, S., Calandra, R. (eds.) Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 108, pp. 3219–3229. PMLR (2020). https://proceedings.mlr.press/v108/park20b.html
20. Wang, X., et al.: Transferable calibration with lower bias and variance in domain adaptation. In: Larochelle, H., et al. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 19212–19223. Curran Associates Inc. (2020). https://proceedings.neurips.cc/paper/2020/file/df12ecd077efc8c23881028604dbb8cc-Paper.pdf
21. Gong, Y., et al.: Confidence calibration for domain generalization under covariate shift. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8958–8967 (2021)
22. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. J. Big Data **3**(1), 1–40 (2016)
23. Pampari, A., Ermon, S.: Unsupervised calibration under covariate shift. CoRR, abs/2006.16405 (2020). https://arxiv.org/abs/2006.16405
24. Wald, Y., et al.: On calibration and out-of-domain generalization. In: Advances in Neural Information Processing Systems, vol. 34 (2021)
25. Pleiss, G., et al.: On fairness and calibration. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
26. Kong, L., et al.: Calibrated language model fine-tuning for in- and out-of-distribution data. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1326–1340. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-main.102. https://aclanthology.org/2020.emnlp-main.102
27. Tomani, C., et al.: Post-hoc uncertainty calibration for domain drift scenarios. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10124–10132 (2021)
28. Wang, J., et al.: Generalizing to unseen domains: a survey on domain generalization. IEEE Trans. Knowl. Data Eng. (2022)
29. Dusenberry, M., et al.: Efficient and scalable Bayesian neural nets with rank-1 factors. In: International Conference on Machine Learning, pp. 2782–2792. PMLR (2020)
30. Thulasidasan, S., et al.: On mixup training: improved calibration and predictive uncertainty for deep neural networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
31. Müller, R., Kornblith, S., Hinton, G.E.: When does label smoothing help? In: Wallach, H., et al. (eds.) Advances in Neural Information Processing Systems, vol.

32. Curran Associates Inc. (2019). https://proceedings.neurips.cc/paper/2019/file/f1748d6b0fd9d439f71450117eba2725-Paper.pdf
32. Lienen, J., Hüllermeier, E.: From label smoothing to label relaxation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 8583–8591 (2021)
33. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
34. Wen, Y., Tran, D., Ba, J.: Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In: International Conference on Learning Representations (2020). https://openreview.net/forum?id=Sklf1yrYDr
35. Hébert-Johnson, U., et al.: Multicalibration: calibration for the (computationally-identifiable) masses. In: International Conference on Machine Learning, pp. 1939–1948. PMLR (2018)
36. Shabat, E., Cohen, L., Mansour, Y.: Sample complexity of uniform convergence for multicalibration. In: Advances in Neural Information Processing Systems, vol. 33, pp. 13331–13340 (2020)
37. Bröcker, J.: Reliability, sufficiency, and the decomposition of proper scores. Q. J. Roy. Meteorol. Soc. **135**(643), 1512–1519 (2009)
38. Kull, M., Flach, P.: Novel decompositions of proper scoring rules for classification: score adjustment as precursor to calibration. In: Appice, A., Rodrigues, P.P., Santos Costa, V., Soares, C., Gama, J., Jorge, A. (eds.) ECML PKDD 2015. LNCS (LNAI), vol. 9284, pp. 68–85. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23528-8_5
39. Venkateswara, H., et al.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5018–5027 (2017)
40. Peng, X., et al.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1406–1415 (2019)
41. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-10 (Canadian institute for advanced research) (2009). https://www.cs.toronto.edu/~kriz/cifar.html
42. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations (2019). https://openreview.net/forum?id=HJz6tiCqYm
43. TensorFlow Datasets Team. imagenet2012_corrupted (2020). https://github.com/tensorflow/datasets/blob/master/tensorflow_datasets/image_classification/imagenet2012_corrupted.py
44. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
45. Tan, M., Le, Q.: Efficientnetv2: smaller models and faster training. In: International Conference on Machine Learning, pp. 10096–10106. PMLR (2021)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

