Multi-Domain Meta Calibration Using Differentiable Expected Calibration Error

Niv Kook **Rotem Ben Zion**

{niv.ko, rotm}@campus.technion.ac.il

Abstract

Confidence calibration – the problem of predicting probability estimates representative of the true correctness likelihood – is important for classification models in many applications. An existing paper called "Meta-Calibration: Learning of Model Calibration Using Differentiable Expected Calibration Error" uses meta-learning techniques to train neural networks to be well-calibrated. In this project, we extend that work to the setting of multiple domains; We define and implement a label-smoothing meta learning technique for multiple domains, and test its calibration generalization quality in over-parameterized neural networks.

1 Introduction

When deploying neural networks to real-world applications, it is crucial that models' own confidence estimates accurately match their probability of making a correct prediction. If a model is over-confident about its predictions, we cannot rely on it; while well-calibrated models can abstain or ask for human feedback in the case of uncertain predictions. In addition, some tasks directly rely on calibration such as outlier detection (Hendrycks and Gimpel, 2016). Unfortunately, over-parameterized neural networks tend to be badly calibrated on new data (Guo et al., 2017). This challenge of calibrating neural networks has motivated a growing area of research, some of the existing methods will be discussed in section 2. One of the promising techniques alters the loss function by introducing label smoothing (Müller et al., 2019), though it is not clear how to choose the smoothing hyper-parameters. This motivated the authors of "Meta-Calibration: Learning of Model Calibration Using Differentiable Expected Calibration Error" (Bohdal et al., 2021) to develop a meta-learning technique which allows training a small amount of meta parameters for label smoothing. The training loss is a differetiable approximation of the ECE metric, which is also developed in (Bohdal et al., 2021).

The difference between train and test calibrations can be viewd in terms of distribution shift and out-of-domain generalization. In this project, we test the meta-learning method of (Bohdal et al., 2021) in a multi-domain setting. We compare it to the baseline (no label smoothing) and to a new version, with seperate meta-parameters for each domain. Our goal is to test whether meta-learning the label-smoothing parameters in different domains improves calibration as measured by Average-ECE.

2 Related work

Calibration

Model calibration is a popular area of research in recent years. (Guo et al., 2017) study a variety of potential solutions and find that simple post-training rescaling of the logits – temperature scaling – works relatively well. Other methods, including the ones we implement in this project, apply during model training. This includes the MMCE kernel-base calibration measure provided in (Kumar et al., 2018) which they use as regularization, the Focal loss (Mukhoti et al., 2020) which is a relatively simple weighted alternative to cross-entropy, and the classic Brier score (Brier et al., 1950), which is the squared error between the softmax vector with probabilities and the ground-truth one-hot encoding.

A more relevant work to our project is (Müller et al., 2019), in which label smoothing has been shown to improve model calibration. However, this calibration metric was not optimized as most aforementioned calibration metrics are non-differentiable. The work of (Bohdal et al., 2021) does optimize the calibration metric as will be discussed in section 3.

Out-of-domain generalization

The goal of domain generalization algorithms is to predict well on distributions different from those seen during training. This is a major area of study in modern machine learning, and popular strategies include learning invariant features, sharing parameters, data augmentation and most relevantly - meta learning. For an exhaustive literature review of all these approaches, we refer to (Gulrajani and Lopez-Paz, 2020). In the meta-learning category, one suggested algorithm is MAML (Model-Agnostic Meta-Learning) (Finn et al., 2017), which attempts to build a predictor that learns how to adapt quickly between training domains. (Li et al., 2019) extend the MAML meta-learning strategy to instances of domain generalization where the categories vary from domain to domain.

Multi-domain calibration

The article that motivated this project is (Wald et al., 2021), which suggests that multi-domain calibration leads to better out-of-domain generalization. This raises the need to find training methods which create well-calibrated models over multiple domains.

3 Training objective for meta calibration

An intuitive but non-differentiable calibration metric is the expected calibration error (ECE), which measures the expected difference (in absolute value) between the accuracies and the confidences of the model on examples that belong to different confidence intervals. ECE is defined as

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|$$

where the accuracy and confidence for bin B_m are

$$\text{acc}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}\{\hat{y}_i = y_i\}$$

$$\text{conf}(B_m) = \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i$$

This measure is obviously not differentiable because accuracy is not differentiable, as well as assigning examples into bins. (Bohdal et al., 2021) developed a differentiable metric which approximates both the accuracy and the binning, called differentiable ECE (DECE):

$$\text{DECE} = \sum_{m=1}^{M} \frac{\sum_{i=1}^{n} o_m(x_i)}{n} |\text{acc}(B_m) - \text{conf}(B_m)|$$

$$o_m(x_i) = p(B_m|\hat{p}_i)$$

$$\text{acc}(B_m) = \frac{1}{\sum_{i=1}^{n} o_m(x_i)} \sum_{i=1}^{n} o_m(x_i) \mathbf{1}\{\hat{y}_i = y_i\}$$

$$\text{conf}(B_m) = \frac{1}{\sum_{i=1}^{n} o_m(x_i)} \sum_{i=1}^{n} o_m(x_i) \hat{p}_i$$

4 Meta calibration method: label smoothing

As mentioned in the introduction, the learnable meta-parameters in this project are label smoothing parameters (LS). In this section we provide the definition of label smoothing and distinguish the learnable set of parameters in each of our experimental models.

4.1 label-smoothing definition

Learnable label smoothing learns coefficients to uniformly smooth the one-hot encoded labels. More formally, if there are K classes and D domains in total, for input in domain d, $y_k = 1$ for the correct class $c = k$ and $y_k = 0$ for all classes $k \neq c$, then with learnable label smoothing ω the soft label for class k becomes

$$y_k^{\text{LS}} = y_k(1 - \omega_{c,d}) + \omega_{c,d}/K$$

Note that the resulting soft-labels are differentiable with respect to ω. In the training procedure we will obtain DECE on the validation

set with the classifier after it was updated using these soft labels, meaning the DECE value is also differentiable with respect to ω, and we can use normal optimization methods to optimize for ω.

4.2 learnable parameter sets

In the scope of this project, we implement and test three types of models, varying on the set of learnable meta-parameters.

- **no_meta**: no meta parameters; the loss used is regular cross entropy, equivalent to constant smoothing params $\omega_{c,d} \equiv 0$.
- **one_vec_meta**: a scalar meta-parameter for each class, disregarding the domain. Equivalent to the constraint $\omega_{c,d_1} = \omega_{c,d_2} \forall d_1, d_2$.
- **multi_domain_meta**: a scalar meta-parameter for each combination of class and domain, so $K \cdot D$ parameters in total. This uses the formulation from section 4.1 without additional constraints.

4.3 training procedure

The training procedure pseudocode is provided in algorithm 1. In each training iteration, we use two batches of labeled data, train and meta-validation. We pass the train batch through our network to receive logits, and use our smoothing parameters to receive soft-labels of the train batch. We perform soft-cross-entropy to receive a loss. If the smoothing params aren't learnable ("no_meta" model type, see section 4.2) we simply update our network and move on to the next pair of batches. Otherwise (smooth params are learnable), we use the loss to update only the classifier part of our network (for computational efficiency), and then pass the meta-validation batch through the partially-updated network. We receive validation logits which we use to calculate DECE with the meta-validation labels (see section 3). With the DECE loss we update our meta-parameters. We recalculate the soft-cross-entropy loss and use it to update the entire network (overriding the previous change to the classifier).

A few important notes regarding the training procedure:

- We only calculate the logits (i.e., pass the input through the model) once for each data batch. In addition, the gradient calculation for the meta-step is only calculated over the classifier, which is a simple linear layer. This means no major computational complexity is added.
- The first loss calculation uses the label-smoothing parameters from the previous iteration, and the last one uses the new parameters. This strictly improves DECE over the meta-validation batch since gradient descent is a descent method.
- the classifier is not updated twice; the first update is temporary and used only to calculate DECE to optimize ω.

5 Experimental setup

All code, including the different implementations, data preprocessing, evaluation measures and training utilities is available on GitHub. For visualizations we used Weights & Biases (Biewald, 2020). To reproduce the results, run `meta_calib.py`. In this section we list the datasets and architectures that we have used.

5.1 Datasets

We have chosen three multi-domain settings to test the training techniques on.

1. **ColoredMNIST** (Arjovsky et al., 2019): This is a synthetic binary classification task derived from MNIST. We color each image in a way that correlates strongly (but spuriously) with the class label. We chose to use three domains. More information can be found in the original paper.
2. **RotatedMNIST** (Ghifary et al., 2015): This is a synthetic multiclass classification task derived from MIST. Each domain is defined by a degree of rotation. The samples in each domain are taken randomly (without correlation to the label). We decided to use six domains.

Algorithm 1 TrainProcedure

θ: Encoder, φ: Classifier, ω: LS
for $X_{\text{train}}, Y_{\text{train}}, X_{\text{mval}}, Y_{\text{mval}}$ **do**
  train_logits $= \varphi(\theta(X_{\text{train}}))$
  train_soft_labels $= LS(Y_{\text{train}}; \omega)$
  $\mathcal{L}_{CE} = cross\_entropy(\text{train\_logits}, \text{train\_soft\_labels})$
  if ω is not learnable **then**
    Update $\theta \leftarrow \theta - \alpha_1 \nabla_\theta \mathcal{L}_{CE}, \quad \varphi \leftarrow \varphi - \alpha_2 \nabla_\varphi \mathcal{L}_{CE}$
  else
    $\hat{\varphi} \leftarrow \varphi - \alpha_2 \nabla_\varphi \mathcal{L}_{CE}$
    val_logits $= \hat{\varphi}(\theta(X_{\text{mval}}))$
    $\mathcal{L}_{DECE} = DECE(\text{val\_logits}, Y_{\text{mval}})$
    $\omega \leftarrow \omega - \alpha_1 \nabla_\omega \mathcal{L}_{DECE}$
    train_soft_labels$^{new} = LS(Y_{\text{train}}; \omega)$
    $\mathcal{L}_{CE}^{new} = cross\_entropy(\text{train\_logits}, \text{train\_soft\_labels}^{new})$
    Update $\theta \leftarrow \theta - \alpha_1 \nabla_\theta \mathcal{L}_{CE}^{new}, \quad \varphi \leftarrow \varphi - \alpha_2 \nabla_\varphi \mathcal{L}_{CE}^{new}$
  end if
end for

3. **CorruptedCIFAR10** (Hendrycks and Dietterich, 2019): This dataset contains a variety of image corruptions, such as adding fog, pixelation, changes to the brightness and many others (overall 19 corruption types). We decided to use 22 domains, as in (Bohdal et al., 2021) and (Zhang et al., 2021).

5.2 Architectures

For the CorruptedCIFAR10 setting, which was used in (Bohdal et al., 2021), we used the same network architecture, resnet18. architecture.
For ColoredMNIST and RotatedMNIST, we used an MNIST-finetuned architecture suggested in DomainBed (Gulrajani and Lopez-Paz, 2020). architecture.

6 Results

6.1 RotatedMNIST

In figure 1 we plot the average accuracy over the validation set (averaged over epochs). Unfortunately, the accuracy of the calibration-trained models is significantly lower, at 85 % compared to 98 % by the regular ERM. In addition, the new version adjusted to the multi-domain setting which includes additional trainable parameters performs even worse than the classic one-vector version presented in (Bohdal et al., 2021).
To compare the meta-calibration techniques, we plot the meta loss (DECE) value of each batch during training in figure 2. The result is rather interesting - the meta loss is generally increasing throughout the training when multi-domain LS is used. As we will discuss later in section 7, we believe this can be explained by the justified high confidence characteristic to the MNIST dataset.
Figure 3 shows the average ECE over the validation set for the different models. Note that ECE is an error measure, so good calibration leads to smaller values. To our surprise, this result presents far better calibration when using the regular ERM in comparison to the meta-calibrated models! This result is consistent with our results from the other MNIST dataset, and the relationship between the meta-calibrated methods is consistent with figure 2. We came up with several possible explanations to this phenomenon, which we list in section 7.

6.2 ColoredMNIST

Figure 4 shows the accuracy of the different models, measured periodically during training over the validation set. The effect of over-

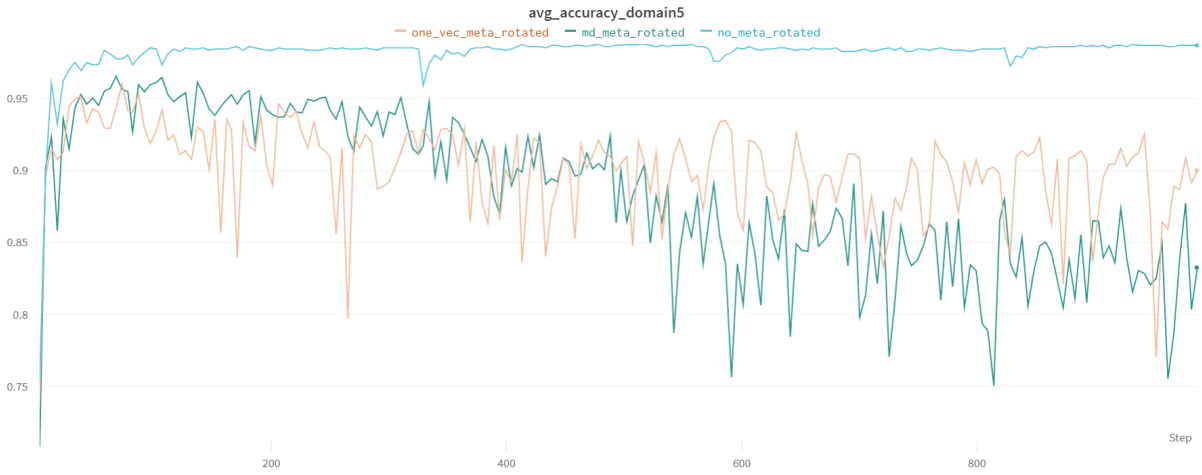

Figure 1: The accuracy of each model over the RotatedMNIST validation set, function of training step.

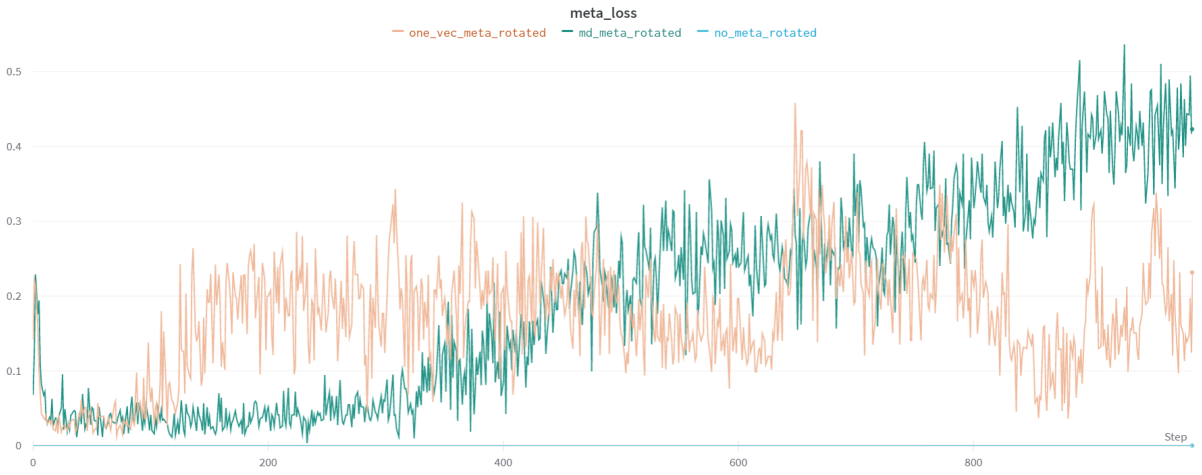

Figure 2: The meta loss over the meta-validation set in RotatedMNIST, comparing the meta-calibration techniques. Since the regular ERM doesn't use meta-training, we set its value to zero.

fitting caused by the artificially synthesized confounders is noticable. It seems that ERM and the classic one-vector variant of the meta-calibrated model achieve similar results of $\approx 74\%$, while the new multi-domain variant has consistently lower accuracy by 0.1%.
As for calibration, in figure 5 we view the ECE metric of each of the models, again as a function of training step, and it shows the same trend: ERM and single-vector meta-calibration achieve similar results and the multi-domain variant lacks behind.

6.3 CorruptedCIFAR10

We examined our new calibration method and the old "one vector method", in all combinations of possible data setups in which the train and meta-validation sets are either uni-domain (UD) or multi-domain (MD). The domains used for the MD test results are disjoint from the ones used during training.
The pure results can be seen in tables 1, 2 and 3. It seems that the new method outperforms the original single-vector method in the UD ece and in the avg MD ece. On the other hand, it is worse on the worst domain in each metric, and fails when we use an MD meta set.

7 Discussion and future work

7.1 MNIST

The results we have received in the MNIST portion of this project were not as we expected. (Bohdal et al., 2021) showed tremendous gains to calibration from the meta-LS technique, which were not reproduced in our experiments.

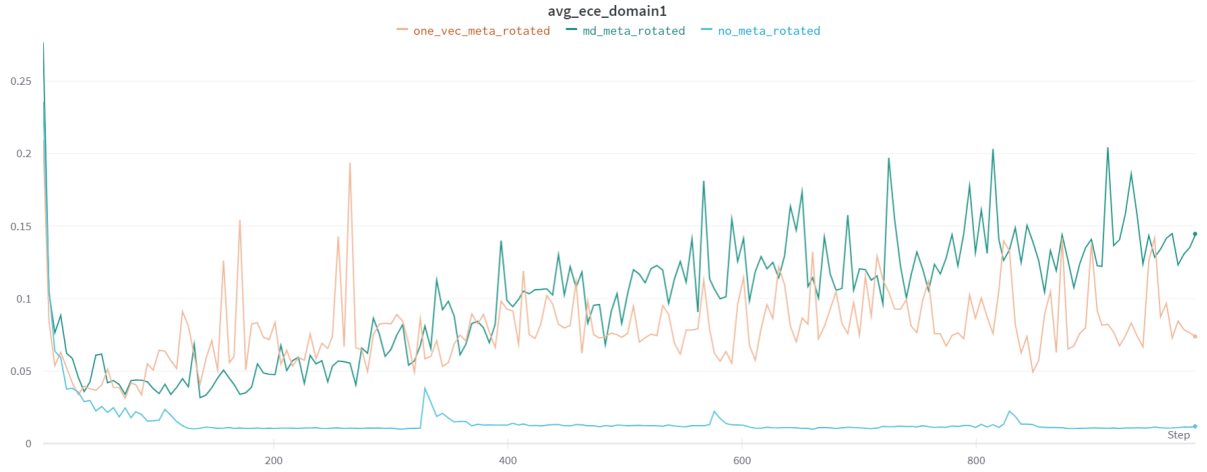

Figure 3: The ECE calibration metric over the validation set in RotatedMNIST as a function of training step.

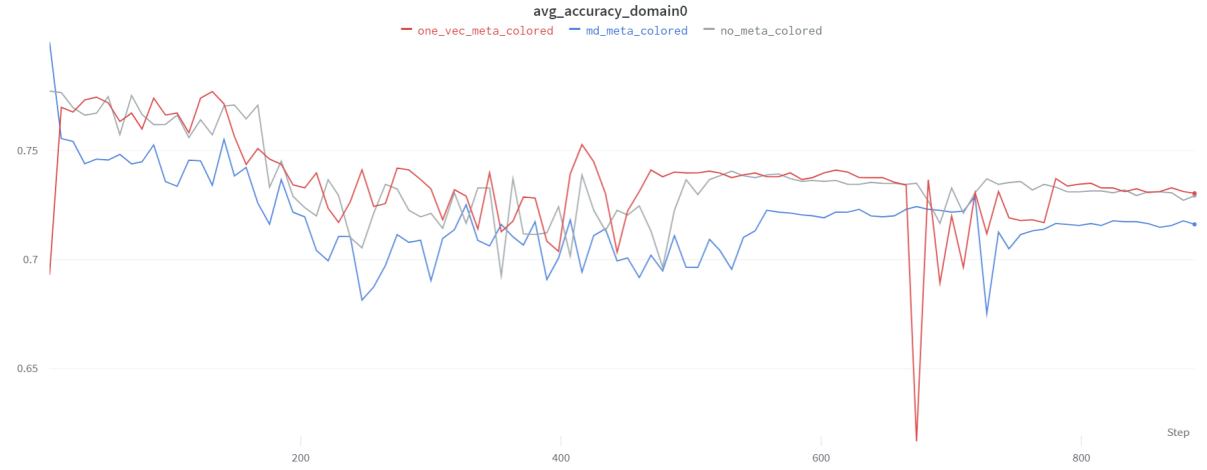

Figure 4: The accuracy of each model over the ColoredMNIST validation set, function of training step.

Note several major differences between our experimental setup and theirs:

- Different dataset: for multi-domain training we leveraged the ColoredMNIST and RotatedMNIST datasets which are fundamentally different from the CIFAR dataset used in (Bohdal et al., 2021) and in the other part of our project.

- Different architecture: while both are convolutional neural networks, we used the architecture advised by DomainBed and (Bohdal et al., 2021) used ResNet-18.

- Different training specifics: Learning rate, number of epochs and optimization decisions are naturally different between the implementations.

These differences raise possible explenations for the gap in results. Firstly, optimization hyper-parameters may be crucial and architecture-dependent for the meta-calibration task, especially since the number of parameters is small. Furthermore, we note that the MNIST dataset is generally an easy classification task, for which many models acheive extremely high levels of accuracy. Hence, high confidence is not necessarily unjustified, and as we have seen, leads to low calibration error (see figures 3, 5). This is plausibly the primary cause to the under-performance of the meta-calibration-trained models in our experiments on MNIST variants.

The new variant for meta-optimization we used in this project, the multi-domain

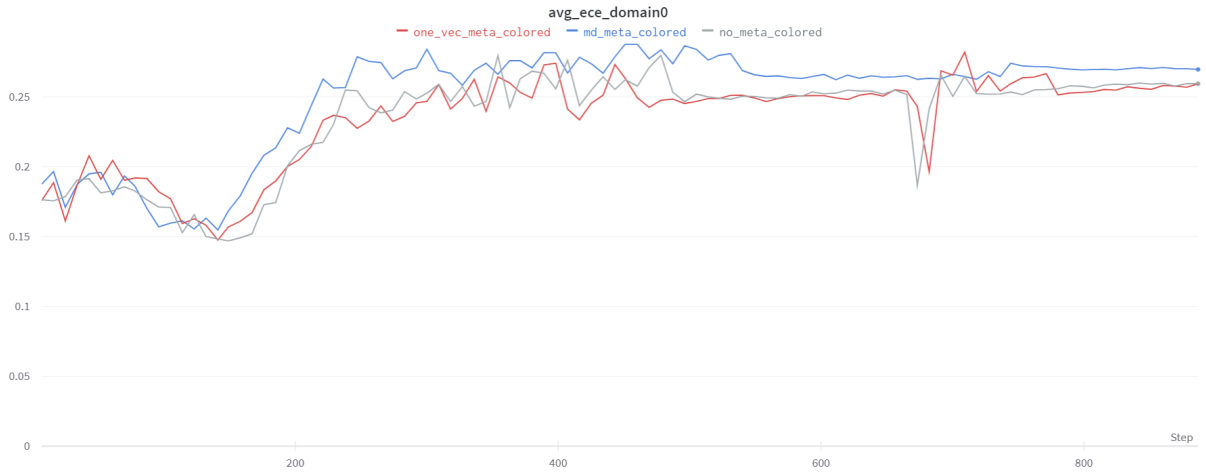

Figure 5: The ECE calibration metric over the validation set in ColoredMNIST as a function of training step.

	UD err	UD ECE	MD max err	MD avg err	MD max ECE	MD avg ECE
no_meta	0.134	9.09	0.282	0.175	19.9	12.1
one_vec_meta	0.135	5.4	0.308	0.183	11.6	5.3
md_meta	0.151	3.22	0.342	0201	12.1	4.2

Table 1: Results over the CorruptedCIFAR10 dataset, where the train set is multi-domain and the meta-validation set is uni-domain.

label smoothing technique, didn't show any improvements over the other methods in the MNIST section. Perhaps this is the same issue as the underperformance of the single-vector meta-calibration; another option is that splitting the parameters to different domains doesn't yield benefit but rather adds unnecessary noise to the training process.

7.2 CIFAR

Unlike the previous section, noticable gains in calibration can be seen in our experiments over the CorruptedCIFAR10 dataset when using the meta calibration methods. Additionally, the new multi-domain adjusted version of meta-calibration yields the best ECE when fitted with a multi-domain training set and single-domain meta-validation set. Thus, it seems that the new method has great potential and can be useful when trying to get calibrated models. Further research is required to determine whether our method has any significant advantage over the existing ones.

7.3 Future work

Firstly, an obvious continuation is experimentation of the same models using other multi-domain datasets and tasks, as the calibration techniques may only be effective for difficult tasks. Other paths forward include different meta-parameters instead/in addition to label smoothing, such as regularization or hyper-parameters of loss functions.

	UD err	UD ECE	MD max err	MD avg err	MD max ECE	MD avg ECE
no_meta	0.134	9.09	0.282	0.175	19.9	12.1
one_vec_meta	0.137	5.64	0.29	0.181	6.3	5.4
md_meta	0.16	8.87	0.366	0.211	15	8.7

Table 2: Results over the CorruptedCIFAR10 dataset, where the train set is multi-domain and the meta-validation set is multi-domain. Note that the no_meta method is equivalent to tabel 1

	UD err	UD ECE	MD max err	MD avg err	MD max ECE	MD avg ECE
no_meta	0.49	4.17	0.786	0.243	74	21.2
MD_meta	0.63	10.73	0.574	0.212	26.9	8.7
UD_meta	0.69	1.22	0.615	0.256	35.5	13.1

Table 3: Results over the CorruptedCIFAR10 dataset, where the train set is uni-domain. Therefore the md_meta is equivalent to the one_vec_meta.

8 Conclusion

Out-of-domain calibration is necessary in real-world applications to provide predictions with reliable confidence output. In this project, we tested a meta-training technique which trains models to be well-calibrated, in a new setting of multiple training/test domains. We implemented an adjusted meta-training version for the multi-domain case and performed experiments over three datasets. The CorruptedCIFAR10 dataset provided promising results, showing improved calibration in some cases over the existing method. However, datasets derived from MNIST led to poor results with both the new technique and the original one. We believe additional research may leverage the multi-domain-adjusted technique to train well-calibrated models efficiently.

References

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. 2021. Meta-calibration: Meta-learning of model calibration using differentiable expected calibration error. *arXiv preprint arXiv:2106.09613*.

Glenn W Brier et al. 1950. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. 2015. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pages 2551–2559.

Ishaan Gulrajani and David Lopez-Paz. 2020. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Dan Hendrycks and Thomas Dietterich. 2019. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*.

Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. 2018. Trainable calibration measures for neural networks from kernel mean embeddings. In *International Conference on Machine Learning*, pages 2805–2814. PMLR.

Yiying Li, Yongxin Yang, Wei Zhou, and Timothy Hospedales. 2019. Feature-critic networks for heterogeneous domain generalization. In *International Conference on Machine Learning*, pages 3915–3924. PMLR.

Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. 2020.

Calibrating deep neural networks using focal loss. *Advances in Neural Information Processing Systems*, 33:15288–15299.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. 2021. On calibration and out-of-domain generalization. *Advances in neural information processing systems*, 34:2215–2227.

Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. 2021. Adaptive risk minimization: Learning to adapt to domain shift. *Advances in Neural Information Processing Systems*, 34:23664–23678.