AdaFocal: Calibration-aware Adaptive Focal Loss

Anonymous Author(s)
Affiliation
Address
email

Abstract

Much recent work has been devoted to the problem of ensuring that a neural network's confidence scores match the true probability of being correct, i.e. the calibration problem. Of note, it was found that training with focal loss leads to better calibrated neural networks than cross-entropy, while achieving the same level of accuracy [19]. This success stems from focal loss regularizing the entropy of the model's prediction (controlled by the parameter γ), thereby reining in the model's overconfidence. Further improvement is expected if γ is selected independently for each training sample; Sample-Dependent Focal Loss (FLSD-53) in [19] does just that, but is based on heuristics that do not generalize well to all dataset-model pairs. In this paper, we propose a calibration-aware adaptive focal loss called AdaFocal that utilizes the calibration properties of focal (and inverse-focal loss) and adaptively modifies γ_t for different groups of samples based on (1) γ_{t-1} from the previous step (2) the magnitude of the model's under/over-confidence. We evaluate AdaFocal on various image recognition tasks and one NLP task, covering a variety of network architectures, to confirm the improvement in calibration while achieving similar levels of accuracy. Additionally, models trained with AdaFocal are shown to exhibit a significant boost in out-of-distribution detection capability.

1 Introduction

Neural networks have found tremendous success in almost every field including computer vision, natural language processing, and speech recognition. Over time, these networks have grown complex and larger in size to achieve state-of-the-art performance and they continue to evolve in that direction. However, it has been well established that such high capacity networks suffer from poor calibration [4], i.e. the confidence scores of the predictions do not reflect the real world probabilities of those predictions being true. For example, if the network assigns 0.8 confidence to a set of predictions, we should expect 80% of those predictions to be correct. However, this is far from reality since modern networks tend to be grossly over-confident. This is of great concern, particularly for mission-critical applications such as autonomous driving or medical diagnosis, wherein the downstream decision making relies not only on the predictions but also on their confidences.

In recent years, there has been a growing interest in developing methods for calibrating neural networks. These roughly divide into two categories (1) post-hoc approaches that perform calibration after training (2) methods that calibrate the model during training itself. The first category includes methods such as temperature scaling [4], histogram binning [31], isotonic regression [32], Bayesian binning and averaging [22, 21], Dirichlet scaling [10], mix-n-match methods [34], and spline fitting [5]. Methods in the second category focus on designing objective functions that account for calibration during training itself, such as Maximum Mean Calibration Error (MMCE) [13], Label smoothing [20], and recently focal loss [19]. These aim to inherently calibrate the model during training, yet when combined with post hoc calibration further improvement is often obtained.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Contribution. Our work belongs to the second category. We first show that while regular focal loss (with fixed γ) improves the overall calibration by preventing samples from being over-confident, it also leaves other samples under-confident. To address this drawback, we propose a modification to focal (and inverse-focal) loss [30, 17] called AdaFocal that adjusts the γ for each training sample (or rather a group of samples) separately by taking into account the model's under/over-confidence about a corresponding sample (or group) in the validation set. AdaFocal also adaptively switches between focal and inverse focal loss when focal loss fails to reach the desired level of confidence. We evaluate our method on four image classification tasks (CIFAR-10, CIFAR-100, Tiny-ImageNet and ImageNet) and one text classification task (20 Newsgroup) using various model architectures. We find that AdaFocal substantially outperforms regular focal loss and other state-of-the-art calibration-during-training techniques in the literature. We further study the performance of AdaFocal on an out-of-distribution detection task and find a substantial improvement in performance. Finally, we see that models calibrated by AdaFocal may further benefit from post hoc calibration techniques.

2 Problem Setup and Definitions

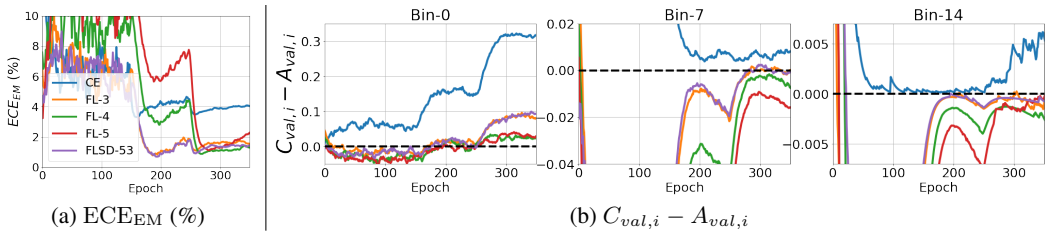
For a classification problem with training data $\{(\mathbf{x}_n, y_{\text{true},n})\}$, where $\mathbf{x}_n$ is the input and $y_{\text{true},n} \in \mathcal{Y} = \{1, 2, \ldots, K\}$ is the ground-truth, we wish to train a model f that outputs a probability vector $\hat{\mathbf{p}}$ over the K classes. We further assume access to a validation set for hyper-parameter tuning and a test set for evaluation. For example, $f_\theta(\cdot)$ can be a neural network with learnable parameters θ, $\mathbf{x}$ is an image, and $\hat{\mathbf{p}}$ is the output of a *softmax layer* whose k^{th} element $\hat{p}_k$ is the probability score for class k. We refer to $\hat{y} = \arg\max_{k \in \mathcal{Y}} \hat{p}_k$ as the network's prediction and the probability score $\hat{p}_{\hat{y}}$ as the confidence. Then, a model is said to be perfectly calibrated if the confidence score $\hat{p}_{\hat{y}}$ matches the true probability of the model classifying $\mathbf{x}$ correctly i.e. $\mathbb{P}(\hat{y} = y_{\text{true}} \mid \hat{p}_{\hat{y}} = p) = p, \ \forall p \in [0, 1]$ [4]. Continuing our example, if the network assigns an average confidence score of 0.8 to a set of predictions then we should expect 80% of those to be correct.

To quantify calibration, we use *Calibration Error* as $\mathcal{E} = \hat{p}_{\hat{y}} - \mathbb{P}(\hat{y} = y_{\text{true}} \mid \hat{p}_{\hat{y}})$ and the *Expected Calibration Error* as $\mathbb{E}_{\hat{p}_{\hat{y}}}[\mathcal{E}] = \mathbb{E}_{\hat{p}_{\hat{y}}}[\ |\hat{p}_{\hat{y}} - \mathbb{P}(\hat{y} = y_{\text{true}} \mid \hat{p}_{\hat{y}})|\ ]$ [4]. Since the true calibration error cannot be computed empirically with a finite sized dataset, the following approximations are generally used in the literature. For a dataset $\{(\mathbf{x}_n, y_{\text{true},n})\}_{n=1}^N$, (1) $\text{ECE}_{\text{EW}} = \sum_{i=1}^M \frac{|B_i|}{N} |C_i - A_i|$ [4], where B_i is equal-width (EW) bin that contains all examples j with $\hat{p}_{\hat{y},j}$ in the range $[\frac{i}{M}, \frac{i+1}{M})$, $C_i = \frac{1}{|B_i|} \sum_{j \in B_i} \hat{p}_{\hat{y},j}$ is the average confidence and $A_i = \frac{1}{|B_i|} \sum_{j \in B_i} \mathbb{1}(\hat{y}_j = y_{\text{true},j})$ is the bin accuracy. Note that $E_i = C_i - A_i$ is the empirical approximation of the calibration error $\mathcal{E}$, (2) $\text{ECE}_{\text{EM}} = \sum_{i=1}^M \frac{|B_i|}{N} |C_i - A_i|$ [24], where $\forall i, j \ |B_i| = |B_j|$ are equal-mass (EM) bins. However, as ECE has been shown to be a biased estimate of the true calibration error [29], we additionally use $\text{ECE}_{\text{DEBIAS}}$ [11] and $\text{ECE}_{\text{SWEEP}}$ [26] to further corroborate the results in the paper.

3 Calibration Properties of Focal Loss

Focal loss [16] $\mathcal{L}_{FL}(p) = -(1-p)^\gamma \log p$ was originally proposed to improve the accuracy of classifiers by focusing on hard examples and down-weighting well classified examples. Recently, it was shown that focal loss can also be used to significantly improve calibration [19]. This is because, based on the relation $\mathcal{L}_{FL} \geq KL(q||\hat{\mathbf{p}}) - \gamma \mathbb{H}(\hat{\mathbf{p}})$ (q is the one-hot target vector), focal loss while minimising the KL divergence objective also increases the entropy of the prediction $\hat{\mathbf{p}}$. This prevents the network from being overly confident on wrong predictions thereby improving calibration.

However, focal loss with fixed γ, as we show next, falls short of achieving the best calibration. In Figure 1, we plot the calibration behaviour of ResNet50 in different probability regions (bins) when trained on CIFAR-10 with different focal losses. The ith bin's calibration error $E_{val,i} = C_{val,i} - A_{val,i}$ is computed on the validation set using 15 equal-mass binning. The figure plots a lower (bin-0), middle (bin-7) and higher bin (bin-14) (rest of the bins and their bin boundaries are shown in Appendix B). From Figure 1 (a), although focal loss $\gamma = 4$ achieves the overall lowest ECE_{EM}, there's no single γ that performs the best across all the bins. For example, in bin-0, $\gamma = 4, 5$ achieves better calibration whereas $\gamma = 0, 3$ are over-confident. On the other hand, in bin-7 $\gamma = 3$ seems to be better calibrated whereas $\gamma = 4, 5$ are under-confident and $\gamma = 0$ is over-confident.



(a) ECE_{EM} (%) (b) $C_{val,i} - A_{val,i}$

Figure 1: Calibration behaviour of ResNet-50 trained on CIFAR-10 with cross entropy (CE), focal loss $\gamma = 3, 4, 5$ (FL-3/4/5) and FLSD-53. These are computed using 15 equal-mass binning on the validation set. (a) ECE_{EM}, and (b) Calibration error $C_{val,i} - A_{val,i}$ for a lower (bin-0), middle (bin-7), and upper (bin-14) bin. The black horizontal lines in (b) represent $E_{val,i} = 0$. These exemplify the downside of regular focal loss i.e. although $\gamma = 4$ achieves the overall lowest calibration error, the best performing γ is different for different bins.

This shows that using different γs for different bins can further improve calibration. Such an attempt called the Sample-Dependent Focal Loss (FLSD-53) is presented in [19] that assigns $\gamma = 5$ if the training sample's true class posterior $\hat{p}_{y_{\text{true}}} \in [0, 0.2)$ and $\gamma = 3$ if $\hat{p}_{y_{\text{true}}} \in [0.2, 1]$. However, from Figure 1(b), FLSD-53 is also not the best calibrated across all bins. Rather, this is a fixed strategy based on heuristics of choosing higher γ for smaller values of $\hat{p}_{y_{\text{true}}}$ and relatively lower γ for higher values of $\hat{p}_{y_{\text{true}}}$. This, therefore, motivates the design of a strategy that can assign appropriate γ_is for each bin-i based on the magnitude and sign of $E_{val,i}$.

To design such a strategy one, however, faces two challenges:

1. How do we find some correspondence between the confidence of training samples (which we can manipulate during training using the parameter γ) and the confidence of the validation/test samples (which is our actual target but we do not have direct control over)? In other words, to indirectly control the confidence of a group of validation samples, how do we know which particular group of training samples' confidence to be manipulated?
2. Given that a correspondence is established, how do we arrive at the appropriate values of γ that will lead to the best calibration?

We try to answer the first question in the next section and answer to the second question leads to the main contribution of the paper: AdaFocal.

Additionally, alongside focal loss, we will make use of the inverse-focal loss [30, 17] for cases where no γ of regular focal loss is able to avoid under-confidence. See, for example, ResNet-50 trained on ImageNet in section 6 and Fig. 5 where even cross entropy ($\gamma = 0$) doesn't achieve the desired level of confidence. Inverse-focal loss, plotted in Fig. 3(a), and given by

$$\mathcal{L}_{InvFL}(p) = -(1 + p)^{\gamma} \log p, \tag{1}$$

serves the opposite purpose of focal loss. While focal loss reduces the over-confidence of the network, inverse-focal loss helps recover from under-confidence by providing larger gradients to the samples with higher confidences (easy samples) by pushing their score even further.

4 Correspondence between Confidence of Training and Validation Samples

One way to check for any correspondences is to simply group the validation samples into M equal-mass bins (henceforth called validation-bins) and compare the confidence with the training samples that fall into the same validation-bins. But first we clarify a few notations that are of interest.

Quantities of interest For binning validation samples, we look at the confidence of the top predicted class $\hat{y}$ denoted by $\hat{p}_{val,top}$ (bin average: $C_{val,top}$). For training samples, on the other hand, instead of the confidence of the top predicted class $\hat{y}$ denoted by $\hat{p}_{train,top}$ (bin average: $C_{train,top}$), we will focus on the confidence of the true class y_{true} denoted by $\hat{p}_{train,true}$ (average: $C_{train,true}$) because during training we only care about $\hat{p}_{train,true}$ which gets manipulated by some loss function (for reference, Figure 8 in Appendix C compares $C_{train,true}$ and $C_{train,top}$ to show that as the training

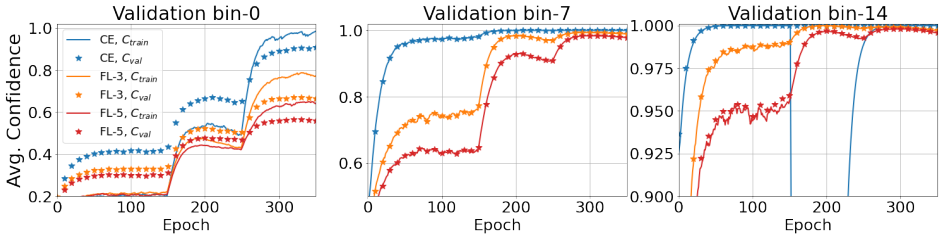


Figure 2: Correspondence between confidence of training (C_{train}) and validation samples (C_{val}) for ResNet-50 trained on CIFAR-10 with focal loss $\gamma = 0$ (CE), 3, 5. The binning involves 15 equal-mass bins where training samples are grouped using validation-bin boundaries. A lower (bin-0), middle (bin-7) and upper (bin-14) bins are shown here (with the rest shown in Fig. 10 in Appendix C).

accuracy approaches 100%, the top predicted class and the true class become the same). For brevity, we will refer to $C_{train} \equiv C_{train,true}$ and $C_{val} \equiv C_{val,top}$.

In Figure 2, we compare $C_{train,i}$ in validation-bin-i [1] with $C_{val,i}$ and find that there is indeed a good correspondence between the two quantities. For example in Figure 2, as γ increases from 0 (CE), to 3 to 5, the solid-line $C_{train,i}$ gets lower, and the same behaviour is observed for the starred-line $C_{val,i}$. For more evidence refer to Fig. 11, 12, and 13 in Appendix C where similar behaviour is observed for ResNet-50 and WideResNet on CIFAR-100, and ResNet-50 on TinyImageNet, respectively.

Additionally, we look at the case when training samples and validation samples are grouped independently into their respective training-bins and validation-bins. Figure 9 in Appendix C compares $C_{train,i}$ in training-bin-i with $C_{val,i}$ in validation-bin-i. Here again we observe a similar correspondence. Further, note that since the binning is independent, the boundaries of training-bin-i will not be exactly the same as that of validation-bin-i, however as shown in Figure 9 they are very close.

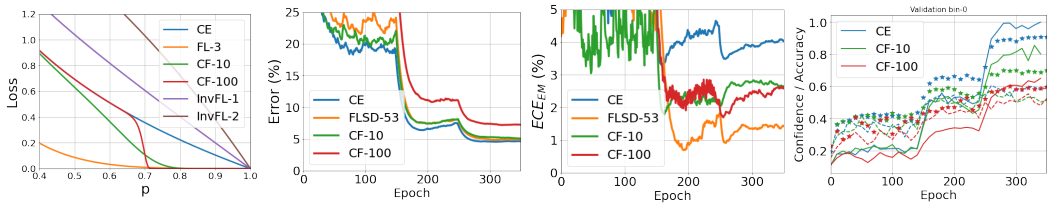
This observation is very encouraging as now we have a way to indirectly control $C_{val,i}$ by manipulating $C_{train,i}$ i.e. we can expect (even though loosely) that if we increase/decrease the confidence of a group of training samples in some lower (or middle, or higher) probability region then the same will get reflected on a group of validation samples in a similar lower (or middle, or higher) probability region. Therefore, from a calibration point of view, our strategy going forward would be to exploit this correspondence to keep $C_{train,i}$ (which we have control over during training) closer to $A_{val,i}$ (the validation set accuracy in validation-bin-i) so that $C_{val,i}$ also stays closer to $A_{val,i}$ and therefore achieve a very low calibration error $C_{val,i} - A_{val,i}$. For simplifying the design of AdaFocal, we will employ the first method of common binning i.e. using validation-bins to bin the training samples.

5 Proposed Method

Let's denote the nth training sample's true class posterior $\hat{p}_{y_{\text{true}}}$ by p_n. Given that p_n falls into validation-bin-b, our goal is to keep p_n (or its averaged equivalent $C_{train,b}$) closer to $A_{val,b}$ so that the same is reflected on $C_{val,b}$. For manipulating p_n, we will utilize the regularization effect of focal loss's parameter γ. At this point, one can choose to update the γ of validation-bin-b denoted by γ_b either based on (1) how far p_n is from $A_{val,b}$ i.e. $\gamma = g(p_n - A_{val,b})$ or (2) how far $C_{val,b}$ is from $A_{val,b}$ i.e. $\gamma = g(C_{val,b} - A_{val,b})$. Such a γ-update-rule should ensure that whenever the model is over-confident, i.e. $p_n > A_{val,b}$ (or $C_{val,b} > A_{val,b}$), γ is increased so that the gradients get smaller to prevent p_n from increasing further. On the other hand, when $p_n < A_{val,b}$ (or $C_{val,b} < A_{val,b}$), i.e. the model is under-confident, we decrease γ so as to get larger gradients that in turn will increase p_n[2]. To test this strategy, we first design a calibration-aware γ-update method (called CalFocal), which with some additional modifications will lead to the final AdaFocal algorithm.

[1] It may happen that no training sample falls into a particular validation-bin. In that case, $C_{train,i}$ is shown to drop to zero, for example in bin-14 in Figure 2.

[2] Note that for focal loss increasing γ does not always lead to smaller gradients. This mostly holds true in the region p_n approximately > 0.2 (see Figure 3(a) in [19]). However, in practice and as shown by the training-bin boundaries of bin-0 and bin-1 in Figure in Figure 9 Appendix C, we find majority of the training samples to lie above 0.2 during the majority of the training, and therefore, for the experiments in this paper, we simply stick to the rule of increasing γ to decrease gradients and stop p_n from increasing and vice versa.



(a) Loss functions (b) Error (%) (c) ECE_{EM} (%) (d) C_{train}, C_{val}, A_{val}

Figure 3: (a) plots different loss functions: Cross Entropy (CE), Focal loss (FL-γ), Inverse-Focal loss (InvFL-γ), CalFocal (CF-λ) with $A_{val,b} = 0.8$. Subfigures (b) and (c) plot error and ECE_{EM}, respectively, for ResNet-50 trained on CIFAR-10 with CalFocal. (d) compares C_{train} (solid lines), C_{val} (starred lines) and A_{val} (dashed lines) in validation bin-0 to show that when CalFocal brings C_{train} closer to A_{val}, C_{val} also approaches A_{val}.

5.1 Calibration-aware Focal Loss (CalFocal)

Case 1: $\gamma = g(p_n - A_{val,b})$ Treating $A_{val,b}$ as the point that we want p_n to not deviate from, we make the focal loss parameter γ a function of $p_n - A_{val,b}$ to get

$$\mathcal{L}_{CalFocal}(p_n) = -(1 - p_n)^{\gamma_n} \log p_n, \quad \text{where } \gamma_n = \exp(\lambda(p_n - A_{val,b})), \tag{2}$$

and b is the validation-bin in which p_n falls. The hyper-parameter λ is the scaling factor which combined with the exponential function helps to quickly ramp up/down γ. The exponential function adheres to the γ-*update rule* mentioned earlier. Figure 3(a) plots $\mathcal{L}_{CalFocal}$ vs. p_n for $A_{val,b} = 0.8$. We see that based on the strength of λ, the loss drastically drops near $p_n = 0.8$ and thereafter remains close to zero. This shows that $\mathcal{L}_{CalFocal}$ aims is to first push p towards 0.8 and then slow its growth towards overconfidence. Next, in Figure 3(c), we find that CalFocal with $\lambda = 10, 100$ is able to reduce the calibration error (ECE_{EM}) but it is still far from FLSD-53's performance. Also note in Figure 3(b) that too high λ (=100) affects the accuracy of model. Most interesting is Fig. 3(d) which compares $C_{train,i}$ with $C_{val,i}$ (and also $A_{val,i}$) for bin-0, where we find evidence that the strategy of bringing p_n or $C_{train,i}$ (solid lines) closer to $A_{val,i}$ (dashed lines) results in $C_{val,i}$ (starred lines) getting closer to $A_{val,i}$ as well, thus reducing the calibration error $E_{val,i} = C_{val,i} - A_{val,i}$.

Case 2: $\gamma = g(C_{val,b} - A_{val,b})$ Note that Eq. 2 assigns a different γ_n for each training sample p_n. To reduce computation and avoid keeping track of γ_n for each training sample, we can assign one γ_b to each validation-bin-b by making it a function of $C_{val,b} - A_{val,b}$ (instead of $p_n - A_{val,b}$). Then all p_n that fall into the validation-bin-b are assigned γ_b and the loss function is modified to

$$\mathcal{L}_{CalFocal}(p_n) = -(1 - p_n)^{\gamma_b} \log p_n, \quad \text{where } \gamma_b = \exp(\lambda(C_{val,b} - A_{val,b})) \tag{3}$$

b is the validation-bin in which p_n falls. The performance of this strategy, as shown in Appendix D, is very similar (or slightly better than) Eq. 2. Further, it makes more sense to update γ based on how far $C_{val,b}$ is from $A_{val,b}$ instead of how far p_n is from $A_{val,b}$ because, as shown in Figure 3(d) bin-0, one may find $C_{val,b}$ (starred lines) quite closer to $A_{val,b}$ (dashed lines) even when p_n or C_{train} (solid lines) is still far from $A_{val,b}$. At this point where $C_{val,b} = A_{val,b}$, we should stop updating γ further, even if $p_n - A_{val,b} \neq 0$, as we have reached our goal of making $E_{val,b} = C_{val,b} - A_{val,b} = 0$. Therefore, for AdaFocal we will use Eq. 3 as the base for AdaFocal loss function.

Limitations of CalFocal: (1) Let's say at some step of training, a high γ_b over some epochs reduces the error $C_{val,b} - A_{val,b}$. Then, it is desirable to continue training with the same high γ_b. However, note CalFocal's update rule in Eq. 3 which will reduce $\gamma \to 1$ as the $C_{val,b} - A_{val,b} \to 0$. (2) Let's say, at some point $C_{val,b} - A_{val,b}$ is quite high. This will set γ_b to some high value depending on the hyper-parameter λ. Assuming this γ_b is still not high enough to bring down the confidence, we would want a way to further increase γ_b. However, CalFocal is incapable of doing so as it will continue to hold at $\gamma_b = \exp(\lambda(C_{val,b} - A_{val,b}))$.

5.2 Calibration-aware Adaptive Focal Loss (AdaFocal)

We propose to address the above limitations by making $\gamma_{t,b}$ depend on $\gamma_{t-1,b}$ from previous time step

$$\gamma_{t,b} = \gamma_{t-1,b} * \exp(\lambda(C_{val,b} - A_{val,b})). \tag{4}$$

This update rule address the limitations of CalFocal in the following way: let's say at some point we observe over-confidence i.e. $E_{val,b} = C_{val,b} - A_{val,b} > 0$. Then, in the next step γ_b will be increased. In the subsequent steps, it will continue to increase unless the calibration error $E_{val,b}$ starts decreasing (this additional increase in γ was not possible with CalFocal). At this point, if we find $E_{val,b}$ to start decreasing, that would reduce the increase in γ_b over the next epochs and γ_b will ultimately settle down to a value when $E_{val,b} = 0$ (CalFocal at $E_{val,b} = 0$ will cause γ to go down to 1). Next, if this current value of γ_b starts causing under-confidence i.e. $C_{val,b} - A_{val,b} < 0$, then the update rule will kick in to reduce γ thus allowing $C_{val,b}$ to be increased back to $A_{val,b}$. This oscillating behaviour of AdaFocal around the desired point of $C_{val,b} = A_{val,b}$ is its main strength in reducing the calibration error in every bin.

Next, to deal with cases where even cross entropy suffers from under-confidence, we switch to inverse-focal loss which can further increase the confidence of the predictions. For the switch between focal and inverse-focal loss, we simply set a threshold S_{th} below which if gamma falls, we switch to the other loss function. Note here that, for notational purpose, we will denote the inverse-focal loss by a negative γ i.e. $\gamma_{t,b} > 0$ means focal loss with parameter $\gamma_{t,b}$ whereas $\gamma_{t,b} < 0$ implies inverse-focal loss with parameter $|\gamma_{t,b}|$. The complete gamma-update rule and the switching criteria is given in Algorithm 1. If not stated explicitly, we use $S_{th} = 0.2$ for all AdaFocal experiments. For reference, we also plot results with $S_{th} = 0.5$ for ImageNet, ResNet-50 in Fig. 4 (d) and Fig. 5(b).

Finally, note the unbounded exponential update in Eq. 4 which is an undesirable property. This may easily cause γ_t to explode as when expanded $\gamma_t = \gamma_{t-1} \exp(E_{val,t}) = \gamma_0 \exp(E_{val,0} + E_{val,1} + ... + E_{val,t-1} + E_{val,t})$, and if $E_{val,t} > 0$ for quite a few number of epochs, γ_t will become so large that even if $E_{val,t} < 0$ in the subsequent epochs, it may not reduce to a desired level. We remedy this by constraining γ_t to an upper bound $\gamma_{\max}$ when $\gamma > 0$ (focal loss) and lower bound of $\gamma_{\min}$ when $\gamma < 0$ (inverse-focal loss). One may argue that $\gamma_{\max}$ and $\gamma_{\min}$ are again hyper-parametera; however, note that they do not require any special fine-tuning and their sole purpose is to stop γ from exploding. For all our experiments, we use $\gamma_{\min} = -2$ and $\gamma_{\max} = 20$. For comparison of results with $\gamma_{\max} = 20$, $\gamma_{\max} = 50$ and $\gamma_{\max} = \infty$, please refer to Appendix O. The choice of $\gamma_{\min} = -2$ is based on the observation that on ImageNet values beyond -2 led to unstable training for ResNet-50. Additionally, note that although λ is included in the exponent of Eq. 4, we find $\lambda = 1$ to work well for all our experiments, therefore, one can treat AdaFocal to be free of this hyper-parameter.

Therefore, the final loss function for AdaFocal is given by

$$\mathcal{L}_{AdaFocal}(p_n, t) = \begin{cases} -(1-p_n)^{\gamma_{t,b}} \log p_n, & \text{if } \gamma_{t,b} \geq 0 \\ -(1+p_n)^{|\gamma_{t,b}|} \log p_n, & \text{if } \gamma_{t,b} < 0, \end{cases} \tag{5}$$

where the gamma-update rules, along with the complete algorithm, is presented in Algorithm 1.

6 Experiments

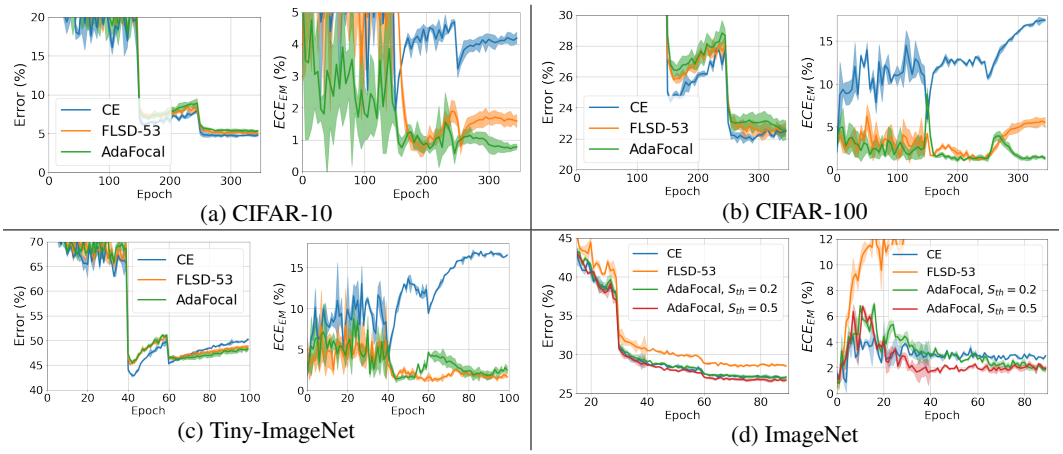
Experimental setup. We evaluate the performance of our proposed method on image and text classification tasks. For image classification, we use CIFAR-10, CIFAR-100 [9], Tiny-ImageNet [2], and ImageNet [27] to analyze the calibration of ResNet50, ResNet-100 [6], Wide-ResNet-26-10 [33], and DenseNet-121 [8] models. For text classification, we use the 20 Newsgroup dataset [14] and train global-pooling CNN [15] and fine-tune a pre-trained BERT model [3]. More details about datasets, models and experimental configurations are given in Appendix E. In the main paper, we report results using only $\mathrm{ECE}_{\mathrm{EM}}$, whereas other ECE metrics are reported in Appendix.

Baseline. Among calibration-during-training methods we use MMCE [13], Brier loss [1], Label smoothing (LS-0.05) [20] and sample-dependent focal loss FLSD-53 as baselines. For post hoc calibration, we report the effect of temperature scaling, ensemble temperature scaling (ETS) [34] and spline fitting [5] on top of these methods. For temperature scaling, we select the optimal temperature $\in (0, 10]$ (step size 0.1) that gives the lowest $\mathrm{ECE}_{\mathrm{EM}}$ on the validation set.

Results. In Figure 4, we compare AdaFocal against cross entropy (CE) and FLSD-53 for ResNet-50 trained on various small to large-scale image datasets. Among various focal losses, we chose FLSD-53 as our baseline because it was shown to be consistently better than MMCE, Brier Loss and Label smoothing [19] across many datasets-model pairs. The figure plots the test set error and $\mathrm{ECE}_{\mathrm{EW}}$. In Figure 5, for ResNet-50 on CIFAR-10 and ImageNet, we plot (1) the calibration statistics

Algorithm 1: AdaFocal

Input: $D_{train} = \{\mathbf{x}_n, y_{true}\}_{n=1}^{N_{train}}$, $D_{val} = \{\mathbf{x}_n, y_{true}\}_{n=1}^{N_{val}}$;
Bin initialization: **for** $i = 1$ **to** M **do**
  $B_{t=0,i} = \left(\frac{i-1}{M}, \frac{i}{M}\right]$, $\gamma_{t=0,i} = 1$ // Initialize validation-bins to equal-width with gamma set to 1;
Training: **for** $t = 0$ **to** T **do**
  $Loss_t = 0$;
  for $n = 1$ **to** N_{train} **do**
    $p_n = f_{\theta_t}(\mathbf{x}_n)$ // Denoting $p_{y_{true,n}}$ by p_n;
    b = get_bin_index$(p_n, \{B_{t,i}\})$ // Bin in which p_n lies;
    if $\gamma_{t,b} \geq 0$ **then**
      $Loss_t \mathrel{+}= -(1-p_n)^{\gamma_{t,b}} \log p_n$ // Focal loss;
    else if $\gamma_{t,b} < 0$ **then**
      $Loss_t \mathrel{+}= -(1+p_n)^{|\gamma_{t,b}|} \log p_n$ // Inverse-focal loss;
  $\theta_{t+1} = \text{gradient_update}(\theta_t, Loss_t)$;
  γ**-update step**: **for** $i = 1$ **to** M **do**
    Re-compute $B_{t+1,i}$, $C_{val,t+1,i}$ and $A_{val,t+1,i}$ using the updated model $f_{\theta_{t+1}}$ and D_{val};
    if $\gamma_{t,i} \geq 0$ **then**
      $\gamma_{t+1,i} = \min\left\{\gamma_{\max},\ \gamma_{t,i} * e^{\lambda(C_{val,t+1,i} - A_{val,t+1,i})}\right\}$ // Focal loss γ-update;
      if $|\gamma_{t+1,i}| < S_{th}$ **then**
        $\gamma_{t+1,i} = -S_{th}$ // Switch to inverse-focal loss;
    else if $\gamma_{t,i} < 0$ **then**
      $\gamma_{t+1,i} = \max\left\{\gamma_{\min},\ \gamma_{t,i} * e^{-\lambda(C_{val,t+1,i} - A_{val,t+1,i})}\right\}$ // Inverse-focal γ-update;
      if $|\gamma_{t+1,i}| < S_{th}$ **then**
        $\gamma_{t+1,i} = S_{th}$ // Switch to focal loss;

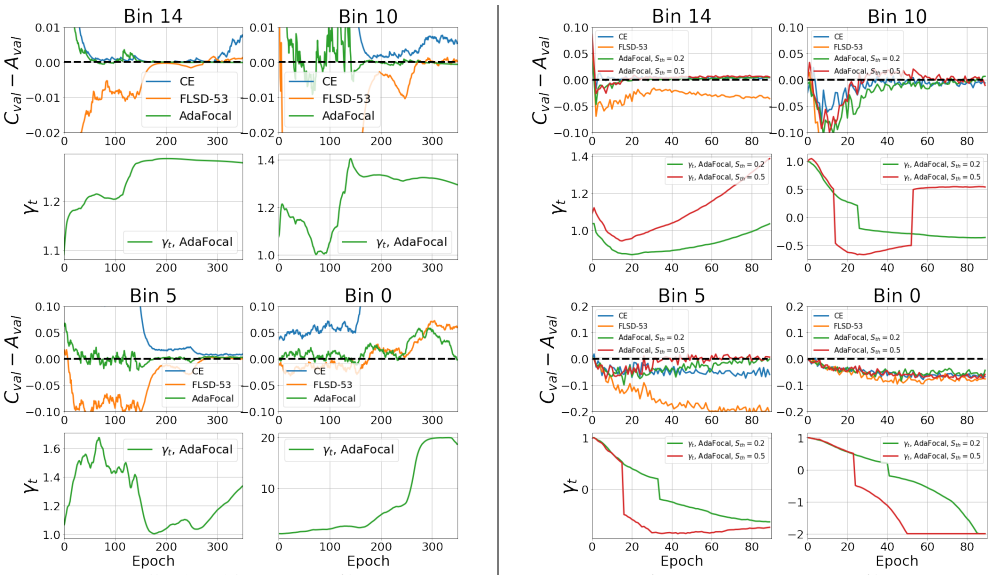


(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet (d) ImageNet

Figure 4: ResNet-50 trained with cross entropy (CE), FLSD-53 and AdaFocal. In each subfigure, **left**: Error (%), **right**: ECE_{EM} (%) on the test set are plotted for mean and standard deviation over 5 runs. We observe that throughout the training AdaFocal maintains a low calibration error while achieving similar accuracy.

$E_{val} = C_{val} - A_{val}$ of the validation set used by AdaFocal during training and (2) the dynamics of γ_t for a few bins in lower, middle, and higher probability regions.

First, we observe that for CIFAR-10, CIFAR-100 and Tiny-ImageNet, FLDS-53 is much better calibrated than CE. This is because, as shown in Fig. 5(a) for ResNet-50, CIFAR-10, CE is over-confident compared to FLSD-53 in almost every bin. For ImageNet, however, the behaviour is reversed: FLSD-53 is poorly calibrated than CE. The reason, as shown in Figure 5(b), is that due to the use of high values of γ $(= 5, 3)$, FLSD-53 makes the model largely under-confident in each bin, leading to a overall high calibration error. AdaFocal, on the other hand, is well calibrated throughout the training for all dataset-model pairs.



(a) CIFAR-10, ResNet-50 (b) ImageNet, ResNet-50

Figure 5: Dynamics of γs in different validation-bins when ResNet-50 is trained on CIFAR-10 and ImageNet with AdaFocal. For each bin, **top**: $C_{val} - A_{val}$ from the validation set, **bottom**: γ_t vs. epochs. Black dotted line in top plot represent zero calibration error. We observe that for each bin AdaFocal is able to find the gammas that result in low calibration error. For CIFAR-10, $\gamma_t > 1$ whereas for ImageNet, starting from $\gamma_t = 1$ (focal loss), AdaFocal ultimately switches to inverse-focal loss ($\gamma_t < 0$) at $S_{th} = 0.2$ for most of the bins.

Next, for ResNet-50 CIFAR-10 we find γ_t to be closer to 1 for higher bins and closer to 20 for lower bins. These gammas result in better calibration than $\gamma_t = 5, 3$ of FLSD-53. For ImageNet, on the other hand, we find, except bin-14, AdaFocal switches to inverse-focal loss at some point in the training. This makes sense because for ImageNet even cross entropy ($\gamma_t = 0$) suffers from under-confidence, therefore, AdaFocal, starting from $\gamma_t = 1$, approaches cross entropy ($\gamma = 0$) to ultimately switch to inverse-focal loss ($\gamma_t < 0$) at $S_{th} = 0.2$. Since, in retrospect, we already know that switching to inverse focal loss is beneficial for ImageNet, switching early at $S_{th} = 0.5$ helps reach a better level of calibration early. However, ultimately $S_{th} = 0.2$ and $S_{th} = 0.5$ lead to the same calibration error. Overall, these experiments confirm that during training AdaFocal being aware of the network's current under/over-confidence is able to guide the γ_ts to values that maintain a well calibrated model at all times. Also note that for an unseen dataset-model pair there's no way to know apriori which γ will perform better but AdaFocal will automatically find these appropriate values thus avoiding an expensive and extensive hyper-parameter search.

Comparison of test set error and ECE_{EM} for rest of the experiments are shown in Table 2 and Table 1 respectively. From Table 1, we observe that prior to temperature scaling AdaFocal outperforms the baseline methods by a substantial margin in all 13 cases. Post-temperature scaling, AdaFocal achieves the lowest calibration error in 11 out of the 13 cases. Also observe that the optimal temperatures are mostly close to 1 indicating that AdaFocal produces innately calibrated models from training itself. Effects of other post hoc calibration methods (ETS and spline fitting) are shown in Appendix F. Again, we observe that pre-calibrated models from AdaFocal benefit from post processing to further lower the overall calibration error. Besides ECE_{EM}, the consistency of the results across other calibration metrics is shown through $\text{ECE}_{\text{DEBIAS}}$, $\text{ECE}_{\text{SWEEP}}$ (equal-width and equal-mass) in Appendix G. Statistical significance of the results is confirmed through ECE_{EW} error bars in Appendix H where mean and standard deviations are plotted over 5 runs.

Number of bins. For AdaFocal, we experiment with 5, 10, 15, 20, 30, and 50 equal-mass bins used during training to draw calibration statistics form the validation set. As reported in Appendix I, the best results are in the range of 10 to 20. Performance degrades when the number of bins are too small (< 10) or too large (> 20). Therefore, for AdaFocal we use 15 bins as well during training. For computing ECE metrics, if not stated explicitly, we always use 15 bins.

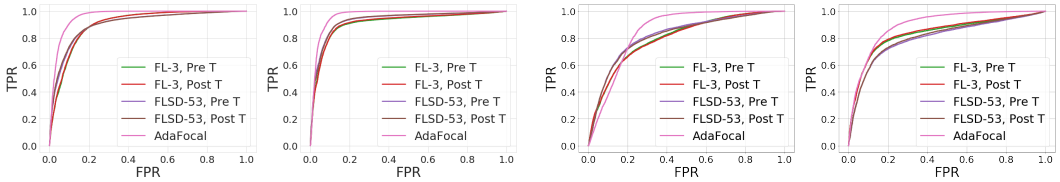
Dataset	Model	Pre Temperature scaling						Post Temperature scaling					
		CE	Brier	MMCE	LS	FLSD-53	AdaFocal	CE	Brier	MMCE	LS	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	4.24	1.78	4.52	3.86	1.63	**0.66**	2.11(2.52)	1.24(1.11)	2.12(2.65)	2.97(0.92)	1.42(1.08)	**0.44(1.06)**
	ResNet-110	4.39	2.63	5.16	4.44	1.90	**0.71**	2.27(2.74)	1.75(1.21)	2.53(2.83)	4.44(1.00)	1.25(1.20)	**0.73(1.02)**
	Wide-ResNet	3.42	1.72	3.31	4.26	1.82	**0.64**	1.87(2.16)	1.72(1.00)	1.6(2.22)	2.44(0.81)	1.57(0.94)	**0.44(1.06)**
	DenseNet-121	4.26	2.09	5.05	4.40	1.40	**0.62**	2.21(2.33)	2.09(1.00)	2.26(2.52)	3.31(0.94)	1.40(1.00)	**0.59(1.02)**
CIFAR-100	ResNet-50	17.17	6.57	15.28	7.86	5.64	**1.36**	3.71(2.16)	3.66(1.13)	2.32(1.80)	4.10(1.13)	2.97(1.17)	**1.36(1.00)**
	ResNet-110	19.44	7.70	19.11	11.18	7.08	**1.40**	6.11(2.28)	4.55(1.18)	4.88(2.32)	8.58(1.09)	3.85(1.20)	**1.40(1.00)**
	Wide-ResNet	14.83	4.27	13.12	5.10	2.25	**1.95**	3.23(2.12)	2.85(1.08)	4.23(1.91)	5.10(1.00)	2.25(1.00)	**1.95(1.00)**
	DenseNet-121	19.82	5.14	19.16	12.81	2.58	**1.73**	3.62(2.27)	2.58(1.09)	3.11(2.13)	8.95(1.19)	1.80(1.10)	**1.73(1.00)**
Tiny-ImageNet	ResNet-50	7.81	3.42	8.49	9.12	2.86	**2.61**	3.73(1.45)	2.98(0.93)	4.25(1.36)	4.66(0.78)	2.48(1.05)	**2.29(0.96)**
	ResNet-110	8.11	3.74	7.40	9.36	1.88	**1.85**	1.93(1.20)	2.83(0.91)	1.95(1.20)	4.51(0.83)	1.88(1.00)	**1.85(1.00)**
ImageNet	ResNet-50	2.93	3.91	9.30	10.05	16.77	**1.87**	**1.50(0.88)**	3.59(0.92)	4.22(1.34)	4.53(0.82)	2.62(0.74)	1.87(1.00)
20Newsgroup	CNN	18.57	13.52	15.23	4.36	8.86	**2.62**	4.08(3.78)	3.13(2.33)	6.45(2.21)	2.62(1.12)	**2.13(1.58)**	2.46(1.10)
	BERT	8.47	5.91	8.30	6.01	8.63	**3.96**	4.46(1.44)	4.40(1.24)	4.60(1.46)	5.69(1.14)	3.91(0.80)	**3.73(1.04)**

Table 1: Test $\mathrm{ECE}_{\mathrm{EM}}$ (%) averaged over 5 runs. Bold marks the lowest in pre and post temperature scaling groups separately. Optimal temperature, given in brackets, is cross-validated on $\mathrm{ECE}_{\mathrm{EM}}$.

Dataset	Model	CE	Brier	MMCE	LS	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	**4.95**	5.00	4.99	5.29	4.98	5.30
	ResNet-110	**4.89**	5.48	5.40	5.52	5.42	5.27
	Wide-ResNet-26-10	**3.86**	4.08	3.91	4.20	4.01	4.50
	DenseNet-121	**5.00**	5.11	5.41	5.09	5.46	5.20
CIFAR-100	ResNet-50	23.30	23.39	23.20	23.43	23.22	**22.60**
	ResNet-110	22.73	25.10	23.07	23.43	**22.51**	22.79
	Wide-ResNet-26-10	20.70	20.59	20.73	21.19	20.11	**20.07**
	DenseNet-121	24.52	23.75	24.0	24.05	22.67	**22.22**
Tiny-ImageNet	ResNet-50	**42.90**	46.27	45.96	44.42	45.12	45.49
	ResNet-110	42.53	45.47	**42.22**	44.13	44.88	44.55
ImageNet	ResNet-50	27.08	28.80	27.12	28.43	28.53	**27.07**
20Newsgroup	CNN	26.68	27.06	27.23	**26.03**	27.98	28.53
	BERT	**16.05**	16.52	16.16	16.18	17.57	17.22

Table 2: Test set error (%). Lowest error is marked in bold.

Out-of-Distribution (OOD) detection. Following [19], we report the performance of AdaFocal on an OOD detection task. We train ResNet-110 and Wide-ResNet26-10 on CIFAR-10 as the in-distribution data and test on SVHN [23] and CIFAR-10-C [7] (with level 5 Gaussian noise corruption) as OOD data. Using entropy of the softmax as the measure of uncertainty, the corresponding ROC plots are shown in Figure 6 and AUROC scores are reported in Table 8 in Appendix L. We see that models trained with AdaFocal outperform focal loss $\gamma = 3$ (FL-3) and FLSD-53. For the exact AUROC scores, please refer to Appendix L. These results further highlight the benefits of inherently calibrated model produced by AdaFocal as post-hoc calibration techniques such as temperature scaling, as shown in the figure, is ineffective under distributional shift [28].



(a) SVHN: ResNet-110, WideResNet (b) CIFAR-10-C: ResNet-110, WideResNet

Figure 6: ROC for ResNet-110 and Wide-ResNet-26-10 trained on in-distribution CIFAR-10 and tested on out-of-distribution (a) SVHN and (b) CIFAR-10-C. Pre/Post T refers to pre and post temperature scaling.

7 Conclusion

In this work, we first revisit the calibration properties of regular focal loss to highlight the downside of using a fixed γ for all samples. Particularly, by studying the calibration behaviour of different samples in different probability region, we find that there's no single γ that achieves the best calibration over the entire region. We use this observation to motivate the selection of γ independently for each sample (or group of samples) based on the knowledge of models's under/over-confidence. We propose a calibration-aware adaptive strategy called AdaFocal that accounts for such information and updates the γ_t at every step based on γ_{t-1} from the previous step and the magnitude of the model's under/over-confidence. We find AdaFocal to perform consistently better across different dataset-model pairs producing innately calibrated models that benefit further from post-hoc calibration techniques. Additionally, we find models trained with AdaFocal to be much better at out-of-distribution detection.

References

[1] Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 1950.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2017.

[5] Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of neural networks using splines. In *International Conference on Learning Representations*, 2021.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[8] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[9] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[10] Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. In *Advances in Neural Information Processing Systems*, 2019.

[11] Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[12] Aviral Kumar. 20 newsgroups mmce. `https://github.com/aviralkumar2907/MMCE`, 2018.

[13] Aviral Kumar, Sunita Sarawagi, and Ujjwal Jain. Trainable calibration measures for neural networks from kernel mean embeddings. In *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2018.

[14] Ken Lang. Newsweeder: Learning to filter netnews. In *in Proceedings of the 12th International Machine Learning Conference (ML95*, 1995.

[15] M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, 1312.4400, 2014.

[16] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[17] Mingsheng Long, ZHANGJIE CAO, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[18] Jishnu Mukhoti. Focal calibration. `https://github.com/torrvision/focal_calibration`, 2020.

[19] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip HS Torr, and Puneet K Dokania. Calibrating deep neural networks using focal loss. In *Advances in Neural Information Processing Systems*, 2020.

[20] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, 2019.

[21] Mahdi Pakdaman Naeini and Gregory F Cooper. Binary classifier calibration using an ensemble of near isotonic regression models. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, 2016.

[22] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 2015.

[23] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[24] Khanh Nguyen and Brendan O'Connor. Posterior calibration and exploratory analysis for natural language processing models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2015.

[25] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014.

[26] Rebecca Roelofs, Nicholas Cain, Jonathon Shlens, and Michael C. Mozer. Mitigating bias in calibration error estimation, 2021.

[27] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[28] Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D. Sculley, Joshua V. Dillon, Jie Ren, and Zachary Nado. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019.

[29] Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön. Evaluating model calibration in classification. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89, pages 3459–3467. PMLR, 16–18 Apr 2019.

[30] Deng-Bao Wang, Lei Feng, and Min-Ling Zhang. Rethinking calibration of deep neural networks: Do not be afraid of overconfidence. In *Advances in Neural Information Processing Systems*, volume 34, pages 11809–11820. Curran Associates, Inc., 2021.

[31] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, 2001.

[32] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *KDD*, 2002.

[33] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[34] Jize Zhang, Bhavya Kailkhura, and T Yong-Jin Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. In *International Conference on Machine Learning*, pages 11117–11128. PMLR, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] in Appendix A.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] in Appendix A.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] As part of the supplementary material and details are mentioned in Appendix E.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Appendix E.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] see Appendix H.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] in Appendix E.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] in Appendix E.
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]