AdaFocal: Calibration-aware Adaptive Focal Loss

Arindam Ghosh
3M Health Info. Systems
Pittsburgh, PA 15217
aghosh4@mmm.com

Thomas Schaaf
3M Health Info. Systems
Pittsburgh, PA 15217
tschaaf@mmm.com

Matt Gormley
Carnegie Mellon University
Pittsburgh, PA 15213
mgormley@cs.cmu.edu

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 calibration than cross-entropy while achieving similar 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) [19]). However, FLSD-53 is based on heuristics and does not generalize well. 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 γ_{t-1} from the previous step and the knowledge of model's under/over-confidence on the validation set. We evaluate AdaFocal on various image recognition and one NLP task, covering a wide variety of network architectures, to confirm the improvement in calibration while achieving similar levels of accuracy. Additionally, we show that models trained with AdaFocal achieve a significant boost in out-of-distribution detection.

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. Along with these advances it has also been well established that these 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 neural network calibration. These roughly fall into two categories (1) post hoc approaches that perform calibration after training (2) methods that calibrate the model during training. 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, 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.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

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 issue, we propose a modification to the focal loss, while utilizing inverse-focal loss [30, 17], named 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 from focal to inverse focal loss when focal loss fails to overcome the level of under-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. Models calibrated by AdaFocal benefit more from post hoc calibration techniques to further reduce the calibration error. Finally, we study the performance of AdaFocal on an out-of-distribution detection task and find a substantial improvement in performance.

2 Problem Setup and Definitions

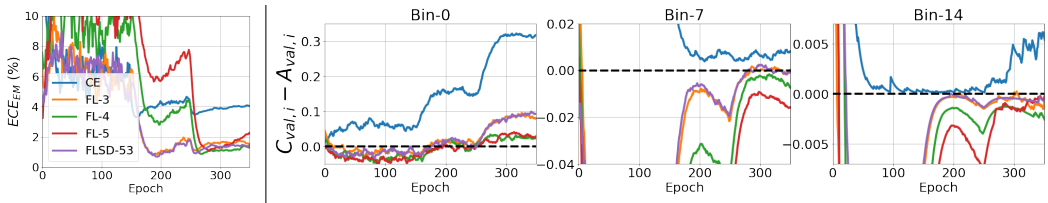
For a classification problem with training data $\{(\mathbf{x}_n, y_{\text{true},n})\}$, where $\mathbf{x}_n$ is the input and $y_{\text{true},n} \in \mathcal{Y} = \{1, 2, \ldots, K\}$ is the ground-truth, we train a model f that outputs a probability vector $\hat{\mathbf{p}}$ over the K classes. We 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 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. Furthermore, 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 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 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, as we show next, focal loss with fixed γ 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). We see that, 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 show that 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 bring further improvement. 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 method across all bins. It is a strategy based on fixed 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 clearly 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 are our actual target but we do not have direct control over them)? 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 some 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 make use of the inverse-focal loss [30, 17] for cases where regular focal loss fails to overcome under-confidence. See, for example, ResNet-50 trained on ImageNet in section 6 and Fig. 5 where even cross entropy ($\gamma = 0$) can not reach 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), thereby pushing their scores 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 respective validation-bin boundaries. Before proceeding further, we first clarify a few notations of interest.

Quantities of interest For binning the 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 $\hat{p}_{train,true}$ that gets manipulated by some loss function

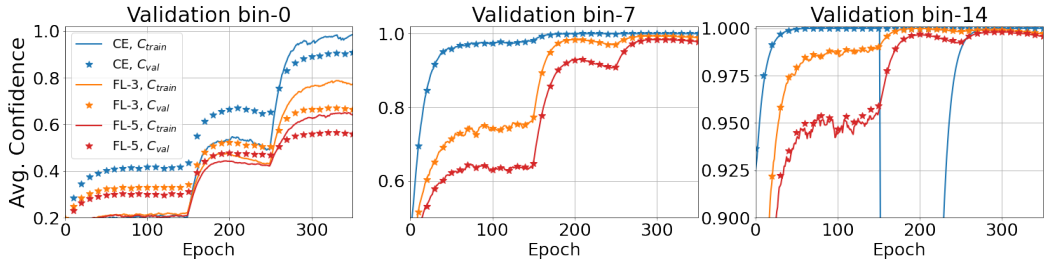


Figure 2: Correspondence between confidence of training (C_{train}) and validation samples (C_{val}) for ResNet-50 trained on CIFAR-10 with focal loss $\gamma = 0$ (CE), 3, 5. The binning involves 15 equal-mass bins where training samples are grouped using validation-bin boundaries. A lower (bin-0),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

(Figure 8 in Appendix C compares $C_{train,true}$ and $C_{train,top}$ to show that as the training accuracy approaches 100%, the top predicted class and the true class become the same). For brevity, we will henceforth 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.

We also 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. Again, we observe a similar correspondence. Note here that, since the binning is independent, the boundaries of training-bin-i will not be exactly the same as that of validation-bin-i, but, as shown in Figure 9, they are very close to each other.

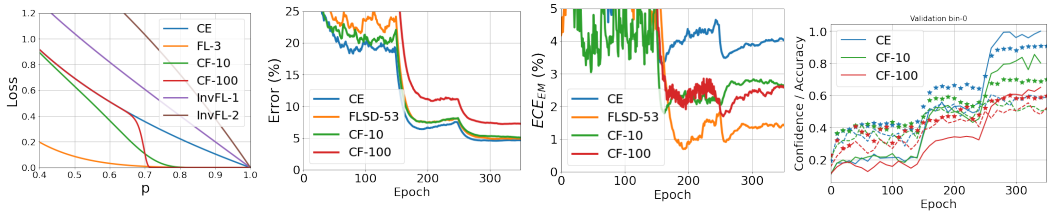
These observations, therefore, are very encouraging as now we have a way to indirectly control $C_{val,i}$ by manipulating $C_{train,i}$, i.e. we can expect (even if loosely) that if we increase/decrease the confidence of a group of training samples in some lower (or middle, or higher) probability region then the same will get reflected on the validation samples in a similar lower (or middle, or higher) probability region. From a calibration point of view, our strategy going forward would be to exploit this correspondence to push $C_{train,i}$ (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 gets 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 and p_n falls into validation-bin b. Our goal then 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].

[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}.

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.

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 P, 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). Therefore, the final AdaFocal loss is given by

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

and the complete algorithm along with the gamma-update rules is given in Algorithm 1. A discussion on the selection of hyper-parameters is presented in the next section.

6 Experiments

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 a global-pooling CNN [15] and fine-tune a pre-trained BERT model [3]. More details about the datasets, models and experimental configurations are given in Appendix D. 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 $\mathrm{ECE}_{\mathrm{EM}}$. In Figure 5, for ResNet-50 on CIFAR-10 and ImageNet, we plot (1) the calibration statistics $E_{val} = C_{val} - A_{val}$ used by AdaFocal during training and (2) the dynamics of γ_t for a few bins in lower, middle, and higher probability regions.

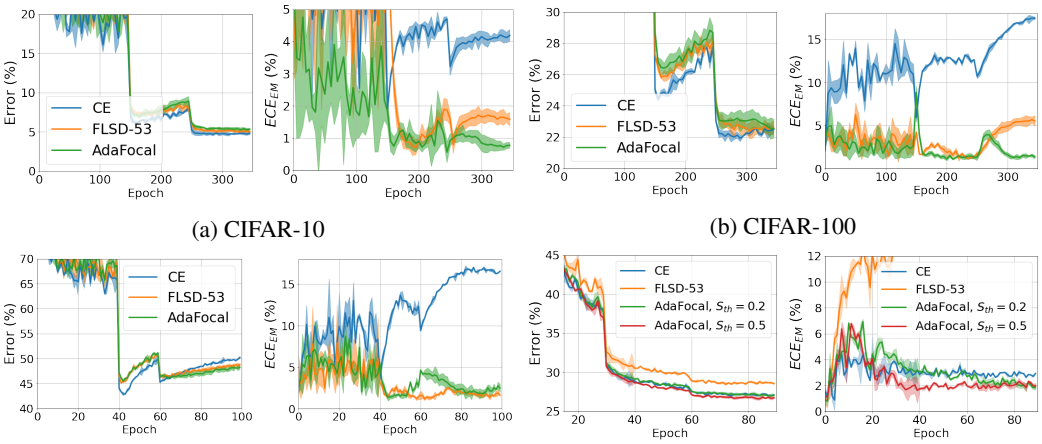
Algorithm 1: AdaFocal

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;
  θ_{t+1} = 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;

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. This, as shown in Figure 5(b), is due to the use of high values of γ $(= 5, 3)$ by FLSD-53 which makes the model largely under-confident in each bin, leading to a overall high calibration error. AdaFocal, on the other hand, maintains a well calibrated model throughout the training for all cases.

Next, for ResNet-50 trained on CIFAR-10, we find γ_t to be closer to 1 for higher bins and closer to 20 for lower bins. These γ_ts 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 $\gamma_t = 1$, first 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 to reach the same level of calibration early. 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 $\mathrm{ECE}_{\mathrm{EM}}$ and test set error on for rest of the experiments are shown in Table 1 and 2 respectively. From Table 1, we observe that prior to temperature scaling, AdaFocal outperforms the other methods in 14 out of 15 cases. Post-temperature scaling, AdaFocal achieves the lowest calibration error in 12 out of the 15 cases. Further, observe that the optimal temperatures are mostly close to 1 indicating that AdaFocal produces inherently calibrated models from training itself. Effects of other post hoc calibration methods (ETS and spline fitting) are shown in Appendix E. Again, we observe that pre-calibrated models from AdaFocal benefit from post processing to further lower the overall calibration error. Besides $\mathrm{ECE}_{\mathrm{EM}}$, the consistency of the results across other calibration



(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**: $\mathrm{ECE}_{\mathrm{EM}}$ (%) 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.

Dataset	Model	Pre Temperature scaling						Post Temperature scaling					
		CE	Brier	MMCE	LS	FLSD-53	AdaFocal	CE	Brier	MMCE	LS	FLSD-53	AdaFocal
CIFAR-10	ResNet50	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)**
	ResNet110	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)**
	WideResNet	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)**
	DenseNet121	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	ResNet50	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)**
	ResNet110	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)**
	WideResNet	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)**
	DenseNet121	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)**
TinyImageNet	ResNet50	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)**
	ResNet110	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	ResNet50	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)
	ResNet110	1.28	3.98	1.83	4.02	18.66	**1.17**	1.28(1.00)	2.87(0.90)	1.83(1.00)	2.76(0.90)	2.51(0.70)	**1.17(1.00)**
	DenseNet121	1.82	2.94	**1.22**	5.30	19.19	1.50	1.82(1.00)	2.21(0.90)	**1.22(1.00)**	1.42(0.90)	2.24(0.70)	1.50(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)**

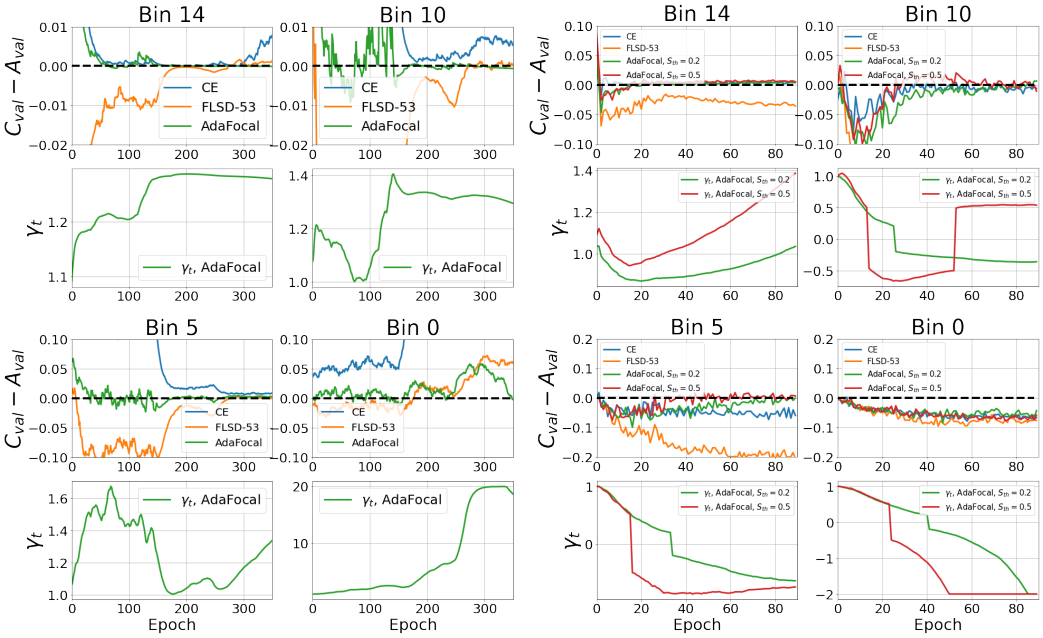
Table 1: Test $\mathrm{ECE}_{\mathrm{EM}}$ (%) averaged over 5 runs. Bold marks the lowest in pre and post temperature scaling groups separately. Optimal temperature, given in brackets, is cross-validated on $\mathrm{ECE}_{\mathrm{EM}}$.

metrics is shown through $\mathrm{ECE}_{\mathrm{DEBIAS}}$, $\mathrm{ECE}_{\mathrm{SWEEP}}$ (equal-width and equal-mass) in Appendix F. Statistical significance of the results is confirmed through $\mathrm{ECE}_{\mathrm{EW}}$ error bars in Appendix G where mean and standard deviations are plotted over 5 runs.

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 K. 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 K. 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].

Hyper-parameters The hyper-parameters introduced by AdaFocal are λ, $\gamma_{\min\,/\,\max}$, and S_{th}. However, these do not require an extensive hyper-parameter search and are much easier to select compared to γ (which, otherwise, needs to be searched for every bin at every time step). Based on our experiments,

- λ is redundant and one may choose to ignore it as for all our experiments $\lambda = 1$ worked very well. However, for an unknown dataset-model pair, if increasing/decreasing the rate of change of γ improves calibration (or helps to reach the desired level faster), then one can use λ to achieve so.



(a) CIFAR-10, ResNet-50

(b) ImageNet, ResNet-50

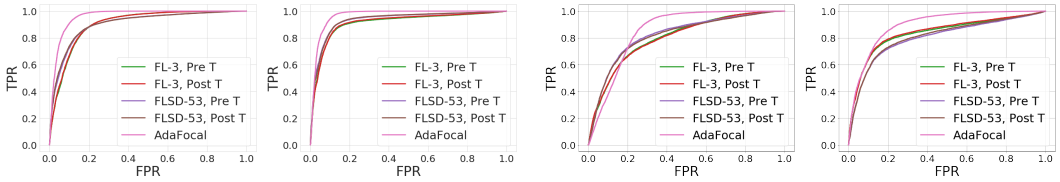
Figure 5: Dynamics of γs in different validation-bins when ResNet-50 is trained on CIFAR-10 and ImageNet with AdaFocal. For each bin, **top**: $C_{val} - A_{val}$ from the validation set, **bottom**: γ_t vs. epochs. Black dotted line in top plot represent zero calibration error. We observe that for each 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 $S_{th} = 0.2$ (or 0.5) for some of the bins.

Dataset	Model	CE	Brier	MMCE	LS	FLSD-53	AdaFocal
CIFAR-10	ResNet50	**4.95**	5.00	4.99	5.29	4.98	5.30
	ResNet110	**4.89**	5.48	5.40	5.52	5.42	5.27
	WideResNet	**3.86**	4.08	3.91	4.20	4.01	4.50
	DenseNet121	**5.00**	5.11	5.41	5.09	5.46	5.20
CIFAR-100	ResNet50	23.30	23.39	23.20	23.43	23.22	**22.60**
	ResNet110	22.73	25.10	23.07	23.43	**22.51**	22.79
	WideResNet	20.70	20.59	20.73	21.19	20.11	**20.07**
	DenseNet121	24.52	23.75	24.0	24.05	22.67	**22.22**
Tiny-ImageNet	ResNet50	**42.90**	46.27	45.96	44.42	45.12	45.49
	ResNet110	42.53	45.47	**42.22**	44.13	44.88	44.55
ImageNet	ResNet50	27.08	28.80	27.12	28.43	28.53	**27.07**
	ResNet110	23.77	24.07	23.72	23.84	25.17	**23.66**
	DenseNet121	27.84	28.02	27.87	27.79	29.12	**27.74**
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.

- $\gamma_{\min\,/\,\max}$ do not require any special fine-tuning as their sole purpose is to stop γ from exploding in either directions. This is similar to the common practice of gradient clipping for stable training. For all our experiments, we use $\gamma_{\max} = 20$, but any reasonable value around that range should also work well in practice. For comparison of results with $\gamma_{\max} = 20$, $\gamma_{\max} = 50$ and $\gamma_{\max} = \infty$, please refer to Appendix N.

- $\gamma_{\min} = -2$ is selected based on the observation that values beyond -2 led to unstable training. However, if, for a new untested dataset-model pair, $\gamma_{\min} = -2$ turns out to be unsuitable, it should



(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.

still be fairly easy to select a new threshold by simply looking at the "dynamics of γ" plots (similar to Fig. 5) at the time step the training becomes unstable.

- We use $S_{th} = 0.2$ for all our experiments. This also does not require extensive tuning and can be easily selected based on the evolution of γ in various bins for a trial run of AdaFocal (without switching to inverse focal loss). For example, for ImageNet (Fig. 5), where some of the bins are always under-confident, AdaFocal decreases γ (starting at $\gamma = 1$) towards negative values. In this case, it makes more sense to have a higher S_{th} $(= 0.5)$ so that AdaFocal can switch early to inverse-focal loss and overcome the under-confidence (see ImageNet results in Fig 4 and 5).
- **Number of bins.** We experimented with AdaFocal using 5, 10, 15, 20, 30, and 50 equal-mass bins during training to draw calibration statistics form the validation set. As reported in Appendix H, the best results are for number of bins in the range of 10 to 20. Performance degrades when the number of bins are too small (< 10) or too large (> 20). Therefore, we use 15 bins for all AdaFocal trainings. Note that for computing ECE metrics as well, we use 15 bins so as to be consistent with previous works in literature [19, 4].

Best choice of AdaFocal + post-hoc calibration From the results in Table 1, 3, and 4 for temperature scaling, ETS and Spline fitting, respectively, there is not a clear choice of post-hoc calibration method that gives the best results across all dataset-model pairs when combined with AdaFocal. However, we do observe that in almost all cases it is the pre-calibrated model by AdaFocal that gives the lowest ECE when combined with one of these post-hoc calibration techniques. Therefore, for an unknown dataset-model pair, the choice of the best post-hoc calibration method (to be used on top of AdaFocal) might require more investigation and is a different study in itself. Overall, the evidence in our paper show that pre-calibration from training leads to even better calibrated models post hoc, and AdaFocal is much better in producing such pre-calibrated models.

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 from the validation set. 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 inherently calibrated models that benefit further from post-hoc calibration techniques. Additionally, we find that models trained with AdaFocal are 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 Section 6.
   (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 D.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] see Appendix D.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] see Appendix G.
   (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 D.
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 D.
   (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]

Appendix

A Broader Impact

Overconfidence in deep neural networks could easily lead to deployments where predictions are made that should have been withheld. For example, in medical diagnosis applications the Bayes optimal decision depends heavily on the model accurately modeling the distribution over its output classes. We hope that our work is a small step towards avoiding high loss errors in decision making applications. The metrics used in this paper assume that improving average calibration is the goal; but other metrics should be considered if we want to, for example, ensure good average calibration across different strata (e.g. if instances correspond to users of different social strata).

B Calibration Behaviour of Focal Loss in Different Bins

In the main paper, we have showed the calibration behavior of different focal losses for ResNet50 trained on CIFAR-10 for only a few bins. For completeness, the rest of the bins and their calibration error $E_i = C_{val,i} - A_{val,i}$ are shown in Figure 7 for focal losses with $\gamma = 0, 3, 4, 5$. We observe that there's no single γ that performs the best across all the bins. Rather, every bin has a particular γ that achieves the best calibration.

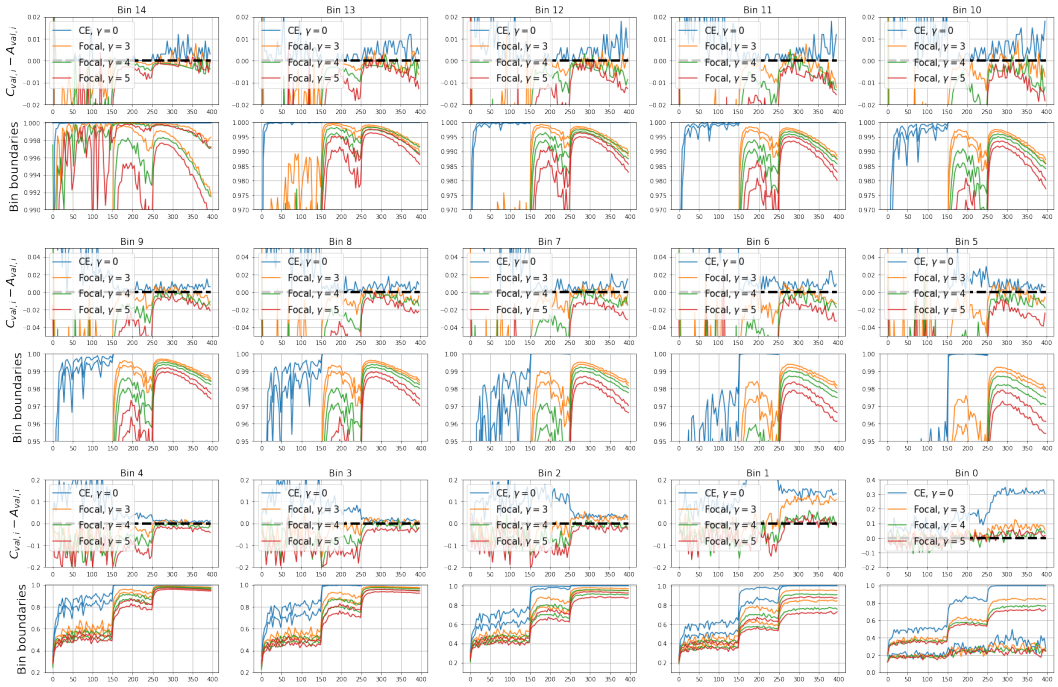


Figure 7: ResNet-50 trained on CIFAR-10 using focal loss $\gamma = 0, 3, 4, 5$. **Top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **Bottom**: bin boundaries. The statistics are computed on the validations set using 15 equal-mass bins. The black horizontal dashed line in every top subfigure represents zero calibration error $E_{val,i} = 0$.

C Correspondence between Confidence of Training and Validation Samples

C.1 Closeness of $C_{train,true}$ and $C_{train,top}$ as Training Progresses

For a training sample, the confidence of the true class y_{true} is denoted by $\hat{p}_{train,true}$ and the average equivalent in a bin by $C_{train,true}$. Similarly, the confidence of the top predicted class $\hat{y}$ (for the training sample) is denoted by $\hat{p}_{train,top}$ and the average equivalent in a bin by $C_{train,top}$. For the training set, we care only about the confidence of the "true class" $\hat{p}_{train,true}$ as that is the quantity which gets manipulated by some loss function. For validation set, on the other hand, we care about the confidence of the "top predicted class". Therefore, it would be more natural to look for correspondence between similar quantities, particularly $C_{train,top,i}$, across the two datasets. However, as we shown in Fig. 8, $C_{train,true,i}$ and $C_{train,top,i}$ are almost the same during major part of the training. This is because as the model approaches towards 100% accuracy on the training set, the top predicted class and the true class for a training sample become the same. Therefore we can directly compare the two different quantities $C_{train,true,i}$ and $C_{train,top,i}$ across the training and validation set.

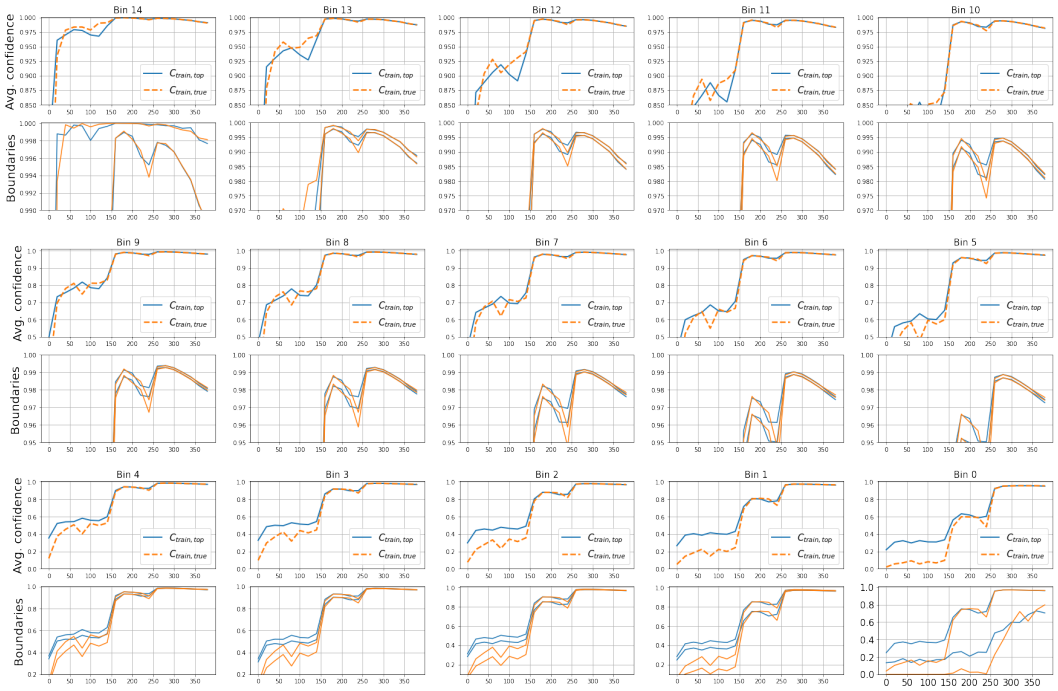


Figure 8: ResNet50 trained on CIFAR-10 with focal loss $\gamma = 3$. The figure shows the closeness of $C_{train,true,i}$ (**orange**) and $C_{train,top,i}$ (**blue**) for training samples as training progresses towards 100% accuracy on the training set.

C.2 CIFAR-10, ResNet-50: Correspondence between C_{train} and C_{val}

Fig. 9 and 10 show the correspondence between the confidence of training samples $C_{train} \equiv C_{train,true}$ and the confidence of the validation samples $C_{val} \equiv C_{val,top}$ for the dataset-model pair CIFAR-10, ResNet-50, under following two cases:

- **Independent binning**: when training samples and validation samples are grouped independently into their respective training-bins and validation-bins (Fig. 9).
- **Common binning**: when training samples are grouped using the common bin boundaries of the validation-bins that were formed by binning the validation bins (Fig. 10).

C.2.1 Independent binning

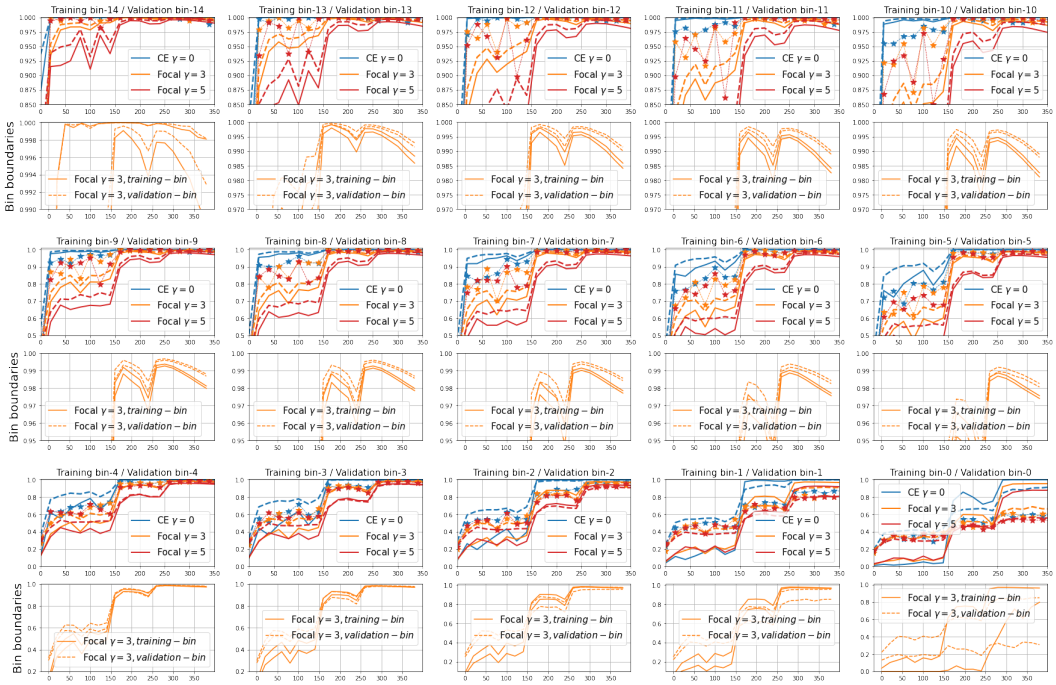


Figure 9: **Independent binning**: training samples and validation samples are grouped independently into training-bin and validation-bin respectively. The top subfigure for each bin shows the correspondence between average confidence of a group of training samples $C_{train,true,i}$ and a group of validation samples $C_{val,top,i}$ when ResNet-50 is trained on CIFAR-10 with focal loss $\gamma = 0, 3, 5$. The binning is adaptive with 15 equal-mass bins. **Solid line**: $C_{train,true,i}$ in training-bin i, **Dashed line**: $C_{val,top,i}$ and **Star-dashed line**: $A_{val,i}$ in validation-bin i. The bottom subfigure shows the bin boundaries for focal loss $\gamma = 3$ as an example.

C.2.2 Common binning

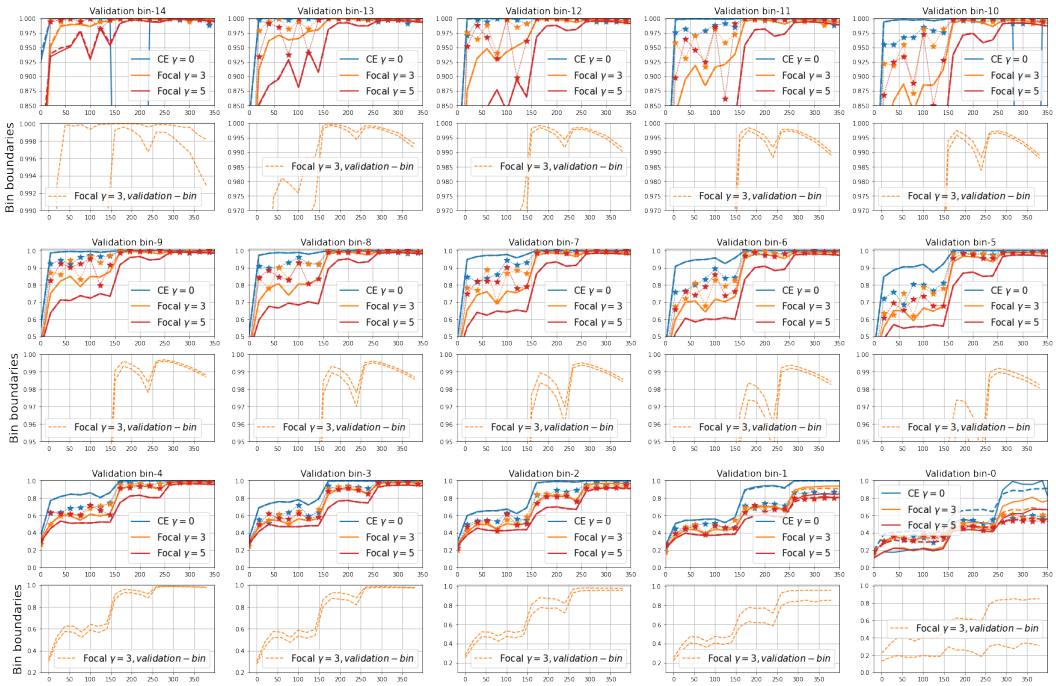


Figure 10: **Common binning**: training samples are grouped using the bin boundaries of the validation-bins. The top subfigure for each bin shows the correspondence between average confidence of a group of training samples $C_{train,true,i}$ and a group of validation samples $C_{val,top,i}$ when ResNet-50 is trained on CIFAR-10 with focal loss $\gamma = 0, 3, 5$. The binning is adaptive with 15 equal-mass bins. **Solid line**: $C_{train,true,i}$ in validation-bin i, **Dashed line**: $C_{val,top,i}$ and **Star-dashed line**: $A_{val,i}$ in validation-bin i. The bottom subfigure shows the bin bin boundaries for focal loss $\gamma = 3$ as an example.

C.3 CIFAR-100, ResNet-50: Correspondence between C_{train} and C_{val}

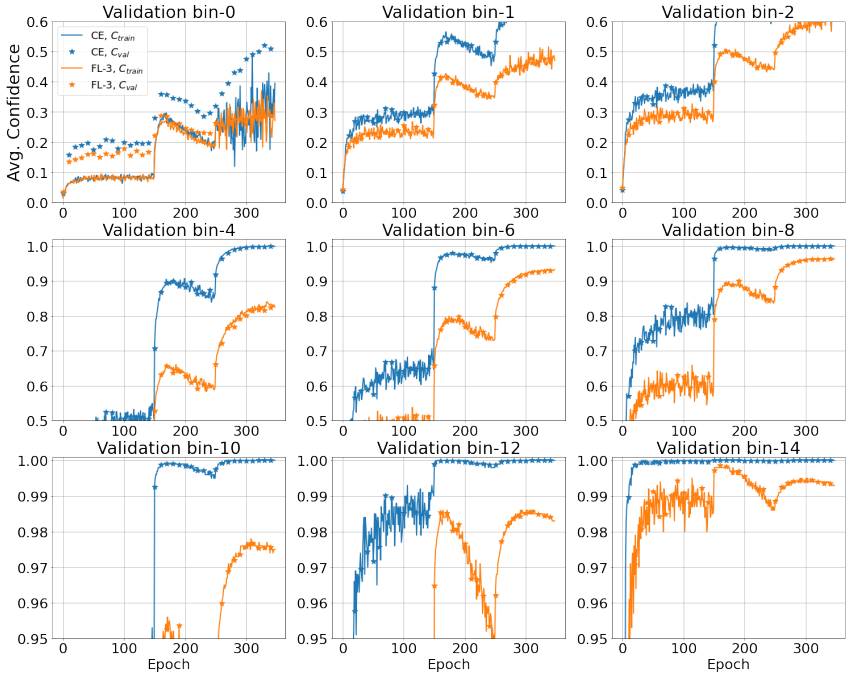


Figure 11: **Common binning**: C_{train} (solid) and C_{val} (star) both binned using validation-bin boundaries. Show here for focal loss $\gamma = 0$ (CE) and $\gamma = 3$ (FL-3).

C.4 CIFAR-100, WideResNet: Correspondence between C_{train} and C_{val}

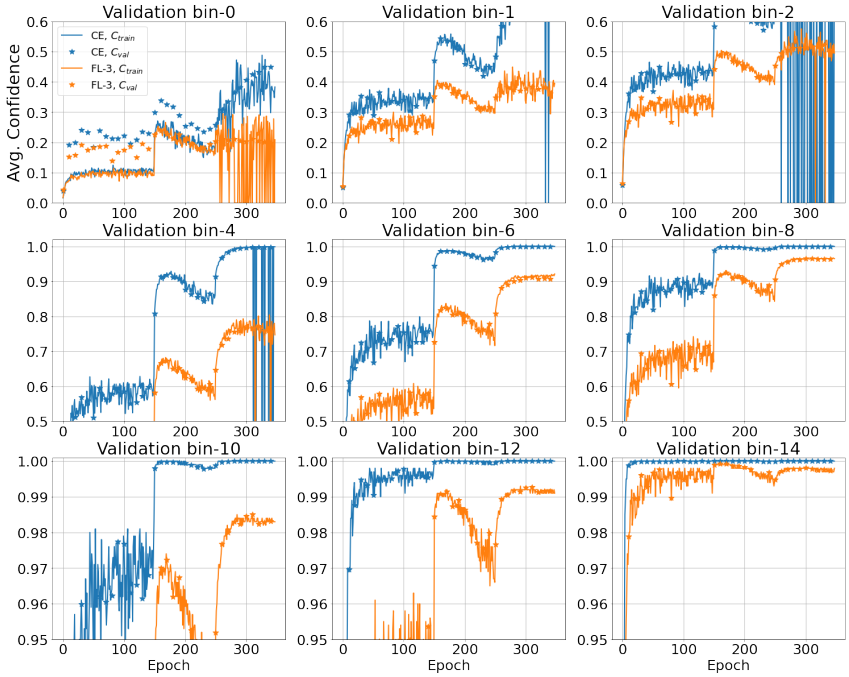


Figure 12: **Common binning**: C_{train} (solid) and C_{val} (star) both binned using validation-bin boundaries. Show here for focal loss $\gamma = 0$ (CE) and $\gamma = 3$ (FL-3).

C.5 TinyImageNet, ResNet-50: Correspondence between C_{train} and C_{val}

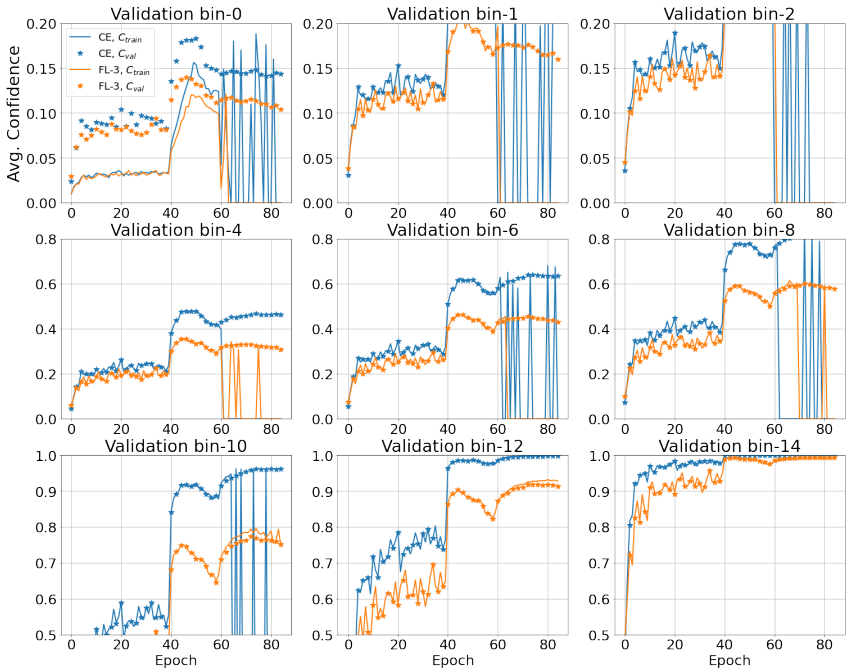


Figure 13: **Common binning**: C_{train} (solid) and C_{val} (star) both binned using validation-bin boundaries. Show here for focal loss $\gamma = 0$ (CE) and $\gamma = 3$ (FL-3).

C.6 20 Newsgroups, CNN: Correspondence between C_{train} and C_{val}

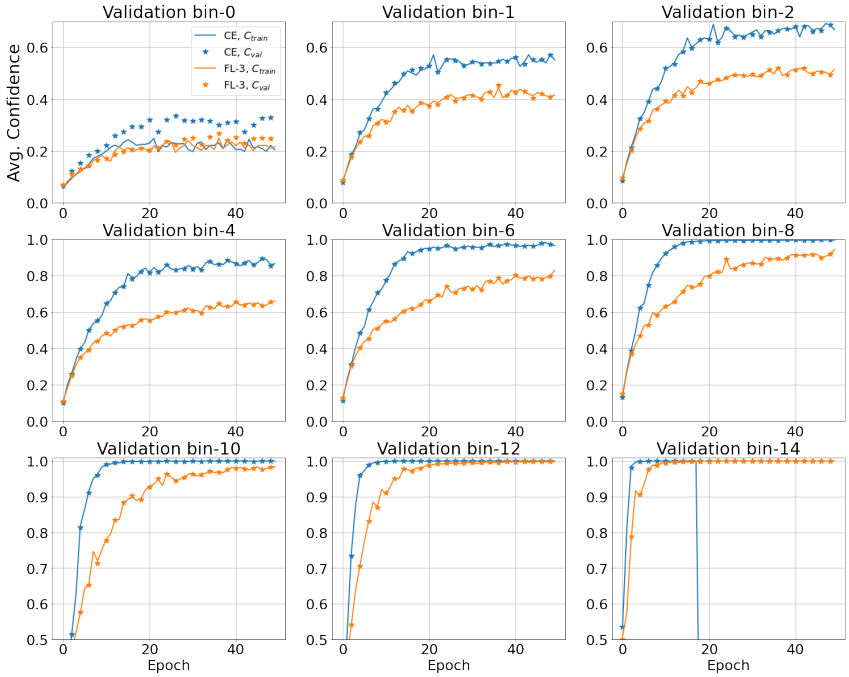


Figure 14: **Common binning**: C_{train} (solid) and C_{val} (star) both binned using validation-bin boundaries. Show here for focal loss $\gamma = 0$ (CE) and $\gamma = 3$ (FL-3).

D Datasets and Experiments

D.1 Dataset Description

CIFAR-10 [9]: This dataset contains $60,000$ coloured images of size 32×32, which are equally divided into 10 classes. A split of $45,000/5,000/10,000$ images is used as train/validation/test sets respectively.

CIFAR-100 [9]: This dataset contains $60,000$ coloured images of size 32×32, which are equally divided into 100 classes. A split of $45,000/5,000/10,000$ images is used as train/validation/test sets respectively.

ImageNet [27]: ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012-2017 is an image classification and localization dataset. This dataset spans 1000 object classes and contains 1,281,167 training images and 50,000 validation images.

Tiny-ImageNet [2]: It is a subset of the ImageNet dataset with 64×64 dimensional images and 200 classes. It has 500 images per class in the training set and 50 images per class in the validation set.

20 Newsgroups [14]: This dataset contains $20,000$ news articles, categorised evenly into 20 different newsgroups. Some of the newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian). We use a train/validation/test split of $15,098/900/3,999$ documents.

D.2 Experiment Configurations

For our experiments, we have used Nvidia Titan X Pascal GPU with 12 GB of memory. Training configuration for each dataset is given below.

CIFAR-10 and **CIFAR-100**: We use SGD with a momentum of 0.9 as our optimiser, and train the networks for 350 epochs, with a learning rate of 0.1 for the first 150 epochs, 0.01 for the next 100 epochs, and 0.001 for the last 100 epochs. We use a training batch size of 128. The training data is augmented by applying random crops and random horizontal flips.

Tiny-ImageNet: We use SGD with a momentum of 0.9 as our optimiser, and train the models for 100 epochs with a learning rate of 0.1 for the first 40 epochs, 0.01 for the next 20 epochs and 0.001 for the last 40 epochs. We use a training batch size of 64. Note that we use 50 samples per class (i.e. a total of 10000 samples) from the training set as the validation set. Hence, the training is only on 90000 images. We use the Tiny-ImageNet validation set as our test set.

ImageNet: We use SGD as our optimiser with momentum of 0.9 and weight decay 10^{-4}, and train the models for 90 epochs with a learning rate of 0.01 for the first 30 epochs, 0.001 for the next 30 epochs and 0.0001 for the last 30 epochs. We use a training batch size of 128. We divide the 50,000 validation images into validation and test set of 25,000 images each.

20 Newsgroups, CNN: We train the Global Pooling CNN Network [15] using the Adam optimiser, with learning rate 0.001, and default betas 0.9 and 0.999. We used Glove word embeddings [25] to train the network. We train the model for 50 epochs and use the model at the end to evaluate the performance.

20 Newsgroups, BERT: We fine-tune a BERT model by adding a single linear classification layer on top of pre-trained "bert-base-uncased" model (12-layer, 768-hidden, 12-heads, 110M parameters) [3], using the AdamW optimiser (Adam with weight decay), with batch size of 32, learning rate $2e-5$, weight decay of 0.01 and warm up steps of 0.2 the number of batches in the training set. We limit the length of the input sequence to 128 for training and 512 for testing. We train the model for 10 epochs and select the model that has the lowest error on validation set.

The experiments are implemented using PyTorch library. The hyperparameters that are not explicitly mentioned above are set to their default values in PyTorch. For CIFAR-10/100 and Tiny-ImageNet, AdaFocal is implemented on top of the base code available at [18]. The code for 20 Newsgroups is implemented in PyTorch by adapting the TensorFlow code available at [12].

D.3 Model Selection

For all experiments, except Tiny-ImageNet, we select the model at the end of the training mainly to be consistent with [19] i.e. the work we are trying to improve upon in this paper. As confirmed by the authors of [19], they use the model at the end of the training to report results in the paper. Therefore, for the following datasets, the error and ECE results are reported for the model at

- CIFAR-10: 350 epochs
- CIFAR-100: 350 epochs
- ImageNet: 90 epochs
- 20 NewsGroups, CNN: 50 epochs

For Tiny-ImageNet and BERT, we have reported the model that has the lowest error on the validation set.

E Other Post Hoc Calibration Techniques

E.1 Ensemble Temperature Scaling (ETS)

Dataset	Model	Cross Entropy	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	2.97	1.71	**0.55**
	ResNet-110	3.18	1.79	**0.57**
	Wide-ResNet-26-10	2.55	2.00	**0.49**
	DenseNet-121	3.40	1.64	**0.57**
CIFAR-100	ResNet-50	3.38	2.46	**1.33**
	ResNet-110	4.60	3.87	**1.24**
	Wide-ResNet-26-10	2.91	2.07	**1.79**
	DenseNet-121	4.48	**1.21**	1.86
Tiny-ImageNet	ResNet-50	3.02	1.46	**1.23**
	ResNet-110	1.26	1.22	**0.62**
ImageNet	ResNet-50	**0.90**	2.13	1.13
	ResNet-110	1.38	2.25	**1.28**
	DenseNet-121	**1.07**	2.36	1.40
20 Newsgroups	CNN	2.46	2.50	**2.29**
	BERT	5.34	**3.91**	4.30

Table 3: $\mathrm{ECE}_{\mathrm{EW}}$ (%) after post hoc calibration with Ensemble Temperature Scaling.

E.2 Spline Fitting

Dataset	Model	Cross Entropy	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	1.69	**0.60**	0.65
	ResNet-110	1.88	0.61	**0.58**
	Wide-ResNet-26-10	1.17	0.65	**0.45**
	DenseNet-121	1.48	0.97	**0.53**
CIFAR-100	ResNet-50	2.56	1.07	**1.01**
	ResNet-110	3.36	1.33	**1.29**
	Wide-ResNet-26-10	2.20	**1.08**	1.53
	DenseNet-121	2.83	**1.03**	1.36
Tiny-ImageNet	ResNet-50	1.44	1.91	**1.39**
ImageNet	ResNet-50	0.82	0.87	**0.66**
	ResNet-110	**0.60**	0.69	0.62
	DenseNet-121	0.72	**0.66**	0.75
20 Newsgroups	Global-pool CNN	1.97	1.38	**1.12**

Table 4: $\mathrm{ECE}_{\mathrm{EW}}$ (%) after post hoc calibration with spline fitting.

F Debiased Estimates of ECE: $\mathrm{ECE}_{\mathrm{DEBIAS}}$ and $\mathrm{ECE}_{\mathrm{SWEEP}}$

As shown in [26], binning-based estimators $\mathrm{ECE}_{\mathrm{EW}}$ and $\mathrm{ECE}_{\mathrm{EM}}$ may suffer from statistical bias ($\mathrm{ECE}_{\mathrm{EM}}$ has lower bias than $\mathrm{ECE}_{\mathrm{EW}}$) and if the bias is strong enough it may lead to mis-estimation

of calibration error and a wrong model selection. Therefore, to confirm that the results presented in the paper using ECE_{EM} and ECE_{EW} are consistent and reliable, we additionally present here $\text{ECE}_{\text{DEBIAS}}$ [11] and $\text{ECE}_{\text{SWEEP}}$ [26] (equal-mass) as debiased estimates of ECE.

Dataset	Model	Pre Temperature scaling			Post Temperature scaling		
		CE	FLSD-53	AdaFocal	CE	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	4.05	1.62	**0.47**	1.70(2.5)	1.62(1.0)	**0.82(0.9)**
	ResNet-110	4.38	1.82	**0.32**	2.20(2.7)	1.30(1.1)	**0.32(1.0)**
	Wide-ResNet-26-10	3.52	2.01	**0.59**	1.89(2.2)	1.50(0.9)	**0.25(1.1)**
	DenseNet-121	4.26	1.56	**0.42**	2.15(2.3)	1.93(0.9)	**0.42(1.0)**
CIFAR-100	ResNet-50	17.73	5.52	**1.46**	3.86(2.2)	2.92(1.1)	**1.46(1.0)**
	ResNet-110	19.44	7.31	**1.35**	6.01(2.3)	3.55(1.2)	**1.35(1.0)**
	Wide-ResNet-26-10	14.91	2.53	**2.12**	3.32(2.1)	2.53(1.0)	**2.12(1.0)**
	DenseNet-121	19.82	2.29	**1.27**	3.44(2.3)	2.12(1.1)	**1.27(1.0)**
Tiny-ImageNet	ResNet-50	7.95	2.90	**2.69**	3.86(1.44)	2.61(1.06)	**2.31(0.96)**
	ResNet-110	8.09	1.65	**1.50**	1.23(1.20)	1.65(1.00)	**1.50(1.0)**
ImageNet	ResNet-50	2.89	16.76	**1.74**	**1.42(0.90)**	2.58(0.70)	1.74(1.00)
	ResNet-110	1.14	18.65	**1.04**	1.14(1.00)	2.41(0.70)	**1.04(1.00)**
	DenseNet-121	1.74	19.18	**1.30**	1.74(1.00)	2.17(0.70)	**1.30(1.00)**
20 Newsgroups	Global-pool CNN	18.36	8.94	**1.84**	5.23(4.1)	**0.94(1.6)**	1.84(1.0)

Table 5: Test set $\text{ECE}_{\text{DEBIAS}}(\%)$ 15 bins. Optimal temperature, shown in brackets, are selected based on the lowest ECE_{EW} on the validation set.

Dataset	Model	Pre Temperature scaling			Post Temperature scaling		
		CE	FLSD-53	AdaFocal	CE	FLSD-53	AdaFocal
CIFAR-10	ResNet-50	4.05	1.54	**0.04**	1.43(2.5)	1.54(1.0)	**0.70(0.9)**
	ResNet-110	4.38	1.83	**0.40**	1.34(2.7)	1.32(1.1)	**0.40(1.0)**
	Wide-ResNet-26-10	3.53	1.64	**0.38**	1.41(2.2)	1.55(0.9)	**0.32(1.1)**
	DenseNet-121	4.27	1.58	**0.34**	2.17(2.3)	1.98(0.9)	**0.34(1.0)**
CIFAR-100	ResNet-50	17.72	5.51	**1.89**	**0.51(2.2)**	2.36(1.1)	1.89(1.0)
	ResNet-110	19.44	7.34	**1.58**	3.71(2.3)	3.65(1.2)	**1.58(1.0)**
	Wide-ResNet-26-10	14.92	2.62	**2.25**	2.62(2.1)	2.62(1.0)	**2.25(1.0)**
	DenseNet-121	19.82	2.25	**1.47**	3.12(2.3)	2.31(1.1)	**1.47(1.0)**
Tiny-ImageNet	ResNet-50	7.98	2.99	**2.78**	3.96(1.44)	2.83(1.06)	**2.56(0.96)**
	ResNet-110	8.11	2.01	**1.97**	1.81(1.20)	2.01(1.00)	**1.97(1.00)**
ImageNet	ResNet-50	2.93	16.77	**1.98**	**1.63(0.90)**	2.58(0.70)	1.98(1.00)
	ResNet-110	1.15	18.66	**1.08**	1.15(1.00)	2.51(0.70)	**1.08(1.00)**
	DenseNet-121	1.80	19.19	**1.40**	1.80(1.00)	2.29(0.70)	**1.40(1.00)**
20 Newsgroups	Global-pool CNN	18.38	8.95	**2.22**	5.53(4.1)	**2.13(1.6)**	2.22(1.0)

Table 6: Test set $\text{ECE}_{\text{SWEEP}}(\%)$ equal-mass. Optimal temperature, shown in brackets, are selected based on the lowest ECE_{EW} on the validation set.

G ECE_{EW} error bars

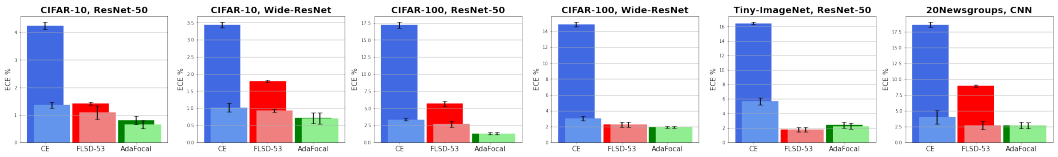


Figure 15: Test set ECE_{EW} (%) error bars with mean and standard deviation computed over 5 runs with different initialization seed. Dark and light shades of a color show pre and post temperature scaling results respectively. Optimal temperatures are cross-validated based on ECE_{EW}.

H Number of bins used for AdaFocal training

Experiment details:

1. ResNet-50 trained on CIFAR-10 for 350 epochs.
2. The reported results below are without temperature scaling.

3. We compare AdaFocal with 5, 10, 15, 20, 30, and 50 equal mass bins vs FLSD-53.

Note that there are two types of binning involved:

- **For training**: the binning that is performed on the validation set from where AdaFocal draws calibration related information to adjust γ. These correspond to the columns in the table 7.
- **For evaluation**: once we have a trained model, the binning that is used to compute the ECE metric. These correspond to the rows in the table below.

		Number of bins used for AdaFocal training					
Evaluation Metric	**FLSD-53**	5 bins	10 bins	15 bins	20 bins	30 bins	50 bins
ECE_{EW} (15bins)	1.35	0.76	0.53	0.51	0.60	0.82	1.16
ECE_{EM} (15bins)	1.67	0.63	0.53	0.56	0.40	0.84	1.10
$\text{ECE}_{\text{DEBIAS}}$ (15bins)	1.62	0.50	0.44	0.47	0.25	0.79	1.07
$\text{ECE}_{\text{DEBIAS}}$ (30bins)	1.57	0.73	0.43	0.46	0.27	0.72	1.06
$\text{ECE}_{\text{SWEEP-EW}}$	1.31	0.66	0.43	0.48	0.48	0.80	1.08
$\text{ECE}_{\text{SWEEP-EM}}$	1.54	0.53	0.21	0.04	0.38	0.07	1.08

Table 7: ECE (%) performance for ResNet-50 trained on CIFAR-10 when AdaFocal training uses different number of equal-mass bins. We observe that the best results are for number of bins in the range of 10 to 20. Performance degrades when the number of bins are too less (< 10) or too many (≥ 30).

I Frequency of γ-update

In this section, we study how the frequency of the γ-update affect the performance of AdaFocal on the 20Newsgroup dataset (with CNN and BERT models) i.e if the validation-bin boundaries and γ are updated every mini-batch or a few times per training epoch.

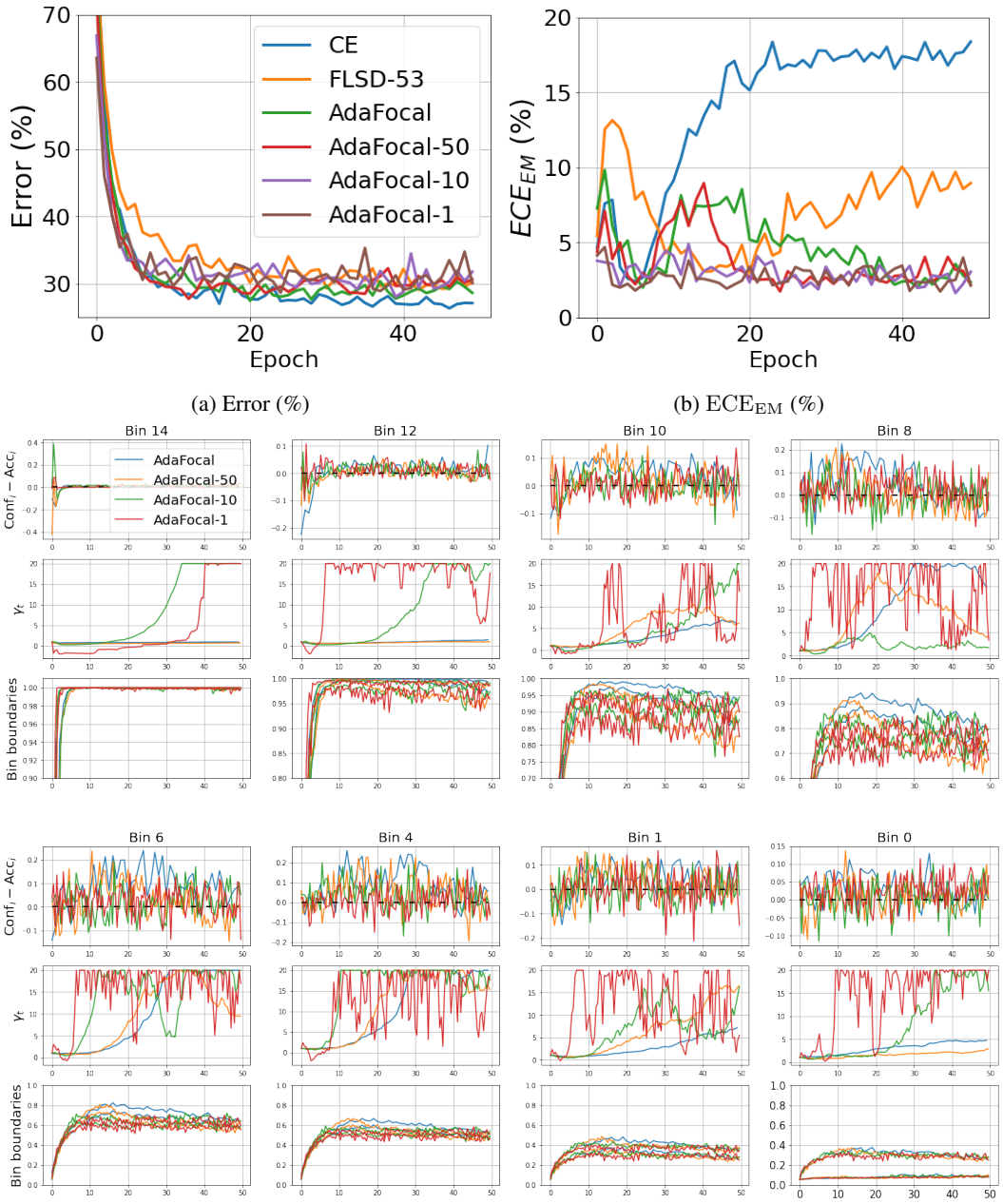
Intuitively, one would expect that if the validation-bin boundaries are updated more frequently then AdaFocal would be able to more closely track the changes in the calibration behaviour of the validation set and accordingly adjust it's γs to better respond to the changes. This is supported by the experiments on 20 Newsgroup dataset using CNN and BERT model as shown in Fig. 16 and 17. In these figures, AdaFocal on its own means that γ is updated at the end of every epoch. AdaFocal-n, where $n = 100, 50, 10, 1$, means that γ is updated every n mini-batches. Since the number of training batches for 20Newsgroup with CNN is 118, AdaFocal-50(respectively 10, 1) means γ is updated 2 (respectively 11, 118) times per epoch. Similarly, for 20Newsgroup and BERT, as the the number of training batches is 472, AdaFocal-100(respectively 10, 1) means γ is updated 4 (respectively 47, 472) times per epoch.

From these experiments, we observe that

1. **Frequent updates keep the model better calibrated at all time steps and prevent it from getting mis-calibrated**. For example in Fig. 16(b), AdaFocal and AdaFocal-50 are miscalibrated at the start of the training and at around 15-20 epoch, whereas AdaFocal-10 and AdaFocal-1 remains very well calibrated at all epochs. We observe the same in Fig. 17(b), where AdaFocal and AdaFocal-100 are miscalibrated at the start of the training and around epoch $6 - 9$, whereas AdaFocal-10 and AdaFocal-1 remains well calibrated at all times.
2. **For cases where updating γ once per epoch leads to only a few γ updates in total, frequent updates may lead to improved ECE performance**. We observe this in Fig. 17(b) where AdaFocal (with only 10 updates as BERT is trained for only 10 epochs) is unable to reach the calibration level of AdaFocal-10 and AdaFocal-1. For 20Newsgroup and CNN, as the model is trained for 50 epochs, AdaFocal, first unable to keep up with AdaFocal-10 and AdaFocal-1, is able to ultimately reach their level of calibration with enough updates.

Drawback: Increased training time. Updating γ more frequently comes at the cost of increased training time. For example, as discussed in J, standard training of ResNet-50 on CIFAR-10 takes

79.1s per epoch and γ-update step takes 2.8s. When trained for 350 epochs, updating γ every epoch adds 16 minutes to the the total training time of 7.7h. Therefore, increasing the update frequency will linearly increase the additional overhead as well.

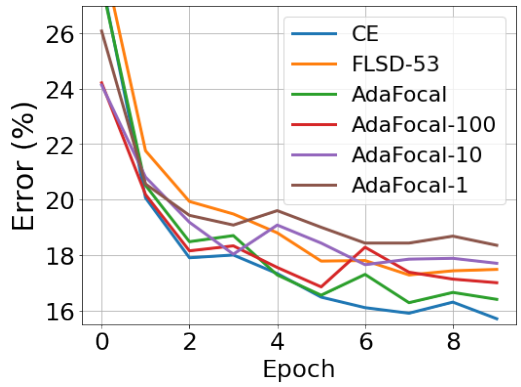


(a) Error (%)

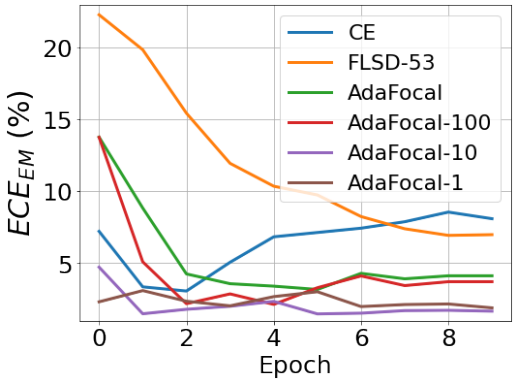
(b) ECE_{EM} (%)

(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

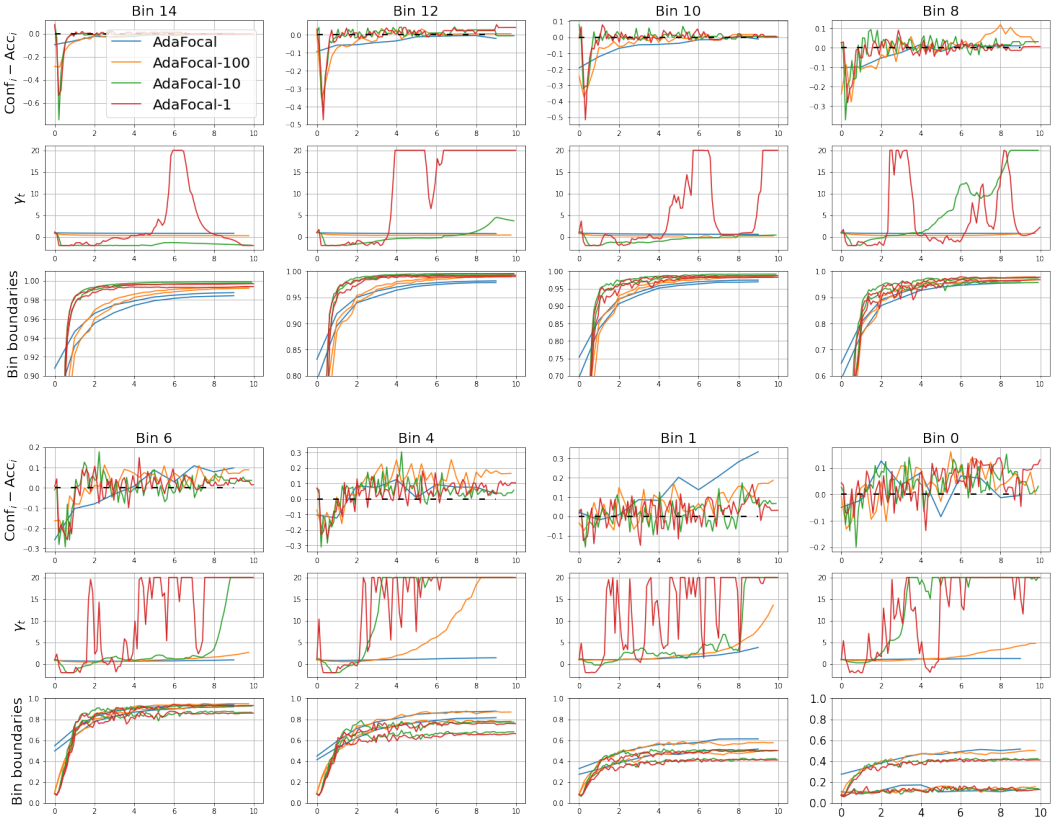
Figure 16: CNN trained on 20Newsgroups with cross entropy (CE), FLSD-53, and AdaFocal. AdaFocal on its own means that γ is updated at the end of every epoch. AdaFocal-n, where $n = 50, 10, 1$, means that γ is updated after every n mini-batches. Since the number of training batches for 20Newsgroup with CNN is 118, AdaFocal-50 (respectively 10, 1) means γ is updated 2 (respectively 11, 118) times per epoch



(a) Error (%)



(b) ECE_{EM} (%)



(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 17: Pre-trained BERT fine-tuned on 20 Newsgroups with cross entropy (CE), FLSD-53, and AdaFocal. AdaFocal on its own means that γ is updated at the end of every epoch. AdaFocal-n, where $n = 100, 10, 1$, means that γ is updated after every n mini-batches. Since the number of training batches for 20Newsgroup with BERT is 472, AdaFocal-100 (respectively 10, 1) means γ is updated 4 (respectively 47, 472) times per epoch.

J Computation overhead for AdaFocal

To update γ of AdaFocal, the extra operations that are required are

1. Forward pass on the validation set to compute the logits/softmaxes.

2. Compute bin statistics and update γ.

In general, if we update γ at the end of every epoch, then compared to the time it takes to train the model for one whole epoch, these two overheads are quite negligible. For example, for ResNet-50 trained on CIFAR-10 (train set contains 45000 examples, val set contains 5000 examples) using Nvidia Titan X Pascal GPU with 12GB memory,

- Training for one epoch = $79,123$ ms = 79.1 s
- Forward pass on validation set = $2,886$ ms = 2.8 s
- Compute bin statistics and update γ = 8 ms

So if the standard training with cross entropy, without any involvement of a validation set, for 350 epochs requires in total $79.1 \times \frac{350}{3600} = 7.7$ hours, then AdaFocal will add $2.808 \times \frac{350}{60} = 16$ minutes on top of the entire training. Naturally, if we update γ more often during the epoch then this overhead will increase and may become significant. However, for all our experiments we update γ at the end of an epoch and that works quite well. Nonetheless, for a comparison of performance of AdaFocal when the update frequency of γ is varied, please refer to Appendix I.

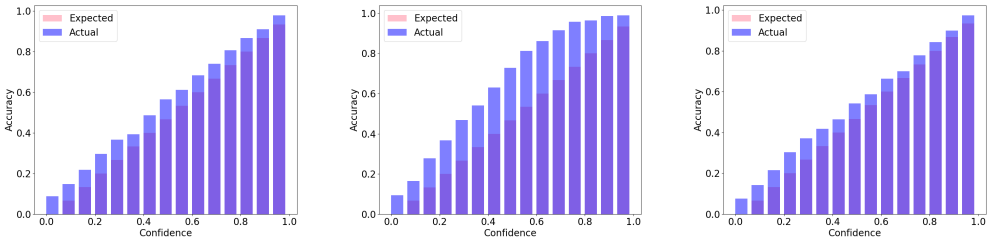
K AUROC for Out-of-Distribution Detection

For ResNet110 trained on in-distribution CIFAR-10 and tested on out-of-distribution SVHN, we were not able to reproduce the reported results of 96.74, 96.92 for focal loss $\gamma = 3$ (FL-3) as given in [19]. Instead we found those values to be 90.27, 90.39 and report the same in Table 8 below.

Dataset	Model	Cross Entropy		Brier Loss		MMCE		LS-0.05		FL-3		FLSD-53		AdaFocal
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T
CIFAR-10 / SVHN	ResNet-110	61.71	59.66	94.80	95.13	85.31	85.39	68.68	68.68	90.27	90.39	90.33	90.49	**96.09**
	Wide-ResNet-26-10	96.82	97.62	94.51	94.51	97.35	97.95	84.63	84.66	90.92	91.30	93.08	93.11	**96.63**
CIFAR-10 / CIFAR-10-C	ResNet-110	77.53	75.16	84.09	83.86	71.96	70.02	72.17	72.18	80.11	79.78	82.06	81.38	**84.96**
	Wide-ResNet-26-10	81.06	80.68	85.03	85.03	82.17	81.72	71.10	71.16	83.33	84.00	80.00	80.76	**89.52**

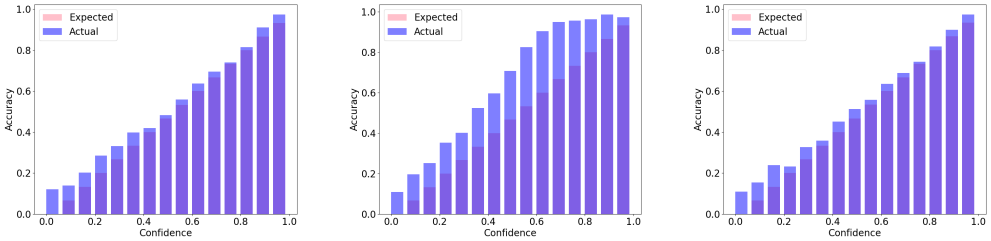
Table 8: AUROC (%) of models trained on CIFAR-10 as the in-distribution data and tested on SVHN and CIFAR-10-C as out-of-distribution data. Temperature scaling is based on $\mathrm{ECE}_{\mathrm{EW}}$.

L Reliability Diagrams



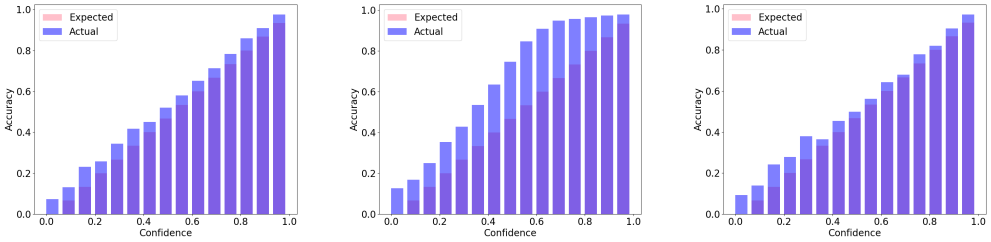
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.93$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 16.77$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.87$

Figure 18: ImageNet, ResNet-50.



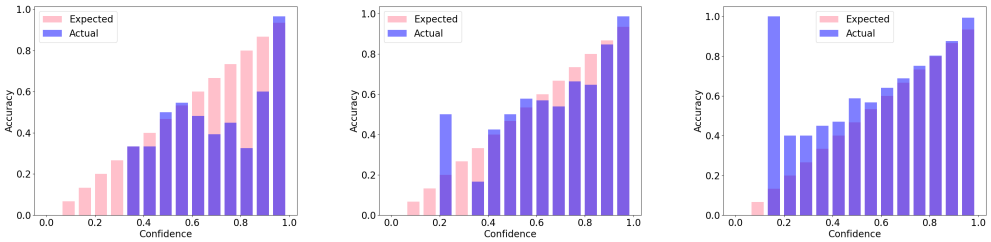
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.28$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 18.66$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.17$

Figure 19: ImageNet, ResNet-110.



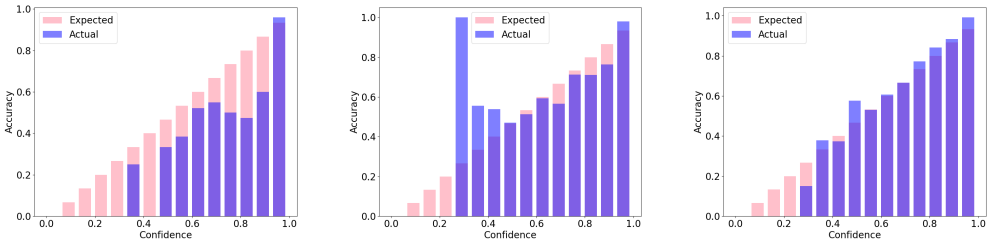
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.82$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 19.19$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.50$

Figure 20: ImageNet, DenseNet-121.



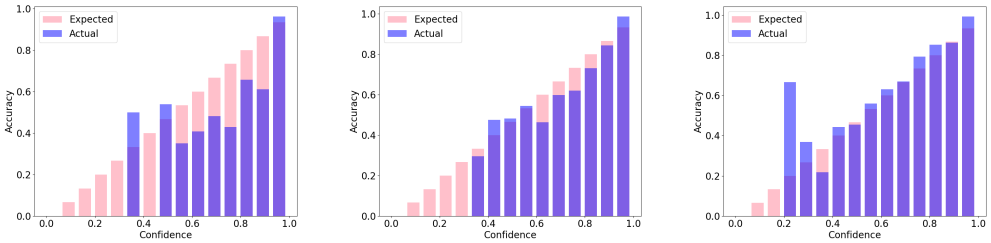
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 4.05$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.67$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 0.56$

Figure 21: CIFAR-10, ResNet-50.



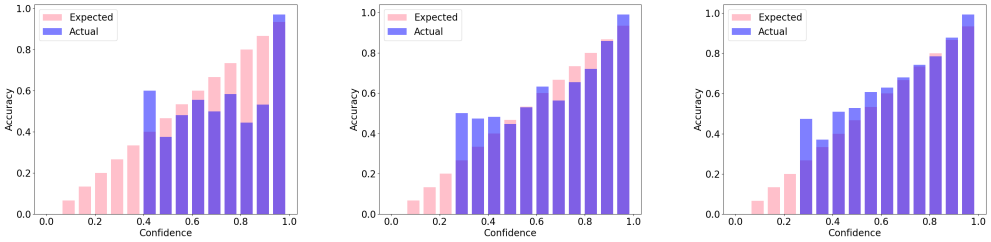
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 4.39$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.90$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 0.44$

Figure 22: CIFAR-10, ResNet-110.



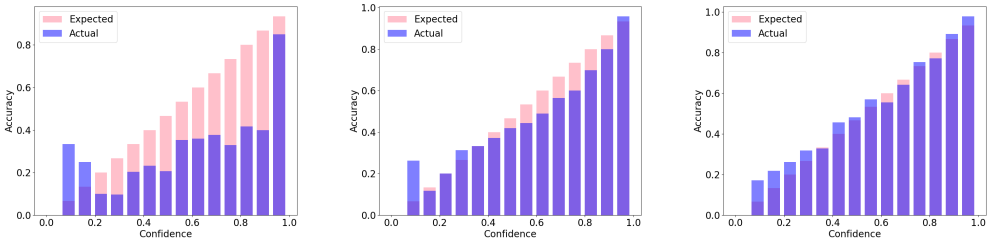
(a) CE, $\%\text{ECE}_{\text{EM}} = 4.26$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 1.62$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 0.54$

Figure 23: CIFAR-10, DenseNet-121.



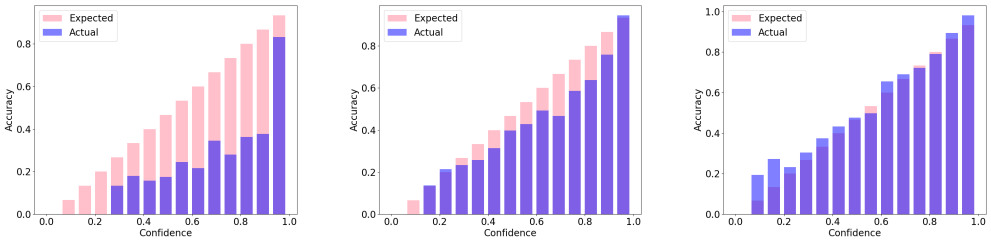
(a) CE, $\%\text{ECE}_{\text{EM}} = 3.52$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 2.01$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 0.64$

Figure 24: CIFAR-10, Wide-ResNet.



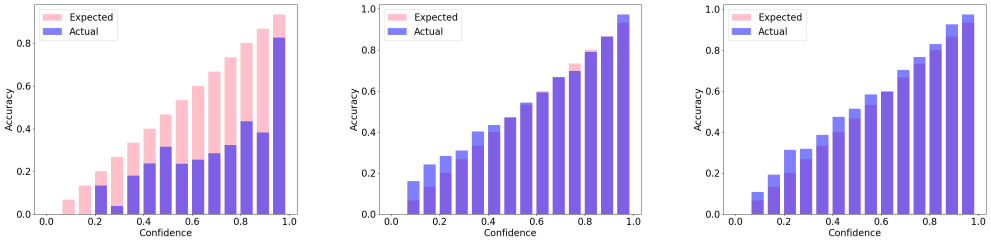
(a) CE, $\%\text{ECE}_{\text{EM}} = 17.72$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 5.57$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 1.72$

Figure 25: CIFAR-100, ResNet-50.



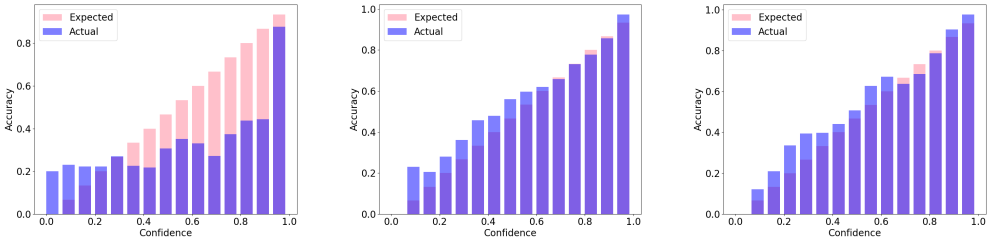
(a) CE, $\%\text{ECE}_{\text{EM}} = 19.44$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 7.34$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 1.57$

Figure 26: CIFAR-100, ResNet-110.



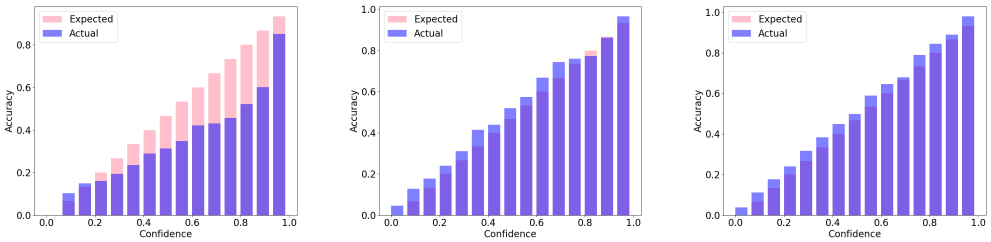
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 19.82$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.4$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.54$

Figure 27: CIFAR-100, DenseNet-121.



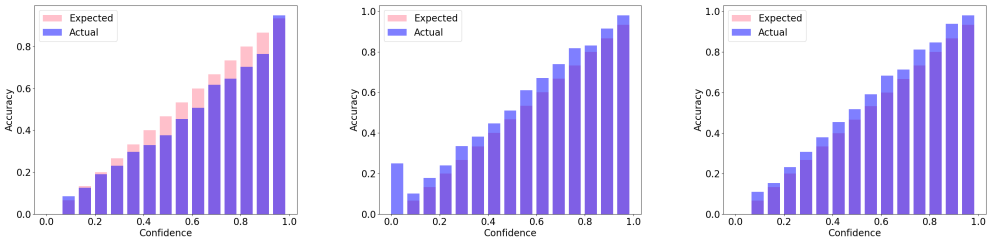
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 14.93$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.63$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.22$

Figure 28: CIFAR-100, Wide-ResNet.



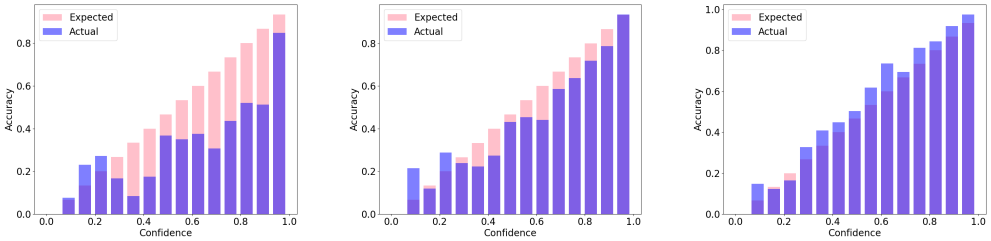
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 16.19$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.70$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 2.56$

Figure 29: Tiny-ImageNet, ResNet-50.



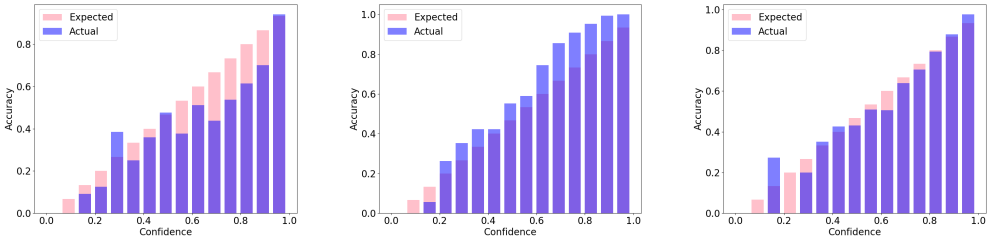
(a) CE, $\%\mathrm{ECE}_{\mathrm{EM}} = 8.11$ (b) FLSD-53, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.94$ (c) AdaFocal, $\%\mathrm{ECE}_{\mathrm{EM}} = 1.82$

Figure 30: Tiny-ImageNet, ResNet-110.



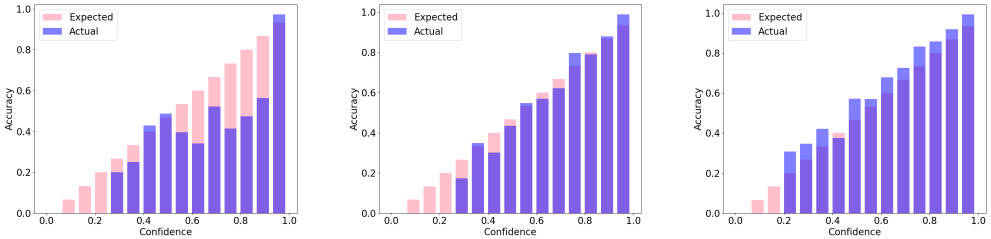
(a) CE, $\%\text{ECE}_{\text{EM}} = 18.37$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 8.95$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 2.38$

Figure 31: 20Newsgroup, CNN.



(a) CE, $\%\text{ECE}_{\text{EM}} = 7.89$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 9.75$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 3.18$

Figure 32: 20Newsgroup, BERT.



(a) CE, $\%\text{ECE}_{\text{EM}} = 3.17$ (b) FLSD-53, $\%\text{ECE}_{\text{EM}} = 0.59$ (c) AdaFocal, $\%\text{ECE}_{\text{EM}} = 0.62$

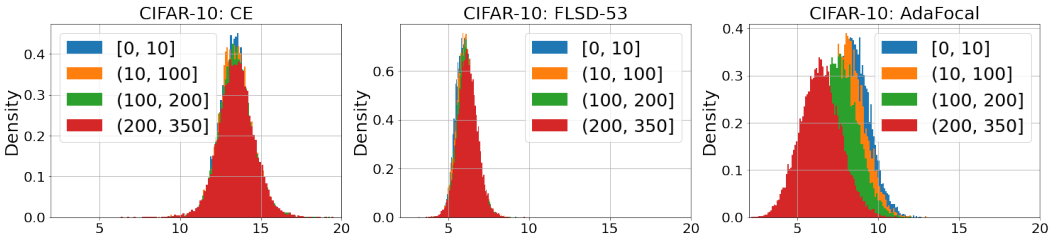
Figure 33: SVHN, ResNet-110.

M "Calibrate-able" Property

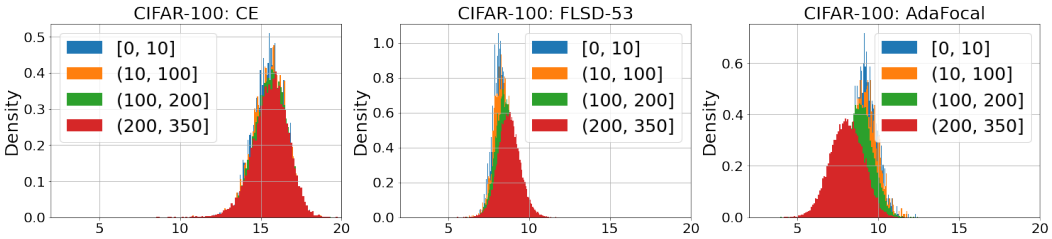
In Fig. 34, following [30], for ResNet-50 on CIFAR-10 and ResNet-50 on CIFAR-100, we plot the distribution of max-logits of training examples at the end of training (i.e. at epoch 350) grouped as per different "learned epochs".

Observations:

- Unlike for ResNet-32 in [30], we do not find cross entropy + Temperature Scaling to be better than focal loss + temperature scaling for ResNet-50 (the same is observed in [19]).
- Although the distribution of FLSD-53 is compressed, similar to what observed for focal loss in [30], the separation of samples is not seen here for cross entropy (CE).
- For AdaFocal, we see better separation of easy and hard examples grouped as per their "learned epoch". This makes AdaFocal more "calibrate-able" as per [30].



(a) ResNet-50 trained on CIFAR-10.



(b) ResNet-50 trained on CIFAR-100.

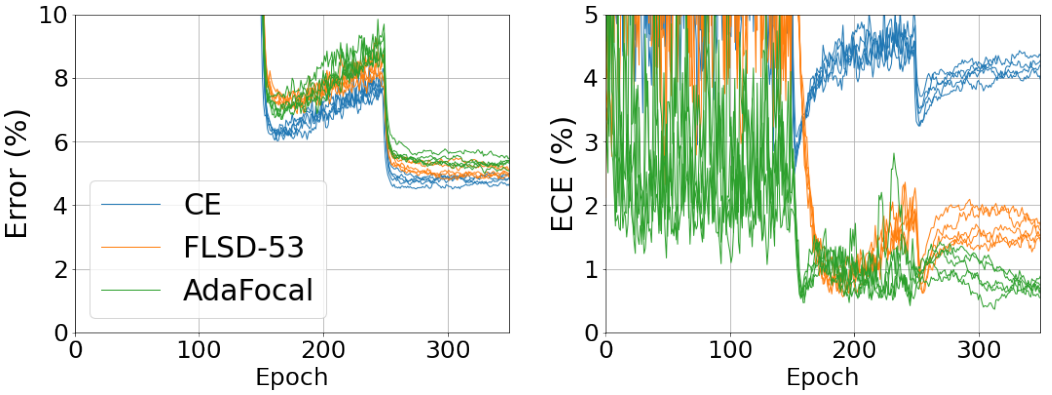
Figure 34: Comparison of the distribution of max-logits of training examples grouped as per different "learned epochs". In the legend, the groups are marked by intervals to which the "learned epoch" belongs.

N Multiple runs of AdaFocal with different $\gamma_{\max}$

Due to the stochastic nature of the experiments, AdaFocal γs may end up following different trajectories across different runs (initialization), which in turn might lead to variations in the final results. In this section, we look at the extent of such variations for ResNet-50 trained on CIFAR-10 for $\gamma_{\max} = 20$, $\gamma_{\max} = 50$ and unconstrained γ ($\gamma_{\max} = \infty$). For all these experiments, the minimum γ for inverse-focal loss is set to $\gamma_{\min} = -2$ and the switching threshold is set to $S_{th} = 0.2$.

N.1 AdaFocal $\gamma_{\max} = 20$

In Fig. 35, we observe that AdaFocal with $\gamma_{\max} = 20$ is consistently better than FLSD-53. Fig. 36 shows the variation in dynamics of γ during training across different runs.



(a) Error (%)

(b) ECE_{EM} (%)

Figure 35: Multiple runs of ResNet-50 trained on CIFAR-10 using cross entropy (CE), FLSD-53 and AdaFocal with $\gamma_{\max} = 20$ for different initialization seeds.

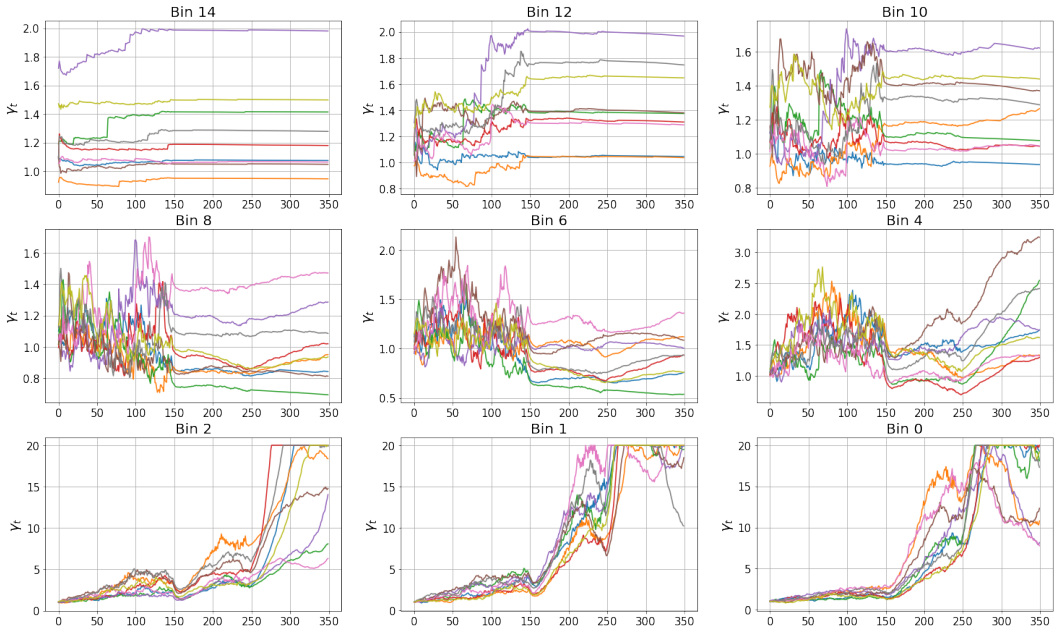
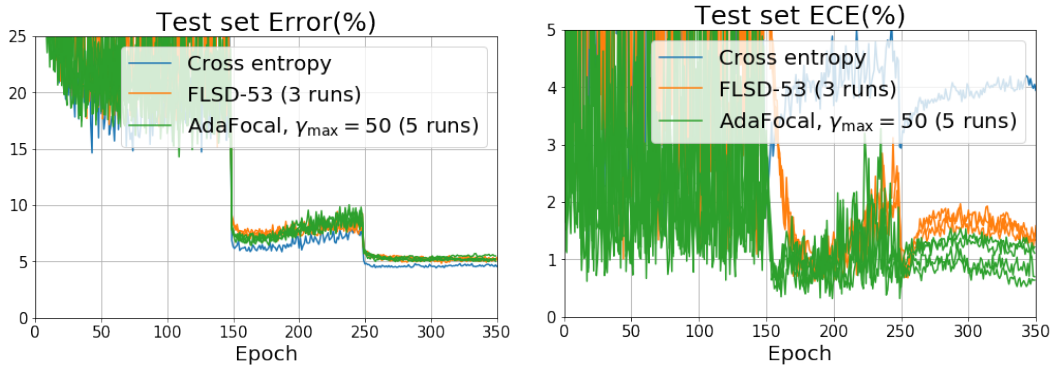


Figure 36: Dynamics of γ_t for different runs of ResNet-50 trained on CIFAR-10 using AdaFocal $\gamma_{\max} = 20$.

N.2 AdaFocal $\gamma_{\max} = 50$

In Fig. 37, we observe that AdaFocal with $\gamma_{\max} = 50$ has more variability than AdaFocal $\gamma_{\max} = 20$ but is mostly better than FLSD-53. Fig. 37 shows the variation in dynamics of γ during training across different runs.



(a) Error (%)

(b) ECE_{EM} (%)

Figure 37: Plots for ResNet-50 trained on CIFAR-10 using cross entropy (1 run), FLSD-53 (3 runs) and AdaFocal with $\gamma_{\max} = 50$ (5 runs). AdaFocal $\gamma_{\max} = 50$, although mostly better than FLSD-53, does exhibit greater variability than AdaFocal $\gamma_{\max} = 20$.

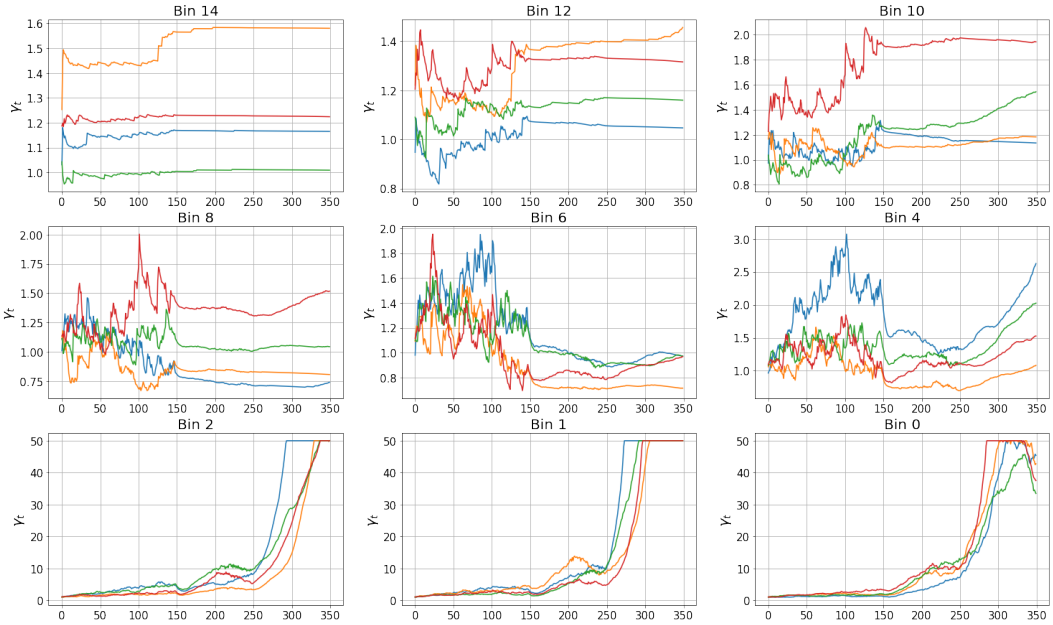
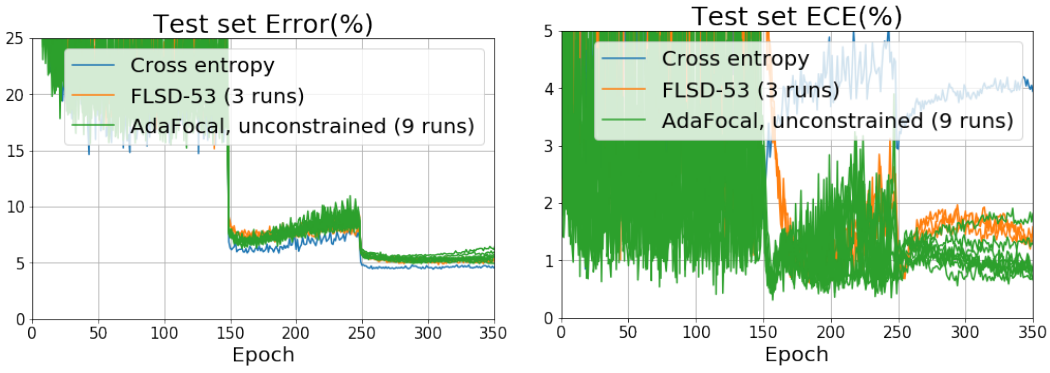


Figure 38: Dynamics of γ_t for different runs of ResNet-50 trained on CIFAR-10 using AdaFocal $\gamma_{\max} = 50$.

N.3 AdaFocal, unconstrained γ ($\gamma_{\max} = \infty$)

In Fig. 39, we observe that AdaFocal with unconstrained γ exhibit grater variability across different runs: 7 out of 9 times it performs better than FLSD-53 whereas the other two times it is similar or slightly worse.



(a) Error (%)

(b) ECE_{EM} (%)

Figure 39: Plots for ResNet-50 trained on CIFAR-10 using cross entropy (1 run), FLSD-53 (3 runs) and AdaFocal $\gamma_{\max} = \infty$ (9 runs). AdaFocal with unconstrained γ exhibits much greater variability across different runs than $\gamma_{\max} = 20$ and $\gamma_{\max} = 50$.

The above behaviour is mostly due to large variations in the trajectory of γs especially for lower bins as shown in Fig. 40. For higher bins, γs do not explode and settle to similar nearby values, whereas, for lower bins, as the γs are unconstrained they blow up to udesirably high values.

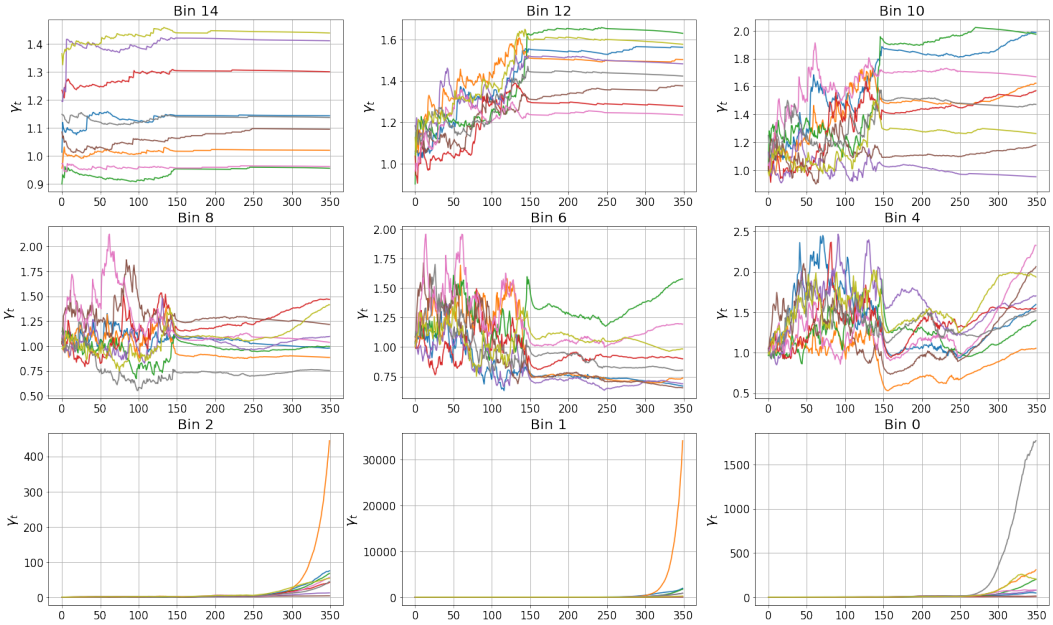
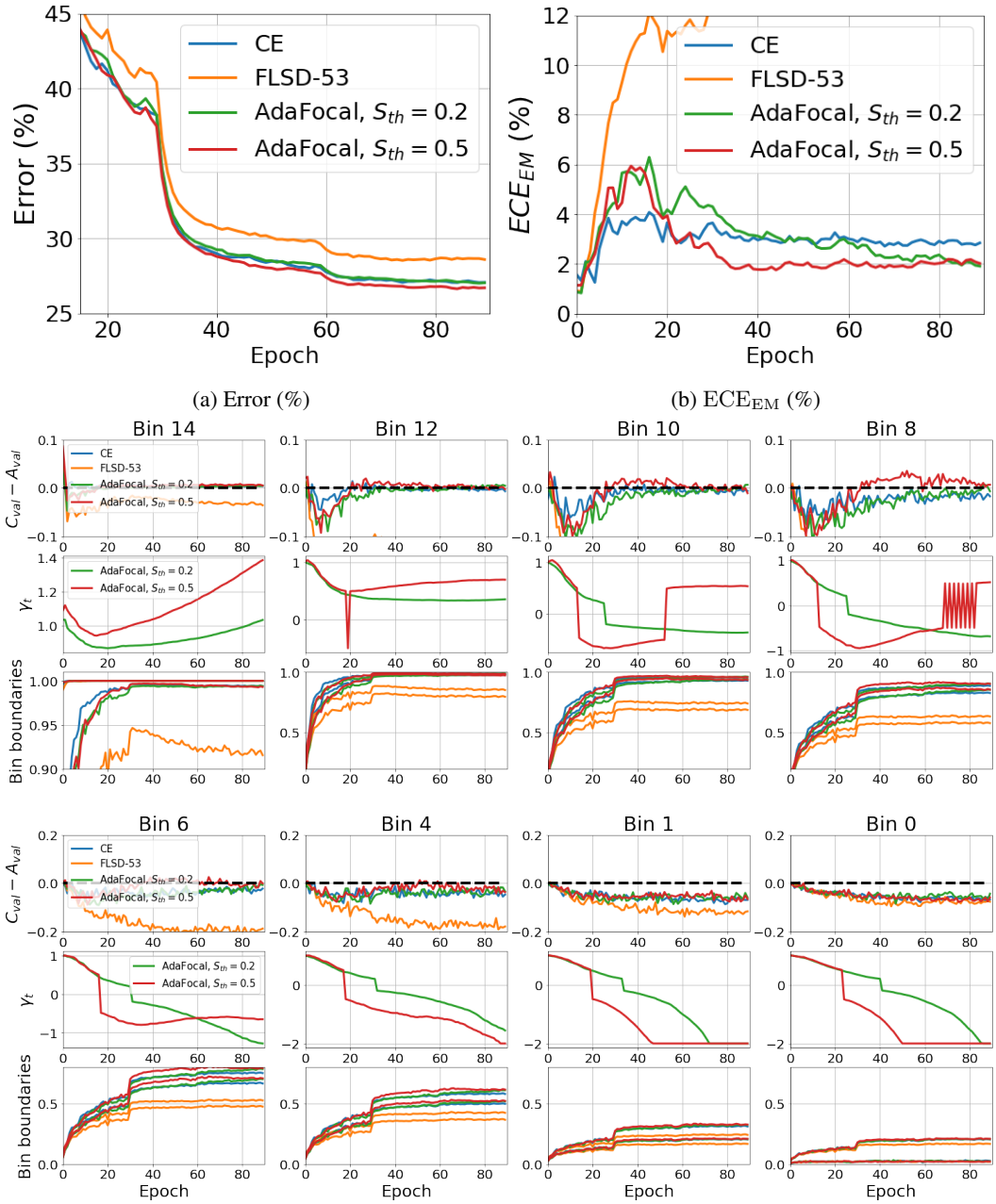


Figure 40: Dynamics of γ_t for different runs of ResNet-50 trained on CIFAR-10 using unconstrained AdaFocal with $\gamma_{\max} = \infty$.

O Error, ECE, dynamics of γ, and bin statistics during training

O.1 ImageNet, ResNet-50



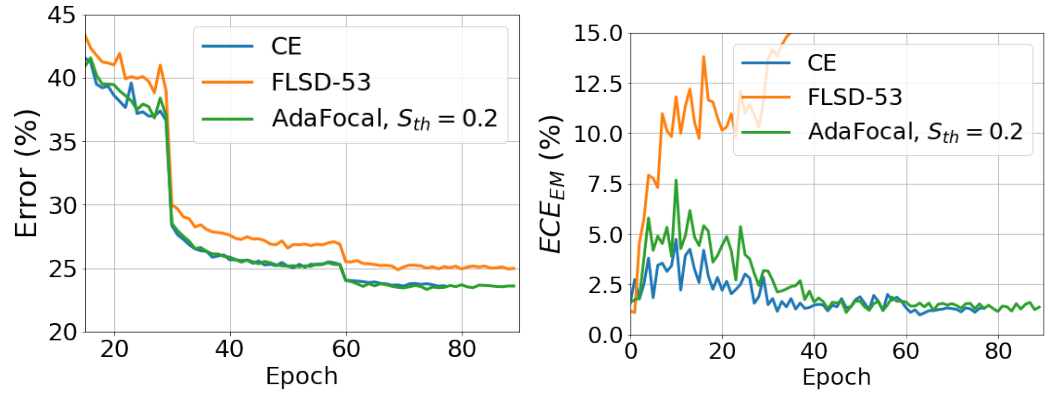
(a) Error (%)

(b) $\mathrm{ECE_{EM}}$ (%)

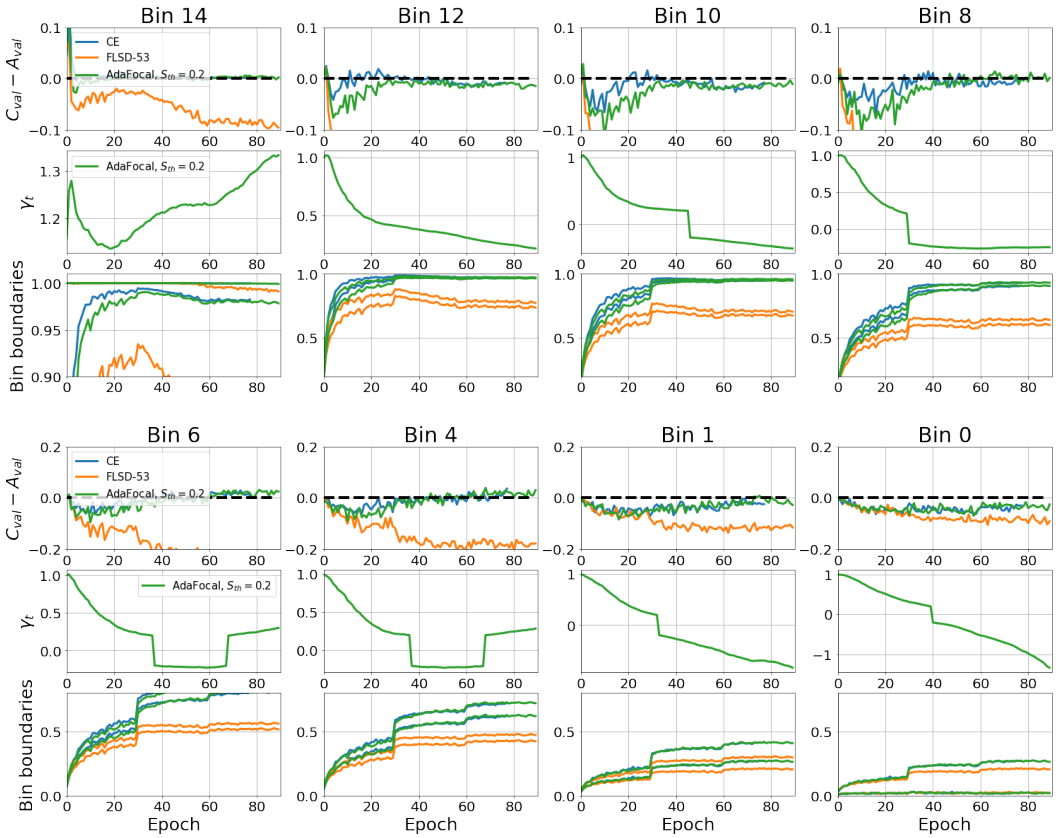
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 41: ResNet-50 trained on ImageNet with cross entropy (CE), FLSD-53, and AdaFocal with $S_{th} = 0.2$ and 0.5.

O.2 ImageNet, ResNet-110



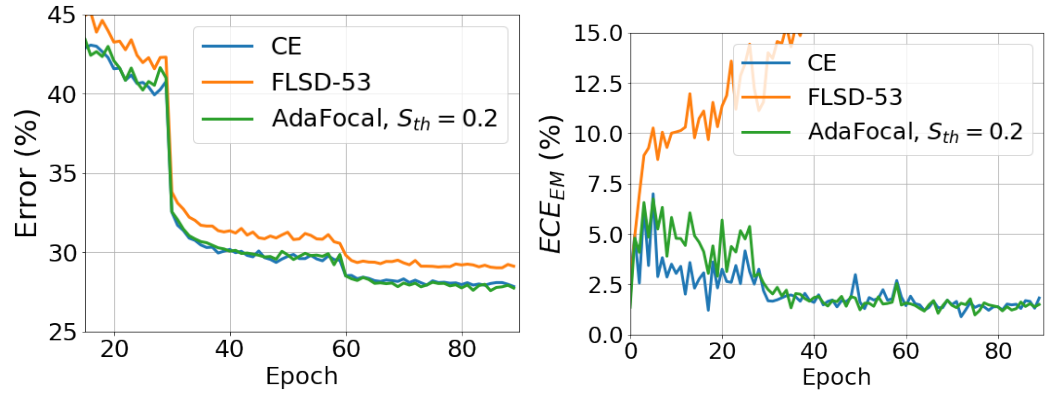
(a) Error (%) (b) $\mathrm{ECE}_{\mathrm{EM}}$ (%)



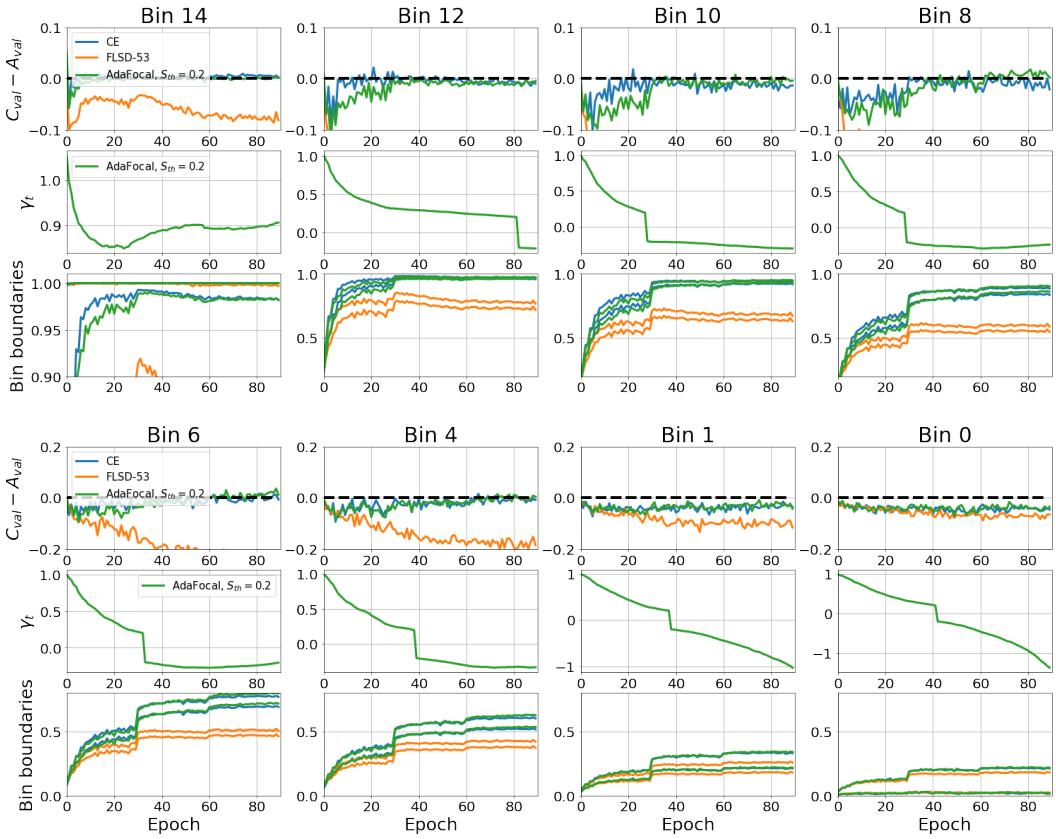
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 42: ResNet-110 trained on ImageNet with cross entropy (CE), FLSD-53, and AdaFocal with $S_{th} = 0.2$.

O.3 ImageNet, DenseNet-121



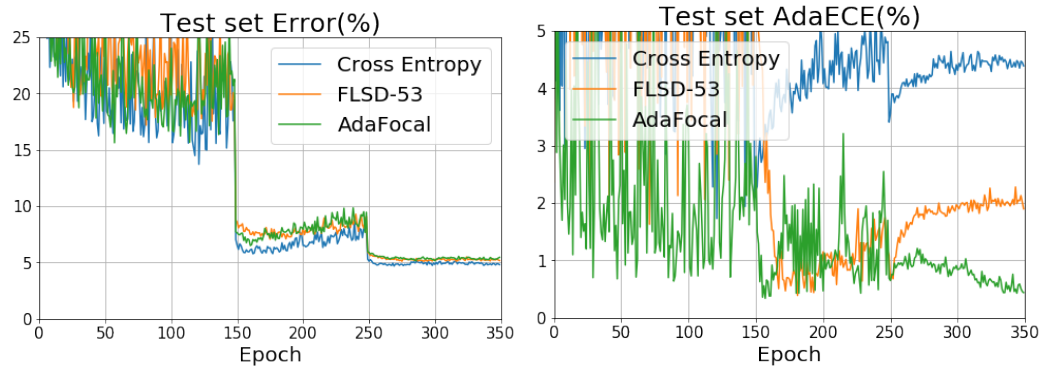
(a) Error (%) (b) $\mathrm{ECE_{EM}}$ (%)



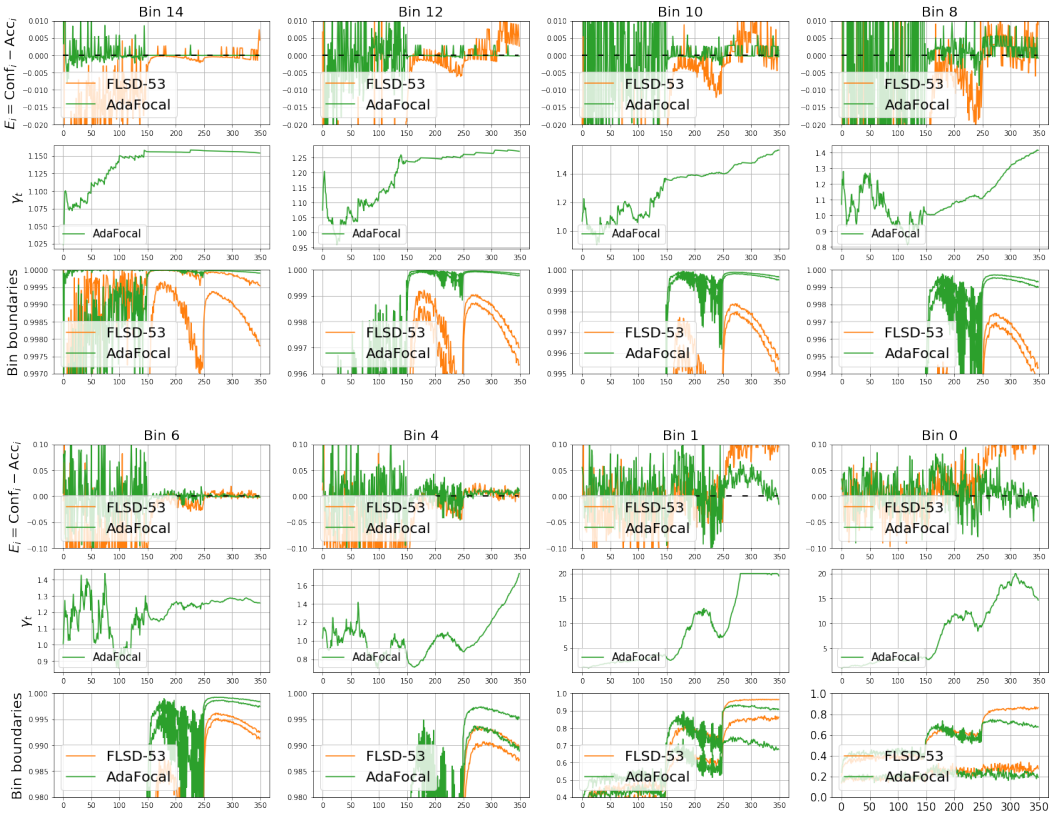
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 43: DenseNet-121 trained on ImageNet with cross entropy (CE), FLSD-53, and AdaFocal with $S_{th} = 0.2$.

O.4 CIFAR-10, ResNet-110



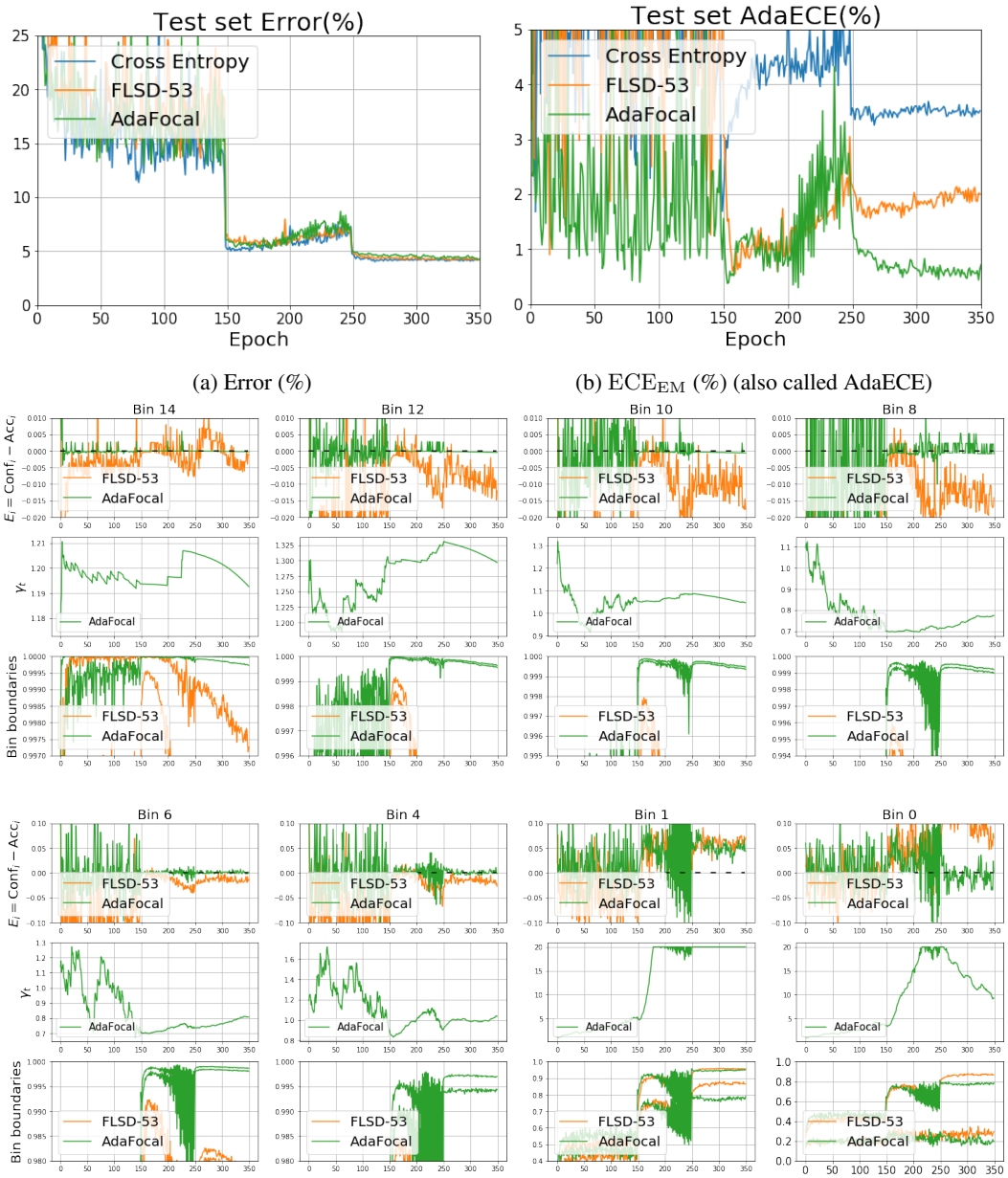
(a) Error (%) (b) $\mathrm{ECE}_{\mathrm{EM}}$ (%) (also called AdaECE)



(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 44: ResNet-110 trained on CIFAR-10 with cross entropy (CE), FLSD-53, and AdaFocal.

O.5 CIFAR-10, Wide-ResNet



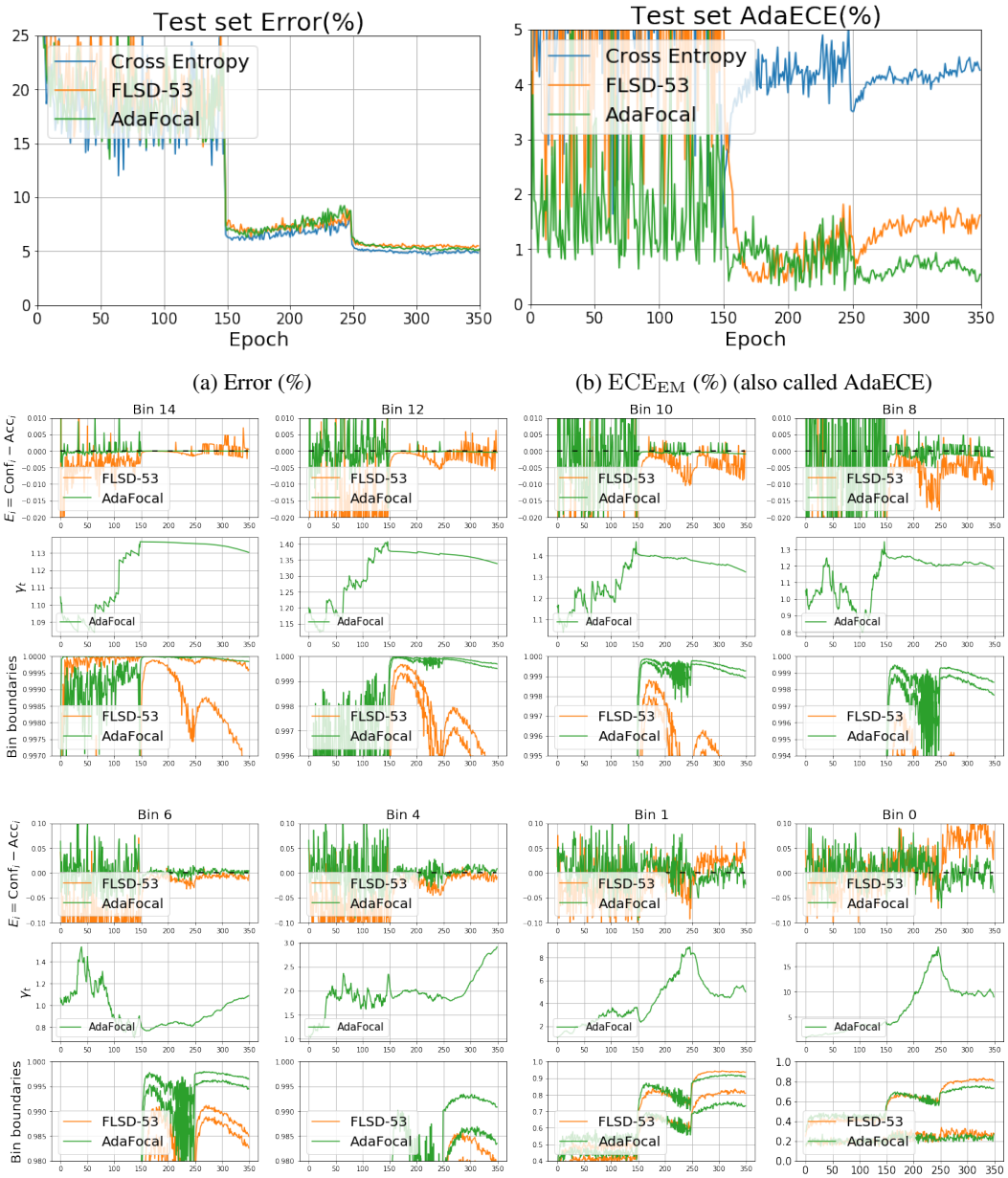
(a) Error (%)

(b) ECE_{EM} (%) (also called AdaECE)

(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 45: Wide-ResNet trained on CIFAR-10 with cross entropy (CE), FLSD-53, and AdaFocal.

O.6 CIFAR-10, DenseNet-121



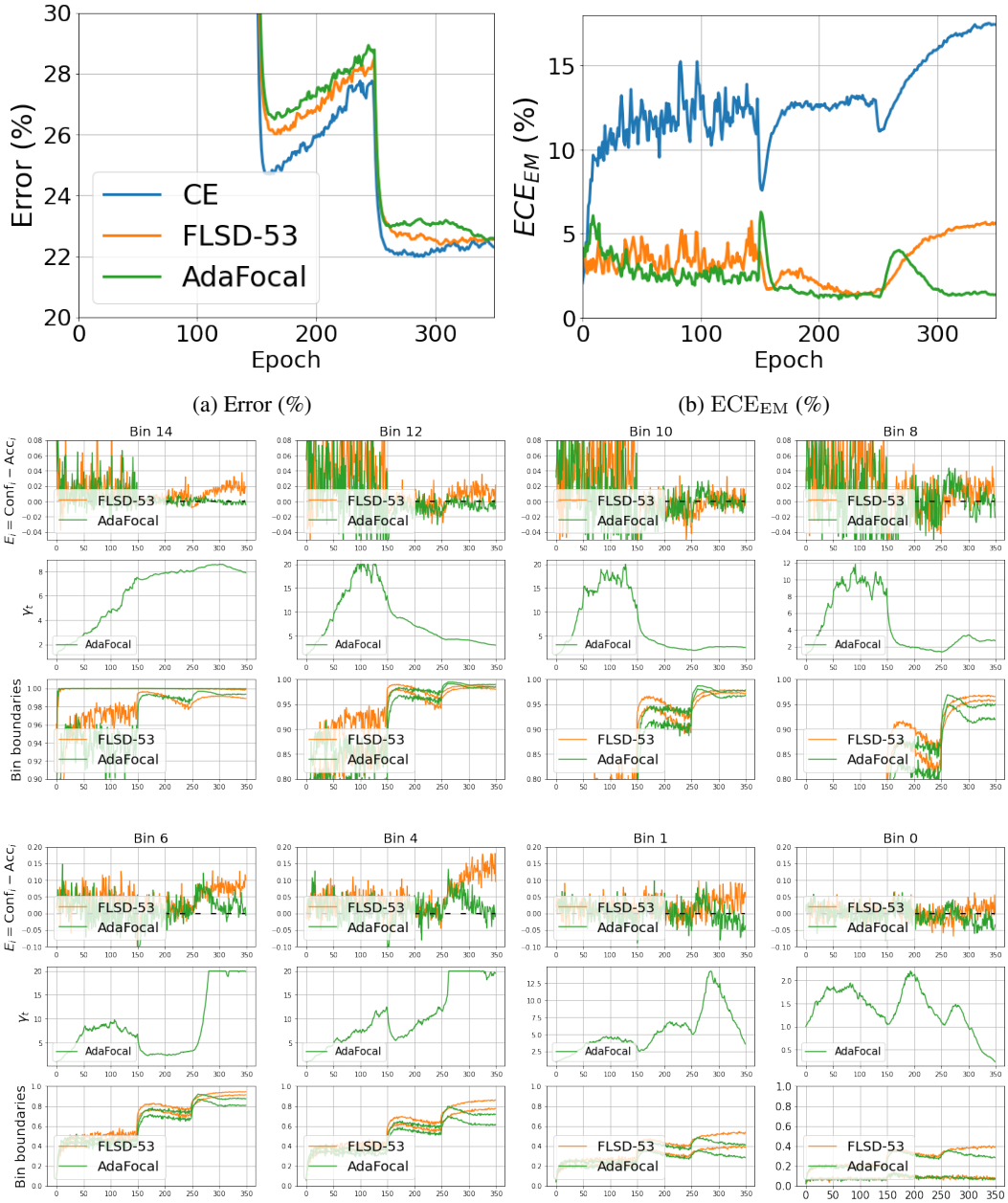
(a) Error (%)

(b) ECE_{EM} (%) (also called AdaECE)

(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 46: DenseNet-121 trained on CIFAR-10 with cross entropy (CE), FLSD-53, and AdaFocal.

O.7 CIFAR-100, ResNet-50



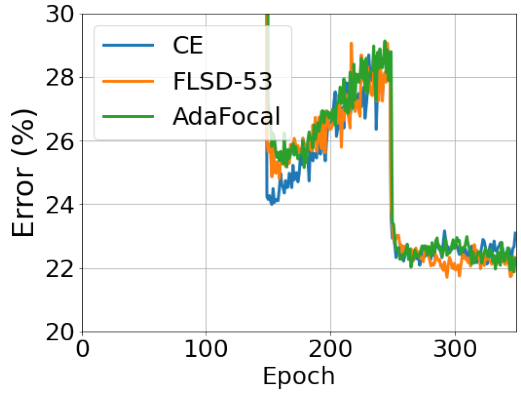
(a) Error (%)

(b) ECE_{EM} (%)

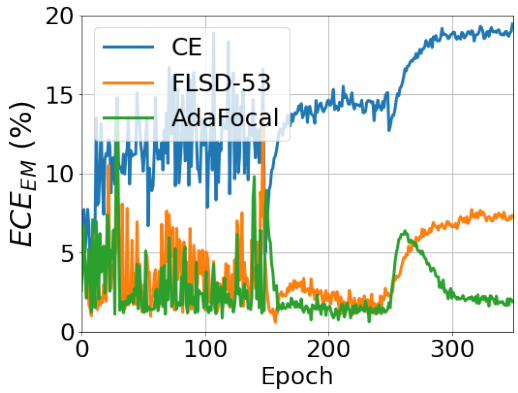
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 47: ResNet-50 trained on CIFAR-100 with cross entropy (CE), FLSD-53, and AdaFocal.

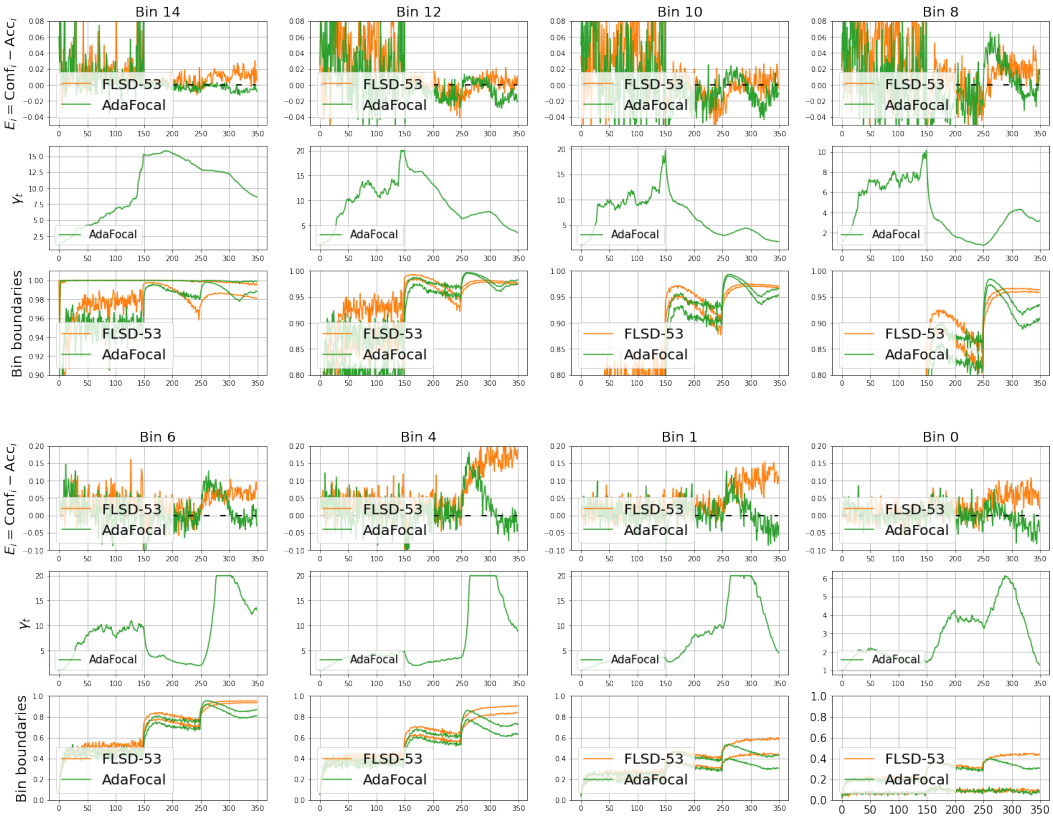
O.8 CIFAR-100, ResNet-110



(a) Error (%)



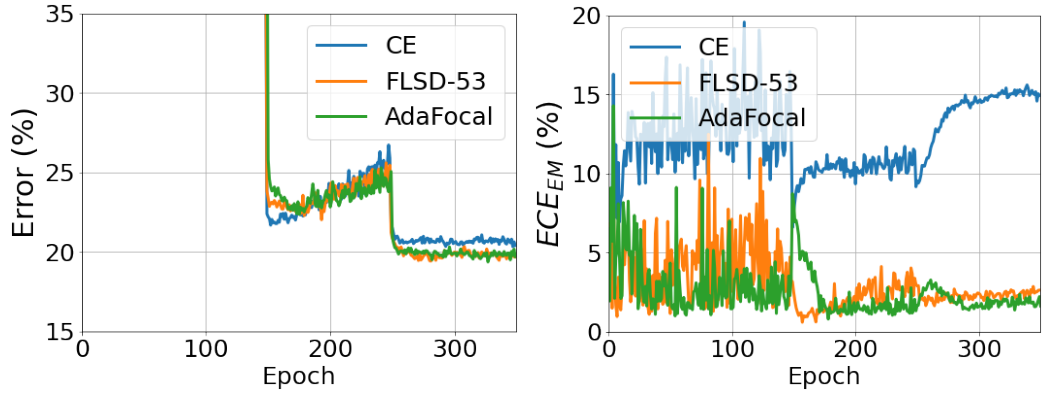
(b) $\mathrm{ECE}_{\mathrm{EM}}$ (%)



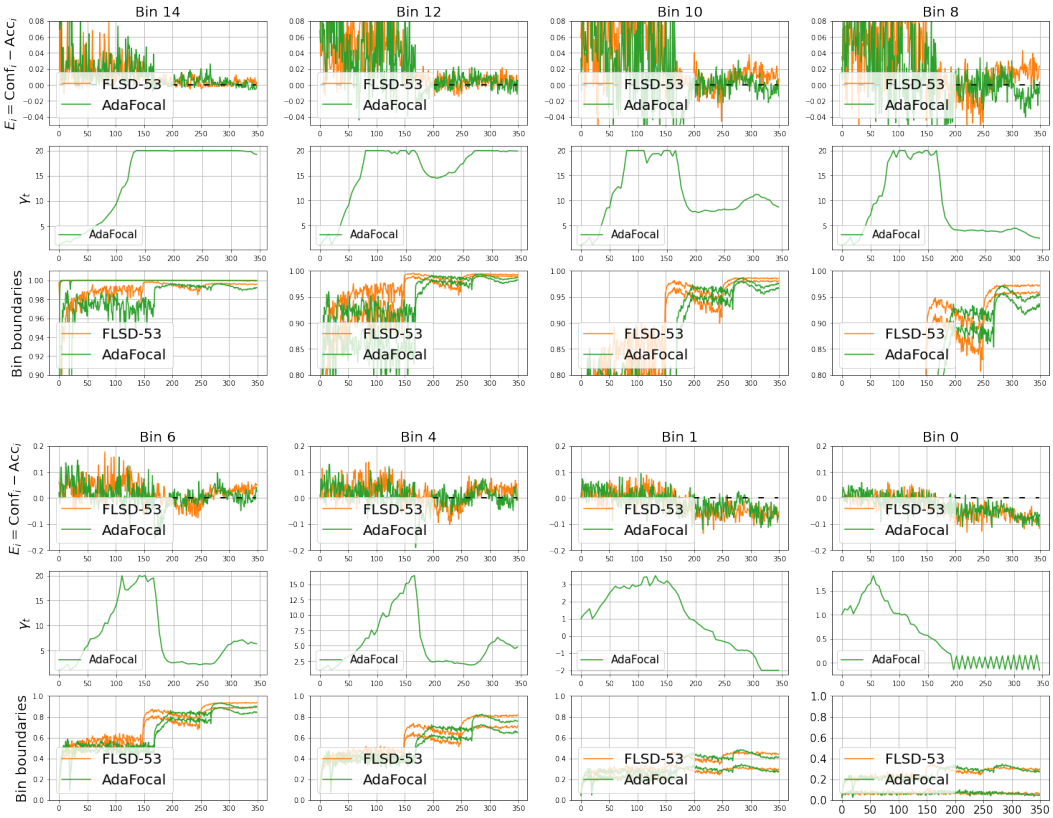
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 48: ResNet-110 trained on CIFAR-100 with cross entropy (CE), FLSD-53, and AdaFocal.

O.9 CIFAR-100, Wide-ResNet



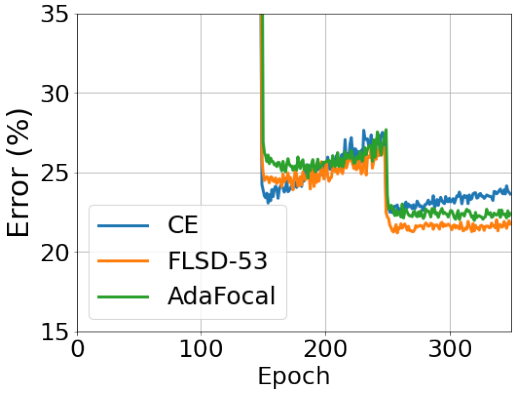
(a) Error (%) (b) $\mathrm{ECE_{EM}}$ (%)



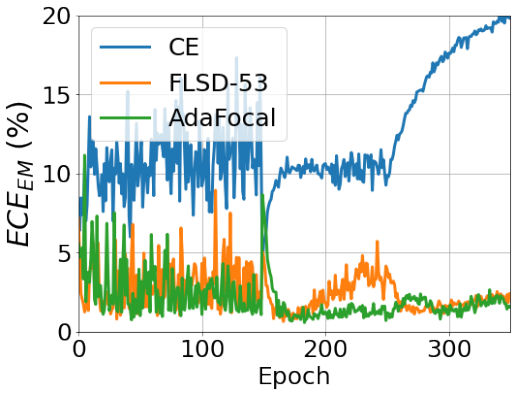
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 49: Wide-ResNet trained on CIFAR-100 with cross entropy (CE), FLSD-53, and AdaFocal.

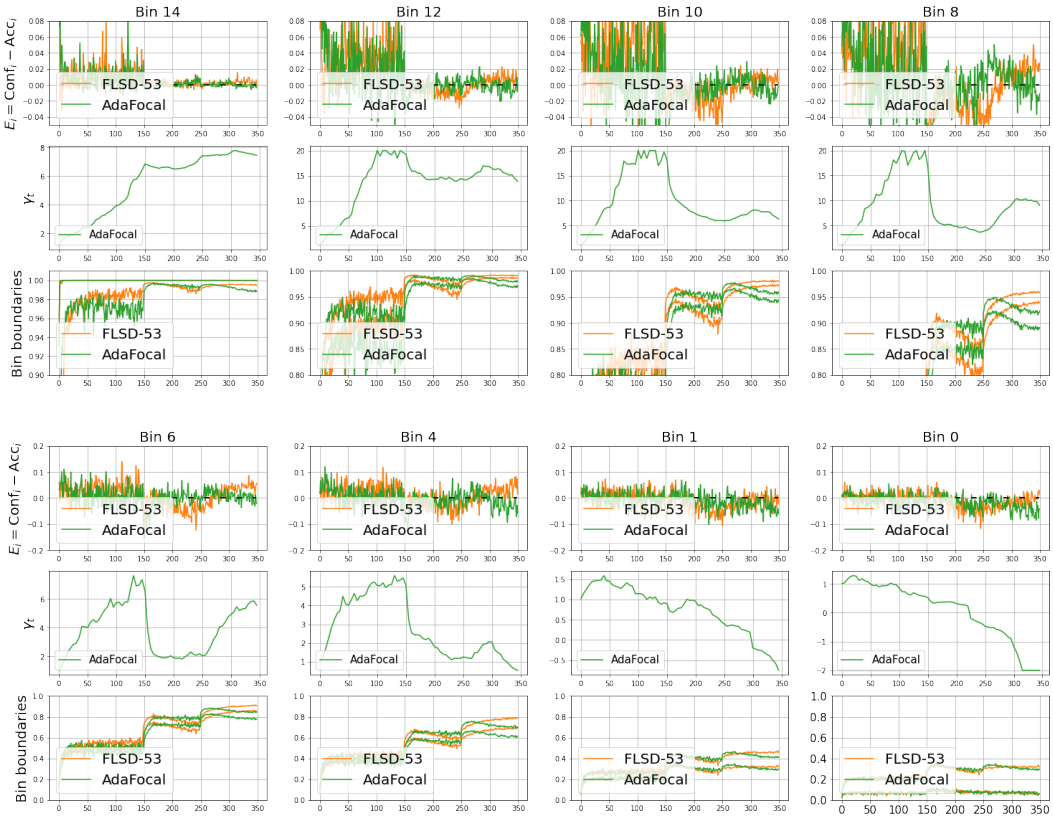
O.10 CIFAR-100, DenseNet-121



(a) Error (%)



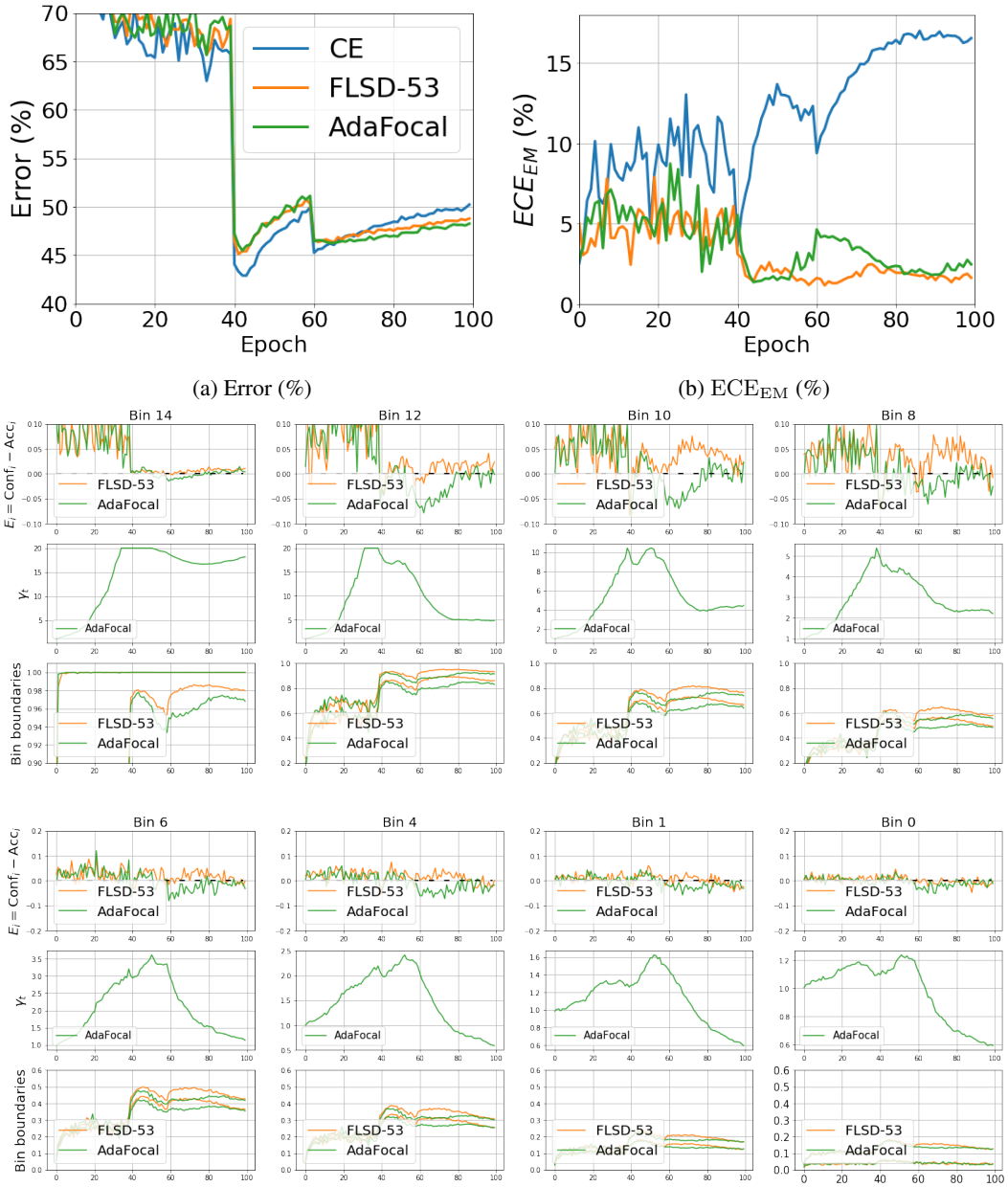
(b) $\mathrm{ECE}_{\mathrm{EM}}$ (%)



(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 50: DenseNet-121 trained on CIFAR-100 with cross entropy (CE), FLSD-53, and AdaFocal.

O.11 Tiny-ImageNet, ResNet-50



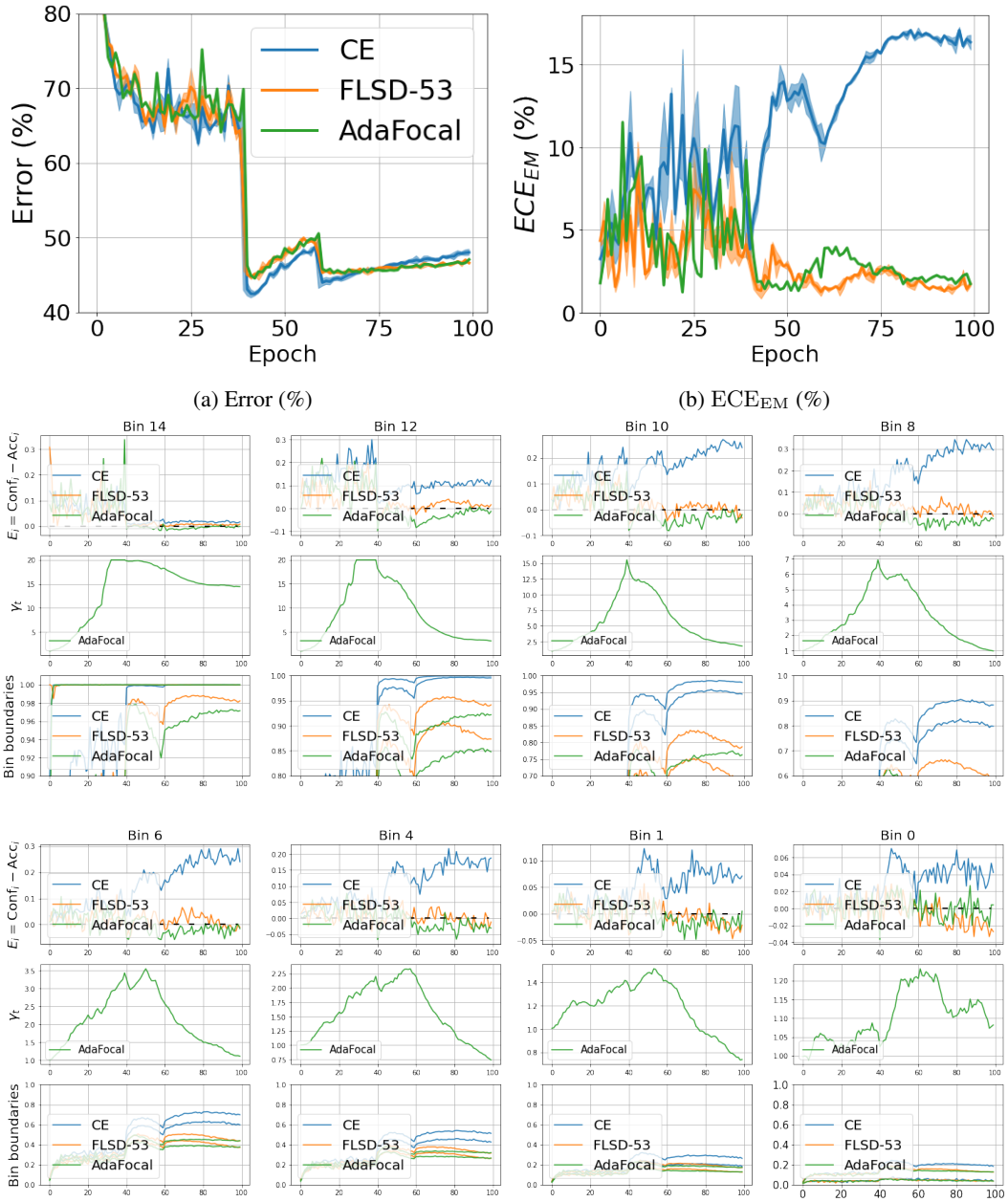
(a) Error (%)

(b) ECE_{EM} (%)

(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 51: ResNet-50 trained on Tiny-ImageNet with cross entropy (CE), FLSD-53, and AdaFocal.

O.12 Tiny-ImageNet, ResNet-110



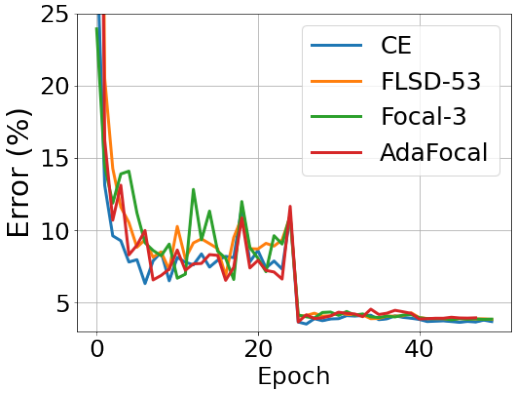
(a) Error (%)

(b) ECE_{EM} (%)

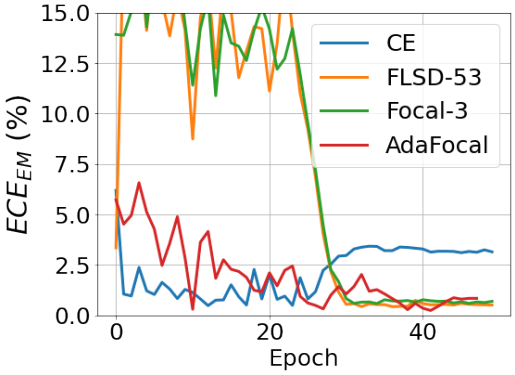
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 52: ResNet-110 trained on Tiny-ImageNet with cross entropy (CE), FLSD-53, and AdaFocal.

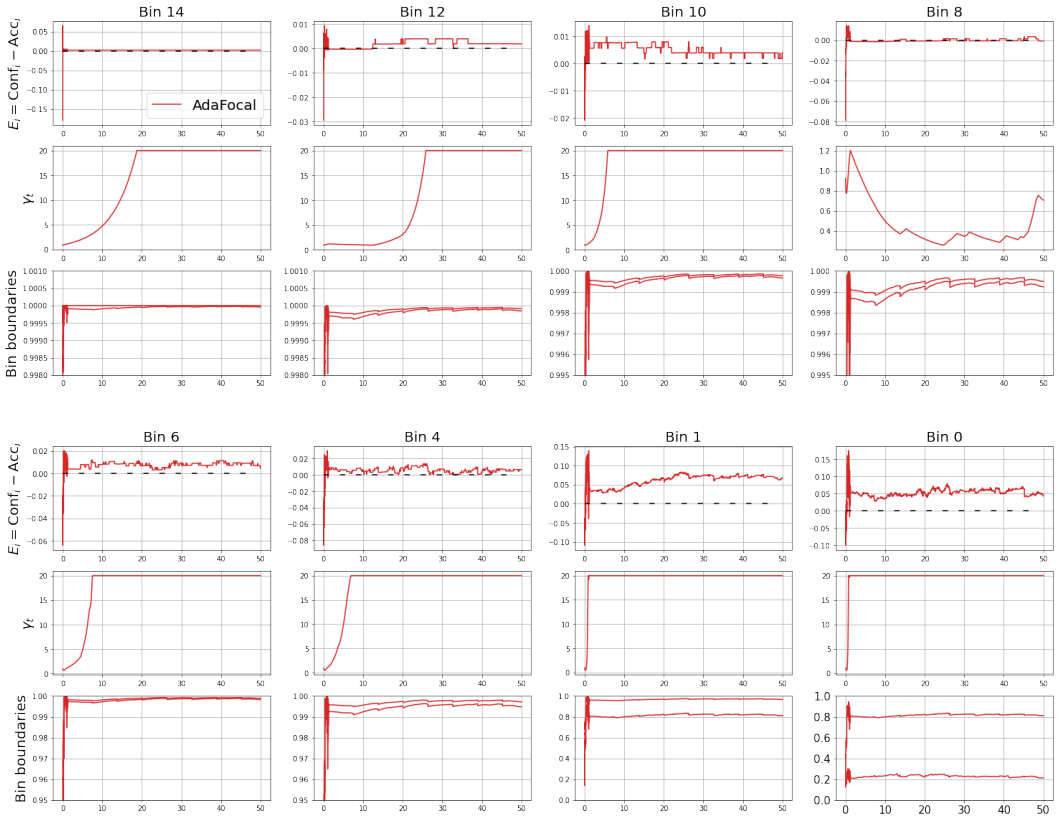
O.13 SVHN, ResNet-110



(a) Error (%)



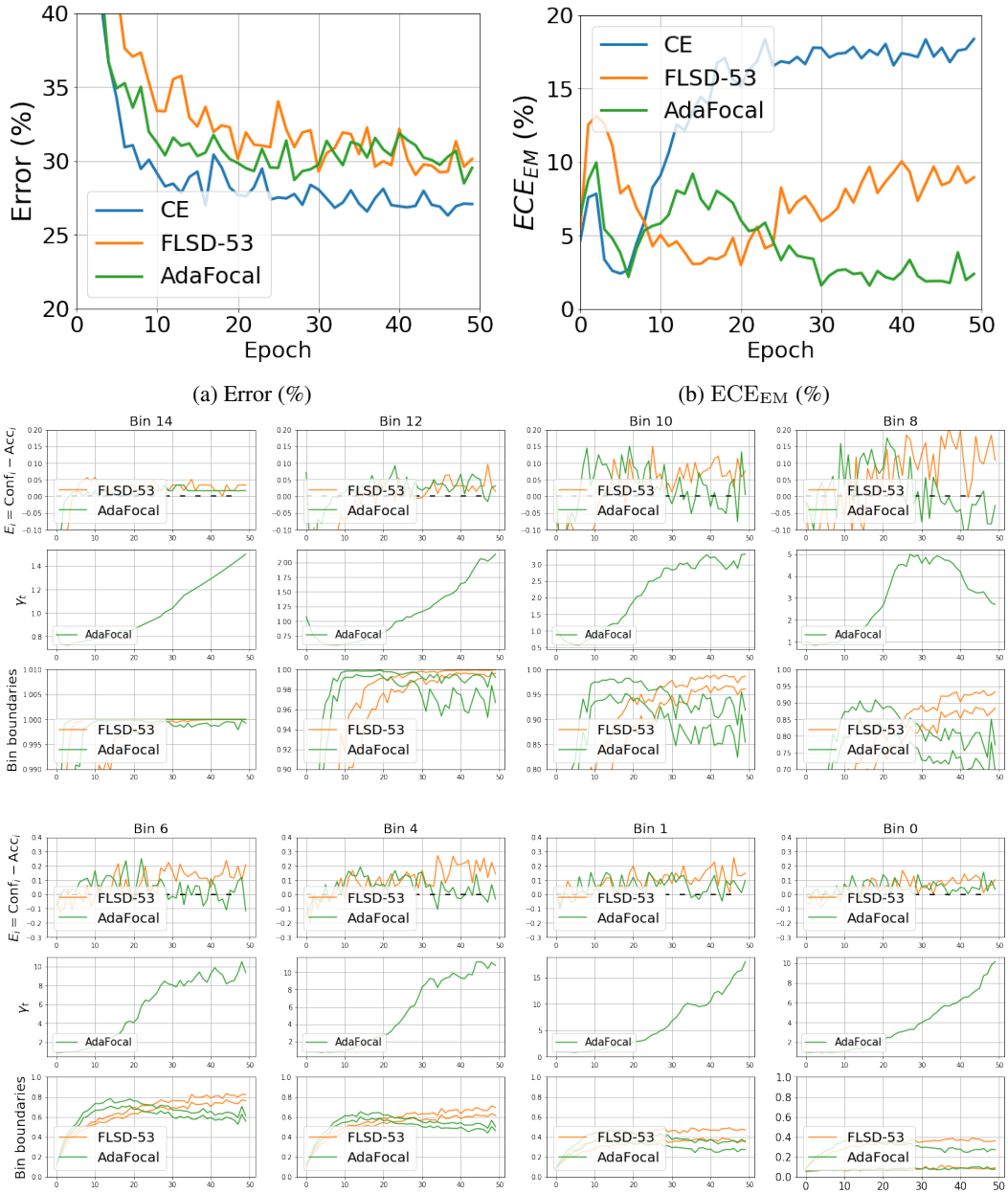
(b) $\mathrm{ECE}_{\mathrm{EM}}$ (%)



(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 53: ResNet-110 trained on SVHN with cross entropy (CE), FLSD-53, AdaFocal, AdaFocal-shcedule. In AdaFocal-schedule, for the first 25 epochs γ is updated every epoch, from epoch 25 to 40, γ is updated every 100 mini-batches, and from epoch 40 to 50 (end of training), γ is updated every mini-batch.

O.14 20 Newsgroups, CNN



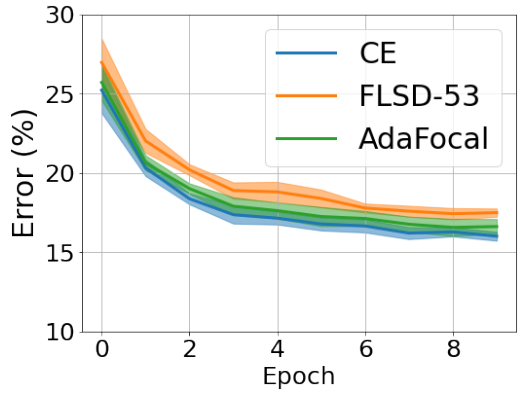
(a) Error (%)

(b) ECE_{EM} (%)

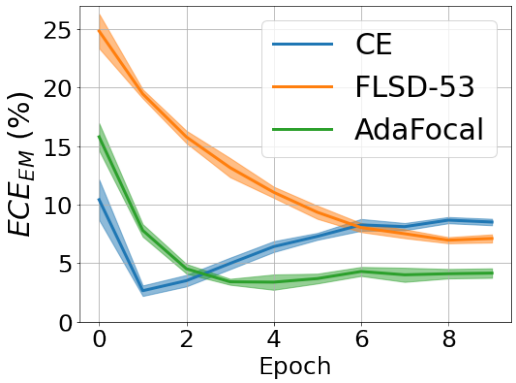
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 54: CNN trained on 20 Newsgroups with cross entropy (CE), FLSD-53, and AdaFocal.

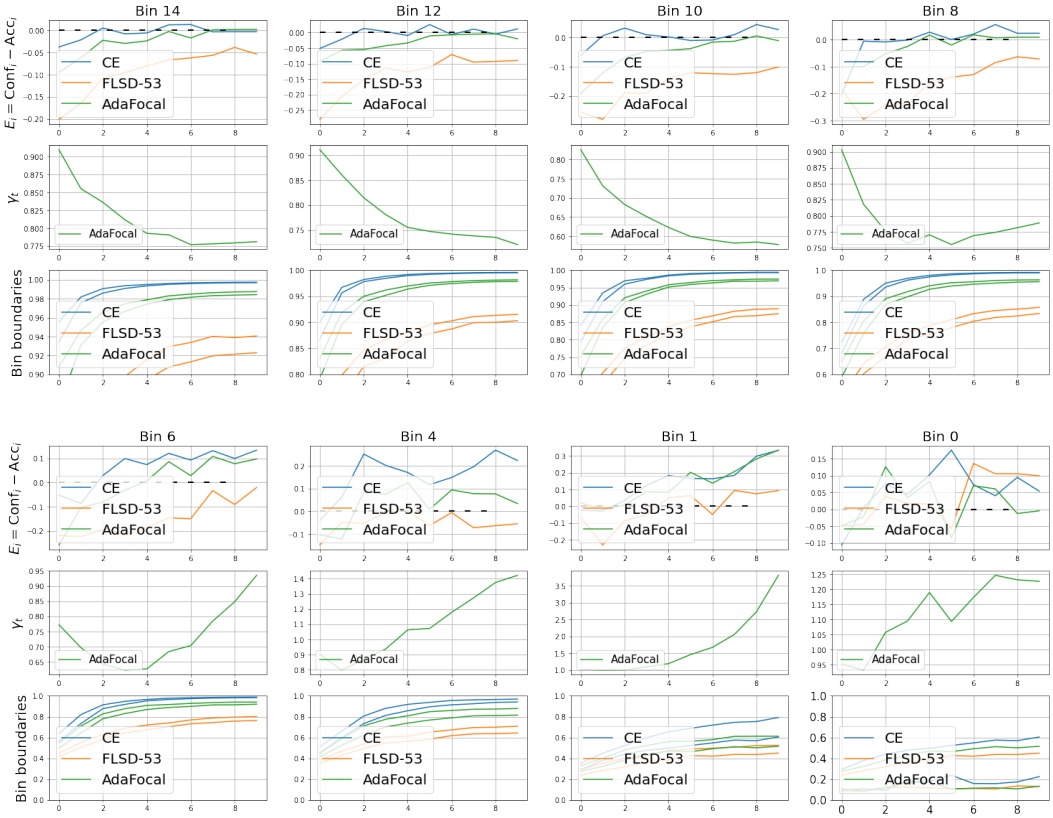
O.15 20 Newsgroups, BERT



(a) Error (%)



(b) $\mathrm{ECE}_{\mathrm{EM}}$ (%)



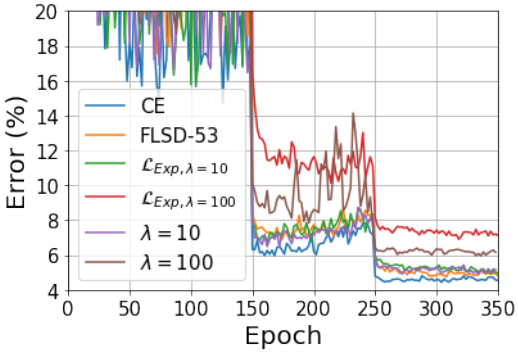
(c) Dynamics of γ and calibration behaviour in different bins. Each bin has three subplots: **top**: $E_{val,i} = C_{val,i} - A_{val,i}$, **middle**: evolution of γ_t, and **bottom**: bin boundaries. Black dashed line in top plot represent zero calibration error.

Figure 55: Pre-trained BERT fine-tuned on 20 Newsgroups with cross entropy (CE), FLSD-53, and AdaFocal.

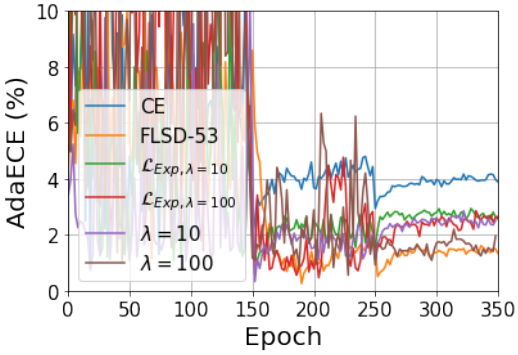
P Comparison of CalFocal Loss Case 1 (Eq. 2) and Case 2 (Eq. 3)

For Fig. 56 below, please note the following legend:

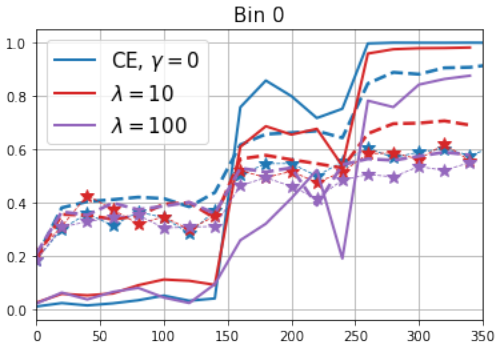
- CE = Cross Entropy.
- $\mathcal{L}_{Exp,\lambda}$ = CalFocal loss case 1 (Eq. 2 in the main paper) which assigns γs to each training sample.
- λ = CalFocal loss case 2 (Eq. 3 in the main paper) which assigns a common γ_b to all training samples that fall in validation-bin b.



(a) Error (%)



(b) $\mathrm{ECE}_{\mathrm{EM}}$ (%) (also called AdaECE in the literature [19]).



(c) For CalFocal loss case 2 (Eq. 3) marked in the legend by "$\lambda =$", the figure compares C_{train} (solid line), C_{val} (dashed line) and A_{val} (starred lines) in validation bin-0 to show that when CalFocal brings C_{train} closer to A_{val}, C_{val} also approaches A_{val}.

Figure 56: ResNet-50 trained on CIFAR-10 using (1) cross entropy (CE), (2) FLSD-53 (3) CalFocal Case 1 loss function in Eq. 2 denoted by $\mathcal{L}_{Exp,\lambda}$, and (4) CalFocal Case 2 loss function in Eq. 3 denoted by "$\lambda =$".