On using Focal Loss for Neural Network Calibration

Jishnu Mukhoti [*1 2] **Viveka Kulharia** [*2] **Amartya Sanyal** [2 3] **Stuart Golodetz** [1] **Philip H. S. Torr** [1 2]
Puneet K. Dokania [1 2]

Abstract

Miscalibration – a mismatch between a model's confidence and its correctness – of Deep Neural Networks (DNNs) makes their predictions hard to rely on. Ideally, we want networks to be accurate and calibrated. In this work, we study focal loss as an alternative to the conventional cross-entropy loss and show that, focal loss allows us to learn models that are comparitively well calibrated while preserving accuracy. We provide a thorough analysis of the factors causing miscalibration, and use the insights we glean from this to justify the superior performance of focal loss. Finally, we perform extensive experiments on a variety of datasets, and with a wide variety of network architectures, and show that focal loss indeed achieves excellent calibration without compromising on accuracy in almost all cases.

1. Introduction

Deep neural networks have dominated computer vision and machine learning in recent years, and this has led to their widespread deployment in real-world systems (Cao et al., 2018; Chen et al., 2018; Kamilaris & Prenafeta-Boldú, 2018; Ker et al., 2018; Wang et al., 2018). However, many current multi-class classification networks in particular are poorly calibrated, in the sense that the probability values that they predict for class labels often overestimate the likelihoods of those class labels being correct in reality. This makes it difficult for downstream components to trust the predictions of such models. The underlying cause is hypothesised to be that these networks' high capacity leaves them vulnerable to overfitting on the negative log-likelihood (NLL) loss they conventionally use during training (Guo et al., 2017).

Numerous suggestions for addressing this problem have been proposed. Much work has been inspired by approaches that were not originally formulated in a deep learning context, such as Platt scaling (Platt, 1999), histogram binning (Zadrozny & Elkan, 2001), isotonic regression (Zadrozny & Elkan, 2002), and Bayesian binning and averaging (Naeini et al., 2015; Naeini & Cooper, 2016). As deep learning has become more dominant, however, various works have begun to directly target the calibration of deep networks. For example, Guo et al. (Guo et al., 2017) have popularised a modern variant of Platt scaling known as *temperature scaling*, which works by dividing a network's logits by a scalar $T > 0$ (learnt on a validation subset) prior to performing softmax. Temperature scaling has the desirable property that it can improve the calibration of a network without in any way affecting its accuracy.

However, whilst its simplicity and effectiveness have made it a popular calibration method, it does have downsides. For example, whilst it scales the logits to reduce the network's confidence in incorrect predictions, this also slightly reduces the network's confidence in predictions that were correct. By contrast, (Kumar et al., 2018) initially eschew temperature scaling in favour of minimising a differentiable proxy for calibration error at training time, called Maximum Mean Calibration Error (MMCE), although they do later also use temperature scaling as a post-processing step to obtain better results than cross-entropy followed by temperature scaling (Guo et al., 2017). Separately, (Müller et al., 2019) propose training models on cross-entropy loss with label smoothing instead of one-hot labels and show that label smoothing has a very favourable effect on model calibration.

In this paper, we empirically study a technique for improving network calibration that works by replacing the cross-entropy loss conventionally used when training classification networks with the focal loss proposed by (Lin et al., 2017). We observe that unlike cross-entropy, which minimises the KL divergence between the predicted (softmax) distribution and the target distribution (one-hot encoding in classification tasks) over classes, focal loss minimises a regularised KL divergence between these two distributions, which ensures minimisation of the KL divergence whilst *increasing the entropy* of the predicted distribution, thereby preventing the model from becoming overconfident.

Overall, we make the following contributions. Firstly, in §3, we perform an empirical study of the link that (Guo et al., 2017) observed between miscalibration and NLL overfitting

[*]Equal contribution [1]FiveAI Ltd. [2]University of Oxford [3]The Alan Turing Institute, London. Correspondence to: Jishnu Mukhoti <jishnu.mukhoti@eng.ox.ac.uk>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning.

and also show that this overfitting is significantly lower in case of focal loss as compared to cross-entropy. Secondly, in §4, we show, via multiple classification experiments, that DNNs trained with focal loss are more calibrated than those trained with cross-entropy loss (both with and without label smoothing), MMCE or Brier loss (Brier, 1950).

2. Problem Formulation

Let $D = \langle(\mathbf{x}_i, y_i)\rangle_{i=1}^N$ denote a dataset consisting of N samples from a joint distribution $\mathcal{D}(\mathcal{X}, \mathcal{Y})$, where for each sample i, $\mathbf{x}_i \in \mathcal{X}$ is the input and $y_i \in \mathcal{Y} = \{1, 2, ..., K\}$ is the ground-truth class label. Let $\hat{p}_{i,y} = f_\theta(y|\mathbf{x}_i)$ be the probability that a neural network f with model parameters θ predicts for a class y on a given input $\mathbf{x}_i$. The class that f predicts for $\mathbf{x}_i$ is computed as $\hat{y}_i = \text{argmax}_{y\in\mathcal{Y}}\ \hat{p}_{i,y}$, and the predicted confidence as $\hat{p}_i = \max_{y\in\mathcal{Y}}\ \hat{p}_{i,y}$. The network is said to be *perfectly calibrated* when, for each sample $(\mathbf{x}, y) \in D$, the confidence $\hat{p}$ is equal to the model accuracy $\mathbb{P}(\hat{y} = y|\hat{p})$, i.e. the probability that the predicted class is correct. For instance, of all the samples to which a perfectly calibrated neural network assigns a confidence of 0.8, 80% should be correctly predicted.

A popular metric used to measure model calibration is the *expected calibration error* (ECE) (Naeini et al., 2015), defined as the expected absolute difference between the model's confidence and its accuracy, i.e. $\mathbb{E}_{\hat{p}}\big[\,|\mathbb{P}(\hat{y} = y|\hat{p}) - \hat{p}|\,\big]$. Since we only have finite samples, the ECE is approximated in practice by dividing the interval $[0, 1]$ into M equispaced bins, where the i^{th} bin is the interval $\big(\frac{i-1}{M}, \frac{i}{M}\big]$. Let B_i denote the set of samples with confidences belonging to the i^{th} bin. The accuracy A_i of this bin is computed as $A_i = \frac{1}{|B_i|}\sum_{j\in B_i} \mathbb{1}\left(\hat{y}_j = y_j\right)$, where $\mathbb{1}$ is the indicator function, and $\hat{y}_j$ and y_j are the predicted and ground-truth labels for the j^{th} sample. Similarly, the confidence C_i of the i^{th} bin is computed as $C_i = \frac{1}{|B_i|}\sum_{j\in B_i} \hat{p}_j$, i.e. C_i is the average confidence of all samples in the bin. The ECE can be approximated as a weighted average of the absolute difference between the accuracy and confidence of each bin: $\text{ECE} = \sum_{i=1}^M \frac{|B_i|}{N}\,|A_i - C_i|$. A similar metric, the *maximum calibration error* (MCE) (Naeini et al., 2015), is defined as the maximum absolute difference between the confidence and accuracy of each bin: $\text{MCE} = \max_{i\in\{1,...,M\}}\,|A_i - C_i|$.

AdaECE: One disadvantage of ECE is the uniform bin width. For a trained model, most of the samples lie within the highest confidence bins, and hence these bins dominate the value of the ECE. We thus also consider another metric, AdaECE (Adaptive ECE), for which bin sizes are calculated so as to evenly distribute samples between bins: $\text{AdaECE} = \sum_{i=1}^M \frac{|B_i|}{N}\,|A_i - C_i|$ s.t. $\forall i, j \cdot |B_i| = |B_j|$.

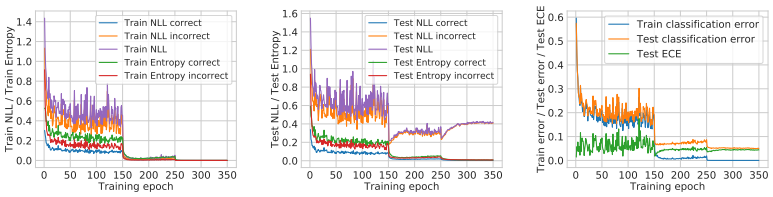
Classwise-ECE: The ECE only considers the probability of the predicted class without considering the other scores in the softmax distribution. A stronger definition of calibration would require the probabilities of all the classes in the softmax distribution to be calibrated (Vaicenavicius et al., 2019; Kumar et al., 2019). This can be achieved with a simple classwise extension of the ECE metric: $\text{ClasswiseECE} = \frac{1}{K}\sum_{i=1}^M\sum_{j=1}^K \frac{|B_{i,j}|}{N}\,|A_{i,j} - C_{i,j}|$, where K is the number of classes, B_{ij} denotes the set of samples from jth class in the ith bin, $A_{ij} = \frac{1}{|B_{ij}|}\sum_{k\in B_{ij}} \mathbb{1}\left(j = y_k\right)$ and $C_{i,j} = \frac{1}{|B_{ij}|}\sum_{k\in B_{ij}} \hat{p}_{kj}$.

3. Miscalibration and Focal Loss

A key empirical observation on miscalibration made by (Guo et al., 2017) was that poor calibration of such networks appears to be linked to overfitting on the negative log-likelihood (NLL) during training. We further inspect this observation here. For our analysis, we train a ResNet-50 on CIFAR-10 (getting state-of-the-art test set accuracy, all the training settings can be found here: (PyTorch-CIFAR)). We minimise cross-entropy (a.k.a. NLL) $\mathcal{L}_c = -\log \hat{p}_{i,y_i}$, where $\hat{p}_{i,y_i}$ is the probability assigned to the correct class y_i for the ith sample. Note that the NLL is minimised when for each training sample i, $\hat{p}_{i,y_i} = 1$, whereas the classification error is minimised when $\hat{p}_{i,y_i} > \hat{p}_{i,y}$ for all $y \neq y_i$. Thus, even when the classification error is 0, the NLL can be positive, and the optimiser can still try to reduce it to 0 by further increasing the value of $\hat{p}_{i,y_i}$ for each sample.

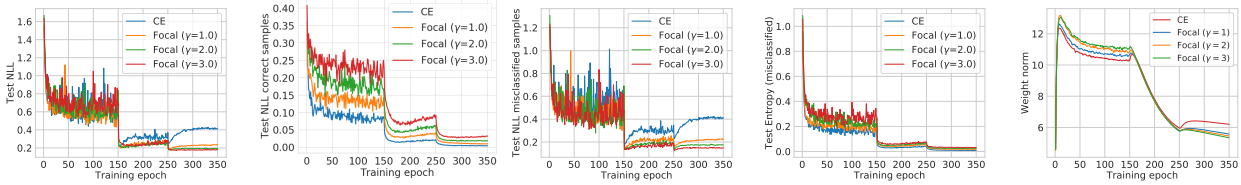
Peak at the wrong place: We plot the average train and test NLL and entropy of the softmax distribution at each training epoch in Figures 1(a) and 1(b). In Figure 1(c), we plot the train and test classification error and the test set ECE. Figures 1(a) and 1(b) show that although the average train NLL broadly decreases throughout training, after the 150^{th} epoch (where the learning rate drops by a factor of 10), there is a marked rise in the average test NLL, indicating that the network starts to overfit on average NLL. This increase in average test NLL is caused only by the incorrectly classified samples. Note that after epoch 150, the test set ECE also rises, indicating that the network is becoming miscalibrated. Moreover, the softmax entropies for both correctly and incorrectly classified test samples decrease throughout training (in other words, the distributions get peakier). This indicates that *for the misclassified test samples, the network gradually becomes more and more confident on its incorrect predictions*.

Weight magnification: The increase in confidence of the network's predictions can happen if the network increases the norm of its weights W to increase the magnitudes of the logits. In fact, cross-entropy loss is minimised when for each training sample i, $\hat{p}_{i,y_i} = 1$, which is possible only when $||W|| \to \infty$. Cross-entropy loss thus inherently induces this tendency of weight magnification in neural network



(a) (b) (c)

Figure 1. How metrics related to model calibration change whilst training a ResNet-50 network on CIFAR-10.



(a) (b) (c) (d) (e)

Figure 2. How metrics related to model calibration change whilst training several ResNet-50 networks on CIFAR-10, using either cross-entropy loss, or focal loss with γ set to 1, 2 or 3.

optimisation. The promising performance of weight decay (regulating the norm of weights) on the calibration of neural networks can perhaps be explained using this observation.

Why might focal loss improve calibration? In this work, we study an alternative loss function, popularly known as *focal loss* (Lin et al., 2017), that weights loss components generated from individual samples in a mini-batch by how well the model classifies them. For classification tasks where the target distribution is a one-hot encoding, it is defined as $\mathcal{L}_f = -(1-\hat{p}_{i,y_i})^\gamma \log \hat{p}_{i,y_i}$, where γ is a user-defined hyperparameter. We know that cross-entropy forms an upper bound on the KL-divergence between the target distribution q and the predicted distribution $\hat{p}$, i.e. $\mathcal{L}_c \geq \mathrm{KL}(q||\hat{p})$, so minimising cross-entropy results in minimising $\mathrm{KL}(q||\hat{p})$. Interestingly, a general form of focal loss can be shown to be an upper bound on the regularised KL-divergence, where the regulariser is the negative entropy $\mathbb{H}[\hat{p}]$ of the predicted distribution $\hat{p}$, and the regularisation parameter is γ: $\mathcal{L}_f \geq \mathrm{KL}(q||\hat{p}) - \gamma\mathbb{H}[\hat{p}]$ (proof in Appendix A).

This shows that replacing cross-entropy with focal loss has the effect of adding a maximum-entropy regulariser (Pereyra et al., 2017). Encouraging the predicted distribution to have higher entropy can help avoid the overconfident predictions produced by modern neural networks (see the 'Peak at the wrong place' above), and thereby improve calibration.

Empirical observations on focal loss: To compare focal loss with cross-entropy, we use the same training settings as mentioned above, and train four ResNet-50 networks on CIFAR-10, one using cross-entropy loss, and three using focal loss with $\gamma = 1, 2$ and 3. Figure 2(a) shows that while the test NLL for the cross-entropy model significantly increases towards the end of training (before saturating), the test NLLs for the focal loss models remain low. This is further corroborated in Figures 2(b) and 2(c). Figure 2(b) shows that the test NLLs on correctly classified samples for the focal loss models remain consistently higher than that for the cross-entropy model throughout training, implying that the focal loss models are relatively less confident than the cross-entropy model for these samples. This is important, as we have already discussed that it is overconfidence that normally makes deep neural networks miscalibrated. In fact, Figure 2(c) shows that in contrast to the cross-entropy model, the NLL for misclassified test samples is significantly lower for focal loss models after epoch 150 (where we observe NLL overfitting). Additionally, in Figure 2(d), we notice that the entropy of the softmax distribution for misclassified test samples is consistently (if marginally) higher for focal loss than for cross-entropy.

We had hypothesised earlier that the overconfidence of cross-entropy models is caused by an increase in the weight norms of the network over the course of training. In Figure 2 (e), we plot the weight norms of the last linear layer for all four models against training epochs. Although focal loss models have higher initial weight norms, there is a complete reversal of the order of weight norms after epoch 150, i.e., the point from where the networks start getting miscalibrated. This observation is encouragingly in favour of focal loss. In the next section, we empirically evaluate the performance of focal loss on multiple datasets and network architectures.

4. Experiments

We conduct image and document classification experiments to test the performance of focal loss. For the former, we use CIFAR-10/100 (Krizhevsky, 2009) and Tiny-ImageNet (Deng et al., 2009), and train ResNet-50, ResNet-110 (He et al., 2016), Wide-ResNet-26-10 (Zagoruyko & Komodakis, 2016) and DenseNet-121 (Huang et al., 2017) models, and for the latter, we use Stanford Sentiment Treebank (SST)

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		Focal Loss ($\gamma = 3$)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet 50	17.52	3.42(2.1)	6.52	3.64(1.1)	15.32	2.38(1.8)	7.81	4.01(1.1)	**5.13**	**1.97(1.1)**
	ResNet 110	19.05	4.43(2.3)	**7.88**	4.65(1.2)	19.14	**3.86(2.3)**	11.02	5.89(1.1)	8.64	3.95(1.2)
	Wide ResNet 26-10	15.33	2.88(2.2)	4.31	2.7(1.1)	13.17	4.37(1.9)	4.84	4.84(1)	**2.13**	**2.13(1.0)**
	DenseNet 121	20.98	4.27(2.3)	5.17	2.29(1.1)	19.13	3.06(2.1)	12.89	7.52(1.2)	**4.15**	**1.25(1.1)**
CIFAR-10	ResNet 50	4.35	1.35(2.5)	1.82	**1.08(1.1)**	4.56	1.19(2.6)	2.96	1.67(0.9)	**1.48**	1.42(1.1)
	ResNet 110	4.41	1.09(2.8)	2.56	1.25(1.2)	5.08	1.42(2.8)	2.09	2.09(1)	**1.55**	**1.02(1.1)**
	Wide ResNet 26-10	3.23	0.92(2.2)	**1.25**	1.25(1)	3.29	**0.86(2.2)**	4.26	1.84(0.8)	1.69	0.97(0.9)
	DenseNet 121	4.52	1.31(2.4)	1.53	1.53(1)	5.1	1.61(2.5)	1.88	1.82(0.9)	**1.32**	**1.26(0.9)**
Tiny-ImageNet	ResNet 50	15.32	5.48(1.4)	4.44	4.13(0.9)	13.01	5.55(1.3)	15.23	6.51(0.7)	**1.87**	**1.87(1)**
SST Binary	Tree LSTM	7.37	2.62(1.8)	9.01	2.79(2.5)	5.03	4.02(1.5)	**4.84**	4.11(1.2)	16.05	**1.78(0.5)**

Table 1. ECE (%) computed for different approaches both pre and post temperature scaling (cross-validating T on ECE). Optimal temperature for each method is indicated in brackets.

(Socher et al., 2013) dataset and train Tree-LSTM (Tai et al., 2015) models. Further details on the datasets and training can be found in Appendix B.

Baselines Along with cross-entropy loss, we use the following baselines: a) *MMCE* (Maximum Mean Calibration Error) (Kumar et al., 2018), a continuous and differentiable proxy for calibration error that is normally used as a regulariser alongside cross-entropy, b) *Brier loss* (Brier, 1950), the squared error between the predicted softmax vector and the one-hot ground truth encoding (Brier loss is an important baseline as it can be decomposed into calibration and refinement (DeGroot & Fienberg, 1983; Snoek et al., 2019)), c) *Label smoothing* (Müller et al., 2019) (LS): given a one-hot ground-truth distribution $\mathbf{q}$ and a smoothing factor α (hyperparameter), the smoothed vector $\mathbf{s}$ is obtained as $\mathbf{s}_i = (1-\alpha)\mathbf{q}_i + \alpha(1-\mathbf{q}_i)/(K-1)$, where $\mathbf{s}_i$ and $\mathbf{q}_i$ denote the i^{th} elements of $\mathbf{s}$ and $\mathbf{q}$ respectively, and K is the number of classes. Instead of $\mathbf{q}$, $\mathbf{s}$ is treated as the ground truth. We train models using $\alpha = 0.05$ and $\alpha = 0.1$, but find $\alpha = 0.05$ to perform better, and (d) *Focal Loss (fixed γ)*: We trained models using focal loss setting γ to 1,2 and 3 and found $\gamma = 3$ to perform the best.

Performance Gains: We report the optimal temperatures and their corresponding ECE% (computed using 15 bins) in Table 1. We also report test set classification error, AdaECE, and Classwise-ECE in Appendix C. Firstly, for all dataset-network pairs, we obtain very competitive test set classification accuracies (shown in Table C.3 in the appendix). *Secondly, it is clear from Tables 1, C.1 and C.2 that focal loss with $\gamma = 3$ generally outperforms all the other baselines.* It broadly produces the lowest ECE, AdaECE and Classwise-ECE scores *both before and after temperature scaling*. Finally, note that for focal loss, the optimal temperatures are generally (with some exceptions) very close to 1, mostly lying between 0.9 and 1.1. This property is shown by the Brier score and the label smoothing models as well. By contrast, the optimal temperatures for the baselines (cross-entropy with hard targets and MMCE) are significantly higher, with values lying between 2.0 to 2.8. An optimal temperature close to 1 indicates that the model is innately calibrated and cannot be made significantly more calibrated by temperature scaling. Furthermore, an optimal temperature that is much greater than 1 can make the network underconfident in general, as its outputs are temperature-scaled irrespective of their correctness.

5. Conclusion

In this paper, we study an alternative loss function, focal loss (Lin et al., 2017) and observe that training using focal loss can yield multi-class classification networks that are more naturally calibrated than those trained using the more conventional cross-entropy loss. We also show in §3 that focal loss implicitly maximises entropy while minimising the KL divergence between the predicted and the target distributions. Futhermore, we empirically observe that the NLL overfitting phenomenon is significantly reduced in case of focal loss as compared to cross-entropy and hence, it can be expected to produce more calibrated models. Finally, we conduct extensive experiments on a variety of datasets and network architectures, and observe that this expectation is also borne out in practice. Our results show that in almost all cases, networks trained with focal loss are more calibrated than those trained with cross-entropy loss, label smoothing, Brier score and MMCE, whilst having similar levels of accuracy, making their predictions much easier for downstream components to trust.

6. Acknowledgement

This work was started whilst J Mukhoti was at FiveAI, and completed after he moved to the University of Oxford. V Kulharia is wholly funded by a Toyota Research Institute grant. A Sanyal acknowledges support from The Alan Turing Institute under the Turing Doctoral Studentship grant TU/C/000023. This work was also supported by ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1, EPSRC/MURI grant EP/N019474/1, and the Royal Academy of Engineering.

References

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 1950.

Cao, C., Liu, F., Tan, H., Song, D., Shu, W., Li, W., Zhou, Y., Bo, X., and Xie, Z. Deep Learning and Its Applications in Biomedicine. *Genomics, Proteomics & Bioinformatics*, 16(1):17–32, 2018.

Chen, H., Engkvist, O., Wang, Y., Olivecrona, M., and Blaschke, T. The rise of deep learning in drug discovery. *Drug Discovery Today*, 23(36):1241–1250, 2018.

DeGroot, M. H. and Fienberg, S. E. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 1983.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On Calibration of Modern Neural Networks. In *ICML*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

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

Kamilaris, A. and Prenafeta-Boldú, F. X. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018.

Ker, J., Wang, L., Rao, J., and Lim, T. Deep Learning Applications in Medical Image Analysis. *IEEE Access*, 6:9375–9389, 2018.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Kumar, A., Sarawagi, S., and Jain, U. Trainable Calibration Measures For Neural Networks From Kernel Mean Embeddings. In *ICML*, 2018.

Kumar, A., Liang, P. S., and Ma, T. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, pp. 3787–3798, 2019.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal Loss for Dense Object Detection. In *ICCV*, 2017.

Müller, R., Kornblith, S., and Hinton, G. E. When does label smoothing help? In *Advances in Neural Information Processing Systems*, pp. 4696–4705, 2019.

Naeini, M. P. and Cooper, G. F. Binary Classifier Calibration using an Ensemble of Near Isotonic Regression Models. In *ICDM*, 2016.

Naeini, M. P., Cooper, G. F., and Hauskrecht, M. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In *AAAI*, 2015.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., and Hinton, G. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Platt, J. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. *Advances in Large Margin Classifiers*, 10(3):61–74, 1999.

PyTorch-CIFAR. Train CIFAR10 with PyTorch, 2019. Available online at https://github.com/kuangliu/pytorch-cifar (as of 22nd May 2019).

Snoek, J., Ovadia, Y., Fertig, E., Lakshminarayanan, B., Nowozin, S., Sculley, D., Dillon, J., Ren, J., and Nado, Z. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pp. 13969–13980, 2019.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

Tai, K. S., Socher, R., and Manning, C. D. Improved semantic representations from tree-structured long short-term memory networks. In *Association for Computational Linguistics (ACL)*, 2015.

TreeLSTM. TreeLSTM. https://github.com/ttpro1995/TreeLSTMSentiment/, 2015. Accessed: 2019-05-22.

Vaicenavicius, J., Widmann, D., Andersson, C., Lindsten, F., Roll, J., and Schön, T. B. Evaluating model calibration in classification. In *AISTATS*, 2019.

Wang, J., Ma, Y., Zhang, L., Gao, R. X., and Wu, D. Deep learning for smart manufacturing: Methods and applications. *JMSY*, 48:144–156, 2018.

Zadrozny, B. and Elkan, C. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *ICML*, 2001.

Zadrozny, B. and Elkan, C. Transforming Classifier Scoes into Accurate Multiclass Probability Estimates. In *SIGKDD*, 2002.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

Appendix

A. Relation between Focal Loss and Entropy Regularised KL Divergence

Here we show why focal loss favours accurate but relatively less confident solutions. We show that it inherently provides a trade-off between minimizing the KL-divergence and maximizing the entropy, depending on the strength of γ. We use $\mathcal{L}_f$ and $\mathcal{L}_c$ to denote the focal loss with parameter γ and cross entropy between $\hat{p}$ and q, respectively. K denotes the number of classes and q_y denotes the ground-truth probability assigned to the y-th class (similarly for $\hat{p}_y$). We consider the following simple extension of focal loss:

$$
\begin{aligned}
\mathcal{L}_f &= -\sum_{y=1}^{K}(1-\hat{p}_y)^{\gamma} q_y \log \hat{p}_y \\
&\geq -\sum_{y=1}^{K}(1-\gamma\hat{p}_y) q_y \log \hat{p}_y && \text{By Bernoulli's inequality } \forall\gamma \geq 1. \text{ Note, } \hat{p}_y \in [0,1] \\
&= -\sum_{y=1}^{K} q_y \log \hat{p}_y - \gamma \left| \sum_{y=1}^{K} q_y \hat{p}_y \log \hat{p}_y \right| && \forall y, \log \hat{p}_y \leq 0 \\
&\geq -\sum_{y=1}^{K} q_y \log \hat{p}_y - \gamma \max_j q_j \sum_{y=1}^{K} |\hat{p}_y \log \hat{p}_y| && \text{By Hölder's inequality } ||fg||_1 \leq ||f||_\infty ||g||_1 \\
&\geq -\sum_{y=1}^{K} q_y \log \hat{p}_y + \gamma \sum_{y=1}^{K} \hat{p}_y \log \hat{p}_y && \forall j, q_j \in [0,1] \\
&= \mathcal{L}_c - \gamma \mathbb{H}[\hat{p}].
\end{aligned}
$$

We know that $\mathcal{L}_c = \mathrm{KL}(q||\hat{p}) + \mathbb{H}[q]$, thus, combining this equality with the above inequality leads to:

$$
\mathcal{L}_f \geq \mathrm{KL}(q||\hat{p}) + \underbrace{\mathbb{H}[q]}_{constant} - \gamma \mathbb{H}[\hat{p}].
$$

In the case of one-hot encoding (Delta distribution for q), focal loss would maximize $-\hat{p}_y \log \hat{p}_y$ (let y be the ground-truth class index), the component of the entropy of $\hat{p}$ corresponding to the ground-truth index. Thus, would prefer learning $\hat{p}$ such that $\hat{p}_y$ is assigned a higher value (because of the KL term) but not too high (because of the entropy term) which eventually would avoid preferring overconfident models (as opposed to the cross-entropy loss). Experimentally, we found the solution of the cross entropy and focal loss equations, i.e. the value of the predicted probability $\hat{p}$ which minimizes the loss, for various values of q in a binary classification problem (i.e. $K = 2$) and plotted it in Figure A.1. As expected, focal loss favours a more entropic solution $\hat{p}$ that is closer to 0.5. In other words, as Figure A.1 shows, solutions to focal loss (Eqn 1) will always have higher entropy than that of cross entropy depending on the value of γ.

$$
\hat{p} = \mathrm{argmin}_x \; -(1-x)^{\gamma} q \log x - x^{\gamma}(1-q)\log(1-x) \quad 0 \leq x \leq 1 \qquad (1)
$$

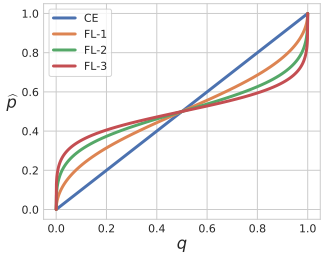


Figure A.1. Optimal $\hat{p}$ for various values of q.

B. Dataset Description and Implementation Details

We use the following image and document classification datasets in our experiments:

1. **CIFAR-10** (Krizhevsky, 2009): This dataset has 60,000 colour images of size 32×32, divided equally into 10 classes. We use a train/validation/test split of 45,000/5,000/10,000 images.

2. **CIFAR-100** (Krizhevsky, 2009): This dataset has 60,000 colour images of size 32×32, divided equally into 100 classes. (Note that the images in this dataset are not the same images as in CIFAR-10.) We again use a train/validation/test split of 45,000/5,000/10,000 images.

3. **Tiny-ImageNet** (Deng et al., 2009): Tiny-ImageNet is a subset of ImageNet with 64 x 64 dimensional images, 200 classes and 500 images per class in the training set and 50 images per class in the validation set. The image dimensions of Tiny-ImageNet are twice that of CIFAR-10/100 images.

4. **Stanford Sentiment Treebank (SST)** (Socher et al., 2013): This dataset contains movie reviews in the form of sentence parse trees, where each node is annotated by sentiment. We use the dataset version with binary labels, for which 6,920/872/1,821 documents are used as the training/validation/test split. In the training set, each node of a parse tree is annotated as positive, neutral or negative. At test time, the evaluation is done based on the model classification at the root node, i.e. considering the whole sentence, which contains only positive or negative sentiment.

For training networks on 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. Furthermore, we augment the training images by applying random crops and random horizontal flips. For Tiny-ImageNet, we train 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. It should be noted that for Tiny-ImageNet, we saved 50 samples per class (i.e., a total of 10000 samples) from the training set as our own validation set to fine-tune the temperature parameter (hence, we trained on 90000 images) and we use the Tiny-ImageNet validation set as our test set.

For the SST Binary dataset, we train the Tree-LSTM (Tai et al., 2015) using the AdaGrad optimiser with a learning rate of 0.05 and a weight decay of 10^{-4}, as suggested by the authors. We used the constituency model, which considers binary parse trees of the data and trains a binary Tree-LSTM on them. The Glove word embeddings (Pennington et al., 2014) were also tuned for best results. The code framework we used is inspired by (TreeLSTM). We trained these models for 25 epochs and used the models with the best validation accuracy.

For all our models, we use the PyTorch framework, setting any hyperparameters not explicitly mentioned to the default values used in the standard models. For MMCE, we used $\lambda = 2$ for all our experiments as we found it to perform better over all the values we tried. A calibrated model which does not generalise well to an unseen test set is not very useful. Hence, for all the experiments, we set the training parameters in a way such that we get best test set accuracies on all datasets for each model.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		Focal Loss ($\gamma = 3$)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	17.52	3.42(2.1)	6.52	3.64(1.1)	15.32	2.38(1.8)	7.81	4.01(1.1)	**5.08**	**2.02(1.1)**
	ResNet-110	19.05	5.86(2.3)	**7.73**	4.53(1.2)	19.14	4.85(2.3)	11.12	8.59(1.1)	8.64	**4.14(1.2)**
	Wide-ResNet-26-10	15.33	2.89(2.2)	4.22	2.81(1.1)	13.16	4.25(1.9)	5.1	5.1(1)	**2.08**	**2.08(1.0)**
	DenseNet-121	20.98	5.09(2.3)	5.04	2.56(1.1)	19.13	3.07(2.1)	12.83	8.92(1.2)	**4.15**	**1.23(1.1)**
CIFAR-10	ResNet-50	4.33	2.14(2.5)	**1.74**	**1.23(1.1)**	4.55	2.16(2.6)	3.89	2.92(0.9)	1.95	1.83(1.1)
	ResNet-110	4.4	1.99(2.8)	2.6	1.7(1.2)	5.06	2.52(2.8)	4.44	4.44(1)	**1.62**	**1.44(1.1)**
	Wide-ResNet-26-10	3.23	1.69(2.2)	**1.7**	1.7(1)	3.29	1.6(2.2)	4.27	2.44(0.8)	1.84	**1.54(0.9)**
	DenseNet-121	4.51	2.13(2.4)	2.03	2.03(1)	5.1	2.29(2.5)	4.42	3.33(0.9)	**1.22**	**1.48(0.9)**
Tiny-ImageNet	ResNet-50	15.23	5.41(1.4)	4.37	4.07(0.9)	13.0	5.56(1.3)	15.28	6.29(0.7)	**1.88**	**1.88(1)**
SST Binary	Tree-LSTM	7.27	3.39(1.8)	8.12	2.84(2.5)	**5.01**	4.32(1.5)	5.14	4.23(1.2)	16.01	**2.16(0.5)**

Table C.1. Adaptive ECE (%) computed for different approaches both pre and post temperature scaling (cross-validating T on ECE). Optimal temperature for each method is indicated in brackets.

Dataset	Model	Cross-Entropy		Brier Loss		MMCE		LS-0.05		Focal Loss ($\gamma = 3$)	
		Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T	Pre T	Post T
CIFAR-100	ResNet-50	0.38	0.22(2.1)	0.22	0.20(1.1)	0.34	0.21(1.8)	0.23	0.21(1.1)	**0.20**	**0.20(1.1)**
	ResNet-110	0.41	0.21(2.3)	0.24	0.23(1.2)	0.42	0.22(2.3)	0.26	0.22(1.1)	**0.24**	**0.22(1.2)**
	Wide-ResNet-26-10	0.34	0.20(2.2)	**0.19**	0.19(1.1)	0.31	0.20(1.9)	0.21	0.21(1)	0.21	**0.18(1.0)**
	DenseNet-121	0.45	0.23(2.3)	0.20	0.21(1.1)	0.42	0.24(2.1)	0.29	0.24(1.2)	**0.20**	**0.20(1.1)**
CIFAR-10	ResNet-50	0.91	0.45(2.5)	0.46	**0.42(1.1)**	0.94	0.52(2.6)	0.71	0.51(0.9)	**0.43**	0.48(1.1)
	ResNet-110	0.91	0.50(2.8)	0.59	0.50(1.2)	1.04	0.55(2.8)	0.66	0.66(1)	**0.44**	**0.41(1.1)**
	Wide-ResNet-26-10	0.68	0.37(2.2)	0.44	0.44(1)	0.70	**0.35(2.2)**	0.80	0.45(0.8)	**0.44**	0.36(0.9)
	DenseNet-121	0.92	0.47(2.4)	0.46	0.46(1)	1.04	0.57(2.5)	0.60	0.50(0.9)	**0.43**	**0.41(0.9)**
Tiny-ImageNet	ResNet-50	0.22	0.16(1.4)	0.16	0.16(0.9)	0.21	0.16(1.3)	0.21	0.17(0.7)	**0.16**	**0.16(1)**
SST Binary	Tree-LSTM	5.81	3.76(1.8)	6.38	**2.48(2.5)**	**3.82**	2.70(1.5)	3.99	3.20(1.2)	6.35	2.81(0.5)

Table C.2. Classwise-ECE (%) computed for different approaches both pre and post temperature scaling (cross-validating T on ECE). Optimal temperature for each method is indicated in brackets.

Dataset	Model	Cross-Entropy	Brier Loss	MMCE	LS-0.05	Focal Loss ($\gamma = 3$)
CIFAR-100	ResNet-50	23.3	23.39	23.2	23.43	22.75
	ResNet-110	22.73	25.1	23.07	23.43	22.92
	Wide-ResNet-26-10	20.7	20.59	20.73	21.19	19.69
	DenseNet-121	24.52	23.75	24.0	24.05	23.25
CIFAR-10	ResNet-50	4.95	5.0	4.99	5.29	5.25
	ResNet-110	4.89	5.48	5.4	5.52	5.08
	Wide-ResNet-26-10	3.86	4.08	3.91	4.2	4.13
	DenseNet-121	5.0	5.11	5.41	5.09	5.33
Tiny-ImageNet	ResNet-50	49.81	53.2	51.31	47.12	49.69
SST Binary	Tree-LSTM	12.85	12.85	11.86	13.23	12.19

Table C.3. Error (%) computed for different approaches.

C. Additional Results

In Tables C.1, C.2 and C.3 we report Adaptive ECE (%), Classwise-ECE (%) and test set classification error computed for different approaches both pre and post temperature scaling. The optimal temperatures are indicated in brackets in Tables C.1 and C.2.