Improving Image Classifiers with IQ Loss functions: Non-adversarial f-Divergence Minimization

Div Garg
Stanford University
divgarg@stanford.edu

Avi Gupta
Stanford University
avigupta@stanford.edu

Roy Yuan
Stanford University
ryuan19@stanford.edu

Abstract

*Image classification is an important and broadly researched topic in computer vision, and to perform this task, classifiers utilize loss functions to distinguish the models predictions and the ground-truth labels. These loss functions guide the training of the classifier. However, concerns have emerged regarding the miscalibration of deep neural networks for multi-class classification tasks. Particularly worrisome is the misestimation of class probabilities that manifests as overconfidence in the model's final prediction, which has significant downstream effects. Moreover, existing loss functions lack robustness to distributional shifts. Existing literature suggests that viewing classifiers as energy-based models can improve classification by providing superior calibration and robustness. In this paper, we present **IQ Loss**, a loss function that can minimize f-divergences while avoiding adversarial training and thereby achieve superior calibration and robustness to distributional shifts to existing loss functions, including focal loss and cross-entropy. We empirically validate this approach by training standard computer vision architectures using IQ Loss on the CIFAR-10 and the CIFAR-100 datasets. We then assess the performance and calibration of these models and demonstrate that IQ Loss outperforms both focal loss and cross entropy on these vital classification tasks.*

1. Introduction

In this project, we seek to improve the robustness and calibration of image classification models through the development and emperical testing of better loss functions. Existing loss functions, which include cross-entropy, are not very robust across problem spaces. Moreover, many deep learning models for multi-class tasks exhibit poor calibration, which manifests as overestimation of class labels in their predictions. This miscalibration is particularly pronounced in modern neural networks [6], Because models tend to be overly confident in their predictions, they exhibit a systematic error that can affect any downstream components that rely on the probabilities assigned to a given class [11]. One hypothesis attributes the overconfidence of multi-class classification networks to the negative log-likelihood loss function commonly used to train deep neural networks. A similar study demonstrated that models with high capacity can overfit to the loss function even if there is no evidence of overfitting in the classification results. In other words, the overfitting of models produces errors not in the classification task itself, but in the probabilistic estimates produced by the model.

In another vein, a large body of research has sought to bring the potential of generative models to bear on discriminator tasks, including classification [2]. Among the most promising generative approaches for this task is energy-based modeling, which is more closely aligned with modern classifier architectures than other generative techniques. Moreover, applications of energy-based modeling to discriminative problems have shown promise, with one work finding ”incorporation of generative modeling gives our models improved calibration, out-of-distribution detection, and adversarial robustness, performing on par with or better than hand-tailored methods for multiple tasks” [5].

Therefore, we propose leveraging a novel Regularized Energy Model (REM)-based loss function in order to achieve enhanced robustness and calibration for image classification tasks. Our key technique is to apply the idea of treating a classifier as though it were an energy-based model, with the logits of the classifier being the energy for each class label. We employ a new objective to train energy-based models that has favorable theoretical properties and can minimize different divergences such as Jensen-Shannon and reverse KL. Our method is based on a new loss formulation for tractably calculating f-divergences. Models such as GANs trained with different divergences have been shown to perform better in image generation and related tasks compared to using forward-KL divergence, i.e. cross-entropy losses. We compare a model trained using our loss function to other loss functions that are similarly designed to address the miscalibration of deep neural networks. We perform this comparison on a common computer vision model architecture and datasets and show that we achieve comparable or improved results on a host of metrics.

2. Related Work

The first category of literature we engaged with was the study of improved loss functions that address the lack of calibration in deep neural networks. Miscalibration, in this case, is defined as “a mismatch between a model’s confidence and its correctness”. The primary paper we consulted in this category was the Focal Loss paper from researchers at University of Oxford [11]. This paper presents focal loss, a loss function that the authors show reduces miscalibration while preserving accuracy across a variety of problem spaces, including computer vision. Moreover, they apply their loss function to temperature-based models to achieve strong performance. The authors demonstrate the empirical improvement of focal loss training of ResNet and other common model architectures using their loss function on CIFAR-10 and other computer vision datasets. The authors also analyze the causes of miscalibration and provide empirical data. This inspired the development of our loss function. Moreover, the results presented in the focal loss paper provide a useful point of comparison for our loss function, and we also draw on a modified version of their codebase to assess the calibration of models trained using our loss function on computer vision tasks compared to focal loss. We also drew on several other papers on miscalibration of deep neural networks in order to better understand the problem. One paper we reviewed is this 2017 paper, which proposes temperature scaling as a technique to address miscalibration [6]. Lastly, we studied this paper, which suggests that certain model architectures are more likely to become miscalibrated and that the issue is particularly pronounced in highly accurate models [10].

The second category of papers we drew upon focused on understanding energy-based models. These articles gave us the background we needed to understand energy-based models since a key idea of our project is to treat a classifier like an energy-based model and apply appropriate loss functions. The most helpful paper in this vein was this “tutorial” from Yann LeCun (and others) from the Courant Institute at New York University [8]. We also reviewed another paper from 2006 that provided an example of how to use energy-based learning to effectively generate sparse features on the MNIST data set [15]. Energy-based models have also been applied to generative adversarial nets (GANs) by viewing the discriminator as an energy-based model [18]. This relates to our work not just in terms of providing an example of an application of energy-based learning, but also because it treats a classifier as an energy-based model.

The third category of papers explained how to represent classification as an energy-based approach. The primary paper we consulted for this was from Google and the University of Toronto [5]. Specifically, this paper tries to solve a similar problem of miscalibration to the focal loss paper. We are also trying to address this issue through energy-based methods. Other authors have employed this energy-based approach to perform useful tasks such as identifying inputs that are out of distribution [9], improving GANs without additional training [3] and providing general tips for training energy-based models [16]. Because our work focused on developing a loss function that treats a classification as a task for which we can employ an energy-based model, drawing on a wide variety of literature helped us to better refine our approach and avoid common pitfalls.

3. Methods

The key idea behind our approach is interpreting a classifier as an energy-based model, where the logits of the classifier act as the energy for each class label. Based on this interpretation, we can apply novel loss functions that minimize different f-divergences via non-adversarial training using Regularized Energy-based Models (REMs) [4]. Then we can use the methodology from the REM paper to minimize different divergences using *IQ Losses* to train our classifier. The theory is based on the REM paper (which is an ongoing research project in the Ermon Lab) and we empirically apply it to image classification tasks. Originally, IQ Loss was proposed for sequential-decision making by IQ-Learn [4] and subsequently extended to general generative modeling by REM. In this work, we apply this theory for the first-time to image classification tasks and empirically demonstrate improvements on existing methods.

3.1. REMs

REMs are energy-based models trained using likelihood maximization with an extra regularization on the energy. The choice of the energy regularizer is equivalent to learning densities by minimizing a wide range of statistical distances such as f-divergences and Integral Probability Metrics (IPMs). More generally, the choice of the statistical distance corresponds to different priors placed on the energy and can be used for systematically choosing the appropriate distance measure for a problem. This lends itself to the development of better loss functions for classification tasks. This should have more robust properties, and be given better calibration similar to focal calibration and other methods that treat classifiers as energy-based models. A direct application is in a classification where losses based on more robust distance measures can be readily formulated and optimized based on this framework. This can be useful in regards to calibration, multi-label classification, distribution shift, etc. This approach expands on ongoing work in the Ermon Lab.

3.2. IQ Loss

For a concave function $\phi : \mathcal{R}_\phi \to \mathbb{R}$, define a distance measure between a ground-truth density ρ_E and a learnt density ρ:

$$d_\phi(\rho, \rho_E) = \max_{\epsilon \in \mathcal{R}_\phi} \mathbb{E}_{\rho_E}[\phi(\epsilon)] - \mathbb{E}_\rho[\epsilon], \tag{1}$$

here $\mathcal{R}_\phi \in \mathbb{R}$ is a subset of reals, corresponding to the domain over which ϕ is defined.

Then d_ϕ includes all Integral Probability Metrics (IPMs) and f-divergences (IQ-Learn [4], Appendix B, with corresponding values in Table 4).

For deriving our IQ Loss, we minimize entropy-regularized f-divergences. Thus, our objective is given as:

$$\min_\rho J(\rho) = \min_\rho d_\phi(\rho, \rho_E) - \alpha \mathcal{H}(\rho) \tag{2}$$

where $\alpha \geq 0$ is the entropy-regularization strength. Here, entropy regularization on learnt density alleviates over-fitting on the training dataset and prevents any mode-collapse issues.

Let a convex function $\psi(\epsilon) : \mathbb{R} \to \mathbb{R} := \epsilon - \phi(\epsilon)$. Now, the REM paper shows that for the above primal objective, the dual objective is given as:

$$\begin{aligned} \max_\epsilon F(\epsilon) &= \max_\epsilon \mathbb{E}_{\rho_E}[\phi(\epsilon)] - \alpha \log Z & (3) \\ &= \max_\epsilon \mathbb{E}_{\rho_E}[\epsilon - \psi(\epsilon)] - \alpha \log Z & (4) \\ &= \max_\epsilon \alpha \mathbb{E}_{\rho_E}\left[\log \frac{1}{Z} e^{\epsilon/\alpha - \psi(\epsilon)/\alpha}\right] & (5) \end{aligned}$$

with $Z = \int_{x \in X} e^{\epsilon/\alpha}$. This is simply the maximization of the log-likelihood of the EBM $e^{\epsilon/\alpha}$ with an additional energy regularization using a convex regularizer $\psi(\epsilon)$.

This gives us our IQ loss objective, which we can use to minimize different divergence functions such as Jensen-Shannon Divergence, Reverse KL divergence, χ^2 divergence, etc.

4. Practical Algorithm

To derive our practical approach, we first note the choices of ϕ corresponding to different f-divergences in Table 1.

In our work, we will look at minimizing the Reverse KL, χ^2 and Jensen-Shannon (JS) divergences, as they have been noted to work well for the training of GANs [13], an we find them to be also perform well in our setup.

4.1. Divergence Functions

χ^2 divergence. For training using χ^2 divergence, the choice of ϕ corresponds to adding an extra energy regularizer $-\epsilon^2/4$. This extra term can be easily added to the training loss function and penalizes the logits of the classifier from becoming too peaky. This prevents the model from being over-confident, which has found to be a common issue with minimizing cross-entropy losses.

Reverse KL divergence. Similarly, we can train classifiers by minimizing the Reverse KL divergence. Here, the choice of ϕ has a similar effect, where energies with high magnitude are penalized preventing over-confident predictions of the model. In practice, we found this loss to be easily implemented as shown in the Appendix.

JS divergence. Finally, we train using JS divergence which is equivalent to training a vanilla GAN to solve the required task. Nevertheless, we find one complication in practice that the domain of energy functions is restricted in $(-\log 2, \infty)$. To implement this, we add an activation function to the logits to restrict the final outputs in the required range for training using JS divergence. We use an activation $a(x) = softplus(x) - \log(2) + \epsilon$, where the $softplus$ function provides a differentiable transform to restrict the outputs in $\mathbb{R}^+$. We also add a small constant ϵ to restrict the outputs to be greater than $-log2$, and we use a choice of $1e - 5$ in our experiments. After transforming the logits using the activation function, we can then train using the corresponding choice of ϕ.

We show the full training code for IQ-Loss using different divergences in the Appendix.

5. Datasets

5.1. CIFAR-10 and CIFAR-100

We validated our model on the CIFAR-10 dataset as loaded in the PyTorch framework. We employed a training set containing 45,000 examples, a validation set of 5000 examples, and a test set of 10,000 examples. Each image was rendered in color with a resolution of 32 x 32. The dataset had 10 classes (each of which were mutually exclusive) corresponding to common object types such as automobiles, birds, and frogs. [7]

We also validated our model on the CIFAR-100 dataset (PyTorch version), which is identical to the CIFAR-10 dataset except that it has 100 classes. The CIFAR-100 classes are grouped into 20 superclasses. The 20 superclasses include aquatic mammals, for example, which has classes beaver, dolphin, otter, seal, and whale. We split the CIFAR-100 dataset in the same manner as CIFAR-10 into 45,000 training examples, 5000 validation examples, and 10,000 test examples.

We performed data normalization on the training, validation, and test sets. On the training set, we additionally performed data augmentation consisting of random cropping and horizontal flipping. Our data was batched in groups of

Table 1. List of divergence functions, ϕ, and optimal energy estimators

Divergence	$f(t)$	$\phi(x)$	ϵ
Forward KL	$-\log t$	$1 + \log x$	$\frac{\rho_E}{\rho}$
Reverse KL	$t \log t - t + 1$	$-e^{-x}$	$\log \frac{\rho_E}{\rho}$
Squared Hellinger	$(\sqrt{t} - 1)^2$	$\frac{x}{1+x}$	$\sqrt{\frac{\rho_E}{\rho}} - 1$
Pearson χ^2	$(t - 1)^2$	$x - \frac{x^2}{4}$	$2(1 - \frac{\rho}{\rho_E})$
Total variation	$\frac{1}{2}\lvert t - 1\rvert$	x	$\frac{1}{2}\text{sign}(1 - \frac{\rho}{\rho_E})$
Jensen-Shannon	$-(t + 1)\log(\frac{t+1}{2}) + t \log t$	$\log(2 - e^{-x})$	$\log \frac{1}{2}(1 + \frac{\rho_E}{\rho})$

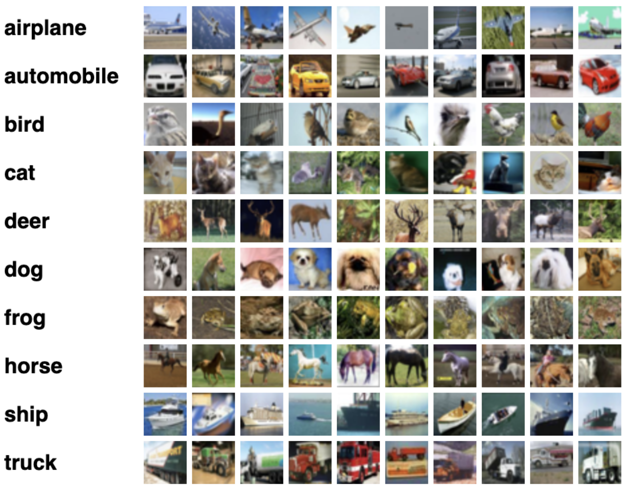

Figure 1. Ten Classes of CIFAR-10 Dataset with Ten Images from each [7]

128 for both training and testing. In future work, we would train a variety of model architectures using our loss function on a greater number of common computer vision datasets. For the purposes of the proof of concept intended in this project, our emphasis was on demonstrating the viability of our loss function on an important dataset in the computer vision literature.

Out of distribution (OOD) testing was performed using the SVHN dataset with batch sizes of 128. The SVHN data set contains 73257 training digits, 26032 digits for testing, and 531131 additional samples for use as extra training data, which are somewhat less challenging. Each image is represented by a 32 x 32 MNIST-style image. [12]

6. Experiments

In order to standardize the comparison of our loss function to focal loss and binary entropy, we leveraged the existing implementation associated with the focal loss paper. [11] This codebase provided some data and model utilities, an implementation of the focal loss and binary cross-entropy, and some visualization tools. We also modified their evaluation notebook, which performed out of distribution testing, to produce results for our model.

We trained ResNet50 models on CIFAR-10 and CIFAR-100 on the SAIL cluster using the Weights and Biases model visualization tool. [1] Our training script was modified from the focal loss codebase. The training hyperparameters are identical to the focal loss paper, with an stochastic gradient descent optimizer with momentum of 0.9, and learning rate schedule of 0.1, 0.01, 0.001 for the first 150, next 100, and last 100 epochs. [11] We varied the loss functions with which the models were trained to include cross-entropy, focal loss, and our loss function (with different divergences), and also experimented with different initial learning rates.

We also performed out of distribution testing on models trained using both focal loss and IQ-loss using a modified version of the evaluation code provided in the focal loss codebase. This had the effect of assessing the robustness of loss functions to distributional shifts.

6.1. Error Values

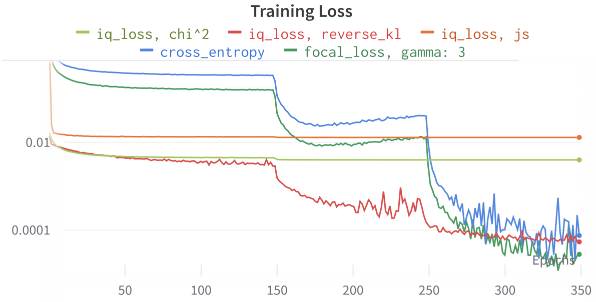

Figure 2. Training Loss vs epochs for different Losses on CIFAR-10

Figures 2, 3 and 4 depict training loss, validation loss, and validation error for several models with different loss functions (and different loss function hyperparameters, in the case of IQ Loss) on the CIFAR-10 dataset. The models were trained on the SAIL cluster and the visualization was made using the Weights&Biases visualization tool. [1]

It is important to avoid over-interpreting the unnormalized training loss values in Figure 2, since the loss functions are scaled differently and therefore raw loss values are meaningless in absolute terms. However, the change in the

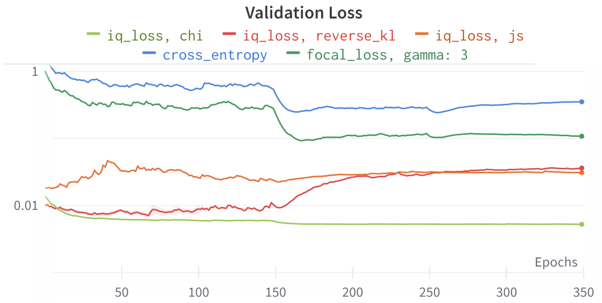

Figure 3. Validation Losses vs epochs for different Losses on CIFAR-10

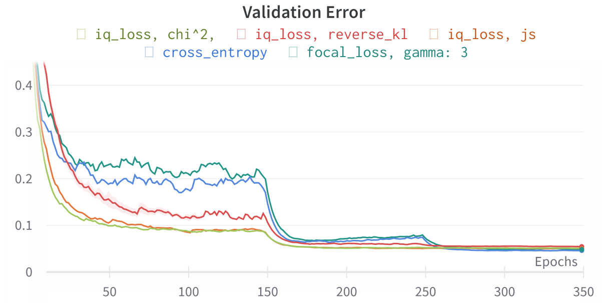

Figure 4. Validation Errors over training epochs for different Losses on CIFAR-10

loss function is reflective of the training convergence of the model. We observe that the IQ Loss models converge more rapidly (especially between epochs 150 and 250) and exhibit less noise due to overfitting (after epoch 250). In this respect, IQ Loss outperforms focal loss and cross-entropy. The sudden decreases in loss are attributable to the learning rate scheduler described earlier.

Figure 3 provides further evidence that IQ Loss is less prone to overfitting on the validation set compared to focal loss and cross-entropy. In particular, the χ^2 (chi-squared) and JS divergences with IQ loss appear less prone to overfitting.

Perhaps the most suggestive of the three graphs is Figure 4, which depicts the validation error for models trained using different loss functions. This is particularly useful because it does not suffer from the scaling issue that makes direct comparisons between loss functions impossible. These results show that models trained using IQ Loss converge more quickly than focal loss and cross-entropy. The strongest-performing IQ Loss functions are calculated with chi-squared and JS divergences, while reverse-KL appears to be less optimal for this task. However, all IQ Losses converge more rapidly and smoothly than cross-entropy and focal loss.

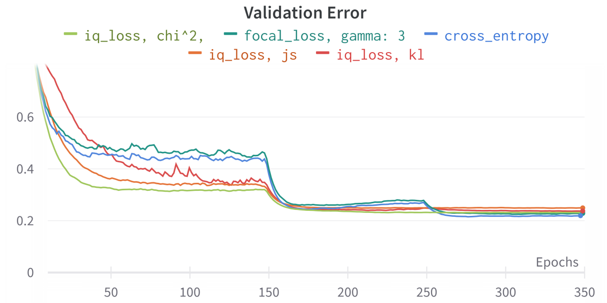

Figure 5. Validation Errors over training epochs for different Losses on CIFAR-100

Figure 5 depicts the results of training ResNe 50 using different loss functions on the CIFAR-100 dataset. As in Figure 4, which was the same experiment performed on CIFAR-10, IQ Losses substantially outperformed focal loss and cross-entropy, with JS and chi-squared performing especially well. These results demonstrate the generalization of IQ Loss-based training to another important computer vision dataset.

6.2. Out of Distribution Testing

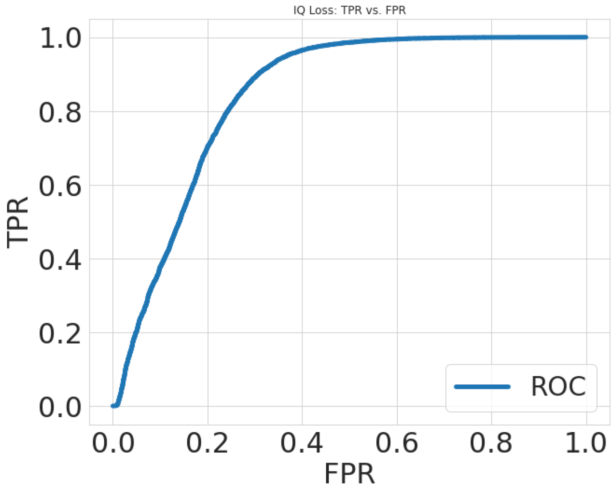

Figure 6. ROC Curves: OOD Testing Using IQ Loss

Figures 6 and 7 and Table 2 contain the results of OOD testing for IQ Loss and Focal Loss. Figures 6 and 7 depict the Receiver Operating Characteristic (ROC) curves (which plots False Positive Rates against True Positive Rates) for models trained using Focal Loss and IQ Loss. Table 2 presents Area Under the ROC (AUROC). The difference in AUROC values demonstrates the superiority of the model trained using IQ Loss compared to focal loss. A perfect AUROC value is 1, and a higher value indicates superior

Table 2. Comparison of Area Under the curve (AUROC) for IQ Loss and Focal Loss during OOD testing on Cifar10

Loss Function	AUROC
Focal Loss	0.689
IQ Loss	0.843

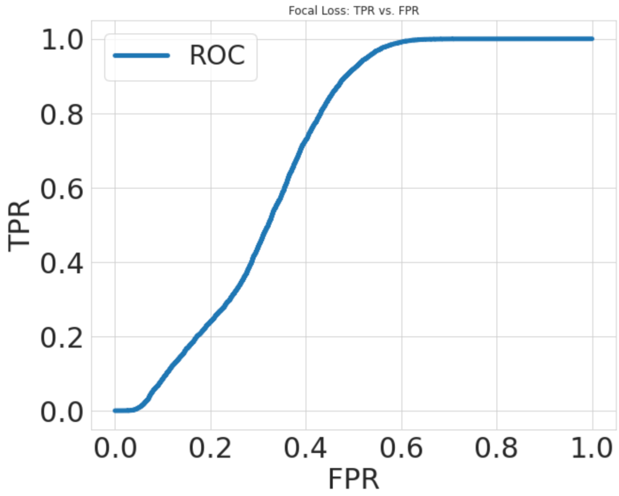

Figure 7. ROC Curves: OOD Testing Using Focal Loss

performance in robustness to anomalous inputs. [17]

7. Conclusion

In this project, we have sought to address the miscalibration of deep neural networks on classifications by applying a regularized energy-based loss function. This was inspired by recent articles that suggest the potential of applying energy-based modeling to classification tasks as well as articles expressing concern about the effects of classifier miscalibration on downstream tasks. We present the derivation of an original loss function, which we term IQ Loss and suggest some theoretical properties it holds. Training using the IQ Loss function can minimize different divergences.

We then empirically validate our proposed loss function on image classification tasks. We train models using our loss function (optimizing for different divergences) on standard computer vision datasets and architectures and show that models trained using IQ Loss achieve superior performance and robustness to distributional shifts compared to other focal loss (another loss function that seeks to address miscalibration) and the standard cross-entropy loss function.

In future work, we would seek further empirical validation of IQ Loss by applying it to an ever greater variety of image classification tasks. Although we were constrained by the scope of this project and the available resources, the preliminary results we present suggest that models trained using IQ Loss could achieve superior performance on many computer vision problems.

8. Contributions and Acknowledgements

Div Garg implemented the IQ Loss function and provided the theoretical analyses presented in the methods section. IQ Loss is the result of his research efforts and builds on his REM and IQ-Learn works. Avi Gupta performed the initial experimentation and debugged the IQ loss implementation. Avi also trained models and collected results using the evaluation framework and performed the literature review. Roy Yuan performed the bulk of the experimentation, helping to debug the IQ loss function and training and evaluating models.

The authors wish to thank David Witten, a non-CS 231N student who lended us his GitHub account when Roy's computer had issues with pushing to and pulling from our repository.

References

[1] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com. 4

[2] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. *Semi-Supervised Learning*. The MIT Press, 2006. 1

[3] Tong Che, Ruixiang Zhang, Jascha Sohl-Dickstein, Hugo Larochelle, Liam Paull, Yuan Cao, and Yoshua Bengio. Your gan is secretly an energy-based model and you should use discriminator driven latent sampling. *Advances in Neural Information Processing Systems*, 33:12275–12287, 2020. 2

[4] Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. Iq-learn: Inverse soft-q learning for imitation, 2021. 2, 3

[5] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one, 2019. 1, 2

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017. 1, 2

[7] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 3, 4

[8] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006. 2

[9] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33:21464–21475, 2020. 2

[10] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[11] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip H. S. Torr, and Puneet K. Dokania. Calibrating deep neural networks using focal loss, 2020. 1, 2, 4

[12] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 4

[13] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS*, 2016. 3

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 7

[15] Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann Cun. Efficient learning of sparse representations with an energy-based model. *Advances in neural information processing systems*, 19, 2006. 2

[16] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021. 2

[17] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey, 2021. 6

[18] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016. 2

9. Appendix

Appendix A: Implementation of Novel Loss Function

Here we present our implementation of our **IQ Loss** function using the PyTorch framework.

```
class IQLoss(nn.Module):
    def __init__(self, grad_pen=False, div="chi", alpha=0, lambda_gp=0, size_average=True):
        super(IQLoss, self).__init__()
        self.grad_pen = grad_pen
        self.div = div
        self.alpha = alpha
        self.lambda_gp = lambda_gp
        self.size_average = size_average

    def forward(self, input, target):
        if input.dim() > 2:
            input = input.view(input.size(0), input.size(1), -1)  # N,C,H,W => N,C,H*W
            input = input.transpose(1, 2)    # N,C,H*W => N,H*W,C
            input = input.contiguous().view(-1, input.size(2))   # N,H*W,C => N*H*W,C
        target = target.view(-1, 1)

        loss, loss_dict = self.iq_loss(input, target)
        return loss

    # Full IQ-Learn objective with other divergences and options
    def iq_loss(self, logits, targets):

        #  calculate 1st term for IQ loss -E_(rho_expert)[r]
        reward = logits

        # add activations to constrain domain
        if self.div == "hellinger":
            # Hellinger: min rewards = -1
            reward = F.softplus(logits) - 1 + 1e-5
        elif self.div == "js":
            # JS: min rewards = -log(2)
            reward = F.softplus(logits) - math.log(2) + 1e-5

        with torch.no_grad():
            # Use different divergence functions
            # We calculate analytic gradients of phi for stability
            # (For chi2 divergence we instead add a third squared error-like term)
            if self.div == "hellinger":
                # Hellinger
                phi_grad = 1/(1+reward)**2
            elif self.div == "kl":
                # Reverse KL
                phi_grad = torch.exp(-reward)
            elif self.div == "js":
                # Jensen-Shannon
                phi_grad = torch.exp(-reward)/(2 - torch.exp(-reward))
            else:
                phi_grad = 1
        # Gather log_prob on expert labels
        loss = -(phi_grad * reward).gather(1, targets).squeeze()
        loss_dict['softq_loss'] = loss.mean().item()

        # calculate 2nd term for IQ loss (LogZ)
        value_loss = torch.logsumexp(reward, dim=1)

        loss += value_loss
        loss_dict['value_loss'] = value_loss.mean().item()

        if self.div == "chi":
            # Chi^2 divergence
            chi2_loss = 1/4 * self.alpha * (reward**2)
            loss += chi2_loss.mean()
            loss_dict['chi2_loss'] = chi2_loss.mean().item()

        if self.grad_pen:
            # Wasserstein_1 metric: add a gradient penalty to loss
            gp_loss = utils.grad_pen(reward, self.lambda_gp)
            loss_dict['gp_loss'] = gp_loss.mean().item()
            loss += gp_loss

        loss_dict['total_loss'] = loss.mean().item()

        if self.size_average:
            loss = loss.mean()
        else:
            loss = loss.sum()
        return loss, loss_dict
```