Importance Weighting in Neural Networks

Jeewon Ha Duo Jiang Anthony Lanzisera Kuan-Lin Liu
New York University
{jh6926, dj1057, acl673, kll482}@nyu.edu

1 Introduction

In this paper, we aim to assess the influence of importance weighting on deep neural networks such as convolutional neural networks (CNN). We trained and evaluated model performance of a simple CNN and ResNet architecture on binary classification using CIFAR-10 and MNIST datasets with importance-weighted loss. Specifics of the experiments follow as designed by Byrd and Lipton (2019) [2].

Our first step was to prepare the training and test sets by pre-processing features and extracting appropriate labels. Then, each model was trained and evaluated on various configurations (l2 regularization, dropout, and batch normalization). Fractions of images classified as dogs for CIFAR-10 experiments, and 0 for MNIST experiments, were used to measure the influence of importance weighting. We can observe that for most of the configurations, after sufficient training epochs, the effect of importance weighting on the deep neural networks diminishes, and the models agree on similar decision boundary.

2 Related Work

We have realized some areas of related work.

Burda et al [1] propose an importance weighted autoencoder (IWAE), which is a variational autoencoder (VAE) augmented with importance-weighted log-likelihood lower bound. In order to satisfy the assumption that it has on the posterior distribution - posterior is approximately factorial - variational autoencoder often fails to learn expressive representations. The authors demonstrate that with strictly tighter importance-weighted log-likelihood lower bound (IWAE), the IWAE is able to learn richer latent space representation than VAEs, leading to better generative performance.

Closely related is the work by Joachims et al [3] contextual bandit with feedback and we can see importance weighting performance on ResNet for imagery datasets. They use offline policy learning to obtain better model performances. They show the prevalence of Stochastic Gradient Descent (SGD) underlying the model and ResNet can be used with data distributions successfully. We use their results of SGD as inspiration for our training functions.

Kostrikov et al [4] use off-policy evaluation to show importance weighting to have negligible effects on the training process. They show the benefits of the actor-critic method to gain leverage in learning performance. The paper seeks to reduce bias (even to unbiased) as an key objective.

3 Problem Definition and Algorithm

3.1 Task

Importance-weighted risk minimization was proven to be effective with low-capacity mis-specified models in various machine learning domains. For instance, as we discussed in class, importance weighting is able to mitigate the bias introduced by covariate shift: covariate distribution of the train is different from that of the test but the conditional distributions of them are the same. However, little has been known about its impact on deep networks [2]. Thus, we plan to assess the influence of the importance weighting on binary classification of neural networks by training the proposed models with importance-weighted loss.

3.2 Data

To train and evaluate the performance of the proposed models, we will be using the following datasets. The CIFAR-10 dataset consists of 6000 32 x 32 color images from each of the 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. These 60000 images are split into 5 training batches (each with 10000 images) and one test batch of 10000 images. While the test batch contains 1000 random images from each class, distribution of the classes may vary among training batches [5]. The MNIST dataset consists of 70000 28 x 28 gray-scale images of handwritten digits. These images are split into a training set of 60000 images and a test set of 10000 images [6].

3.3 Model

Both model architectures were implemented as described in the original paper [2]. ReLU activation functions are applied after all hidden layers for non-linear transformation.

3.3.1 CNN

The customized convolution neural network (CNN) contains two convolutional blocks and a feed-forward network. The first convolutional block has two convolution layers with 64 3 x 3 filters and stride 1, followed by a 2 x 2 max pooling layer. The second convolutional block has three convolution layers with 128 3 x 3 filters and stride 1, followed by a 2 x 2 max pooling layer. The feed-forward network has 2 dense layers, which map 512 and 128 hidden units respectively, and a fully-connected layer to output the binary result.

3.3.2 ResNet

The customized deep residual network (ResNet) contains a convolution layer with 64 5 x 5 filters and stride 1, followed by two residual blocks with 64 filters, two residual blocks with 128 filters, two residual blocks with 256 filters, an average pooling layer, a dense layer mapping 512 hidden units, and a fully-connected layer to output the binary result. Each residual block consists of a convolution layer with 128 3 x 3 filters and stride 1 and a convolution layer with 512 3 x 3 filters and stride 2.

4 Experiment

4.1 Data Preparation

We used CIFAR-10 dataset [5] to replicate the experiments present in [2] and MNIST dataset [6] to validate the conclusion drawn from prior experiments.

For CIFAR-10 dataset, the training set consists of 5000 images of dog and 5000 images of cat. There exist two test sets: one with 1000 images from dog and cat (total of 2000 images) and the other with 1000 images from the other eight classes (total of 8000 images).

For MNIST dataset, since our model architecture expects an input image with dimension of 32 x 32, we applied resize function from pytorch, which upsamples the images using bilinear interpolation. Similar to training and test sets of CIFAR-10, the MNIST training set consists of images of 0 and 1. There exist two test sets: one with images of 0 and 1 and the other with images of the rest of the eight classes.

For both datasets, images were normalized to have mean of 0.5 and standard deviation of 0.5.

4.2 Training and Evaluation

All models are trained for 1000 epochs by mini-batch SGD without momentum and a batch size of 16. We fix the learning rate to a constant value, 0.1, except for dropout models, which used a learning rate of 0.05 as suggested by [2]. During the training process, average training importance-weighted cross entropy loss was saved for every 100 iteration. For evaluation purpose, fraction of images classified as base class (dog for CIFAR-10 dataset and 0 for MNIST dataset) were computed every epoch. The importance-weight corresponds to class-conditioned weights: ratio of dog to cat for CIFAR-10 experiments and ratio of 0 to 1 for MNIST experiments.

Following are the list of configurations that we explored in this paper.

- Simple CNN and ResNet architecture
- Simple CNN with L2 regularization or dropout
- ResNet architecture without batch normalization

For CIFAR-10 experiments, importance weight was chosen among 256:1, 64:1, 16:1, 4:1, 1:1, 1:4, 1:16, 1:64, and 1:256. For MNIST experiments, importance weight was chosen among 256:1, 16:1, 1:1, 1:16, 1:256.

4.3 Results, Discussion, & Analysis

To observe the effect of various importance weighting ratios, the fraction of images classified as base class (dog for CIFAR-10 dataset and 0 for MNIST dataset) are plotted over epochs. All experiment results can be found in 1, 2, 4, 5, 6, 7, and 8. In general, we can observe that the influence of importance-weighting becomes trivial after sufficient amount of training. interestingly, for both CNN and ResNet architectures, while the fraction of the images classified as base class converges to 0.3 when the models are evaluated on out-of-domain CIFAR-10 images, it converges to 0.5 when they are evaluated on out-of-domain MNIST images.

4.4 CNN

1) Simple CNN, 2) Simple CNN with L2 regularization, and 3) Simple CNN with dropout was trained and evaluated on the datasets proposed in the previous sections. Overall, we can observe that while heavier importance weights significantly influence learned decision boundary in the early stages, after sufficient training epochs, all models, regardless of importance weights and their configurations, predicts similar decision boundary. 2 and 5 demonstrate that the models converge faster to the same decision boundary when evaluated on both test set images and out-of-domain images of MNIST dataset. On the other hand, for Simple CNN with L2 regularization, the predicted decision boundary fails to converge even after sufficient training epochs. Thus, importance weighting seems to be much more interactive with the CNN architecture with L2 regularization.

4.5 ResNet

1) Simple ResNet and 2) Simple ResNet without batch normalization was trained and evaluated on the datasets proposed in the previous sections. Similar to the results derived from CNN, we can observe that in general the influence of importance weighting diminishes after sufficient training epochs. Moreover, when evaluated on MNIST dataset, especially with test set images, importance weighting does not seem to affect the ResNet architecture and its prediction on decision boundary at all. On the other hand, when we remove batch normalization from ResNet architecture, we can observe that importance weighting starts to interact with the decision boundary for extremely heavy importance weights. After removing batch normalization in the architecture, the fractions still converge except for ratio 4:1 and 1:256 as shown in 6. Thus, importance weighting seems to be much more interactive with the ResNet architecture without batch normalization.

5 Conclusions

To examine the effect of importance weighting on model performance of deep neural networks, we conducted experiments on binary classification of the proposed models using CIFAR-10 and MNIST datasets and importance-weighted cross entropy loss. Throughout the experiments, we demonstrate that while importance weighting heavily impacts predicted decision boundary in the early stages of training, after sufficient training epochs, its influence dissipates. For most of the configurations, after 800 epochs, all models agree on similar decision boundary regardless of the importance weights. For future work, we would like to design experiments and ablation studies to closely investigate the reason for exceptional trends displayed by Simple CNN with L2 regularization and Simple ResNet without batch normalization - predicted decision boundary are critically affected by sufficiently heavy importance weight.

6 Contributions

1. Kuan-Lin Liu: Basic pipeline, training script, architecture, experiments, and github repo.
2. Duo Jiang: training script, architecture, experiments, and final write-up.
3. Jeewon Ha: architecture, experiments, and final write-up.
4. Anthony Lanzisera: architecture, experiments, and final write-up.

References

[1] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *ICLR (Poster)*, 2016.

[2] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 872–881. PMLR, 09–15 Jun 2019.

[3] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. Deep learning with logged bandit feedback. In *International Conference on Learning Representations*, 2018.

[4] Ilya Kostrikov, Kumar Agrawal, Sergey Levine, and Jonathan Tompson. Addressing sample inefficiency and reward bias in inverse reinforcement learning, 09 2018.

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

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

7 Appendix

Link to GitHub Repository: https://github.com/DuoJiang/Importance-Weighting-in-Deep-Learning

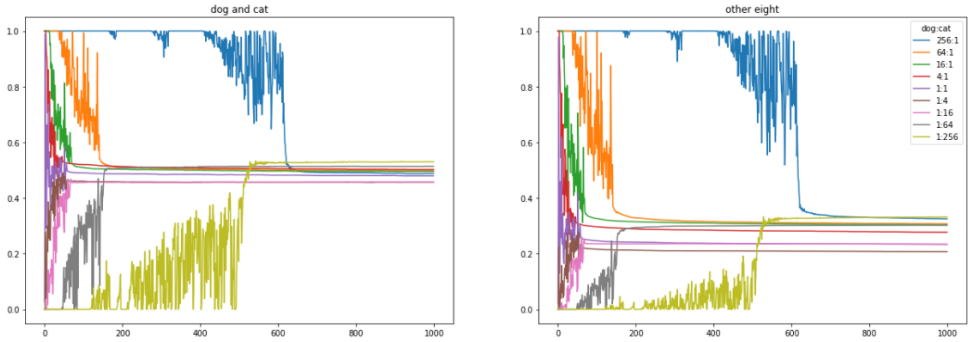


Figure 1: Simple CNN (CIFAR-10)

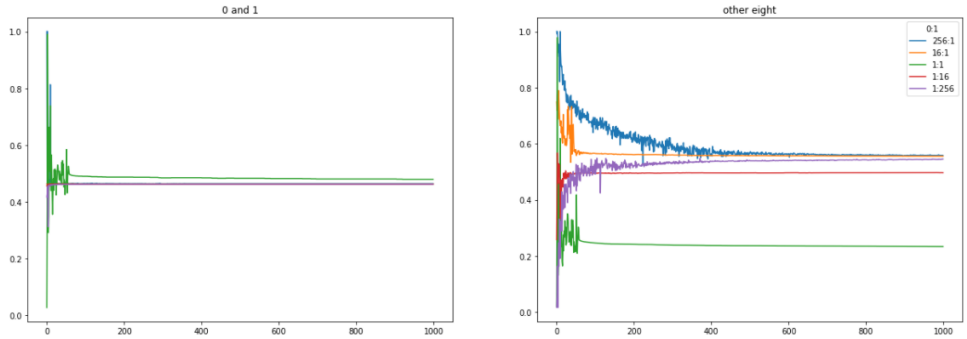
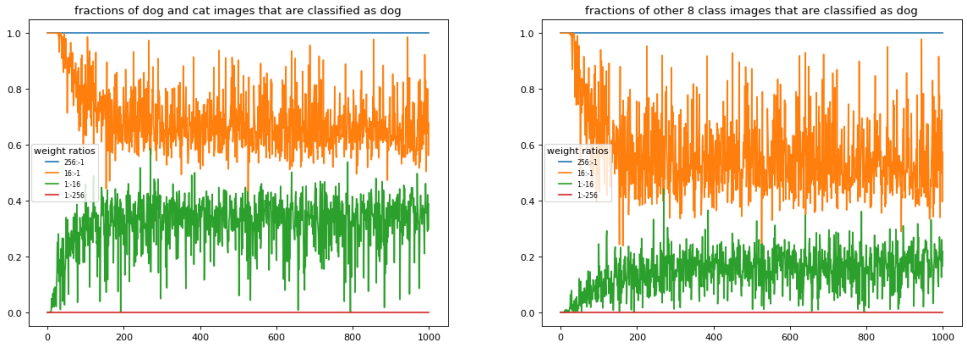
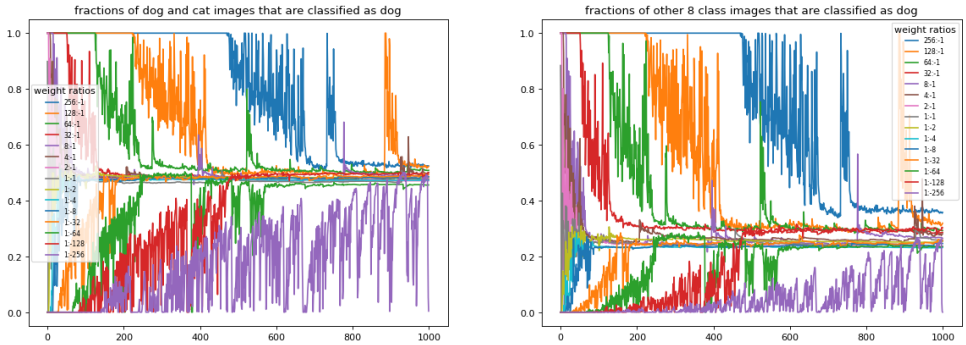


Figure 2: Simple CNN (MNIST)



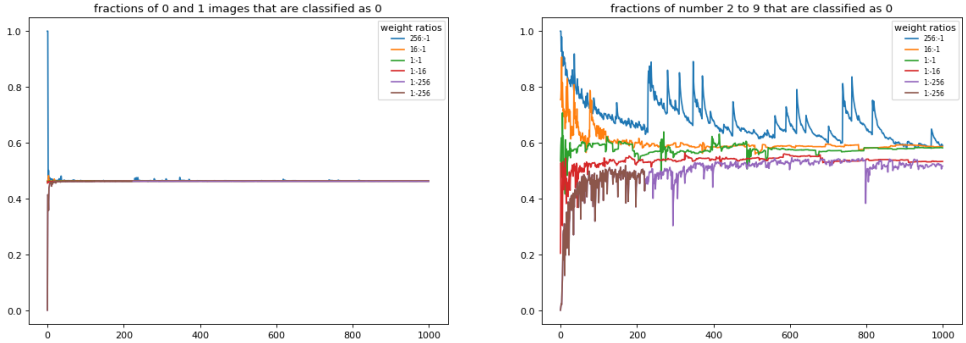
(a) (b)

Figure 3: Simple CNN with l2 (CIFAR-10)



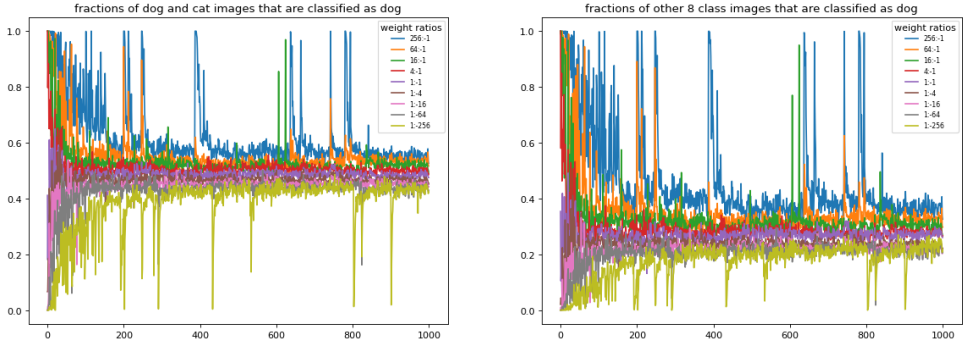
(a) (b)

Figure 4: Simple CNN with dropout (CIFAR-10)



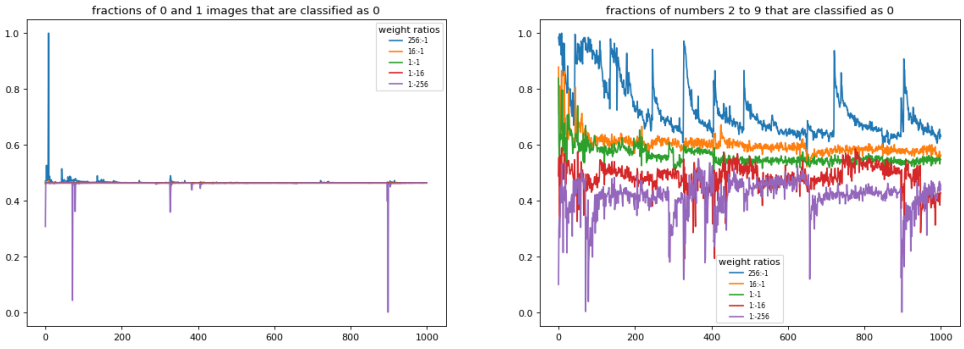
(a) (b)

Figure 5: Simple CNN with dropout (MNIST)



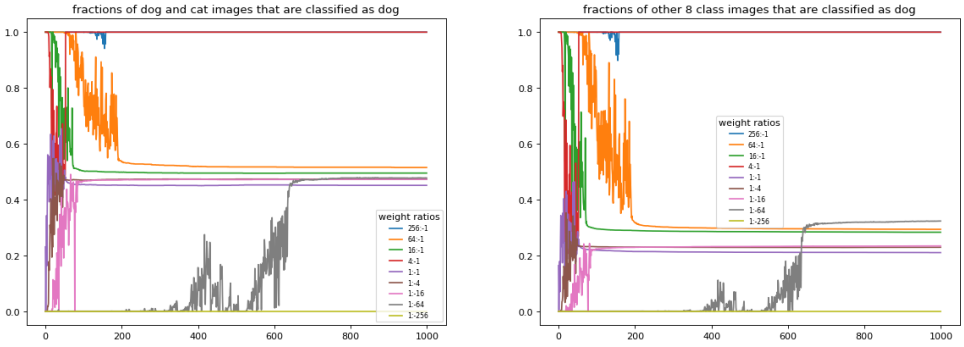
(a) (b)

Figure 6: Simple ResNet (CIFAR-10)



(a) (b)

Figure 7: Simple ResNet (MNIST)



(a) (b)

Figure 8: Simple ResNet without batch normalization (CIFAR-10)