RE-WEIGHTING EXAMPLES FOR DEEP LEARNING - APPLICATIONS TO IMAGE CLASSIFICATION AND CAUSAL INFERENCE

Afek Ilay Adler
Department of Industrial Engineering
Tel Aviv University
afekilayadler@gmail.com

March 14, 2020

ABSTRACT

Curriculum learning and hard example mining are two approaches to re-weight training instances in order to improve learning accuracy in different scenarios. This paper examines new approaches to weighting examples in deep neural networks (NN) proportional to the variance estimation of each sample, a topic with much interest lately. We compare several methods on well known classification data-sets and on estimating average treatment affect (ATE) and show that re-weighting examples based on their variance may lead to increased performance.

***K*eywords** Neural Networks · Curriculum Learning · Hard Example Mining, Active Learning and Causal Inference

1 Introduction

In the last years, deep neural networks (DNN) are the gold standard for many problems in machine learning, due to their capacity to model complex input patterns. Especially In the fields of neural language processing (NLP) and image recognition, these models have shown the produce superior results in classification and regression problems, outperforming different methods and algorithms by a large margin.

Despite their success, DNN are yet prone to training set biases, such as class imbalance. Class imbalance is a very common example in machine learning and can occur in many applications, such as credit fraud prediction where the vast majority of deals are fraudless. In this situations, it is likely that a network will classify fraud transactions as fraudless because there were only a few fraud examples in the training set, thus contributing to the loss function only a small fraction. Another popular training set bias is label noise, e.g., from crowd-sourcing services or weakly supervised data.

Training set biases can be addressed with weighted training loss. Hard example mining prioritize examples with higher training loss since they are more likely to be the minority class, thus re-adjusting the weights of all classes to be approximately uniform in the objective function. On the other hand, by decreasing the weight of difficult examples, robust loss estimators tend to outperform baseline approaches (not weighting at all) and will generally perform better in cases such as the existence of many outliers and label noise, by giving less weight to samples which were labeled incorrectly.

As we have seen, this two approaches work well in different scenarios. Preferring easier examples works well in the beginning of the training or in more challenging situations such as label noise or the presence of outliers. On the other hand, emphasizing hard examples have been shown to accelerate convergence in mini-batch stochastic gradient descent in cleaner data by minimizing the variance of the gradients. Unfortunately, we do not always know the amount of noise in our data, thus choosing the right method is not trivial. In this situations we can use a a small unbiased validation set [1], or consult with experts about the way the data was collected.

We propose to examine how MC dropout [2] compares to different approaches in the literature that emphasize on uncertain samples to improve SGD. specifically, SGD-WPV [3] and prediction entropy [4]. We also propose to show how we can utilize variance estimation techniques in order to solve a problem from the domain of causal inference, estimating which medicine or treatment is better for a patient (ATE) [1].

We present how the different methods compare in terms of accuracy and mean prediction variance, on the well known image classification data sets (FASHION-MNIST, CIFAR 10) and on TWINS for ATE estimation.

2 Related work

The idea of weighting each training example is closely related to importance sampling, both been well studied in the literature. Boosting methods rely on iteratively fitting harder examples with a weak learner, with Adaboost [5] as the pioneering work in the field. Today, gradient boosting trees [6] are more popular due to enhanced performance, and are implemented in many real time systems such as click through rate prediction. In the context of class imbalanced problems, the less represented classes are usually "ignored" by the classifier because those few examples contribute only a little to the overall objective function. Thus, the harder examples are usually the least representative classes. Hard example mining methods try to fix this biases by focusing more on those examples, e.g. [7].

Inspired by the observation that studying easy material before harder material is often beneficial to human understanding, curriculum learning [8], has shown that learning from easier instances first can improve neural network training. In the presence of noise or outliers, robust loss estimators are usually preferred, by doing exactly the opposite than hard example mining - down-weighing examples with high loss. This approach has been studied mainly in the context of noisy labels, because the harder examples are usually the ones with uncertain labels, and robust loss estimators try to fix this problem by focusing more on the easier examples [9].

The methods to weight examples is closely related to the way *active learning* models choose which examples to label. Active learning assumes a changing train set, where we can gradually increase the sample size by choosing more examples to label ('query' the set of examples with unknown label). Thus, the learning algorithm learns incrementally. This is beneficial in many situations, e.g., if labeling is expensive. Active learning aims to choose new samples that will be most informative to the learning process, usually, by emphasizing uncertain samples. One such a query strategy is *query by committee*, where a variety of models are trained on the current labeled data, and vote on the output for unlabeled data; the points which the "committee" disagrees the most are chosen to be labeled.

Lately, there has been attempts to learn the weights by backpropagation [1] or by a neural network ("mentor") that provides curriculum for the "student" network [10]. Next we will introduce different weighting methods, focused on variance estimation methods of a given model.

2.1 Uncertainty and variance estimation in neural networks

There are many ways to compute uncertainty or prediction variance. The first class of methods computes high uncertainty for samples that the model is less confidence in classifying. While this methods do not provide prediction variance, **it is a simple heuristic that computes a proxy** for it. Among the common methods of this approach is the *high loss, least confidence, margin sampling, and prediction entropy* [4] which provide classification uncertainty. Given softmax probabilities for a sample $p_i = f(X_i)$ with K classes:

- High loss: if the loss is high then the uncertainty is large.
- Least confidence: $\max_k(p_{ik})$ if the probability of the most likely class (maximum value of the softmax prediction) is small then the uncertainty is large.
- Margin sampling - the margin is defined to be the distance between the probability of the most likely class to the probability of the second most likely class. If the margin is small then the uncertainty is large.
- Prediction entropy $-\sum_k p_{ik} \cdot log(p_{ik})$. If the entropy is large then the uncertainty is large.

This approach provides uncertainty measures which are straightforward and easy to compute but not necessarily capture situations where the model is not certain about its prediction but is consistent about it.

[1] ATE estimation will be discussed later in detail

Therefore [3] introduced variance metric which rely on storing past predictions and computing a variance based on the history of the model; SGD Weighted by Prediction Variance(SGD-WPV):

$$\widehat{\mathrm{std}}_i^{\mathrm{conf}}(H) = \sqrt{\widehat{\mathrm{var}}\left(p_{H_i^{t-1}}(y_i|\mathbf{x_i})\right) + \frac{\widehat{\mathrm{var}}\left(p_{H_i^{t-1}}(y_i|\mathbf{x_i})\right)^2}{\left|H_i^{t-1}\right| - 1}}$$

Where H_i^{t-1} is the set of past predictions of sample i of the true label.

Bayesian neural networks (BNN) provide variance but are expensive to compute due to computing intractable integrals. Gal [2] offered recently to use MC dropout as Bayesian approximation . MC dropout masks randomly hidden layer neurons (dropout) in prediction time, thus outputting a stochastic realization of the prediction probabilities. By doing it several times we get a series of realizations, with mean (the prediction) and variance (uncertainty). Although not computationally expensive as BNN, it does require to perform multiple forward passes in order get a confidence interval for the prediction.

3 Methods

We intend to compare how MC dropout compares to prediction entropy, and SGD-WPV from [3]. The first represents a method which is easily computed and represents model uncertainty over the labels. The second, is the variant that performed best in [3] and models the prediction uncertainty during the learning process. We also compare all the aforementioned methods to a baseline model with uniform weights.

In order to perform a fair comparison between all the models we need to differ only the weights computation between the different experiments. Since MC dropout requires architecture with dropout, we include dropout in each architecture. We also limit our experiments to SGD scan with weighted loss, opposing to sample training instances based on their variance estimation.

We begin to weigh instances after a number of warm up epochs. For each mini batch, we calculate the weight given the aforementioned methods and add an ϵ in order to prevent a case of zero weight. Than, we normalize the weights to have a mean of 1 [3]. Finally, we compute per each sample the prediction variance via MC dropout and compute the results.

Now we present the results [2] on image classification and in the next chapter we will elaborate more on ATE estimation with propensity score methods.

4 Image classification: results

Dataset	Baseline	Entropy	MC Dropout	SGD-WPV
FMNIST	0.878 ± 0.0035	0.885 ± 0.003	0.885 ± 0.0021	0.884 ± 0.0029
CIFAR 10	0.889 ± 0.0416	0.858 ± 0.0813	0.781 ± 0.1145	0.88 ± 0.0522

Table 1: Accuracy on validation set (10 runs)

Dataset	Baseline	Entropy	MC Dropout	SGD-WPV
FMNIST	0.0405 ± 0.00140	0.0394 ± 0.00185	0.0910 ± 0.00628	0.0409 ± 0.00206
CIFAR 10	0.0078 ± 0.00299	0.0086 ± 0.00358	0.0145 ± 0.00446	0.0080 ± 0.00248

Table 2: Mean Variance of the true label on the validation set, the variance is calculated via MC Dropout

4.1 Fashion MNIST

Fashion - MNIST is a dataset consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 gray-scale image, associated with a label from 10 classes (fashion items). The dataset is balanced and does not contain noisy labels. We use Lenet5 architecture, a simple convulsion neural network (CNN) with 2 convulsion layers with max-pooling and 3 fully-connected layers [11].

[2] All code can be found at `https://github.com/AfekIlayAdler/UNI-2020-DeepLearning`

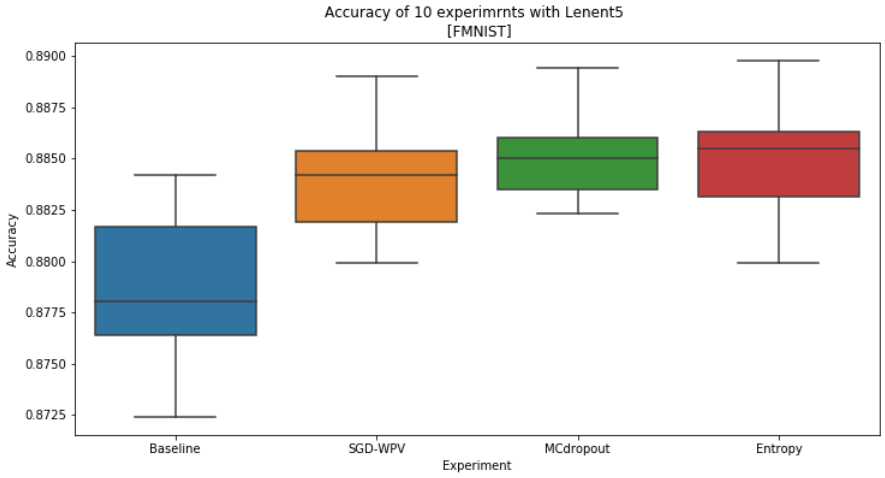

Figure 1: Accuracy on Fashion MNIST with Lenet5 architecture

We use the following hyper-parameters: 10 warm up epoches, 45 epoches overall, ADAM optimizer and initial learning rate of 0.01 which decreases in a factor of 0.5 in epochs 10,20,30 consecutively. We use dropout rate of 0.1 in the last two layers and compute the mean and standard deviation of MC dropout on 15 realizations.

We can see in figure 1 that all weighting methods outperform the baseline by almost 1% (the results are significant with a p value of 2%). As well, we can see that MC dropout is the most stable method with the smallest variance between experiments.

As we can see in figure 3 (Appendix), Directly calculating the variance per each sample via MC dropout produces meaningful results. Even samples that the model predicts with high probability can get high uncertainty, opposed to methods such as entropy. This result is important to many real world situations and adds significant amount of information to the decision maker.

Another interesting result we can see in figure 4. Models which were trained with MC dropout produce significantly higher uncertainty measures than all the other methods. This is a very interesting result that we leave to further study.

4.2 CIFAR 10

CIFAR 10 is a dataset consisting of a training set of 50,000 examples and a test set of 10,000 examples. Each example is a 32x32 colour image, associated with a label from 10 classes (animals and transport automobiles). The dataset is balanced and does not contain noisy labels. In order to check how re-weighting examples performs in deep neural network, we use Resnet18 architecture [12]. Resnet is a notorious neural network that uses convolution layers, batch-norm layers and residual connections.

We use the following hyper-parameters [3]: 150 warm up epoches, 170 epoches overall, SGD optimizer with momentum (0.9) and initial learning rate of 0.1 until epoch 150 and than 0.0001. We use dropout rate of 0.1 in before the last layer and compute the mean and standard deviation of MC dropout on 15 realizations. In order to use dropout on a fully connected layer we add one hidden layer in the end of the architecture with 128 neurons.

As we can see in figure 2. models with weighted training loss do not improve the baseline. MC Dropout performs poorly and its performance decreases when the warm up epochs end and the model start to learn with weighting. As in FMNIST, the mean variance of the true label is significantly higher in MC Dropout than all the other methods (see figure 6). We discuss this result in the conclusions section.

[3] Our model is based on the implementation in `https://github.com/kuangliu/pytorch-cifar`

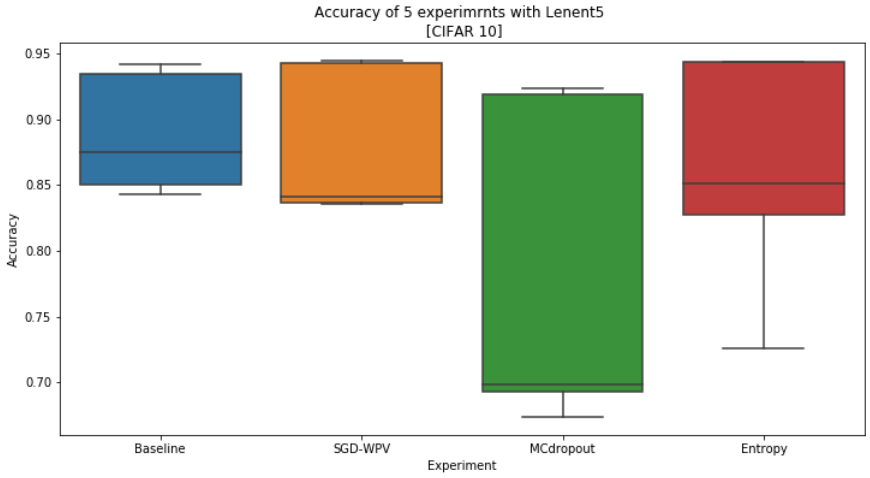

Figure 2: Accuracy on CIFAR 10 with Resnet18 architecture

5 Estimating average treatment affect (ATE)

5.1 Brief introduction

Estimating ATE is a key problem in science, where the goal is to to help decision makers to decide among several possible strategies. e.g., should a doctor prescribe a patient with lung cancer, with treatment A or treatment B? or does the treatment leads to a better results, compared to the control group?

Over the years there has been much confusion when answering these causal questions from observational data. In the last years, though some discrepancies remained, a unified theory of quantitative causal inference had emerged [13].

We are interested in more accurate estimation of average treatment effects under unconfounded treatment assignments, in situations where there is lack of overlap in the covariate distributions between treatment groups.

Next, we will introduce the preliminary background on ATE estimation in observational studies, and the method of inverse probability of treatment weighted estimator that we aim to improve with our weighting scheme [14] .

5.1.1 Potential outcomes - definitions

Let T is a binary treatment $T \in \{0, 1\}$ and Y_{obs} be the observed outcome.
Each unit (sample) has two potential outcomes:

- Y_1 - the unit's outcome had they been subjected to treatment $T = 1$
- Y_0 - the unit's outcome had they been subjected to treatment $T = 0$

We do not observe Y_0, Y_1 In order to connect between the unobserved potential outcomes and the observed outcome we use the consistency assumption:

$$Y_{obs} = TY_1 + (1 - T)Y_1$$

And the ATE:

$$ATE := E[Y_1 - Y_0]$$

Note:

$$Y_1 \; not \; necessarily \; equal \; Y_1|T = 1$$

E.g., say T is job training. $Y|T = 1$ is the income we expect from people who actually went to job training. But Y_1 is the income we expect if we **forced** everyone to go to job training.

5.1.2 Randomized controlled trial (RCT) and observational studies

In a **completely randomized** experiment (A\B testing), the potential outcomes is independent of the treatment:

$$(Y_0, Y_1) \perp\!\!\!\perp T$$

This results leads to:

$$ATE = E[Y_1 - Y_0] = E[Y_{obs}|T = 1] - E[Y_{obs}|T = 0]$$

If we have an **observational** data, we can identify the causal effect by the following assumptions:

- The potential outcomes are independent of treatment assignment, conditioned on observed covariates (variables):

  $$(Y_0, Y_1) \perp\!\!\!\perp T|X$$

  This assumption means that that we measured all the confounder (variables that affect the treatment assignment and the observed values). Or in other words - there are no hidden confounders.

- Common support (overlap) -

  $$p(T = t|X = x) > 0, \forall t, x$$

Under this assumptions we get:

$$ATE = E[Y_1 - Y_0] = E_x[E[Y_{obs}|x, T = 1] - E[Y_{obs}|x, T = 0]]$$

5.1.3 Propensity scores and inverse propensity score weighting (IPSW)

Rosenbaum and Rubin [15] showed that if the first assumption holds than:

$$ATE = E_x[E[Y_{obs}|e(x), T = 1] - E[Y_{obs}|e(x), T = 0]] \tag{1}$$

Where $Propensity\ Score := e(x) = p(T = 1|X)$ implying that adjustment for the propensity score suffices for removing all biases associated with differences in the covariate [16].

The problem in equation (1) is that we assume to have $p(x)$ but we have only samples of $p(x|T)$. By the Bayes role, dividing with the propensity score we end up with the final final estimate for the ATE, the inverse probability of treatment weighted estimator [16]:

$$\frac{1}{N}\sum_{i=1}^{N}\left(\frac{Y_i \cdot T_i}{\hat{p}\left(X_i\right)} - \frac{Y_i \cdot (1 - T_i)}{1 - \hat{p}\left(X_i\right)}\right) \tag{2}$$

The result of equation (2) is important and we can interpret it as simply weighing examples differently in zones where the treatment assignment is not evenly distributed. Imagine a simple classification algorithm that estimates the propensity score. Lets assume that we have 4 samples - 3 of them are positive (T = 1) and one is negative (T=0, denotes as Y_4) centered inside a small grid (close to each other). if a sample falls inside this grid the classifier predicts proportional to the samples in the train set. That is, $\frac{3}{4}$ to be positive and $\frac{1}{4}$ to be negative. Lets see how we compute the ATE:

$$\frac{1}{N}\sum_{i=1}^{N}\left(\frac{Y_i \cdot T_i}{\hat{p}\left(X_i\right)} - \frac{Y_i \cdot (1 - T_i)}{1 - \hat{p}\left(X_i\right)}\right) = \frac{1}{4}(\frac{Y_1 + Y_2 + Y_3}{\frac{3}{4}} - \frac{Y4}{\frac{1}{4}}) = \frac{Y_1 + Y_2 + Y_3}{3} - Y_4 \tag{3}$$

We can see that the ATE is computed by the average over the positive samples minus the negative sample. That is to say, given a sample in the test that falls inside this region, if the sample is going to be treated than we predict the result by the mean of the treated examples in that grid in the train set and vice versa.

Apparently , all we need to do is to train a classifier to predict $p(T|X)$, and compute the ATE with the IPSW equation. But the problem of limited overlap between $p(T = 1, X)$ and $p(T = 0, X)$ remains. If the distributions of control and treated is different than there are places where there is a limited overlap; places where the vast majority of samples belong to either the control or treated population. The problem with limited overlap can lead to conventional estimators of average treatment effects having substantial bias and large variances [17]. for those samples, the denominator of the IPSW will be close to zero, so small deviations in the denominator will cause big changes in the ATE.

In order to solve the limited overlap (Crump et al, 2009) [17] suggested a simple rule of thumb solution; discard all units with with propensity score outside the interval [0.1,0.9].

5.2 Suggested approach

We limit ourselves to estimate the ATE with IPSW, as opposed to methods such as nearest neighbors matching. Since re-weighting training instances has been shown to improve learning accuracy, we may find finer methods for estimating ATE which may lead to important contributions:

1. Estimating $\hat{ATE}$ more precisely
2. Discarding less units when calculating $\hat{ATE}$
3. Lowering the variance of $\hat{ATE}$

We will compare and examine the following approaches:

1. Discard all units with $\hat{PS}$ outside the range $[0.1, 0.9]$ [Baseline].
2. Discard all units with $\hat{PS}$ outside the range $[0.1, 0.9]$ and $\sqrt{\hat{Var}(PS_i)} \geq \beta$.
3. Estimate propensity score with re-weighting examples, in order to obtain a more robust propensity score[4].

We will test our framework on the twins dataset, which has ground truth. We will compare logistic regression with 2 architectures of neural networks.

5.3 Twins

The twins dataset was introduced by [18] -

> " We introduce a new benchmark task that utilizes data from twin births in the USA between 1989-1991 The treatment t = 1 is being born the heavier twin whereas, the outcome corresponds to the mortality of each of the twins in their first year of life. Since we have records for both twins, their outcomes could be considered as the two potential outcomes with respect to the treatment of being born heavier. We only chose twins which are the same sex. Since the outcome is thankfully quite rare (3.5% first-year mortality), we further focused on twins such that both were born weighing less than 2kg. We thus have a dataset of 11984 pairs of twins. The mortality rate for the lighter twin is 18.9%, and for the heavier16.4%, for an average treatment effect of 2.5%. For each twin-pair we obtained 46 covariates rel ating to the parents, the pregnancy and birth: mother and father education, marital status, race and residence; number of previous births; pregnancy risk factors such as diabetes,renal disease, smoking and alcohol use; quality of care during pregnancy; whether the birth was at a hospital, clinic or home; and number of gestation weeks prior to birth.
>
> In this setting, for each twin pair we observed both the case t= 0 (lighter twin) and t= 1 (heavier twin). In order to simulate an observational study, we selectively hide one of the two twins; if we were to choose at random this would be akin to a randomized trial. In order to simulate the case of hidden confounding with proxies, we based the treatment assignment on a single variable which is highly correlated with the outcome: GESTAT10, the number of gestation weeks prior to birth. It is ordinal with values from 0 to 9 indicating birth before 20 weeks gestation, birth after 20-27 weeks of gestation and so on. We then set
>
> $$\mathbf{t}_i|\mathbf{x}_i, \mathbf{z}_i \sim \text{Bern}\left(\sigma\left(w_o^\top \mathbf{x} + w_h(\mathbf{z}/10 - 0.1)\right)\right)$$
>
> $$w_o \sim \mathcal{N}(0, 0.1 \cdot \mathcal{I}), w_h \sim \mathcal{N}(5, 0.1)$$
>
> where z is GESTAT10 and x are the 45 other features. "

5.4 ATE - Results

Unfortunately, numerous attempts to train deep and shallow neural networks failed to outperform our baseline method. We believe it is due to the fact that the Twins dataset has mostly categorical columns and the discriminative signal between the treated and the not treated is weak and not correlated to other features, (after one hot encoding, we have 163 features). Since mostly one feature was manipulated (GESTAT10) in order to create the treatment assignment.

[4] And repeat the aforementioned approaches, with a smaller α

As well, Given x, predicting whether the tween is the heavier or not is hard since we do not have measurements about the twin itself (only birth order). Thus we believe that expressive models like NN are inferior in this case and will hung up to complicated statistical noise that will not generalize. Since all models struggled to classify $P(T|X)$, there were only several propensity scores outside the interval $[0.1, 0.9]$, leading to no improvements in our suggested methods (figure 7).

The shallow neural network we hereby present consists of connecting the input layer to two activation neurons (0 hidden layers), thus contain around 300 parameters. Of course, in this model, we could not use dropout. Although it seems that weighing by entropy has superior results, it is not statistically significant better the baseline network [5].

Method	# of Runs	MAE from the true ATE
Baseline (LR)	50	0.085 ± 0.003
Shallow NN (Baseline)	10	0.033 ± 0.008
Shallow NN (Entropy)	10	$\mathbf{0.032} \pm 0.01$
Shallow NN (MC Dropout)	10	NA
Shallow NN (SGD-WPV)	10	0.033 ± 0.005

Table 3: Mean absolute error of the estimated ATE compared to the true ATE (lower is better)

6 Conclusion and future work

As we have seen on the image classification tasks, variance weighted training has the potential to improve accuracy in some scenarios (even when classes are balanced and there is no label noise), and in some scenarios to decrease or not to improve the model at all. Overall, if one is interested in improved performance in neural networks, we recommend to try weighted training after the model has already converged for a few epochs, with low learning rate in order not to move to much towards hard examples. Unfortunately, we found that the Twins dataset did not suit for deep learning, at least in the context of ATE estimation. In order to push the boundaries in causal inference, we urge the scientific community to create more datasets in causal inference, as there is in the field of image recognition. For future work, we are interested to check why models that were trained with MC Dropout outputs a larger variance per each example (on average). Overall, we enjoyed learning and implementing uncertainty measures in deep learning, for it's a crucial information to use when building real time decision machines.

References

[1] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

[2] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[3] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems*, pages 1002–1012, 2017.

[4] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(12):2591–2600, 2016.

[5] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

[6] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[7] Qi Dong, Shaogang Gong, and Xiatian Zhu. Class rectification hard mining for imbalanced deep learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1851–1860, 2017.

[8] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[9] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.

[5] Code and other experiments regarding ATE estimation available at https://github.com/AfekIlayAdler/UNI-2020-CausalInference

[10] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*, 2017.

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Miguel A Hernán, John Hsu, and Brian Healy. A second chance to get causal inference right: a classification of data science tasks. *Chance*, 32(1):42–49, 2019.

[14] Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

[15] Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

[16] Keisuke Hirano, Guido W Imbens, and Geert Ridder. Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 71(4):1161–1189, 2003.

[17] Richard K Crump, V Joseph Hotz, Guido W Imbens, and Oscar A Mitnik. Dealing with limited overlap in estimation of average treatment effects. *Biometrika*, 96(1):187–199, 2009.

[18] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pages 6446–6456, 2017.

7 Appendix

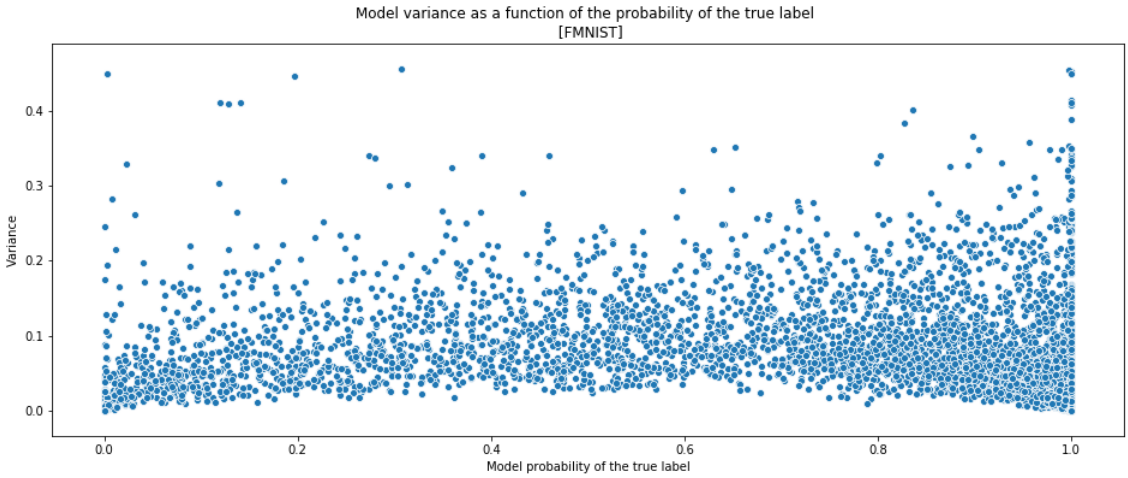

Figure 3: Model variance as a function of the probability of the true label [FMNIST]

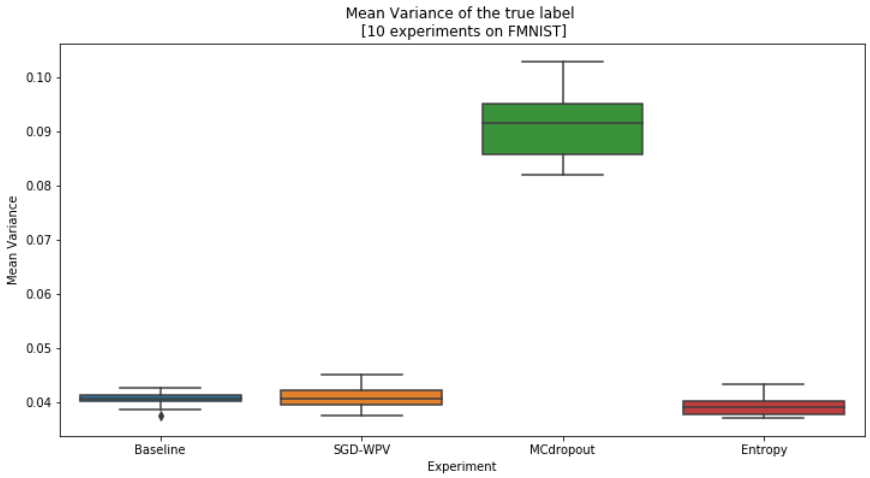

Figure 4: Mean variance of the true label [FMNIST]

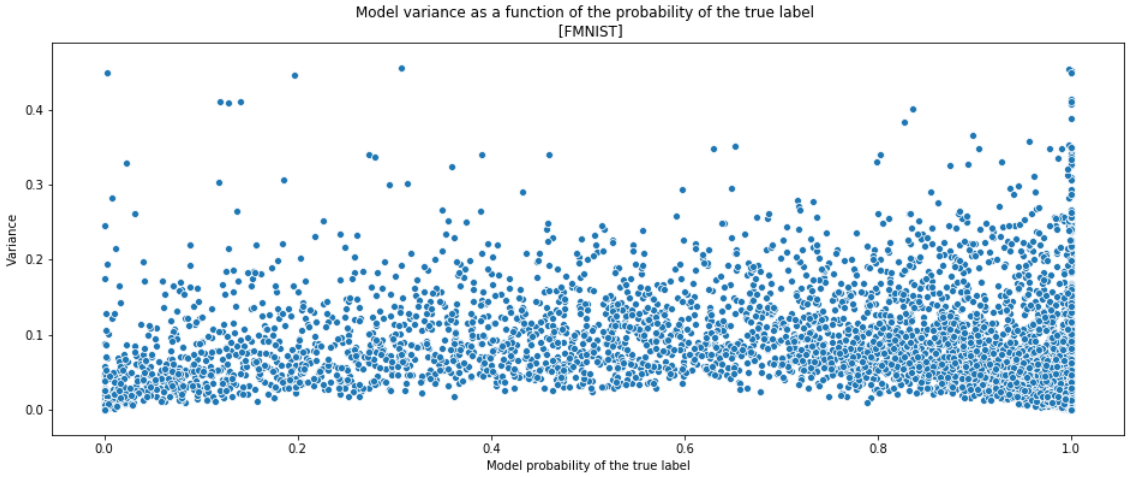

Figure 5: Model variance as a function of the probability of the true label [CIFAR 10]

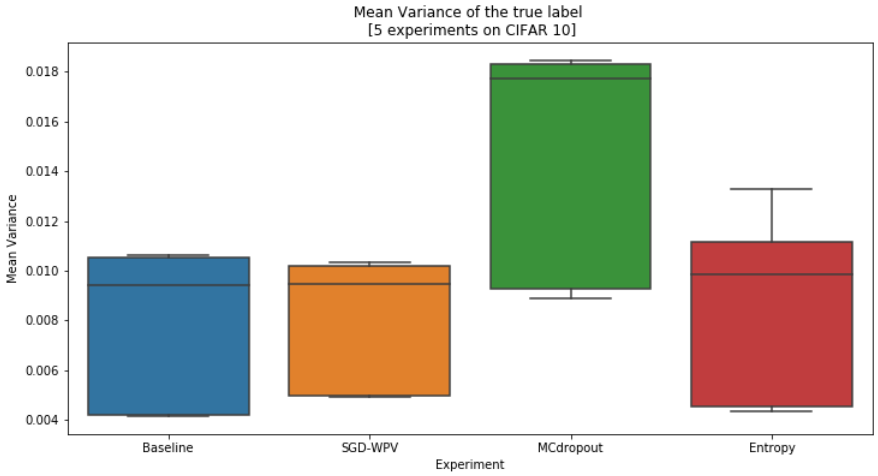

Figure 6: Mean variance of the true label [CIFAR 10]

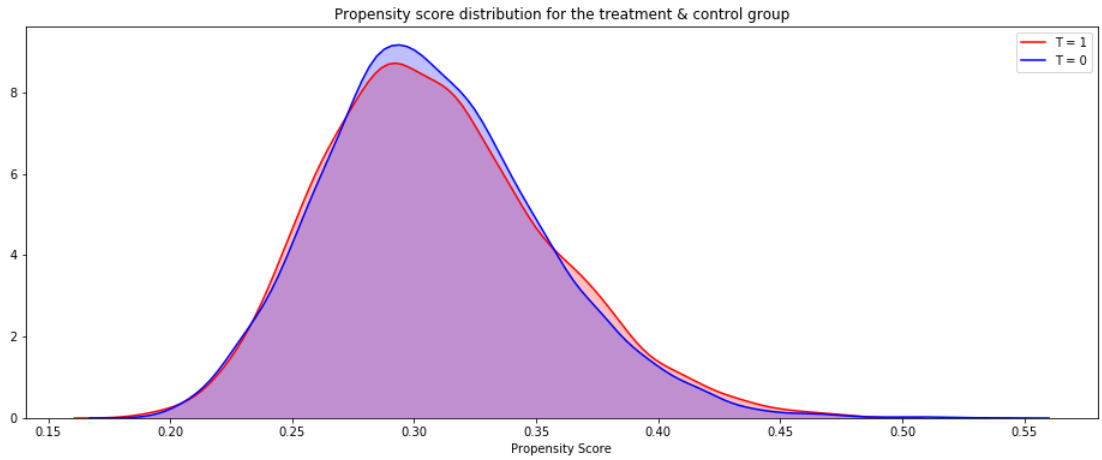

Figure 7: Propensity Score of the Shallow Neural Network