CS 229 Project Report: Weakly Supervised Classifiers with Adversarial Training in High-energy Physics

Sanha Cheong[1]
(CS 229 Project, Autumn 2017)
[1]*Department of Physics, Stanford University, Stanford, CA, 94305, USA*
(Dated: December 16, 2017)

This project implements and studies the performance of a relatively new machine learning paradigm called weakly supervised classifiers (WSC's). WSC's are a particularly useful substitute for fully supervised classification when the truth labels of training data are unavailable or unreliable, which is extremely common in high-energy physics (HEP). Instead of using truth labels, WSC's typically divide the training dataset into (mini-) batches using a variable other than the features, called a batching variable or a latent variable. One key challenge to using WSC is choosing a batching variable that is independent of the features, which is not always an obvious task in HEP.

This work studies the effect of correlation between the batching variable and the features on the performance of WSC's with toy datasets. The WSC network tends to show higher bias and variance when the batching variable is correlated with the features. This project also suggests an adversarial approach to mitigate this effect. Using adversarial training to de-correlate the batches makes WSC's a more universally useful tool.

INTRODUCTION

Classification problems appear extremely frequently in the analysis of experimental data in high-energy physics (HEP). One category of physical objects that is particularly interesting is "jets." The HEP community has been active in applying machine learning (ML) techniques to classify/tag different jets. Most of such efforts utilize fully supervised classifiers (FSC's); FSC's are trained using Monte-Carlo (MC) simulation results labeled with the underlying physical process. However, high-energy collider experiments (especially jets) are extremely difficult to simulate accurately from the first principles. In particular, the MC data generated from different simulation softwares show subtle, but observable differences. Because supervised ML exploit the subtle features in the training data, the classifiers become dependent on the subtleties of simulation tools and do not learn the true distribution.

Some novel ML approaches have been suggested to overcome this issue. One of them is called *weakly supervised classifiers* (WSC's) ([1]). FSC's for binary classification typically solve the optimization problems that look like:

$$f_{\text{full}} = \underset{f:\mathbb{R}^n\to\{0,1\}}{\operatorname{argmin}} \sum_{i=1}^{N} \ell\left(f(\mathbf{x}^{(i)}), t^{(i)}\right) \tag{1}$$

where f is the predictor function, ℓ is the loss for one instance, and $\left\{\mathbf{x}^{(i)}, t^{(i)} : i = 1, \ldots, N\right\}$ is the training dataset with features and truth labels.

In contrast, WSC's solve the following optimization problem:

$$f_{\text{weak}} = \underset{f:\mathbb{R}^n\to\{0,1\}}{\operatorname{argmin}} \sum_{K\text{'s}} \ell\left(\frac{1}{|K|}\sum_{i\in K} f(\mathbf{x}^{(i)}), y_K\right) \tag{2}$$

where K's are different (mini-) batches of the training dataset and y_K is the ratio of signal (labeled $t = 1$) data points in the batch K. (Hereon, we will often refer to each of these partitioned dataset as "a batch" instead of a "minibatch".)

One natural way in HEP to divide the dataset into different batches is to bin them by a latent variable z which is not one of the features $\mathbf{x}$. For instance, one can use jet shape and fragmentation as features and divide the training dataset according to jet masses. This, however, requires that z and $\mathbf{x}$ are independent; if the distribution $p(\mathbf{x})$ of features is dependent of z and therefore are different across batches, then the WSC will not be learning one target function $f(\mathbf{x})$ and be biased with respect to z. We want WSC's to learn the mapping to label predictions based on the distribution of features, not the residual effects of the differences across the batches.

This project will study the performance of WSC's and the effects of the correlation between the batching variable z and the feature $\mathbf{x}$ on the performance using toy datasets. We will also describe an adversarial training method to actively de-correlate the predictor function $f(\mathbf{x})$ against the latent batching variable z, thereby increasing the performance of WSC's and making it a more universally useful technique.

RELATED WORKS

WSC and other approaches using signal ratios instead of individual truth labels have been applied to HEP problems previously, such as [1] and [2]. These works demonstrated that the WSC's and similar approaches can become reasonable alternatives to FSC's, especially for another well-known problem in HEP called quark/gluon tagging.

However, the early works on using WSC in HEP such as [1] do not comment on the correlation between the batching variable & the features or the impacts on the performance thereby. We believe that the WSC's in earlier works can benefit from the adversarial approach we suggest later in this paper.

ML classifiers becoming dependent/biased on unwanted physical variables is also a known issue and has been studied, particularly the jet classifiers being biased on jet masses (see [3]). Other general ML research on de-correlation includes [4]

It seems that there has not been much work in combining the two techniques: WSC's and the de-correlation of unwanted variables.

TOY DATASETS & CORRELATIONS

The toy datasets used in this study are 100000 points with 2-dimensional features generated by two different Gaussians; 80% are background ($t = 0$) and 20% are signal ($t = 1$). See Fig. 1.

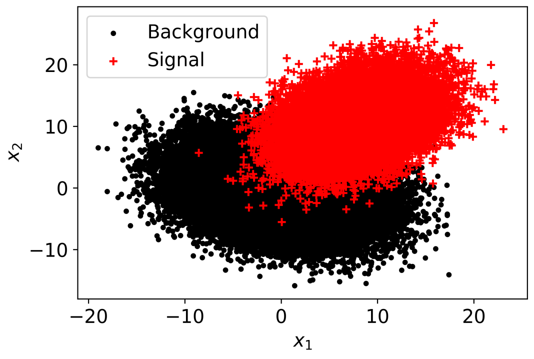

FIG. 1: Scatter plot of the features of the uncorrelated dataset.

The data points have different random distributions in the batching variable z; the background is distributed with a power law ($p \propto z^{-1}$), while signal is distributed with a Gaussian over the same range. See Fig. 2. The data points are binned into 28 different batches over z with a bin size $\Delta z = 1$. The batch in the bin $z \in [15, 16]$, for instance, has about $\sim 60\%$ signal, while the batch in the bin $z \in [20, 21]$ has $\sim 20\%$.

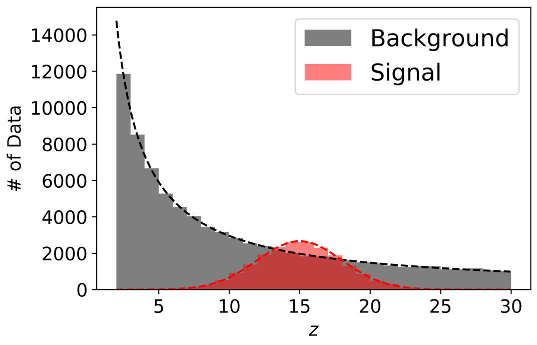

FIG. 2: Theoretical distributions & histograms of the batching variable z.

As they are now, the feature $\mathbf{x} = (x_1, x_2)$ and the latent batching variable z are independent, except for the residual effects; the Pearson-r coefficient is -0.01. However, to study the effects of the correlation on the WSC performance, we need to introduce the correlation between them *without* distorting the distribution of the features themselves.

We achieve this by *partially sorting* $\mathbf{x}$ and z in the signal data. That is, given the uncorrelated training dataset, we select a random subset of the signal data points and swap around their latent variable z values so that they are sorted against one of the features, say x_1. This makes the sorted subset of the signal data to have complete correlation ($r = 1$) between x_1 and z, thereby increasing the overall correlation in the signal events while leaving the overall distributions of the dataset in $\mathbf{x}$ and z unchanged.

Hence, we obtain 10 training datasets with completely identical distribution in $\mathbf{x}$ and z separately, but with varying correlations (Table.I). Because the feature distribution remains completely identical, we expect the FSC's to perform identically over these datasets. We expect, however, that the performance of WSC's will drop as the correlation increases.

We keep 10% of the data out as a validation set and check all performance measures against it.

TRAINING THE WSC'S & FSC'S

As our model, we use a neural network with a single hidden layer with 50 nodes for both WSC's and the FSC's. The FSC's with the same neural network structure is used as our guidance on what to expect in the best case. The activation function used at the hidden layer is a rectified linear unit (ReLU) defined as:

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{3}$$

% of Signal Data Sorted	Pearson-r between x_1 and z
Unsorted	-0.011
10%	0.101
20%	0.197
30%	0.290
40%	0.398
50%	0.497
60%	0.608
70%	0.712
80%	0.796
90%	0.898

TABLE I: The signal correlations in each dataset. Their overall distributions in $\mathbf{x}$ and z respectively are identical.

while the activation function at the output node is the sigmoid function $\sigma(x) = 1/\left(1 + e^{-x}\right)$.

We use the binary cross entropy (BCE) as our loss function for each instance. BCE is defined as:

$$\ell(o, y) = -y\log(o) - (1-y)\log(1-o)$$

where o is an output from a classifier for a data point and y is the truth label.

When training WSC's, as described in Eq. 2, we use the average prediction over a (mini-)batch and the signal ratio in the batch as our o and y instead. Hence, although we are using multiple different minibatches per training epoch, only the average prediction and the signal ratio over each batch are used for one (instance) loss function calculation, essentially treating the average over a batch like a single instance. The FSC's in this study were trained using the batch gradient method.

The learning rates and the epochs are optimized individually for the WSC's and the FSC's; though the results are not shown here, many test runs were made to optimize the learning rates and the epochs and avoid overfitting (especially for the FSC), using the training and the validation losses and the accuracies as the main metrics over each epoch. The optimal values for the learning rate and the training epochs differ significantly due to the different loss functions and optimization algorithms.

One caveat to note is that the WSC calculates the loss function (and the gradient) using only the overall signal ratio, essentially treating the minibtach as one instance. This always has a trivial solution—the constant function $f = y_K$. Hence, the WSC very prone to falling into this trivial solution, especially when the signal ratios do not vary significantly between batches. As a trick to avoid this trivial solution, it is helpful to shuffle the orders of the batches around in each epoch, so that the WSC network sees varying overall ratios of signal across batches. If the batches are iterated over sequentially (e.g. each z-bin as seen in Fig. 2), the WSC sees a smoothly varying sequence of y_K's after each batch and easily falls into the constant solution. When the difference between the consequent y_K's are large, such behavior is discouraged. It is also advised that the learning ratio for the WSC must be not too small, since the loss function is definitely not convex with lots of local minima.

EXPERIMENTS & PERFORMANCE RESULTS

For each dataset, 30 WSC's and FSC's were trained with respectively optimized learning rates and training epochs.

We first demonstrate that WSC's are stable and able to achieve performance similar to the FSC's on average, based on the toy dataset. Although the final average accuracy does not differ significantly from the FSC's, the WSC's show much larger spread, as expected. We can also see from Fig. 3 that the WSC's often get stuck at local minima, as shown by occasional horizontal regions in the curves. However, with a reasonably large learning rates and training epochs and shuffled batches, most of them can be escaped eventually.

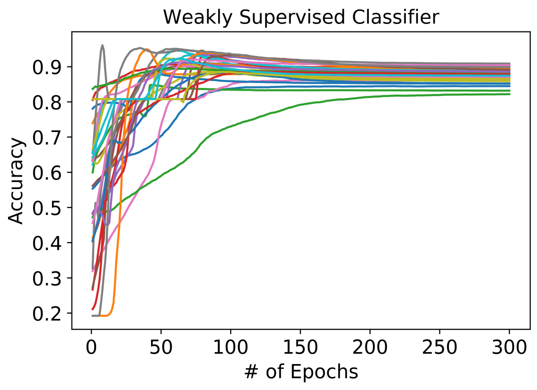

FIG. 3: Validation accuracies of 30 individual WSC's over training epochs.

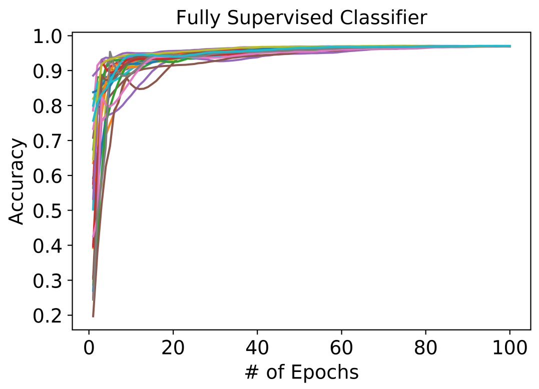

FIG. 4: Validation accuracies of 30 individual FSC's over training epochs.

Their receiver-operating characteristic curves (ROC's) were also calculated along with the areas under the ROC's (AUC's). Training 30 WSC's and FSC's on the uncorrelated training dataset, we saw:

$$\begin{aligned} \mathrm{AUC}_{\mathrm{full}} &= 0.9910 \pm 0.0004 \\ \mathrm{AUC}_{\mathrm{weak}} &= 0.9821 \pm 0.0065 \end{aligned} \tag{4}$$

as the means and the standard deviations.

As described in the previous section, 9 datasets with identical feature distribution but different x_1-z correlations were used for experiments. We report the final validation accuracies and the AUC's over different correlations as our performance metrics. The dots and the error bars in Fig. 5 and 6 represent the median and the interquartile range from the 30 different realizations each for WSC and FSC.

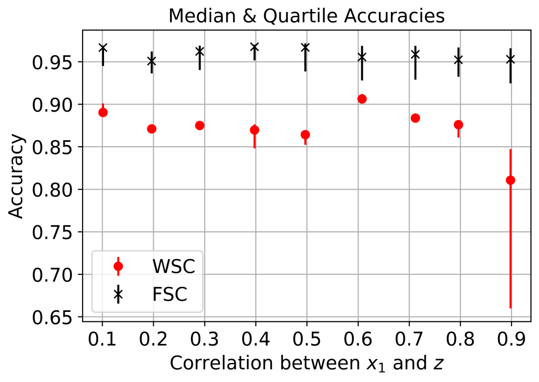

FIG. 5: Final validation accuracies over datasets with different correlations $r(x_1, z)$.

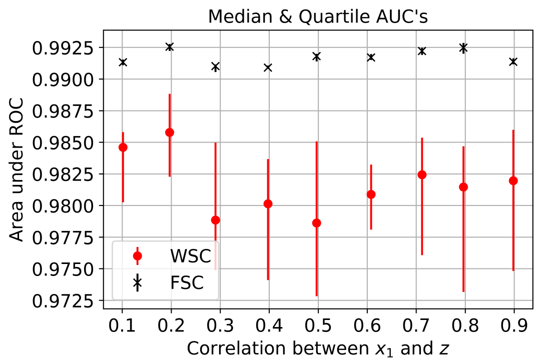

FIG. 6: Final validation accuracies over datasets with different correlations $r(x_1, z)$.

Of course, as expected, we see that the performances of the FSC's are unaffected by adding arbitrary correlations. Meanwhile, we see that the performance of the WSC's drop overall as the correlation gets larger. Although the WSC validation accuracies do not show an obvious tendency, the tendency is AUC is somewhat clearer; the variances in the AUC seems to increase as a function of correlations. It is worth noting that the dataset with 60% of the signal data shuffled is somewhat anomalous, since both the overall accuracy and the spread in AUC is significantly smaller.

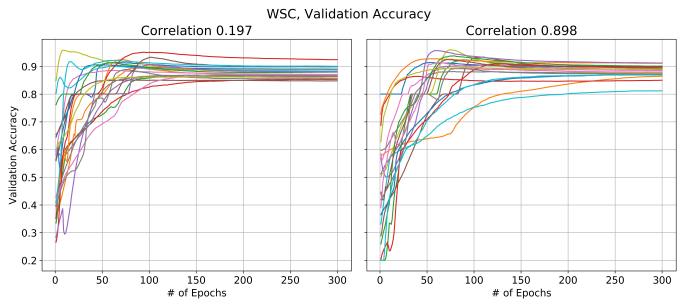

FIG. 7: Comparison of WSC trainings for dataset with different signal correlations.

The fact that correlation makes WSC training less reliable and more difficult is even more clearer in Fig 7. We see that the spread in the validation accuracy curves is much larger in the high-correlation dataset and that the networks tend converge much slower; we expected this because the distribution of the features is not identical across the batches and the networks are therefore not fed the accurate information.

Note that our experiments here only added correlations to the signal data points, which only accounts for the 20% of the total training dataset. If the background data points were correlated as well (which would often be the case in HEP applications), the overall correlation will be increased largely, and WSC performances will be affected much more largely. However, the analysis of background correlation is much more complicated and was not covered in this project.

ADVERSARIAL APPROACH FOR DE-CORRELATION

Mathematically speaking, what we are trying to achieve here is to train a feature-based predictor $f(\mathbf{x};\theta_f)$ that is independent of the batching variable z. i.e.

$$p\left(f(\mathbf{x};\theta_f)=t|z_i\right)=p\left(f(\mathbf{x};\theta_f)=t|z_j\right)$$

for all z_i, z_f bins corresponding to different batches K's.

To achieve this, we define another objective function called the *adversary* which models the latent variable z and captures whether (or how strongly) the feature-based prediction $f(\mathbf{x};\theta_f)$ is dependent on z:

$$g(z|t;\theta_g)=p(z|f(\mathbf{x};\theta_f)=t,\theta_g) \tag{5}$$

If the predictor f is completely independent of z (which is what we want), $g(z|t;\theta_g)=g(z)$; in other words, the adversary will reduce to a random guess based on the prior distribution over z. Then, such an adversary g should maximize the overall adversarial loss defined as $\ell_g(\theta_f,\theta_g)=\frac{1}{N}\sum_{\text{data}}\ell_g\left(p(z|f(\mathbf{x};\theta_f)=t,\theta_g)\right)$ over the training dataset. In our toy model case, g will be a discrete probabilistic prediction over which z-bin the input might belong into, and ℓ_g can be a categorial cross entropy, for example.

To train our neural network towards this goal, we define a new objective function defined by:

$$\mathcal{L}(\theta_f,\theta_g)=\frac{1}{N}\sum_{\text{data}}\ell_f(\theta_f)-\beta\ell_g(\theta_f,\theta_g) \tag{6}$$

where β is some positive constant and solve the optimization problem:

$$\arg\min_{\theta_f}\max_{\theta_g}\mathcal{L}(\theta_f,\theta_g) \tag{7}$$

In each epoch, we first train the adversarial objective function g, so that we come up with the best z-predictor we can, by minimizing ℓ_g (or, equivalently, maximizing $-\beta\ell_g$). Then, we train our classifier $f(\mathbf{x};\theta_f)$ such that it minimizes the classification loss ℓ_f as well as maximize the batching loss ℓ_g; this is to minimize our ability for it to predict the latent batching variable z and therefore make the classifier independent of z.

The parameter β represents which objective we want to emphasize more; the smaller the value of β, our overall objective function $\mathcal{L}$ gets closer to simple ℓ_f, which means that our neural networks will simply focus on the feature-based prediction, but will not pay attention to de-correlating against the latent batching variable z. If β is too high, the network will focus too much on de-correlating against z, but the prediction will not become as strong. When actually using this method in real applications, β must be tuned as a hyperparameter, depending on the what the real-world goal is.

REVIEW & FUTURE PLANS

In this project, we studied the performance of a relatively new ML paradigm called the weakly supervised classification, which is a useful substitute when the truth labels of the training dataset is unavailable or unreliable. This new method instead uses just the ratio of signal (positive-labeled) data points across different batches. The expected downside of this method, however, is that the distribution of the features across the different batches must be identical, which is not always easy to achieve. We study the performance of WSC's in correlated datasets and observe that the WSC's become less stable and show higher variance in their accuracies.

Because constructing feature-independent batches is not always a simple task, we instead suggest an adversarial training method that will help our classifiers become explicitly feature-independent. The key idea is to add a negative of the adversarial loss function to our overall objective function and thereby maximizing the adversarial loss. This corresponds to minimizing our ability to predict the latent batch variable z, which is what we want from our z-independent classifiers. We expect WSC performances to increase significantly and become more robust against correlations between $\mathbf{x}$ and z, when modified with this approach.

Unfortunately, given the time constraints of the project and the effort required in pre-processing the HEP data, the WSC methods have not yet been applied onto real HEP classification challenged, nor the adversarial approaches were tested properly. It will also be helpful to run more experiments on more complicated classification problems, along with the correlations added onto the background data points as well.

[1] L. M. Dery, B. Nachman, F. Rubbo, and A. Schwartzman, JHEP **05**, 145 (2017), arXiv:1702.00414 [hep-ph] .

[2] E. M. Metodiev, B. Nachman, and J. Thaler, (2017), arXiv:1708.02949 [hep-ph] .

[3] C. Shimmin, P. Sadowski, P. Baldi, E. Weik, D. Whiteson, E. Goul, and A. Sgaard, (2017), arXiv:1703.03507 [hep-ex] .

[4] Y. Ganin and V. Lempitsky, ArXiv e-prints (2014), arXiv:1409.7495 [stat.ML] .