Weakly Supervised Classifiers with Adversarial Training

Sanha Cheong
sanha@stanford.edu
Department of Physics, Stanford University

STANFORD UNIVERSITY

Objective

- Implement and study the performance of **weakly supervised classifiers** (WSC)
- Investigate the effect of using **correlated** (not independent) batches
- Design and test **adversarial** strategies to mitigate this effect
- (Apply to high-energy physics (HEP) data: hardonically decaying $W^{\pm}$ boson v.s. light-quark jets)

Weakly Supervised Classifiers

Traditional fully supervised classifiers (FSC) solve the following optimization problem:

$$\arg\min_f \sum_i \ell\left(f(\mathbf{x}^{(i)}), t^{(i)}\right)$$

In contrast, **WSC** only use the **ratios** of signal events (instead of individual labels) across different batches.

$$\arg\min_f \sum_{\text{batches } K} \ell\left(\frac{\sum_{i\in K} f(\mathbf{x}^{(i)})}{N_K}, y^{(K)}\right)$$

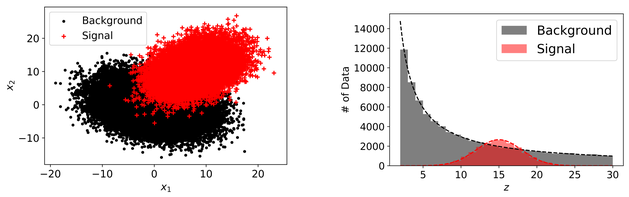
One useful way to partition training data into batches is to **bin** them according to a "batching" variable z (also called the nuisance) that is not in the feature vector $\mathbf{x}$, such that different ratios of signal data are well-known at different z-bins.

This is particularly useful in HEP; we often want to train a classifier over some features $\mathbf{x}$, given some theories providing ratios of signal events across another variable z.

However, this requires that the features of interest $\mathbf{x}$ are **independent** of (uncorrelated with) z. Otherwise, the classifier will learn and classify based on the correlation, not purely based on the features.

Toy Dataset & Model

We use a toy dataset with features distributed according to 2D Gaussians. 20% of the overall data are signal points ($t = 1$).

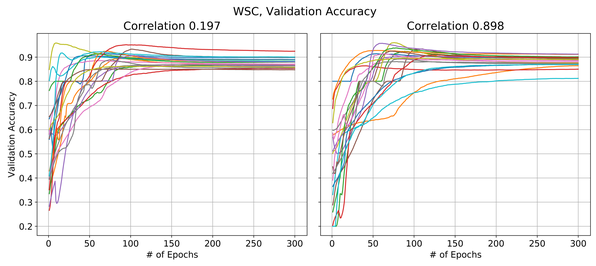


z and $\mathbf{x}$ are initially uncorrelated (only residual effects). To study the effects of correlation, the signal data are **partially sorted** in x_1 and z to introduce correlation. The Pearson correlation coefficient ranges from $r = -0.01$ (uncorrelated) to $r = 0.90$ (90% sorted).

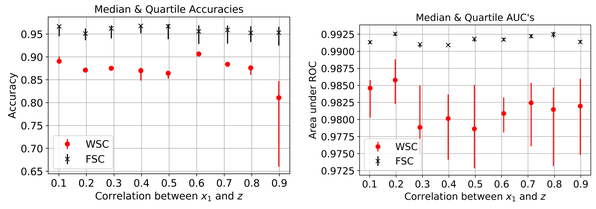
As our model, we use a neural network with one hidden layer with 50 nodes for both WSC and FSC. We use ReLU for the hidden unit activation and sigmoid for the output unit. Binary cross entropy is used as the loss function. Learning rates and epochs are optimized individually.

Performance & Correlational Effects

We train 20 WSC's and FSC's on the toy datasets to study their performances. We see that high correlation tends to decrease the performance of WSC's.



We also compare them against the baseline FSC's.



Adversarial Strategies: Theory

The task at hand is to adversarially **decorrelate** our WSC's to remove the unwanted z-dependence and improve classification performance.

Since we want our prediction $f(\mathbf{x};\theta_f)$ to be **independent** of the batching variable z, we require that:

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

across all bins $z_{i,j}$. If the prediction $f(\mathbf{x};\theta_f) = t$ for a given set of features $\mathbf{x}$ is dependent of z, then this correlation should impact the **adversarial** function (also called the adversary):

$$g(z|t;\theta_g) \equiv p\left(z | f(\mathbf{x};\theta_f) = t;\theta_g\right)$$

Otherwise, $g(z)$ should produce random guesses (according to the prior $p(z)$), which is precisely what we want. Hence, we want to maximize the adversarial loss ℓ_g.

We attempt this with two different optimization processes per epoch. At each epoch, we first minimize $\ell_g(\theta_f, \theta_g)$ to find our best nuisance prediction of z; this can be done using gradient descent updates on θ_g. Then, we want to optimize our feature-based predictor to f to best predict t from $\mathbf{x}$, but we also want the nuisance predictor to perform poorly. Hence, we update θ_f such that it minimizes $\ell_f(\theta_f)$ while maximizing $\ell_g(\theta_f, \theta_g)$; i.e. minimize $\ell_f - \ell_g$.

In summary, we are solving the following optimization problem:

$$\arg\min_{\theta_f} \max_{\theta_g} \left[\ell_f(\theta_f) - \ell_g(\theta_f, \theta_g)\right]$$

Review & Future Work

Difficulties I had...

- Running multiple experiments took much **longer time** than expected
- Preparing properly controlled dataset was non-trivial
- HEP data are high-dimensional and complicated; thorough toy model testing was prerequisite

If I had more time...

- Experiment more cases to investigate the correlational effects further (more in detail)
- Implement the **adversarial** strategy and experiment on the toy dataset
- Generalize to a more complicated toy dataset and then to **HEP problems**

Acknowledgements

I thank Dr. Francesco Rubbo (rubbo@slac.stanford.edu) from the SLAC ATLAS Group and the teaching assistants from CS 229 for their supervisions & help with this project.

References

[1] Y. Ganin, V. Lempitsky. *Unsupervised Domain Adaptaion by Backpropagation*. arxiv:1409.7495

[2] A. Søgaard. *Adversarially trained jet substructure taggers*. ATLAS Machine Learning Forum