Gradient reversal for MC/real data calibration

Artem Ryzhikov[1,2], Andrey Ustyuzhanin[1,2,3]

[1] Yandex School of Data Analysis
[2] NRU Higher School of Economics
[3] Moscow Institute of Physics and Technology

E-mail: artemryzhikov@gmail.com

ACAT 2017 University of Washington,
Seattle, August 21-25, 2017

In this research we propose a technique for training neural networks on mixture of MC-simulated signal and real background sample that allows to avoiding overfitting to simulated artifacts. The technique is based on cross-domain adaptation approach with gradient reversal [1]. The method shows significantly better results than Data Doping [2]. Moreover, gradient reversal gives more flexibility and helps to ensure flatness of the network output wrt certain variables (e.g. nuisance parameters) as well.

Introduction

Usage of Monte Carlo-generated sample is fairly common approach in the High Energy Physics. However, not all variables can be simulated accurately enough, so the discrepancies may lead either to

a) expensive simulation of both signal and background, or to..

b) ML models trained on simulated sample that overfits to the simulated artifacts and work poorly on the real data.

In the research we propose a technique to train neural networks that are safe to train on mixture of simulated and real data.

Problem

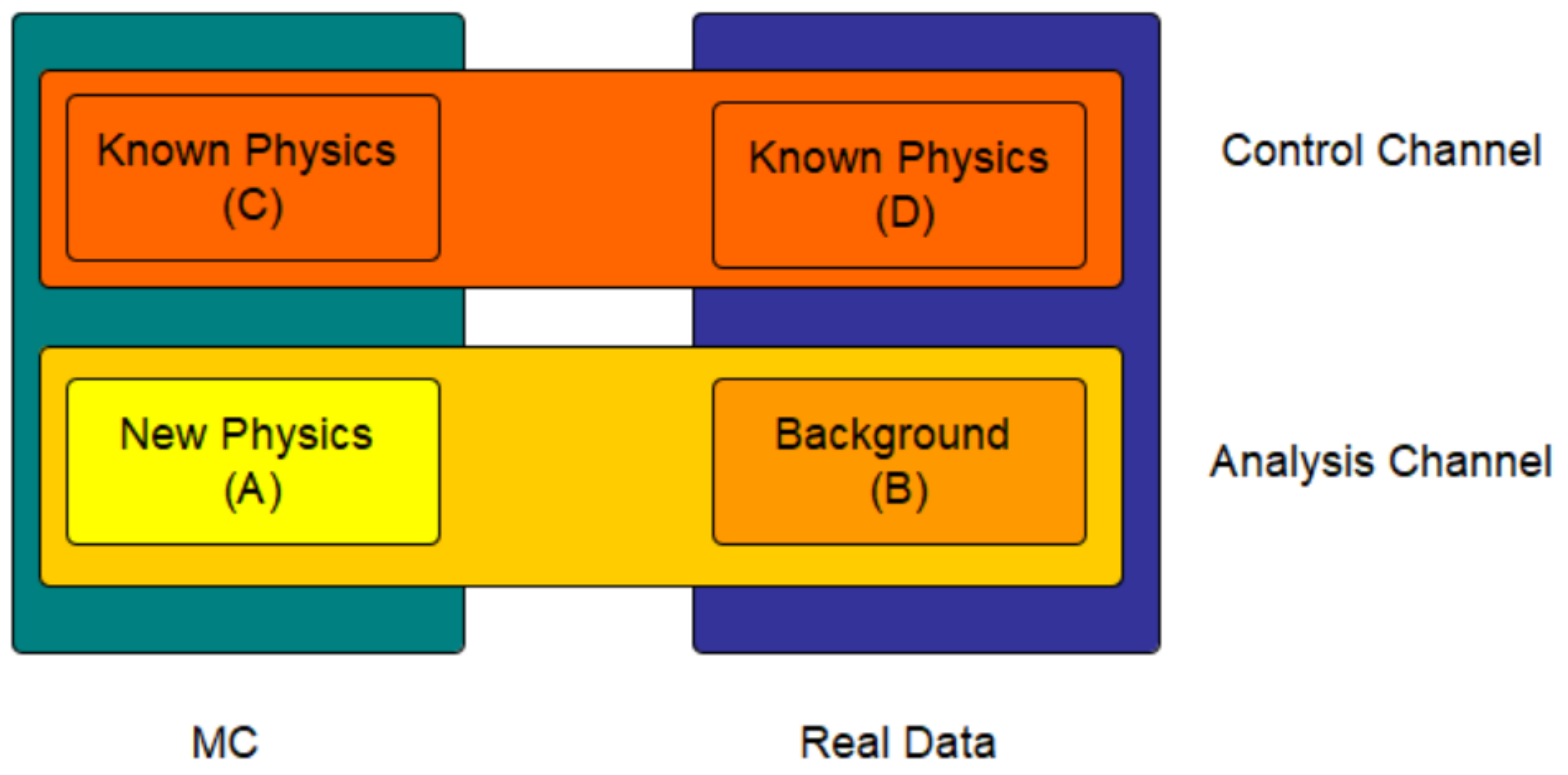

Figure 1. Training on the mixture of simulated (MC) and real data

In the research we use $\tau \rightarrow 3\mu$ events as signal (analysis) channel that has been published at the Data Science challenge on kaggle.com [3]. The challenge is three-fold:

1) Since the classifier is trained on a mixture of simulated signal and real data background, it is possible to reach a high performance by exploiting features that are not perfectly modeled in the simulation. We require that the classifier should not have a large discrepancy when applied to real and simulated data. To verify this, we use a control channel, $D_s \rightarrow \varphi\pi$, that has a similar topology as the signal decay, $\tau \rightarrow 3\mu$ (analysis channel). $D_s \rightarrow \varphi\pi$ is a well-known decay, as it happens much more frequently. So the goal is to train a classifier able to separate A from B but not C from D *(Figure 1.)*,. A Kolmogorov-Smirnov (*KS*) test is used to evaluate the differences between the classifier distribution on each sample. In our problem KS is calculated between prediction's distributions for real and simulated data for $D_s \rightarrow \varphi\pi$ channel. The KS-value of the test should be less than 0.09.

2) The classifier output *should not* be correlated with reconstructed mass feature, i.e. it's output distribution should not sculpt artificial bumps that could be interpreted as a (false) signal. To test the flatness we've used Cramer-von Mises (*CvM*) test that gives the uniformity of the distribution [4].

3) The quality of signal discrimination should be as much as possible. The evaluation metric for signal discrimination is Weighted Area Under the ROC Curve (*truncated AUC*) [3]

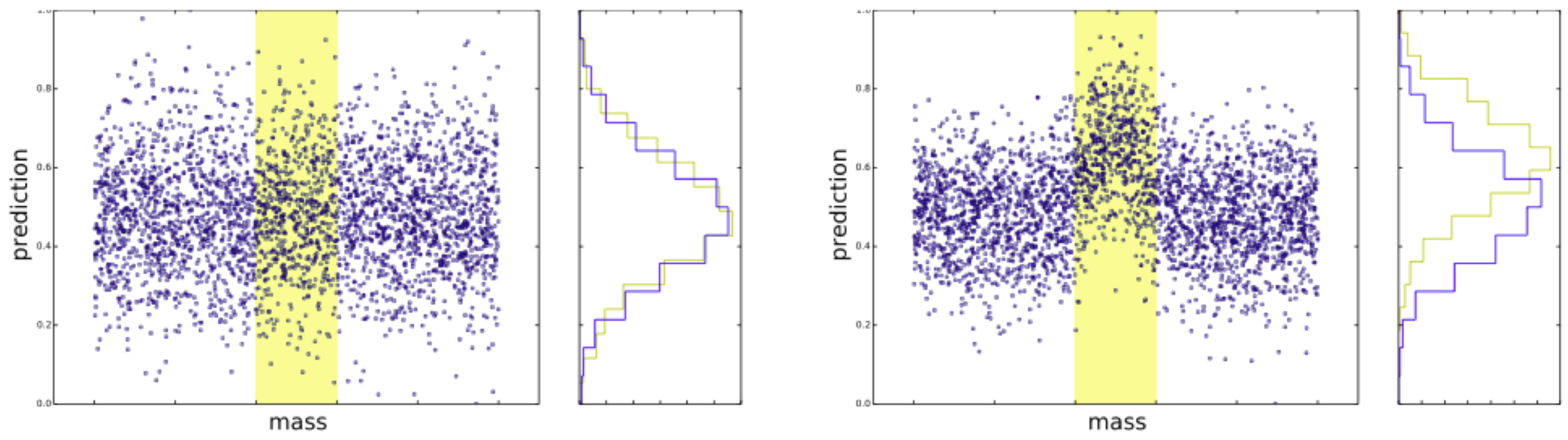

Figure 2. illustration of the CvM correlation test [4]. On the left side there is no correlation with mass (**small CvM values**). On the right side model's predictions are highly correlated with mass (**high CvM values**)

Data Doping (baseline)

In the research we have selected Data Doping [2] as a baseline. The idea is to "dope" the training set with a small number of Monte-Carlo events from the control channel (C), but labeled as background. The optimal number of doping events was taken from [2].

Domain adaptation

The network architecture has a dense 2(3*)-branch structure *(Figure 3*) and consists from following parts:

1. **Feature extractor** – is responsible for feature generation
2. **Label predictor** – is responsible for the target prediction (signal/background discrimination)
3. **Domain classifier** – is responsible for cross-domain adaptation and prevents the network from overfitting to MC domain

4*. **Mass predictor** - helps to eliminate the correlation between classifier predictions and reconstructed mass of the decay

* - **Mass predictor** part (branch) wasn't tested in this research and our architecture was tested without this part. The *Figure 3* was draws ***without*** this part. Theoretically it was designed as additional branch as **domain classifier**, working along the same principle.

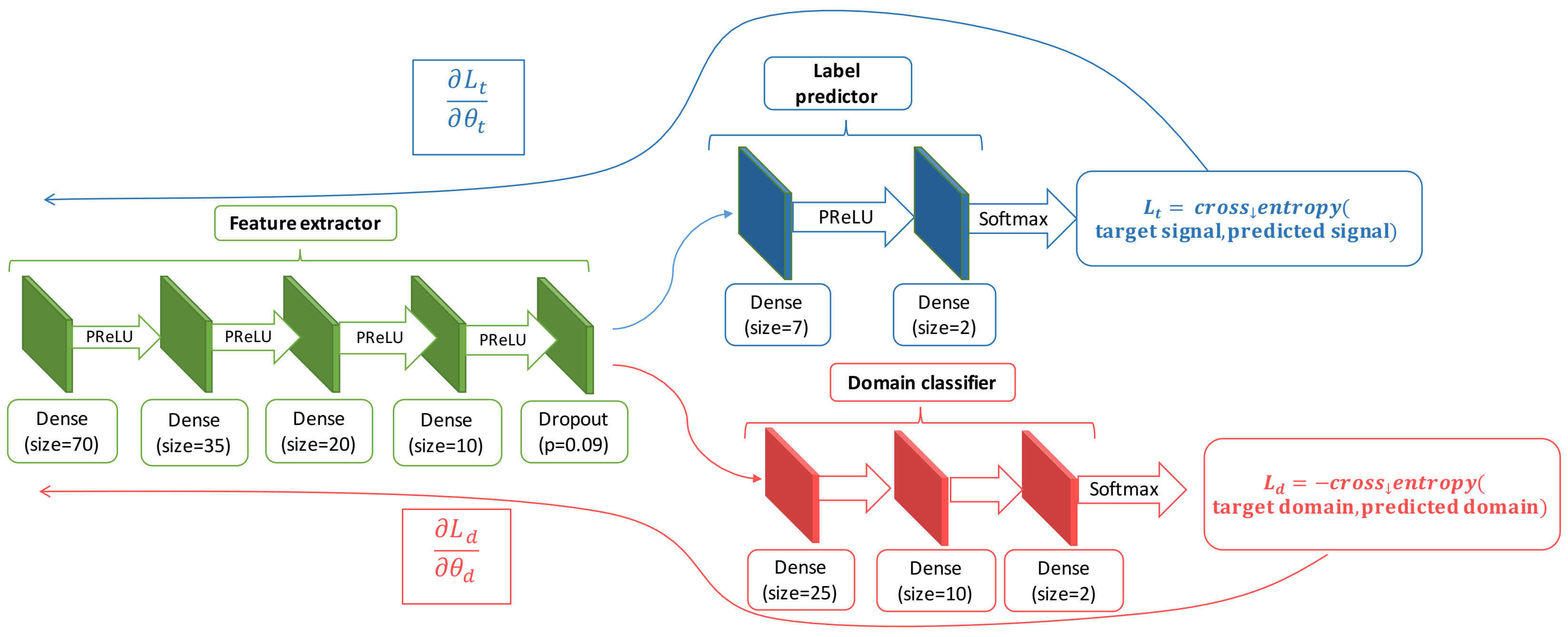

Figure 3. Domain Adaptation

Experiment

Training dataset (Analysis channel) consists of 67000+ events of signal ($\tau \rightarrow 3\mu$) and background events. Control channel consists of 71000+ events of signal ($Ds \rightarrow \varphi\pi$) and background. All events are described by 46 features.

The architecture above was implemented on Python 2.7 using *Lasagne (ver. 2.1)* framework. We tuned the following parameters to obtain stable results:

- learning rates ratio between branches (**learning_rate_multiplier**);
- batch sizes ratio for branches. The best observed values were 1000 and 300 for **Label predictor** and **Domain classifier** respectively
- number of batches per epoch ratio. The best observed ratio between batches number was 6:1 for **Label predictor** and **Domain classifier** respectively

The model was trained for 20 epochs with RMSProp optimizer.

KS-value was eliminated by increasing of **Domain classifier**'s learning rate, increasing corresponding batch size and batches frequency. But too small values of ***KS*** makes ***CvM*** values higher and ***AUC*** metric smaller. *Figure 4* represents such dependency from one of such parameters. So the goal was to find balance between ***KS, CvM*** and ***AUC*** using parameters described above

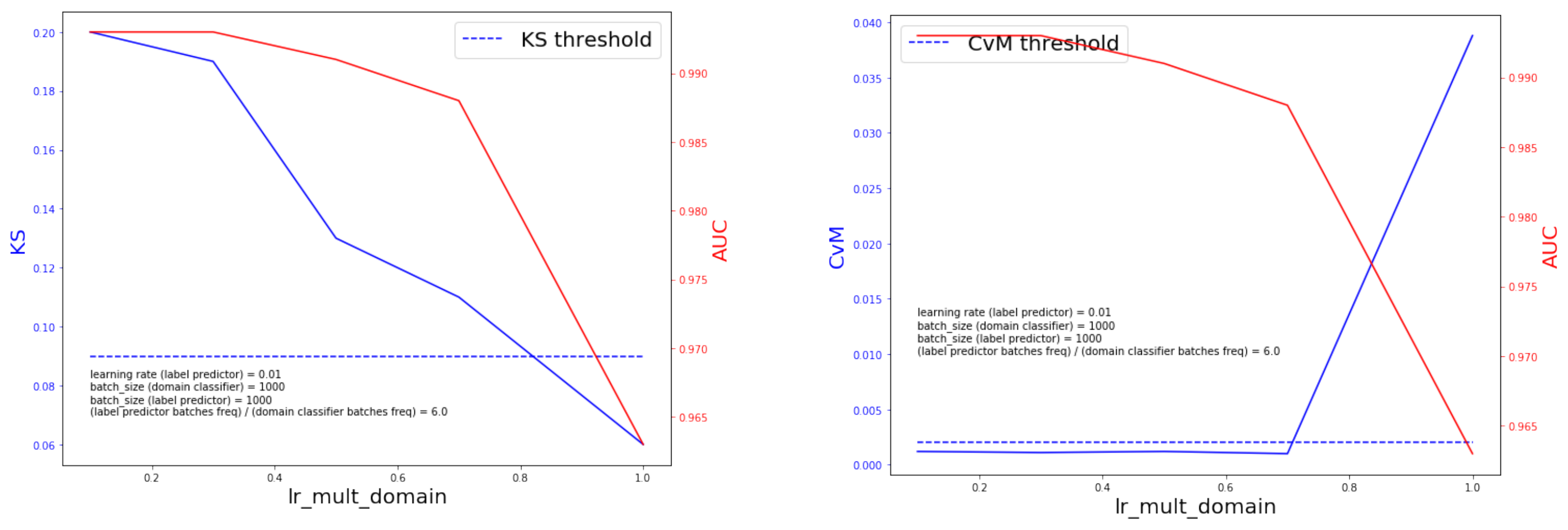

Figure 4. Metrics dependency from domain classifier's **learning_rate_multiplier**

Results

In the research following models were compared: **Baseline** (label predictor from *Figure 3* without Domain Adaptation), **Domain Adaptation** (our approach), **Data Doping**. Models were tested on 85000+ events of signal ($\tau \rightarrow 3\mu$) and background.

The tests showed that this architecture is a robust mechanism for choosing tradeoff between discrimination power and overfitting, moreover, it also improves the quality of the baseline prediction. Thus, this approach allowed us to train deep learning models without reducing the quality, which allow us to distinguish physical parameters, but do not allow us to distinguish simulated events from real ones.

As shown in the table below our method provides the best solution for signal detection problem ($\tau \rightarrow 3\mu$).

Model Metric	Mass-aware Classifier	Data Doping	Domain-adaptation
AUC (truncated)	**0.999**	0.9744	0.979
KS (< 0.09)	0.18	0.087	**0.06**
CvM (< 0.002)	0.0008	0.0011	**0.0008**

Conclusion

The method proposed is shown to work well on a typical particle physics analysis problem:

- Remarkable classification quality;
- Robustness to MC / Real data mixture;
- Uniformity of the output wrt chosen (mass or nuisance parameter) feature
- Tradeoff between discrimination power and overfitting tuned *(Figure 4)*

References

[1] Ganin, Y, and V. Lempitsky. "Unsupervised domain adaptation by backpropagation." International Conference on Machine Learning. 2015.

[2] V. Gaitan, Data Doping solution for "Flavours in Physics" challenge https://indico.cern.ch/event/433556/contributions/1930582/

[3] Flavours of Physics Competition, https://www.kaggle.com/c/flavours-of-physics

[4] A. Rogozhnikov, A. Bukva, V. Gligorov, A. Ustyuzhanin, M. Williams "New approaches for boosting to uniformity", JINST, 2015

[5] A. Ryzhikov, A. Ustyuzhanin Source code for Domain Adaptation research https://github.com/Leensman/Cross-domain-adaptation-on-HEP-HSE-course-work-