National Research Tomsk State University

Лаборатория анализа данных физики высоких энергий Томского государственного университета

Measurement of differential cross-sections of a single top quark produced in association with a W boson with ATLAS at $\sqrt{s} = 13$ TeV

Progress Report

Neda Firoz

Goal: separate **tW (top+anti-top)** from **$t\bar{t}$** in the **1j1b dilepton** region

•**Inputs (9 variables used):**
bdt_centrality_ll_recalc_NOSYS,
bdt_delta_pT_ll_MET_recalc_NOSYS,
bdt_delta_pT_llb_MET_recalc_NOSYS,
bdt_eta_llMetB_recalc_NOSYS,
bdt_m_l1b_recalc_NOSYS,
bdt_m_l2b_recalc_NOSYS,
bdt_pT_llMetB_recalc_NOSYS,
bdt_pT_llb_recalc_NOSYS,
bdt_sum_ET_recalc_NOSYS.

•**Samples / tree:** all files' tree name is analysis

Signal: tW (top) + $\bar{t}$W (anti-top)
Background: $t\bar{t}$ (non-all-had)

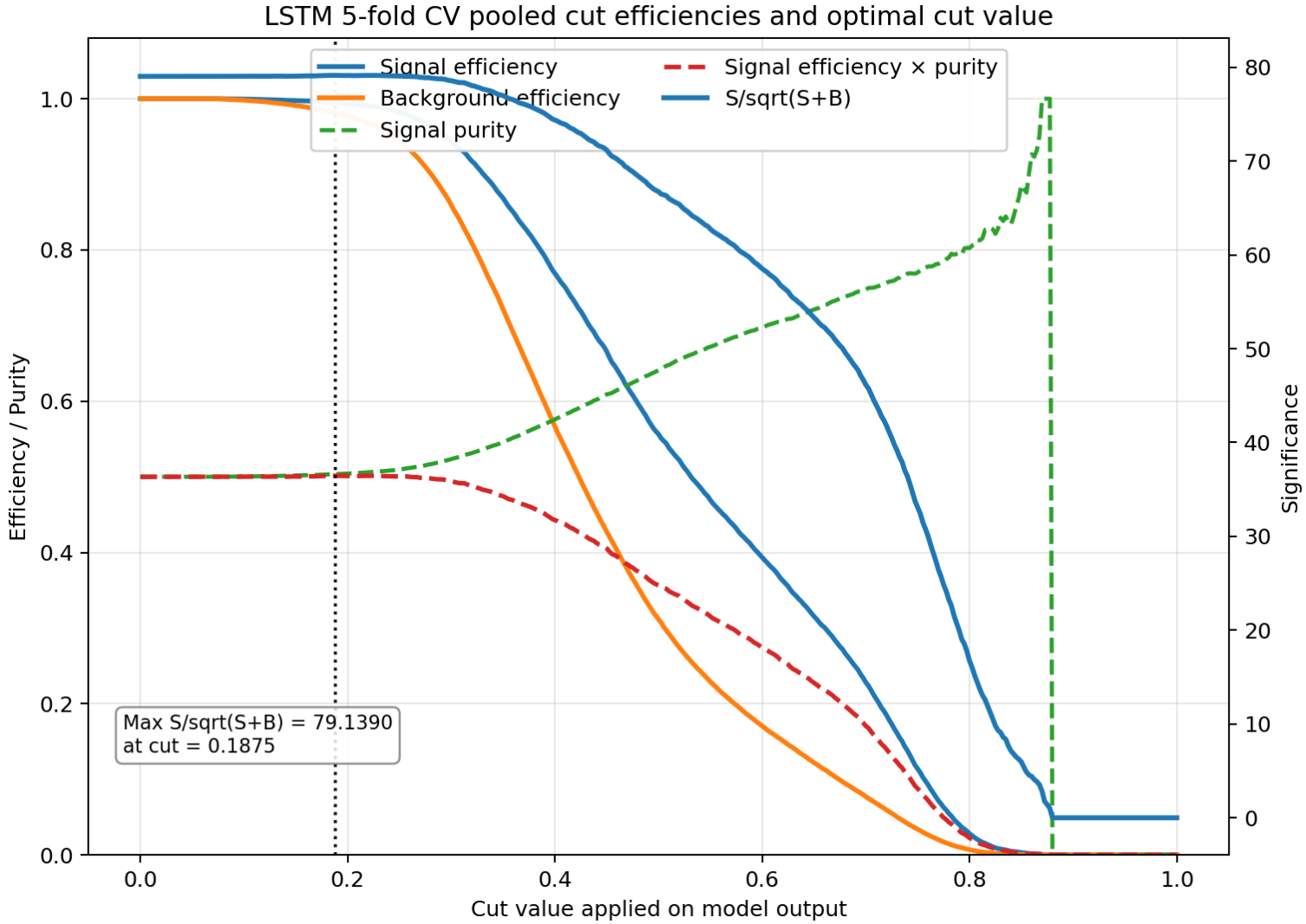
•**Event weights:** auto-resolved to weight_mc_NOSYS * weight_pileup_NOSYS * globalTriggerEffSF_NOSYS.

•**Bad values:** any variable ≤ **−990** or non-finite is masked per event.

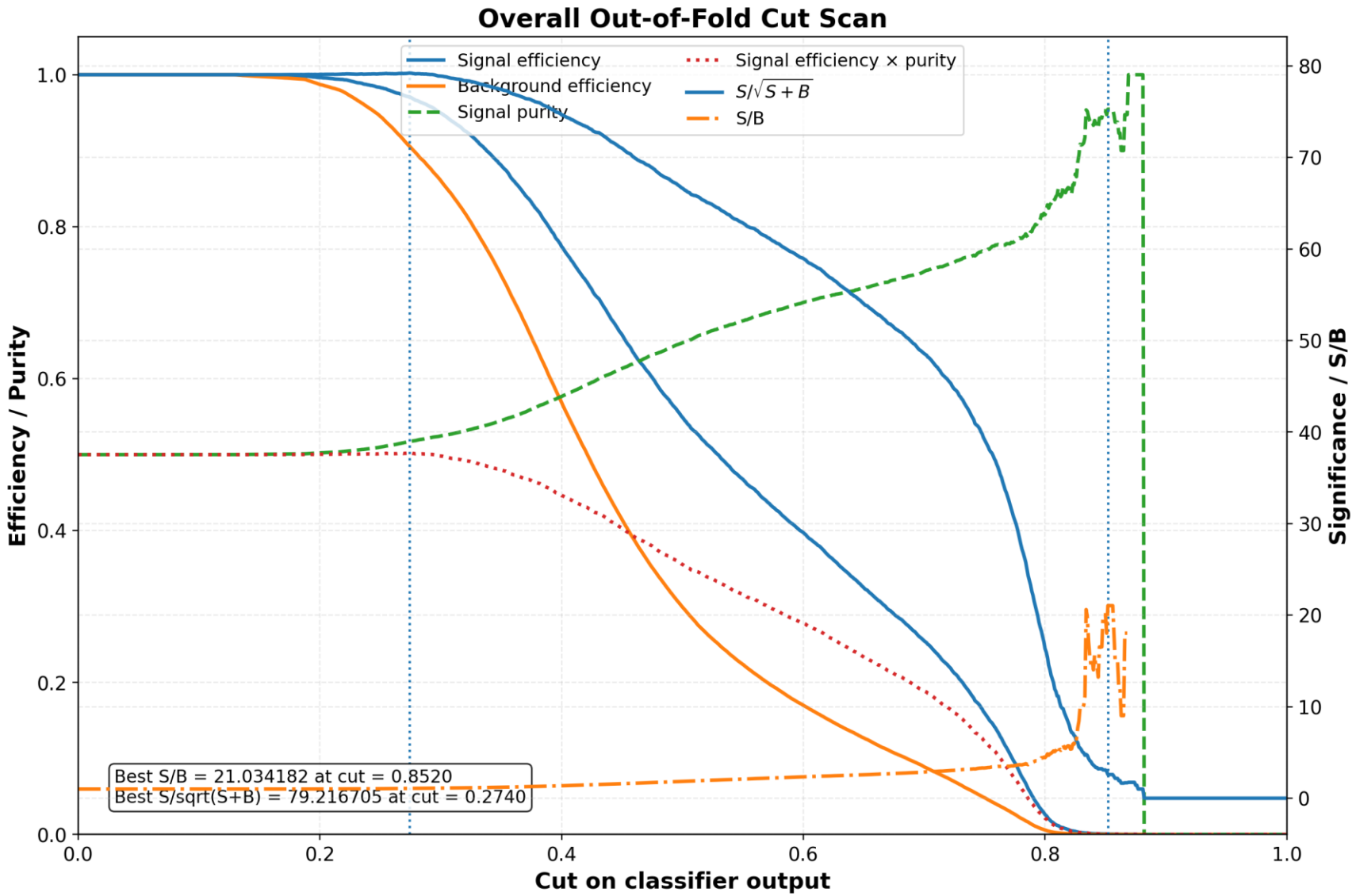
Progress made until 24.03.2026

1. Article writing some progress made, but I have to eliminate some errors before finalizing the methods section since the hyper-parameters need to be finalized.
2. Was working with new variables on Python platform first, but could not make much progress last week, need more time to present some results.
3. Also simultaneously was working to prepare some tasks on R for first year students, still need more time to present everything together….

LSTM


LSTM 5-fold CV pooled cut efficiencies and optimal cut value
Signal efficiency
Background efficiency
Signal purity
Signal efficiency × purity
S/sqrt(S+B)
Max S/sqrt(S+B) = 79.1390
at cut = 0.1875
Efficiency / Purity
Significance
Cut value applied on model output
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
50
60
70
80

Results for Transformers from all folds



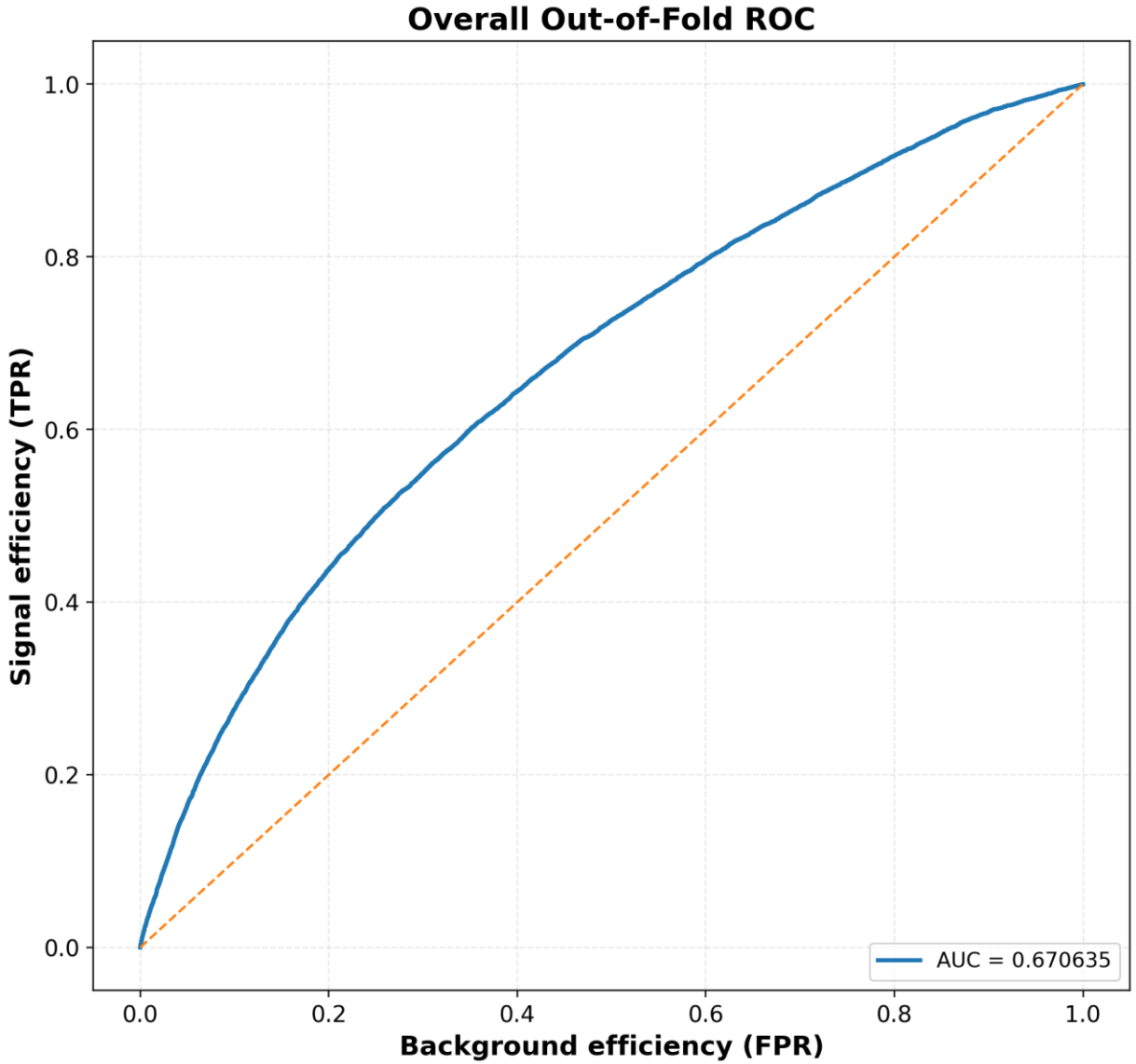
Transformers

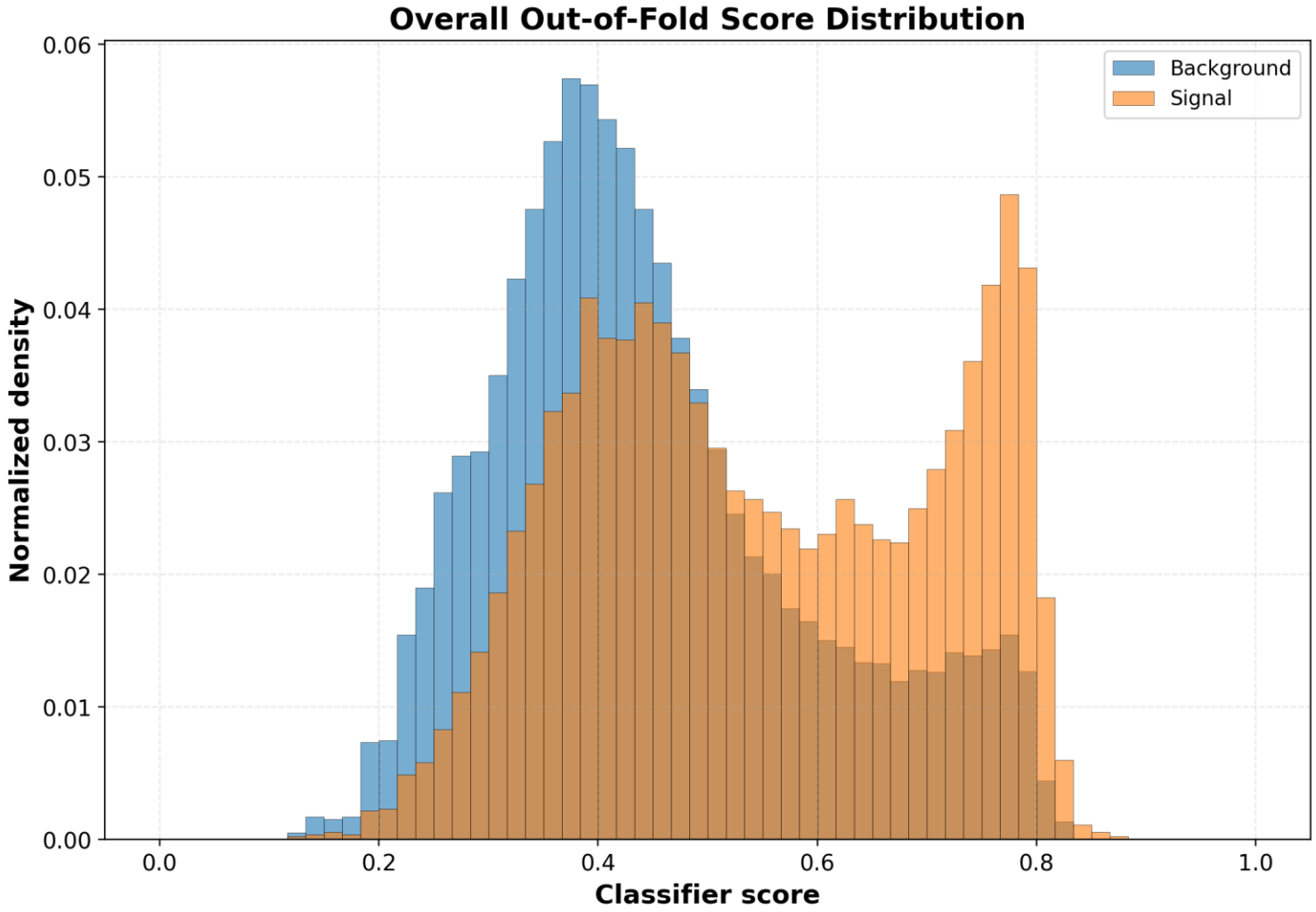
Comment: Your report for LSTM S/sqrt(S+b)=79, while for transformers it's 27(or even 19). Why?

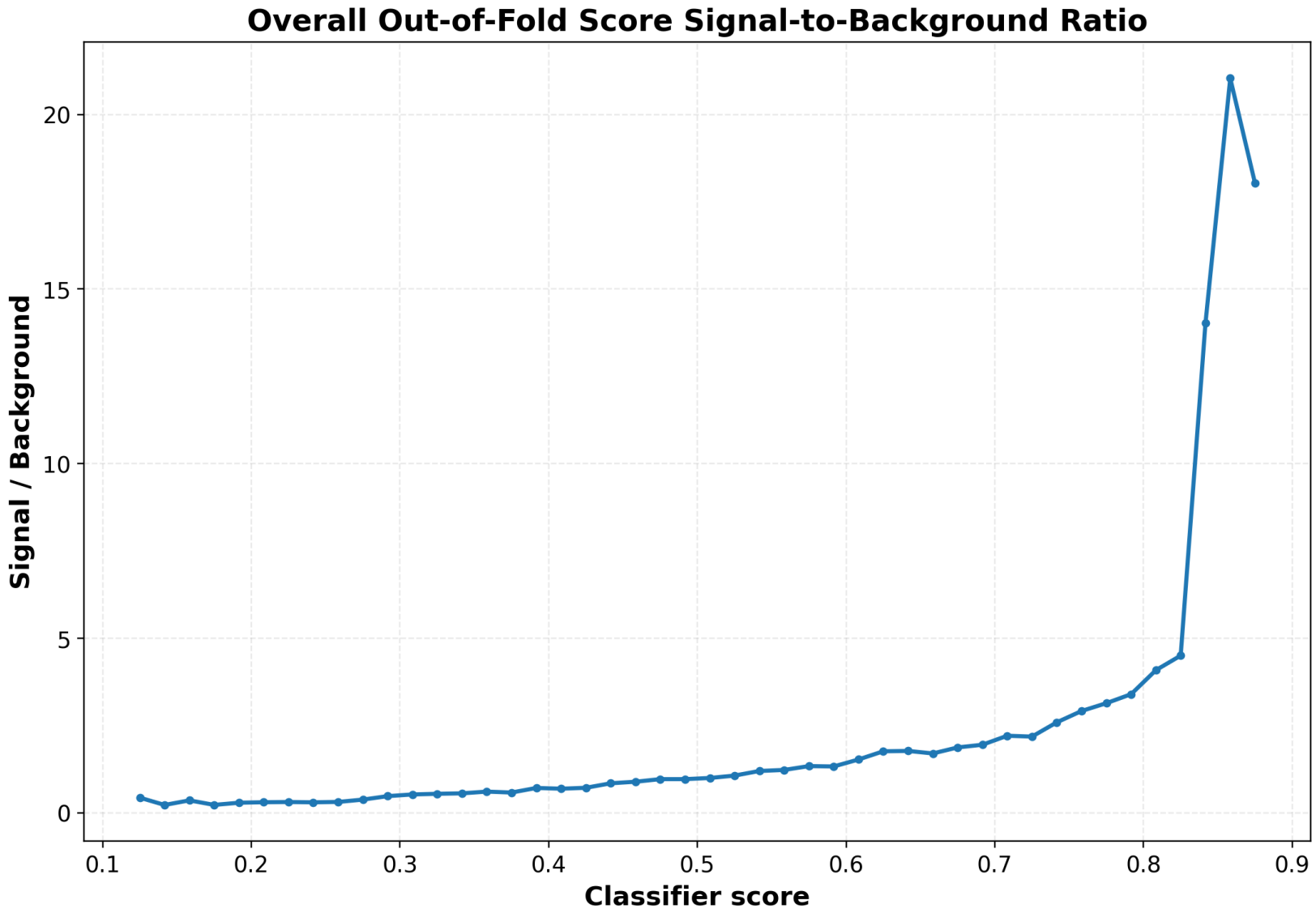
Explanation: Because I did not plot the Overall out of fold, it was because I forgot to put them in the slide.
The OOF value of 79 came from the **balanced-weight** run, while the fold plots are using **raw validation weights.**
So, the fold values around **19–20** are raw-weight per-fold validation results.

Balanced-weight OOF
- total signal weight = **12492**
- total background weight = **12492**
- best significance = **79.22**

Raw-weight OOF
total signal weight = **12492**
total background weight = **75074**
best significance = **42.93**


Overall Out-of-Fold ROC
Signal efficiency (TPR)
Background efficiency (FPR)
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 0.670635


Overall Out-of-Fold Score Distribution
Background
Signal
Normalized density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.0
0.2
0.4
0.6
0.8
1.0
Classifier score


Overall Out-of-Fold Score Signal-to-Background Ratio
Signal / Background
0
5
10
15
20
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Classifier score

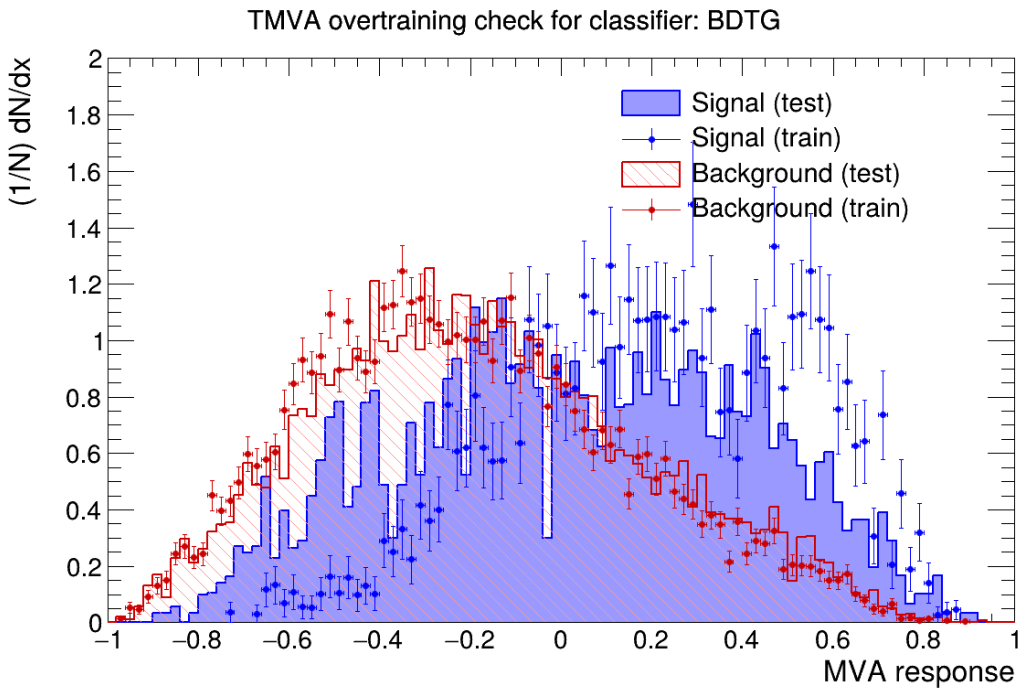
Comment: Current training statistics is limited. Therefore, you need a well defined and clean method to control overtraining for every tested algorithm. It would be nice to have such control results in your sildes.

Explanation:
For the control on overtraining in the same way for all methods:
use **k-fold / nested CV**, report only **out-of-fold validation results**, and check that performance is stable across folds.

Method	How overtraining is controlled via the script under testing the performance	Results to validate the check	Conclusion
LSTM	Uses fold-wise train/validation split, saves **train vs validation loss/AUC/accuracy**, and selects **best epoch** with early stopping. It also saves fold validation scores and cut-scan results.	**OOF combined results:** loss curve, AUC curve, accuracy curve, validation ROC, validation cut scan, best epoch.	"Overtraining is controlled by early stopping and by checking that train and validation curves remain close across folds."
Transformer	Uses fold-wise training, saves **train/validation loss, AUC, accuracy**, stores **best epoch**, and evaluates final ROC and cut scan on **validation raw weights**.	**Train vs validation curves, validation ROC, validation score distribution**, and **validation cut-scan plot for** the combined OOF result.	"Overtraining is controlled by early stopping and by requiring stable validation performance and stable cut-scan behaviour across folds."
Python ML	Uses **nested CV** with **outer and inner folds**, and reports **overall OOF AUC**, **mean fold AUC**, and **std fold AUC**. Stability across folds is the main control.	The summary table with **overall OOF AUC, mean/std fold AUC, weighted accuracy**, and **max significance.**	"For classical ML, overtraining is controlled by nested CV and by showing that the performance is stable across folds, not by training performance alone."
TMVA	Uses explicit **k-fold train/test split**, extracts **train and test scores**, and makes a **TMVA overtraining check plot** by comparing train and test response distributions.	**TMVA overtraining plots, k-fold ROC**, and models with visible train/test mismatch are retuned or rejected.	"For TMVA, overtraining is controlled directly by train–test response agreement; if the signal train/test shapes do not match, that TMVA setup is not accepted."

Method	CV / optimization setup	Number of optimization trials	Main searched hyperparameters
LSTM	outer_k=5, inner_k=3, seed 42; final training uses epochs=60, inner_epochs=20, patience=8, min_epochs=8, min_delta=5e-4	n_iters=8	hidden_size ∈ {32,64,96,128}, num_layers ∈ {1,2,3}, dropout ∈ {0,0.05,0.1,0.2,0.3}, head_dim ∈ {32,64,128}, bidirectional ∈ {False,True}, lr ∈ [1e-4,5e-3] log-uniform, weight_decay ∈ [1e-7,5e-3] log-uniform, batch_size ∈ {128,256,512,1024}, grad_clip ∈ {0,0.5,1,2}, scheduler_factor ∈ {0.3,0.5}, scheduler_patience ∈ {2,3,4}
Transformer	outer_k=5, inner_k=3, seed 42; final training uses epochs=80, inner_epochs=25, patience=12, max_batch=512	n_iters=8	d_model, n_heads, n_layers ∈ {4,5}, dropout ∈ {0.05,0.10,0.15,0.20}, mlp_mult ∈ {2,3,4}, lr ∈ [5e-5,3e-3] log-uniform, weight_decay ∈ [1e-6,5e-3] log-uniform, batch_size from batch_choices, grad_clip ∈ {0.5,1,2}, plus early stopping with min_delta=1e-4 and scheduler patience 2
Python ML	Nested CV with outer_k=5, inner_k=3, seed 42	n_iters=25 for most algorithms, but **SVC uses 12**	Search is algorithm-specific through **ParameterSampler;** examples in the code include n_estimators, learning_rate, max_depth, subsample, min_samples_leaf, max_leaf_nodes, etc., **depending on the classifier**
TMVA	kfold=5, seed 42, no inner nested CV in this script; each trial is evaluated across all 5 folds	n_iters=20	Method-dependent random search over the TMVA parameter space defined in method_space() / draw_params(), then best setting is chosen by mean k-fold AUC

TMVA



Comment: In the "TMVA overtraining check" slide there is a clear overtraining for signal. How are you going to correct it?

I will not use this TMVA result as final in its current form. The correction is to decrease the effective flexibility of the BDTG, retrain under stricter regularization, and the validate with a train–test criterion.

I will try to see if reducing tree depth, and increasing the minimum leaf size, and repeat the training with cross-validation.

For all of it, I will need time to redo the algorithms to avoid the overtraining. But I am not sure, will this overtraining can be eliminated at all…

R

The package installation is under progress, no updates on it yet.. I will be working with it too, sorry but I did not take updates in Eduard. I will write him early morning on Wednesday.

National Research
Tomsk
State
University

Лаборатория
анализа данных
физики высоких энергий
Томского
государственного
университета

Thank you for your attention!!!