GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

TECHNISCHE UNIVERSITÄT DRESDEN

INSTITUTE OF NUCLEAR AND PARTICLE PHYSICS

ErUM-FSP T02 ATLAS

ATLAS EXPERIMENT

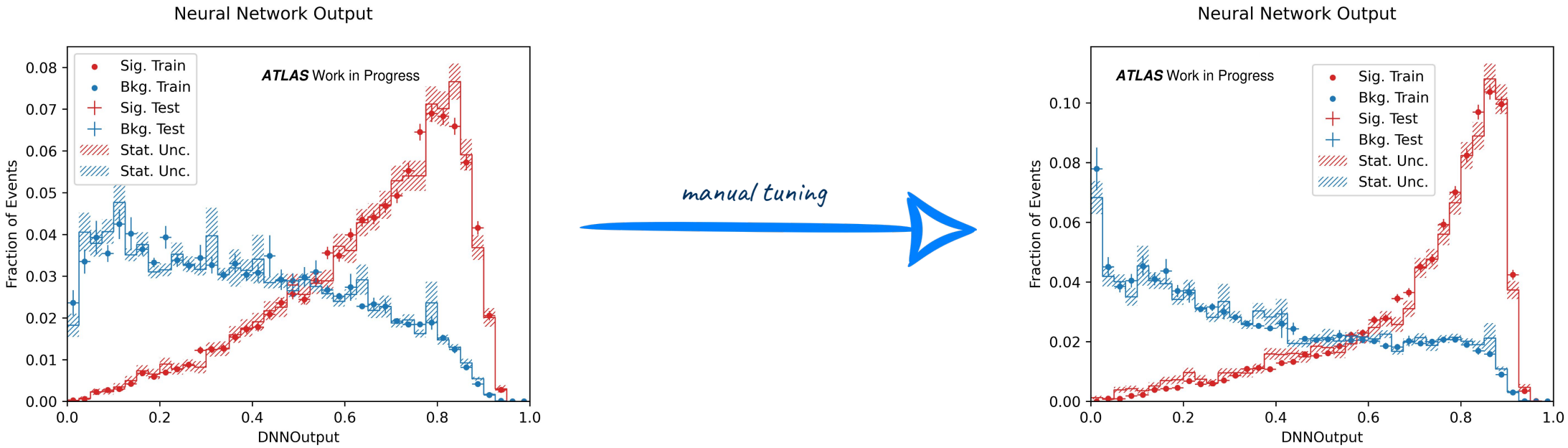
DRESDEN concept

OPTIMA: a framework for automated hyperparameter optimization and input variable selection for neural networks

Erik Bachmann – DPG SMuK Karlsruhe, March 6, 2024

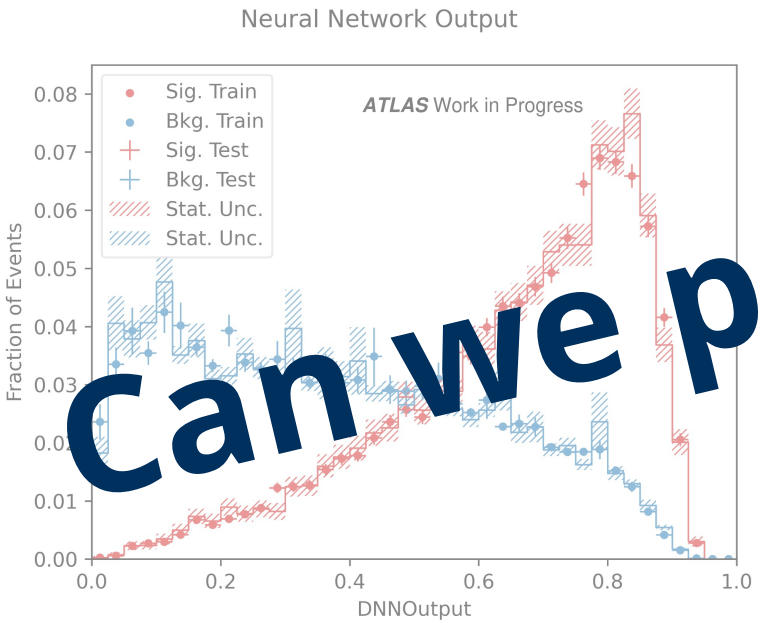
Motivation

- Artificial neural networks (ANNs) are central part in many analyses
- Performance of the ANN determined by:
  - Training dataset:
    - size (fixed)
    - input variables (optimizable)
  - ANN's Hyperparameters (optimizable)



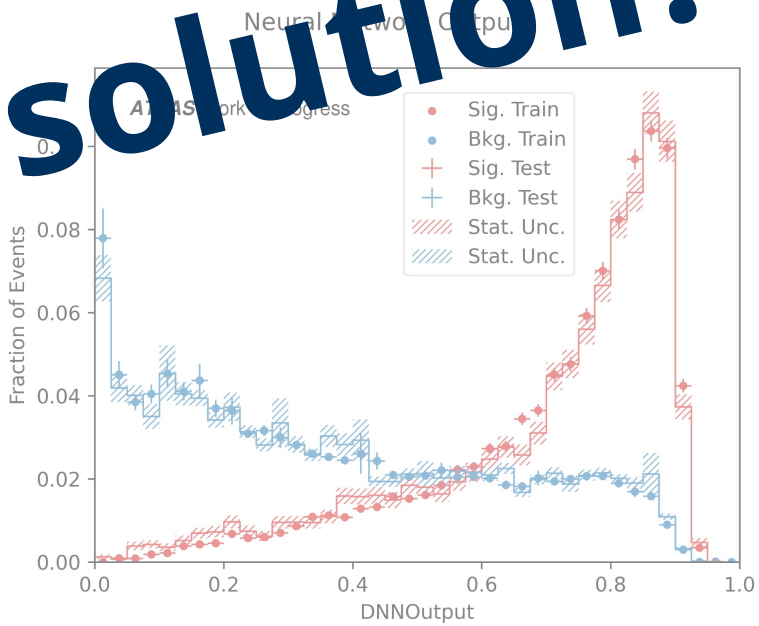
Motivation

- Artificial neural networks (ANNs) are central part in many analyses
- Performance of the ANN determined by:
  - Training dataset:
    - size (fixed)
    - input variables (optimizable)
  - ANN's Hyperparameters (optimizable)



manual tuning

Can we provide a solution?



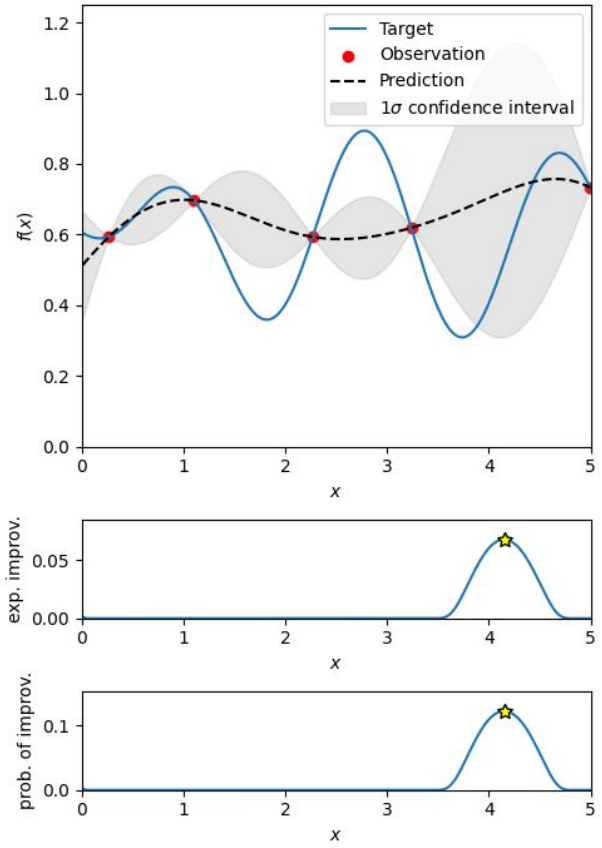
OPTIMA

Highlights:

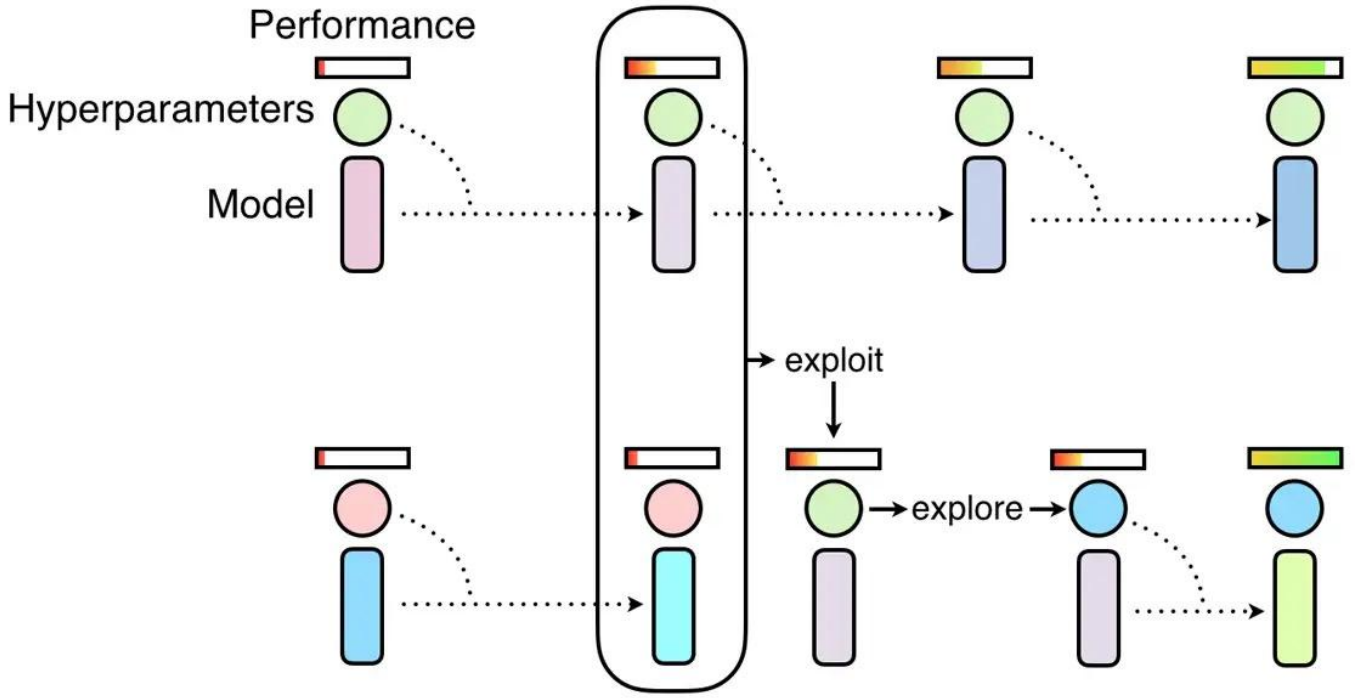
- *Ease of use:*
  - algorithms for automated hyperparameter optimization and input variable selection
  - built-in functionality for multilayer perceptrons and classification
- *Generality:*
  - applicable to all Keras and Lightning models for supervised learning
  - all metrics computable from ANN output and target labels are supported
  - extensively configurable via config-file
- *Scalability:*
  - Based on Ray and Tune for distributed execution

Hyperparameter Optimization Algorithms

Bayesian Optimization (Optuna)

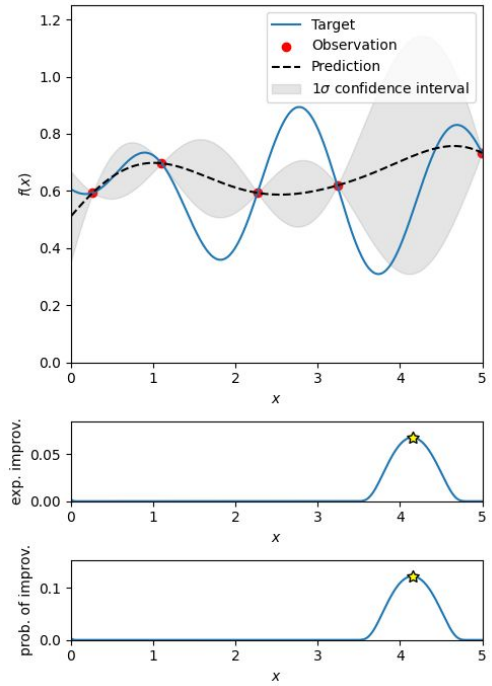


Population Based Training (Tune)



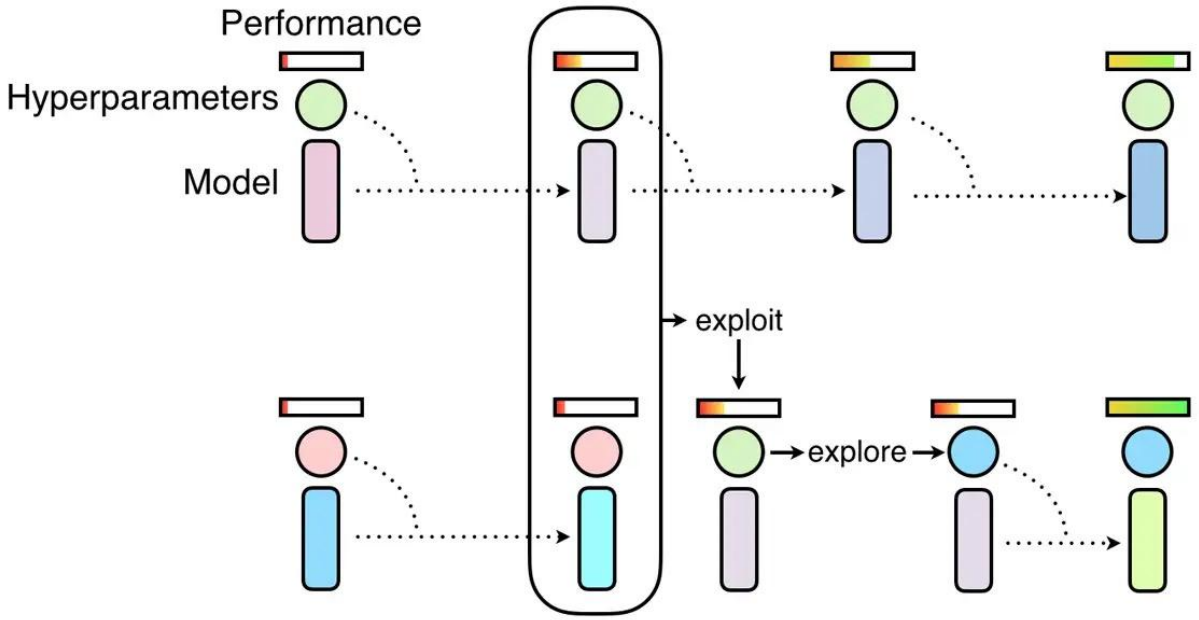
Hyperparameter Optimization Algorithms

Bayesian Optimization (Optuna)



➔ *supports all hyperparameters*
➔ *results are fixed values*

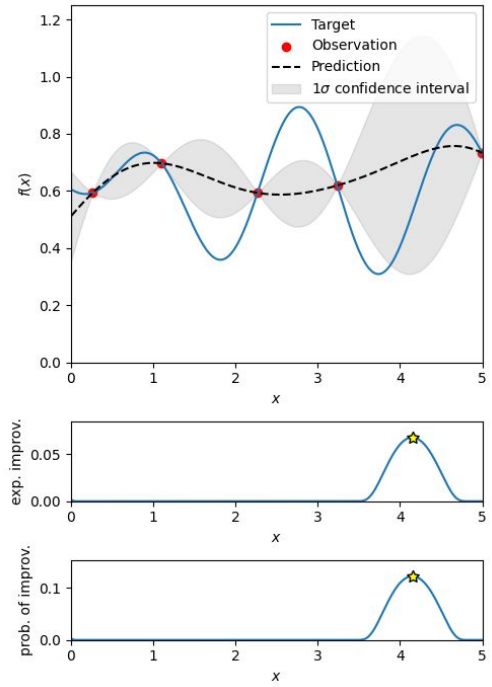
Population Based Training (Tune)



➔ *supports only changeable hyperparameters*
➔ *results in hyperparameter schedules*

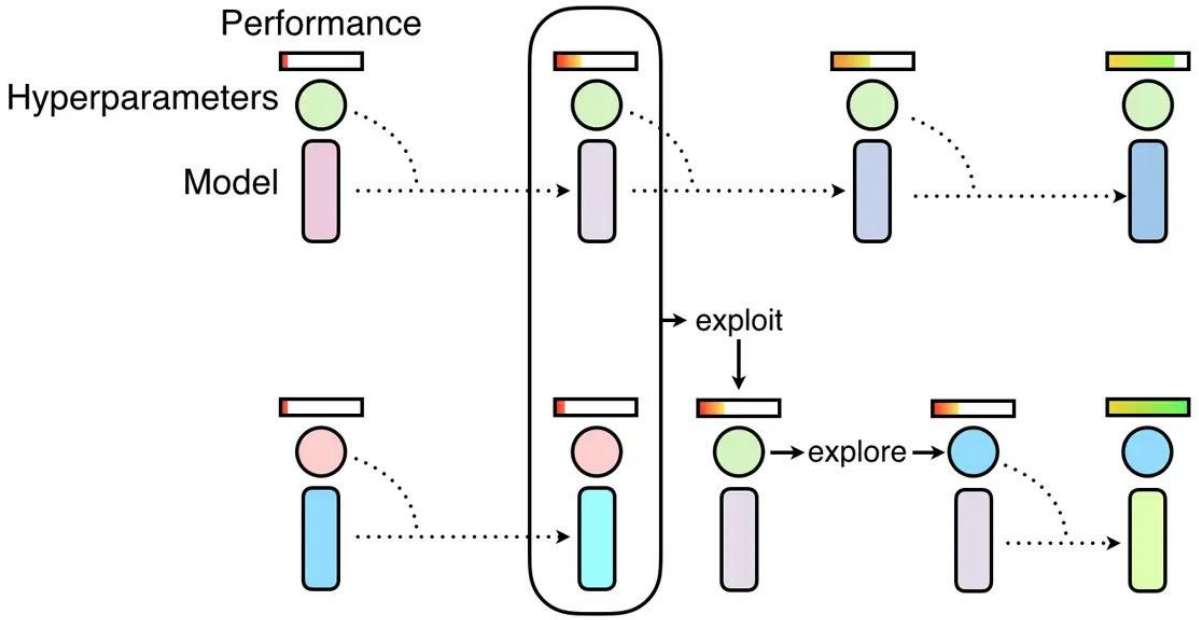
Hyperparameter Optimization Algorithms

Bayesian Optimization (Optuna)



+

Population Based Training (Tune)



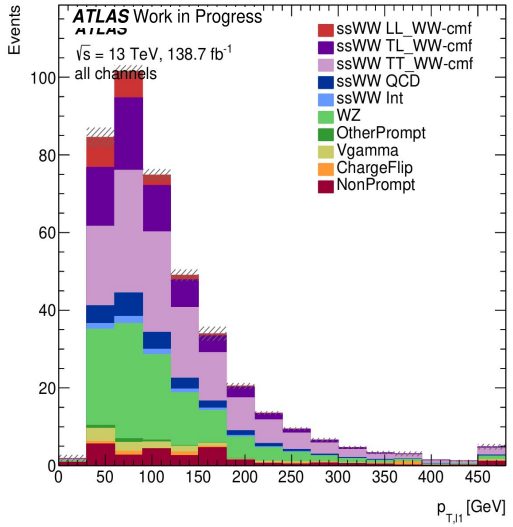
Optimize with Bayesian optimization first, then fine-tune with PBT

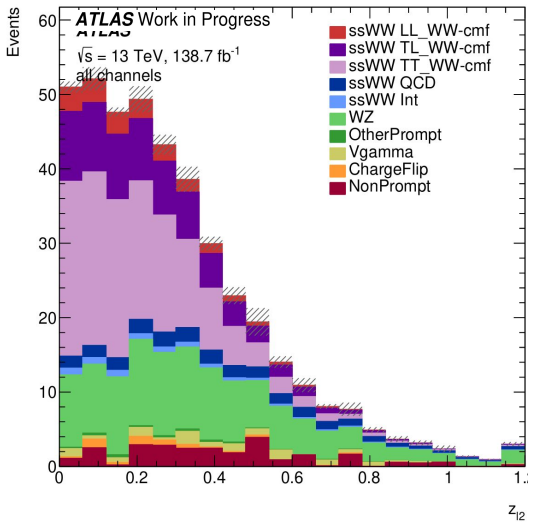
Input Variable Selection

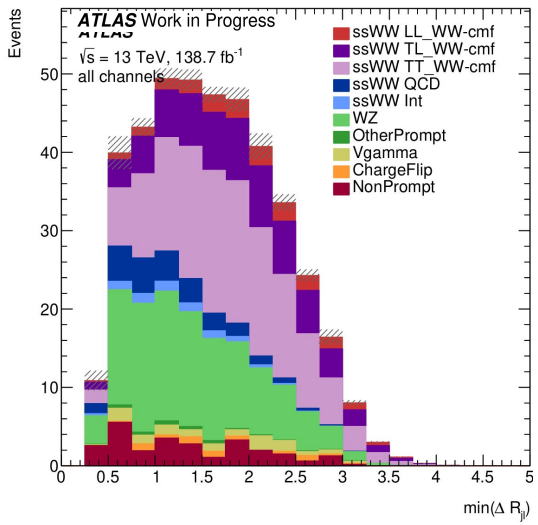
Motivation:

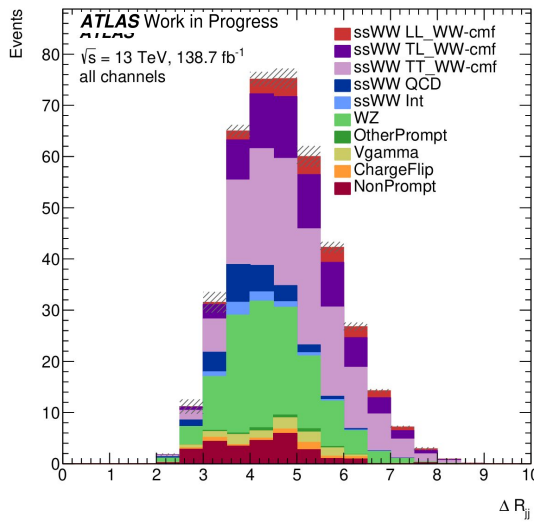
- improve model performance
- reduce overfitting (MLPs)
- reduce modeling uncertainties

Importance of each variable depends on correlations with others
→ hard to judge beforehand







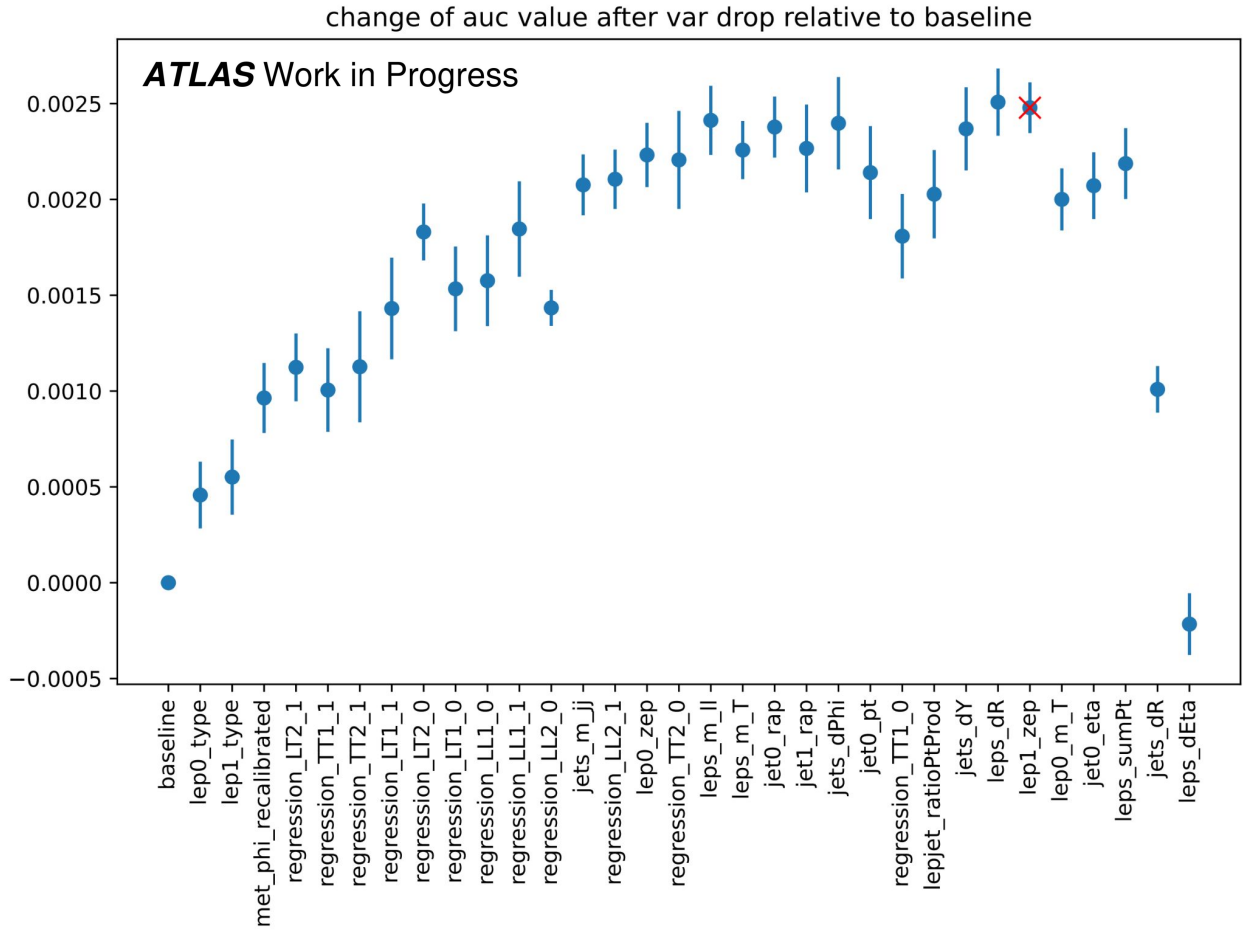


Input Variable Selection

Idea: Backwards Elimination

- Baseline ANN with all input variables
- Iteratively remove unimportant/redundant input variables
- Use “early stopping” for termination

How to evaluate the importance of an input variable?

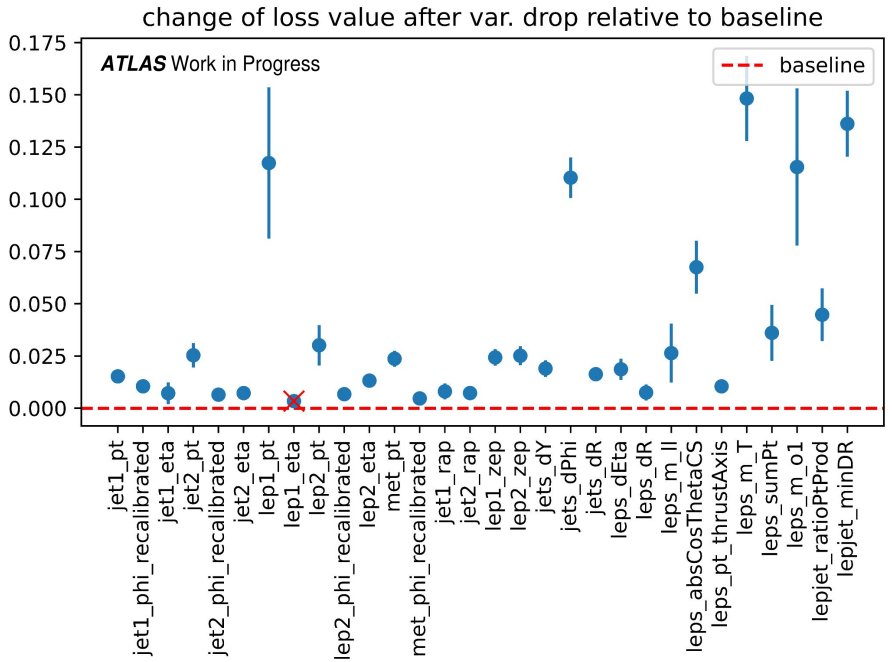


Input Variable Selection

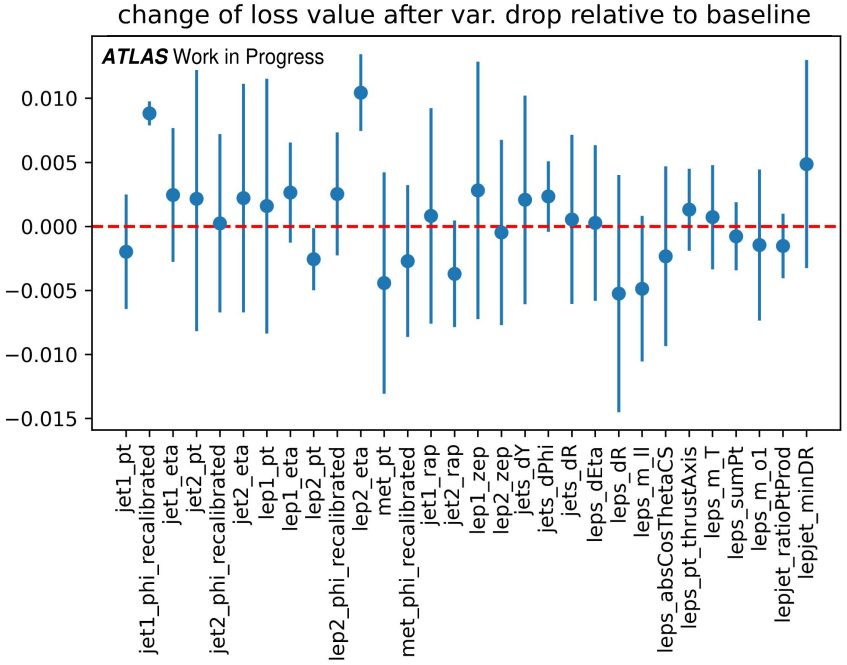
Idea: Backwards Elimination

- Evaluate the importance of an input variable:
  - random shuffling between events → *fast, but no correlations*
  - remove from training set and retrain → *computationally expensive, but finds redundant variables*

Shuffle



Retrain

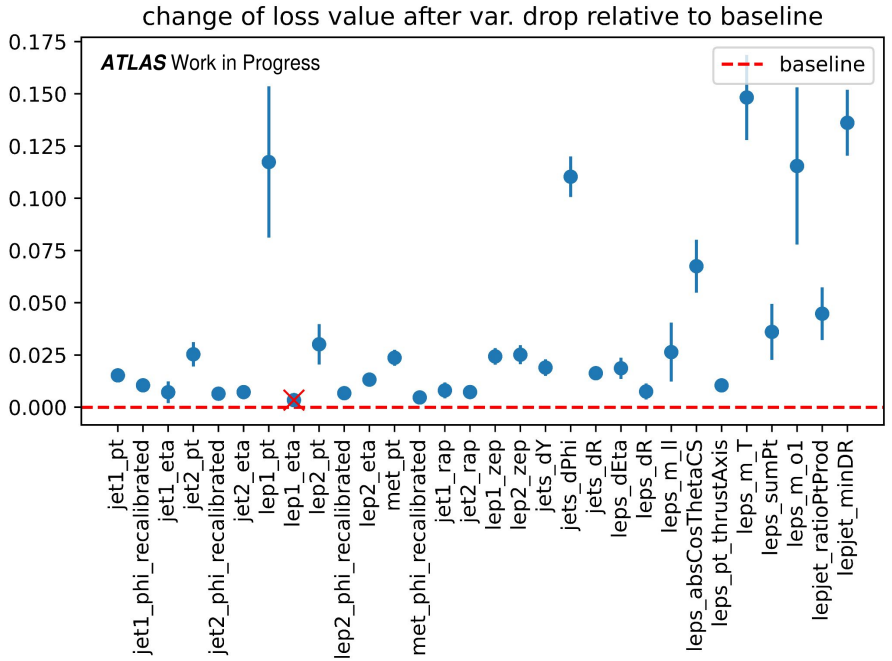


Input Variable Selection

Idea: Backwards Elimination

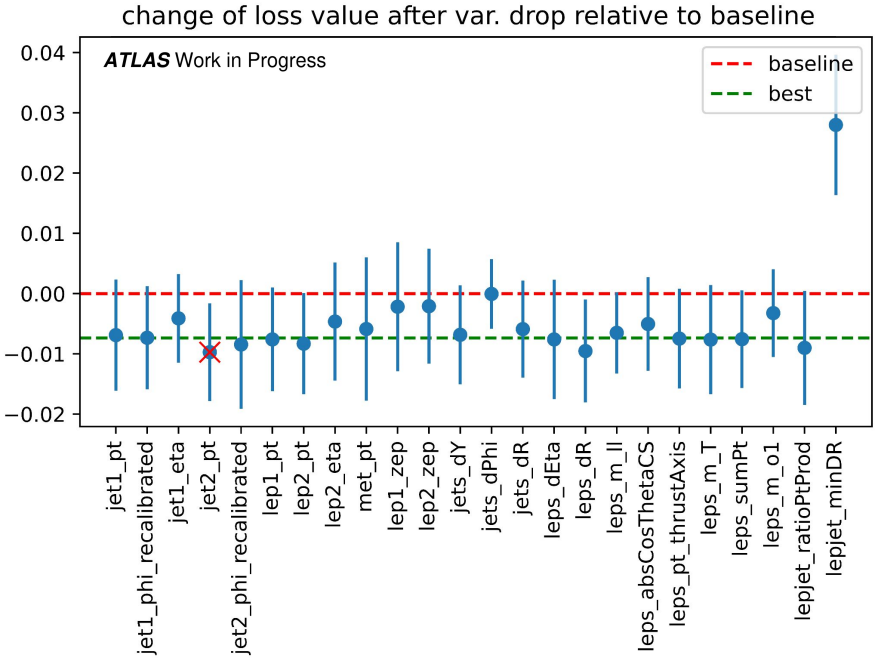
- Combined method: *Shuffle* for the first iterations, then *retrain*

Shuffle

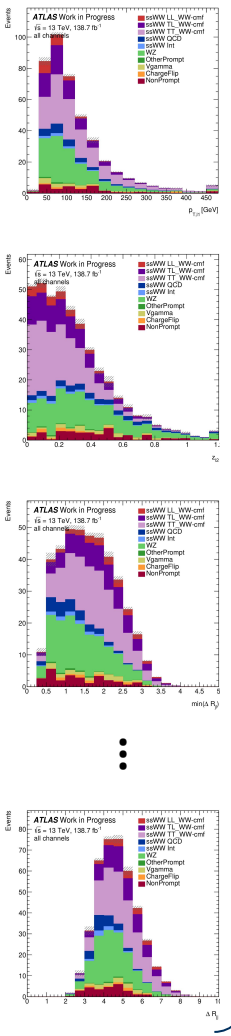


+

Retrain



OPTIMA

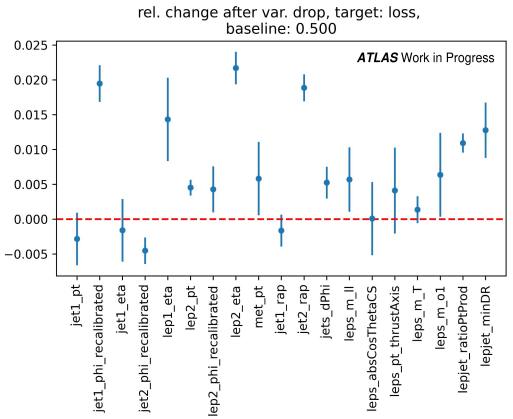


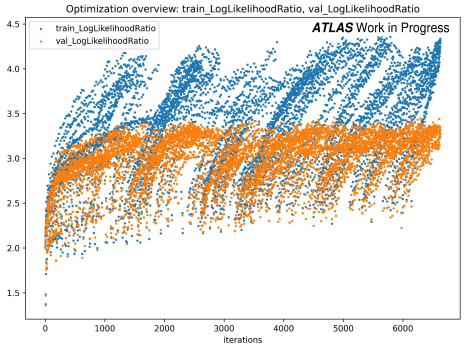
Input Variable Selection → Bayesian Optimization → PBT

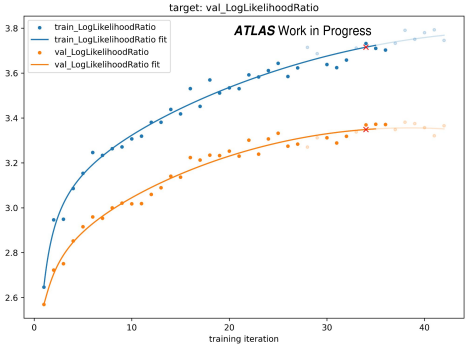
- pre-optimization (BO) with all input variables
- optimize input variables with hyper-parameters from pre-optimization

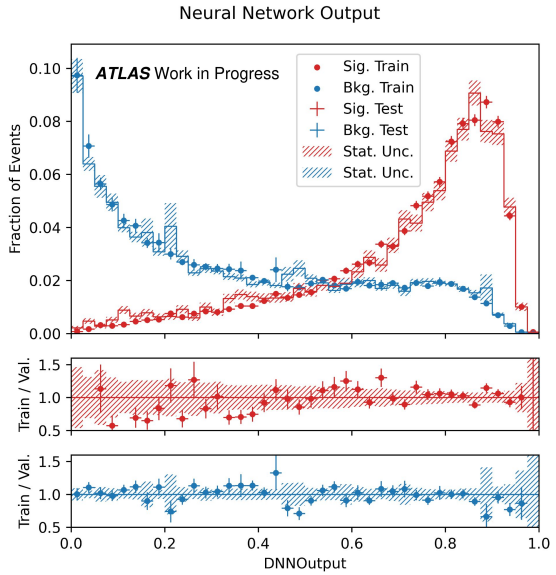
- find best hyper-parameters with optimized variables

- does HP-schedule improve result?









How to try OPTIMA?

- Published on PyPI:
  - pip install optima-ml[keras]
  - pip install optima-ml[lightning]
- Source code and detailed usage instructions available on CERN gitlab: gitlab.cern.ch/atlas-germany-dresden-vbs-group/optima
- API documentation at optima-docs.docs.cern.ch
- Roadmap:
  - Support for Keras 3 → Cross-framework support
  - Support for HTCondor

Any feedback is welcome! Send me a mail at erik.bachmann@tu-dresden.de

atlas-germany-dres... / OPTIMA

OPTIMA: an Optimization Platform for Tuning Input Variables and Model Parameters

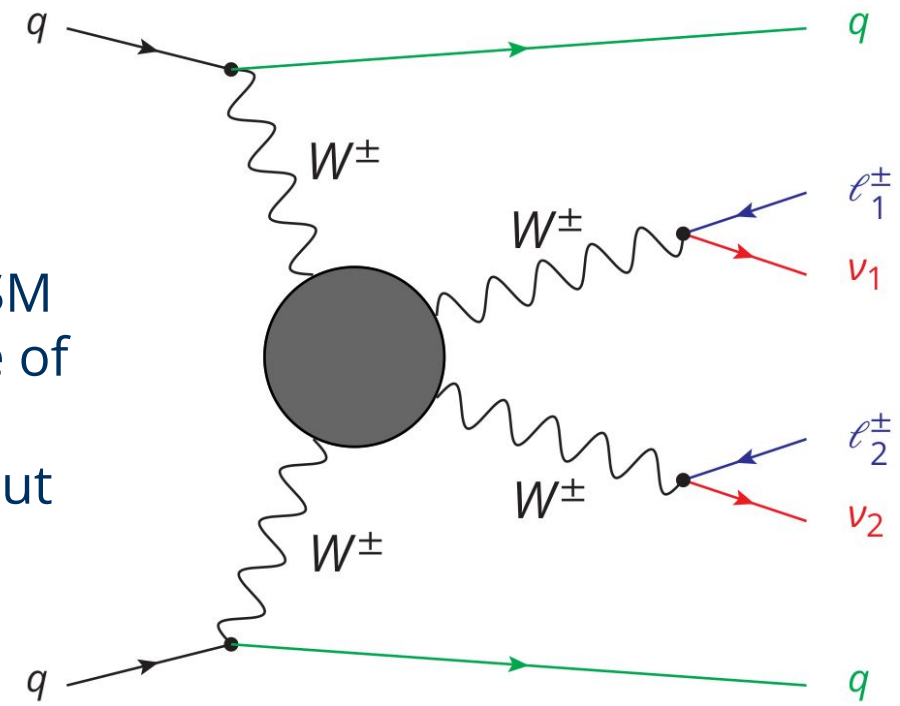
OPTIMA is a framework to perform highly parallelized hyperparameter optimization and input variable selection of arbitrary Keras or Lightning neural networks for supervised learning tasks.

Table of Contents

Thank you for your attention!

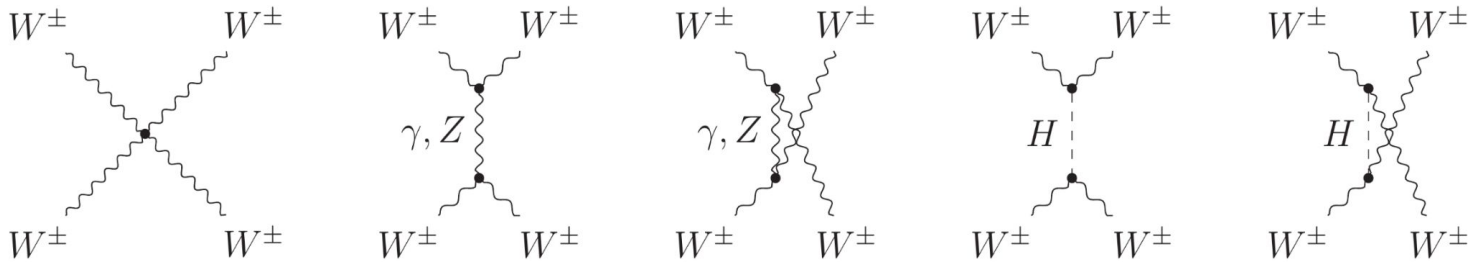
APPENDIX

Application: Polarization in $W^{\pm}W^{\pm}jj$



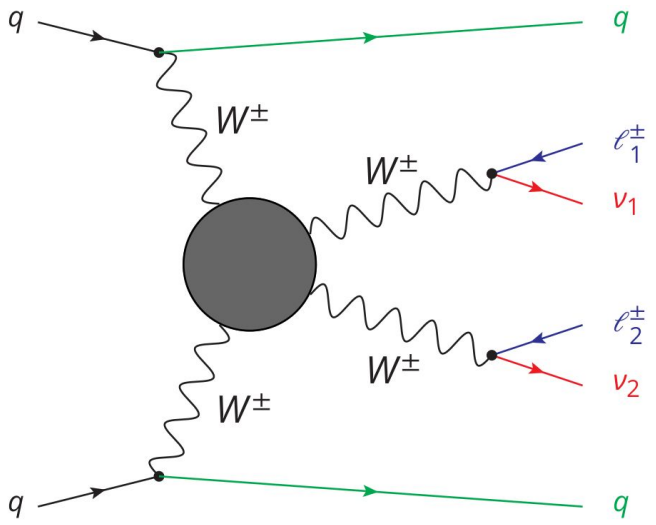
Motivation:

- Self-interaction of $W^{\pm}$-bosons
  → Probe innermost gauge symmetry structure of the SM
- Longitudinal polarization of the VB direct consequence of the EWSB
- Scattering of longitudinally polarized $W^{\pm}$-bosons without the Higgs violates unitarity



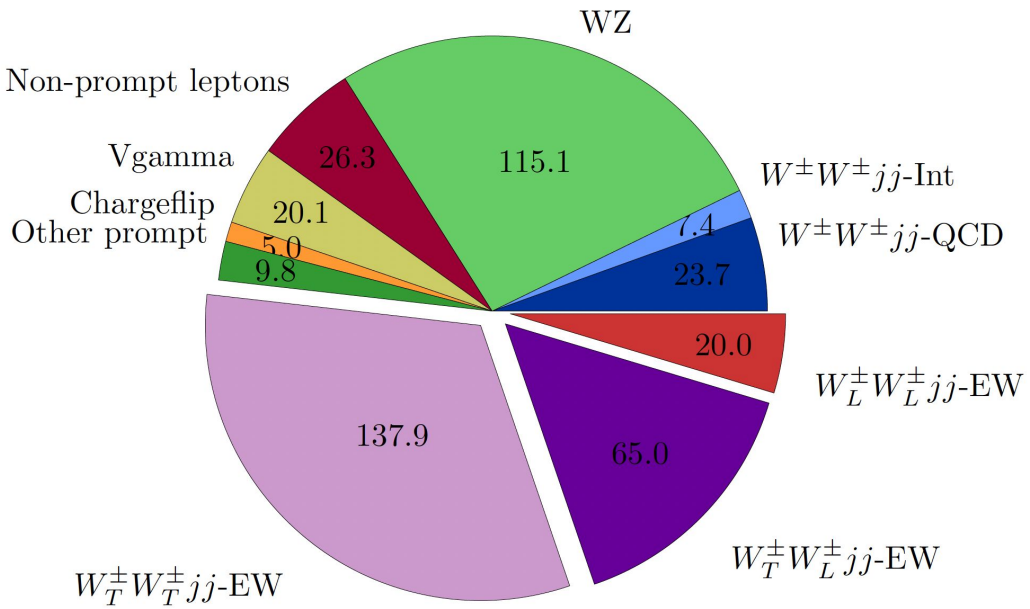
Application: Polarization in $W^{\pm}W^{\pm}jj$

- Use the full Run-2 data (139 fb^{-1})
- Multilayer perceptron to classify $W_L^{\pm}W_L^{\pm}$ vs. Bkg.
- Extract discovery significance for LL from fit on MLP output



ssWW signal region

Exactly two Tight leptons with identical electrical charge
Electrons pass the charge flip rejection (ECIDS)
ee-channel: $\lvert\eta\rvert < 1.37$, $\lvert m_{ee} - m_Z\rvert > 15\,GeV$
$m_{ll'} > 20\,GeV$
$E_T^{miss} > 30\,GeV$
At least two jets Leading jet: $p_T > 65\,GeV$ Subleading jet: $p_T > 35\,GeV$
b-jet veto
$\lvert\Delta y_{jj}\rvert > 2$
$m_{jj} > 500\,GeV$



Stange, M. (2023). Machine Learning Application for Single Boson Polarization Measurement in Same-Charged WW Scattering Within the Atlas Experiment [Conference presentation]. DPG SMuK 2023, Dresden, Germany.

Application: Polarization in $W^{\pm}W^{\pm}jj$

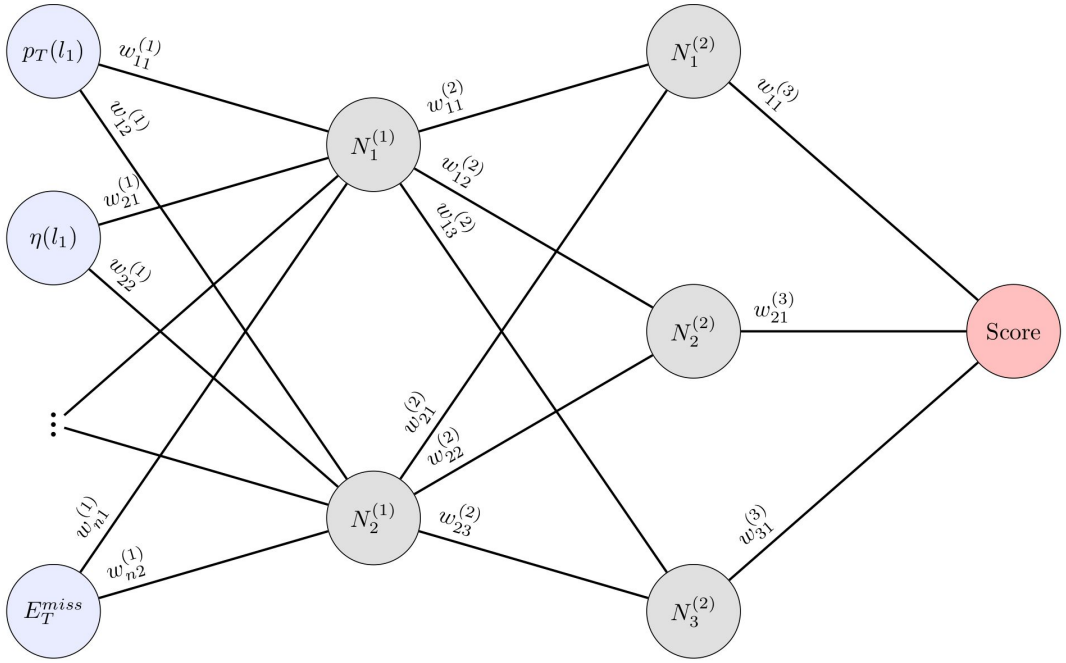
Dataset:

- $\approx$ 250k MC events, 80% used for training
- 30 high- and low-level input variables

Hyperparameter search space:

- number of hidden layers
- number of neurons per hidden layer
- dropout rate
- strength of L1 and L2 regularization
- batch size
- parameters of ADAM optimizer
- weight of signal events

→ *11 free parameters*



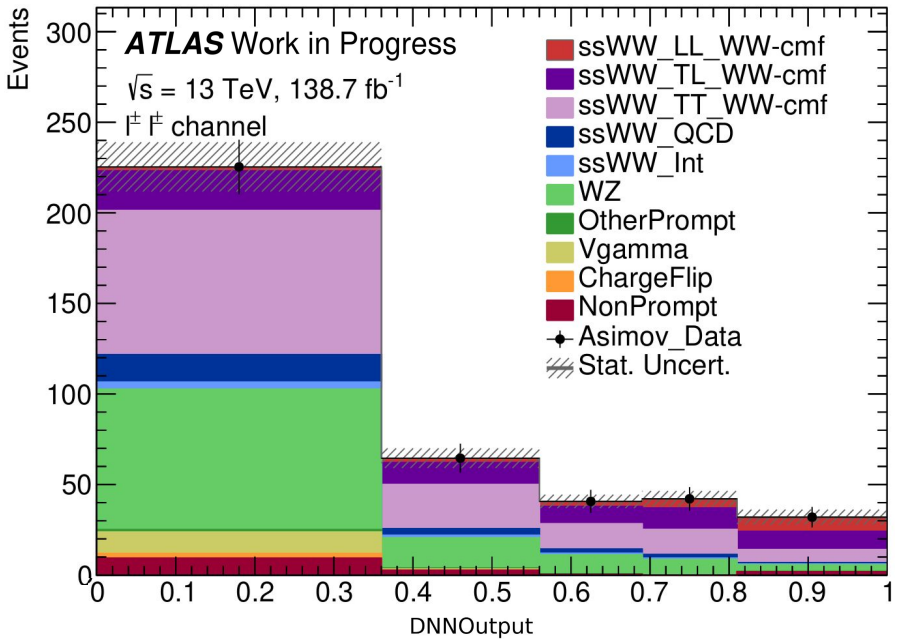
Application: Polarization in $W^{\pm}W^{\pm}jj$

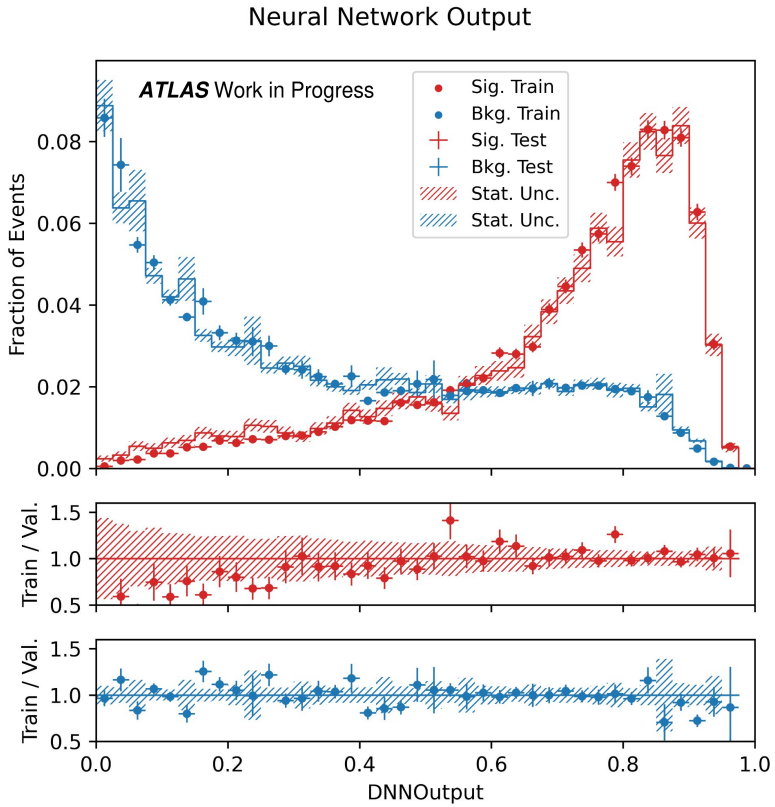
Optimization target:

binary cross-entropy loss $L(y, \hat{y}) = -[y \ln \hat{y} + (1-y) \ln (1-\hat{y})]$

Pre-optimization (all 30 input variables):

Stat.-only significance: $Z_{stat} = 1.33$

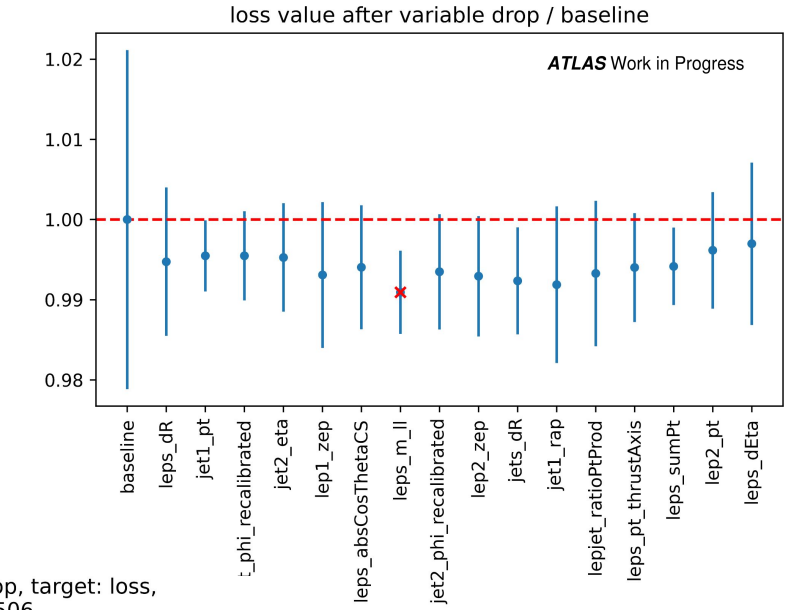


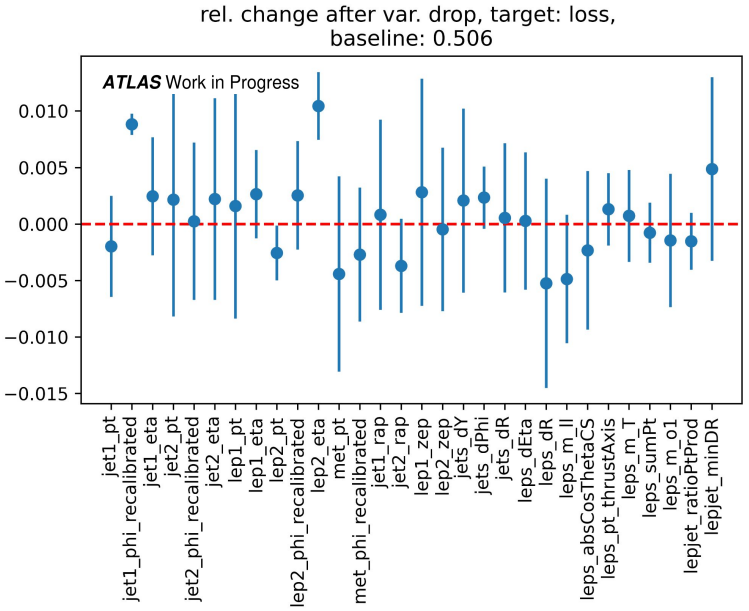


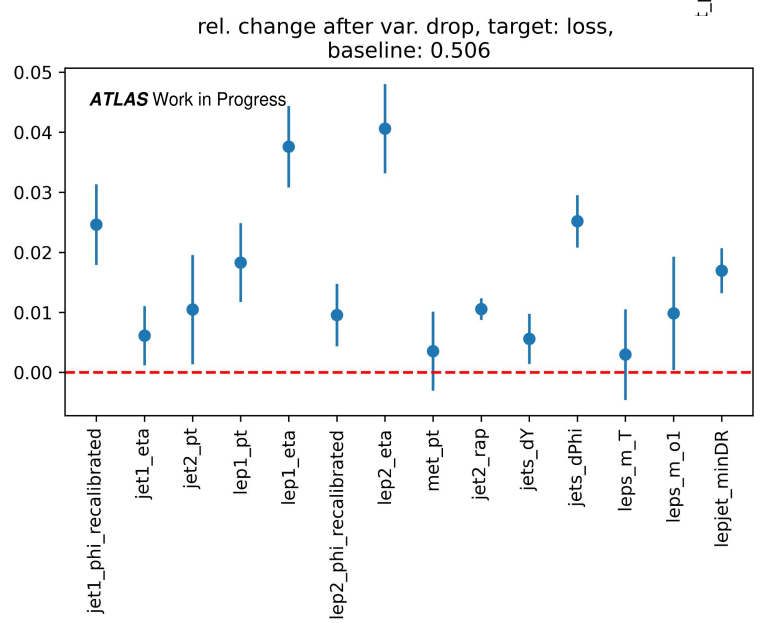
Application: Polarization in $W^{\pm}W^{\pm}jj$

Input variable selection:

- stopped when loss after drop was worse than baseline
- revert to best iteration
  → 8 variables dropped





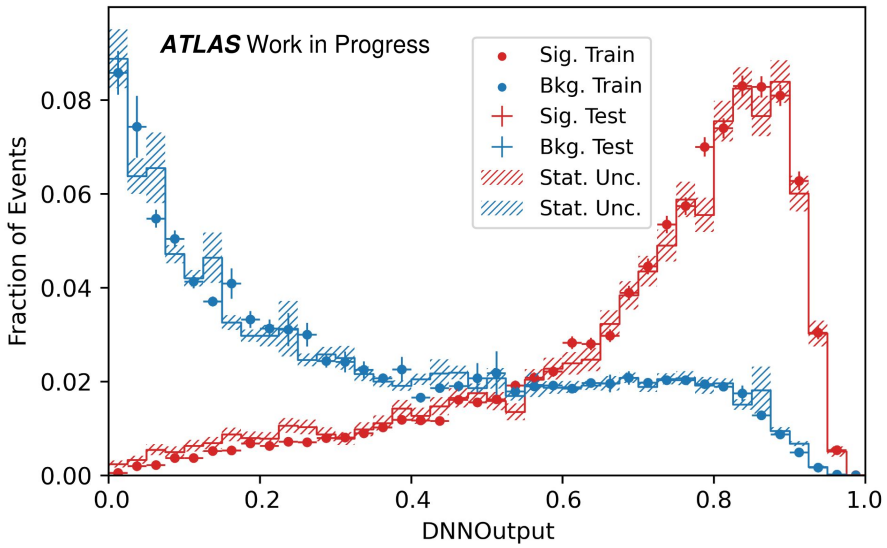


Application: Polarization in $W^{\pm}W^{\pm}jj$

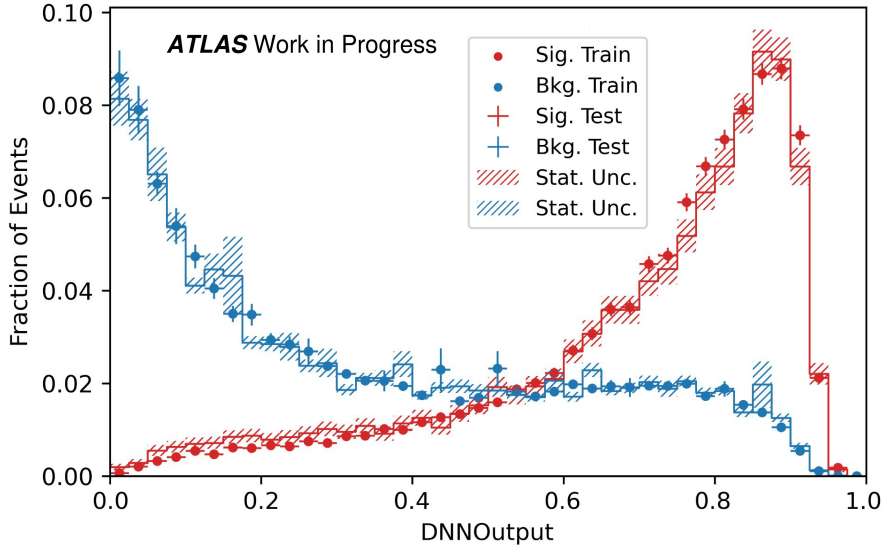
Input variable selection:

- stopped when loss after drop was worse than baseline
- revert to best iteration
  → 8 variables dropped
- smoother output distribution after main optimization

30 input variables



22 input variables

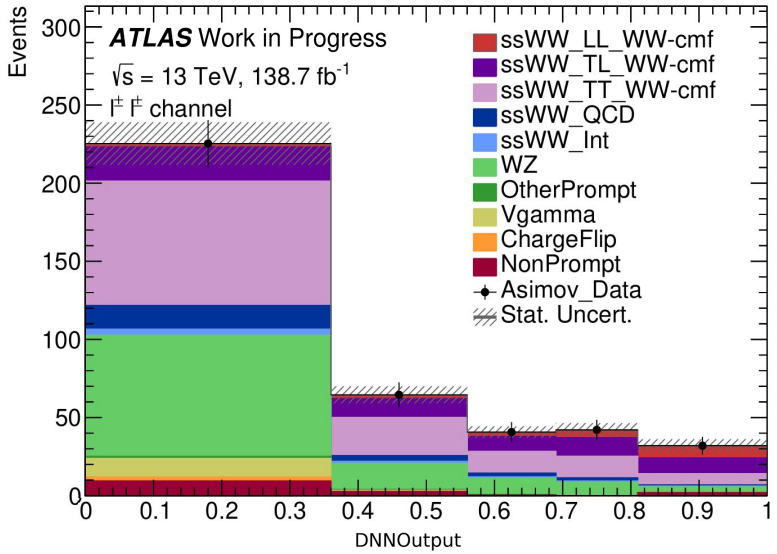


Application: Polarization in $W^{\pm}W^{\pm}jj$

Input variable selection:

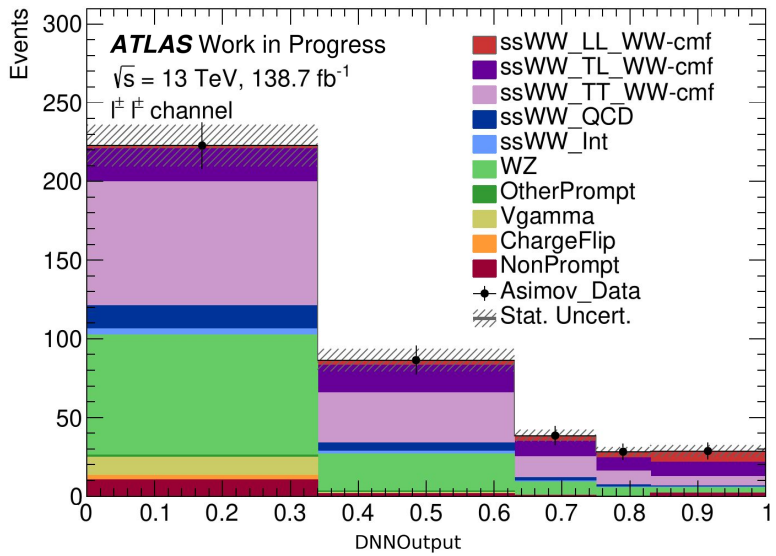
- stopped when loss after drop was worse than baseline
- revert to best iteration
  → 8 variables dropped
- smoother output distribution after main optimization

30 input variables



Stat.-only fit: $Z_{stat} = 1.33$

22 input variables

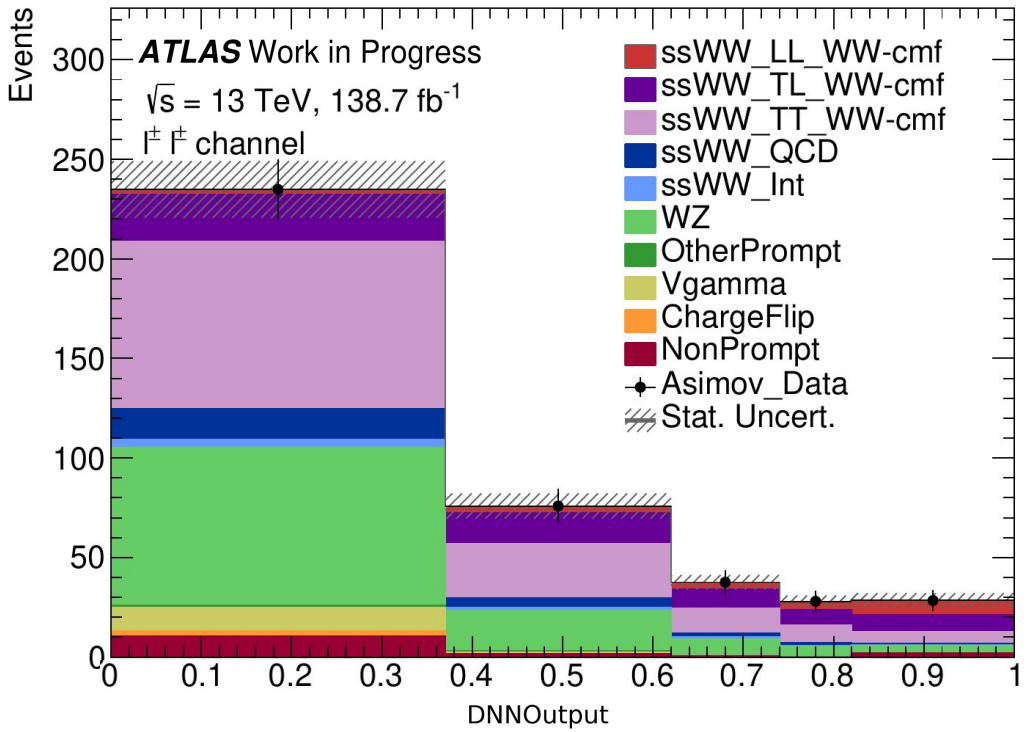


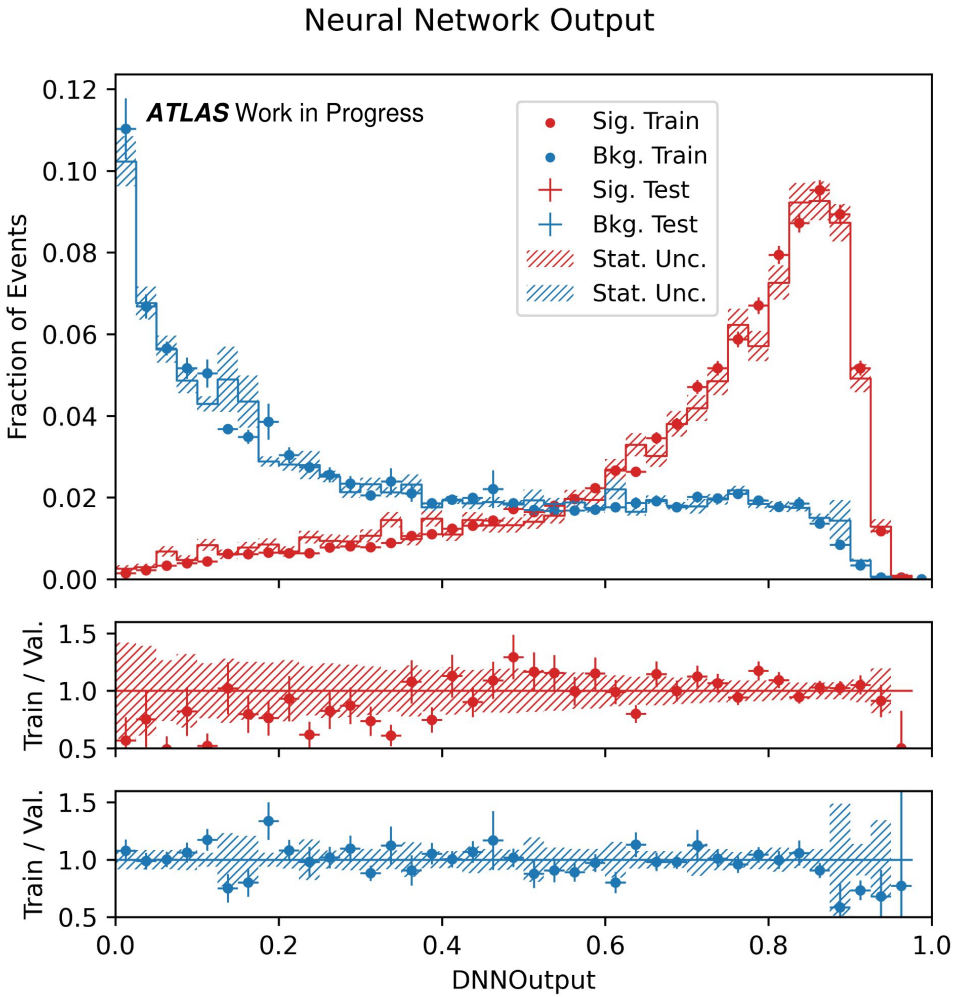
Stat.-only fit: $Z_{stat} = 1.37$

Application: Polarization in $W^{\pm}W^{\pm}jj$

Population Based Training:

Stat.-only significance: $Z_{stat} = 1.39$





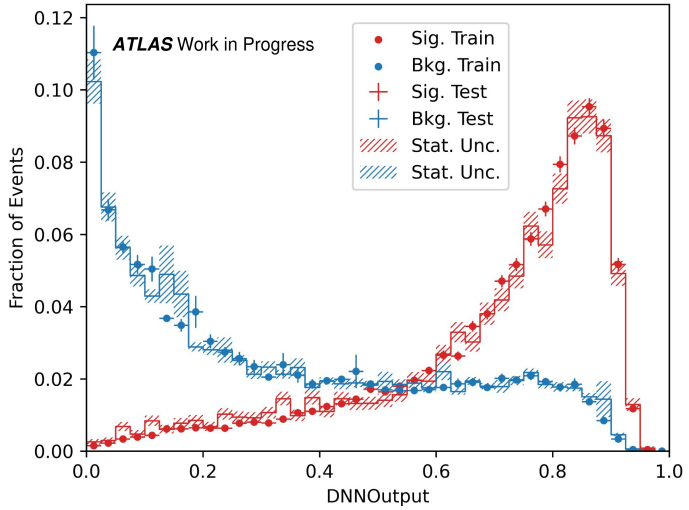
Application: Polarization in $W^{\pm}W^{\pm}jj$

Alternative optimization target: *Stat.-only Asimov Log-Likelihood ratio*

$$\log \frac{\mathcal{L}_B}{\mathcal{L}_{S+B}} = -2 \sum_{i=1}^{N} \left[(S_i + B_i) \log \frac{B_i}{S_i + B_i} + S_i \right]$$

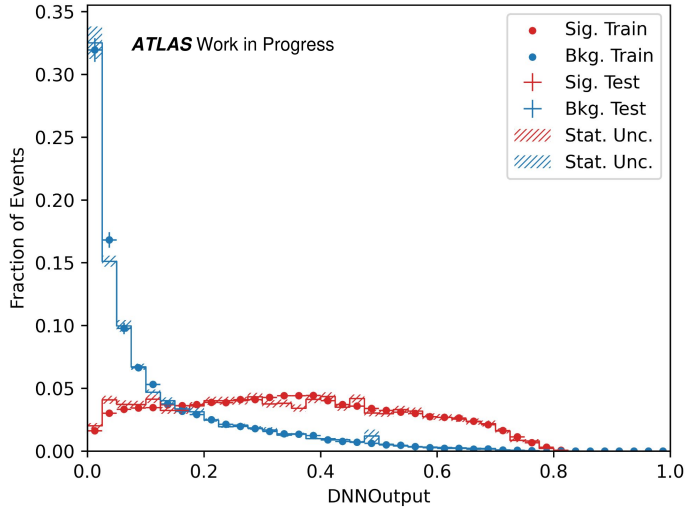
Significantly different output shape, but comparable performance

Binary Cross-entropy Loss



Stat.-only fit: $Z_{stat} = 1.39$

Log-Likelihood Ratio



Stat.-only fit: $Z_{stat} = 1.40$

Overtraining conditions

Comparing metrics on training and validation datasets allows the definition of overtraining conditions