Hyperparameter optimization of Adversarial Neural Networks in the tW dilepton channel using the ATLAS detector

25. März 2019

Christian Kirfel

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

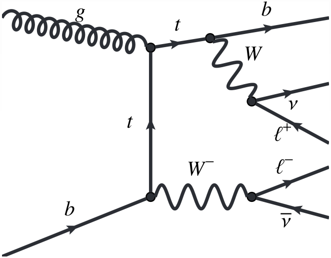
ATLAS EXPERIMENT

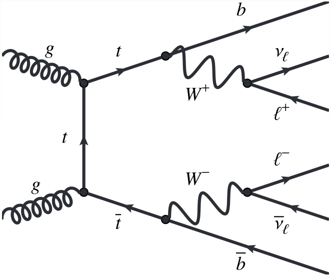
UNIVERSITÄT BONN

Outline

- tW and $t\bar{t}$ separation
- Artificial neural networks and adversarial neural networks as a possible solution
- Introduction to hyperparameters
- Preliminary training results for an adversarial neural network

tW and $t\bar{t}$ separation

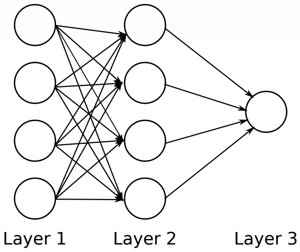




- Problem: Cross-sections of tW about 10 times smaller than $t\bar{t}$
- Interference in NLO order
- Instead of applying cut $\longrightarrow$ Neural networks

Neural Networks

- Built of layers of simple processors called nodes
- Each node is connected to each node in the previous and next layer
- Connection: linear function with a weight ω and a bias b and a non linear activation function σ
- Learning accomplished by backpropagation



Loss function and backpropagation

- Loss = performance estimation, deviation of the model from the truth tag

$$\mathcal{L} = -(y \log p + (1-y)\log(1-p))$$

- Backpropagation estimates the parameters' impact on the cost function using the partial derivatives

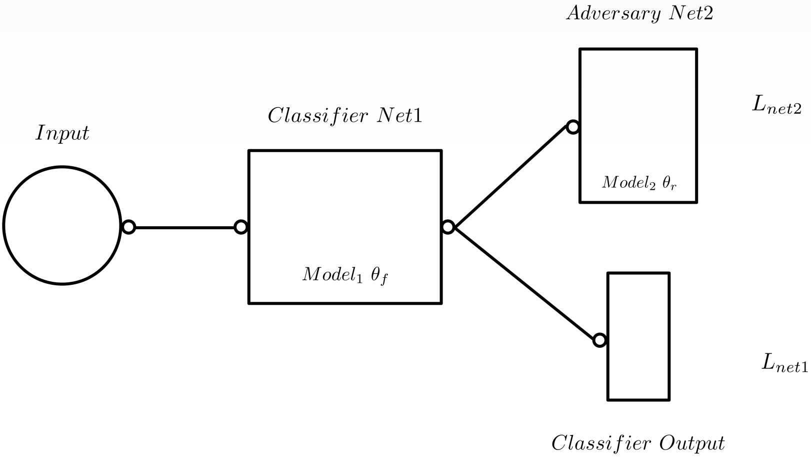
$$\frac{\partial \mathcal{L}}{\partial a_k^{L-1}} = \sum_{j=1}^{N} \frac{\partial z_j^L}{\partial a_k^{L-1}} \frac{\partial a_j^L}{\partial z_j^L} \frac{\partial C}{\partial a_j^L}$$

- Parameter update following negative gradient

Adversarial Neural Networks

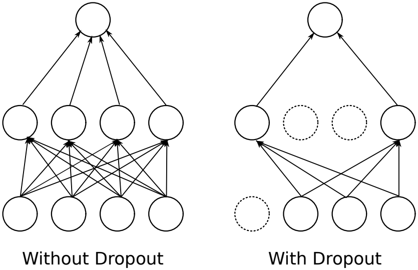
- Neural networks have no info on systematic uncertainties
- Introduction of a second, adversarial network classifying between nominal and systematic
- Combined loss function
  $\mathcal{L}_{adversarial}(\theta_f, \theta_t) = \mathcal{L}(\theta_f) - \mathcal{L}(\theta_f, \theta_r)$
- Network 1: signal/background separation
- Network 2: nominal/systematic separation
- Expectation: network 1 succeeds, network 2 fails

Setup of the ANN

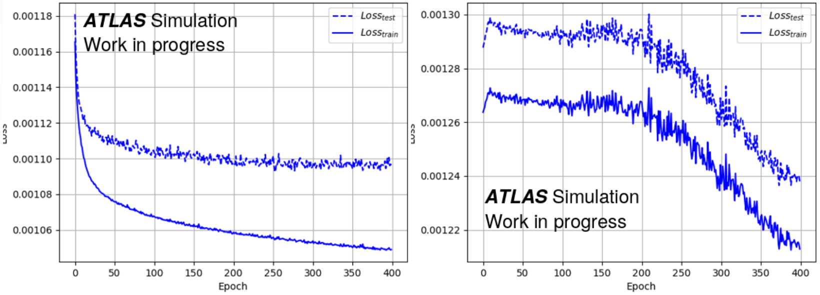


Dropout

- Long training and complicated architecture $\rightarrow$ focus on sub-dominant features or mask of the training set
- Removes a percentage of nodes for each training step
- Disincentives mask of the set or wrong features



Impact of dropout on the loss function



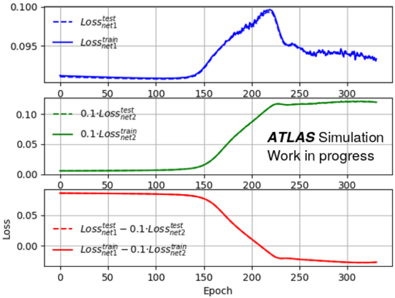
Loss curve for dropout = 0.1 Loss curve for dropout = 0.8

- For low dropout the training saturates fast
- High dropout makes the losses more linear as the training keeps on running

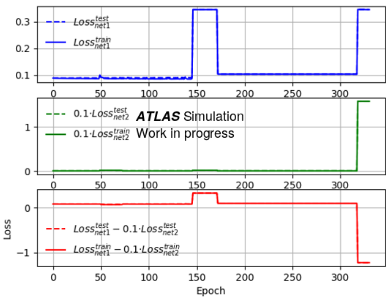
Optimisers

- Gradient based optimisation
  - for example SGD, stochastic gradient descent
  - updates based on negative gradient
  - Learning rate determines step-size
  - Step-size does not automatically decay
- Adaptive optimisation
  - for example Adam, adaptive momentum estimation
  - learning rate and other parameters get updated based on the past gradients

Results for switching from SGD to Adam



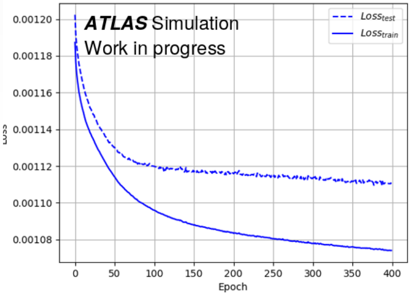
Loss curve using SGD



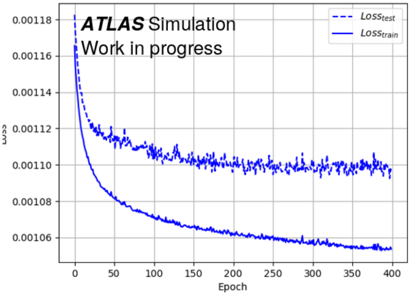
Loss curve using adam

- Default optimiser setup is definitely not always a good choice
- Also Adam sounds superior to SGD it is not easily interchangeable

Learning rate impact



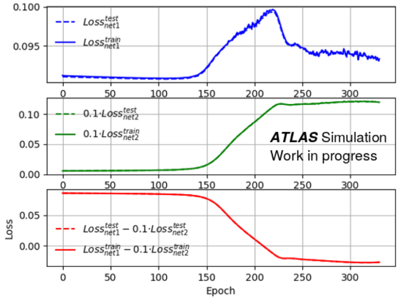
Using lr $= 0.02$



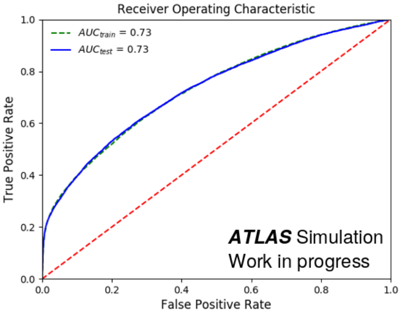
Using lr $= 0.1$

- High learning rate induces oscillations that might lead to missing a minimum
- A low learning rate makes the training converge very slowly

Losses and ROC curve for the adversarial training



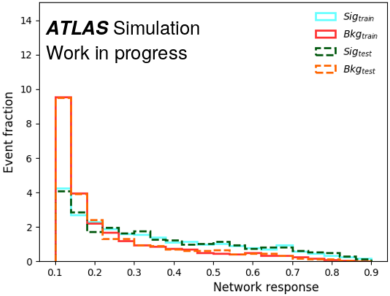
Loss curves, from top to bottom: Signal/background separation, nominal/systematics separation, combined losses



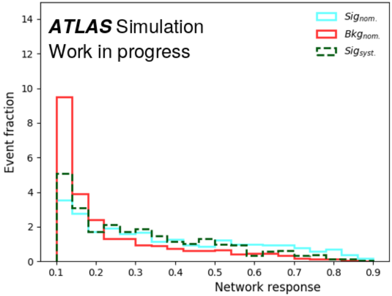
ROC curve

- Maximum for adversary is found
- Bad classififer training

Separation and Systematics



Separation



Nominal systematics agreement

- The separation is visible. Both networks train.
- Neither separation nor nom/sys agreement is very convincing

Conclusions

- Hyperparameters of optimisers and optimisers in general are not always a trivial problem
- The architecture and setup of the network have to be carefully tested and thought out for every problem
- Adversarial neural networks are a promising approach to limit the impact of problematic systematic uncertainties
- In this work the classifier cannot find a model that controls the systematics properly