Electron Identification Using Neural Networks

Supervisors:
Cyril Becot
Sarah Heim
Stefan Richter

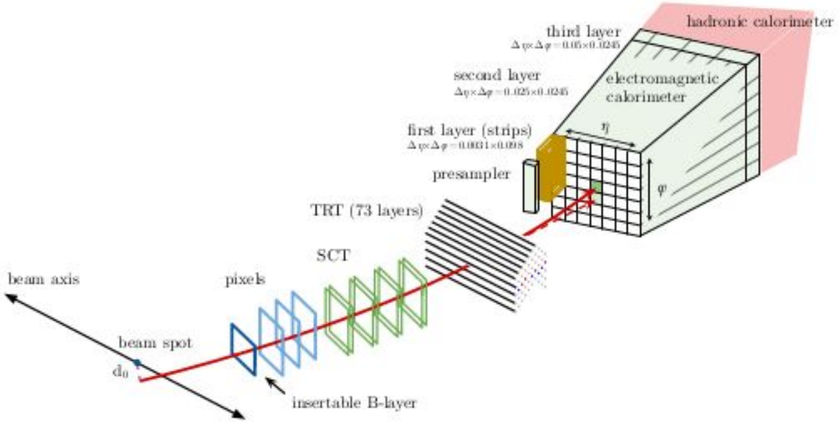
Paweł Drabczyk

Overview

- Electrons have very clear signature: it is relatively easy to measure them
- Now: likelihood-based method
- Now: neglecting correlations between ID variables
- Future: modern tools - Neural Networks
- GOAL OF PROJECT: improve ratio of signal to background using neural networks

Electron Reconstruction[1]

- Electron reconstruction means creating track-cluster pairs



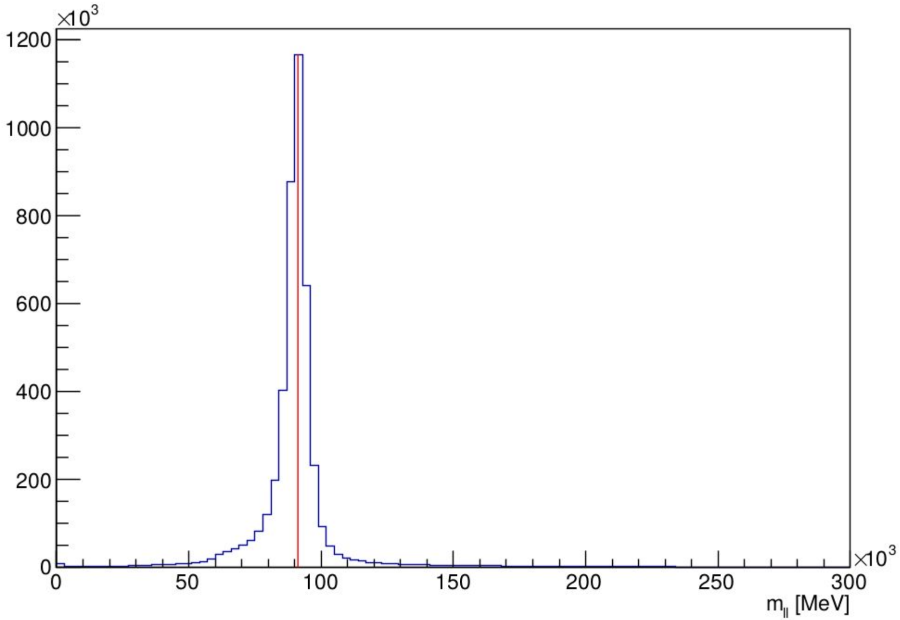
Electron Identification

- 19 variables: shower-shapes, track quality, track-calo matching
- They have different distributions for signal and background
- We extract LH for every variable separately
- Particle LH is computed as product of single-variable LHs
- Neglecting correlation between single-variables’ LHs
- Some variables are used as fixed cuts

Type	Description	Name	Rejects LF	Rejects γ	Rejects HF	Usage
Hadronic leakage	Ratio of E_T in the first layer of the hadronic calorimeter to E_T of the EM cluster (used over the range $\lvert\eta\rvert < 0.8$ or $\lvert\eta\rvert > 1.37$)	R_{had1}	x	x		LH
	Ratio of E_T in the hadronic calorimeter to E_T of the EM cluster (used over the range $0.8 < \lvert\eta\rvert < 1.37$)	R_{had}	x	x		LH
Third layer of EM calorimeter	Ratio of the energy in the third layer to the total energy in the EM calorimeter. This variable is only used for $E_T < 80$ GeV, due to inefficiencies at high E_T, and is also removed from the LH for $\lvert\eta\rvert > 2.37$, where it is poorly modelled by the simulation.	f_3	x			LH
Second layer of EM calorimeter	Lateral shower width, $\sqrt{(\Sigma E_i\eta_i^2)/(\Sigma E_i) - ((\Sigma E_i\eta_i)/(\Sigma E_i))^2}$, where E_i is the energy and η_i is the pseudorapidity of cell i and the sum is calculated within a window of 3×5 cells	$w_{\eta 2}$	x	x		LH
	Ratio of the energy in 3×3 cells over the energy in 3×7 cells centred at the electron cluster position	R_ϕ	x	x		LH
	Ratio of the energy in 3×7 cells over the energy in 7×7 cells centred at the electron cluster position	R_η	x	x	x	LH
First layer of EM calorimeter	Shower width, $\sqrt{(\Sigma E_i(i - i_{\mathrm{max}})^2)/(\Sigma E_i)}$, where i runs over all strips in a window of $\Delta\eta \times \Delta\phi \approx 0.0625 \times 0.2$, corresponding typically to 20 strips in η, and i_{max} is the index of the highest-energy strip, used for $E_T > 150$ GeV only	w_{stot}	x	x	x	C
	Ratio of the energy difference between the maximum energy deposit and the energy deposit in a secondary maximum in the cluster to the sum of these energies	E_{ratio}	x	x		LH
	Ratio of the energy in the first layer to the total energy in the EM calorimeter	f_1	x			LH
Track conditions	Number of hits in the innermost pixel layer	n_{Blayer}		x		C
	Number of hits in the pixel detector	n_{Pixel}		x		C
	Total number of hits in the pixel and SCT detectors	n_{Si}		x		C
	Transverse impact parameter relative to the beam-line	d_0		x	x	LH
	Significance of transverse impact parameter defined as the ratio of d_0 to its uncertainty	$\lvert d_0/\sigma(d_0)\rvert$		x	x	LH
	Momentum lost by the track between the perigee and the last measurement point divided by the momentum at perigee	$\Delta p/p$	x			LH
TRT	Likelihood probability based on transition radiation in the TRT	eProbabilityHT	x			LH
Track–cluster matching	$\Delta\eta$ between the cluster position in the first layer and the extrapolated track	$\Delta\eta_1$	x	x		LH
	$\Delta\phi$ between the cluster position in the second layer of the EM calorimeter and the momentum-rescaled track, extrapolated from the perigee, times the charge q	$\Delta\phi_{\mathrm{res}}$	x	x		LH
	Ratio of the cluster energy to the track momentum, used for $E_T > 150$ GeV only	E/p	x	x		C

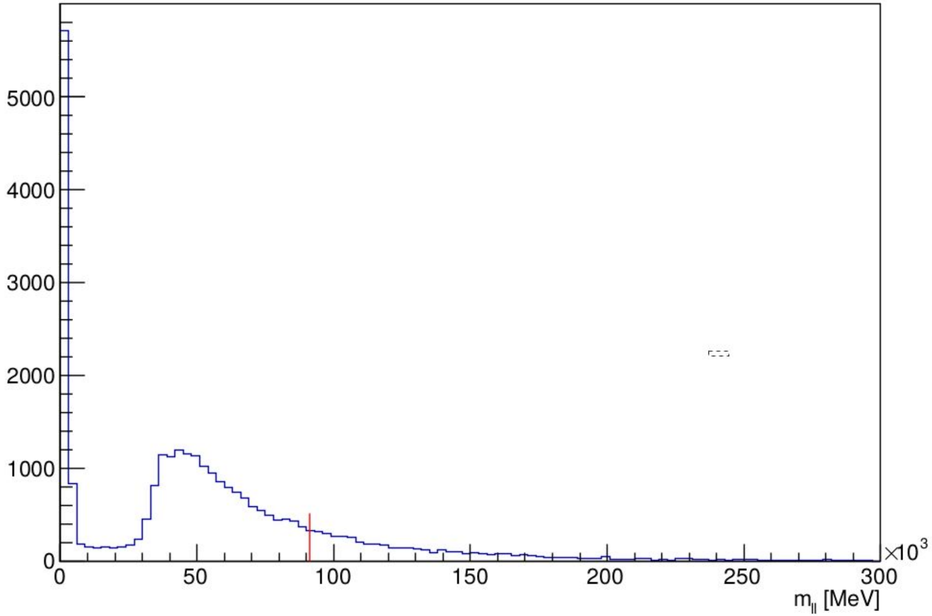
Data - Signal

- Signal sample used for training and testing NN comes from $Z \to e^+e^-$ decay

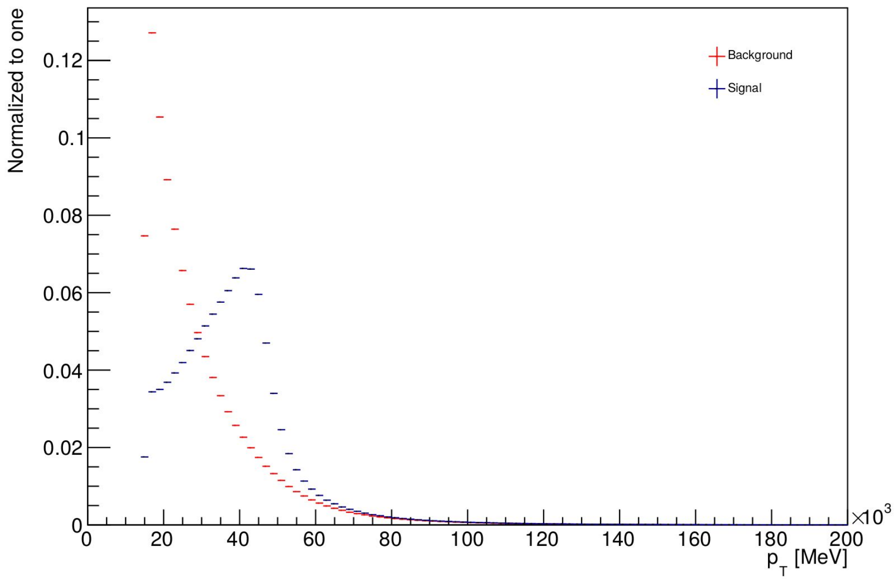


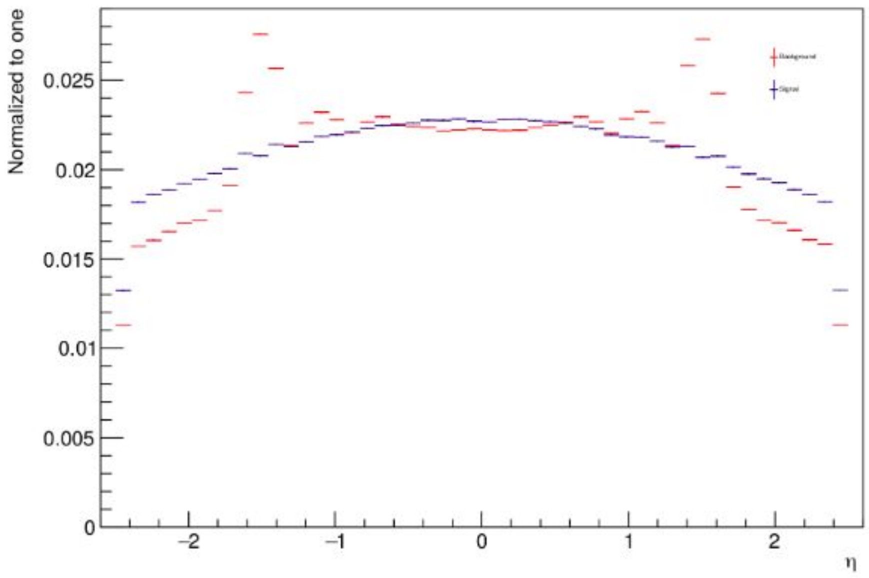
Data - background

- Background sample contains two hadronic jets with filter over jet momentum

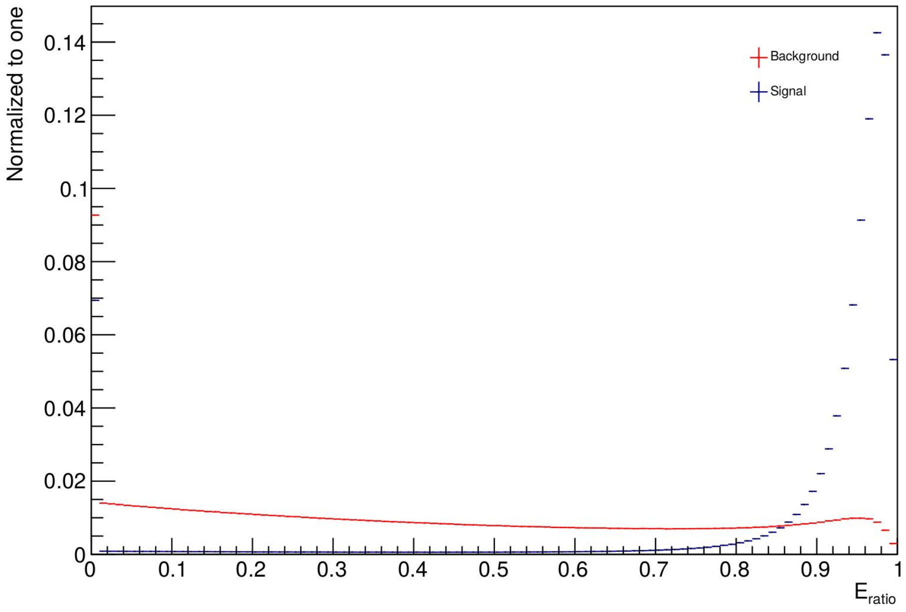


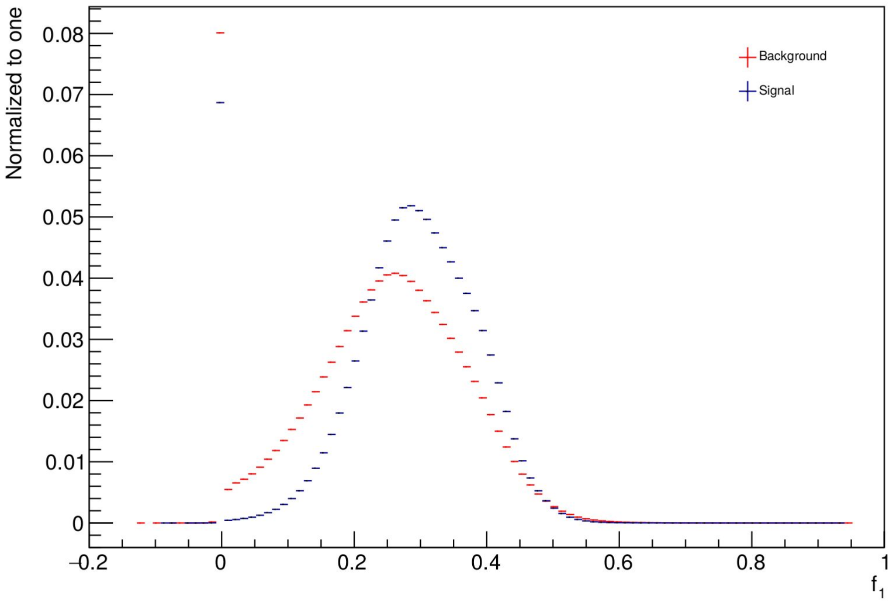
Kinematic Plots





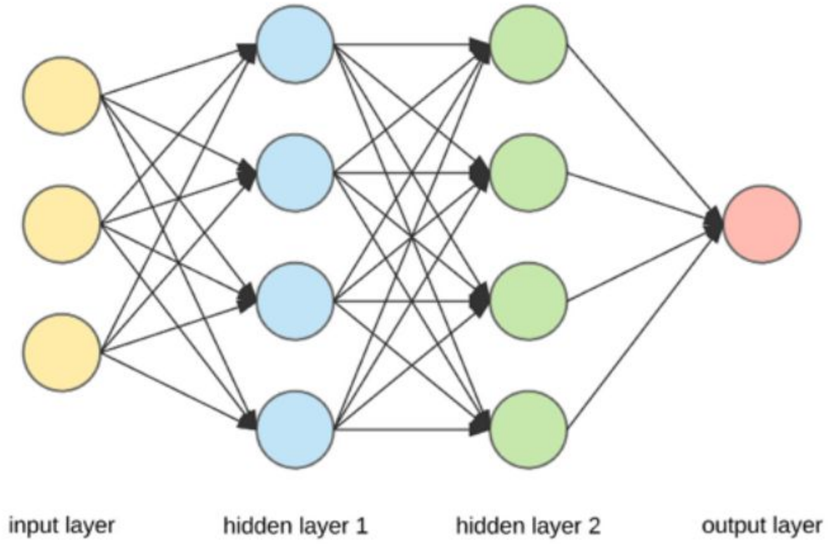
Discriminating Plots



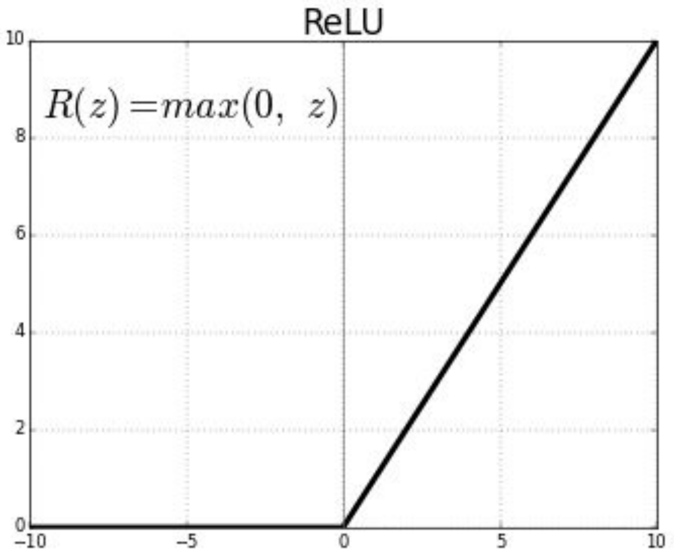


Model Architecture[2]

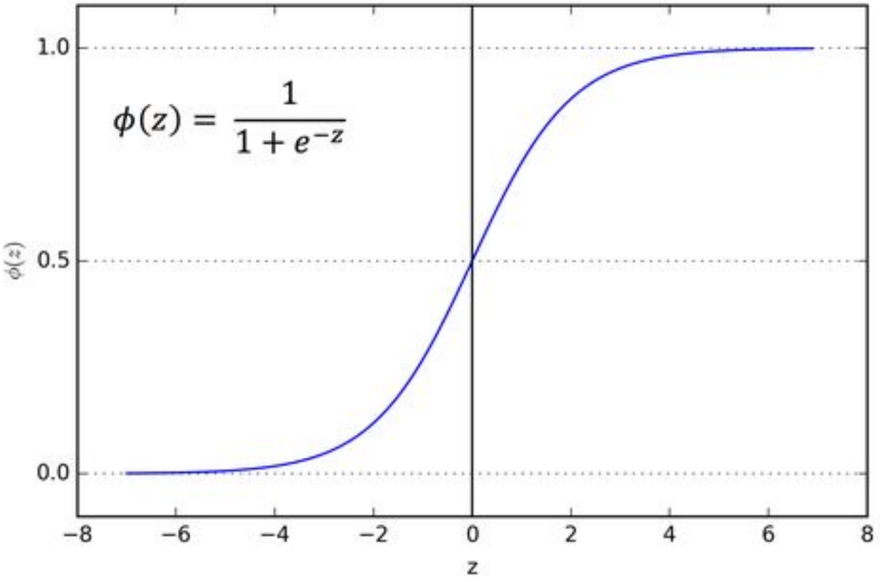
- Implemented using Keras.
- Same input variables as LH method
- Different number of layers (2-8)
- Number of neurons getting smaller in deep layers
- Up to 21000 of trainable parameters
- ReLU activation function (sigmoid for output layer)
- Loss function is binary cross entropy
- In ideal case output (number 0-1) is proportional to LH of being electron



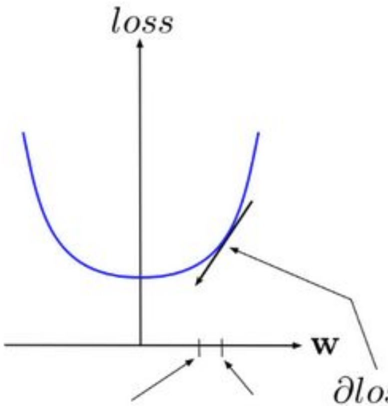
Activation Function



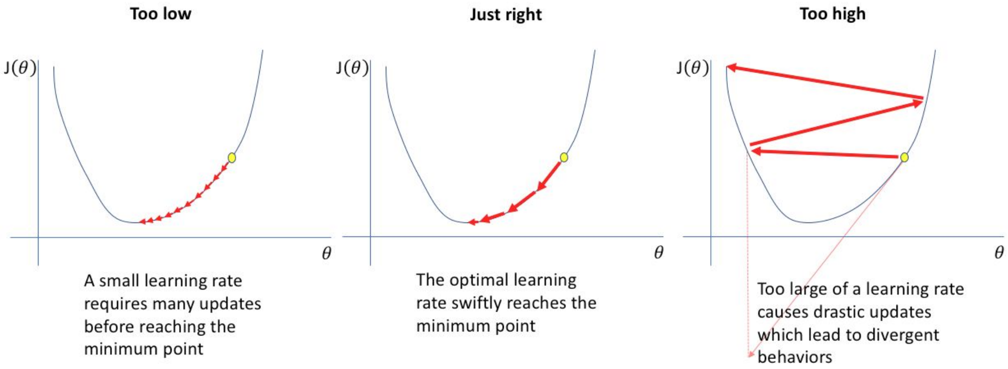
Sigmoid



Loss Function and Learning Rate[6]



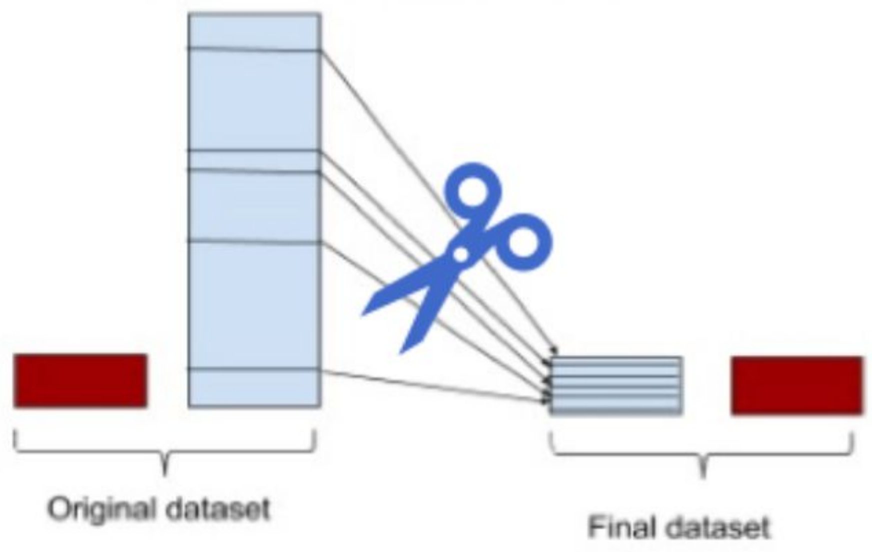
$$\mathbf{w_{i+1}} = \mathbf{w_i} - \left.\frac{\partial loss}{\partial \mathbf{w}}\right|_{\mathbf{w_i}} \times learning\ rate$$



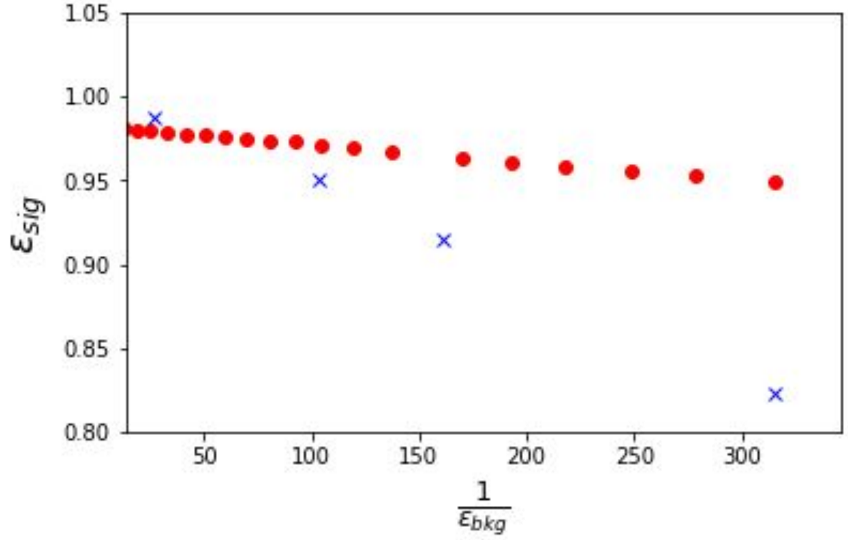
$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

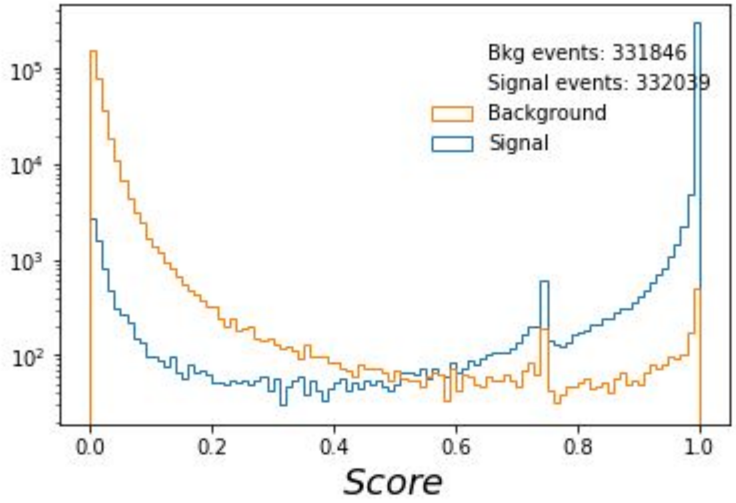
Balancing Data - Undersampling

- Neural Networks perform the best with the same amount of signal and background in training sample
- One of the ways to reach the balance is to use undersampling

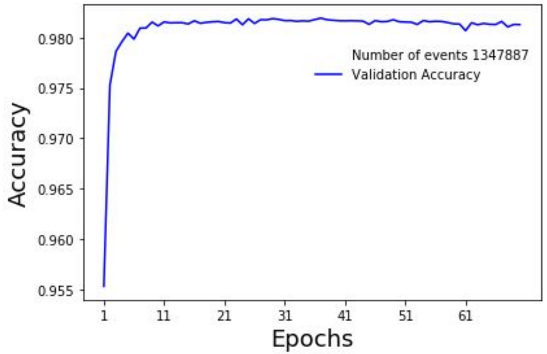


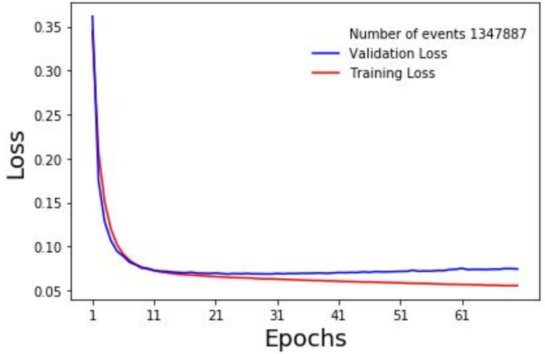
NN performance (p_T: 30-40 GeV)

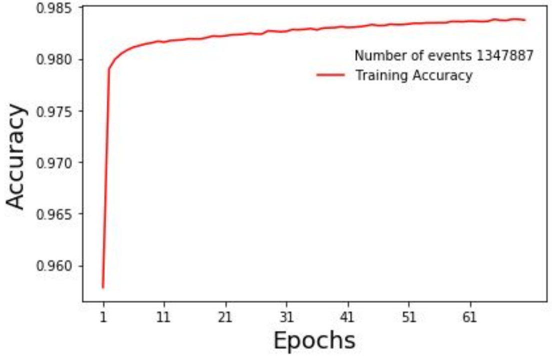


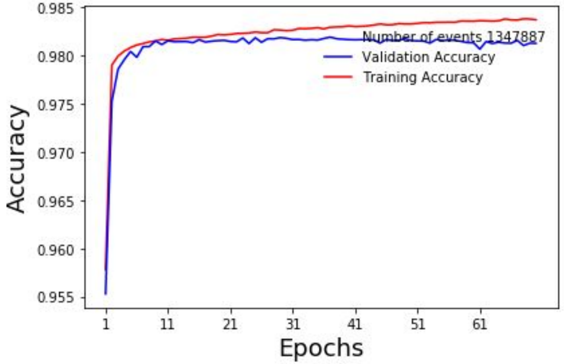


NN performance (p_T: 30-40 GeV)

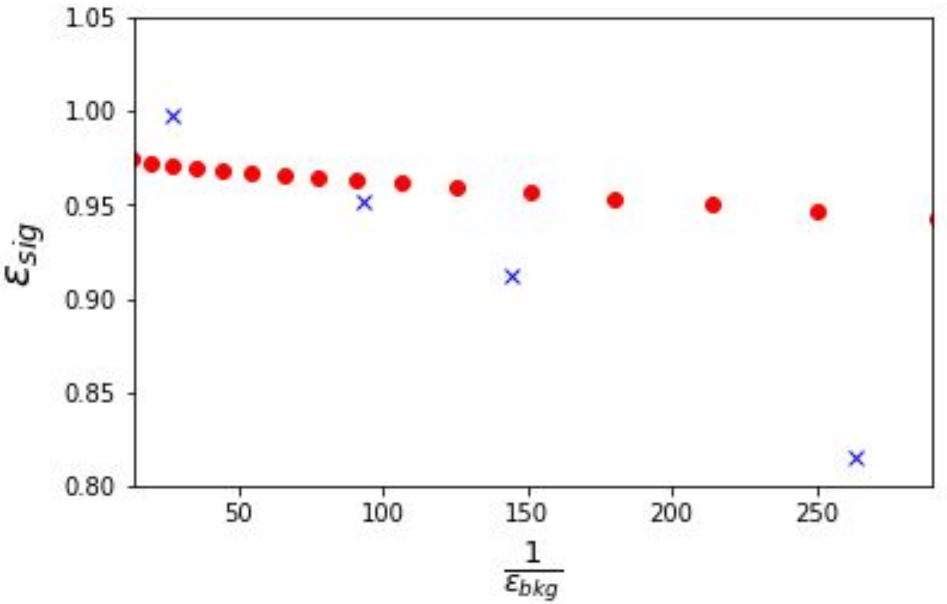


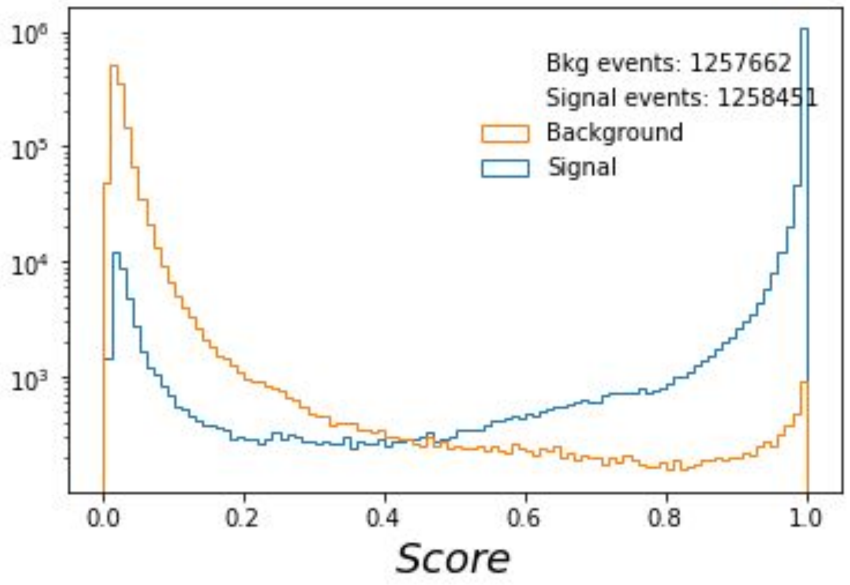




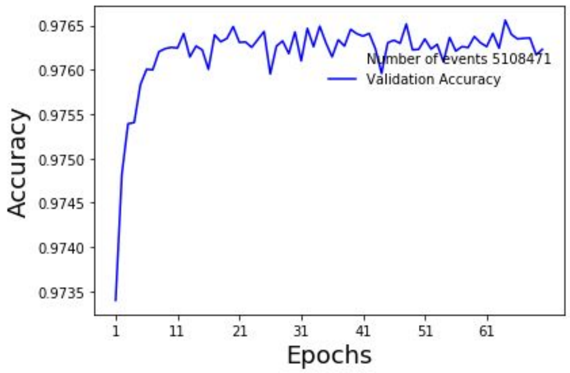


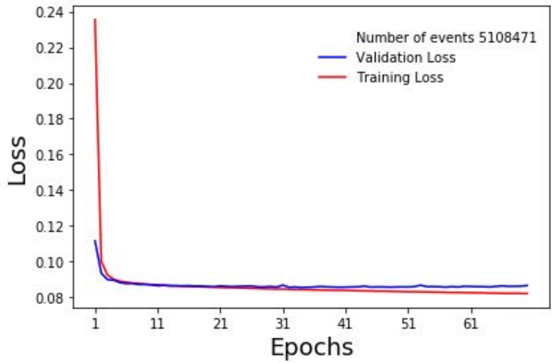
NN performance (p_T: 20-60 GeV)

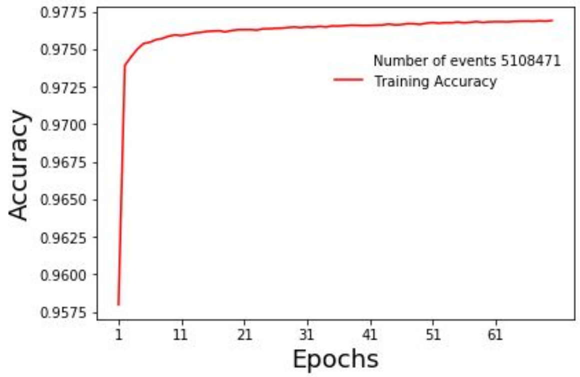


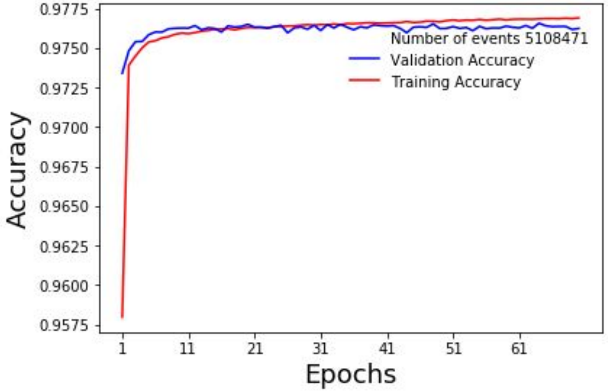


NN performance (p_T: 20-60 GeV)



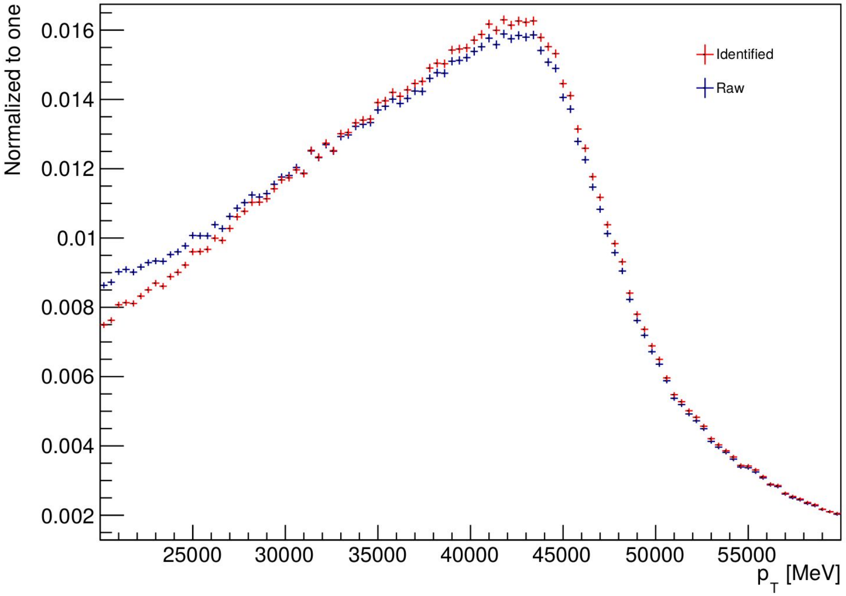






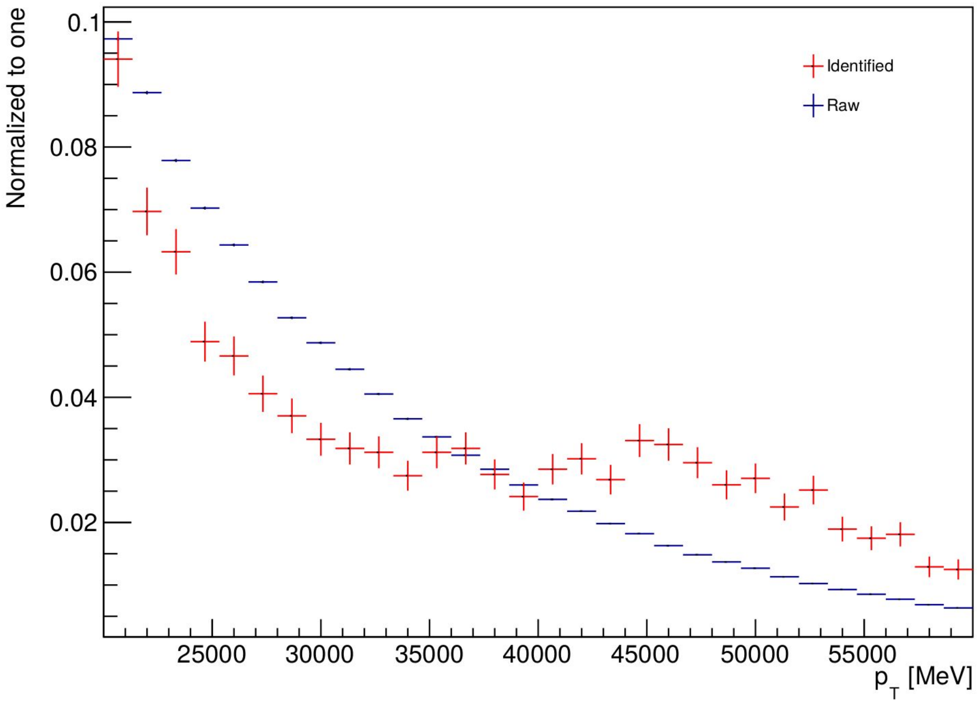
Signal performance - p_T dependence

- The plot shows number of events before and after the identification
- Using 0.85 score treshort (medium)
- p_T : 20-60 GeV



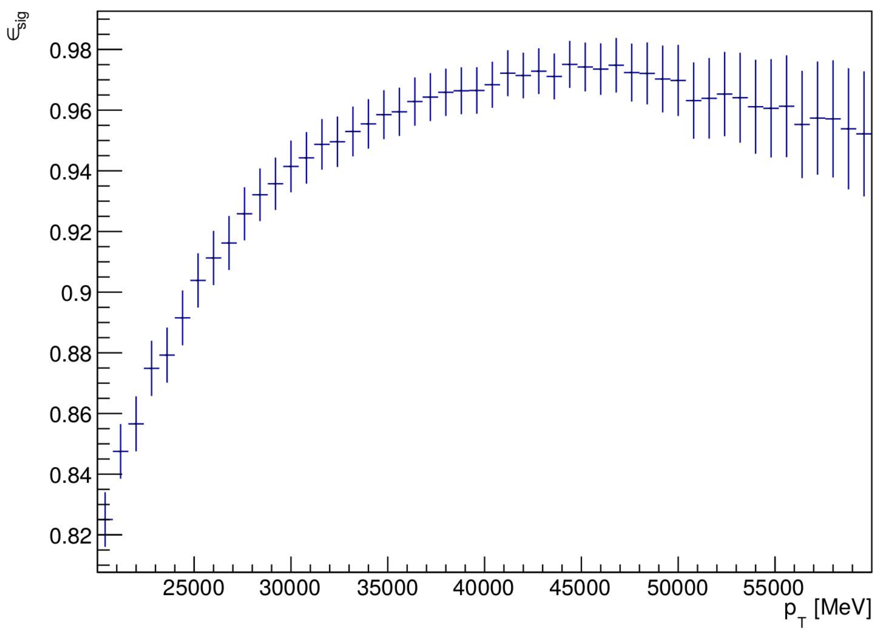
Background performance - p_T dependence

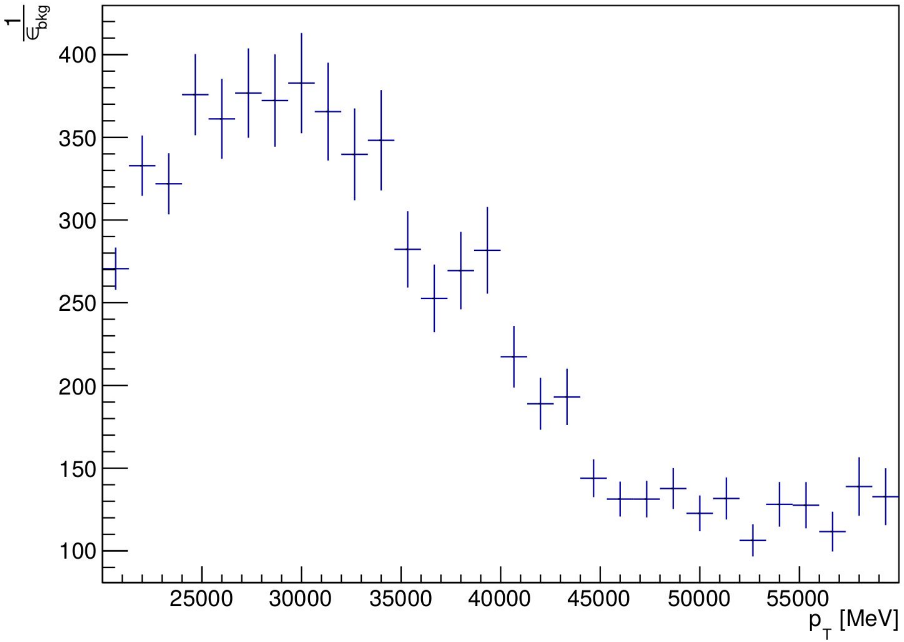
- p_T : 20-60 GeV



Signal Efficiency and Background Rejection

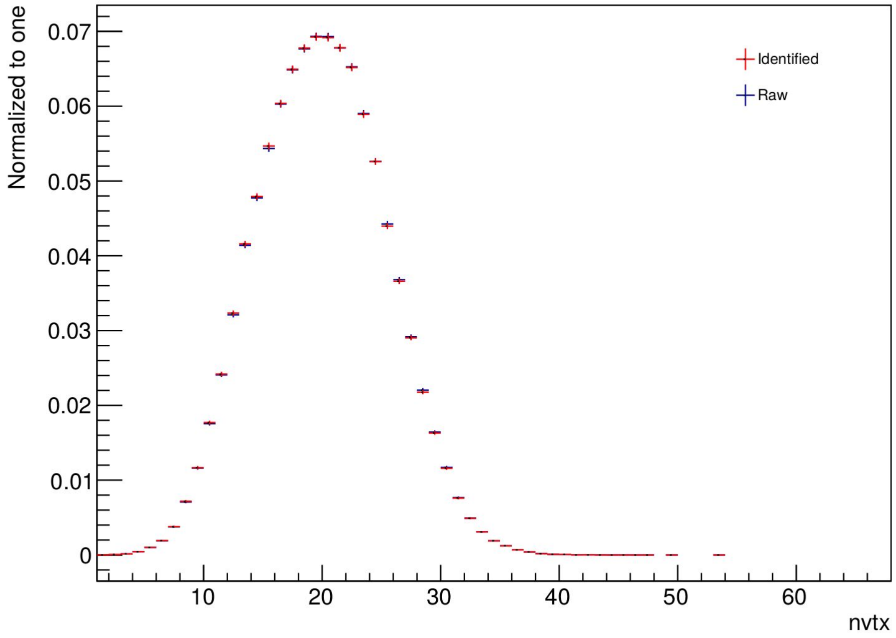
- p_T : 20-60 GeV



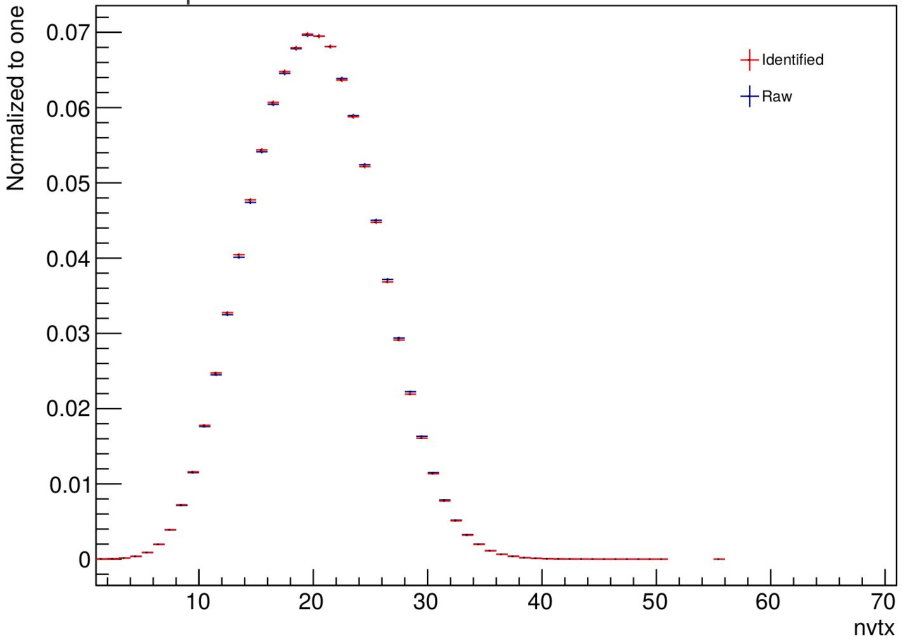


Signal performance - nvtx dependence

- p_T : 30-40 GeV

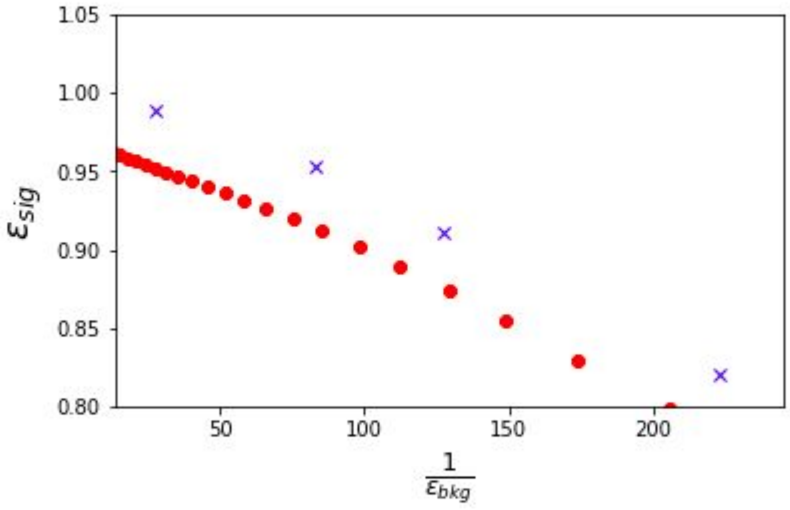


- p_T : 20-60 GeV

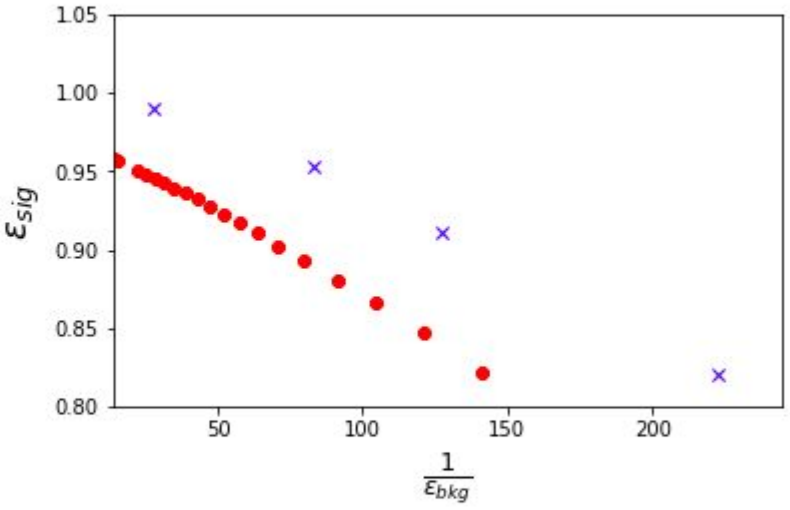


Small Data Sample - Question Mark

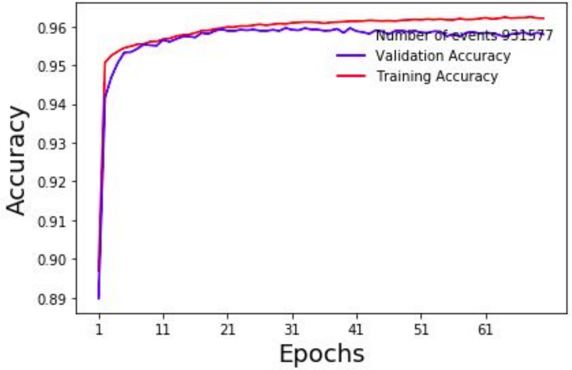
Full η (-0.7,0.7)

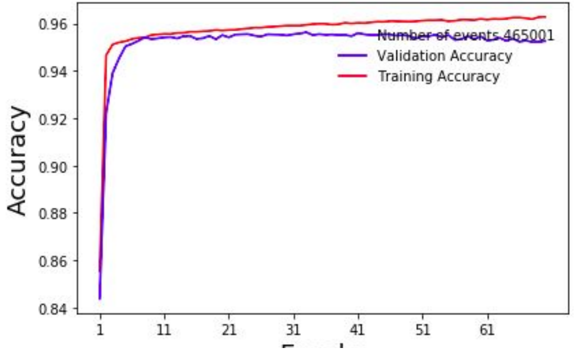


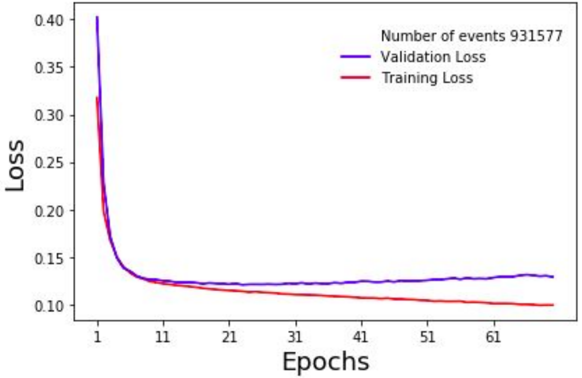
Half η (-0.7,0.)

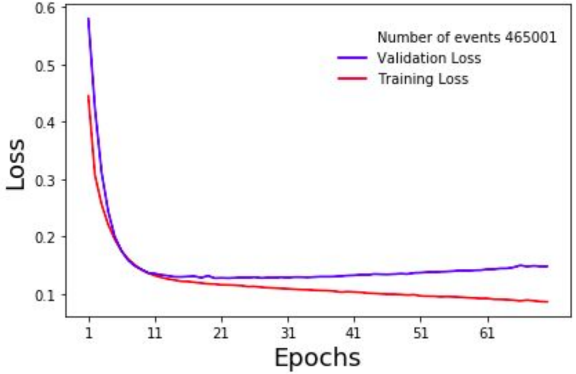


Small Data Sample - Question Mark









Summary and Future Stuff

- NN are powerful tools with huge potential in electron identification. They might give better results than current LH method, however their performance have to be evaluated carefully.
- Large amount of events needed in training sample
- p_T dependence of model performance
- They seem to be pile-up independent
- The idea is to use NN with the same architecture, but trained on different p_T and η bins
- Test on real data (similar to previous LH method's test)
- Additional input variables (e.g. p_T,raw informations about cells in cluster)

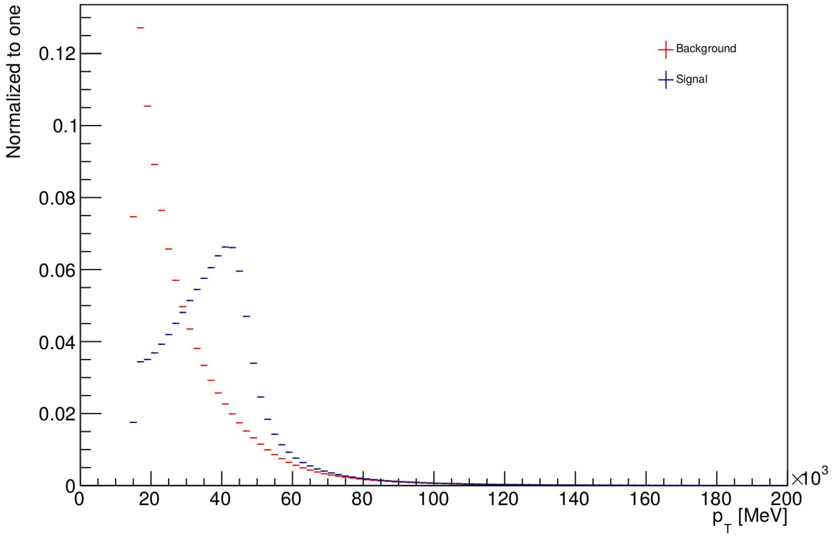
References

- [1] ATLAS Collaboration, Electron reconstruction and identification in the ATLAS experiment using the 2015 and 2016 LHC proton–proton collision data at $\sqrt{s}$ = 13 TeV
  arXiv:1902.04655 [physics.ins-det]
- [2] https://indico.desy.de/indico/event/21278/contribution/13/material/slides/0.pdf
- [3] https://miro.medium.com/max/335/1*YH_vPYQEDIW0JoUYMeLz_A.png
- [4] Image: https://deepai.org/machine-learning-glossary-and-terms/sigmoidal-nonlinearity
- [5] Image: https://miro.medium.com/max/714/1*oePAhrm74RNnNEolprmTaQ.png
- [6] Image: https://www.jeremyjordan.me/nn-learning-rate/

Backup slides

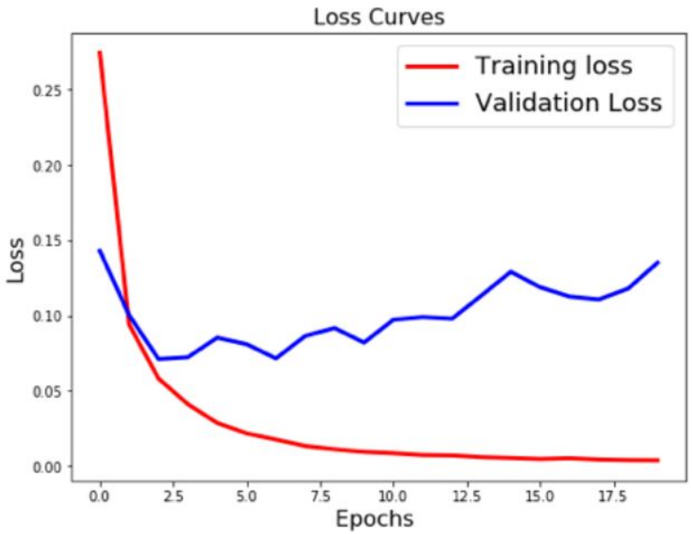
“p_T weights”

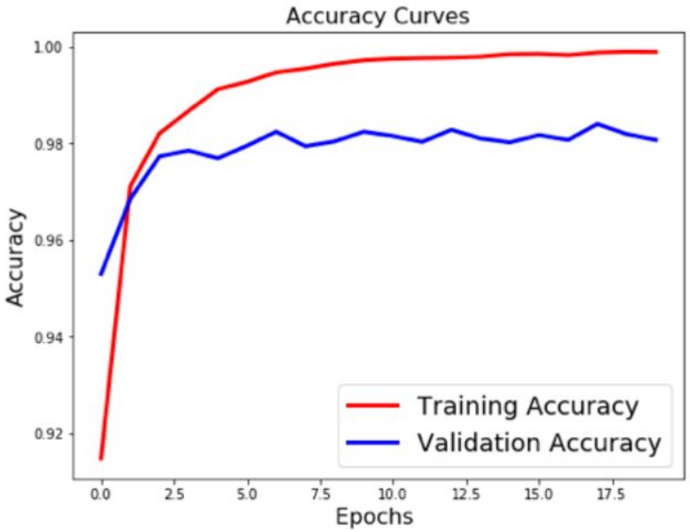
- Training sample should be balanced, also inside of every very fine p_T bin
- we apply additional weight called “p_T weights”(except of event weights)
- $$f_{minibin} = \frac{\sum_i \omega_i^{sig}}{\omega_i^{bkg}}$$



Watch out for overtraining...[2]

- After too many epochs NN on the same data tend to get overtrained (especially small training samples are sensitive).
- It means that NN starts to “remember” properties of training sample and losses its generality

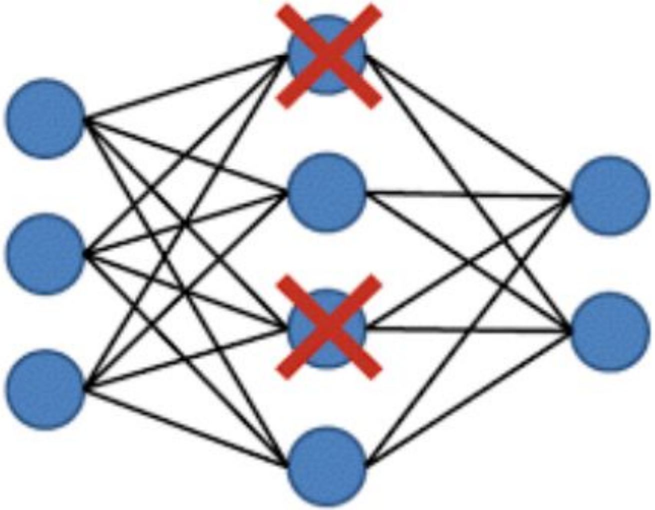




[2] https://indico.desy.de/indico/event/21278/contribution/13/material/slides/0.pdf

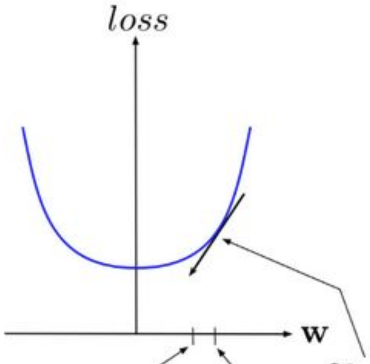
Study of Dropout

- Dropout is a way to avoid overtraining
- Fraction of neurons is randomly turned off during the training, reducing the dependency on the training set
- I tried different amounts of dropout, but at the end only symbolic amount (5%) was needed or even 0%



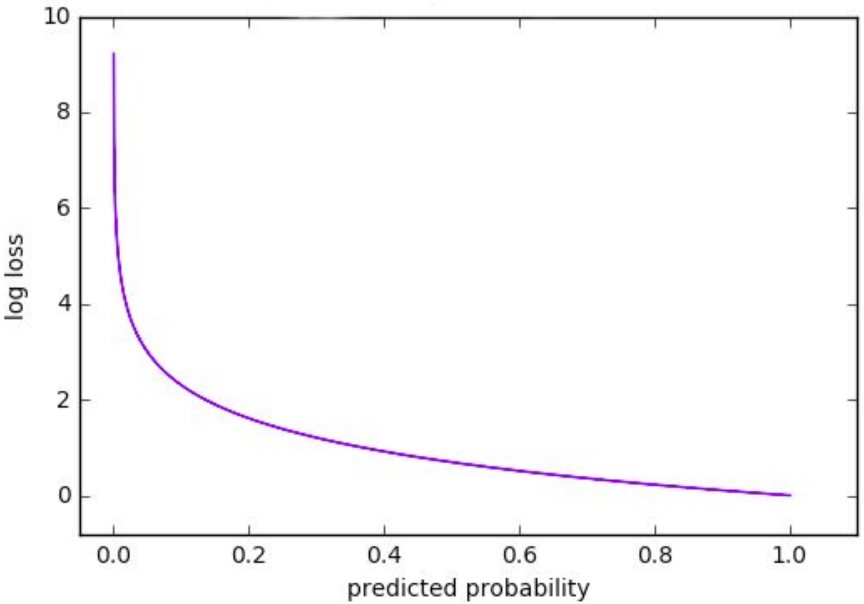
[2] https://indico.desy.de/indico/event/21278/contribution/13/material/slides/0.pdf

Loss Function - Binary Cross Entropy



$$\mathbf{w_{i+1}} = \mathbf{w_i} - \left.\frac{\partial loss}{\partial \mathbf{w}}\right|_{\mathbf{w_i}} \times learning\ rate$$

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$



Learning rate

- Default learning rate was 0.001
- With Keras function ReduceLROnPlateau we can avoid overtraining
- Luckily, in our case it was not needed