Methodology for the use of neural networks in the data analysis of the collider experiments

- Applications of NNs in collider experiments
- Classification task with NN
- The general recipes to move application of neural networks from state of the art stage to deterministic procedure

Lev Dudko

SINP MSU, Moscow

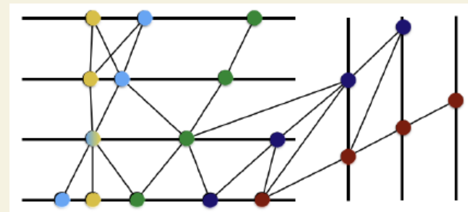
supported by RSF-22-12-00152

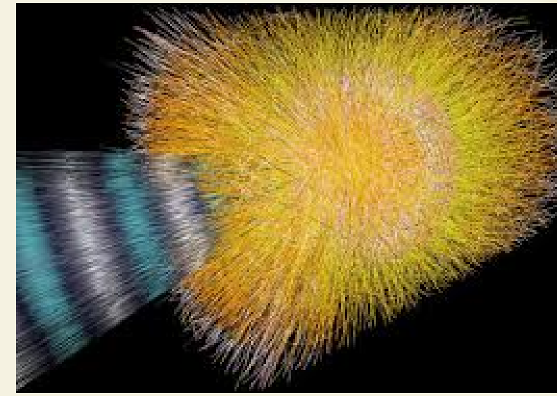
Modern applications of DNN in collider experiments

- **Triggers**, online event selection, hardware and software DNN implementations
- **Reconstruction and identification of the objects**, software implementations of DNN.
- **Classification of events**, distinguishing of some signal process from background processes. Problem of event negative weights.
- **Anomaly detection**, search for some deviations in data, unsupervised training, autoencoders. Low efficiency.
- **Fast simulation**, using GAN to simulate more events, or detector response. Usually does not decrease statistical uncertainty
- Parton density functions **NNPDF** – most used PDF now
- **Unfolding**, back from detector level to parton level
- **Regression tasks** to estimate some model parameter(s)
- **Symbolic regression**, to estimate an analytic function from data
- **Self-driving lab**oratory

Reconstruction and identification of the objects

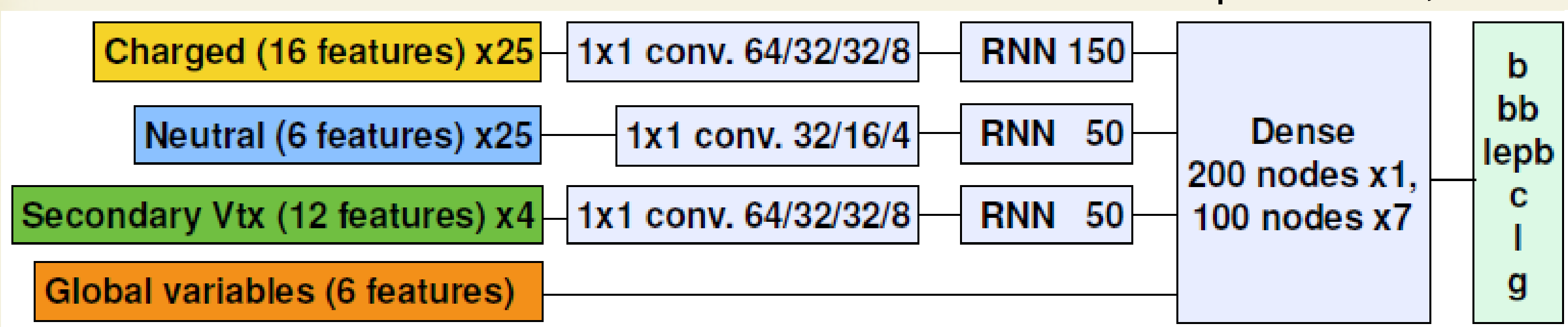
- Track reconstruction (GraphNN, ...)

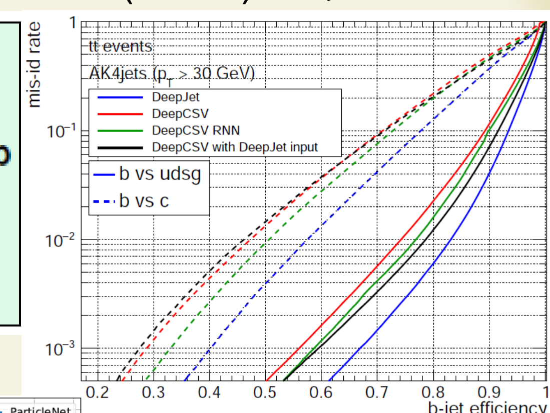




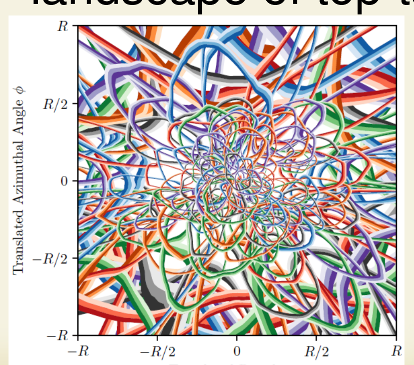
- Identification of the objects (jet b-tagging, top-tagging, …)

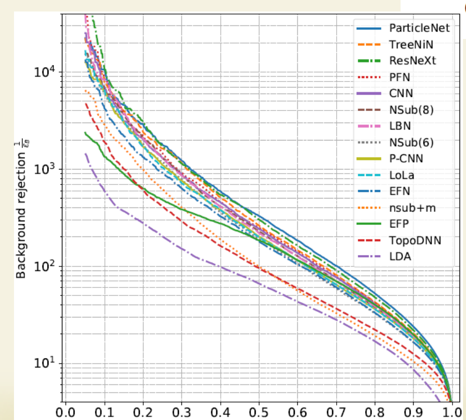
DeepJet CMS, JINST 15 (2020) 12, P12012



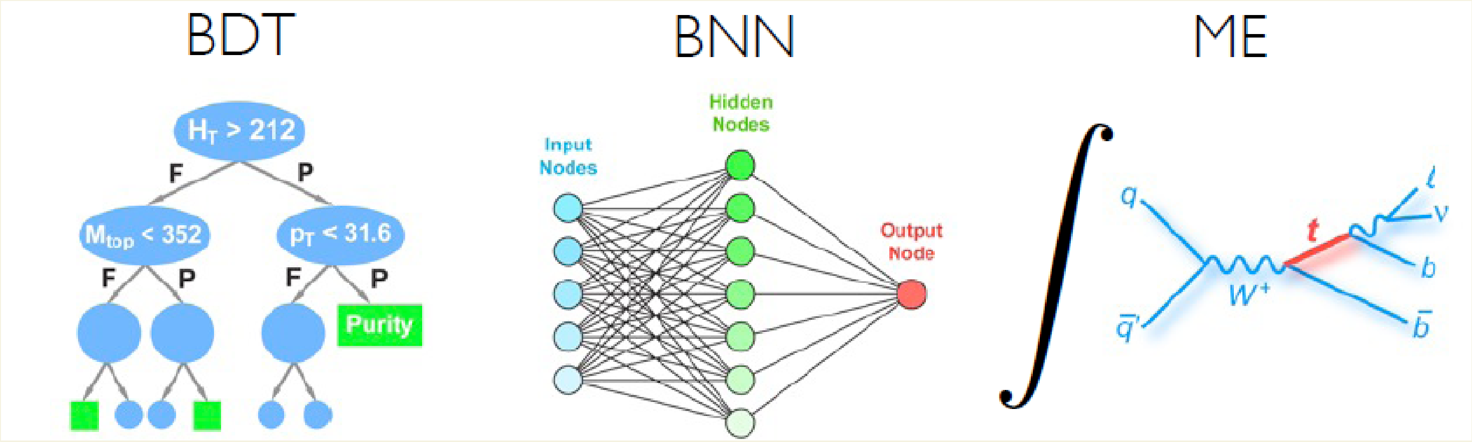


SciPost Phys. 7 (2019) 014 – landscape of top-taggers



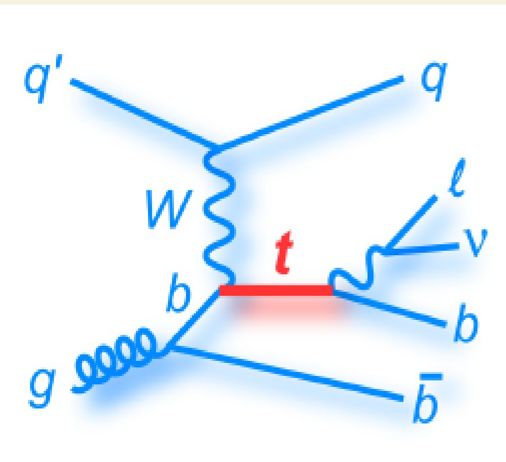


Classification of the events in collider experiments. Choice of multivariate technique.



Classification of events

Signal process signature



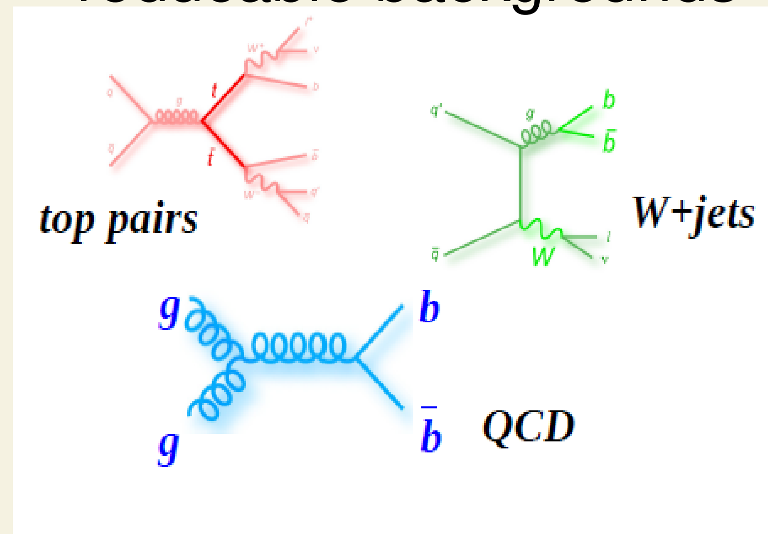
Light flavor jet

Lepton
Missing Et

High Pt b-jet

Low Pt b-jet

Irreducable and reducable backgrounds



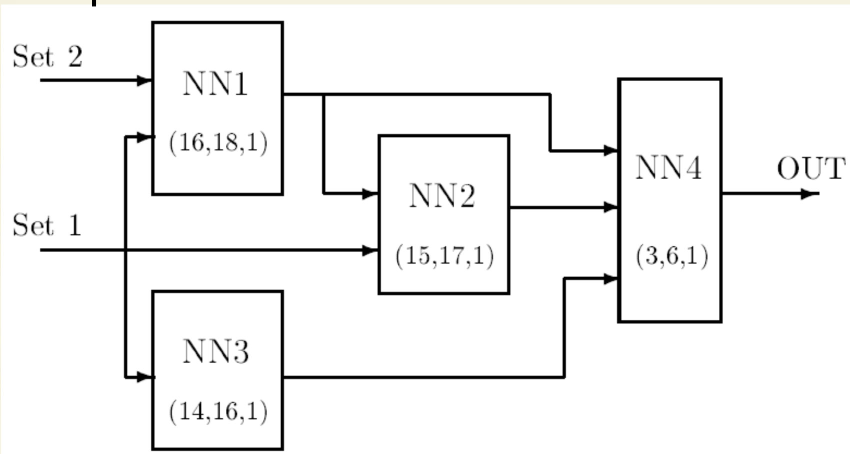
Total and differential cross sections $d\sigma \sim M^2(p_i \cdot p_f, s, t, u)$ are the functions of scalar products of four-momenta and/or Mandelstam variables.
Example of squared matrix element for u,d→t,b process:

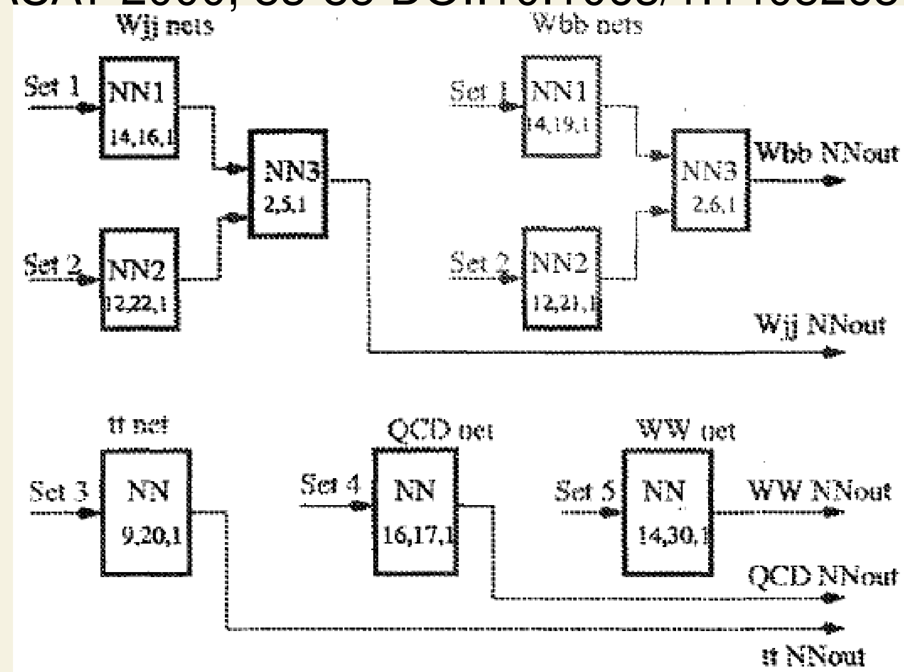
$$|M|^2 = V_{tb}^2 V_{ud}^2 (g_W)^4 \frac{(p_u p_b)(p_d p_t)}{(\hat{s} - m_W^2)^2 + \Gamma_W^2 m_W^2}, \quad |M|^2 = V_{tb}^2 V_{ud}^2 (g_W)^4 \frac{\hat{t}(\hat{t} - M_t^2)}{(\hat{s} - m_W^2)^2 + \Gamma_W^2 m_W^2}.$$

Optimization of single top-quark search with NN in D0 experiment.

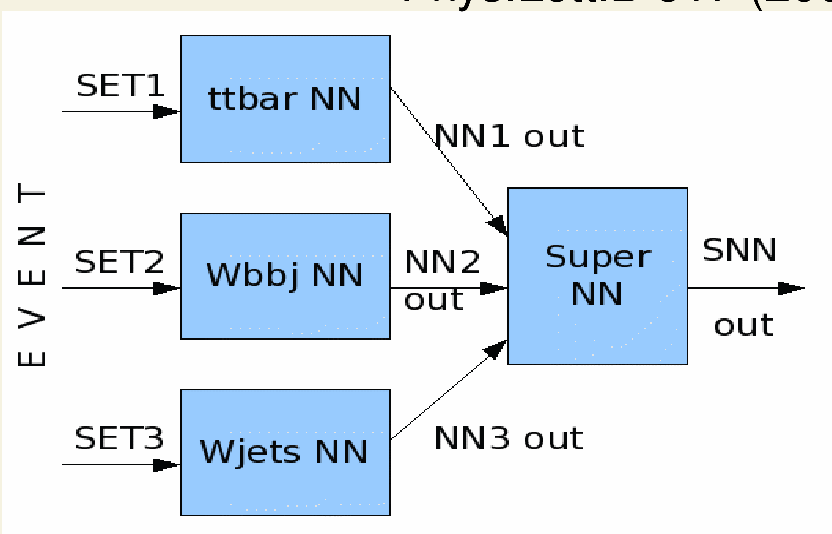
ACAT 2000, 83-85 DOI:10.1063/1.1405268

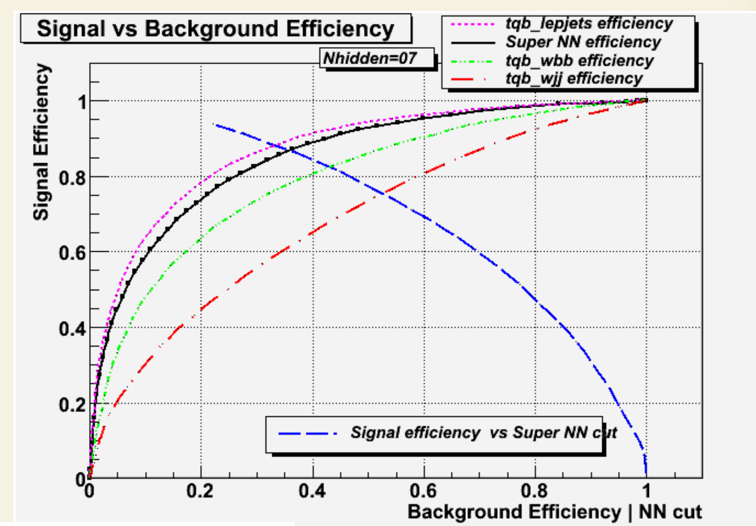
hep-ex/9907041





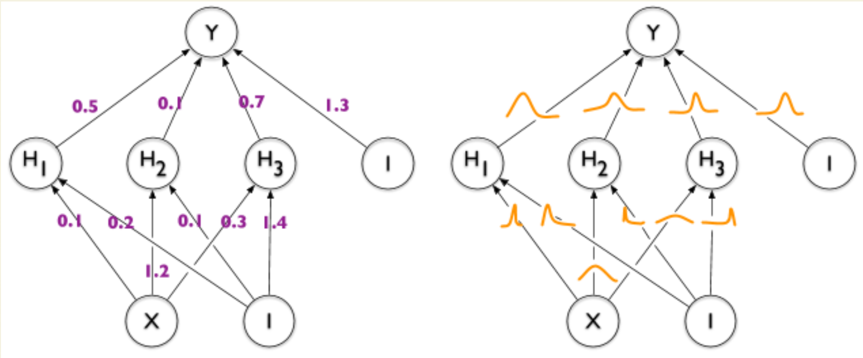
Phys.Lett.B 517 (2001) 282-294

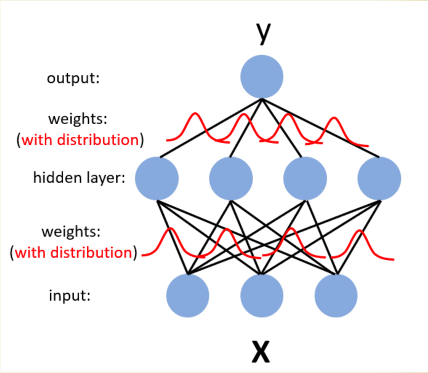




- $\sigma(p\bar{p} \to tb + X) < 17$ pb (классический анализ $\sigma^{tb}_{classic} < 39$ pb)
- $\sigma(p\bar{p} \to tqb + X) < 22$ pb (классический анализ $\sigma^{tqb}_{classic} < 58$ pb)

Bayesian Neural Networks (BNN)



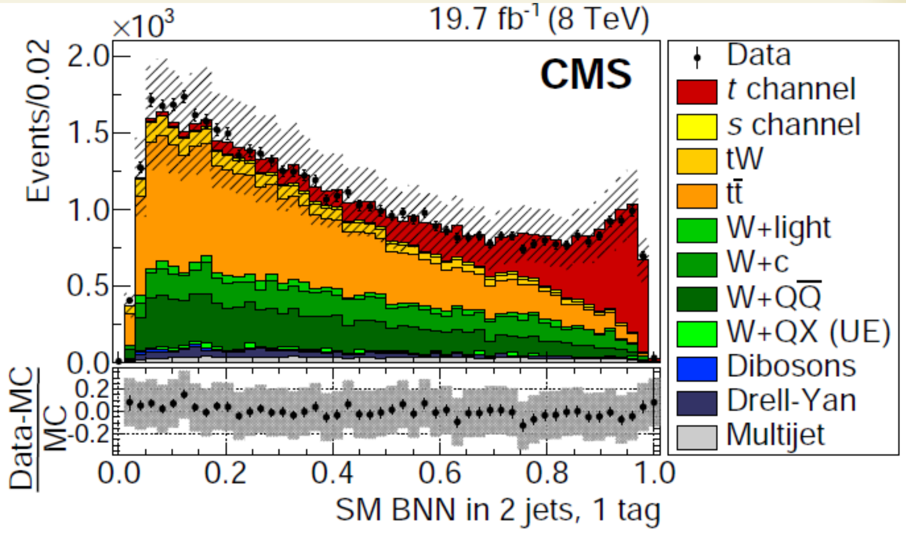


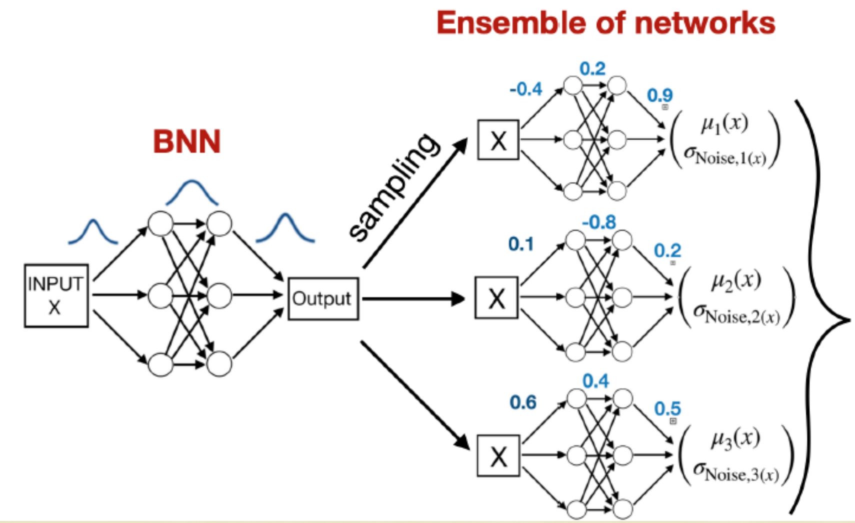
P. C. Bhat and H. B. Prosper, "Bayesian Neural Networks" PHYSTAT 2005;
R. M. Neal, Bayesian Learning of Neural Networks (1996); FBM package;

All of D0 analyses after 2005 use BNN not NN
e.g. D0, Observation of Single Top Quark Production
Phys.Rev.Lett. 103 (2009) 092001

Partial realisation in deep NN: tensorflow_probability, variational dropout, … , with fixed form of distribution.

BNN in CMS (LHC) JHEP 02 (2017) 028





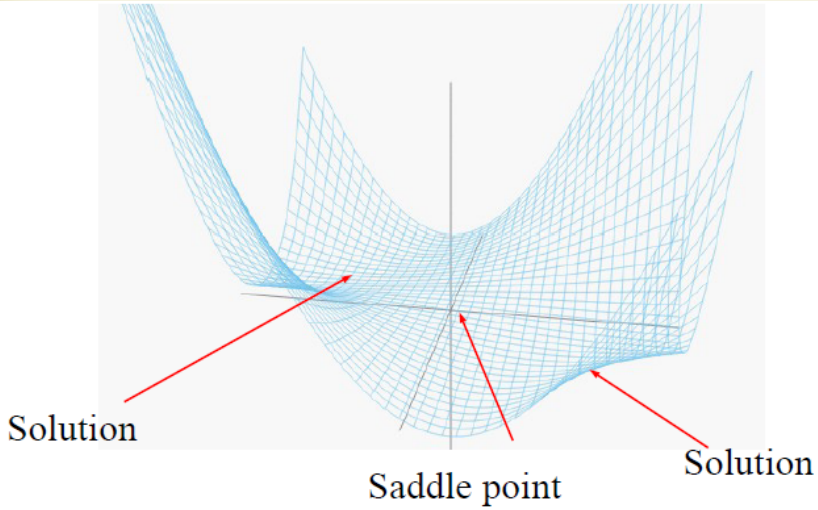
$$\mu = \frac{1}{N}\sum \mu_i$$

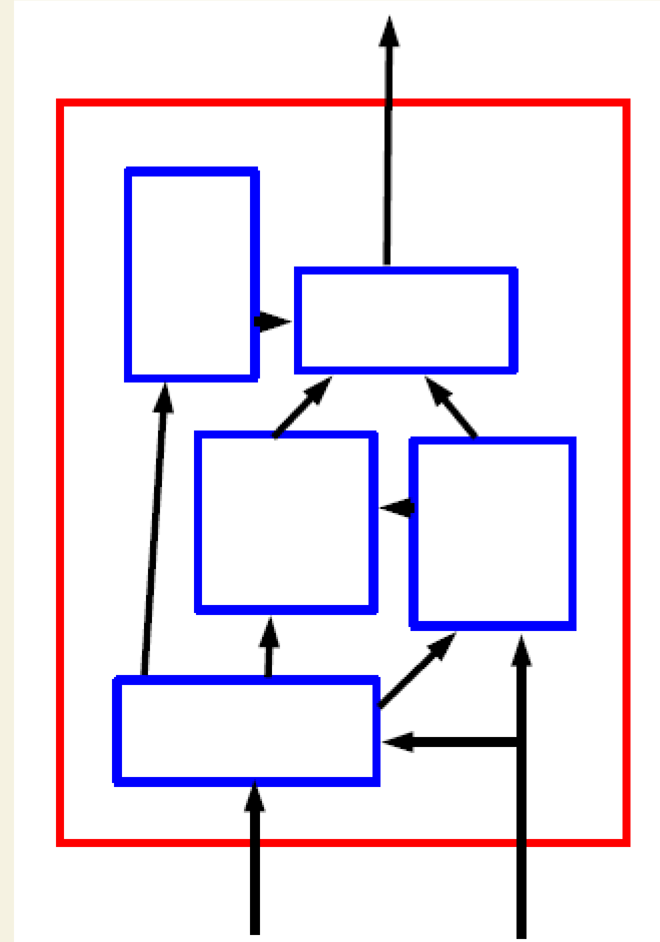
$$\sigma^2_{\text{Noise}} = \frac{1}{N}\sum \sigma^2_{\text{Noise},i}$$

$$\sigma^2_{\text{pred}} = \frac{1}{N}\sum (\mu - \mu_i)^2$$

Recent BNN talk T.Plehn (06/22)

Deep Learning Neural Networks, DNN





Hinton, G. E., Osindero, S., & Teh, Y. W. (2006).
A fast learning algorithm for deep belief nets.
Neural computation, 18(7), 1527-1554.

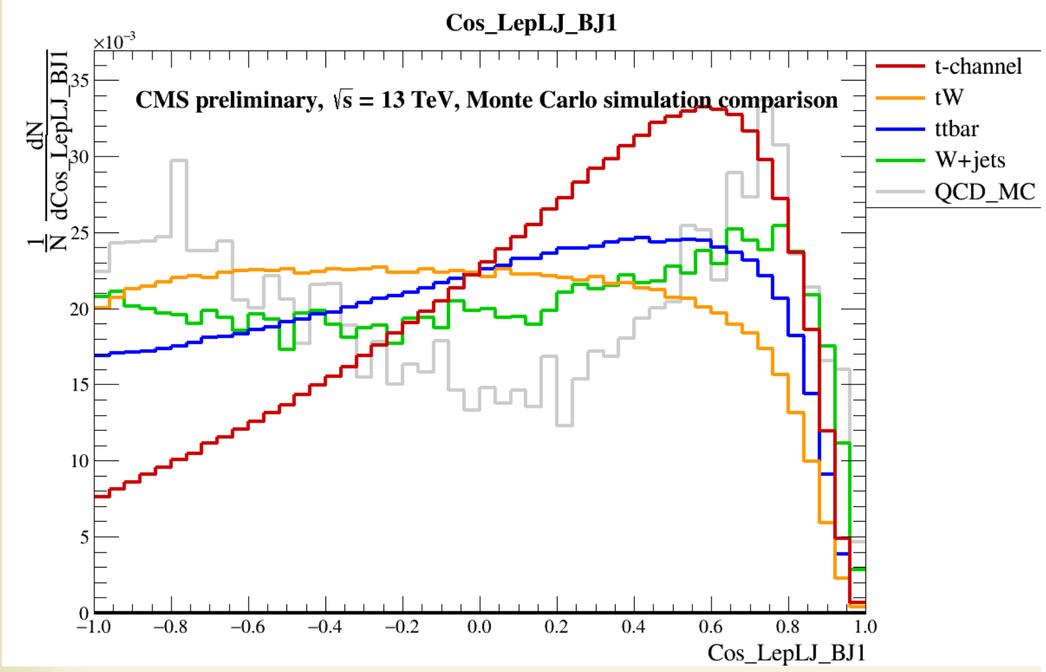
The main advantage of DNN is the ability to analyze raw, not preprocessed data.

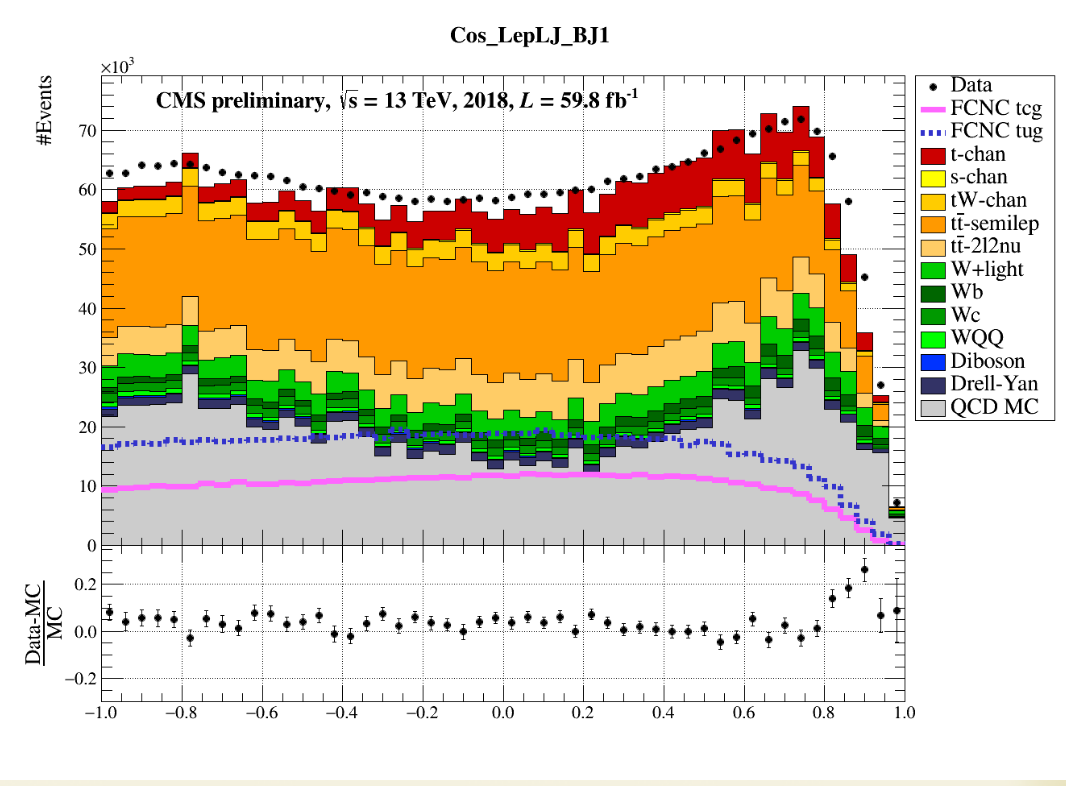
Probably, the first application in HEP: Nature Commun. 5 (2014) 4308

$$gg \to H^0 \to W^{\mp}H^{\pm} \to W^{\mp}W^{\pm}h^0$$

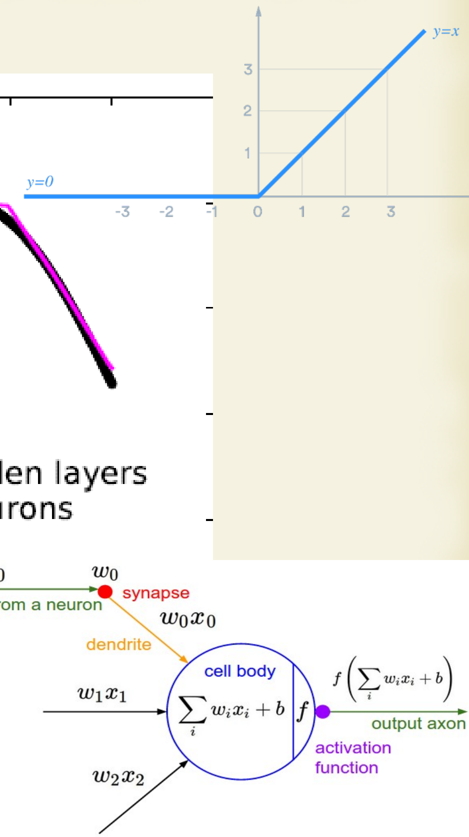
	Discovery significance		
Technique	Low-level	High-level	Complete
NN	2.5σ	3.1σ	3.7σ
DN	4.9σ	3.6σ	5.0σ

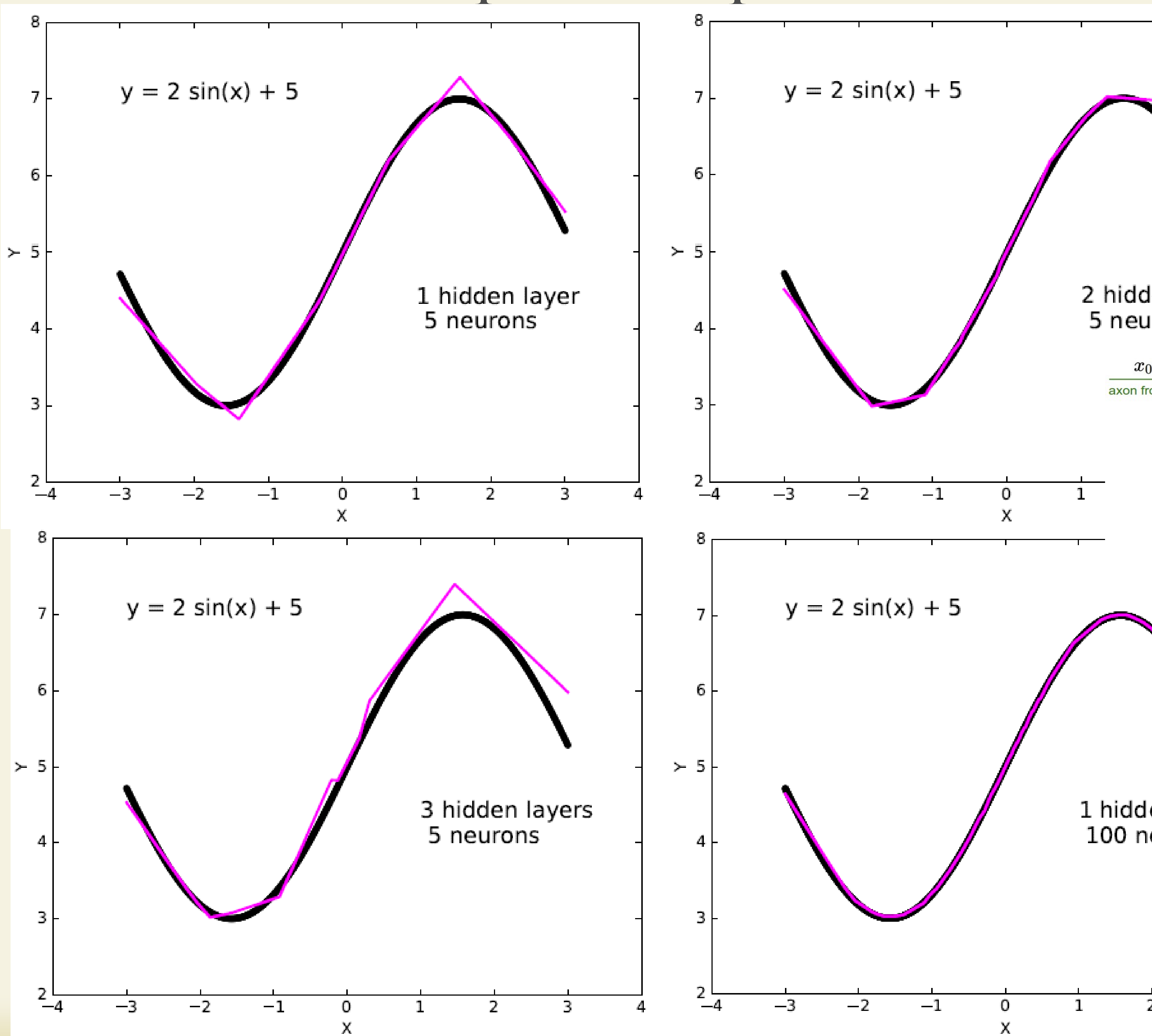
Selection of experimental observables and formaition of DNN input set of variables.





Simple example of NN





Method of high level "optimal observables"

- **Provides general recipe how to choose most sensitive high-level variables to separate signal and background**
  - It is based on the analysis of Feynman diagrams (FD) contributing to signal and background processes
  - Distinguish **three classes** of sensitive variables for the signal and each of kinematically different backgrounds: **Singular** variables (denominators of FD), **Angular** variables (numerators of FD) and **Threshold** variables (Energy thresholds of the processes)
  - Set of variables can be extended with other type of information, like detector relative variables (jet width, b-tagging discriminant)

Described in different examples for the top and Higgs searches

  - E.Boos, L.Dudko, T.Ohl Eur.Phys.J. C11 (1999) 473-484
  - E.Boos, L.Dudko Nucl.Instrum.Meth. A502 (2003) 486-488
  - E.Boos, V.Bunichev, L.Dudko, A.Markina, M.Perfilov Phys.Atom.Nucl. 71 (2008) 388-393

- **Applied in different experimental analysis in D0 and CMS**
  - Phys.Lett. B517 (2001) 282-294 and other D0 publications
  - JHEP02(2017)028 , ...

General method of low level "optimal observables"

The main advantage of Deep NNs (many layers, neurons) is the possibility to analyze raw, not preprocessed, information.
2 → n particles hard process has (3n-4) independent variables
What are the general low level observables?
[Int.J.Mod.Phys.A 35 (2020) 21, 2050119]

$$|M|^2 = V_{tb}^2 V_{ud}^2 (g_W)^4 \frac{(p_u p_b)(p_d p_t)}{\left(\hat{s} - m_W^2\right)^2 + \Gamma_W^2 m_W^2},$$

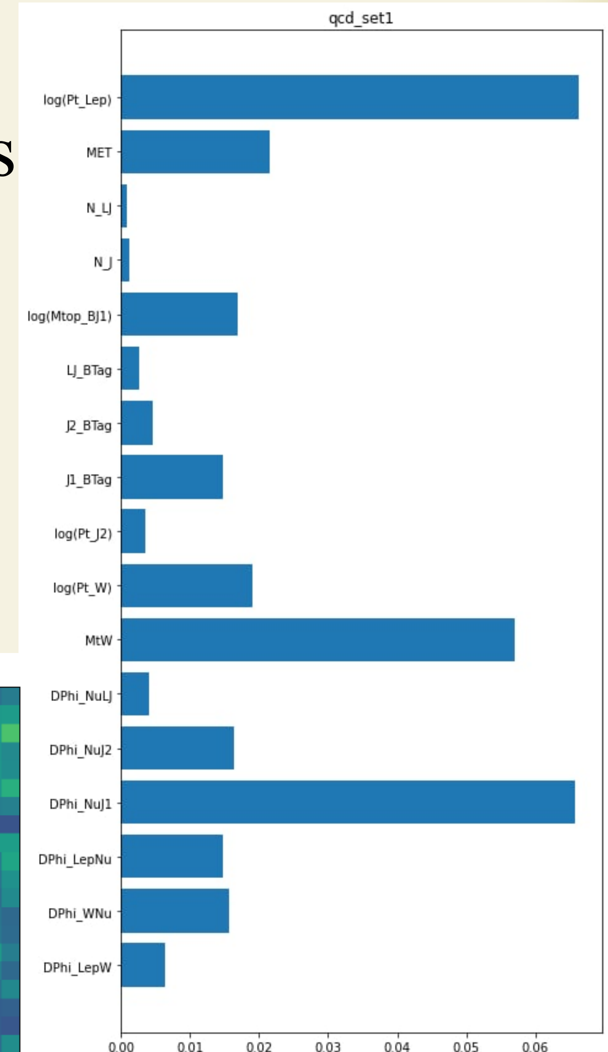
The proposed recipe is simple, need to use the following classes:

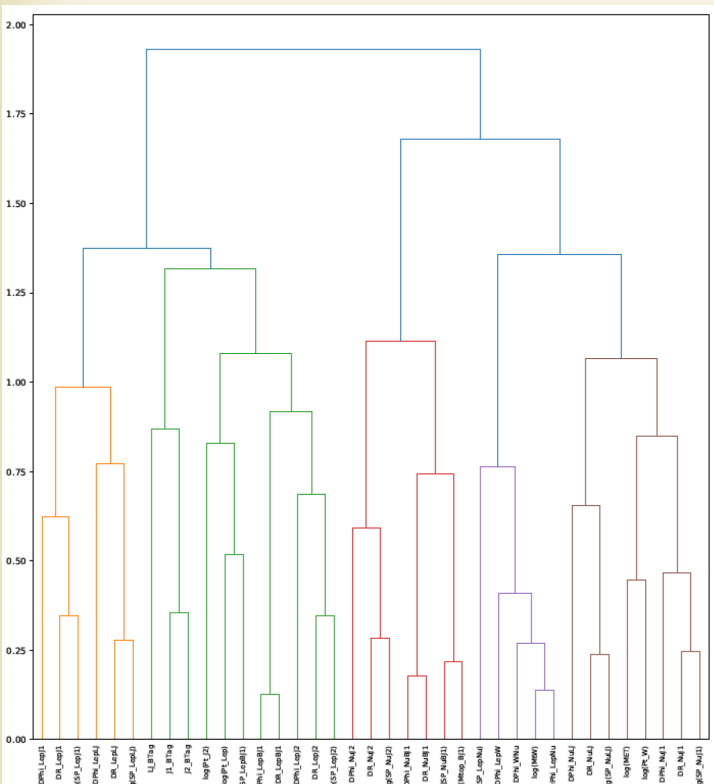
- scalar products of 4-momenta of the final particles,
- Mandelstam variables (only **s** are available for pp; **t**,**u** for lepton colliders),
- transverse momenta and pseudorapidity of the final particles (to approximate t-channel Mandelstam variables which depends on initial particles momenta).

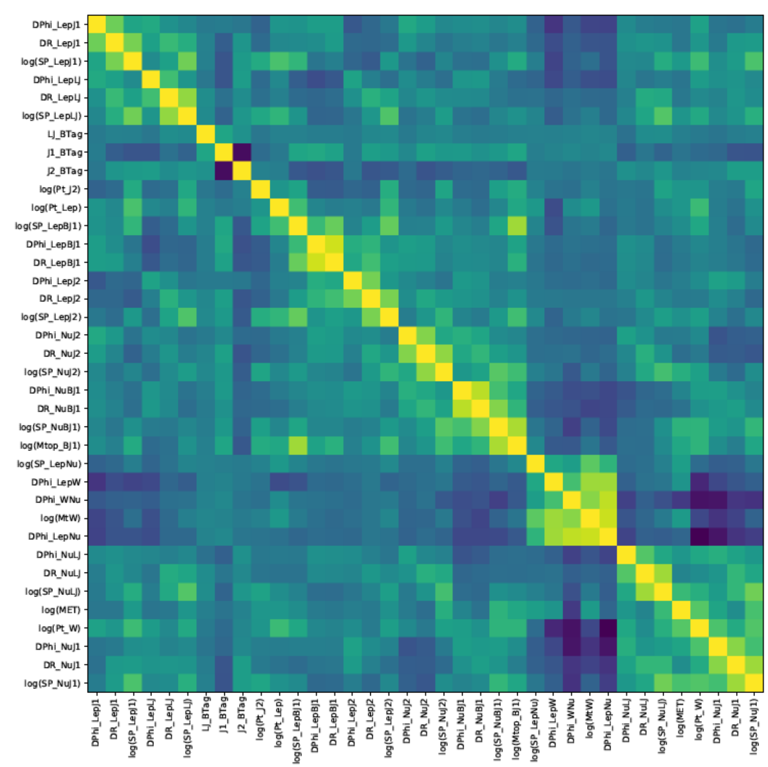
The proposed set of raw observables covers the kinematic differences in hard processes. In additional, it is possible to add some other type of information (e.g. b-tagging discriminant, charge of lepton, …)

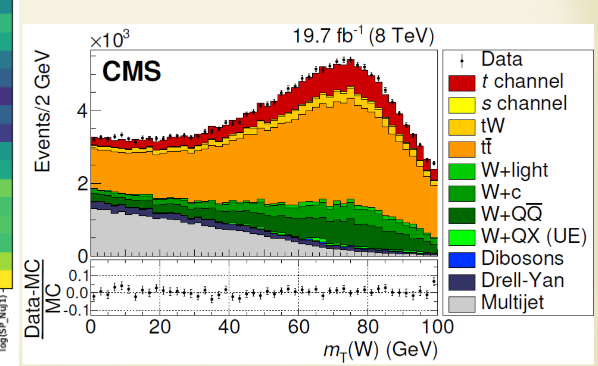
Optimization of input variables for DNN

- Exclude variables with high linear correlations
- Check simulation of each variable (compare with data, or different simulations)
- Can check an importance of each variable
- Transform variables to the same dynamic scale [$(x-\bar{x})/\sigma$, log()]

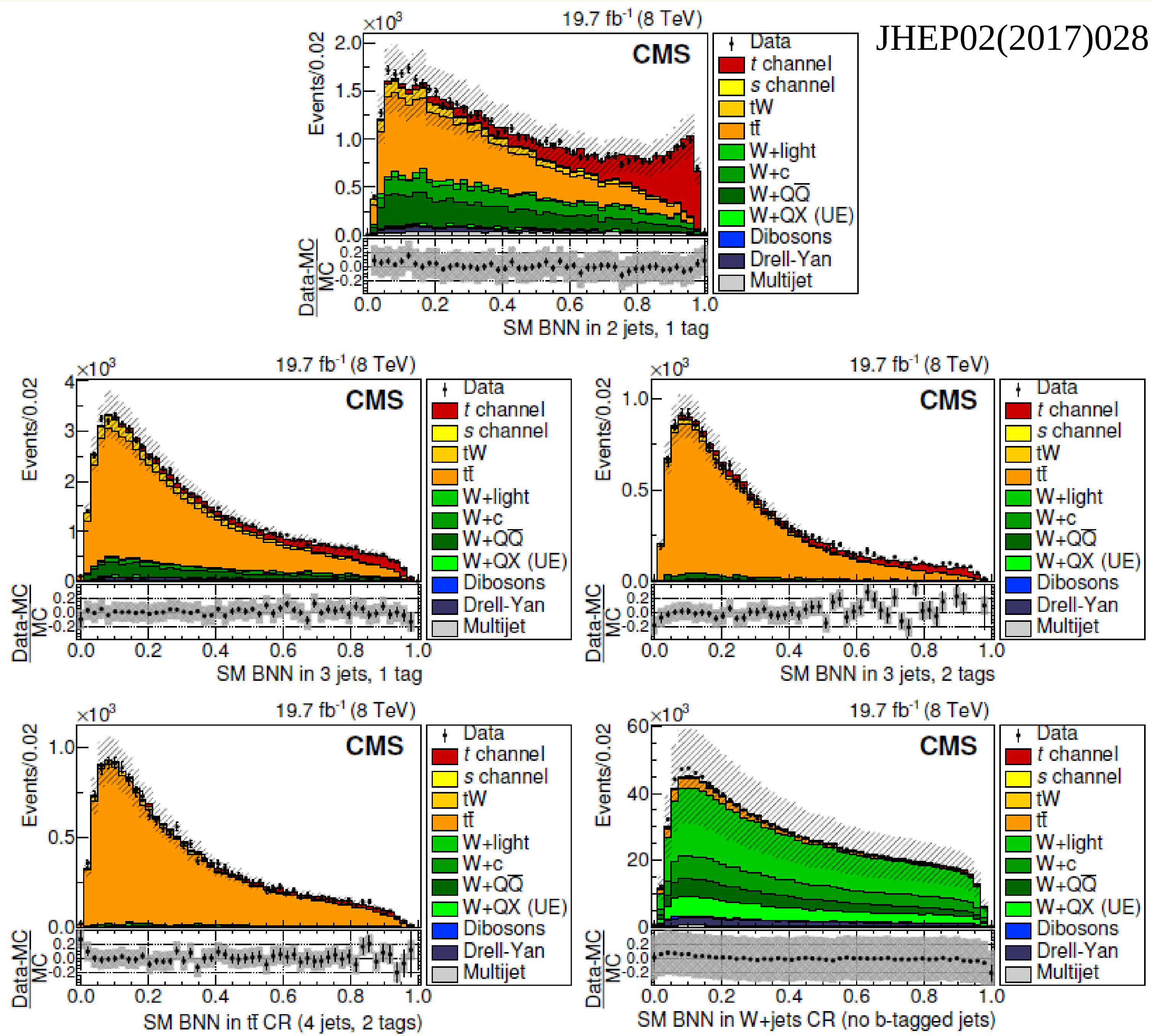






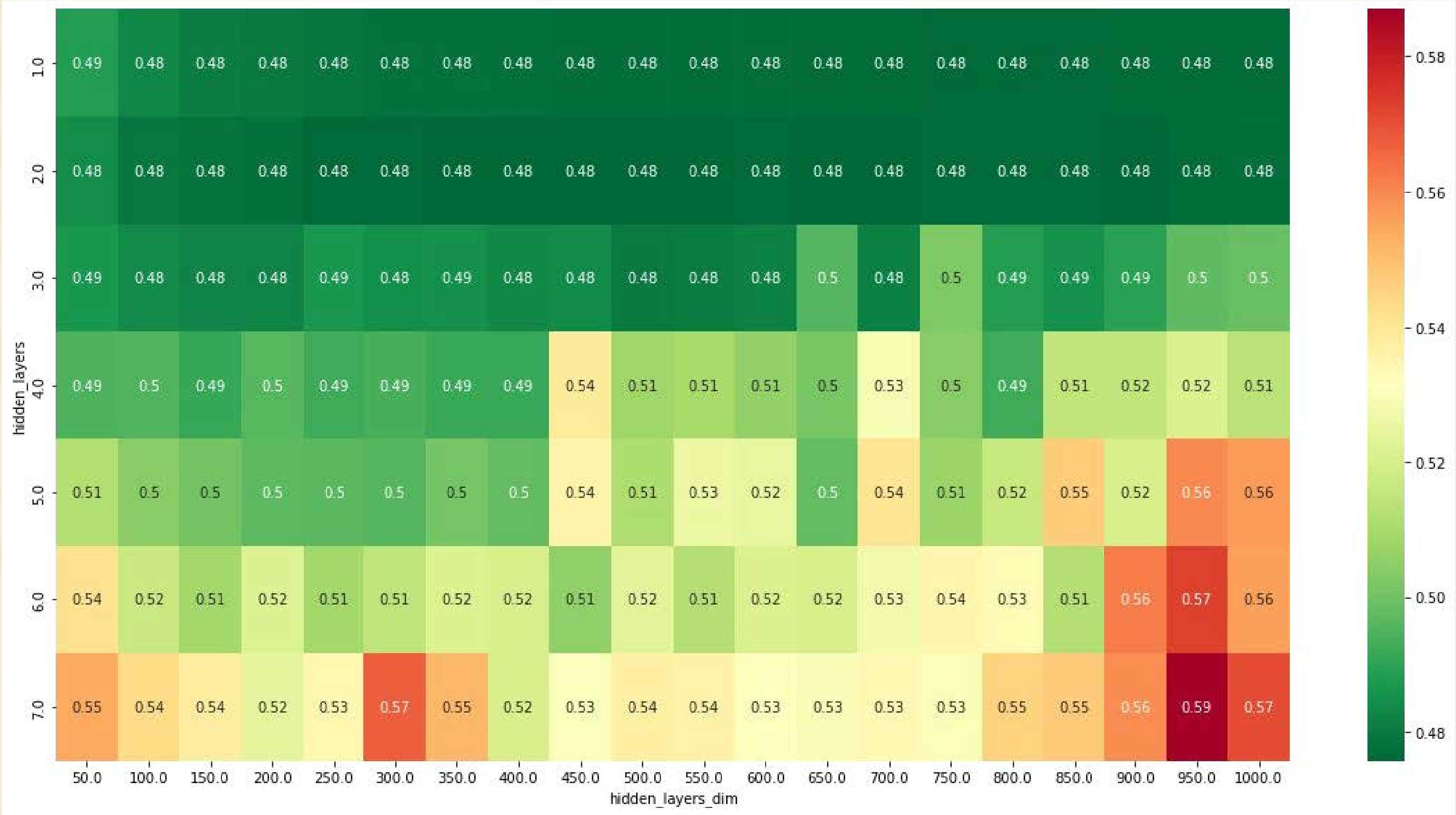


Check network for stability in different control regions, for different backgrounds

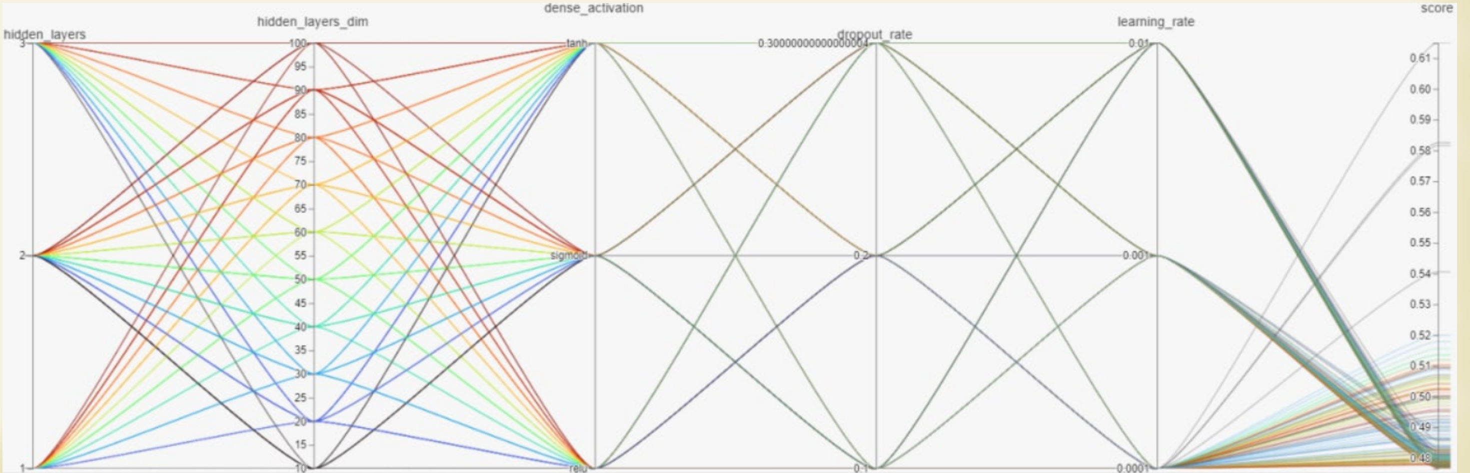


JHEP02(2017)028

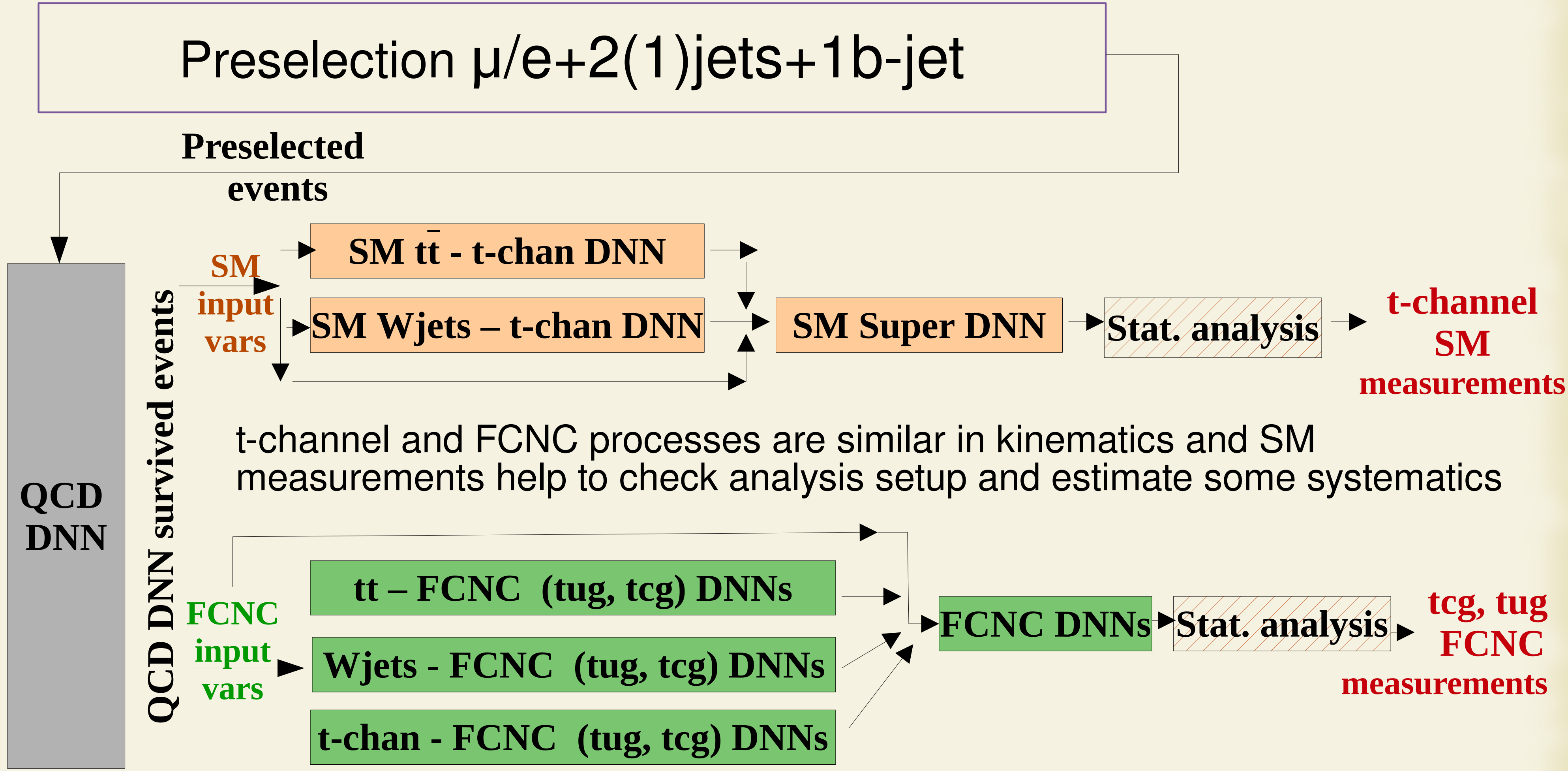
Tune hyper-parameters: number of nodes and hidden layers, dropout, training parameters, …, based on ROC/AUC, Score or other metrics (Optune, Keras tuner, ...).



Phys.Atom.Nucl. 85 (2022) 6, 708-720



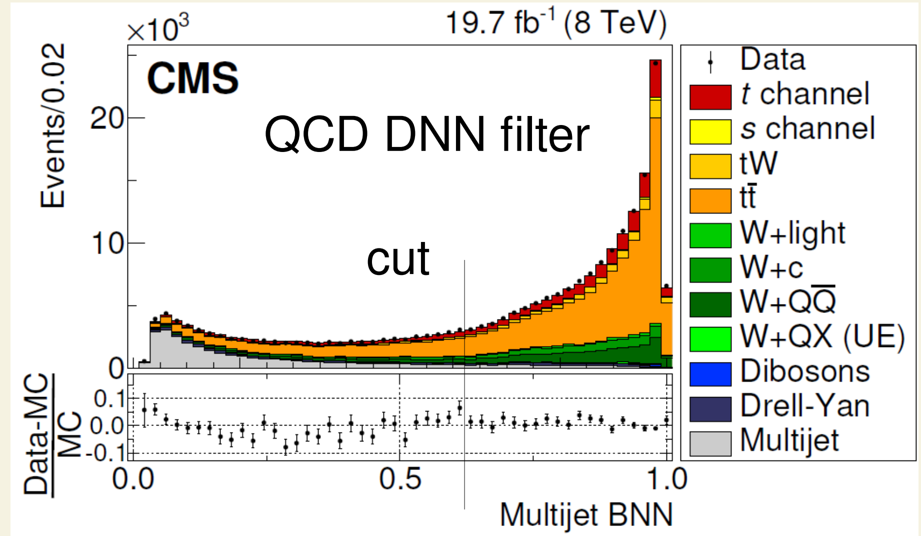
Complicated/Efficient architectures of DNN

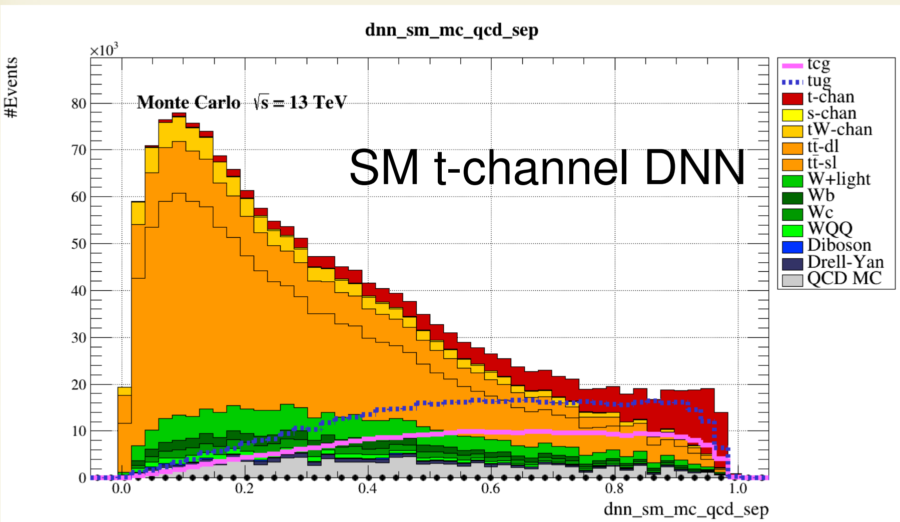


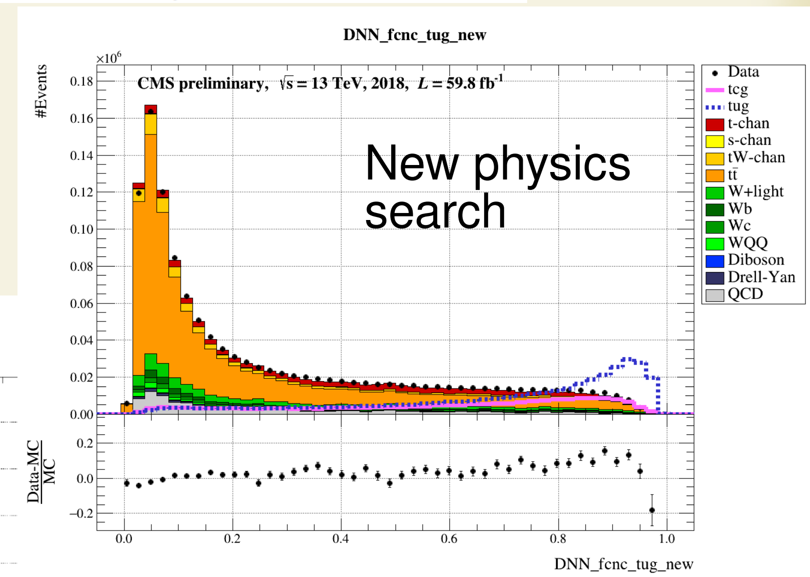
Some published results:

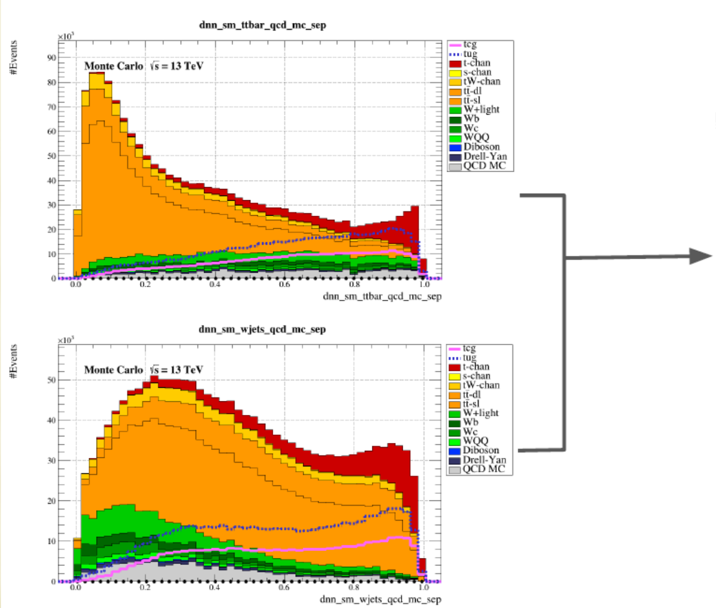
1) 7&8 TeV JHEP02(2017)028 (FCNC tqg & aWtb)
2) 14 TeV HL-LHC YR, PAS-FTR-18-004; extrapolation to HE-LHC (FCNC tqg)
3) FCC 100 TeV, CDR vol.1 (FCNC tqg)

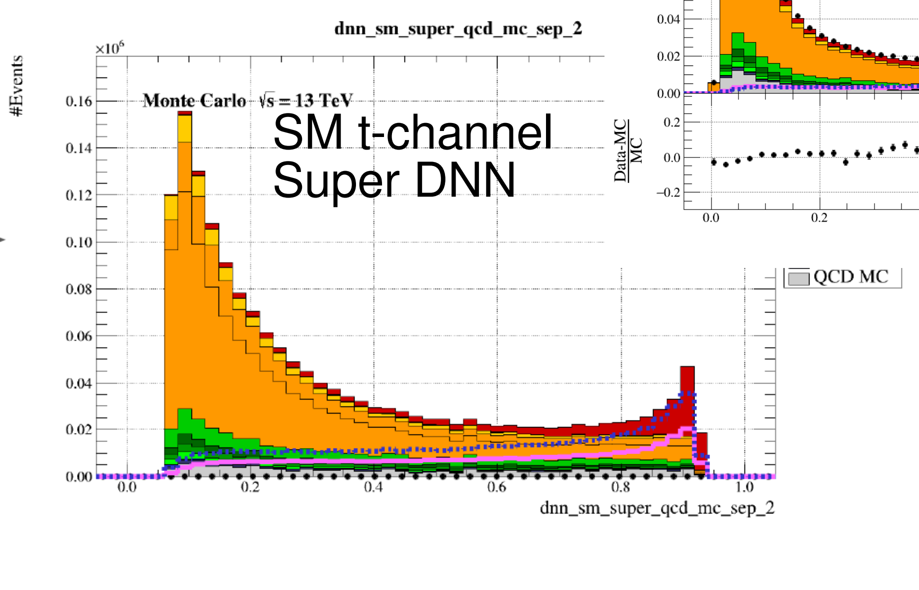
Cascades and ensembles of DNNs







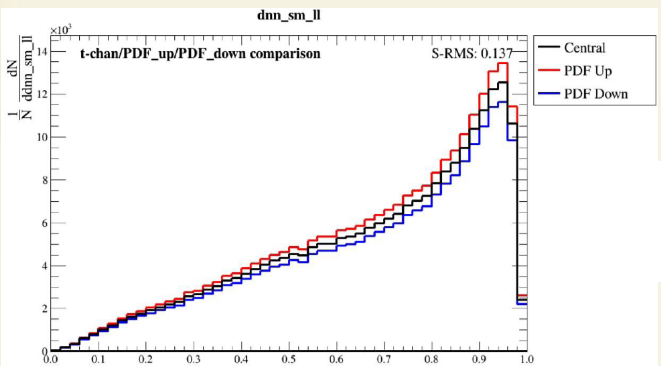




Uncertainties

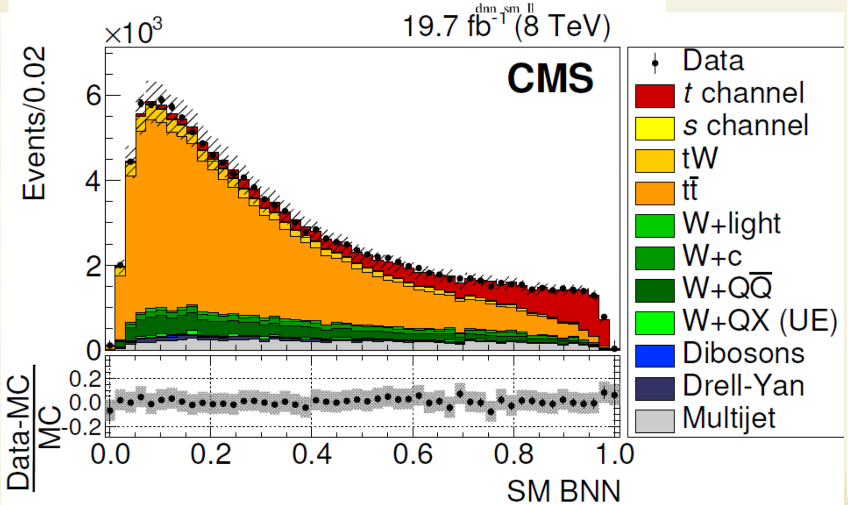
Aleatoric (statistical noise), Epistemic (systematical shift) uncertainties.

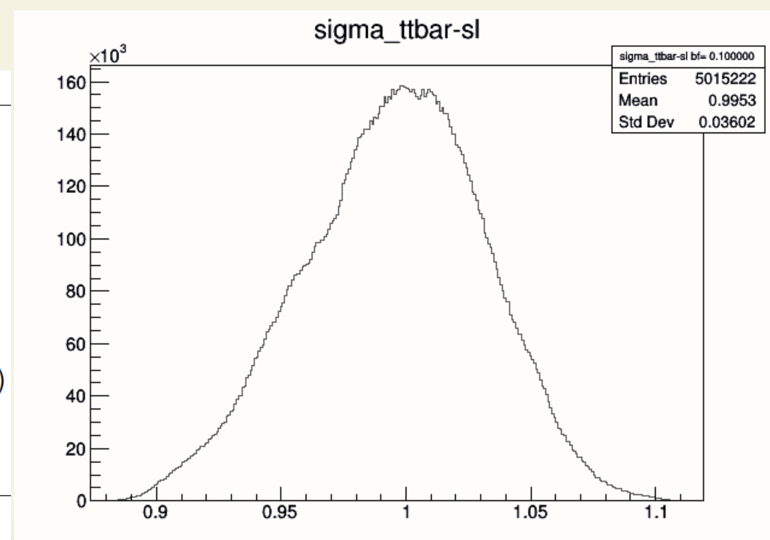
1. statistical uncertainty (data)
2. normalisation systematic uncertainties (cross sections, luminosity)
3. shape systematic uncertainties, correlated shift in all histogram bins (identification,corrections, …)
4. shape systematic uncertainties, uncorrelated shift between bins (scale, PDF, theory uncertainties)
5. finit statistics in Monte-Carlo generated event samples (Barlow-Beaston method)

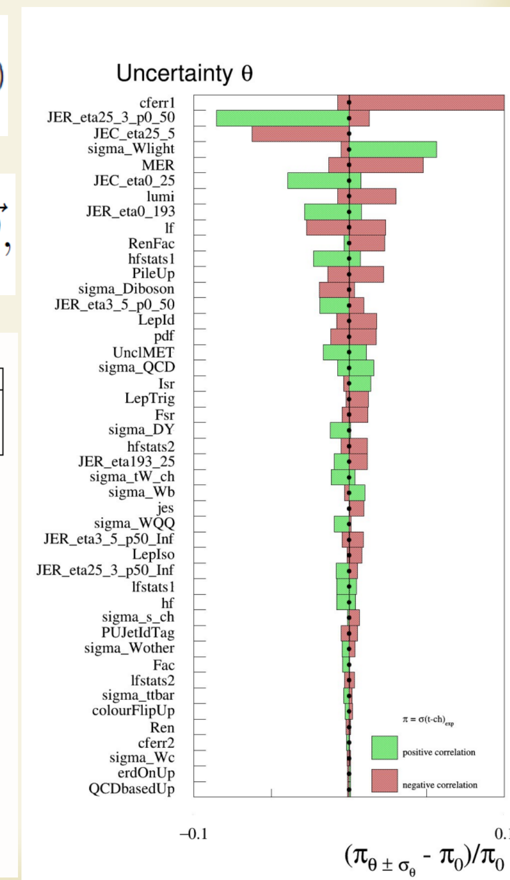


$$p_m(d|\vec{p}) = \prod_{i=1}^{N}\prod_{l=1}^{b_i} \mathrm{Poisson}(d_{i,l}|m_{i,l}(\vec{p}))$$

$$p(\vec{\mu}_s|d) = \int p(d|\vec{\mu}_s,\vec{\mu}_b,\vec{\theta})\frac{\pi(\vec{\mu}_s)\pi(\vec{\mu}_b)\pi(\vec{\theta})}{\pi(d)}\mathrm{d}\vec{\mu}_b\mathrm{d}\,\vec{\theta},$$







Some general remarks, based on experience

1) Increasing the complexity of DNN (number of nodes, layers) leads to complicate training and usually decrease efficiency of DNN.

2) Input information (input vector) should contains complete set of important observables, without overabundant information which complicates training.

3) Decrease the order of nonlinearity in the task for DNN
e.g. $F(x)=x^2 \rightarrow NN(x^2)$ not $NN(x)$

3) Preprocessing of input data improves training. Understand your data.

4) Use minimally sufficient size of DNN (number of nodes, layers). Use tuners of hyper-parameters.

5) Control and minimize overfitting (dropout, regularisation, test samples) to keep stability of the result.

5) Control propagation of input uncertanties to DNN output (precision means low uncertanty, not only efficient classification).