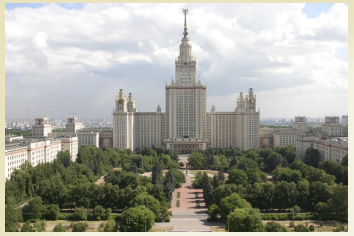
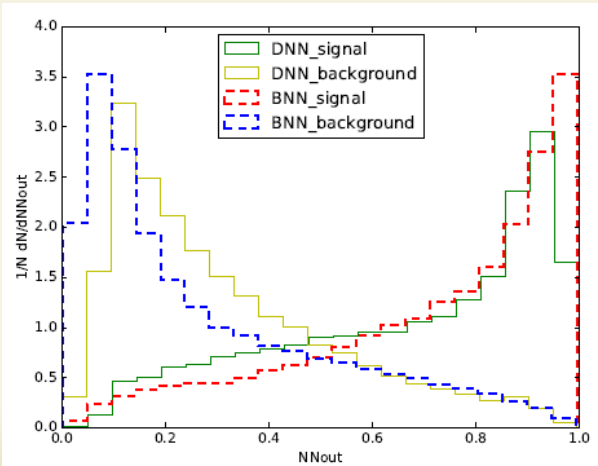


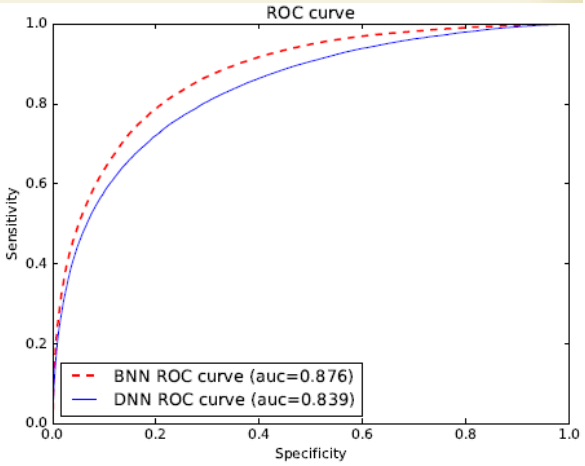
Is there a general recipe to form optimal input space for deep learning analysis of the collider hard scattering processes?

L. Dudko, M. Perfilov, P. Volkov, G. Vorotnikov *SINP MSU, Moscow*

[Int.J.Mod.Phys.A 35 (2020) 21, 2050119, hep-ph:2002.09350]

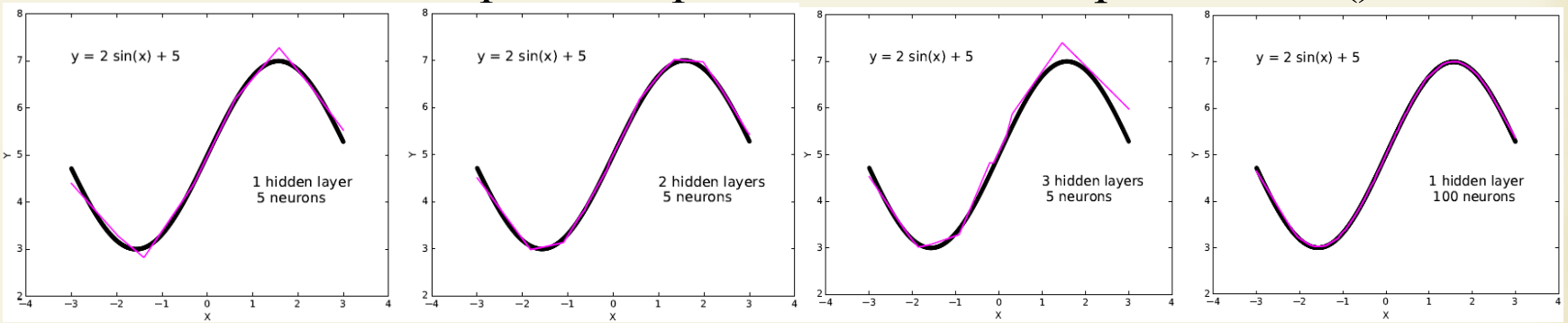
- Deep learning (DNN) makes possible to move from sophisticated input set of observables to some general set, and leave sensitive features extraction to DNN
- Kolmogorov-Smirnov representation theorem can be used as a proof that it will be possible in principle. NN is able to reproduce all continuous functions.
- $2 \to n$ particles; $(3n-4)$ independent variables; $d\sigma \sim M^2(p_i \cdot p_f, s, t, u)$
- As a benchmark we use distinguishing of single top quark production from pair top quark production. Not a trivial example, but very well investigated already [JHEP02(2017)028]
- First compare naive set of final particles 4-momenta and highly optimized sophisticated set of variables from CMS analysis.
  ROC curve demonstrates suboptimal efficiency.
  (BNN uses sophisticated optimized set, DNN is trained with general set)





Why the 4-momenta is not an optimal general set?

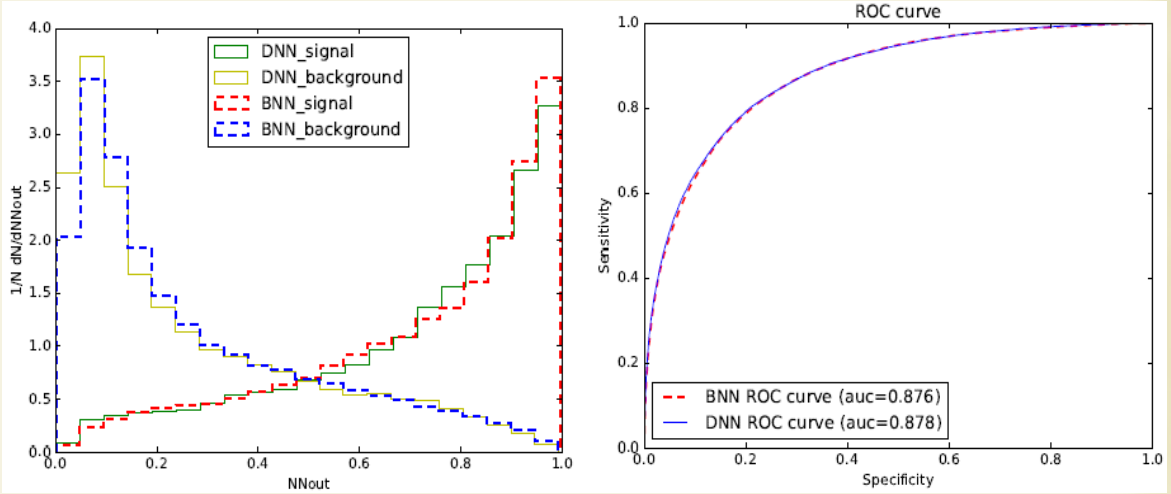
- The only non linear part of feed-forward NN is the activation function. M^2 is a function of scalar products of 4-momenta and quadratic forms (s,t,u), not 4-momenta. Therefore, it is needed to decrease the order of non-linearity or increase the number of hidden nodes. Simple example with ReLU to reproduce sin():



- M^2 is the function of initial particles momenta as well (not available for hadron colliders). In the massless case p_{in} can be represented as $\hat{t}_{i,f} = -\sqrt{\hat{s}}e^{Y}\, p_T^f e^{-|y_f|}$ and approximated with P_T and pseudorapidity [Phys.Atom.Nucl. 71 (2008) 2, 388-393]

► General recipe to form input space for DNN analysis of the collider hard processes: take scalar products of 4-momenta of the final particles, Mandelstam variables (s,t,u), transverse momenta and pseudorapidity.

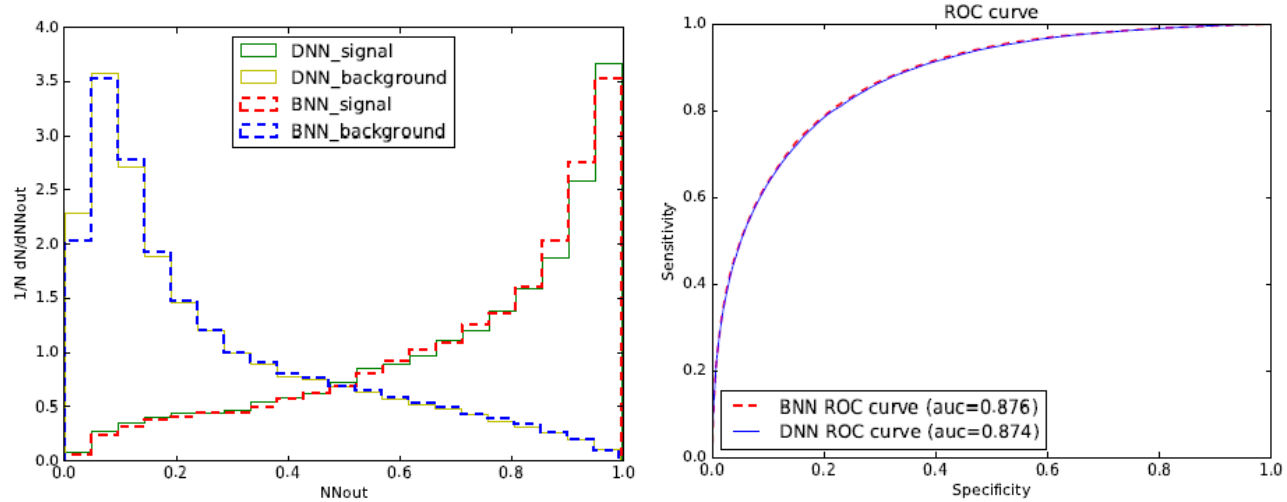
ROC curve demonstrates desired efficiency with the recipe



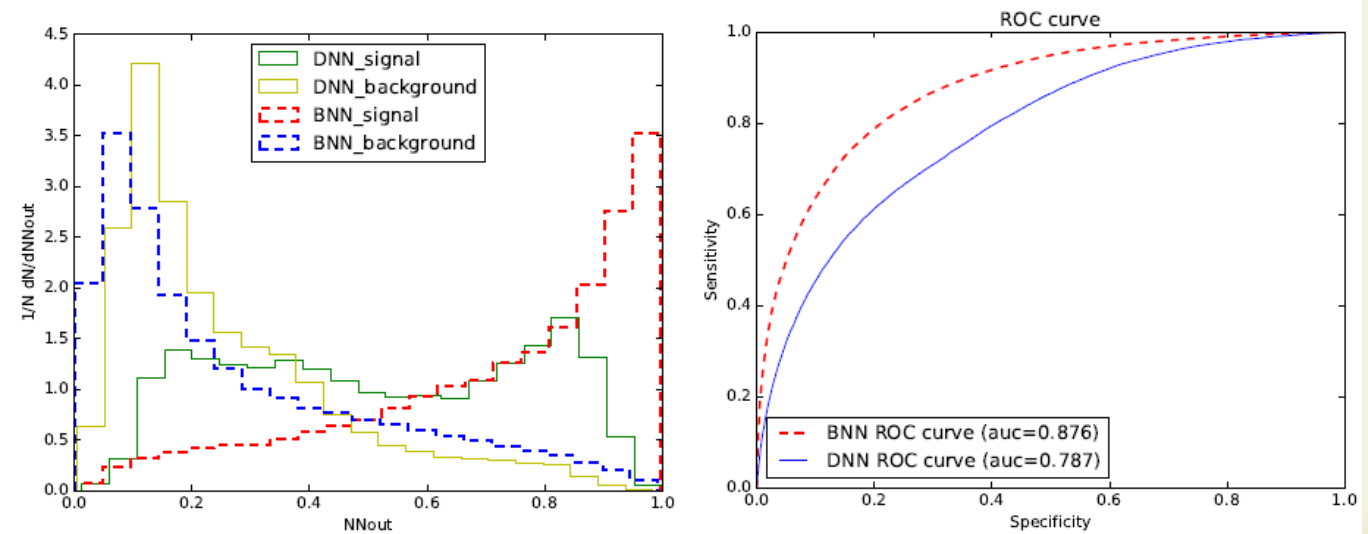
Back Up

cross checks

- Comparison of the Bayesian NN (BNN) and Deep Learning NN (DNN) methods with the same highly optimized benchmark set of variables:

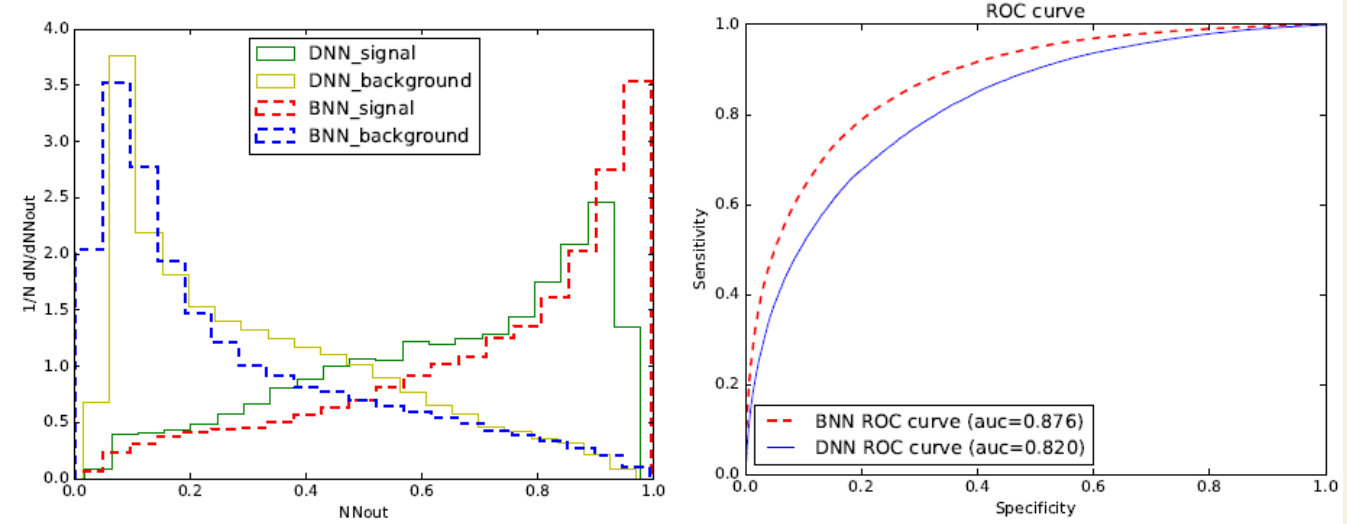


- Comparison of benchmark set and set with only scalar products of 4-momenta of final particles.

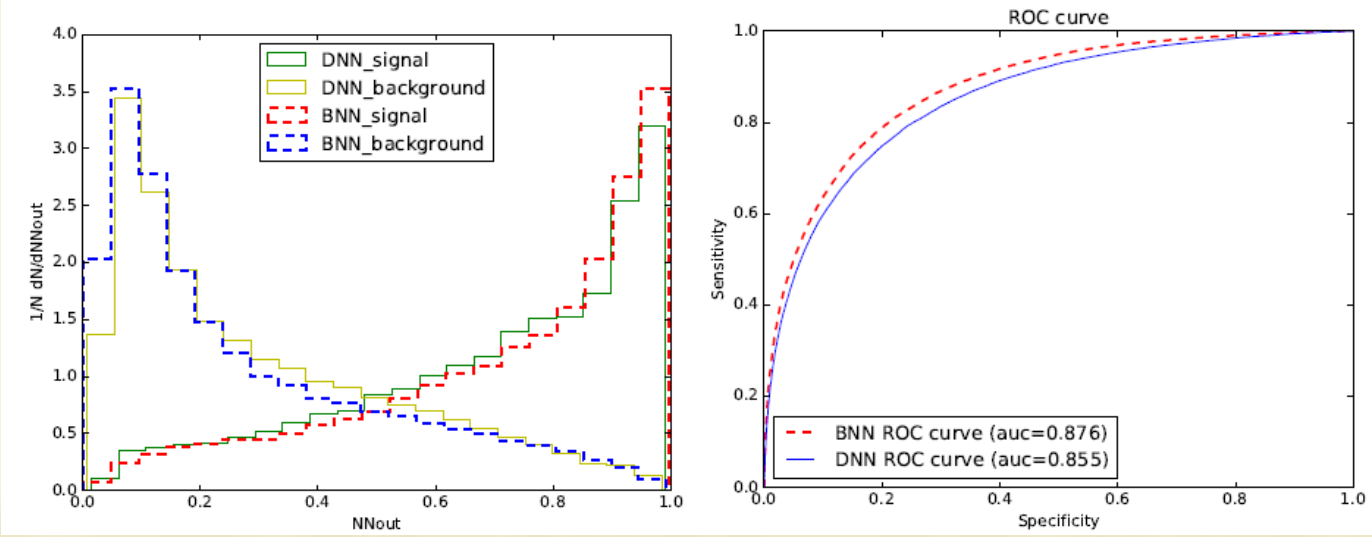


cross checks

The comparison of benchmark BNN with DNN trained on the scalar-products of four-momenta of the final particles and s-channel Mandelstam variables as the set of input variables. DNN has five hidden layers. The left plot demonstrates outputs of DNN and BNN for the signal and background processes. The ROC curves are shown in the right plot.

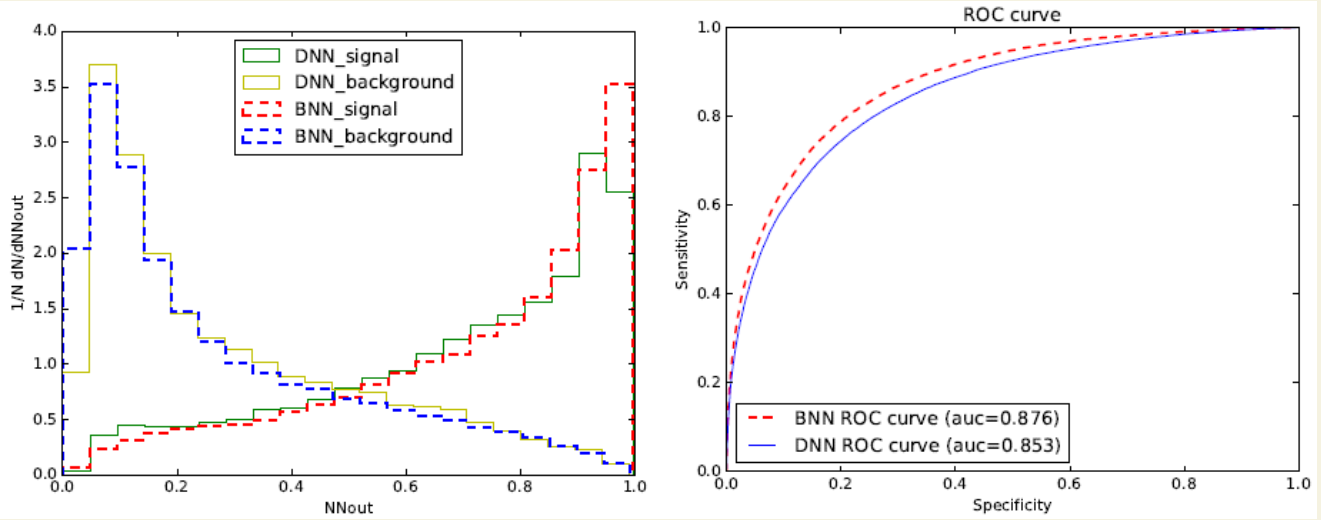


The comparison of benchmark BNN with DNN trained on the scalar-products of four-momenta and four-momenta of the final particles as the set of input variables. DNN has five hidden layers.

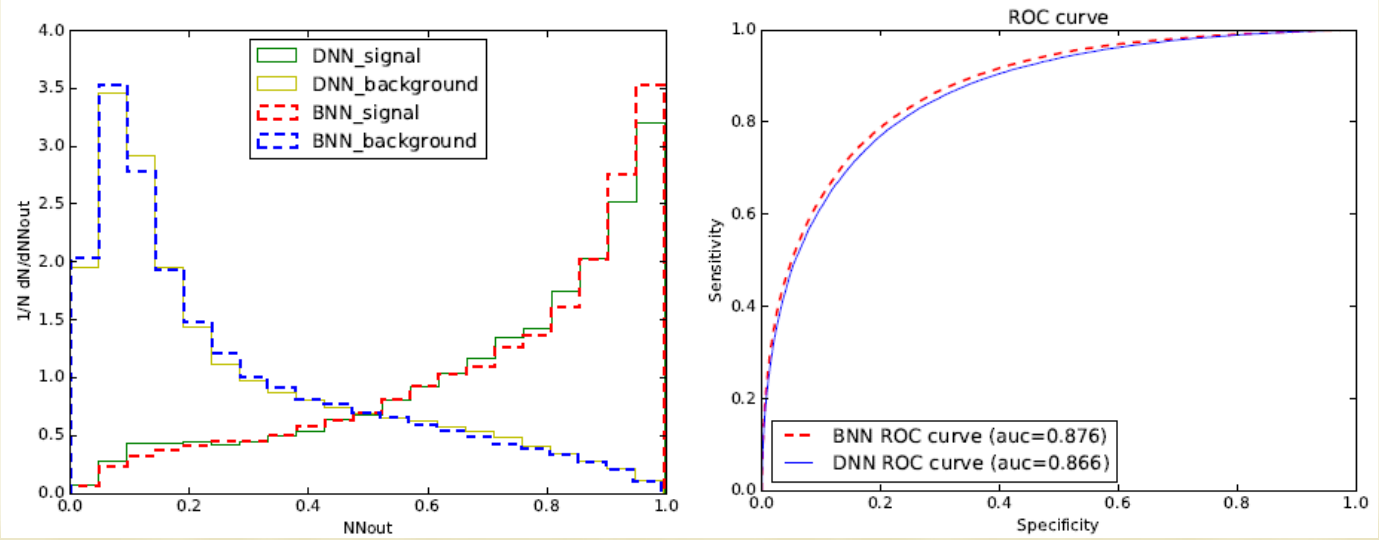


cross checks

The comparison of benchmark BNN with DNN trained on the scalar-products of four-momenta, four-momenta of the final particles and s-channel Mandelstam variables as the set of input variables. DNN has five hidden layers. The left plot demonstrates outputs of DNN and BNN for the signal and background processes. The ROC curves are shown in the right plot.



The comparison of benchmark BNN with DNN trained on the scalar-products of four-momenta, transverse momenta of the final particles and Mandelstam variables as the set of input variables. DNN has three hidden layers.



cross checks

- The comparison of benchmark BNN with DNN trained on the scalar-products of four-momenta, four-momenta and transverse momenta of the final particles and Mandelstam variables as the set of input variables. DNN has five layers.

  ROC curve demonstrates the completeness of the recipe, 4-momenta does not add more information, but increase the dimensionality which leads to more difficult training.