

Нейронные сети. Применение нейронных сетей в анализе данных коллайдерных экспериментов.

- ~ Основные принципы
- ~ Практическое использование NN для анализа процессов рассеяния на коллайдерах
- ~ Глубокие сети и современные этапы анализа
- ~ Обсуждение

Л. Дудко

Лаборатория электрослабых и новых взаимодействий

Отдела экспериментальной физики высоких энергий

НИИЯФ МГУ

The Brain: an Amazingly Efficient "Computer"

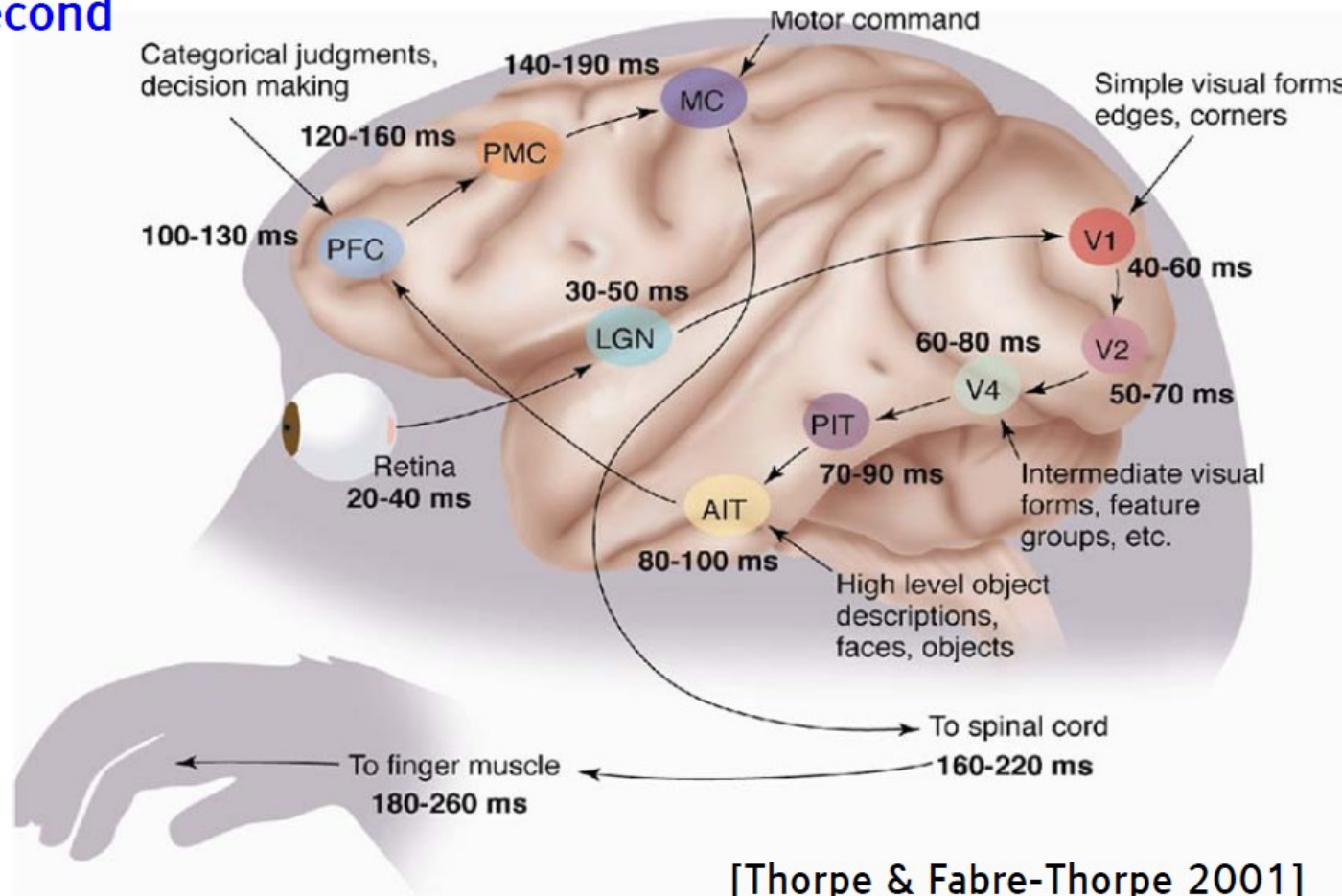
Y LeCun

- 10^{11} neurons, approximately
- 10^4 synapses per neuron
- 10 "spikes" go through each synapse per second on average
- 10^{16} "operations" per second

■ 25 Watts
▶ Very efficient

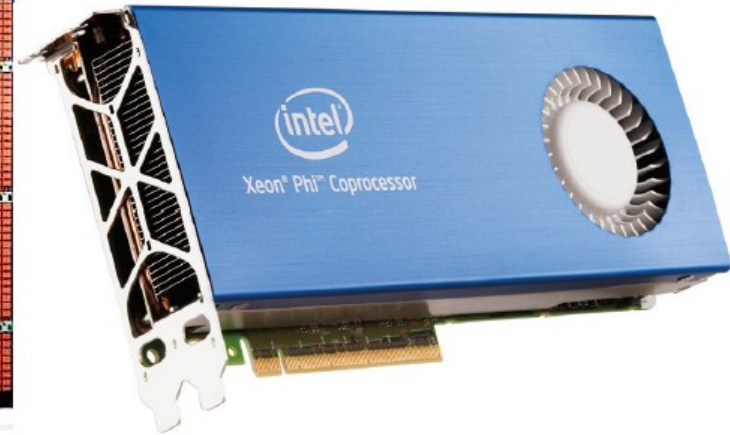
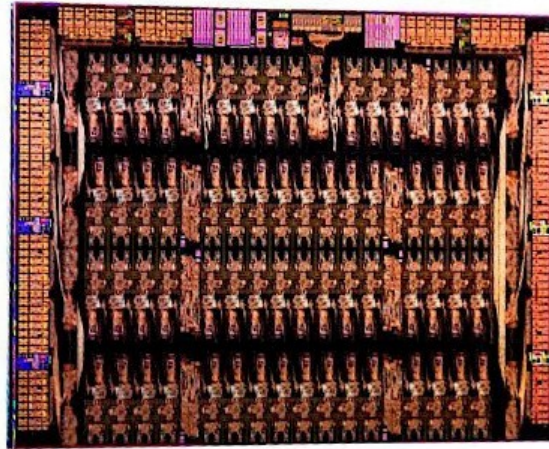
■ 1.4 kg, 1.7 liters

■ 2500 cm²
▶ Unfolded cortex



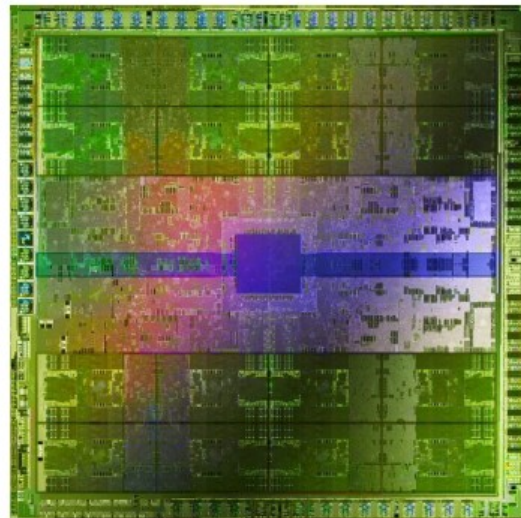
Intel Xeon Phi CPU

- ▶ 2×10^{12} operations/second
- ▶ 240 Watts
- ▶ 60 (large) cores
- ▶ \$3000



NVIDIA Titan-Z GPU

- ▶ 8×10^{12} operations/second
- ▶ 500 Watts
- ▶ 5760 (small) cores
- ▶ \$3000

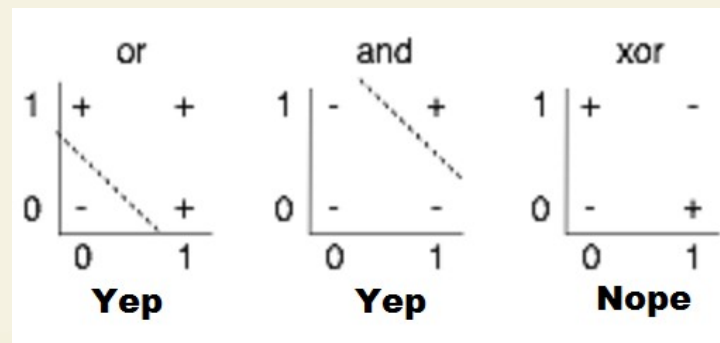
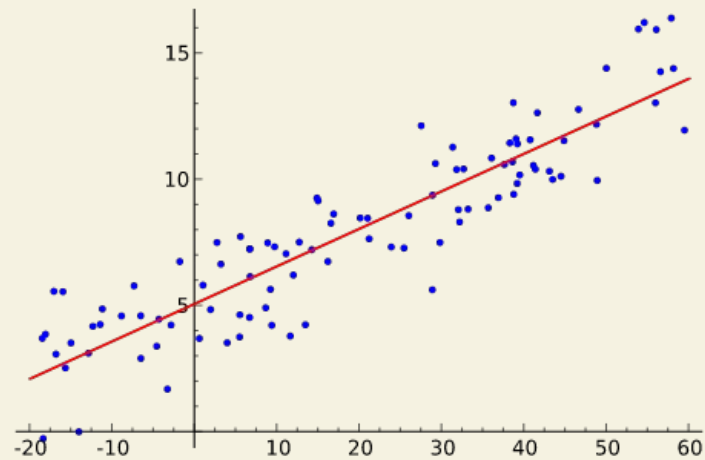
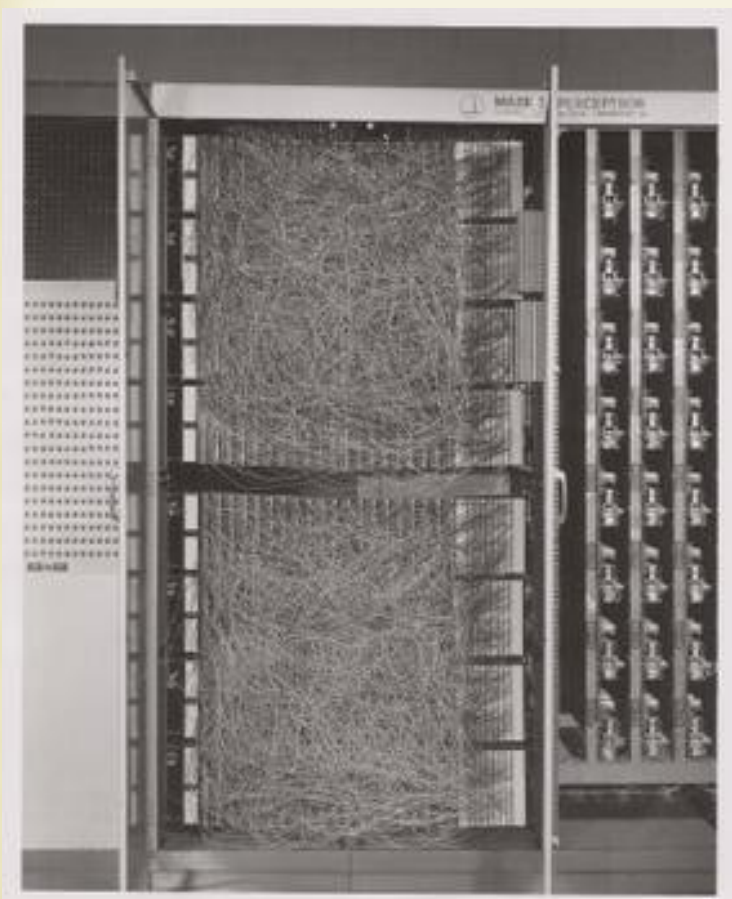


Are we only a factor of 10,000 away from the power of the human brain?

- ▶ Probably more like 1 million: synapses are complicated
- ▶ A factor of 1 million is 30 years of Moore's Law
- ▶ 2045?

Первая аппаратная реализация NN

Разработка современных методов машинного обучения (ML — machine learning) началась задолго до появления полноценных компьютеров, в современном понимании. Одним из наиболее эффективных современных методов многомерного анализа данных и машинного обучения является метод нейронных сетей (NN — neural net). Первой технической реализацией NN считается созданная в 1959 году F. Rosenblatt «Mark I Perceptron at the Cornell Aeronautical Laboratory» [Cornell University Library, [1] F. Rosenblatt, «The perceptron: A probabilistic model for information storage and organization in the brain.» Psychological Review, Vol 65(6), Nov 1958, 386-408; Rosenblatt, Frank (1957), «The Perceptron--a perceiving and recognizing automaton.» Report 85-460-1, Cornell Aeronautical Laboratory.]



Математические основы

Линейная регрессия – примерно 200 лет назад.

13-я проблема Гильберта (1900г.): возможно ли представление функции нескольких переменных в виде суперпозиции функций меньшего количества переменных.

Проблема была решена В. И. Арнольдом совместно с А. Н. Колмогоровым “Любая непрерывная функция любого количества переменных представляется в виде суперпозиции непрерывных функций одной и двух переменных.”

$$F(x_1, x_2, \dots, x_n) = \sum_{j=1}^{2n+1} g_j \left(\sum_{i=1}^n h_{ij}(x_i) \right)$$

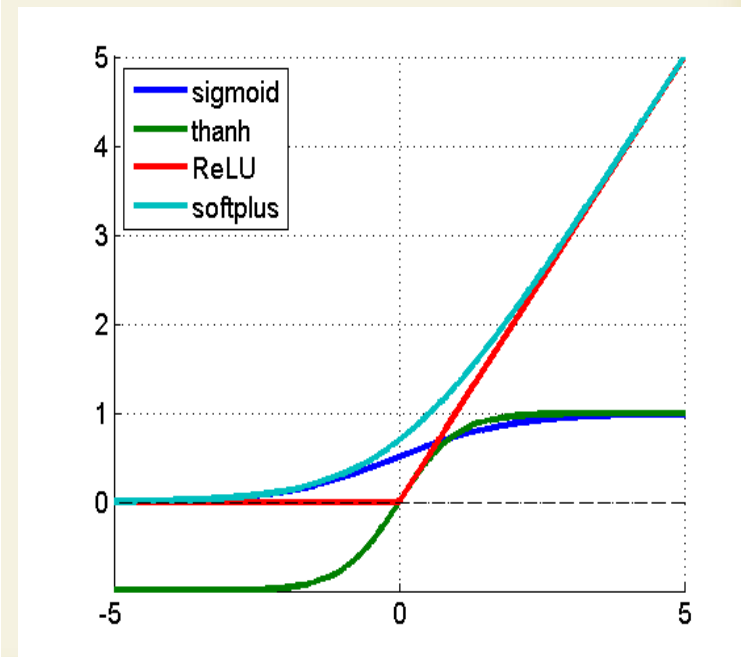
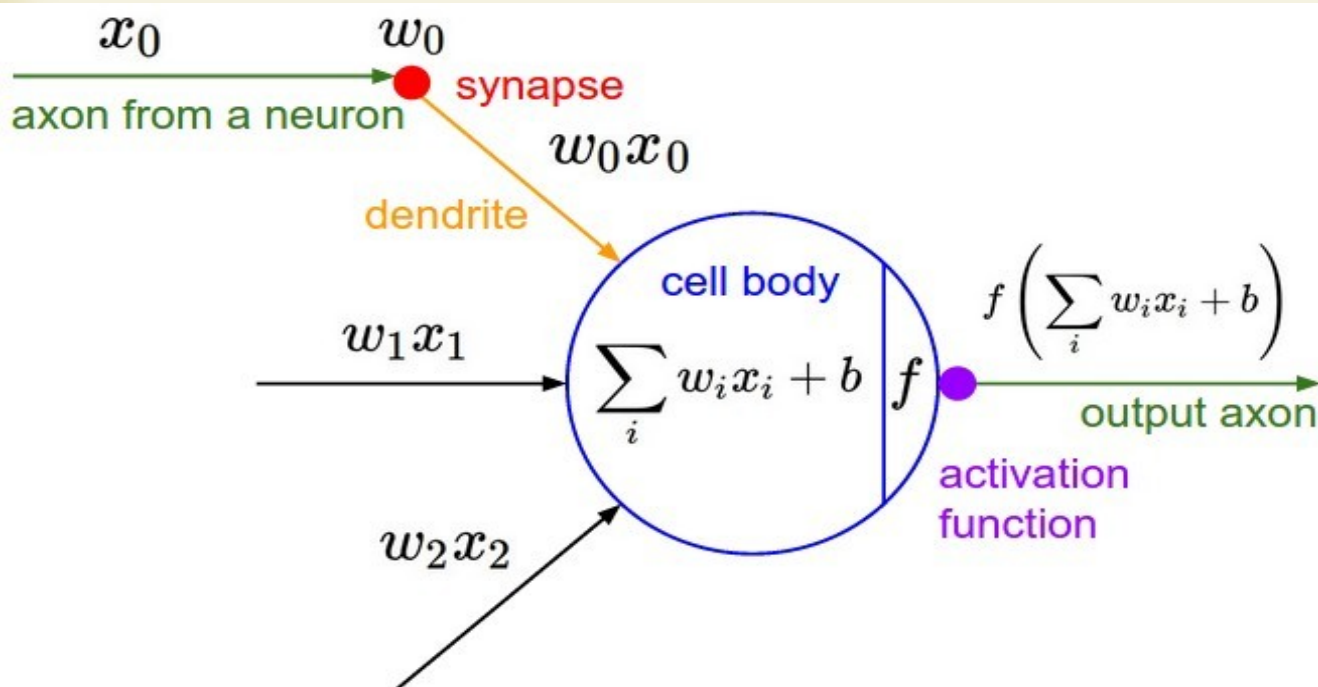
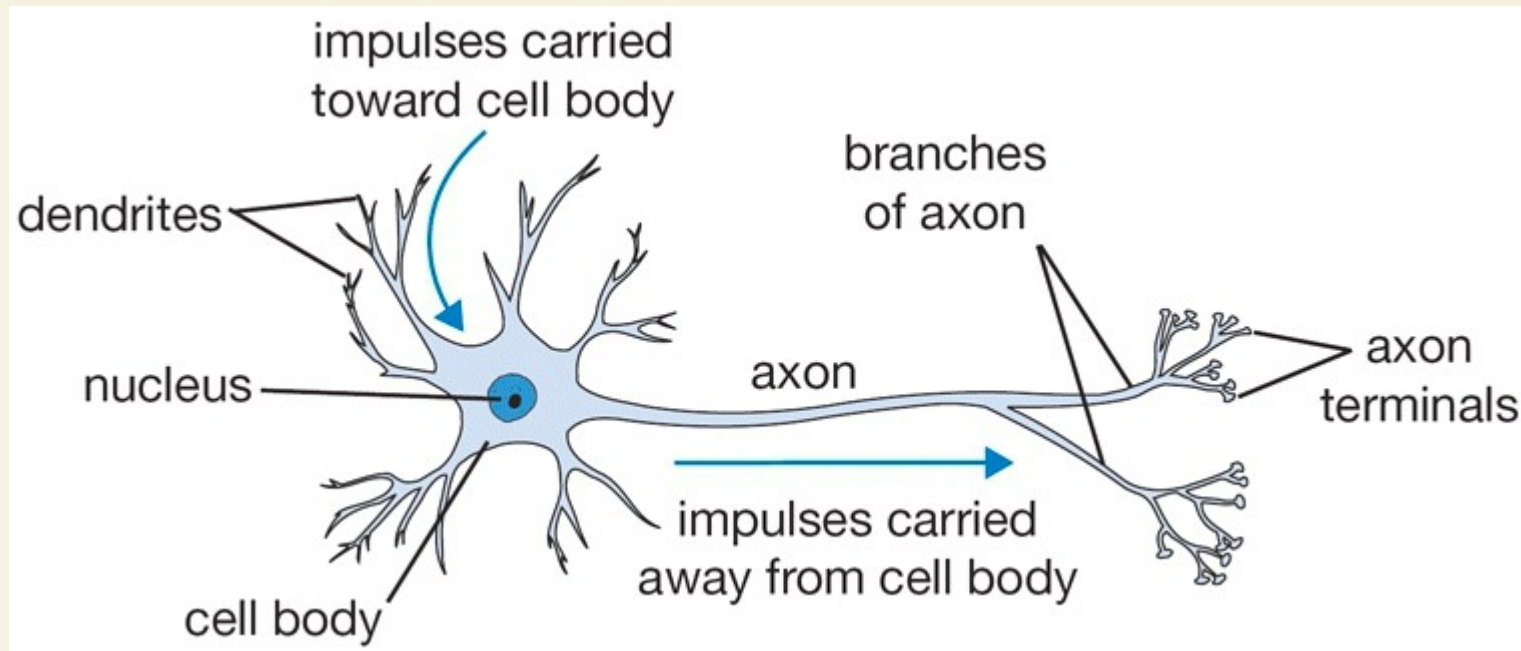
В.И.Арнольд, «О функциях трех переменных» ДАН СССР. - 1957. - Т.114. - N4. - С.679-681;

В.И.Арнольд, «О представлении непрерывных функций трех переменных суперпозициями непрерывных функций двух переменных», Матем. сб., 48(90):1 (1959), 3–74, <http://mi.mathnet.ru/msb4884> ;

А.Н.Колмогоров, «О представлении непрерывных функций нескольких переменных суперпозициями непрерывных функций меньшего числа переменных» ДАН СССР. - 1956. - Т.108. - N2. - С.179-182;

А.Н.Колмогоров, «О представлении непрерывных функций нескольких переменных в виде суперпозиций непрерывных функций одной переменной и сложения», ДАН СССР. — 1957. — Т. 114, вып. 5. — С. 953—956

Концепция перцептрона



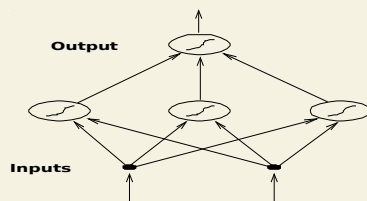
Полносвязная сеть с прямым распространением сигнала

Теорема. Существуют такое число k , набор чисел w_{ij} , θ_i и набор чисел ν_i , что функция

$$f(x_1, x_2, \dots, x_n) = \sigma\left(\sum_{i=1}^k \nu_i \sigma\left(\sum_{j=1}^n w_{ij} x_j + \theta_i\right) + \theta\right) \quad (6)$$

приближает данную функцию $F(x_1, x_2, \dots, x_n)$ с погрешностью, не более ε на всей области определения, где функция σ – некоторая нелинейная функция, к примеру сигмоид:

$$\sigma(x) = \frac{1}{1 + e^{-2x}}. \quad (7)$$

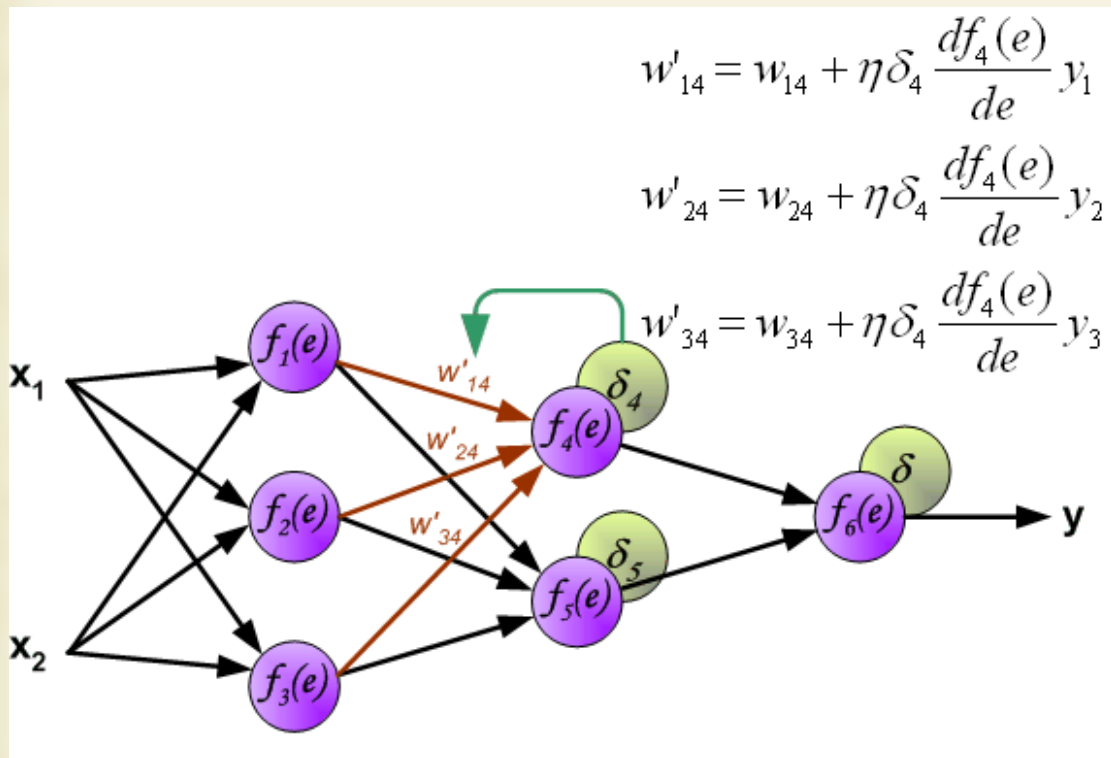


Наиболее популярным критерием сходимости при тренировке выбирается функция ошибки:

$$\chi^2 = \frac{1}{2N} \sum_{i=1}^N (f_i - t_i)^2, \quad (9)$$

здесь f_i реальный выход сети (6) для i -ого события, t_i желаемый выход, а N число тренировочных событий.

Обратное распространение ошибки (back-propagation)



[1] Обычная многослойная NN с прямым распространением ошибки способна аппроксимировать любую непрерывную функцию с заранее заданной точностью.

Возможная безуспешность таких попыток связана:

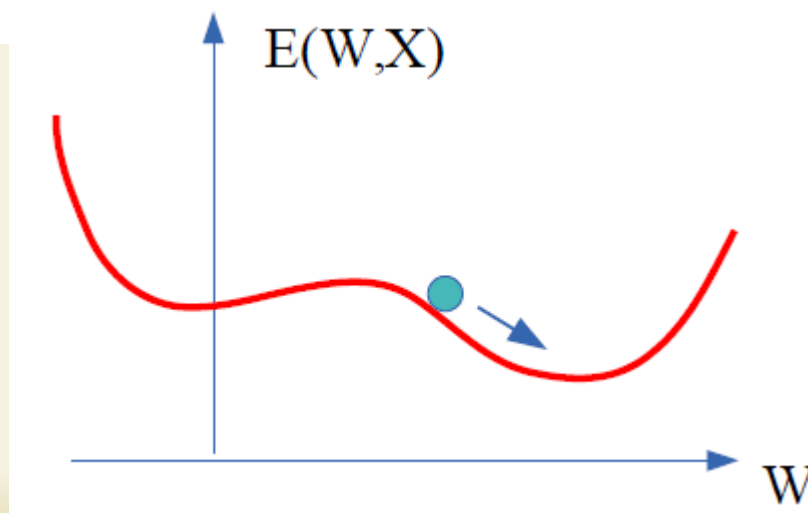
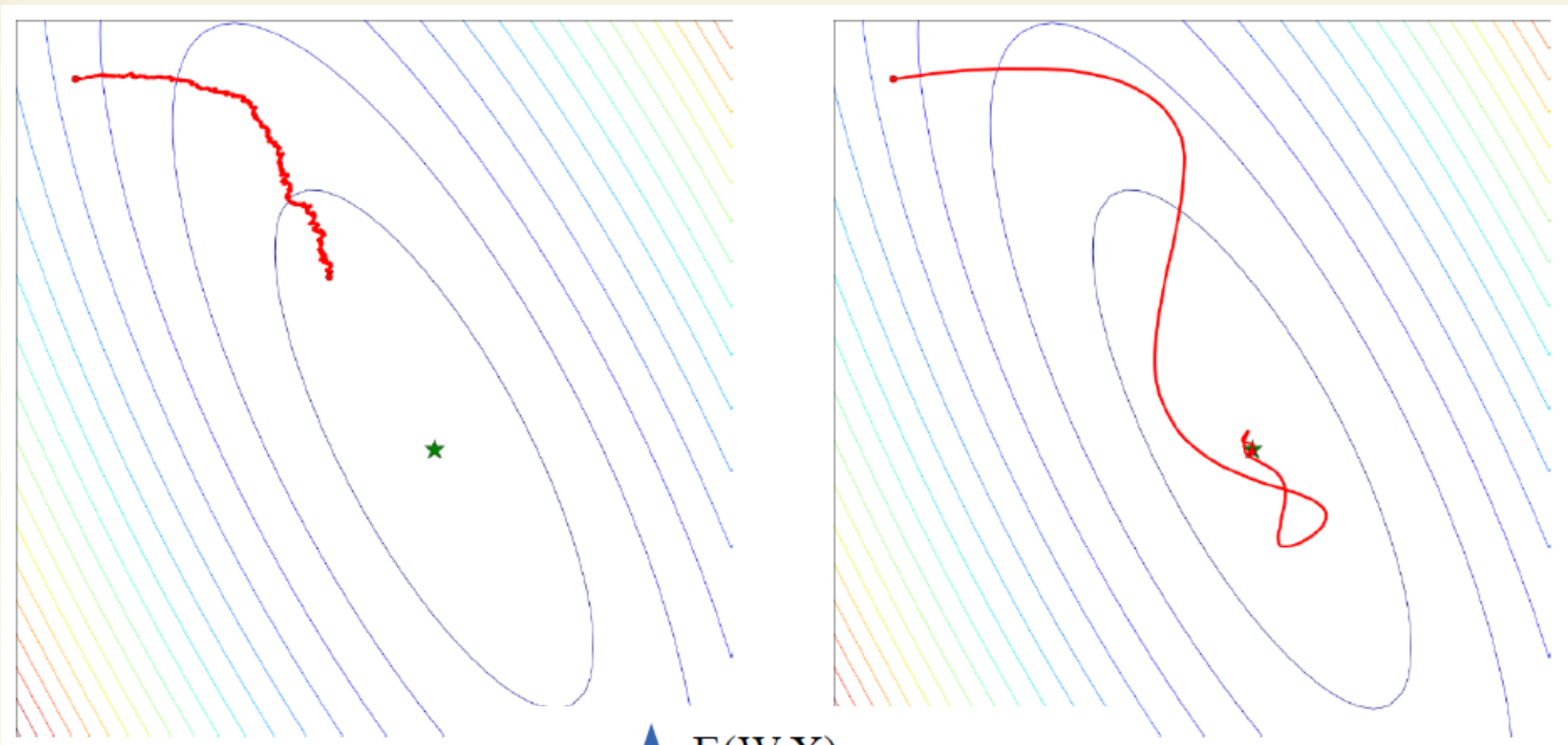
1. неадекватная тренировка
2. недостаточное количество скрытых узлов
3. недостаточная информация на входе

1) Hornik, Stinchcombe, White. Multilayer Feedforward Networks are Universal Approximators. Neural Networks, 1989, v. 2, 5. ;

2) Cybenko. Approximation by Superpositions of a Sigmoidal Function. Mathematical Control Signals Systems, 1989, 2.;

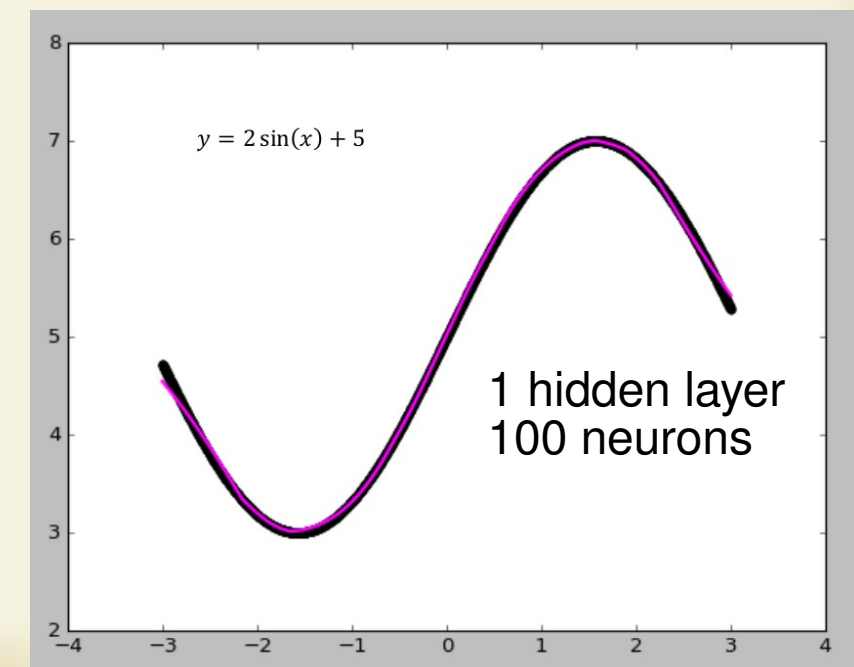
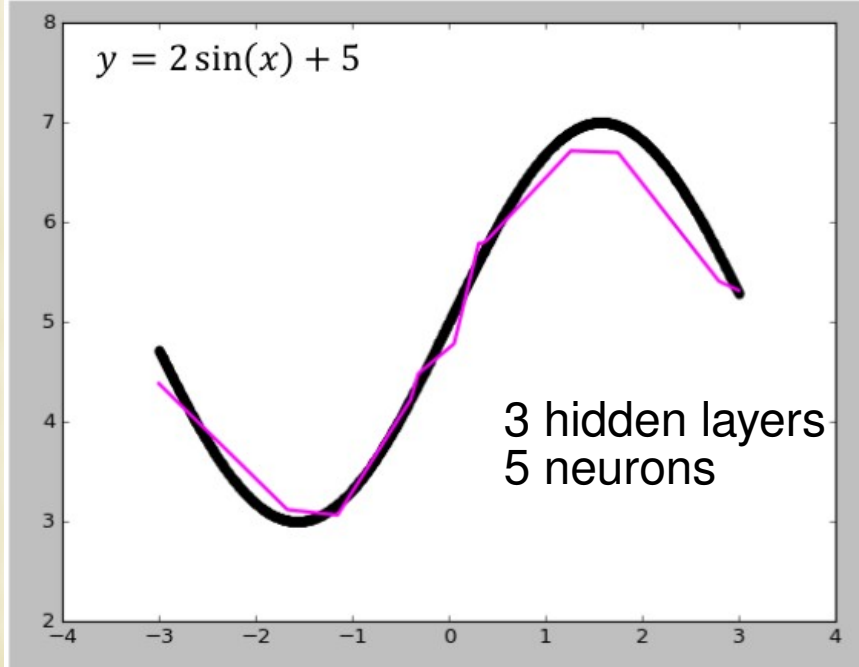
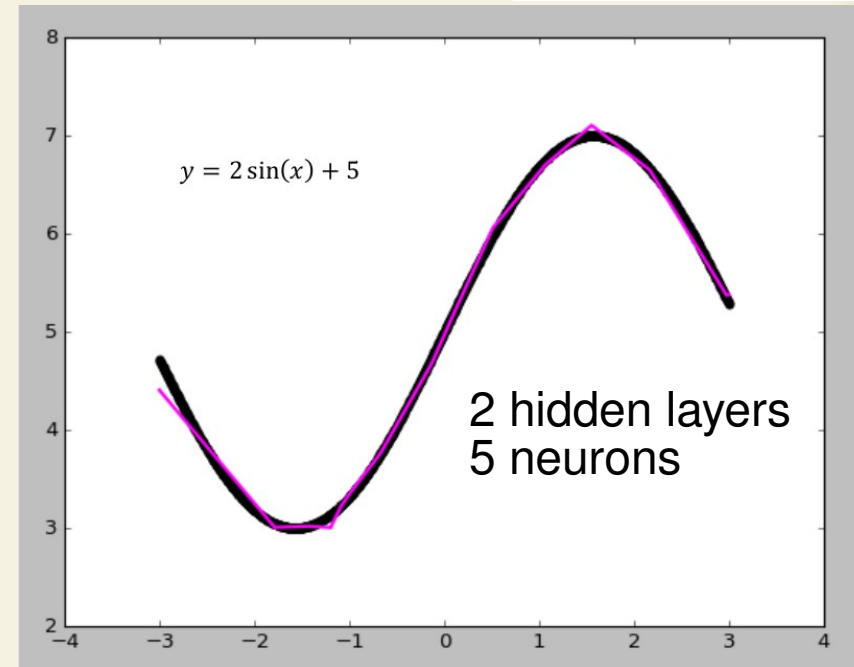
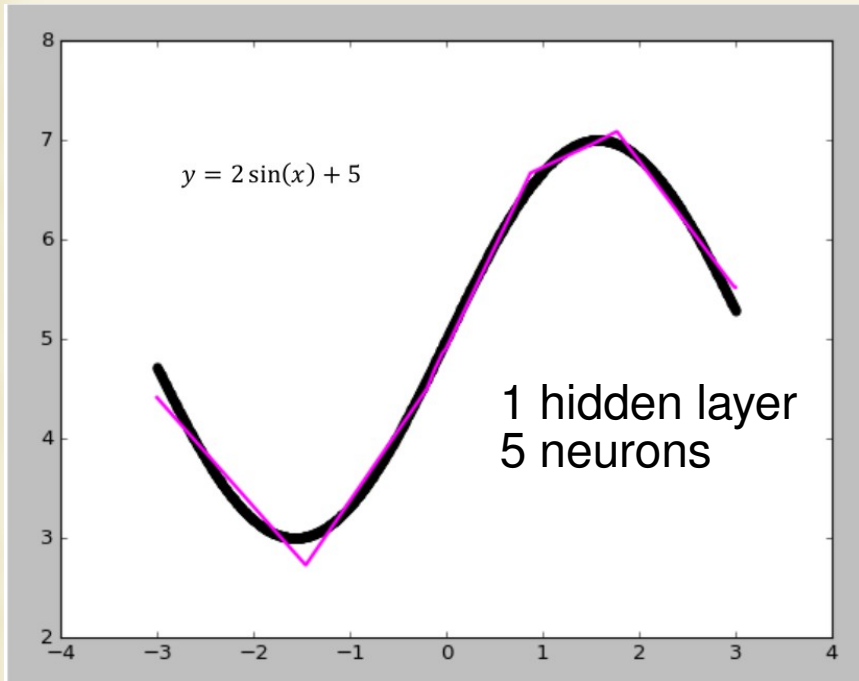
3) Funahashi. On the Approximate Realization of Continuous Mappings by Neural Networks. Neural Networks, 1989, v. 2, 3.

Тренировка NN

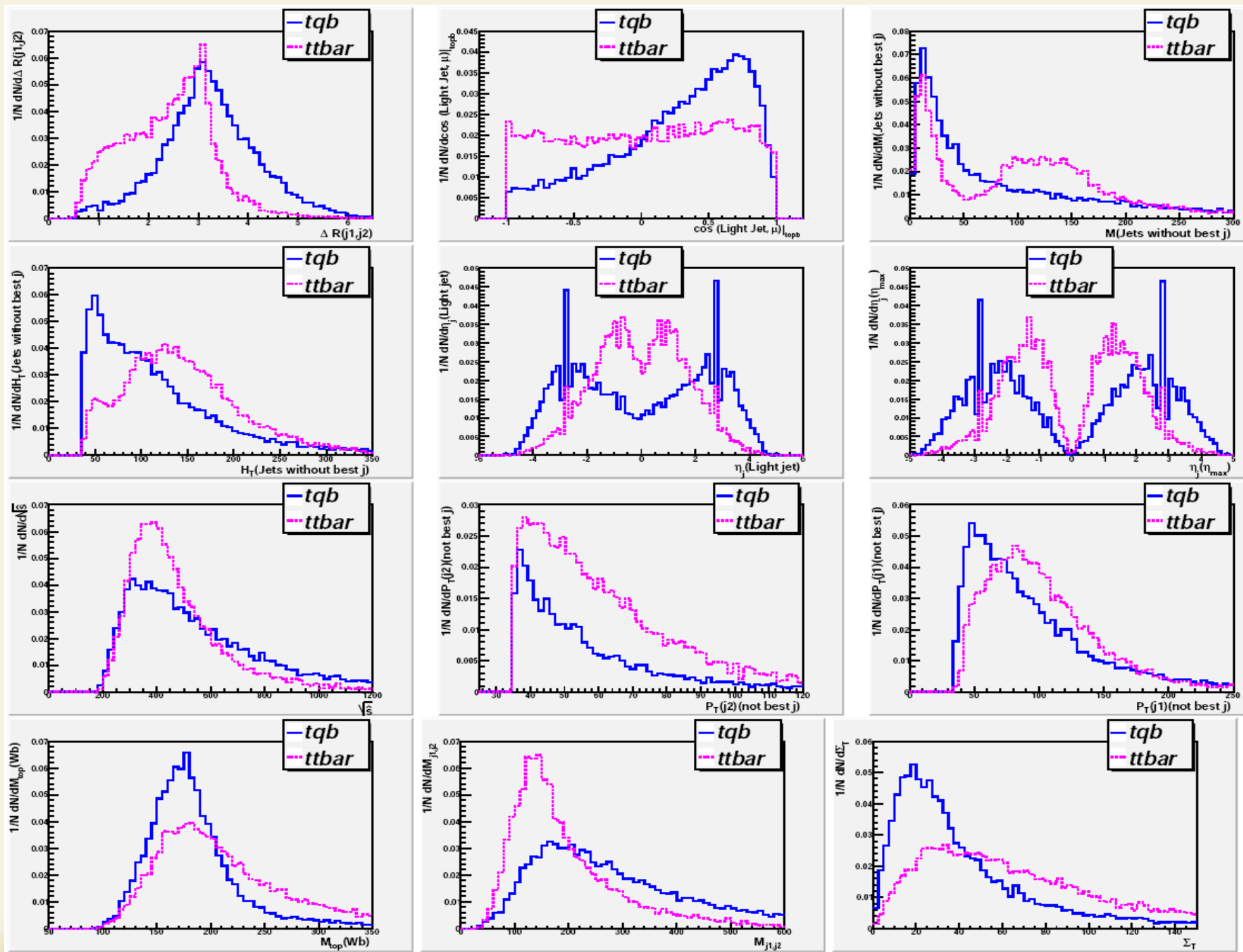


Реализация простейшей NN для фитирования простой нелинейной функции

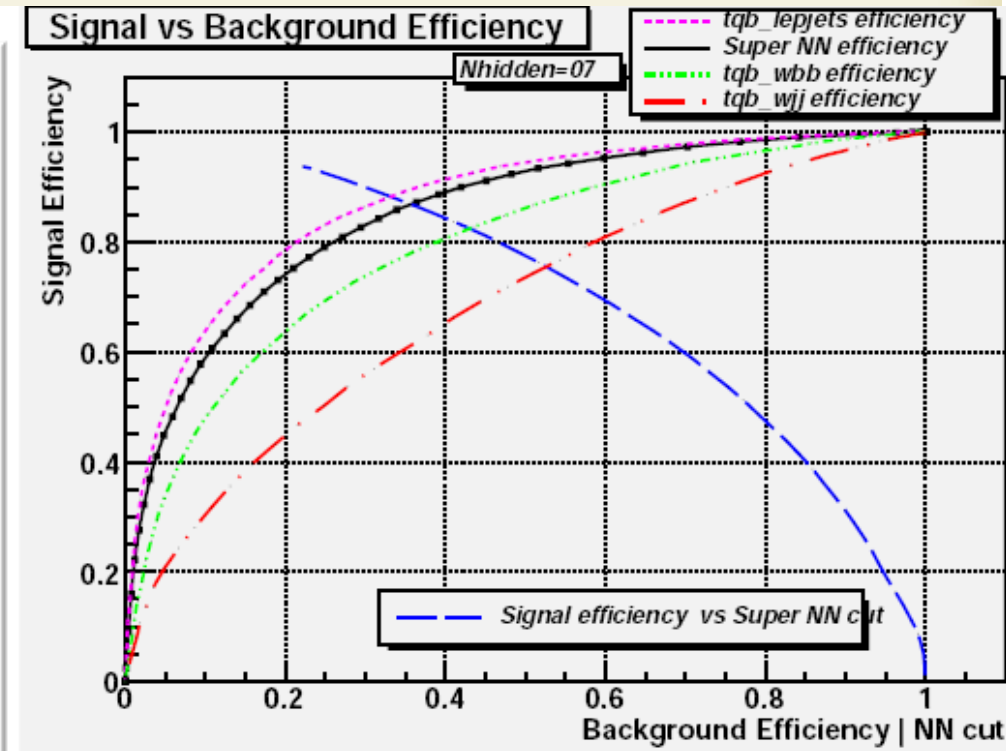
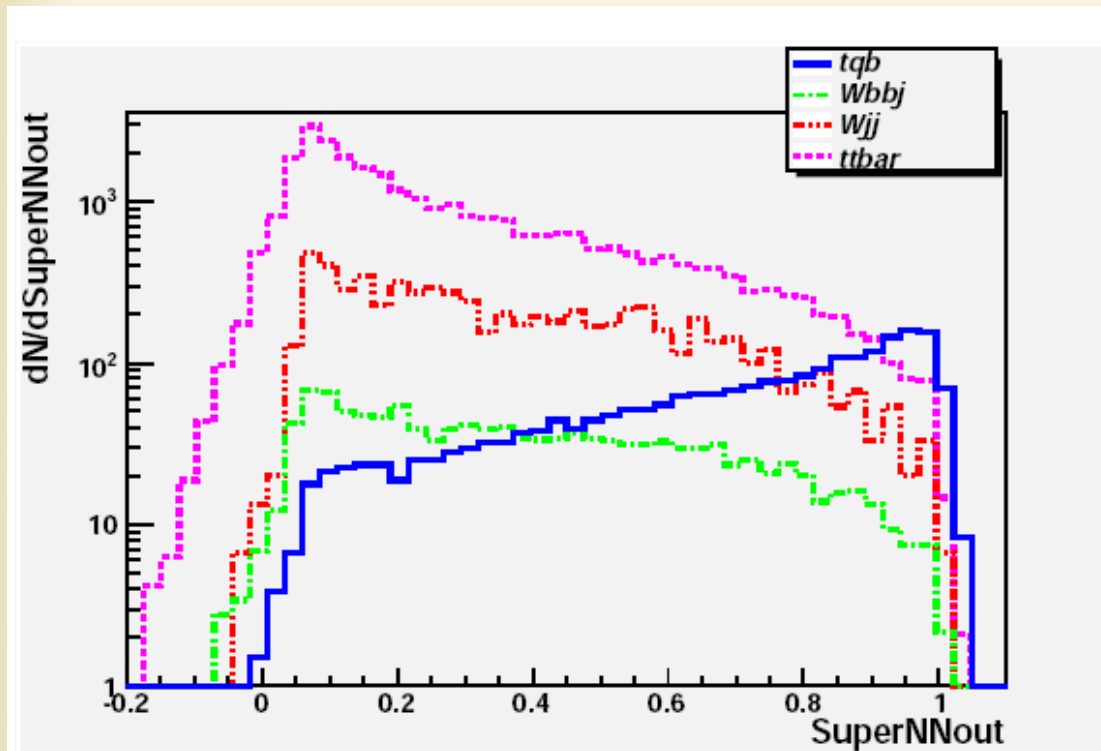
ReLU
 $\max(0, x)$



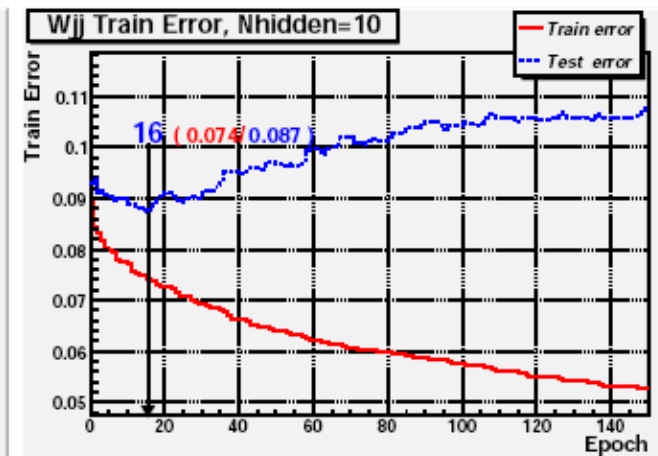
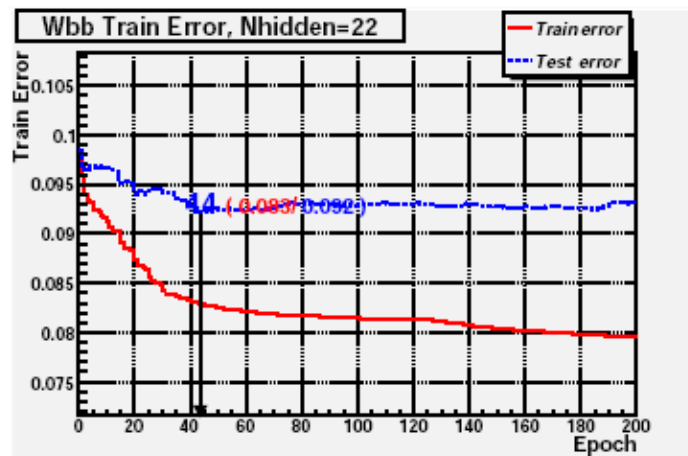
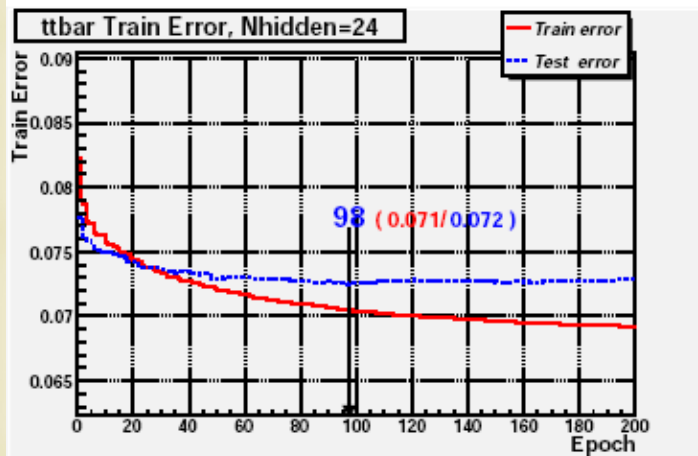
Постановка задачи классификации и регрессии в НЕР



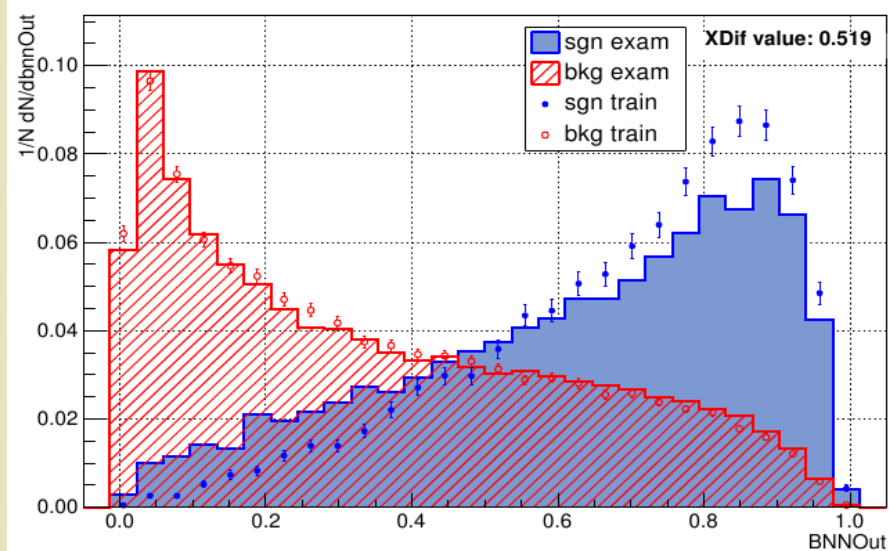
Критерии эффективности NN



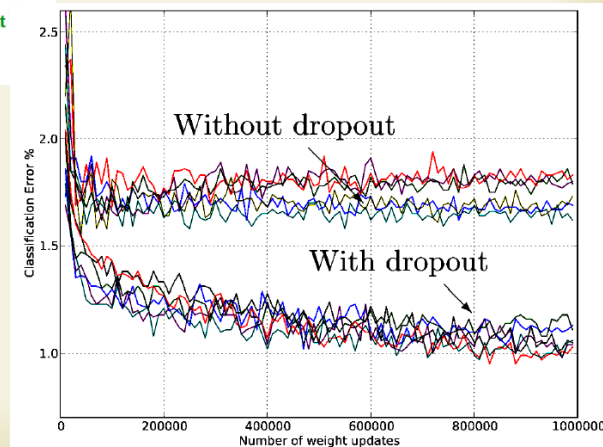
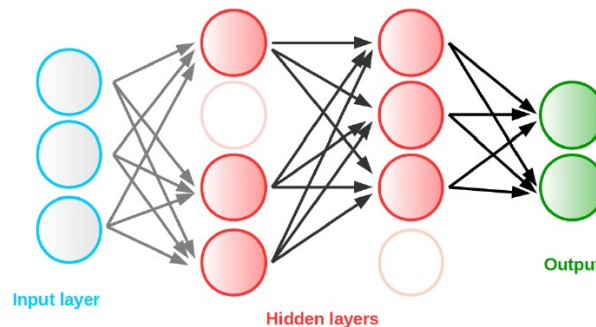
Проблема перетренировки (overfitting)



SM vs tcg FCNC couplings, bnn_tcg

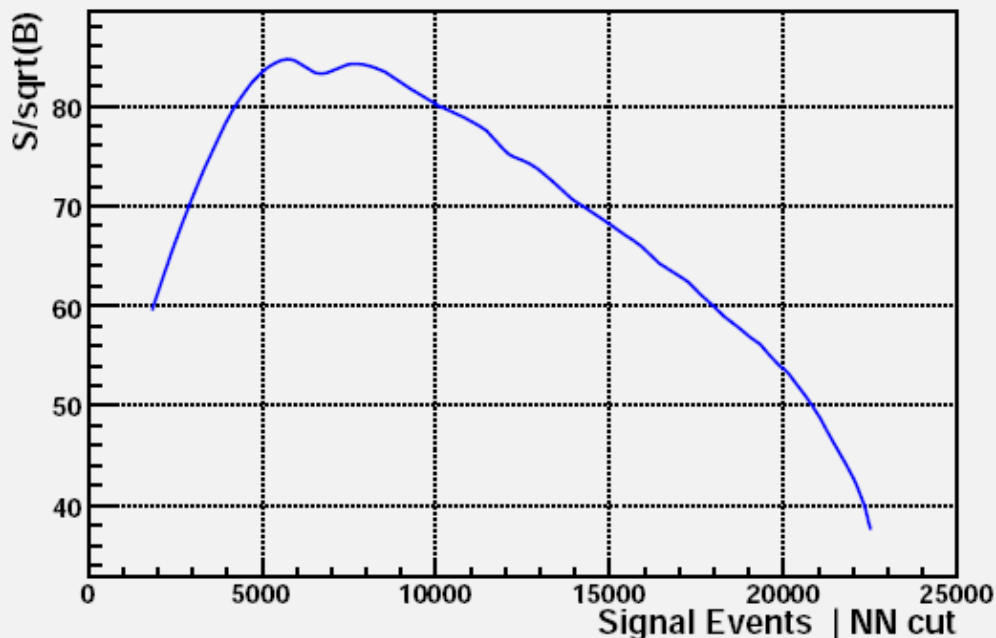


WITH DROPOUT (25%)

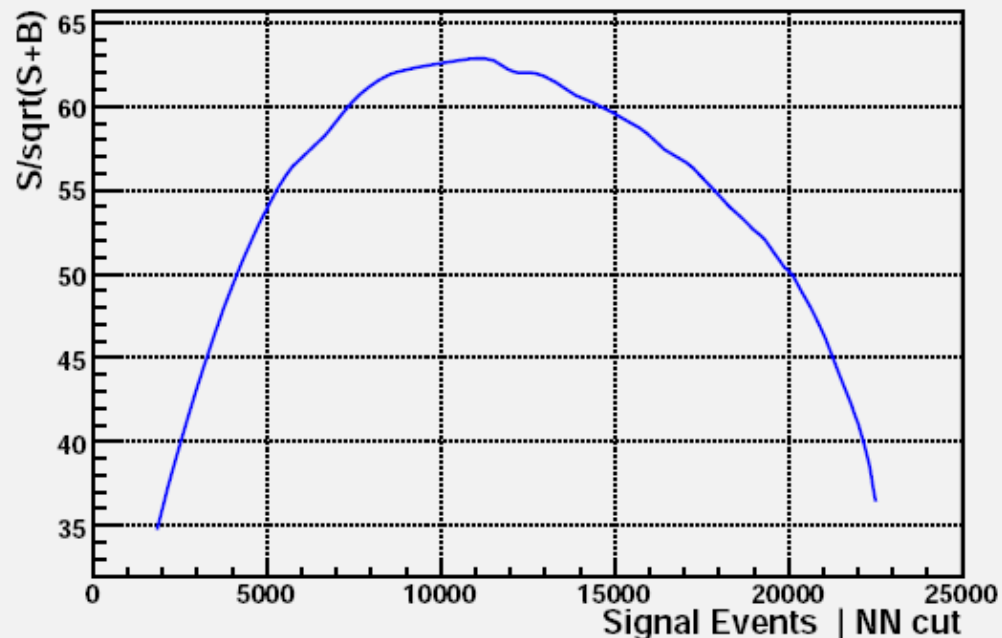


Использование выхода NN, счетный эксперимент

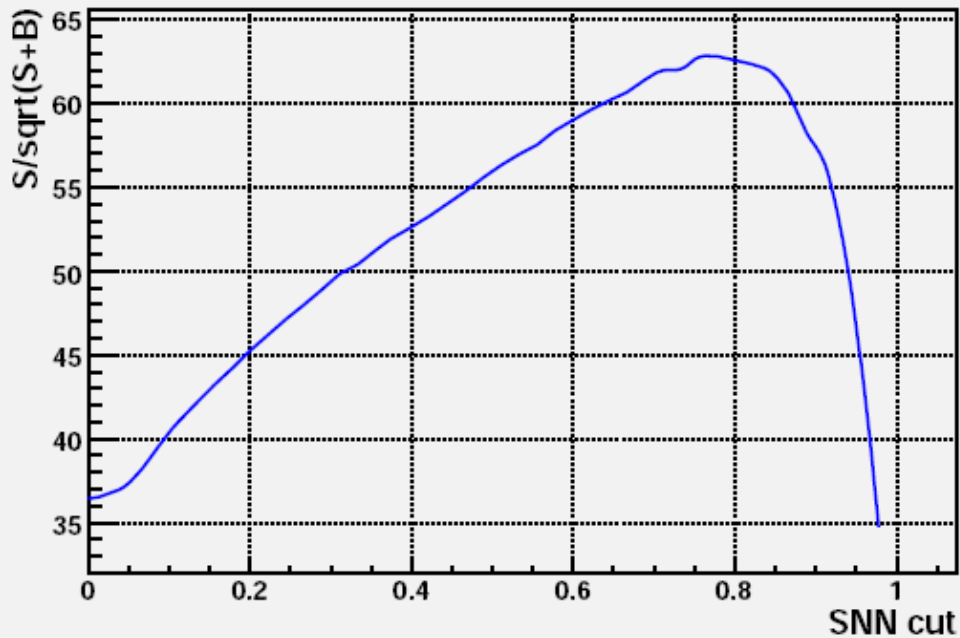
S/\sqrt{B} vs Signal Events



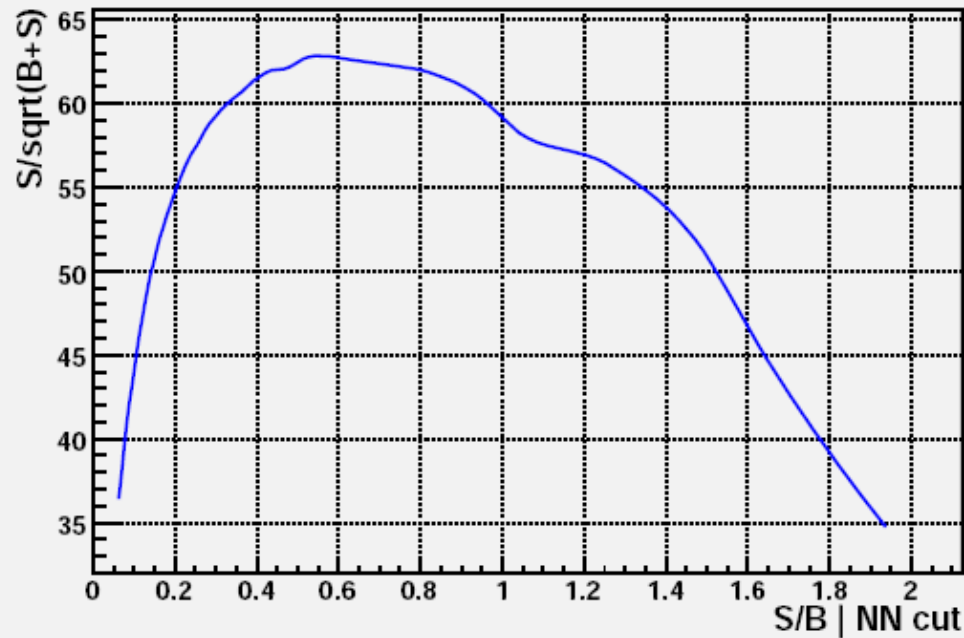
$S/\sqrt{S+B}$ vs Signal Efficiency



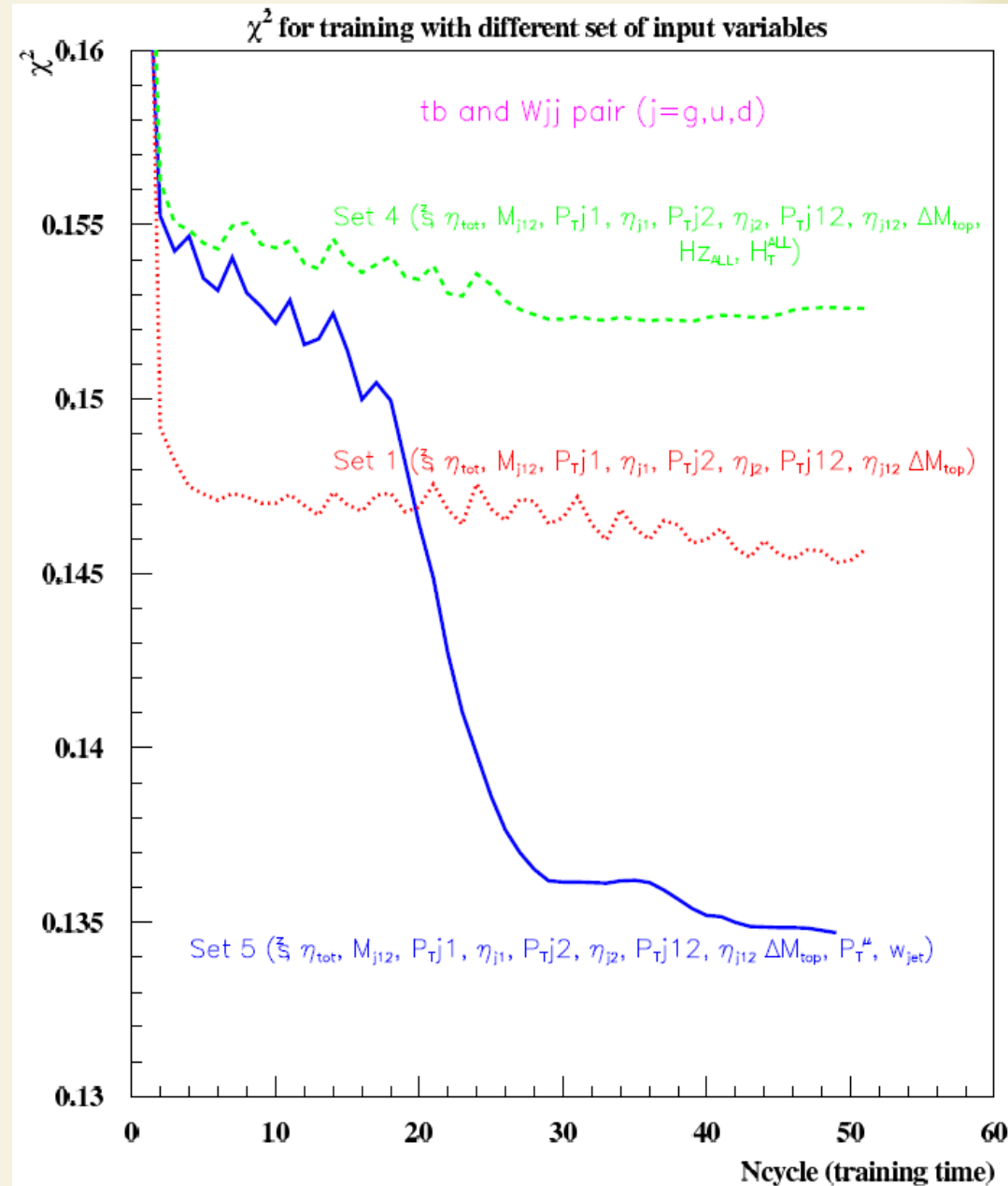
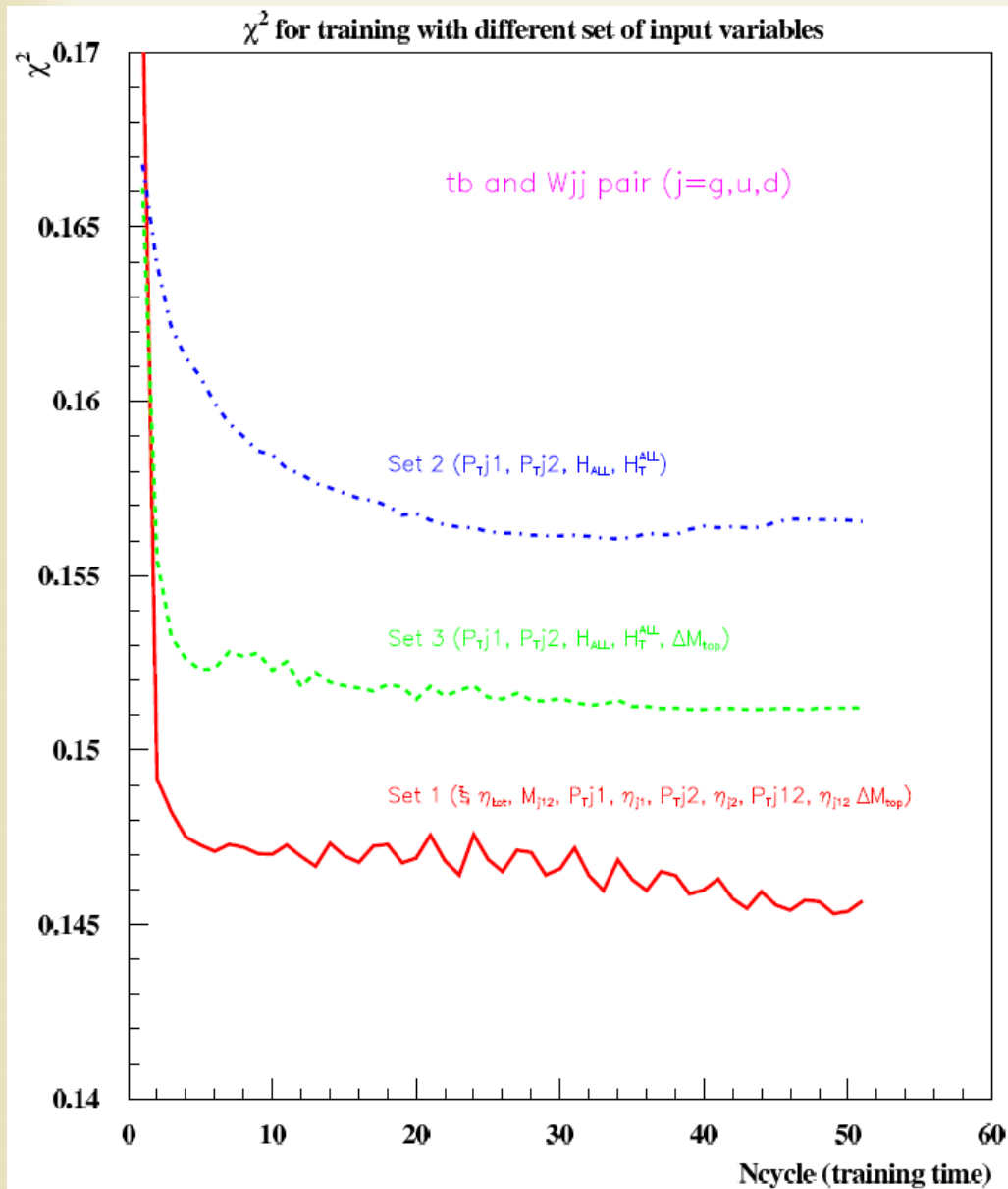
$S/\sqrt{S+B}$ vs SNN cut value



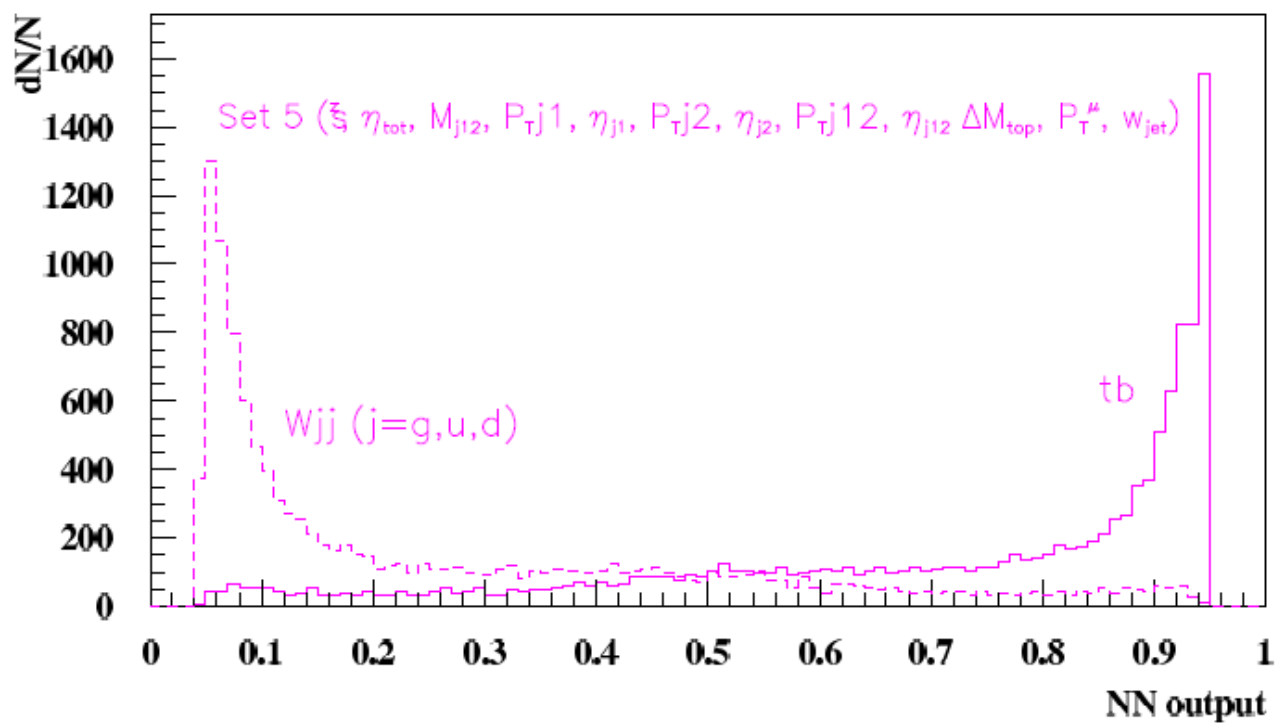
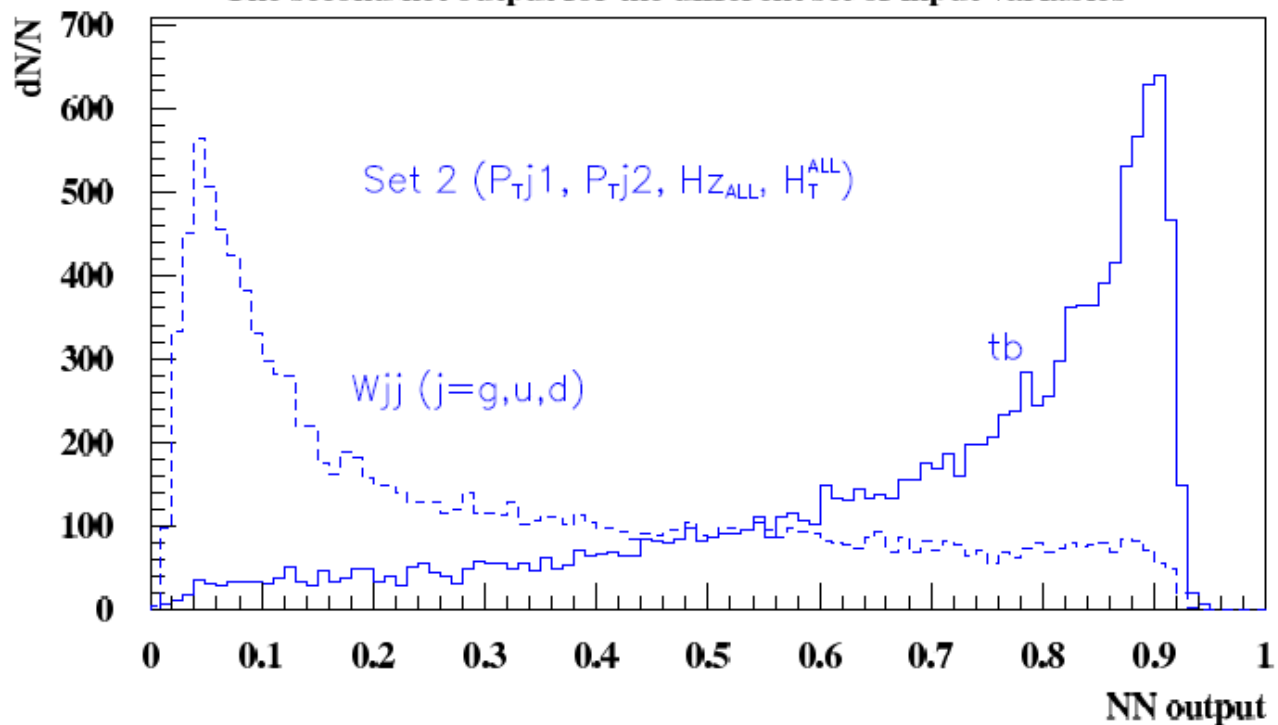
$S/\sqrt{B+S}$ vs S/B



Наборы входных переменных



The second net output for the different set of input variables



Method of “optimal observables” to find high level observables

- **Developed for single top analysis in D0 (1998). Provides general recipe how to choose most sensitive high level variables to separate signal and background by NN**
 - 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: s- and t-channel singularities), **Angular** variables (numerators of FD) and **Threshold** variables (Energy thresholds of the processes, H_T , ...)
 - 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 single top and Higgs searches

- D0-Note-3612 (1999) “Singularities of Feynman Diagrams and Optimal Kinematic Variables for Neural Networks.”
- E.Boos, V.Bunichev, L.Dudko, A.Markina, M.Perfilov **Phys.Atom.Nucl. 71 (2008) 388-393**
- E.Boos, L.Dudko, T.Ohl Eur.Phys.J. C11 (1999) 473-484
- E.Boos, L.Dudko Nucl.Instrum.Meth. A502 (2003) 486-488
- **Applied in different experimental analysis in D0 and CMS**
 - Phys.Lett. B517 (2001) 282-294 and some other single top D0 publications
 - JHEP02(2017)028 (CMS-TOP-14-007)

1) Увеличивая размер сети (количество входов, узлов, слоев) мы увеличиваем количество свободных параметров (весов) и усложняем тренировку.

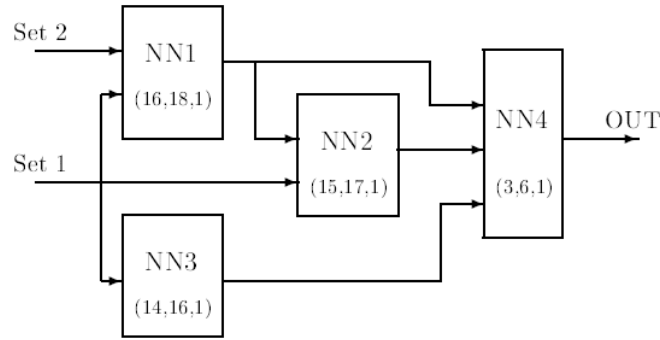
2) Входная информация для сети должна отражать полный набор отличительных признаков, но исключать избыточную информацию усложняющую тренировку

3) наиболее эффективная полносвязная сеть — полный набор однородных признаков на входе (нормализация, логарифмическая трансформация) и минимально-достаточное количество узлов в архитектуре.

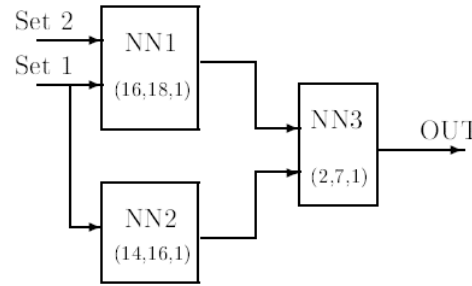
Поиск одиночного топ кварка в эксперименте D0 (Tevatron), Run I

hep-ex/9907041; ACAT 2000, AIP Conf.Proc. 583 (2001) 1, 83-85; Phys.Lett.B 517 (2001) 282-294

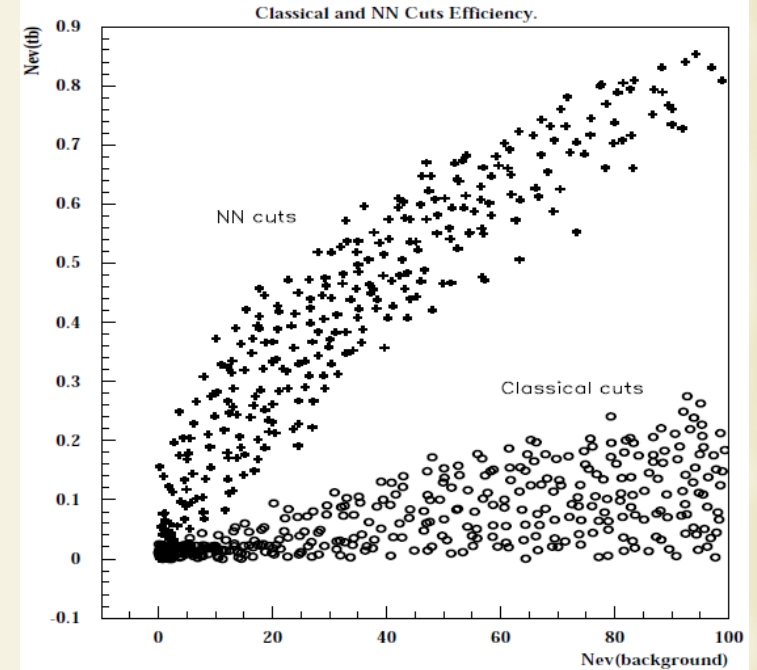
Wjj and Signal Network
(j=g,u,d)



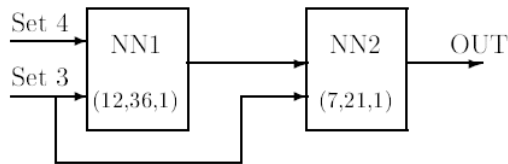
Wjj and Signal Network
(j=c,s,b)



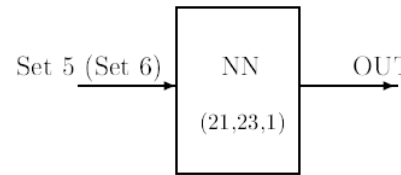
Set 1: $M_{j1,j2}, M_t, \hat{s}, Y_{tot}, P_T^{j1}, y_{j1}, P_T^{j2}, y_{j2}, P_T^{j12}, Y_{j12}, M_T^{j1,j2}, A, w_{jet1}, w_{jet2}$
 Set 2: $P_T^{tag \mu^1}, P_T^{tag \mu^2}$



$t\bar{t}$ and Signal network



QCD(WW) and Signal network

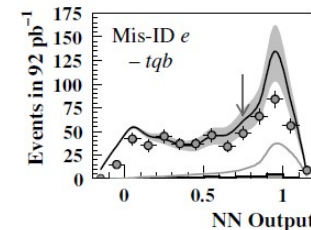
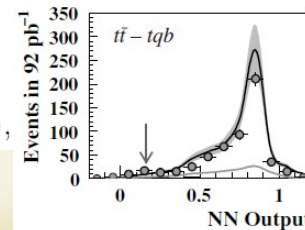
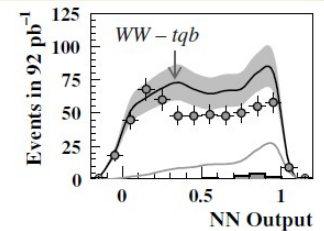
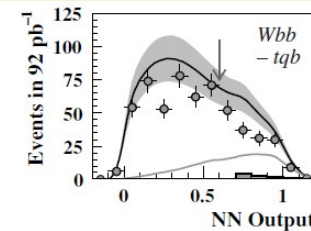
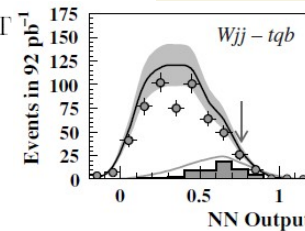


Set 3: $M_{jALL}, \cancel{E}_T, \hat{s}, H_T(j_{all} - j_{best}), M(j_{all} - j_{best}), H(j_{all} - j_{best})$

Set 4: $P_{Tj2}, P_{Tj3}, A, P_T(j_{all} - j_{best}), n_{j2}(0 \text{ or } 1), n_{j3}(0 \text{ or } 1)$

Set 5: (Set 1), (Set 2), $\Delta P_T(W, j_{ALL})$

Set 6: $P_T^W, M_T^{j1,j2}, H_T^{j1,j2}, P_T^{tag \mu}, P_T^{j1}, P_T^{j2}, P_T^{j12}, Y_{j12}, \Delta M(m_W, m_{j1,j2}),$



- D0 Data (Initial Cuts)
- Signal + Bkgds (Initial Cuts)
- 50 × tqb Signal (Initial Cuts)
- Cut on NN Output
- D0 Data (NN Cuts)

$$O_{NN}^{comb} = \frac{3}{\frac{1}{O_{NN1}} + \frac{1}{O_{NN2}} + \frac{1}{O_{NN3}}}$$

Optimized Event Analysis

Input:

discriminating variables

Method:

multivariate analysis

Output:

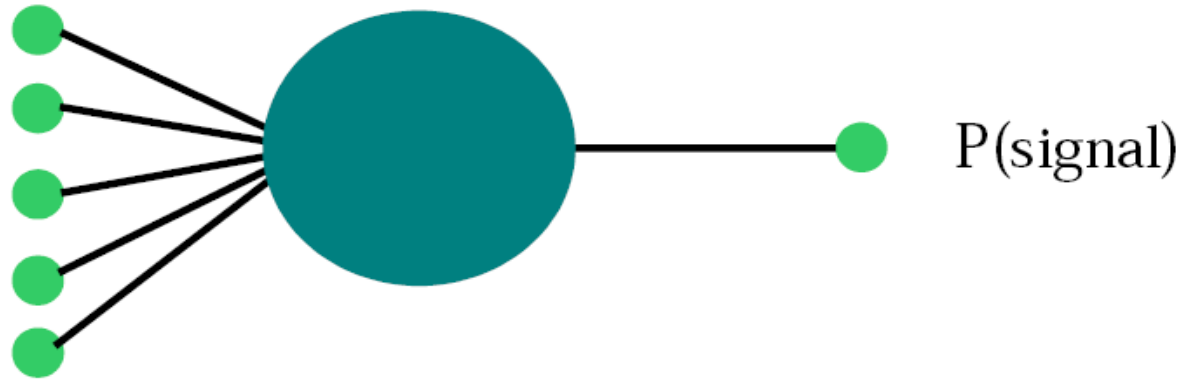
signal probability

Singular variables

Angular variables

Threshold variables

.....



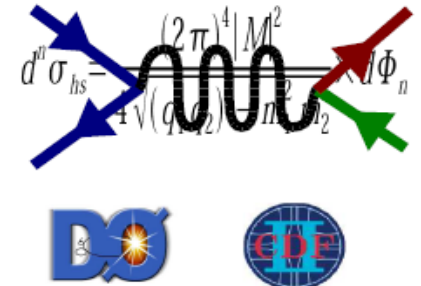
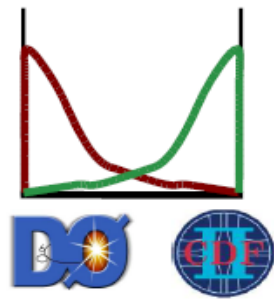
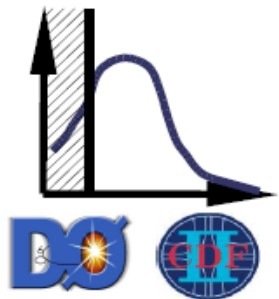
Cut-Based

Likelihoods

Decision Trees

Neural Networks

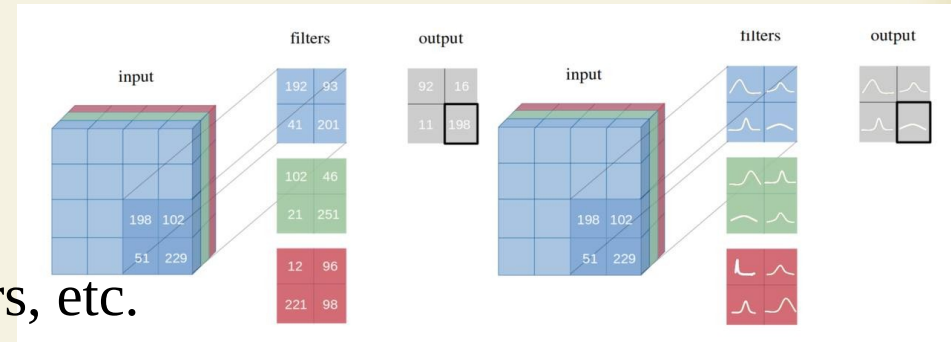
Matrix Elements



Feed-forward NNs and Bayesian NNs

- **Optimal FF NN requires manual tuning**

- Need to avoid over-fitting problem
- Tune architecture of NN, training parameters, etc.
- It is not very stable relative to the tuning



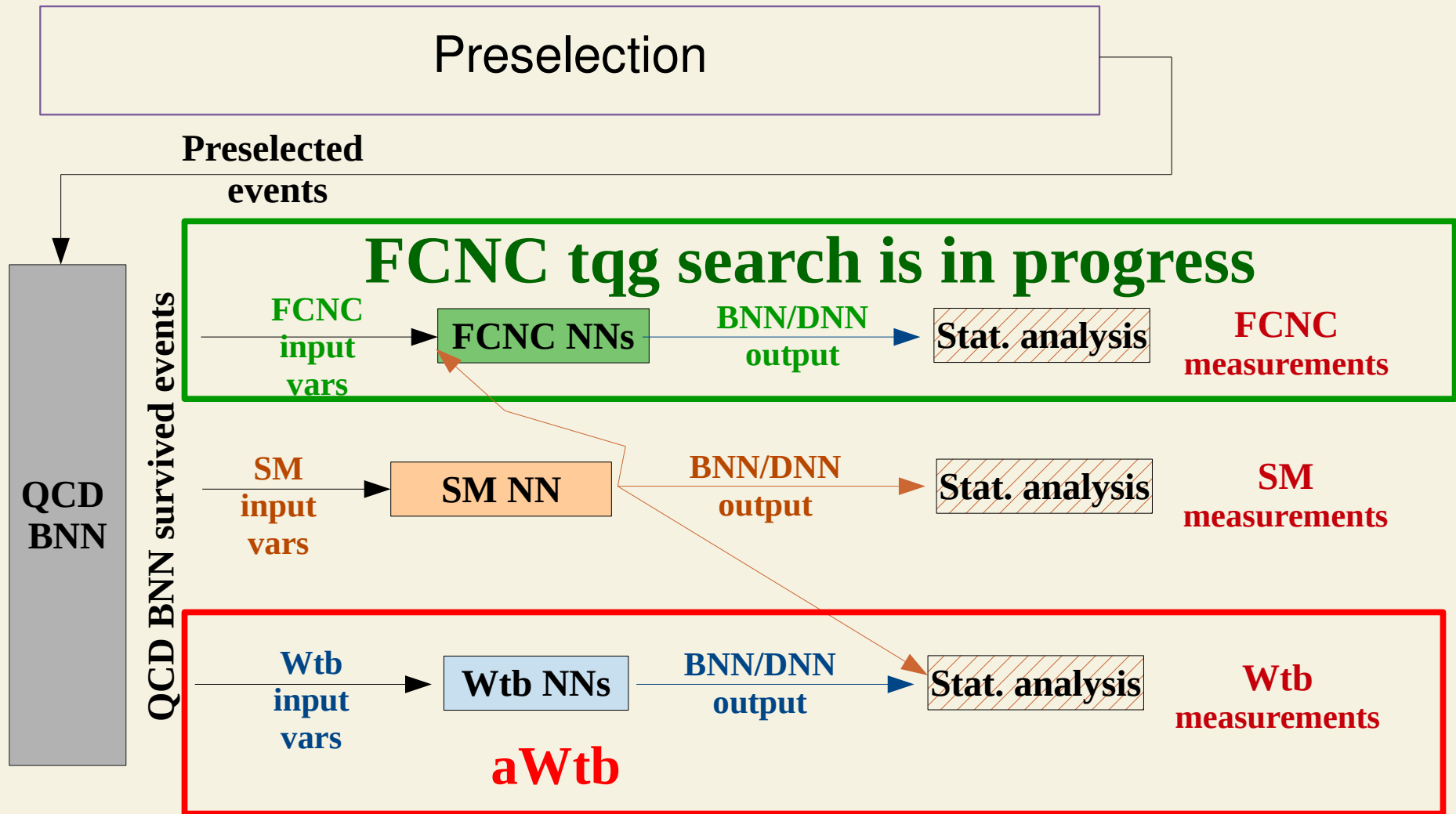
- **R. M. Neal proposed Bayesian NN approach where each NN internal weight is not a number but a distribution**

- "Bayesian Learning for Neural Networks"
<http://www.cs.utoronto.ca/~radford/bnn.book.html>
- FBM package is the software realization for this idea
<http://www.cs.toronto.edu/~radford/fbm.software.html>
- First use in HEP: P. Bhat and H. Prosper, "Bayesian neural networks", Conf. Proc. C050912 (2005) 151.

- **BNN provides the same level of Eff. as FF NN but it is very stable to the modifications of tuning parameters, architecture and practically does not affected by over-fitting problem**

- Does not require manual tuning. Require significantly more CPU resources

Схема реального анализа на LHC



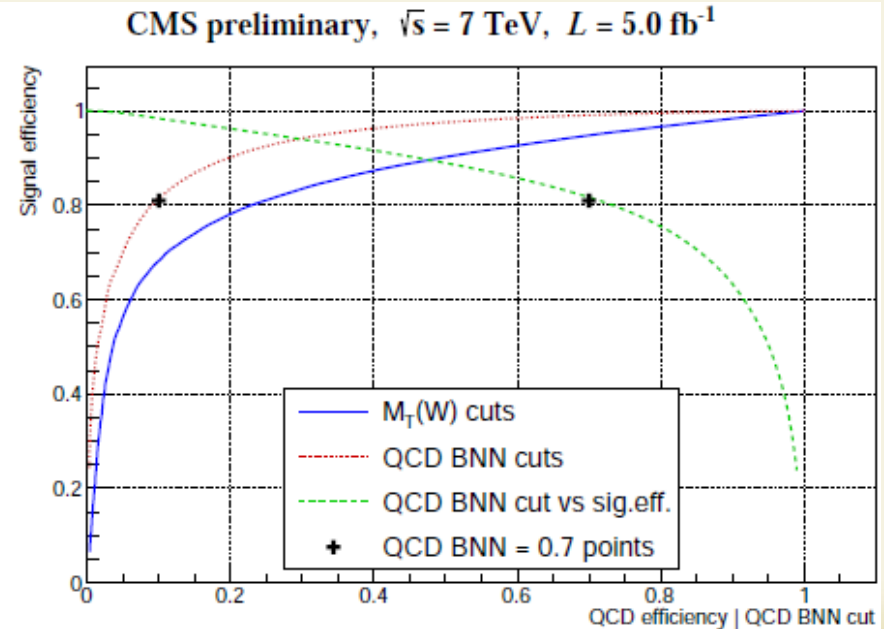
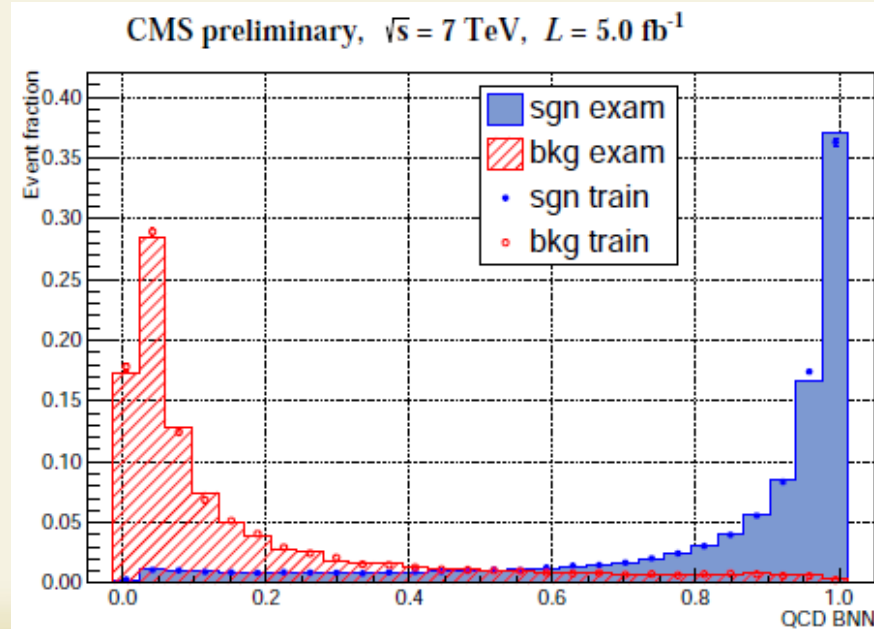
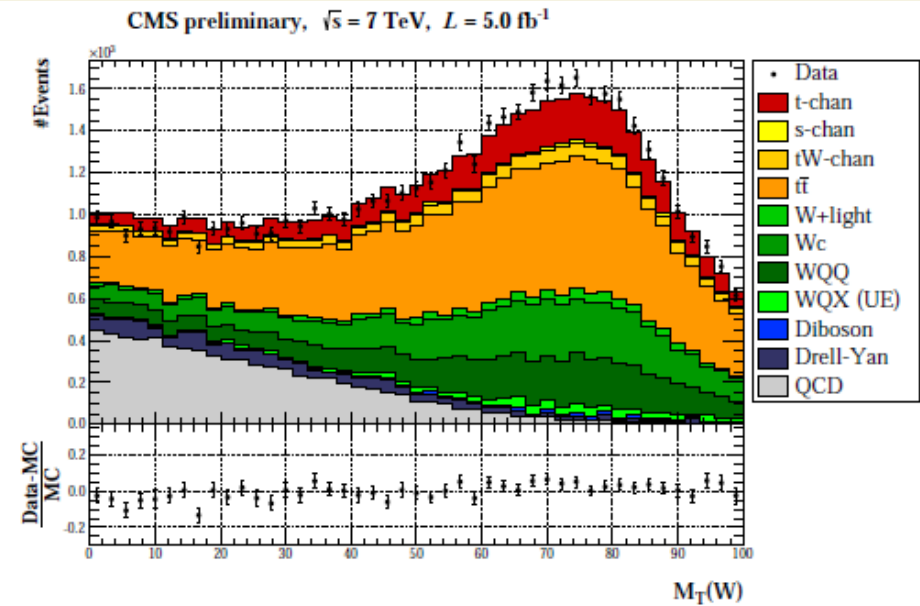
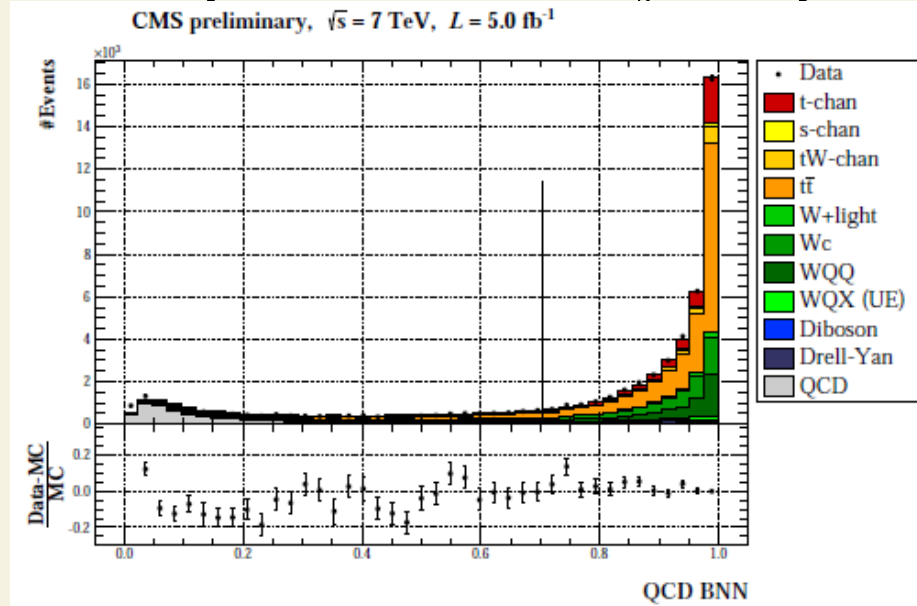
Published results:

- 1) 7&8 TeV **JHEP 02(2017)028** (FCNC tqq & aWtb)
- 2) 14 TeV HL-LHC YR, PAS-FTR-18-004; extrapolation to HE-LHC (FCNC tqq)
- 3) FCC 100 TeV, CDR vol.1 (FCNC tqq)

SM measurements: t-channel total cross section, fiducial cross section, differential cross sections, V_{tb} , $R(t/\bar{t})$ measurements

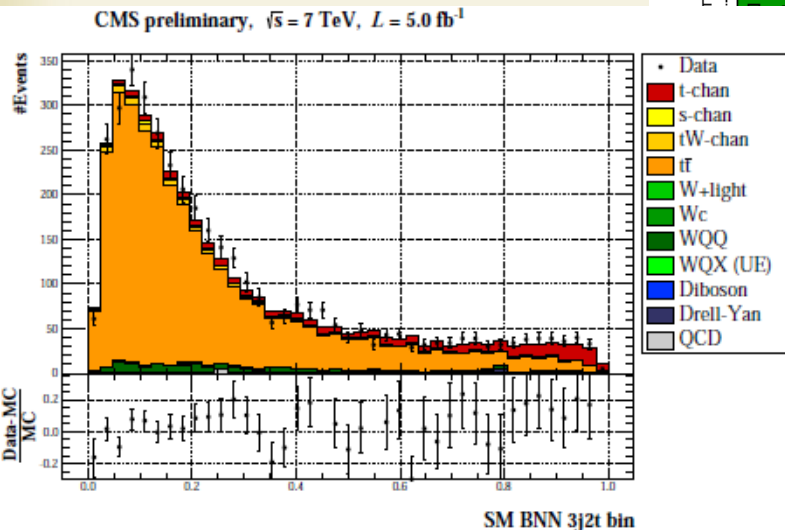
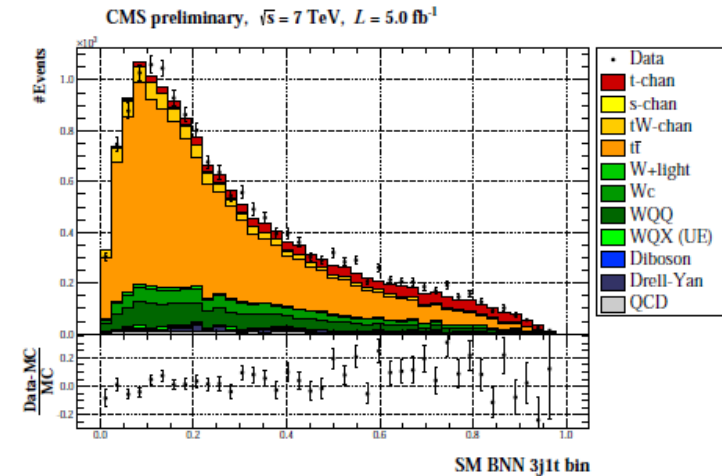
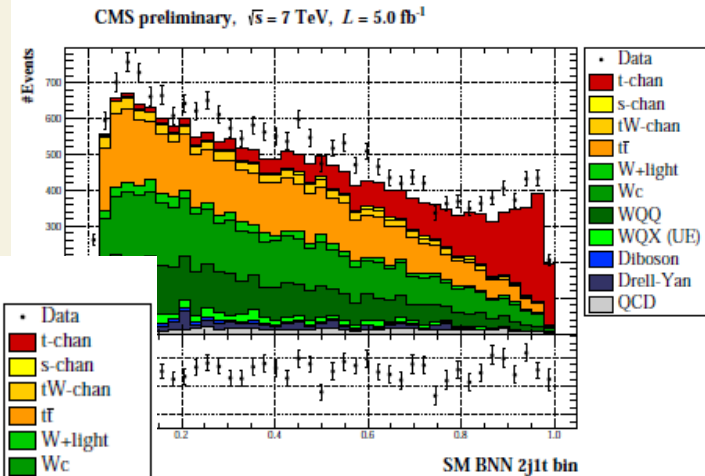
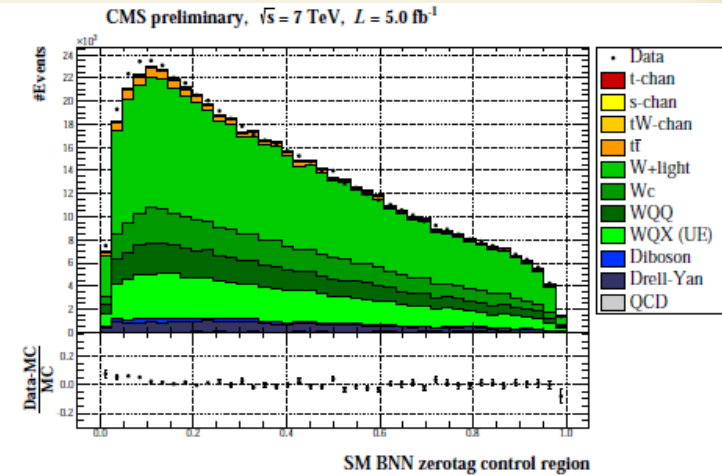
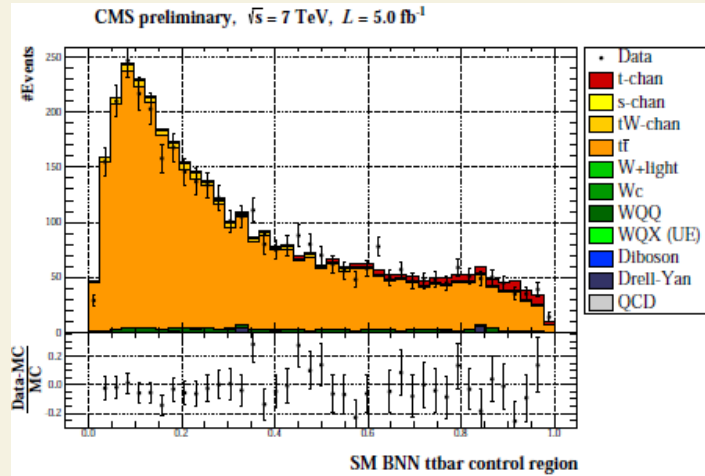
QCD BNN

Use 4 simple variables to minimize sys. uncert. Separation Eff. is significantly better than with one most efficient variable. Cut QCD BNN at 0.7 and use it as a filter. Survived events are passed to next analysis steps.



Cross checks of BNN

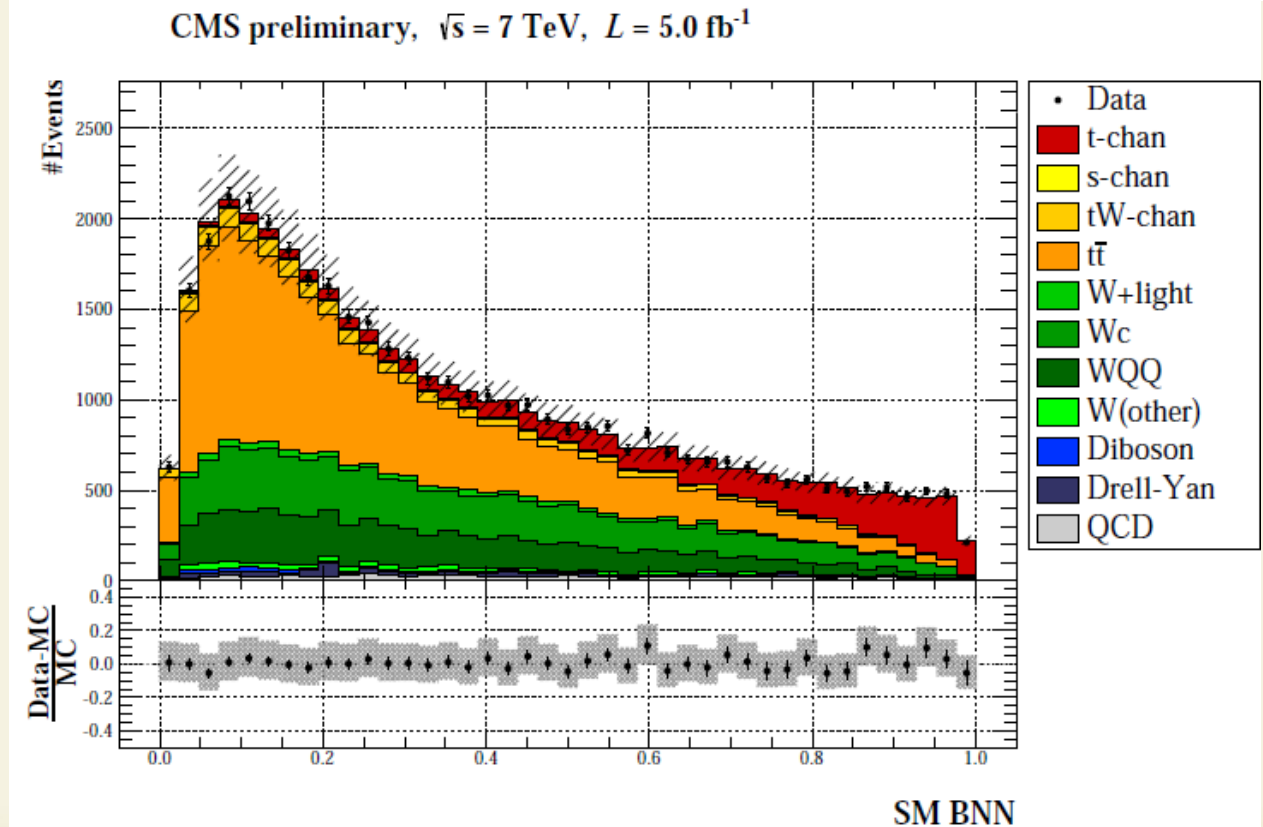
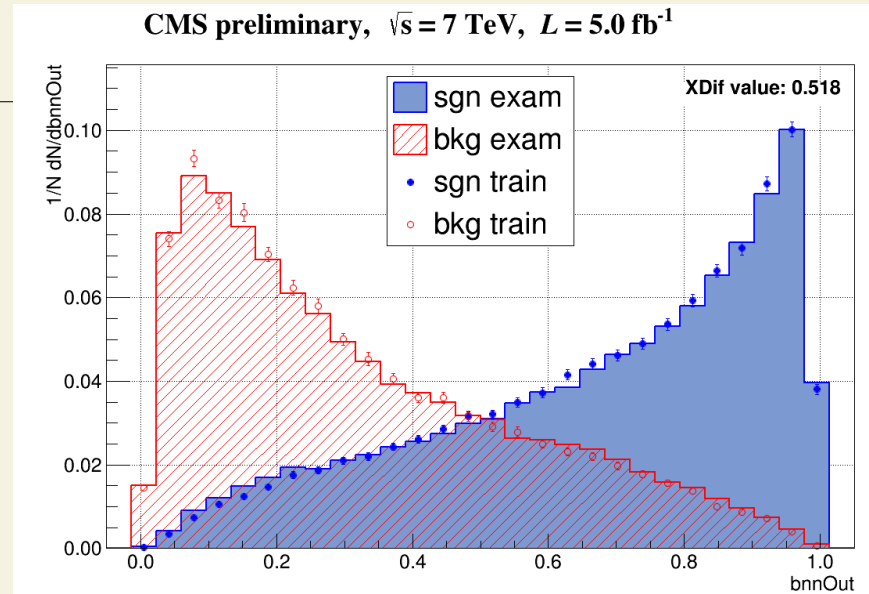
One of the main step of the analysis is to check the BNN with orthogonal event samples, in different regions of phase space and compare model response with real data if it is available



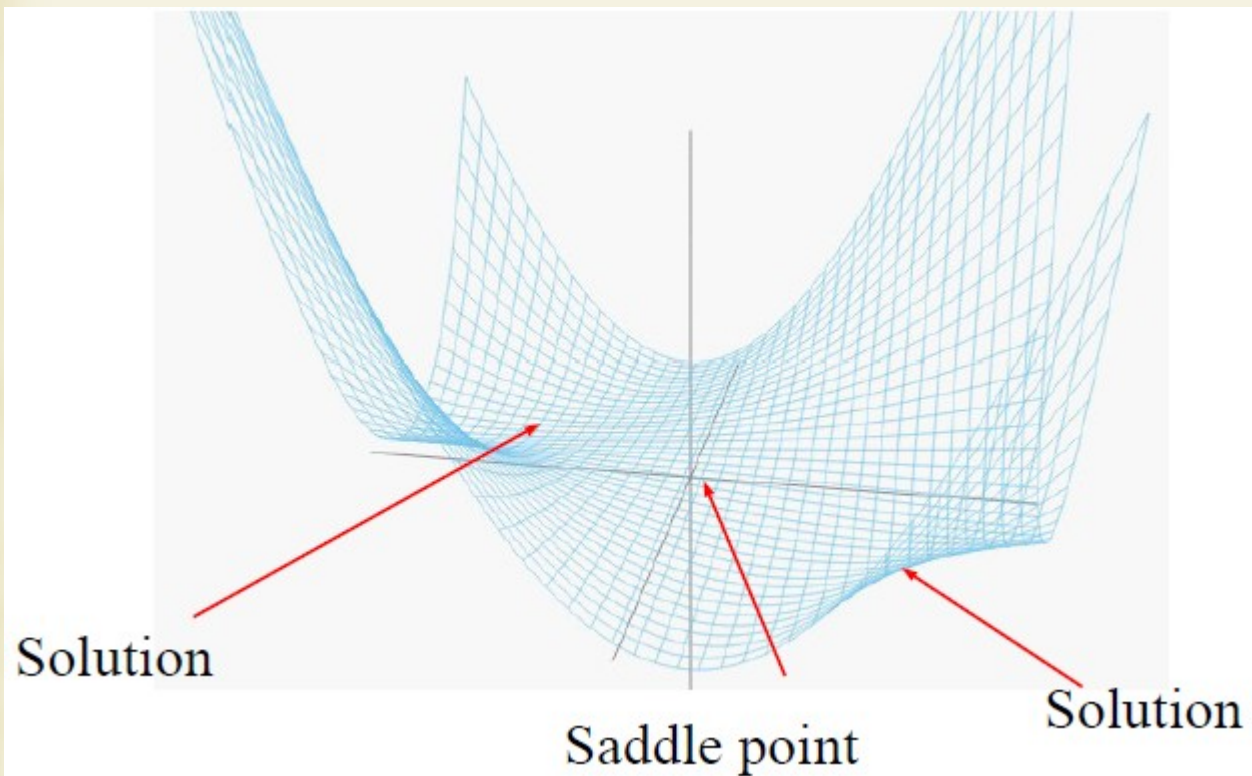
Final BNN classifier

After the set of cross checks and evaluation of the uncertainties the final classifier is ready for the statistical analysis to measure necessary physics parameter based on the shape of the BNN output.

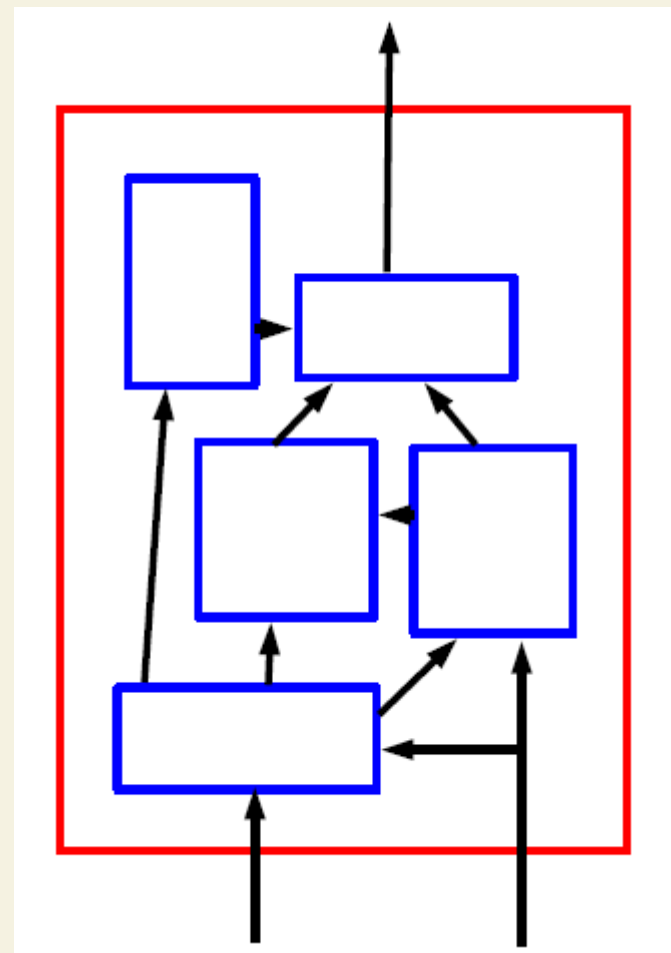
This real example includes optimization of the S/B separation Eff. and minimization of all types of uncertainties (syst-normalisation, syst-shape, syst-shape-umarginalized, stat).



Глубокие нейронные сети (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.



Novel approach with deep learning neural networks (DNN)

- ~ Starting from: 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 Deep NNs (many layers, neurons) is the possibility to analyze raw, not preprocessed, information.
- ~ One of the first examples in HEP, DNN increases possible significance from 3.1σ up to 5.0σ in comparison with NN with high level variables:

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

$$gg \rightarrow H^0 \rightarrow W^\mp H^\pm \rightarrow W^\mp W^\pm h^0$$

Nature Commun. 5 (2014) 4308

Глубокие нейронные сети (Deep Learning Neural Networks, DNN)

Traditional Pattern Recognition: Fixed/Handcrafted Feature Extractor



Mainstream Modern Pattern Recognition: Unsupervised mid-level features

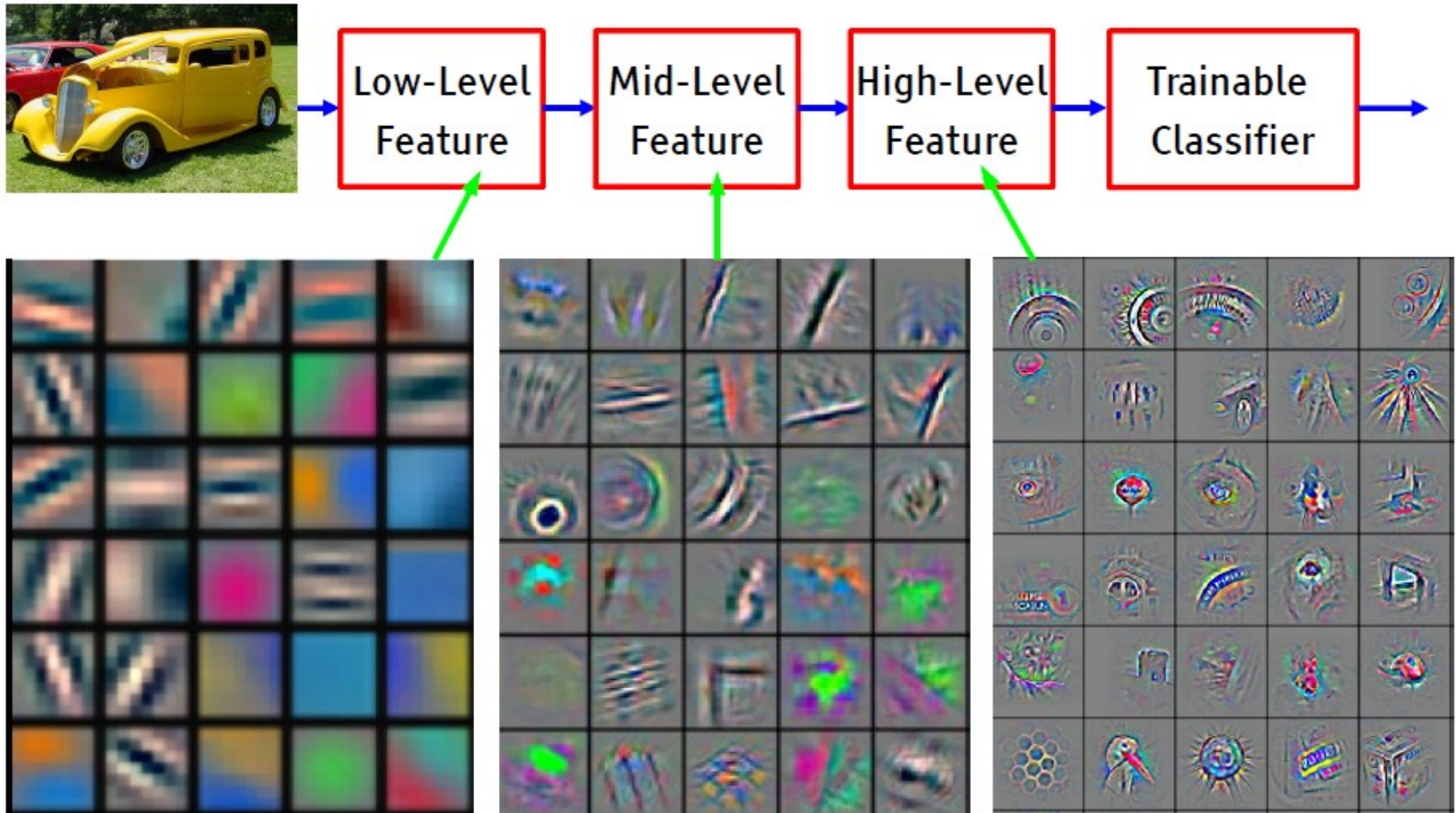


Deep Learning: Representations are hierarchical and trained



Глубокие нейронные сети (Deep Learning Neural Networks, DNN)

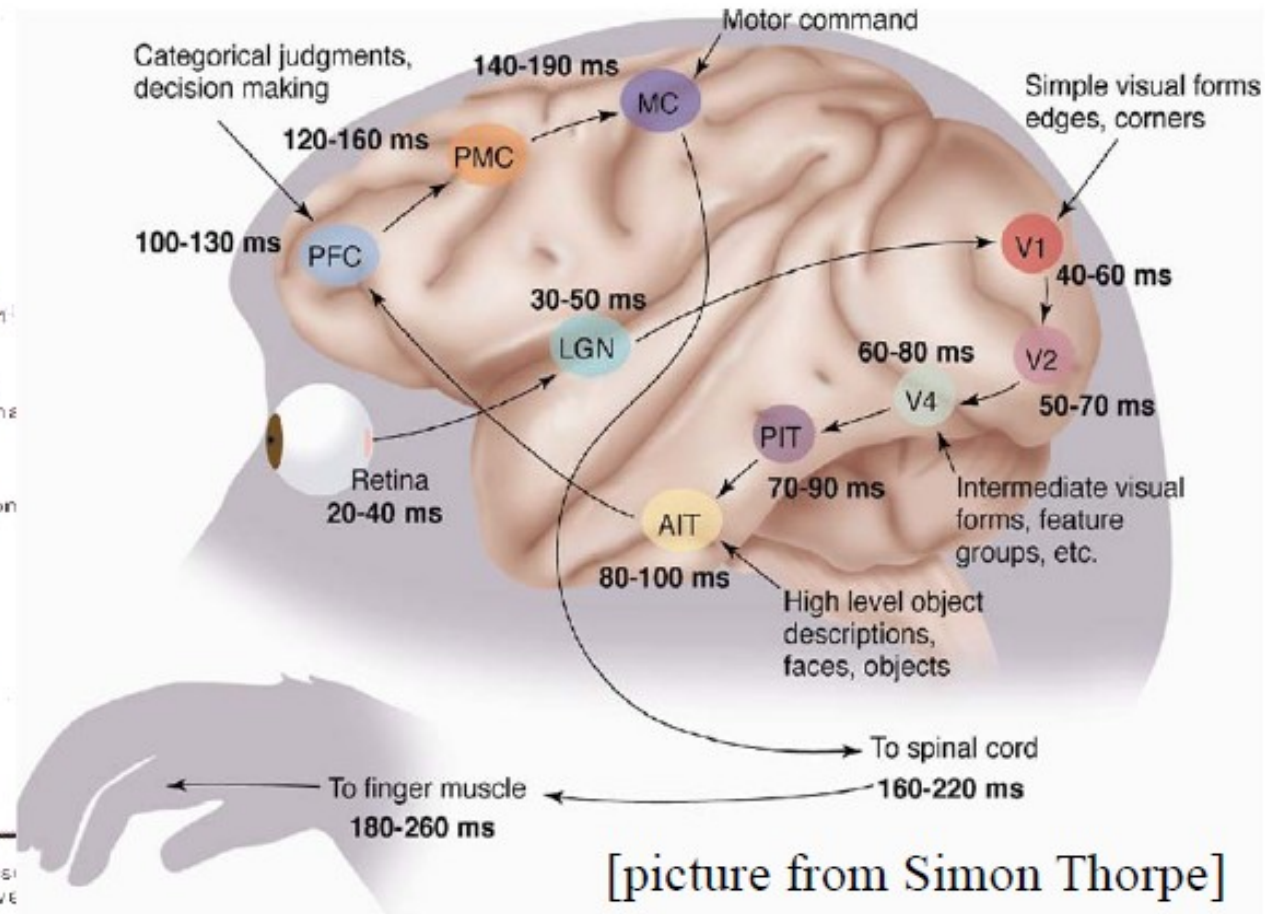
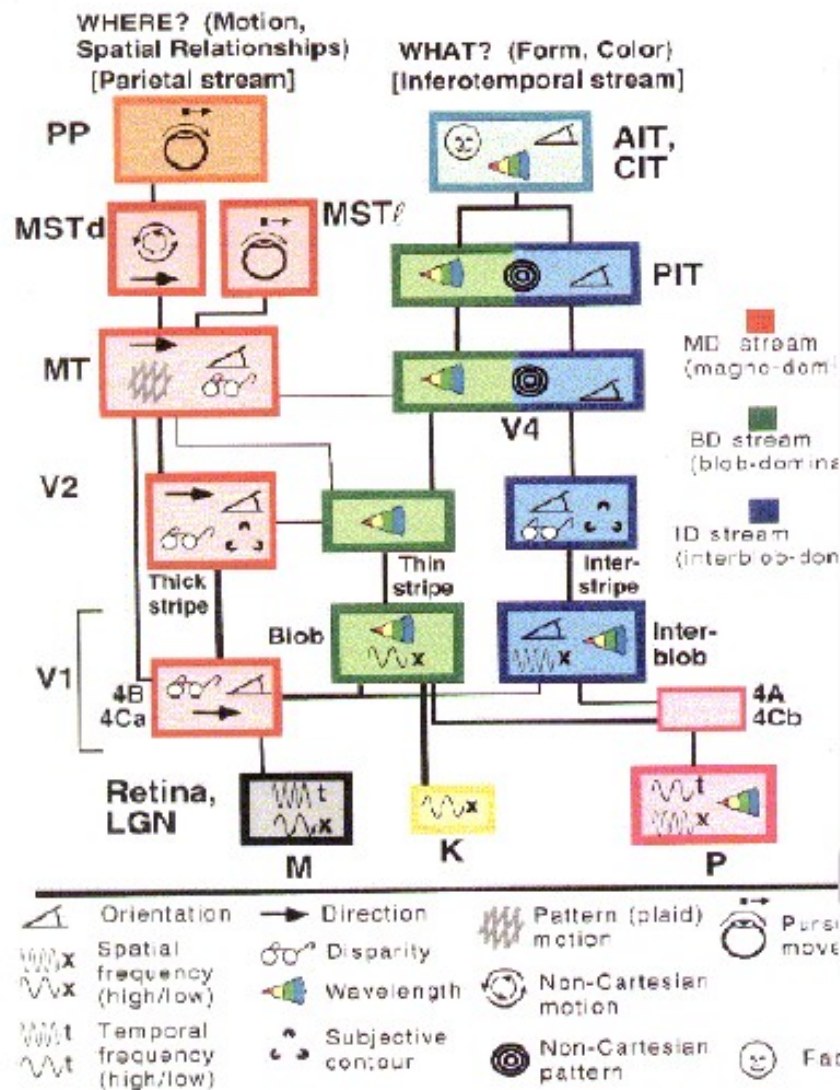
It's **deep** if it has **more than one stage** of non-linear feature transformation



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Глубокие нейронные сети (Deep Learning Neural Networks, DNN)

- The ventral (recognition) pathway in the visual cortex has multiple stages
- Retina - LGN - V1 - V2 - V4 - PIT - AIT



[picture from Simon Thorpe]

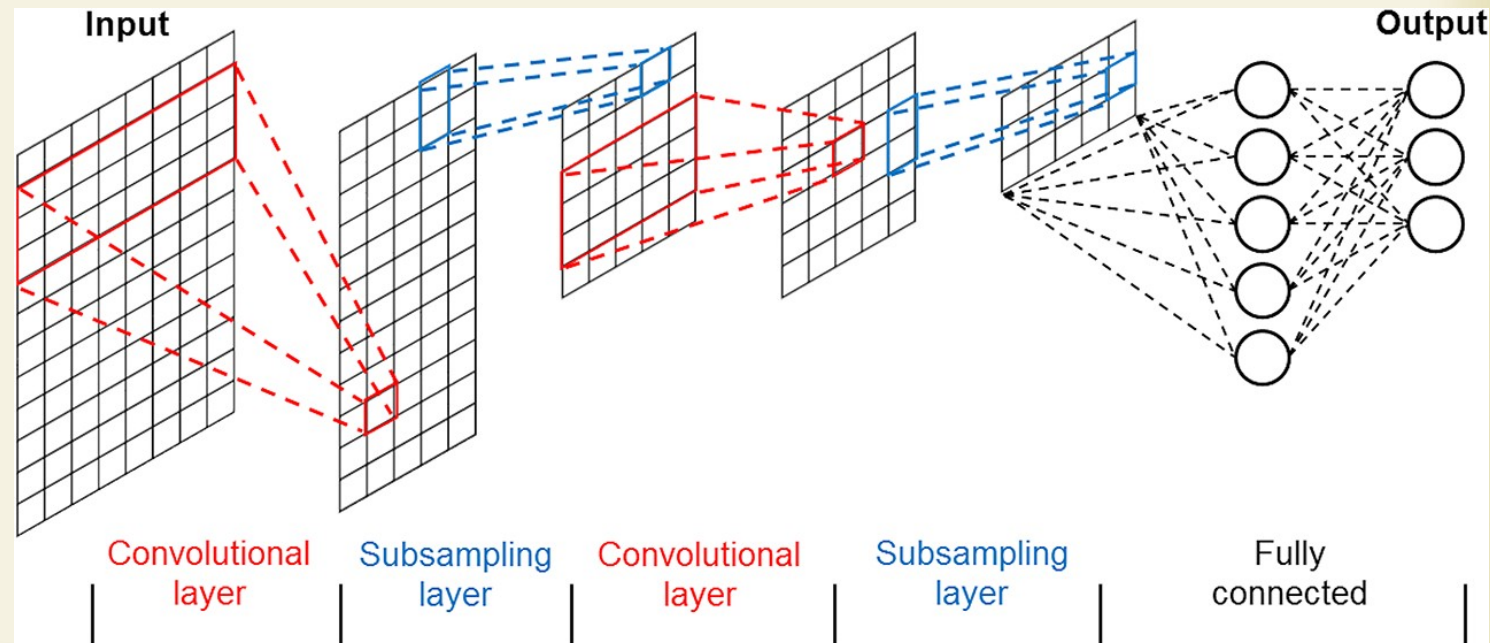
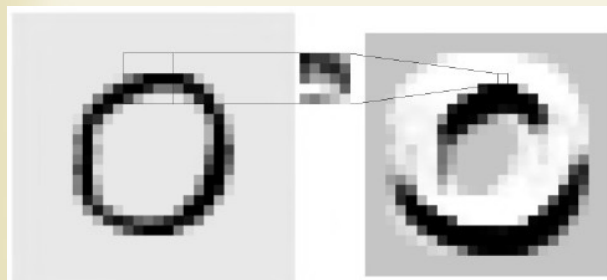
[Gallant & Van Essen]

Сверточные нейронные сети (Convolutional Neural Networks, CNN)

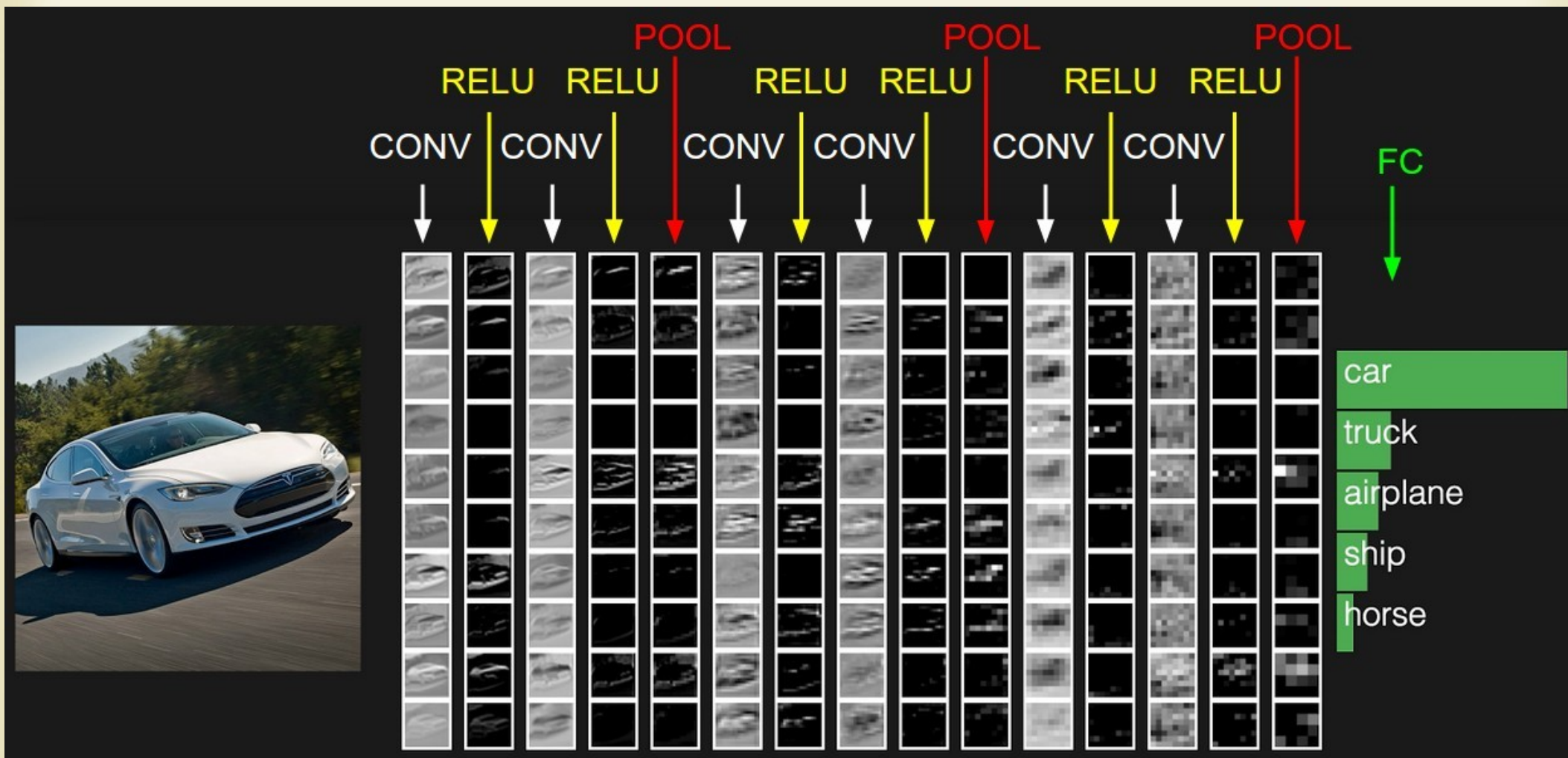
Использование внутренней структуры данных для снижения размерности сети. Можно выделить три характерных элемента CNN.

- 1) свертка - взвешенная сумма нескольких близких пикселей с последовательным сдвигом такого окошка из близких пикселей по всему изображению, т. е. свертка входного тензора с некоторым ядром свертки (convolutional layer). Например, пиксели в окошке 3×3 суммируются с некоторыми весами, далее окно сдвигается на 2 пикселя и повторяется суммирование, с теми-же весами.
- 2) подвыборка (pooling, subsampling layer), из получившегося тензора, обычно выбирается взятие максимума (или среднего) по значениям в некотором окне, далее окно сдвигается. Этапы свертки и подвыборки чередуются и выделяют карту признаков на изображении.
- 3) На заключительном этапе полносвязная сеть анализирует полученную карту признаков и выдает требуемый ответ.

LeCun et. al. NIPS 1989



Сверточные нейронные сети (Convolutional Neural Networks, CNN)



<http://cs231n.github.io/convolutional-networks/>

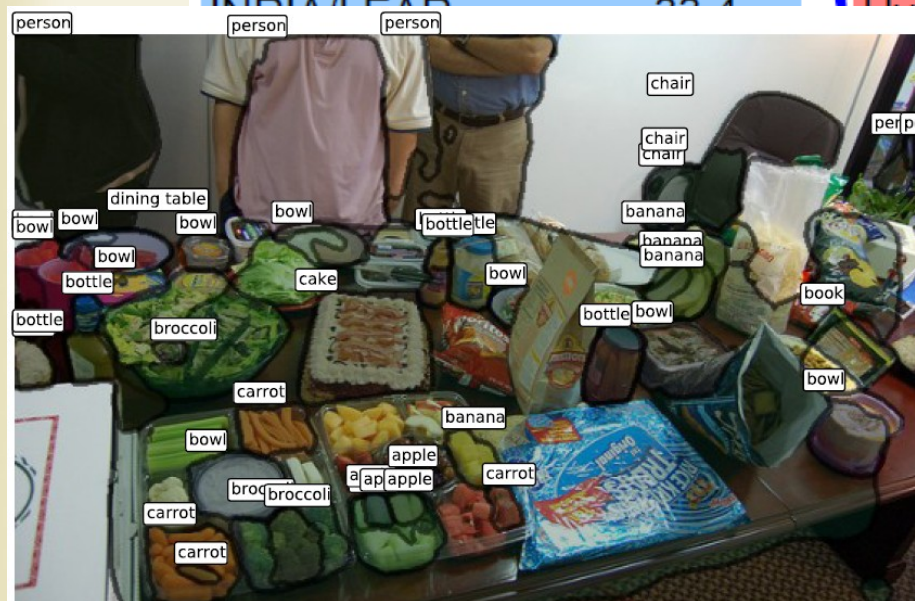
CNN: Изображение усредненного кота



Классификация объектов на фотографиях

- Give the name of the dominant object in the image
- Top-5 error rates: if correct class is not in top 5, count as error
 - Red: ConvNet, blue: no ConvNet

2012 Teams	%error	2013 Teams	%error	2014 Teams	%error
Supervision (Toronto)	15.3	Clarifai (NYU spinoff)	11.7	GoogLeNet	6.6
ISI (Tokyo)	26.1	NUS (singapore)	12.9	VGG (Oxford)	7.3
VGG (Oxford)	26.9	Zeiler-Fergus (NYU)	13.5	MSRA	8.0
XRCE/INRIA	27.0	A. Howard	13.5	A. Howard	8.1
UvA (Amsterdam)	29.6	OverFeat (NYU)	14.1	DeeperVision	9.5
INDIA/LEAP	33.4	UvA (Amsterdam)	14.2	NUS-BST	9.7
		Clarifai (NYU spinoff)	15.2	TTIC-ECP	10.2
		VGG (Oxford)	15.2	XYZ	11.2
		VGG (Oxford)	23.0	UvA	12.1



Сверточные нейронные сети

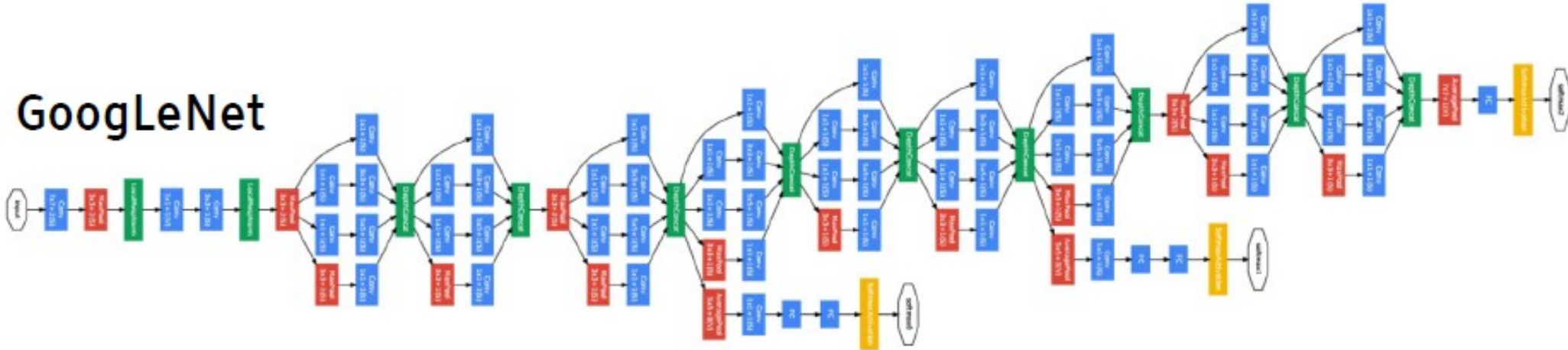
Рабочие архитектуры

Small kernels, not much subsampling (fractional subsampling).

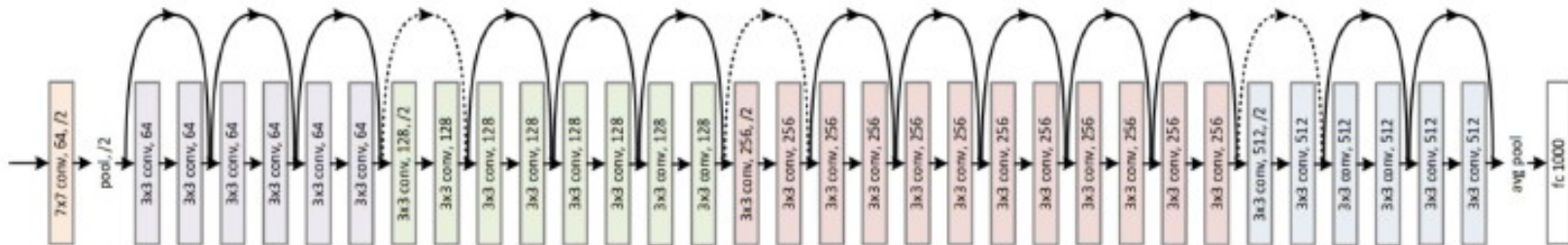
VGG



GoogLeNet



ResNet



CNN, генерация описания картинки

[Lebret, Pinheiro, Collobert 2015] [Kulkarni 11] [Mitchell 12] [Vinyals 14] [Mao 14]



A man riding skis on a snow covered ski slope.
NP: a man, skis, the snow, a person, a woman, a snow covered slope, a slope, a snowboard, a skier, man.
VP: wearing, riding, holding, standing on, skiing down.
PP: on, in, of, with, down.
A man wearing skis on the snow.



A man is doing skateboard tricks on a ramp.
NP: a skateboard, a man, a trick, his skateboard, the air, a skateboarder, a ramp, a skate board, a person, a woman.
VP: doing, riding, is doing, performing, flying through.
PP: on, of, in, at, with.
A man riding a skateboard on a ramp.



The girl with blue hair stands under the umbrella.
NP: a woman, an umbrella, a man, a person, a girl, umbrellas, that, a little girl, a cell phone.
VP: holding, wearing, is holding, holds, carrying.
PP: with, on, of, in, under.
A woman is holding an umbrella.



A slice of pizza sitting on top of a white plate.
NP: a plate, a white plate, a table, pizza, it, a pizza, food, a sandwich, top, a close.
VP: topped with, has, is, sitting on, is on.
PP: of, on, with, in, up.
A table with a plate of pizza on a white plate.



A baseball player swinging a bat on a field.
NP: the ball, a game, a baseball player, a man, a tennis court, a ball, home plate, a baseball game, a batter, a field.
VP: swinging, to hit, playing, holding, is swinging.
PP: on, during, in, at, of.
A baseball player swinging a bat on a baseball field.

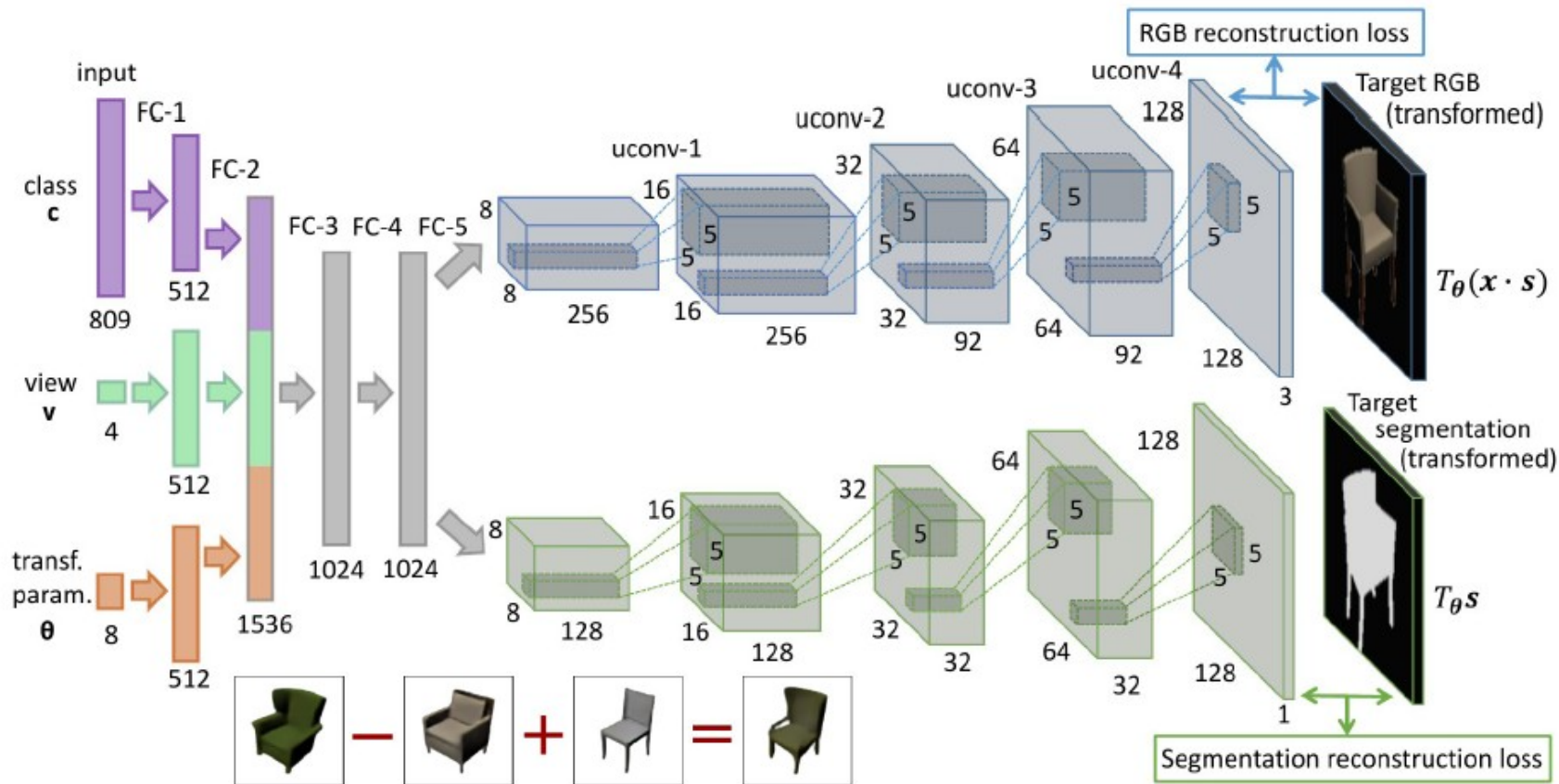


A bunch of kites flying in the sky on the beach.
NP: the beach, a beach, a kite, kites, the ocean, the water, the sky, people, a sandy beach, a group.
VP: flying, flies, is flying, flying in, are.
PP: on, of, with, in, at.
People flying kites on the beach.

CNN, генерация изображений

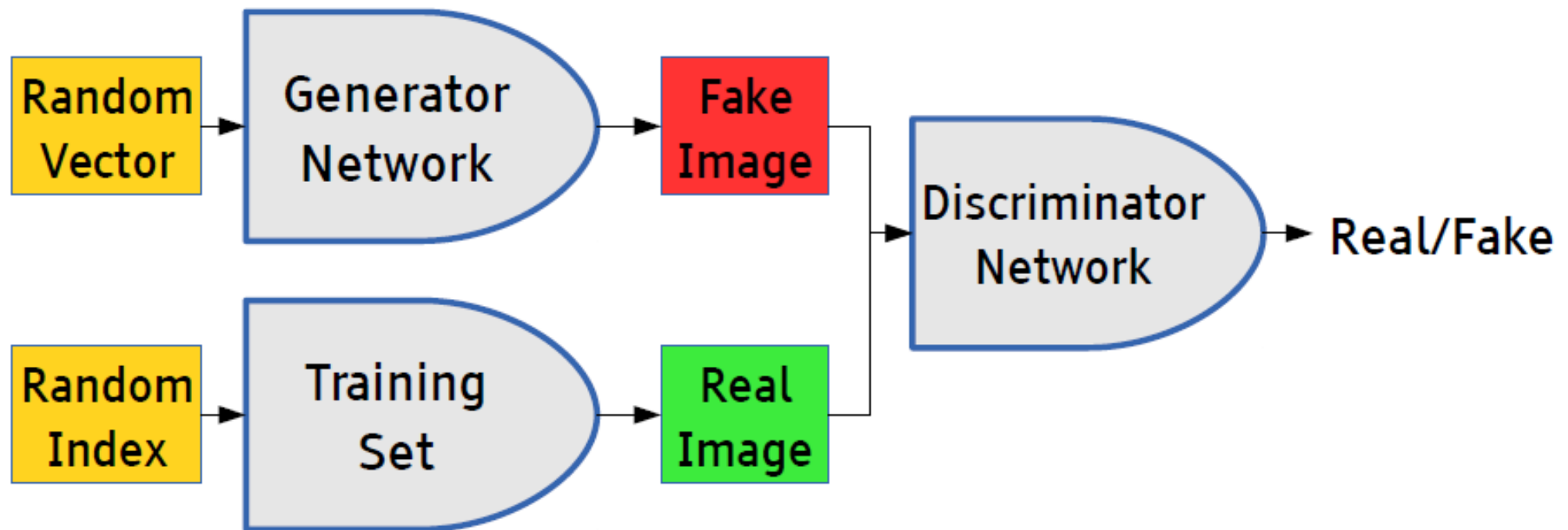
Using ConvNets to Produce Images

[Dosovitskiy et al. Arxiv:1411:5928]

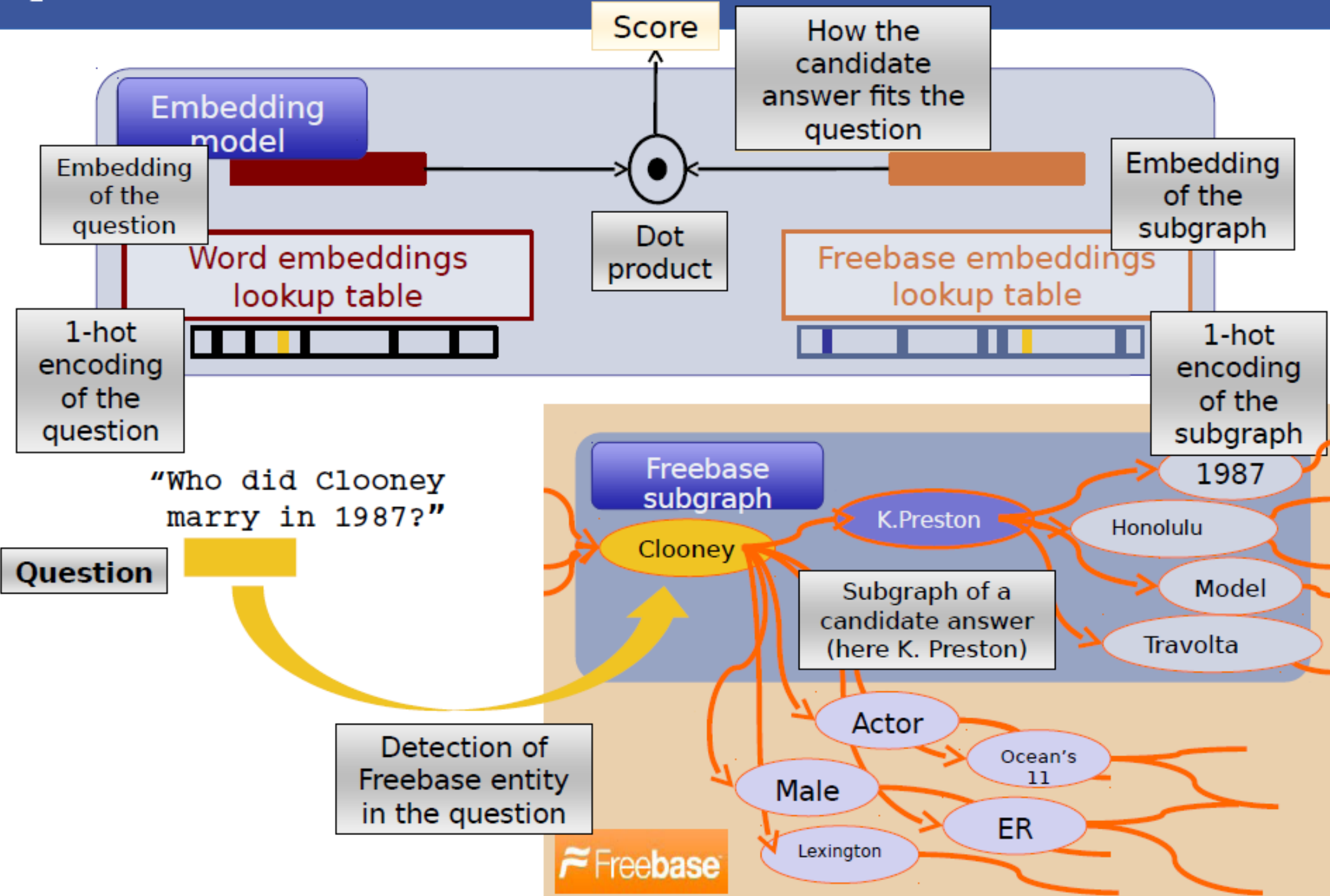


f Generative Adversarial Networks

- [Goodfellow et al. NIPS 2014]
- Generator net maps random numbers to image
- Discriminator learns to tell real from fake images.
- Generator can cheat: it knows the gradient of the output of the discriminator with respect to its input



f Question-Answering System



Типы обучения

Reinforcement Learning

- The machine predicts a scalar reward given once in a while.
- A few bits for some samples



Supervised Learning

- The machine predicts a category or a few numbers for each input
- 10→10,000 bits per sample














Unsupervised Learning

- The machine predicts any part of its input for any observed part.
- Predicts future frames in videos
- Millions of bits per sample



Neural Networks

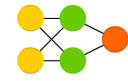
©2016 Fjodor van Veen - asimovinstitute.org

-  Backfed Input Cell
-  Input Cell
-  Noisy Input Cell
-  Hidden Cell
-  Probabilistic Hidden Cell
-  Spiking Hidden Cell
-  Output Cell
-  Match Input Output Cell
-  Recurrent Cell
-  Memory Cell
-  Different Memory Cell
-  Kernel
-  Convolution or Pool

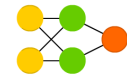
Perceptron (P)



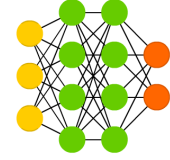
Feed Forward (FF)



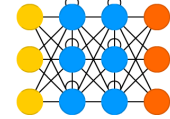
Radial Basis Network (RBF)



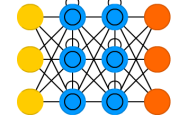
Deep Feed Forward (DFF)



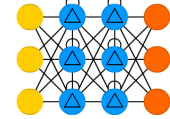
Recurrent Neural Network (RNN)



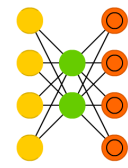
Long / Short Term Memory (LSTM)



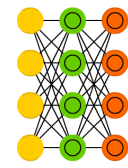
Gated Recurrent Unit (GRU)



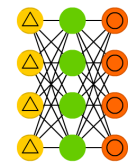
Auto Encoder (AE)



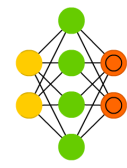
Variational AE (VAE)



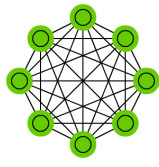
Denoising AE (DAE)



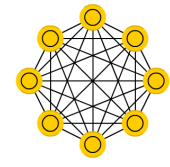
Sparse AE (SAE)



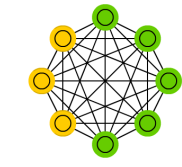
Markov Chain (MC)



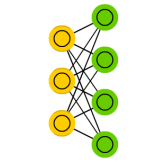
Hopfield Network (HN)



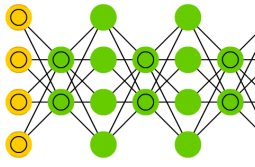
Boltzmann Machine (BM)



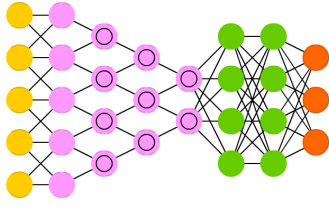
Restricted BM (RBM)



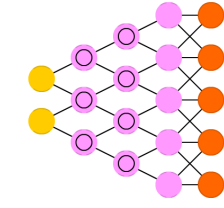
Deep Belief Network (DBN)



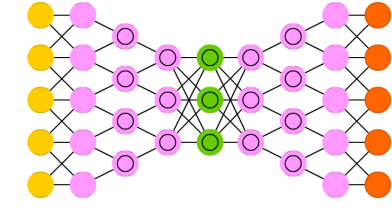
Deep Convolutional Network (DCN)



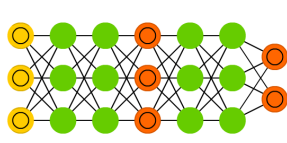
Deconvolutional Network (DN)



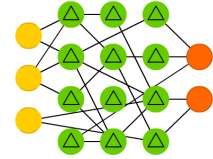
Deep Convolutional Inverse Graphics Network (DCIGN)



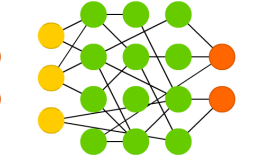
Generative Adversarial Network (GAN)



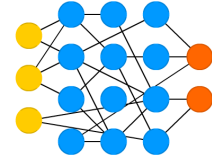
Liquid State Machine (LSM)



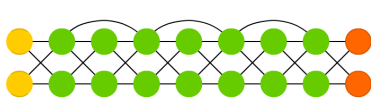
Extreme Learning Machine (ELM)



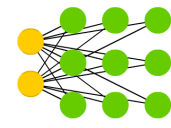
Echo State Network (ESN)



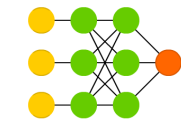
Deep Residual Network (DRN)



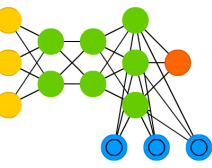
Kohonen Network (KN)



Support Vector Machine (SVM)

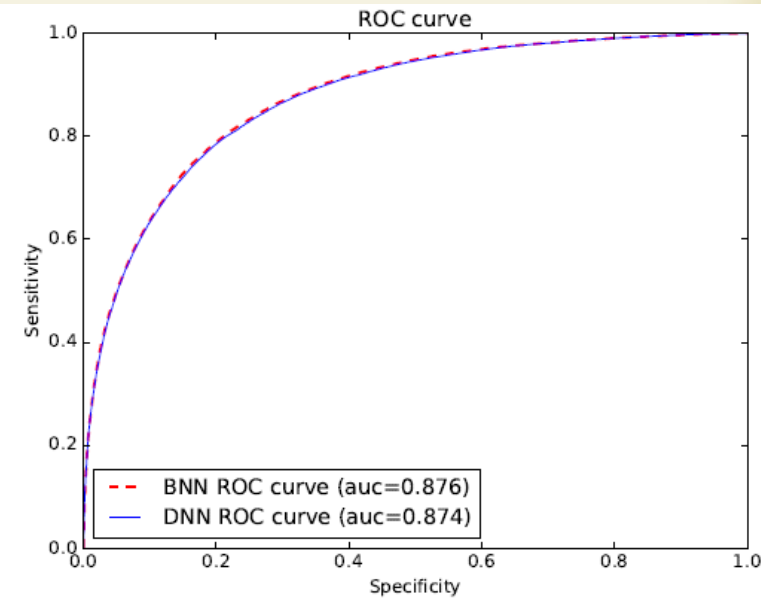
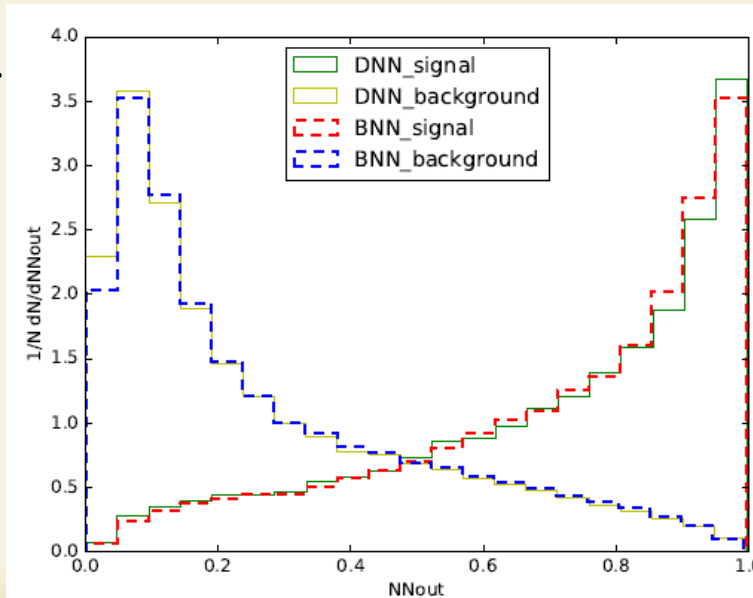


Neural Turing Machine (NTM)



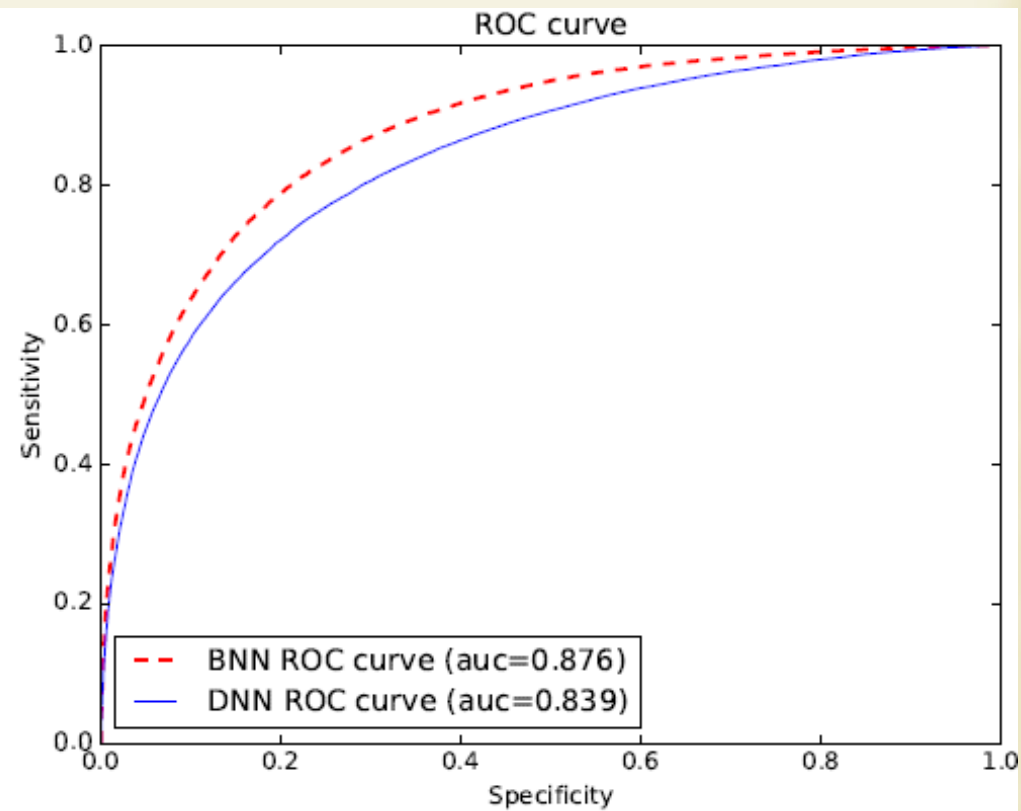
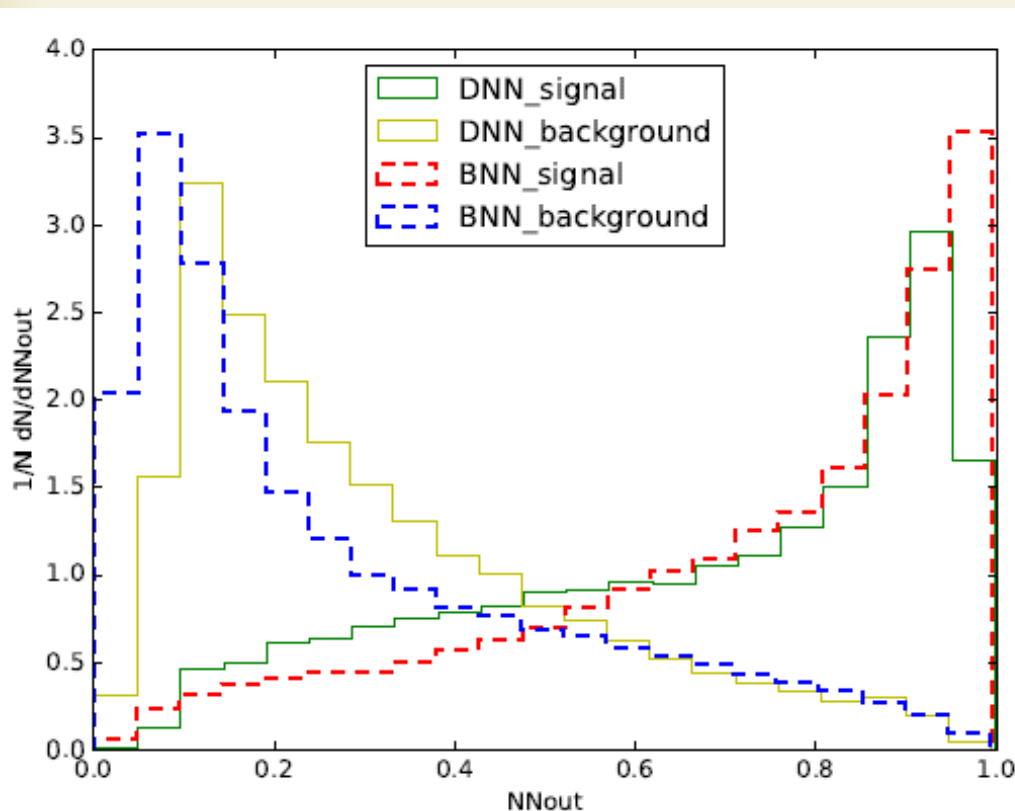
Move to Deep Learning NNs

- Deep learning (DNN) makes possible to move from sophisticated input set of observables to some general raw level observables, and leave sensitive features extraction to DNN, [[Int.J.Mod.Phys.A 35 \(2020\) 21, 2050119](#)]
- Differential cross sections $d\sigma \sim M^2(p_i \cdot p_f, s, t, u)$ is a function of 4-momenta. One needs to separate several functions of 4-momenta.
- 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](#)] take the highly optimized set of optimal variables to compare efficiency
- First compare the methods for benchmark: Bayesian 1-layer NN (BNN, FBM package), as it is taken in CMS analysis, and DNN (Tensorflow) with the same set of high level variables:



Check of the most simple set with Deep Learning NNs

- For the first step one can take 4-momenta of the final particles as an input set for DNN
- Comparison of DNN (3 hidden layers) with benchmark BNN (trained with highly optimized set of high level variables) demonstrates not an optimal efficiency



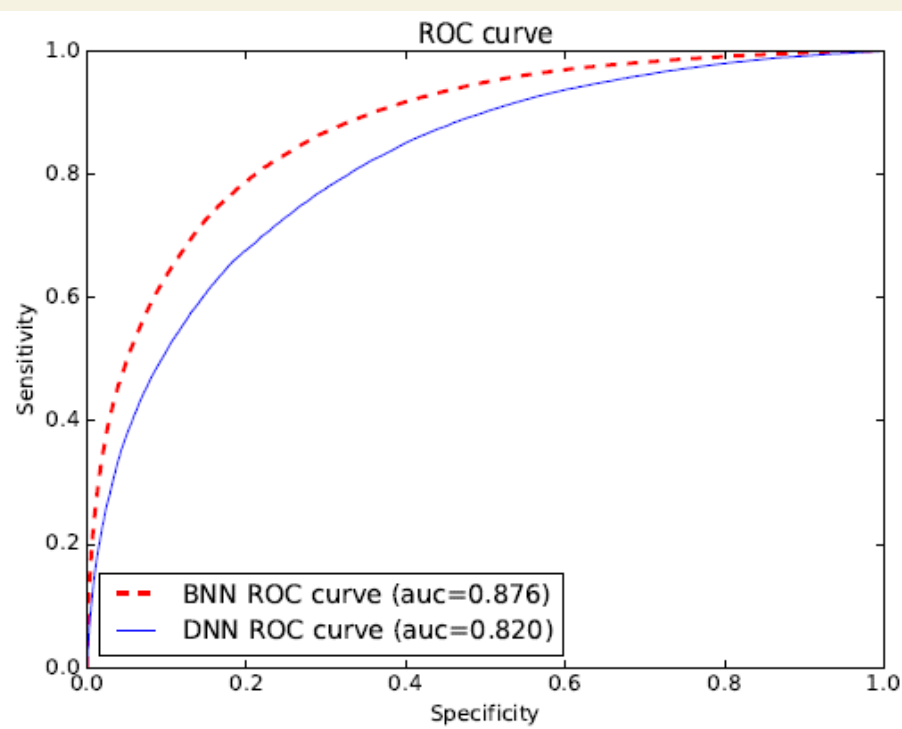
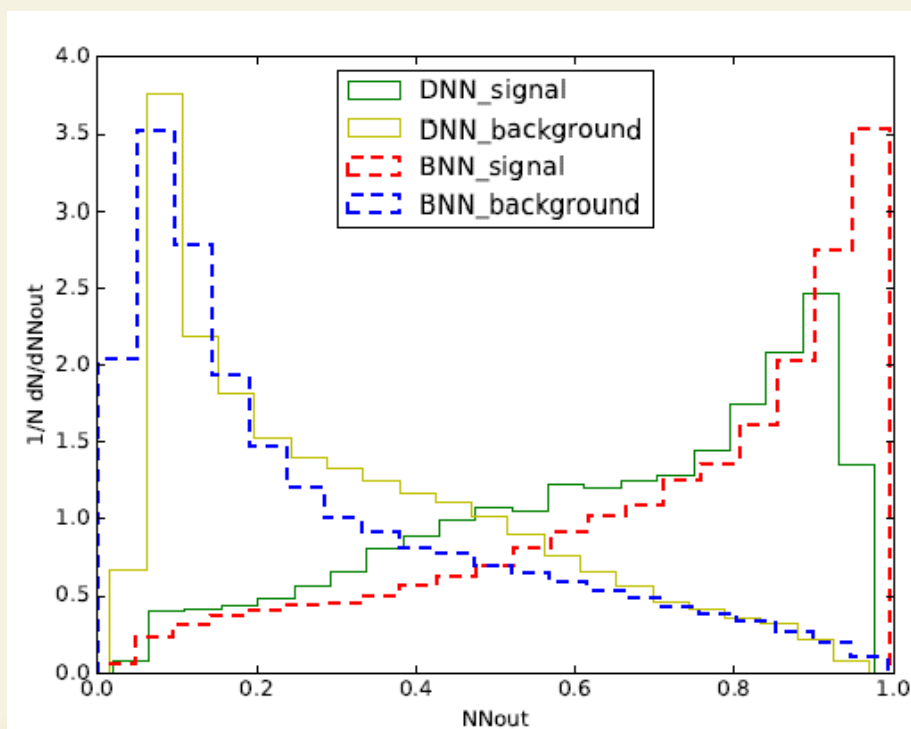
Check of quadratic forms of 4-momenta as an input for DNN

- ~ An example of simple matrix element (s-channel $ud \rightarrow tb$ single top production) depends on scalar products of 4-momenta and Mandelstam variables (s,t,u in general case)

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

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

- ~ Comparison of benchmark BNN with DNN trained on scalar products of final particles and s-Mandelstam variables. Eff. is still far from benchmark BNN.

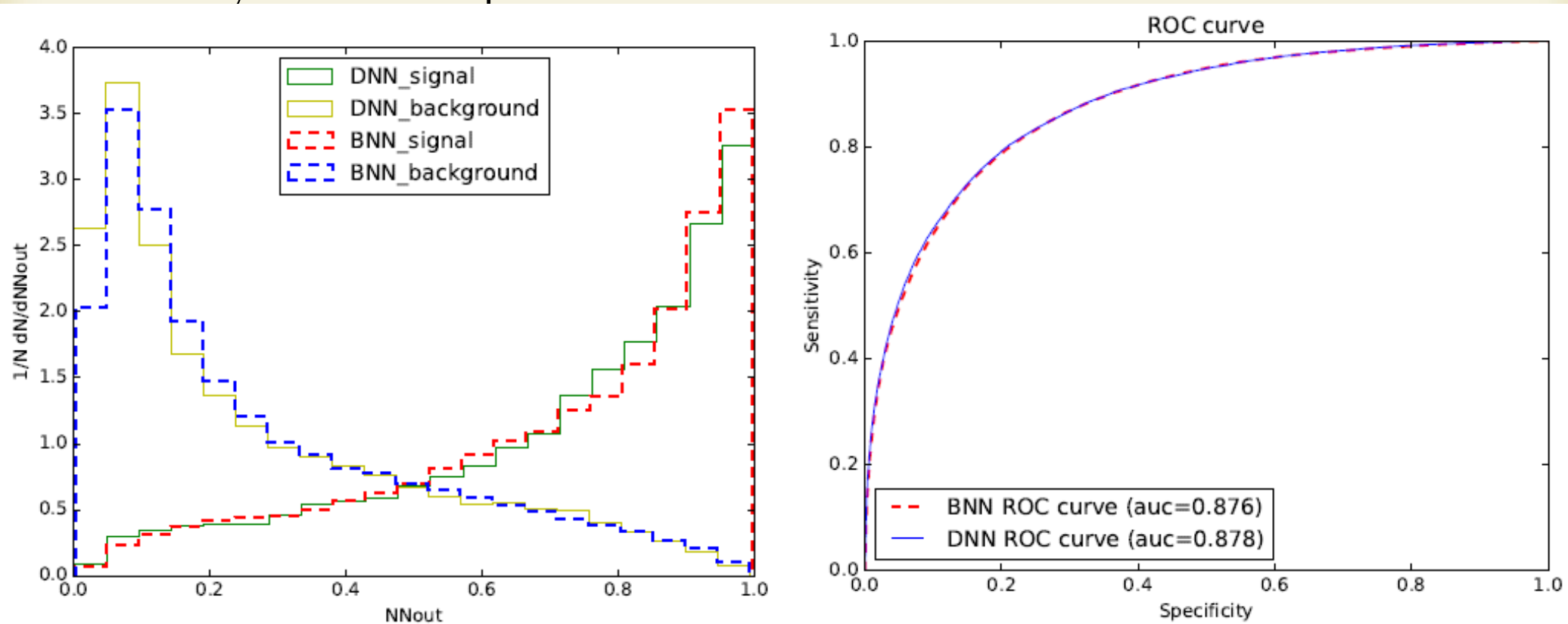


Initial state particles 4-momenta as an input for DNN

- ~ M^2 is a function of not only final particles momenta, but initial particles momenta as well (not available for hadron colliders). In the massless case p_{in} can be represented as and approximated with P_T and pseudorapidity
- [Phys.Atom.Nucl. 71 (2008) 2, 388-393]

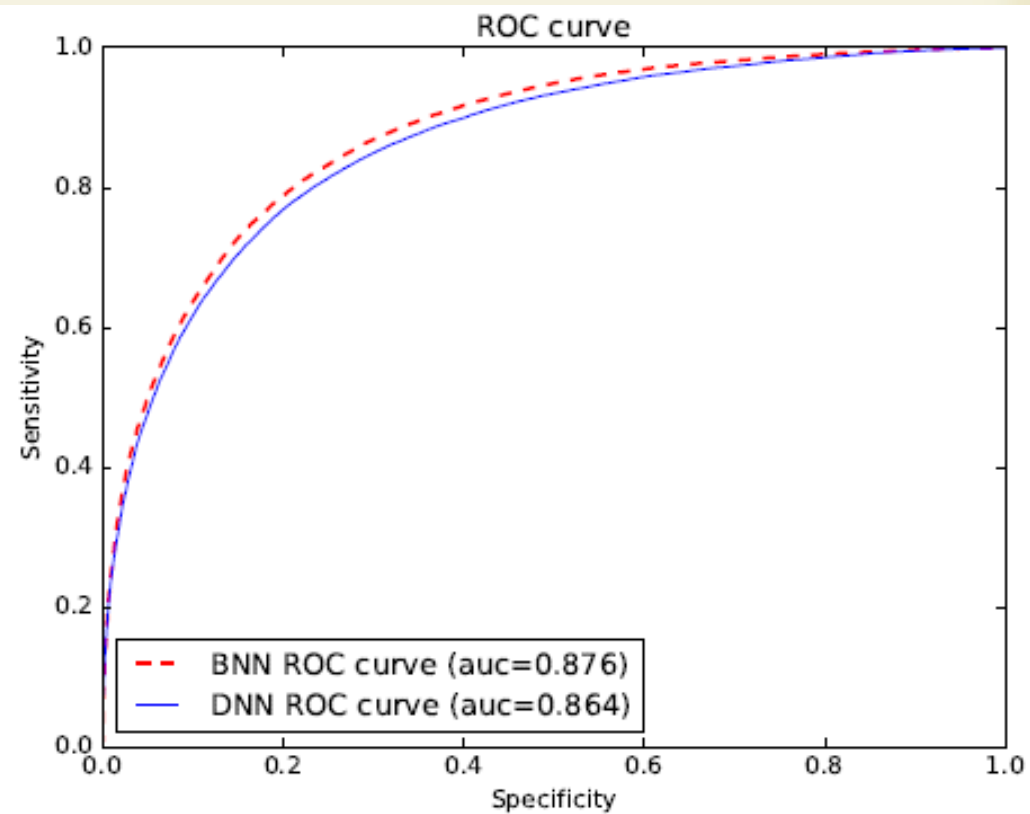
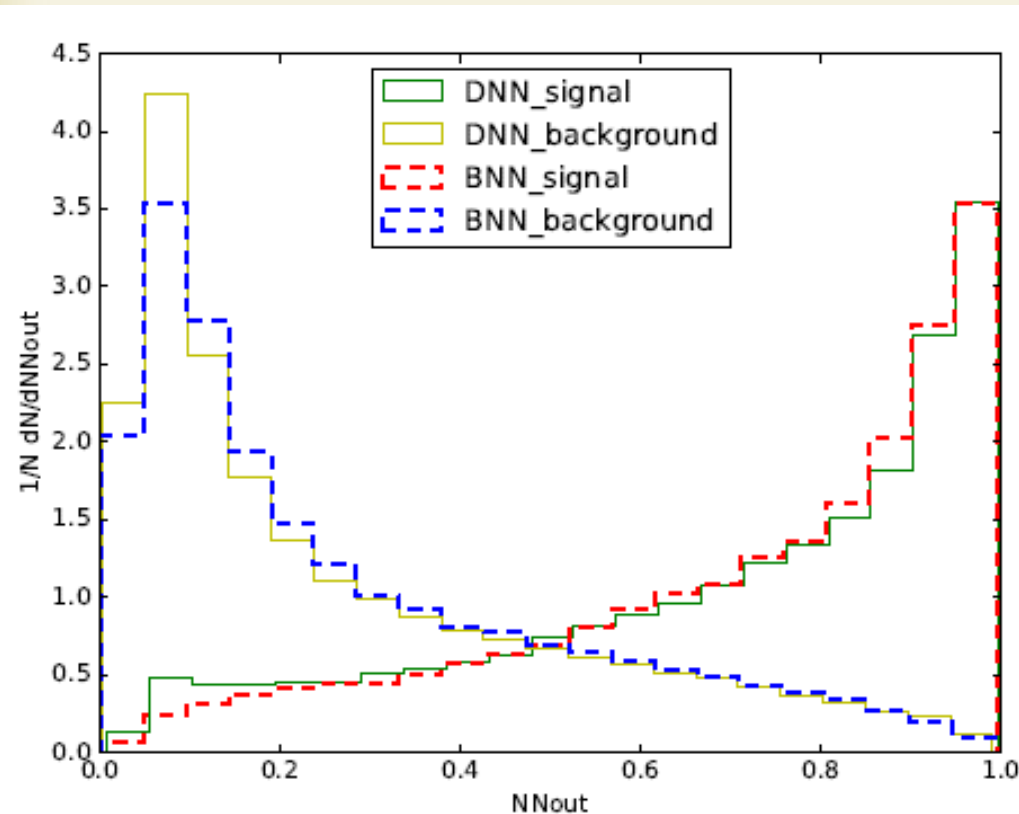
$$\hat{t}_{i,f} = -\sqrt{\hat{s}}e^Y p_T^f e^{-|y_f|}$$

- ~ 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



Check for the completeness

- ~ In addition to the proposed set one can add 4-momenta of the final particles.
- ~ 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.



Примеры современных DNN в НЕР, b-таггирование в эксперименте CMS (LHC)

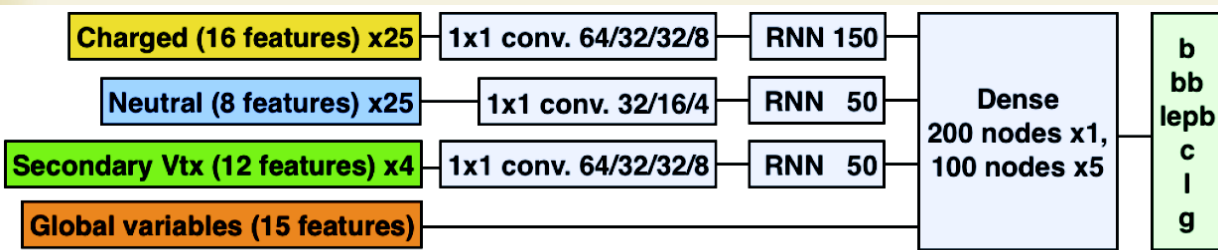
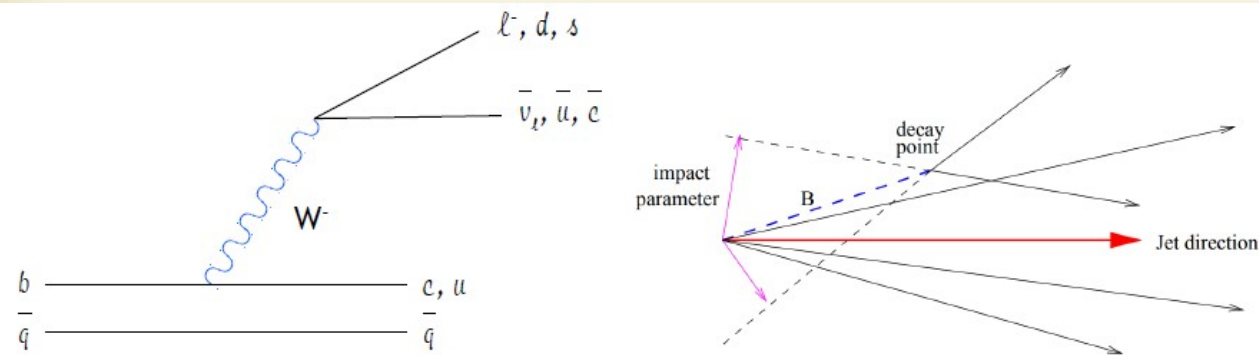
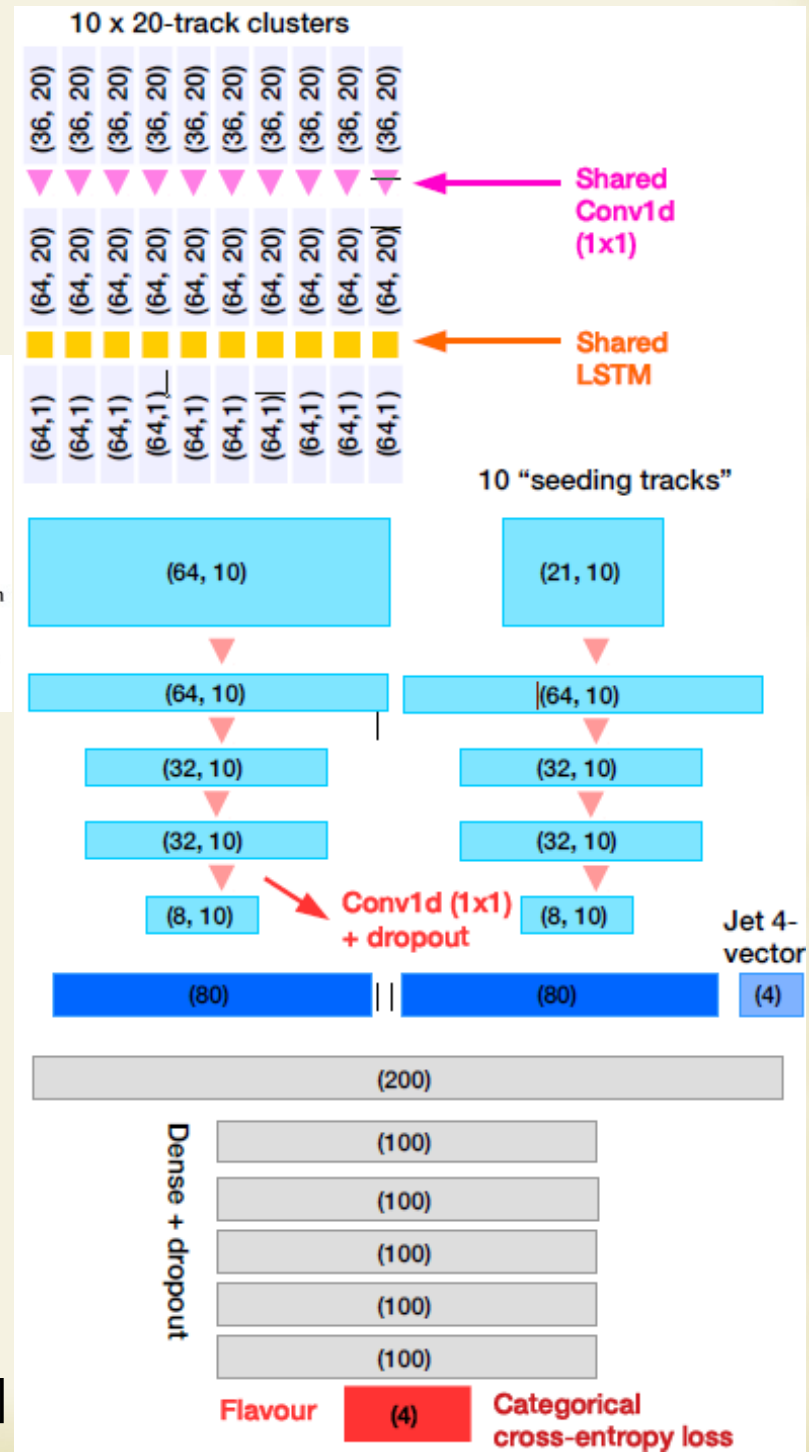


FIGURE 5.18: DNN architecture of the DeepJet network. The total number of trainable parameters is about 265K.

[L.Giannini CERN-THESIS-2020-107]



Flavour (4) Categorical cross-entropy loss

Аппаратная и программная реализация

~ Аппаратная реализация

- ~ CPU, Графические процессоры (GPU) Nvidia (CUDA, GeForce 3080 Ti, Titan, Tesla)
- ~ Специализированные чипы; мемристорные матрицы
- ~ Суперкомпьютеры (параллельные вычисления)

~ Программная реализация

- ~ TensorFlow, Keras
 - ~ Theano, Keras
 - ~ Torch
 - ~ Caffe, ...
- ~ Наши текущие исследования методов применения DNN: использование байесовского подхода в глубоких сетях (tensorflowProbability), применение variational dropout [1701.05369]

Другие NN сети

- ~ Осцилляторные сети, модель скоррелированного возбуждения нейронов в мозге
- ~ Мемристорные сети, аппаратная реализация NN на основе нового элемента - мемристора (электрическое сопротивление зависит от величины ранее пропущенного эл. тока)
- ~ Нейроморфные интерфейсы, мемристорные сети соединенные с живыми клетками выращенными на электродных матрицах

■ Deep Learning is enabling a new wave of applications

- ▶ **Today:** Image recognition, video understanding: **vision now works**
- ▶ **Today:** Better speech recognition: **speech recognition now works**
- ▶ **Soon:** Better language understanding, dialog, and translation

■ Deep Learning and Convolutional Nets are being widely deployed

- ▶ **Today:** image understanding at Facebook, Google, Twitter, Microsoft.....
- ▶ **Soon:** better auto-pilots for cars, medical image analysis, robot perception

■ We need hardware (and software) for embedded applications

- ▶ For smart cameras, mobile devices, cars, robots, toys....

■ **But we are still far from building truly intelligent machines**

- ▶ We need to integrate **reasoning** with deep learning
- ▶ We need a good architecture for **“episodic” (short-term) memory**.
- ▶ We need to find good principles for **unsupervised learning**