Neural networks in the Baikal-GVD experiment: selection of neutrino events and neutrino energy reconstruction

Matseiko Albert[1,2] , MIPT, INR RAS

Quarks-2024
The 22nd conference on high-energy physics
Pereslavl
19-24 May 2024

Outlook

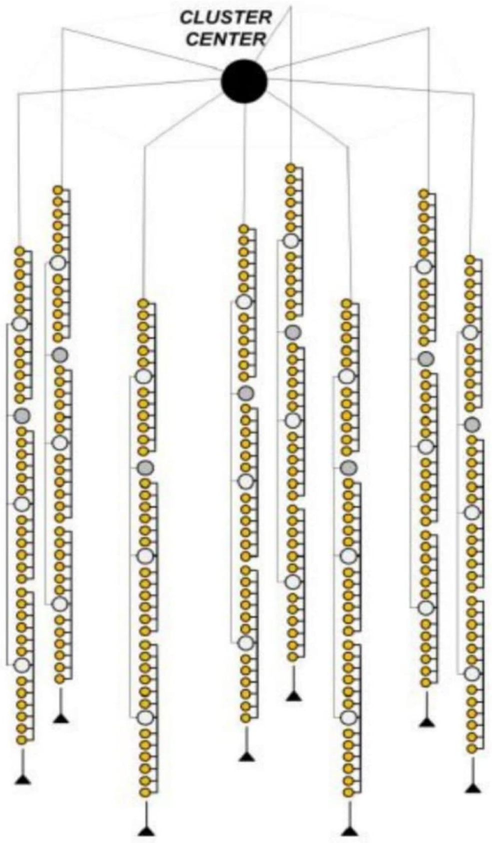
1. **BaikalGVD experiment:**
   a) structure
   b) events
   c) data

2. **Neutrino selection** against the EAS background

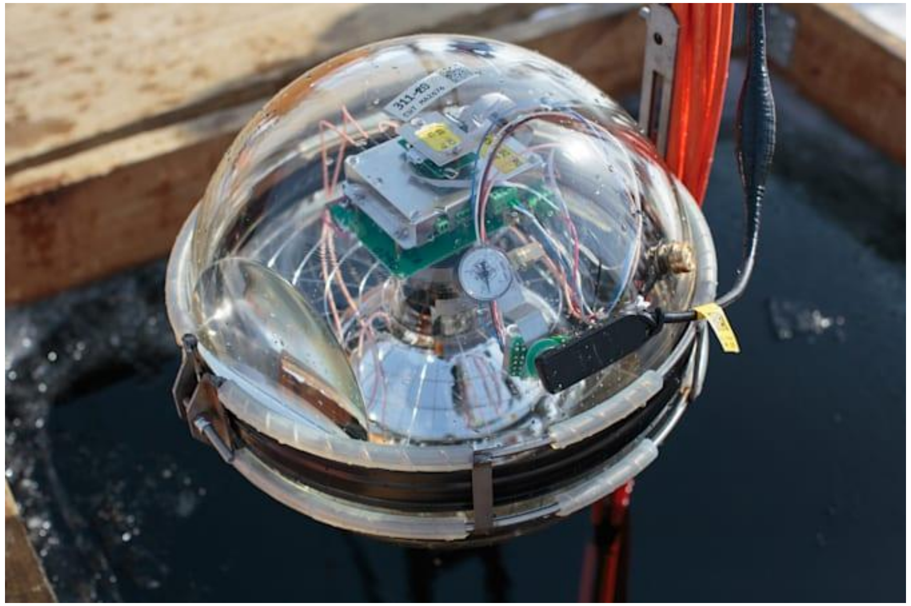
3. **Neutrino energy reconstruction**

I. Baikal-GVD

Baikal-GVD



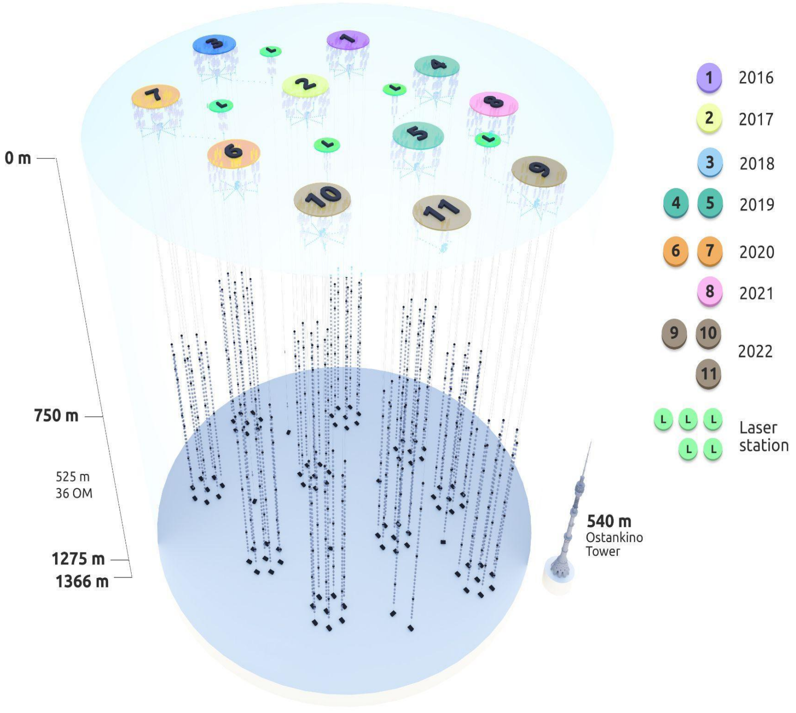
8 strings with 36 OM (optic modules)



- Detect cherenkov radiation
- Live calibration of OM positions (accuracy ~20 cm)
- Accuracy of determining the response time ~2 ns.

Baikal-GVD

Cluster setup process



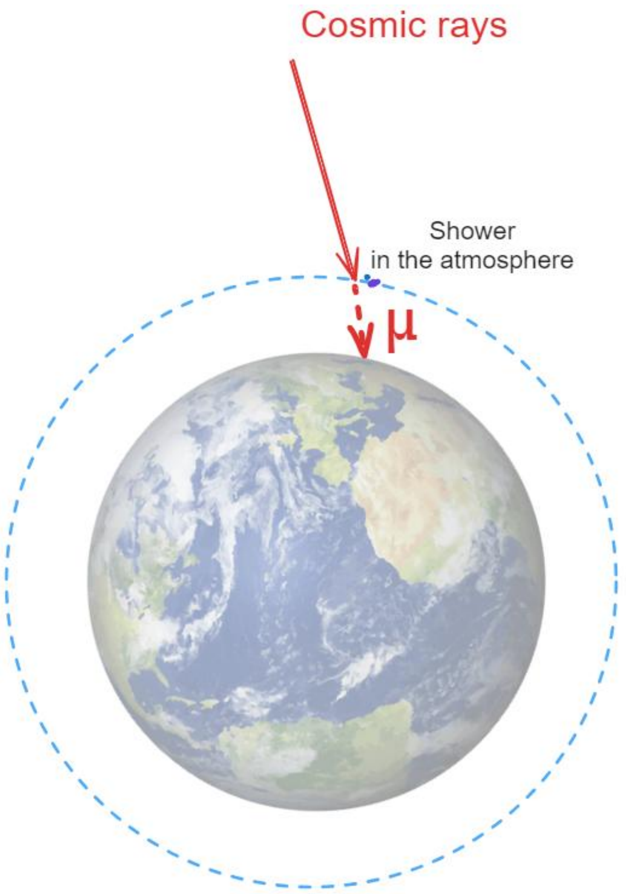
13 clusters
are setup by now

Events of all origins are registered:
~2 million per day

EAS and ν induced events

1) EAS

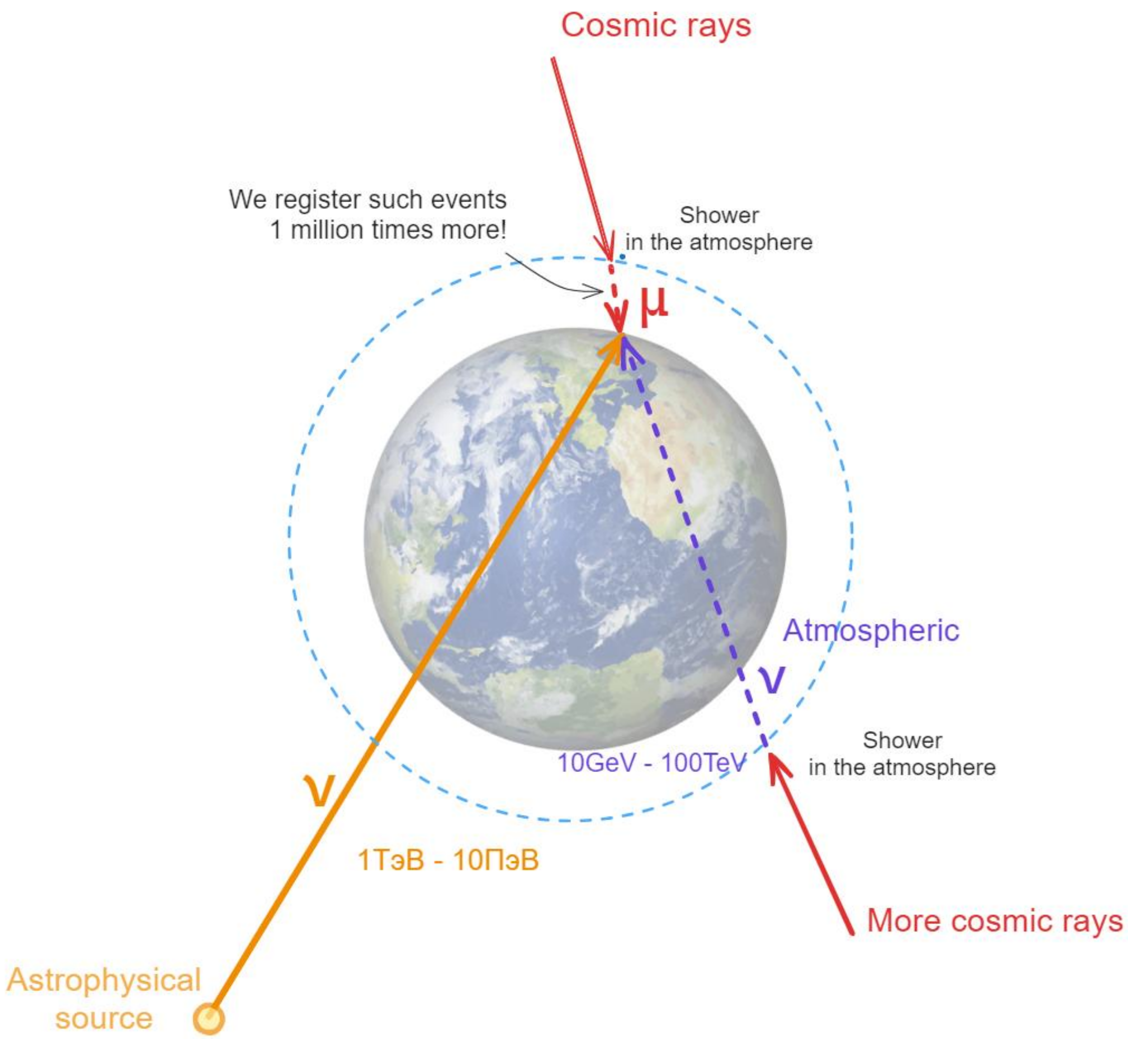
- showers from cosmic rays
- mostly muons reach the cluster
- «down-going» events



EAS and ν induced events

1) EAS
- showers from cosmic rays
- mostly muons reach the cluster
- «down-going» events

2) Neutrino
- astrophysical or atmospheric
- leptons are born in the cluster
- easily pass the Earth
  -> «up-going» events



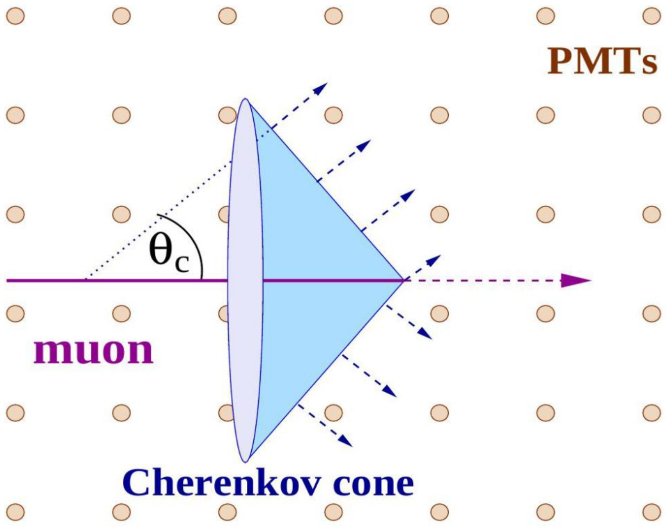
Baikal-GVD

Event options

ν_μ or atmosperic μ

⇩

Track events

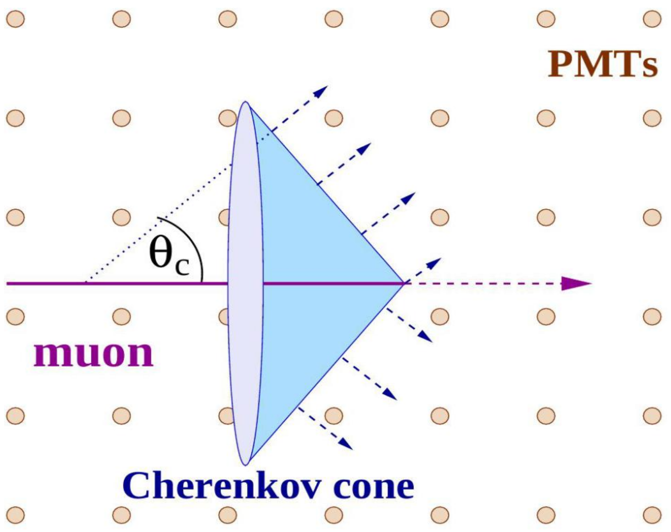


Baikal-GVD

Event options

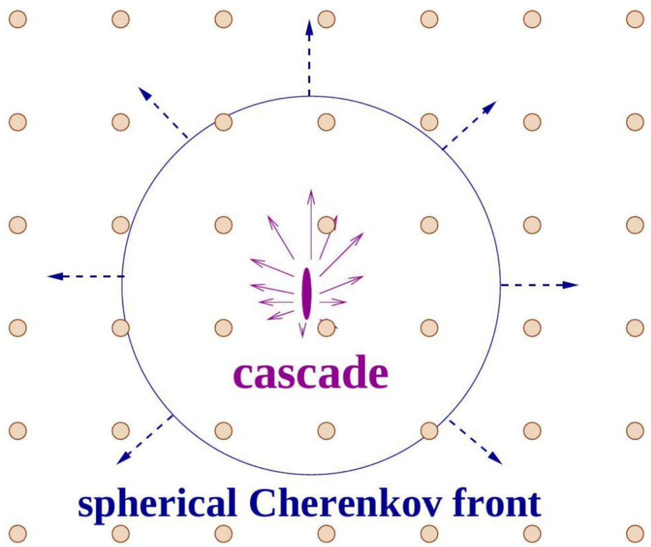
ν_μ or atmosperic μ

Track events



ν_e , ν_μ , ν_τ

Cascade events

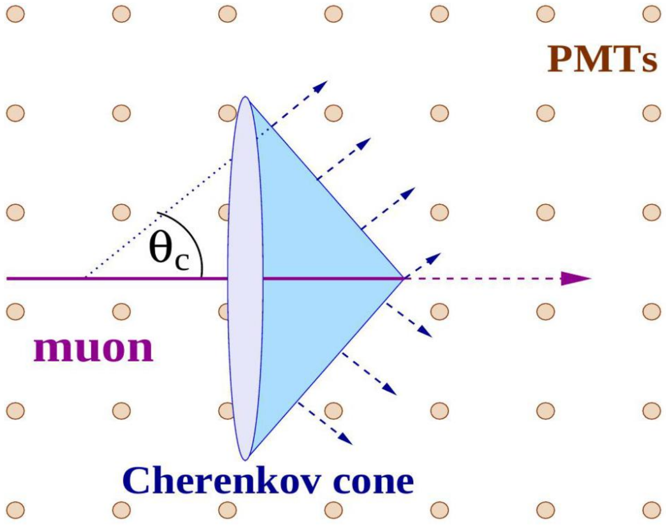


Baikal-GVD

Event options

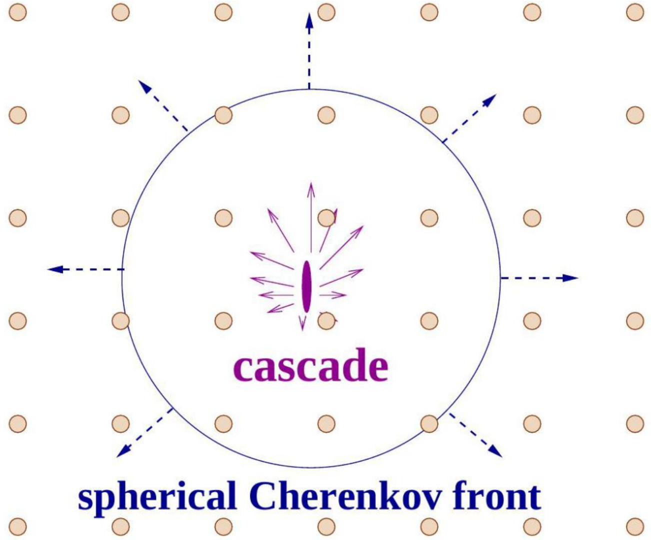
ν_μ or atmosperic μ

Track events

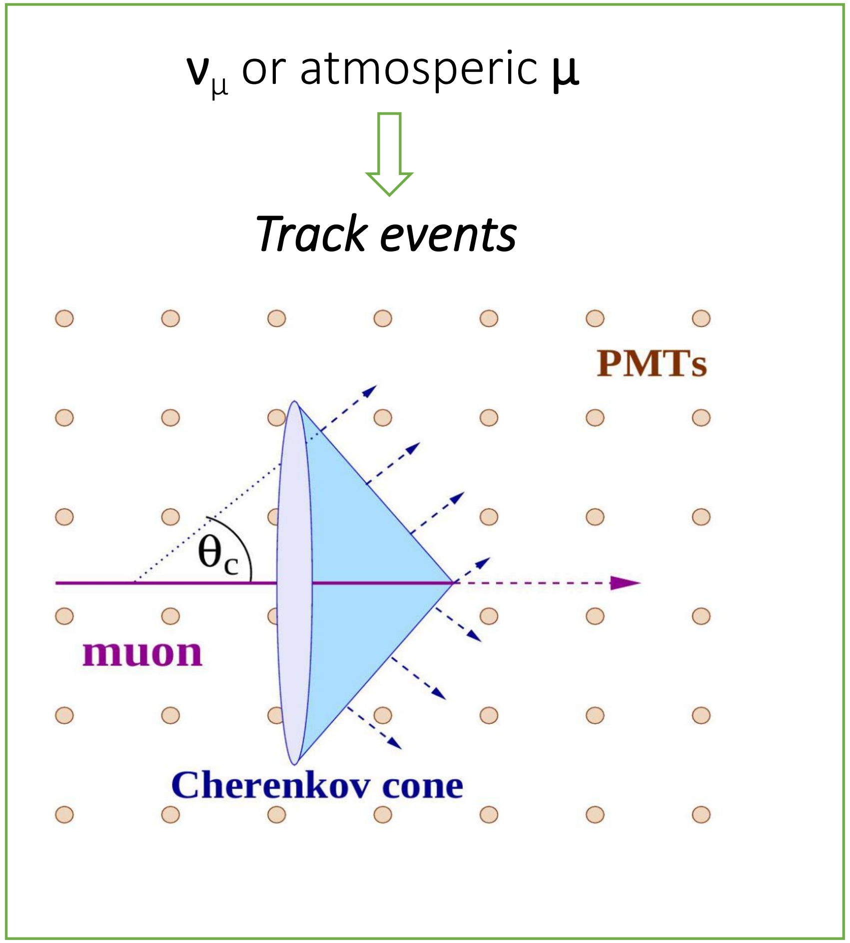


ν_e , ν_μ , ν_τ

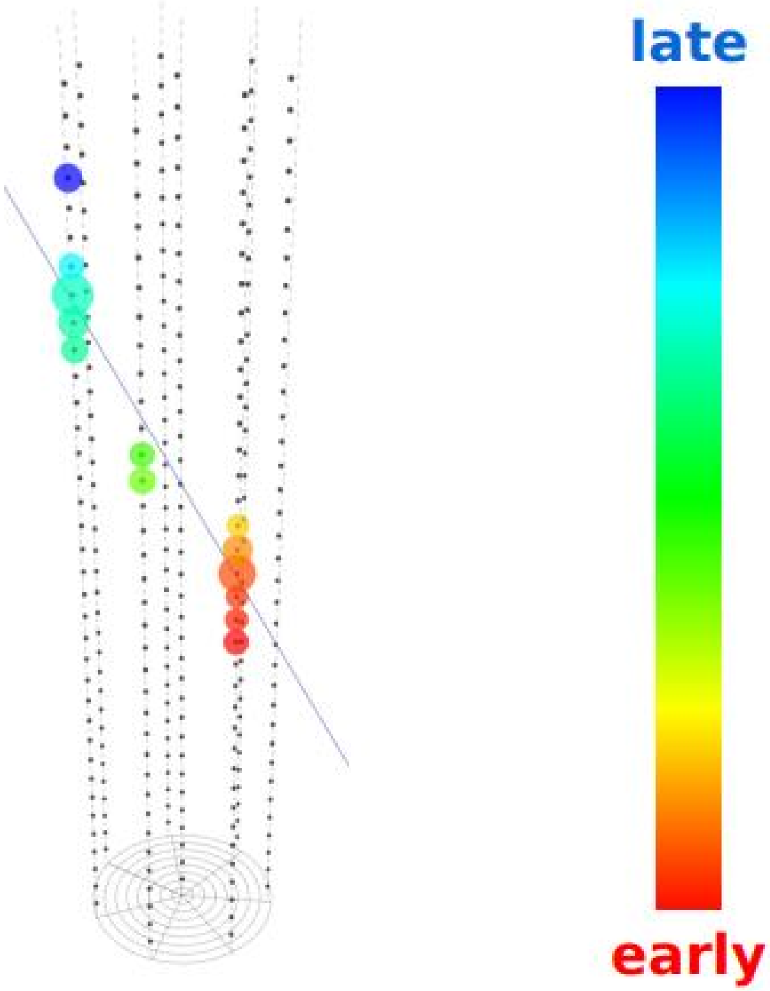
Cascade events



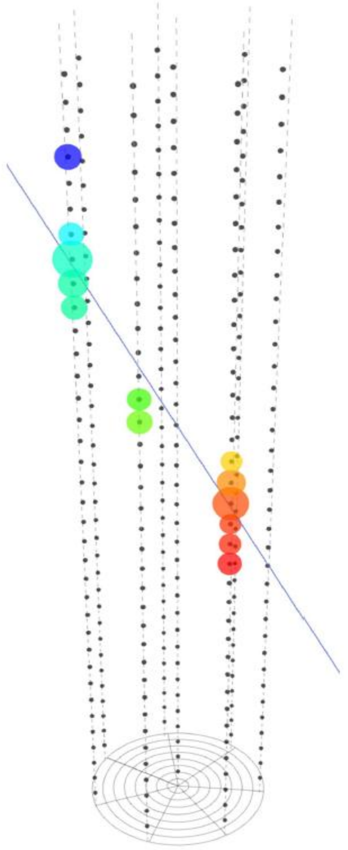
Baikal-GVD



Track event picture

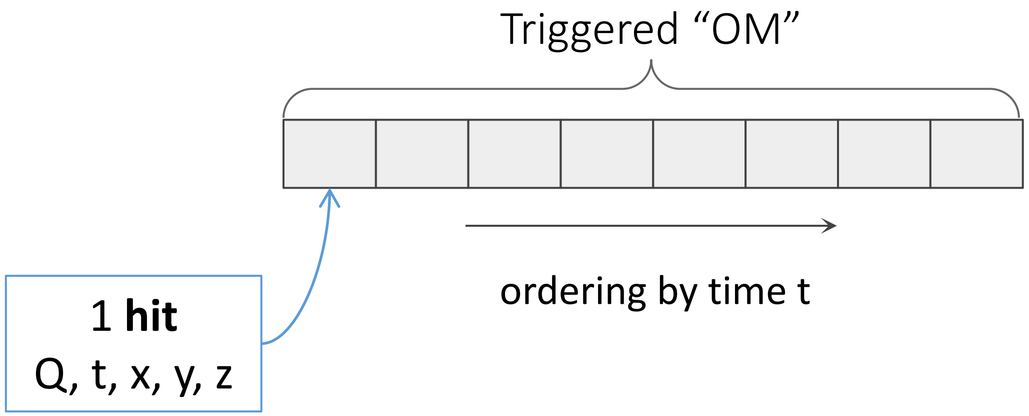


Data representation

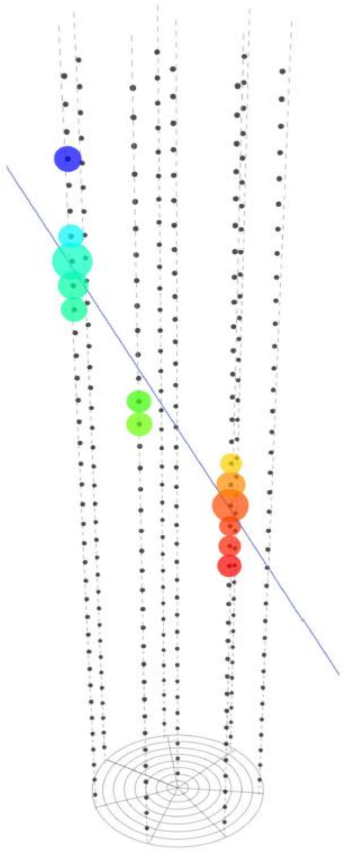


Single cluster

Causal (time-ordered)



Data representation



Single cluster

Causal (time-ordered)

Triggered "OM"

1 **hit**
Q, t, x, y, z

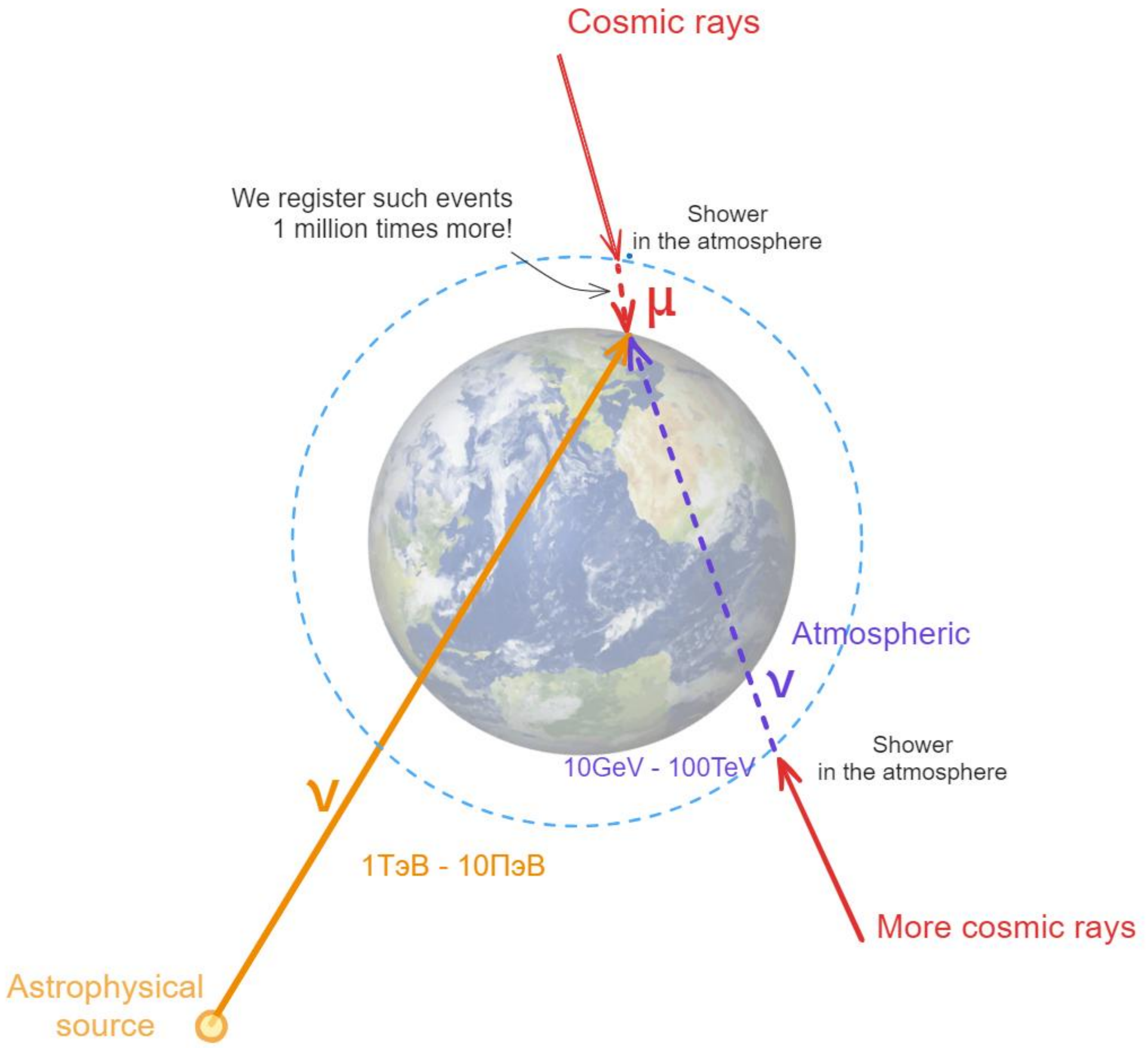
ordering by time t

1D convolutions, recurrent networks, transformers

II. Neutrino selection against the EAS background

Motivation

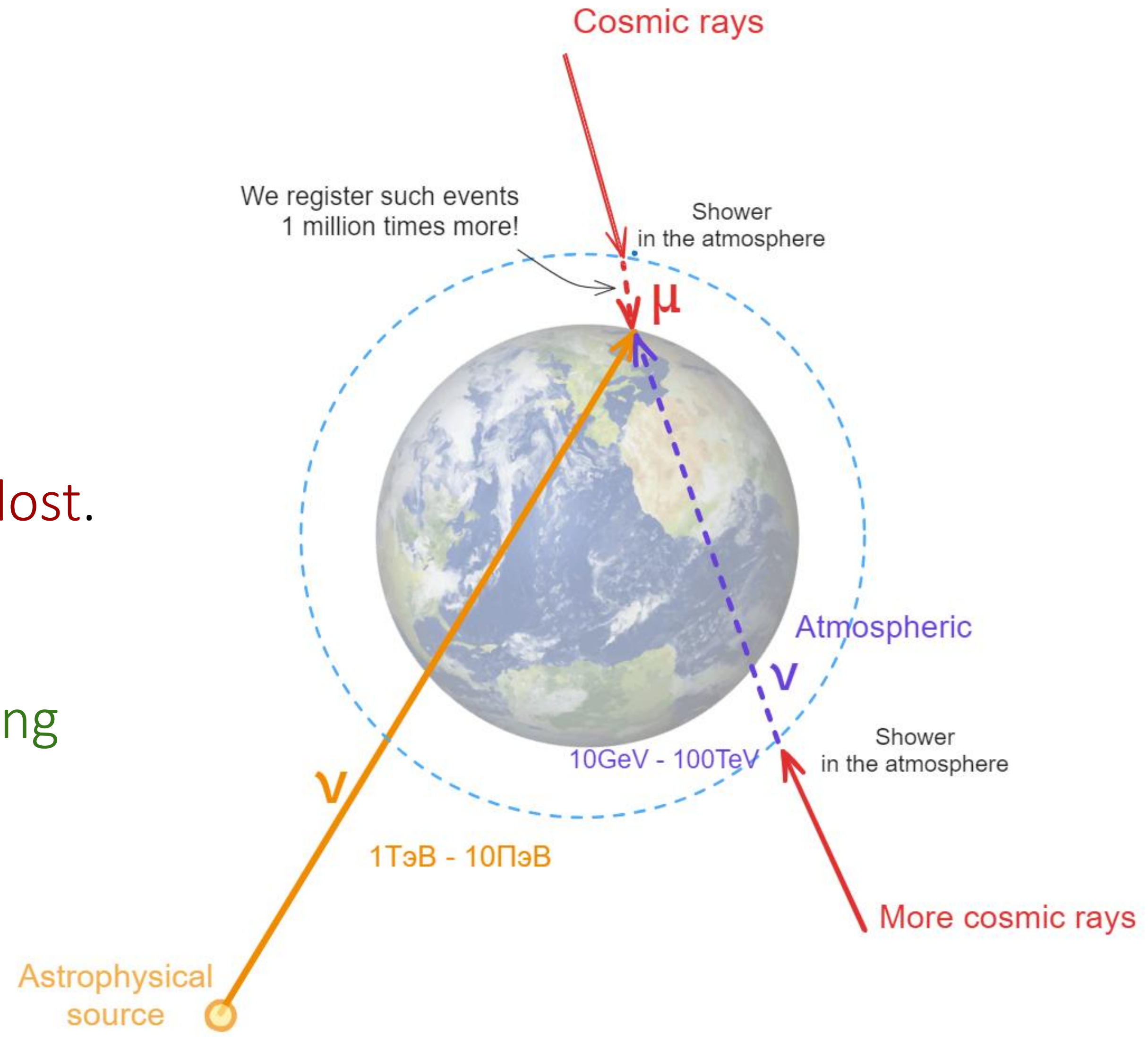
- EAS to ν events ratio = 10^6-10^7.



Motivation

- EAS to ν events ratio = 10^6-10^7.
- Standard approach:
  reconstruction of the zenith angle + cut.
  Computationally expensive, ~50% ν are lost.

The goal is to achieve better separation using neural networks.



Dataset

- Using **Monte-Carlo simulation**[1]**!**
  EAS evolution and propagation of particles in water.

- Track events:
  1) Muons from EAS
  2) $\boldsymbol{\nu}_\mu$

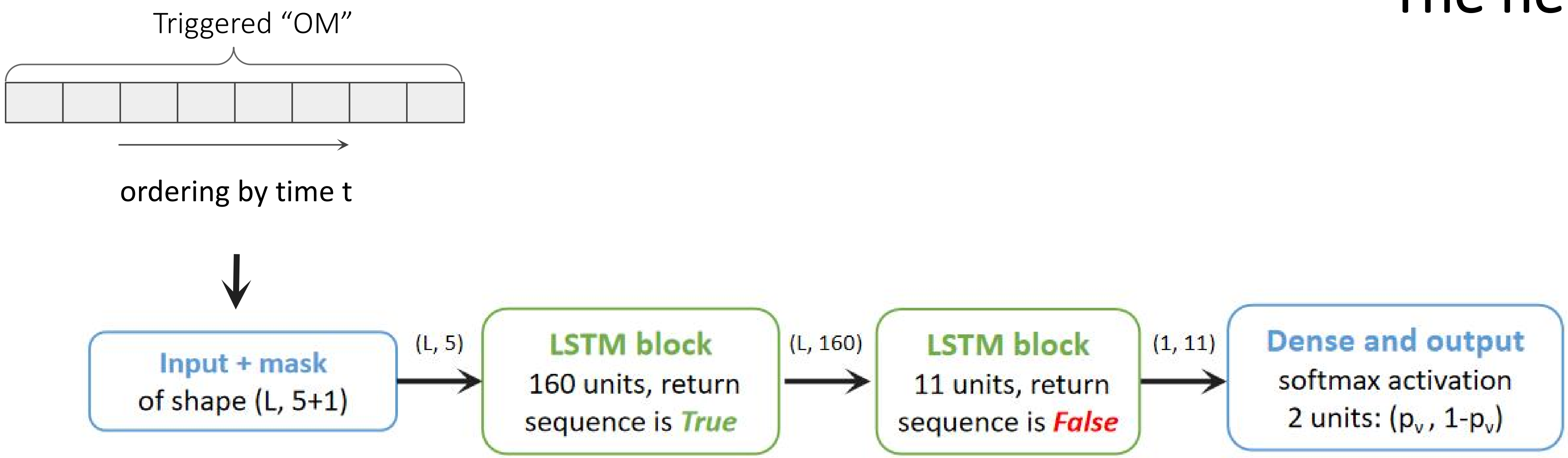
- Cuts:
  min 8 signal hits
  min 2 strings triggered

- Target feature — type of particle
  Labels: 0 — EAS, 1 — neutrino

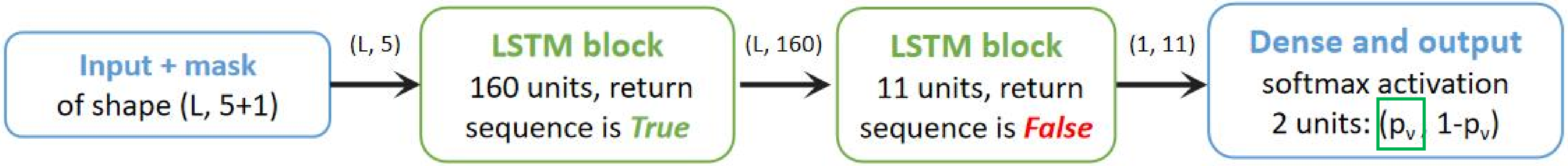
	train	test	validation
EAS	≈$5*10^5$	≈10^5	≈$5.4*10^6$
Neutrino	≈$5*10^5$	≈10^5	≈$1.4*10^7$

[1] arxiv.org/abs/2106.06288

The network

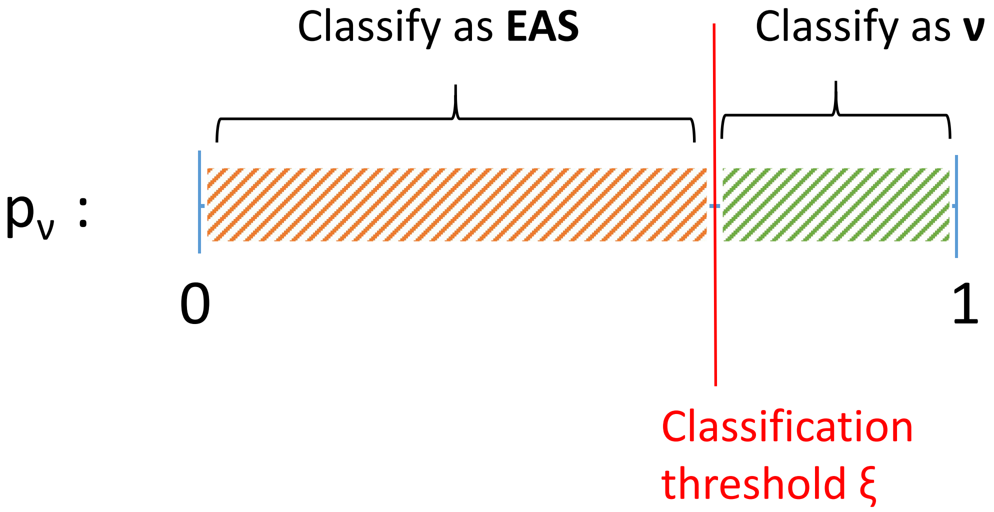

Triggered “OM”
ordering by time t
Input + mask
of shape (L, 5+1)
(L, 5)
LSTM block
160 units, return sequence is *True*
(L, 160)
LSTM block
11 units, return sequence is *False*
(1, 11)
Dense and output
softmax activation
2 units: (p_v , 1-p_v)

The network



- predictions p_v
  --> classification

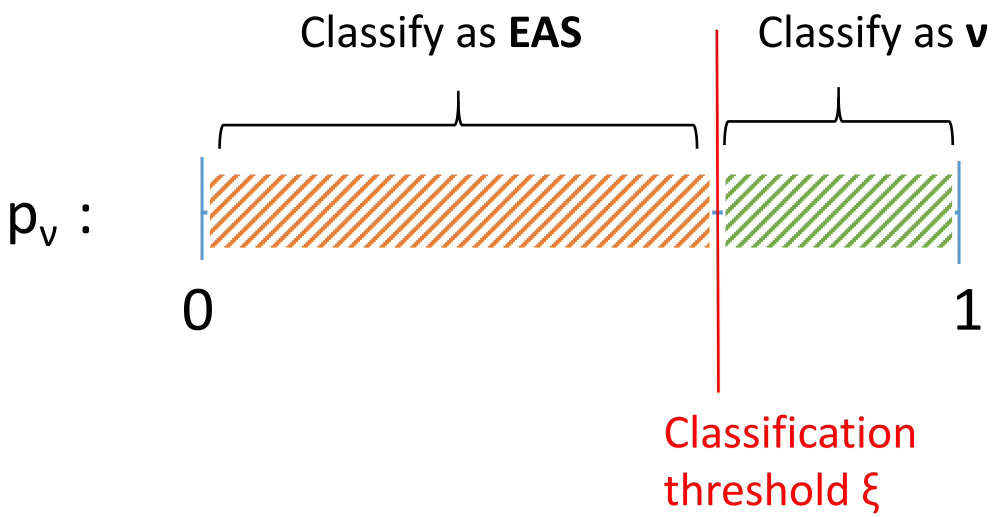
- threshold ξ is close to 1!



Metrics

- True Positive Rate (TPR)
  fraction of ν identified correctly

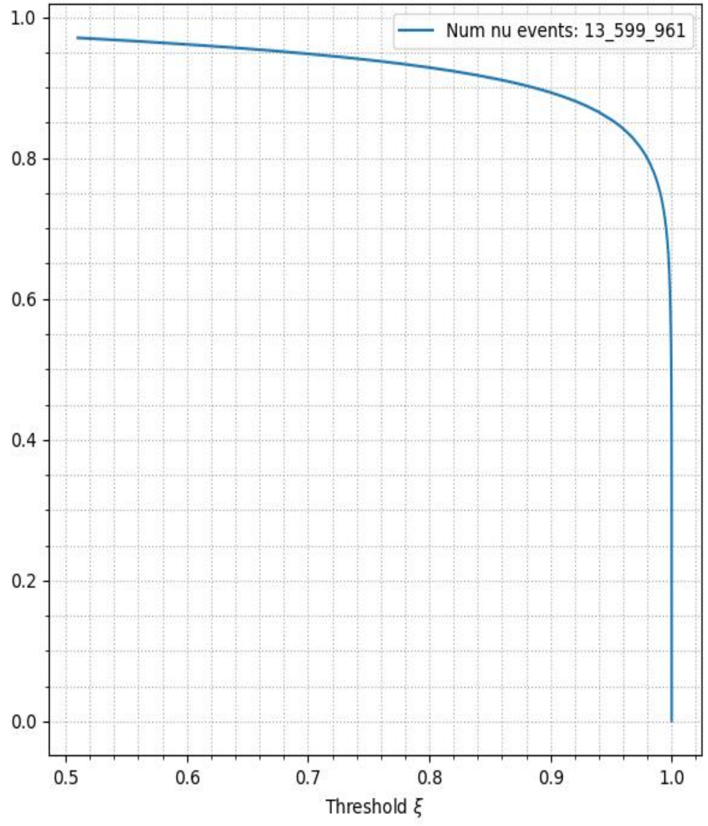
- False Positive Rate (FPR):
  fraction of EASs falsely assigned to ν



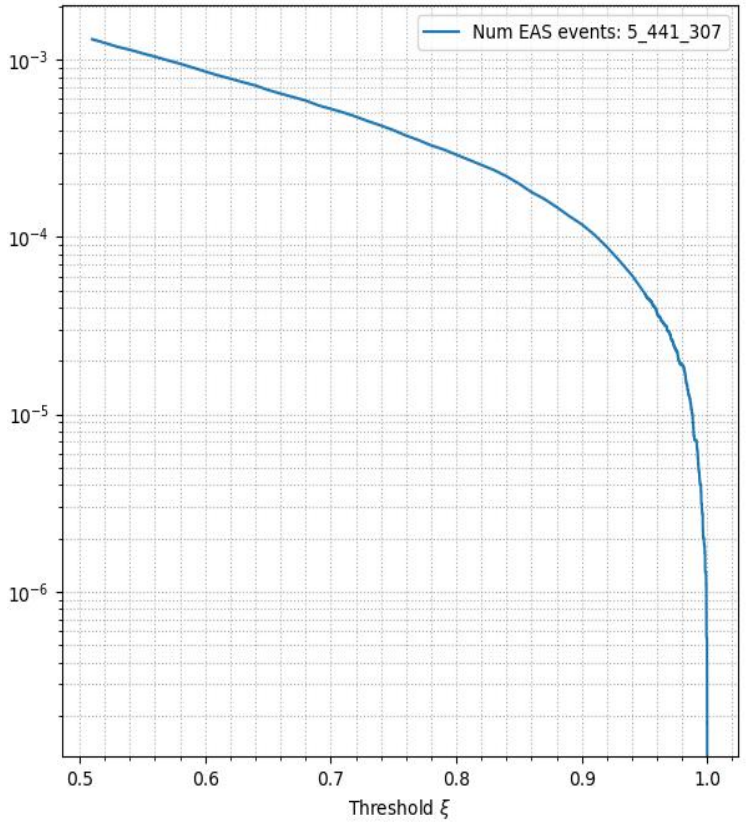
Interested in region, where FPR=10^{-6}

Metrics

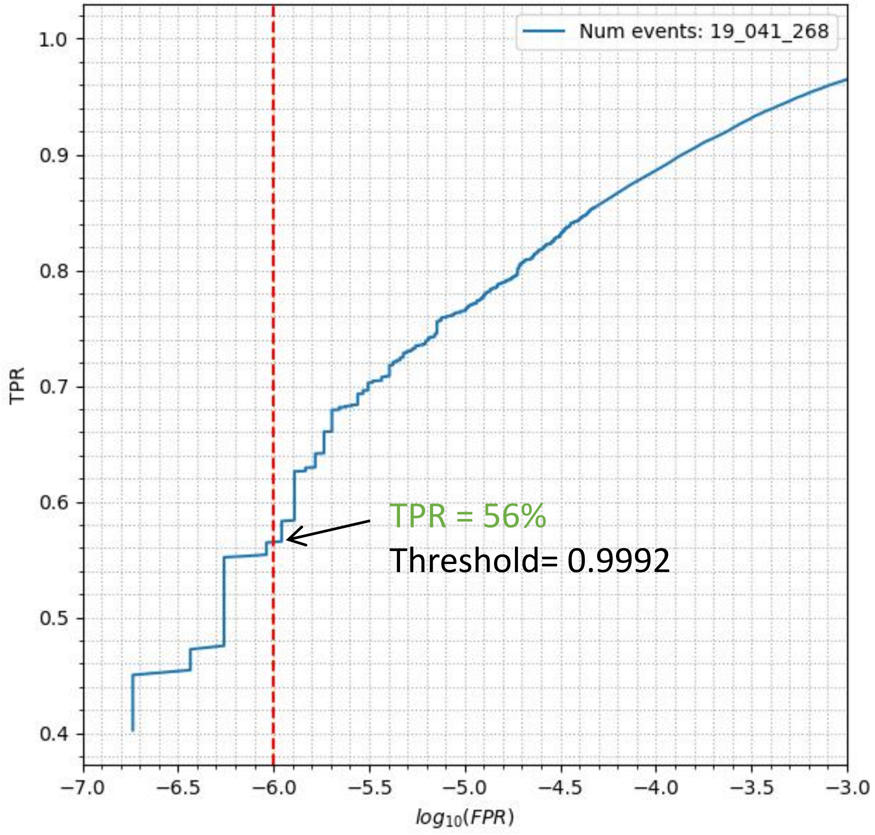
TP rate



FP rate

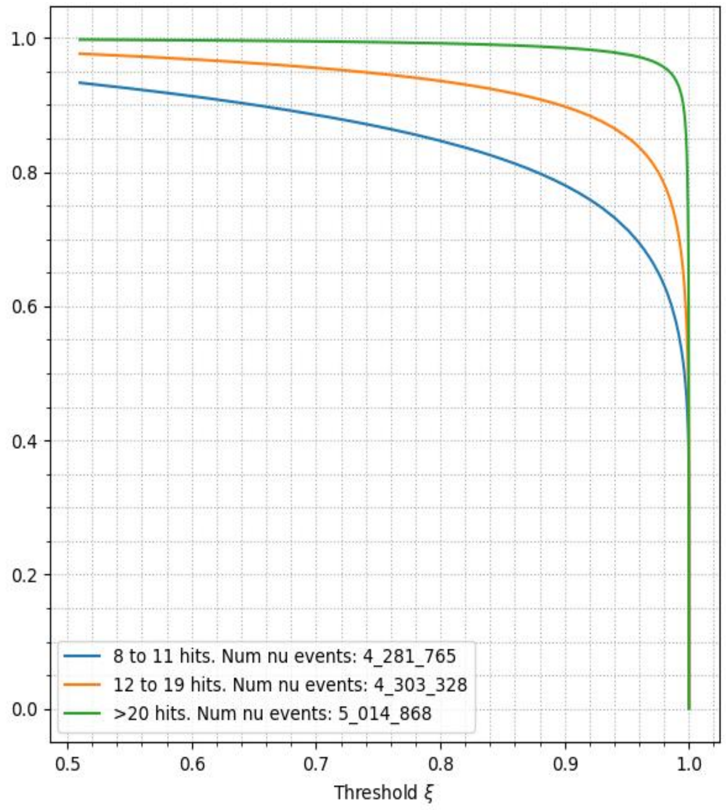


TP rate vs FP rate

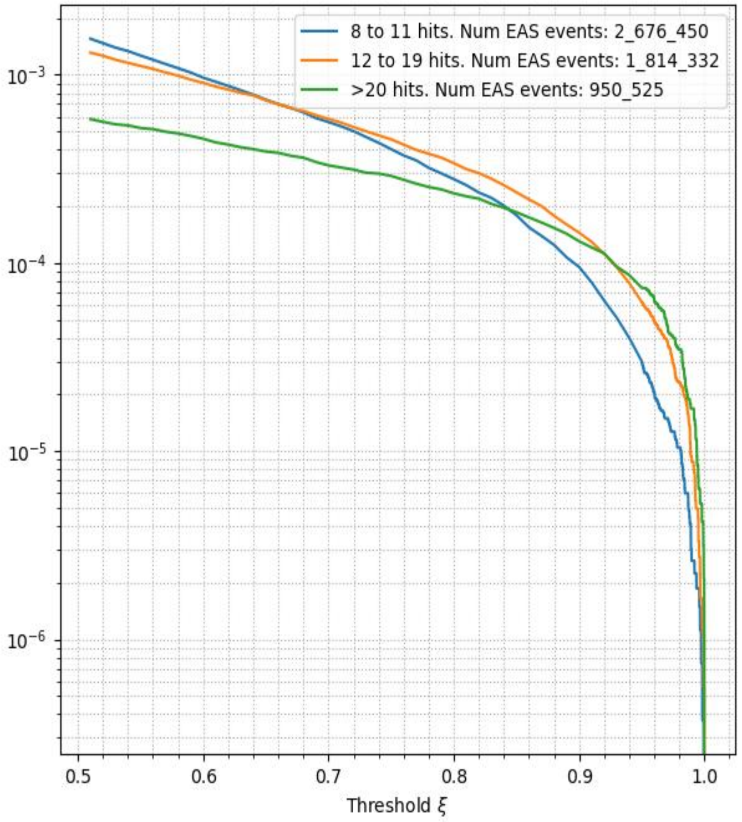


Metrics: different hits numbers

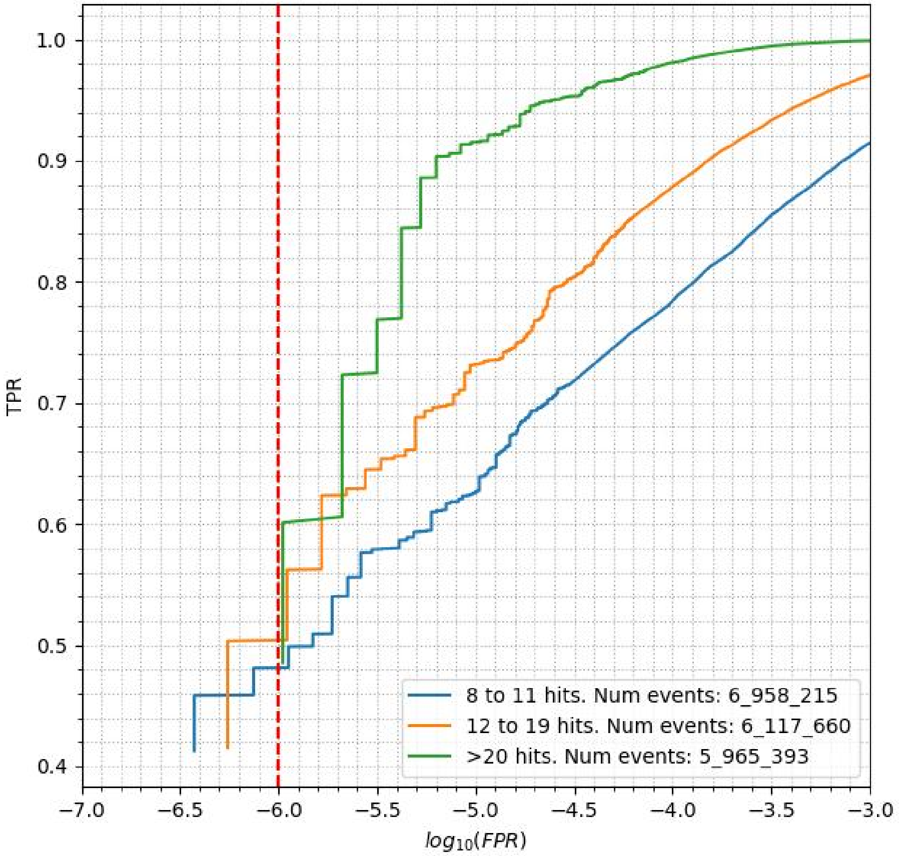
TP rate



FP rate

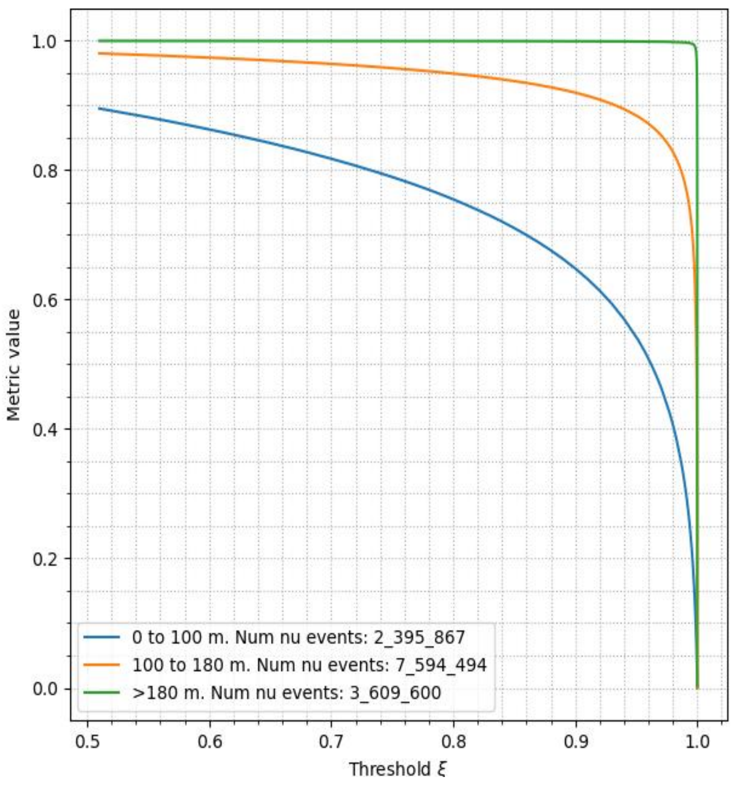


TP rate vs FP rate

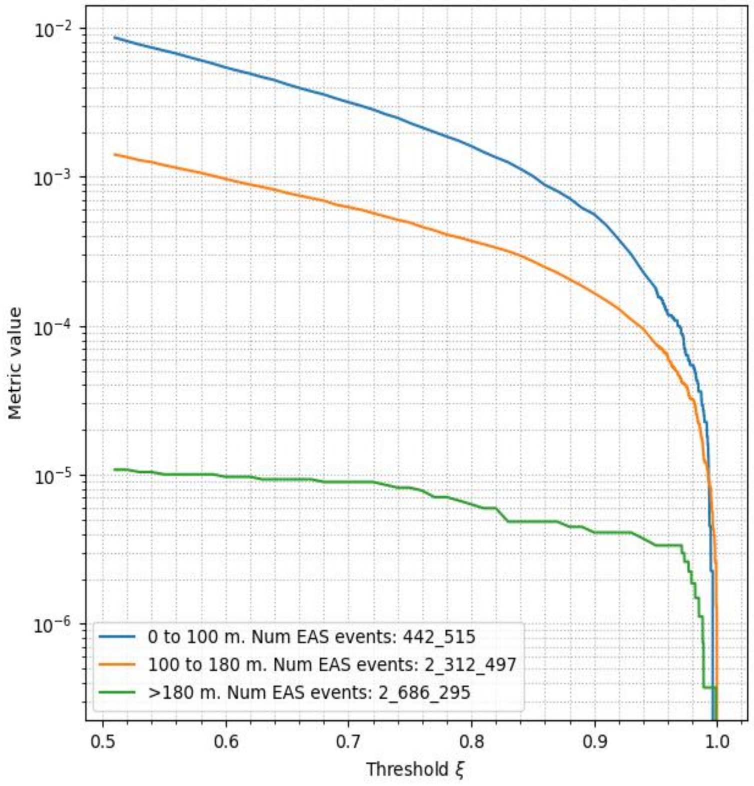


Metrics: different “lengths” of track

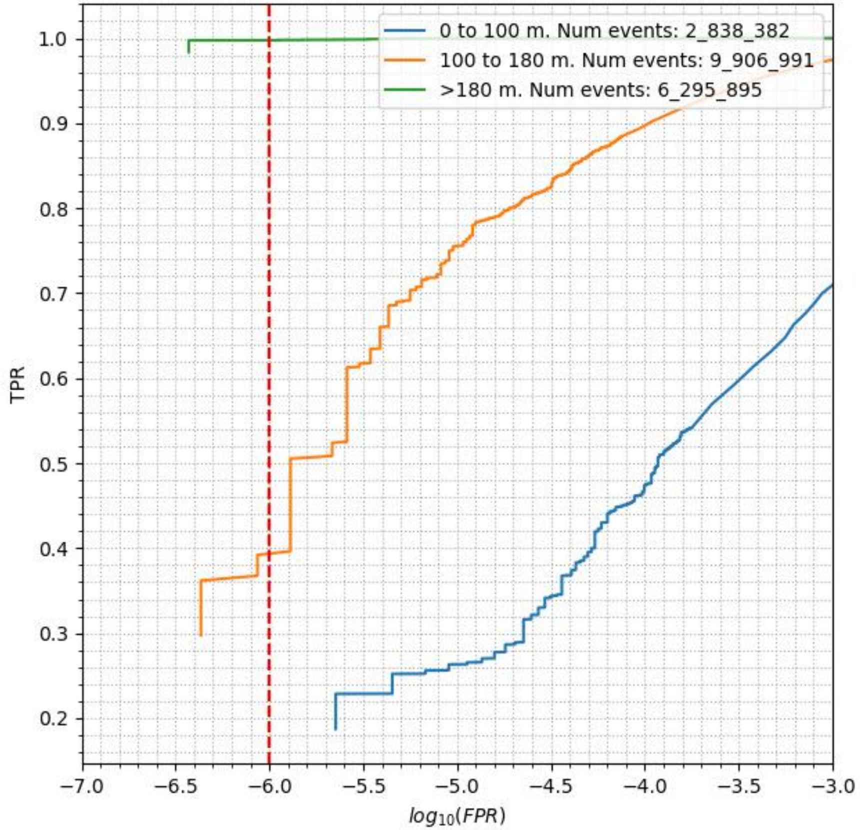
TP rate



FP rate



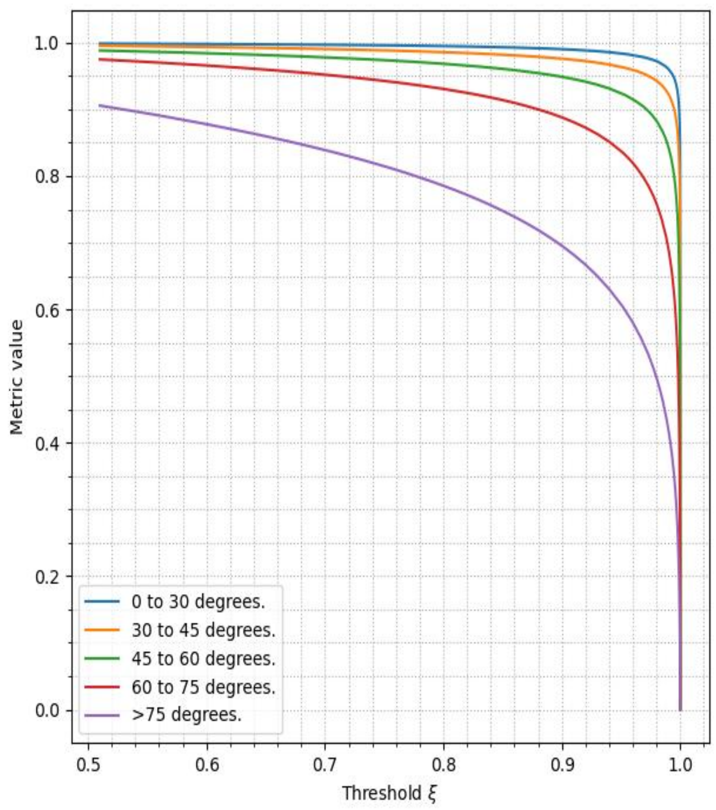
TP rate vs FP rate



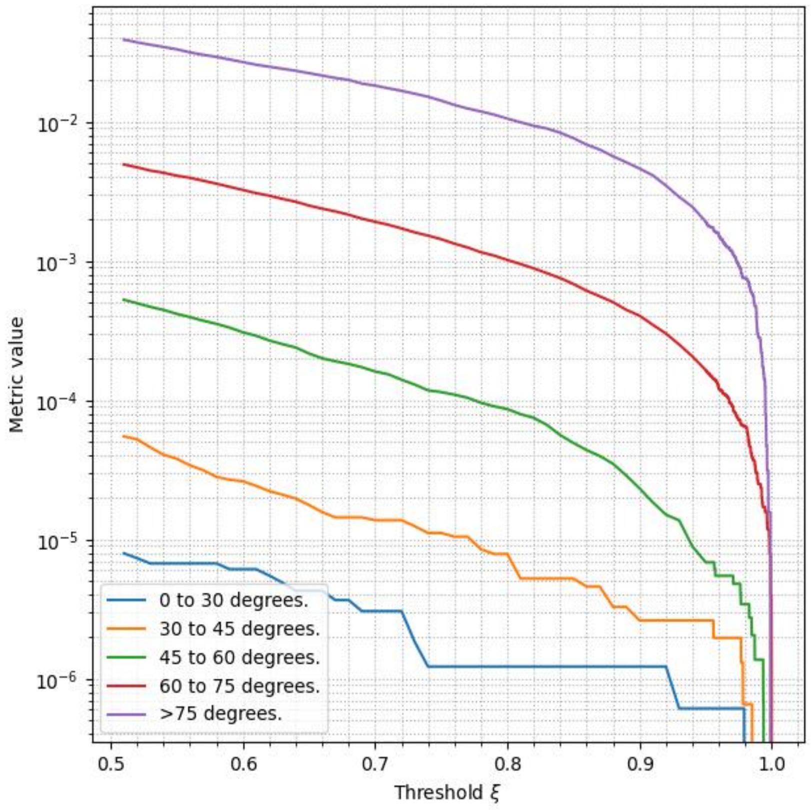
Strong and observable feature!

Metrics: different zenith angles

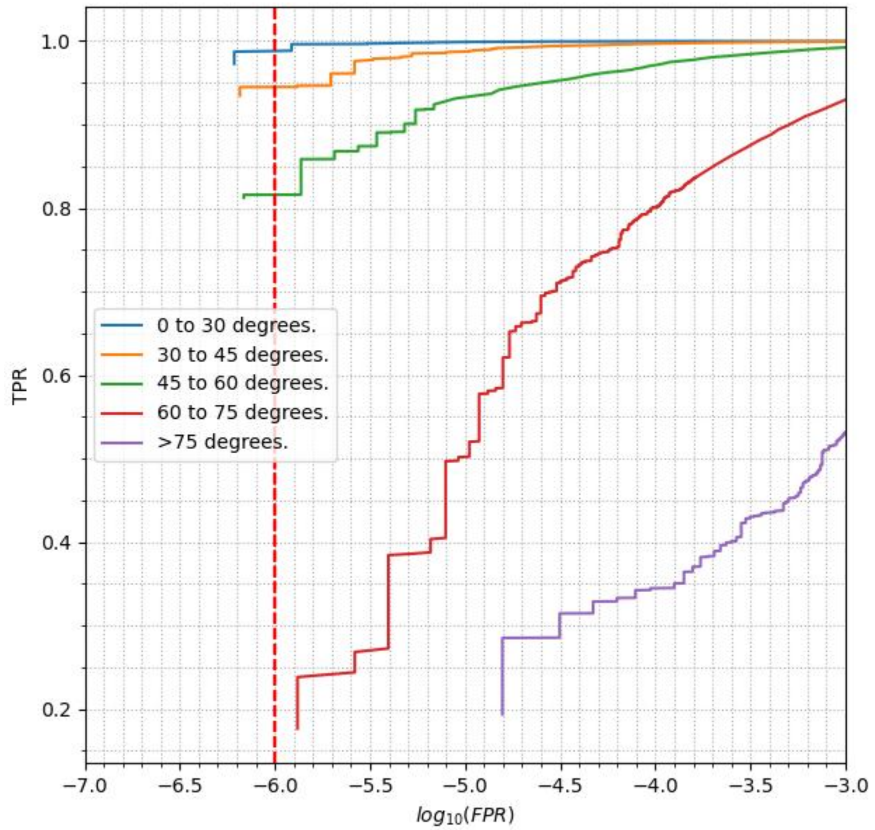
TP rate



FP rate



TP rate vs FP rate



Sub-conclusion

- It is possible to select ~50% of neutrinos against the background of the EAS flow
- Selection quality may be measured separately for events with different track length
- Possible uses:

1) create “clean” catalogs of neutrino events (with indicated background fraction)

2) quickly filter out the background, reducing the amount of calculation for reconstruction algorithms

3)Knowing TPR and FPR, one can estimate the integral neutrino flux

III. Neutrino energy reconstruction

Motivation

- Energy is an important parameter of a particle: the spectrum of astrophysical neutrinos can tell a lot about the sources

- Current reconstruction error: factor from 3 to 5

We want to improve the reconstruction quality using neural networks

Dataset

- Monte-Carlo (again)

- Only ν_μ - track events
  We reconstruct the muon's energy E!
  since it is a directly observable particle

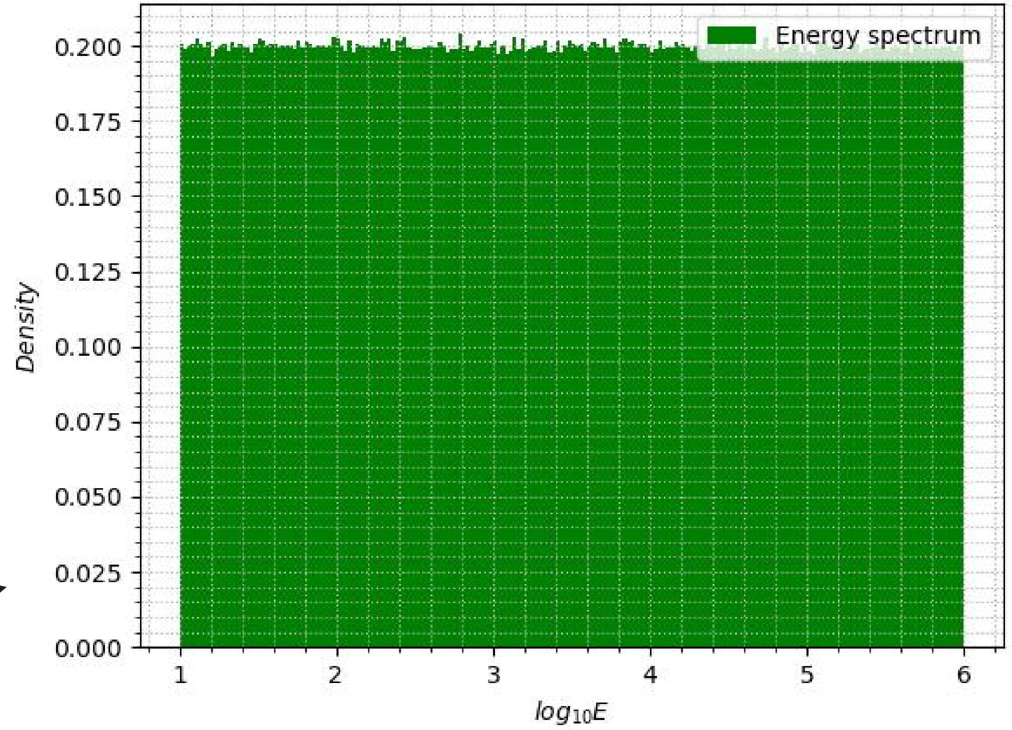
- Cuts:
  1) min 8 hits
  2) min 2 strings
  3) E from 10 to 10^6 GeV

- Target feature: $log_{10}E$

- A **uniform spectrum** was selected

Splitting a Dataset

	train	test	validation
Число ν	≈$1.5*10^6$	≈10^5	≈$5*10^5$



Energy spectrum in the dataset

Loss function

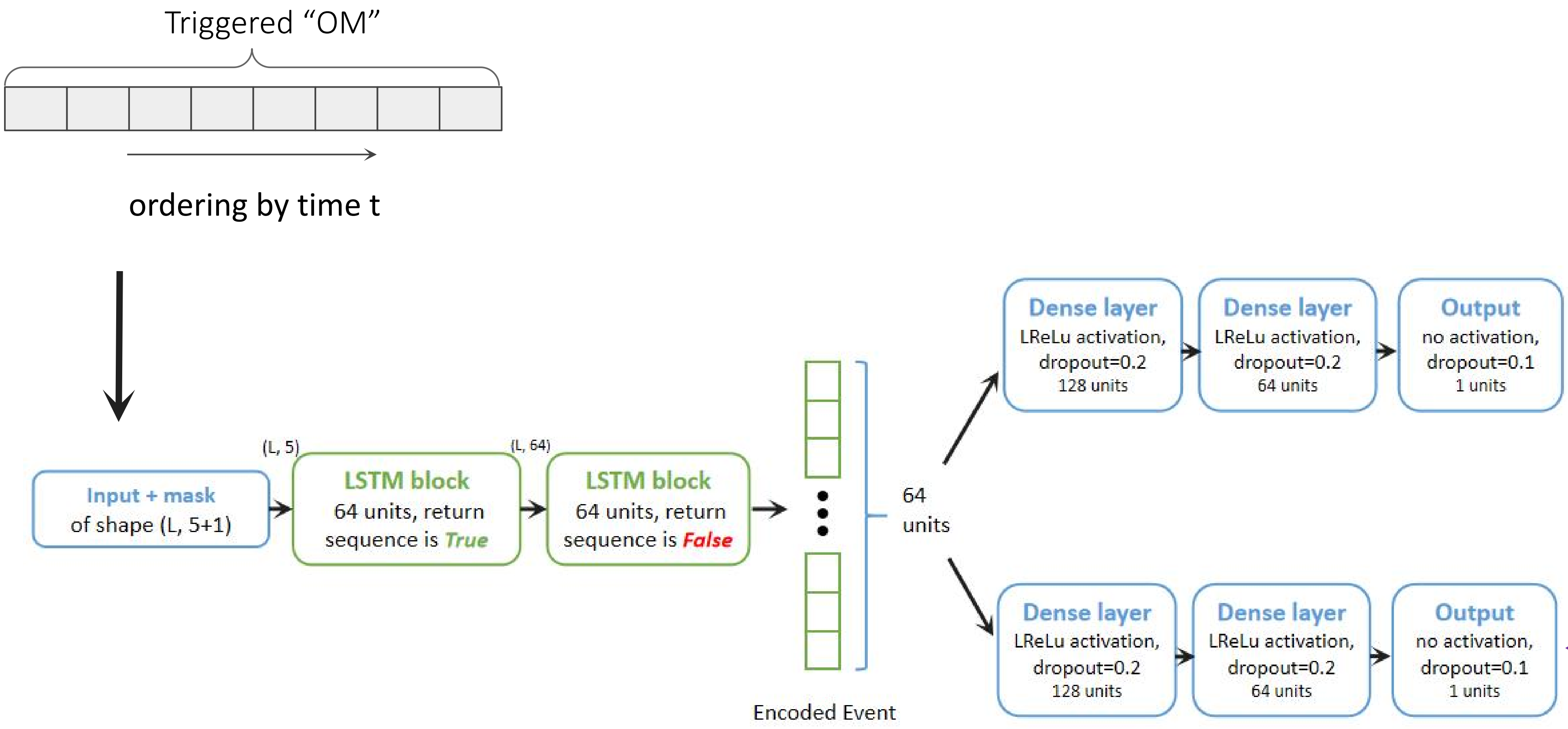
- We want to make the network to predict the error of $logE$!
  Let's denote: σ

- We use a special type of ***loss***:

$$loss \;= \frac{1}{n}\sum_{i=1}^{n}\left(ln(\sigma_i^2) + \frac{(logE_i - logE_{true_i})^2}{\sigma_i^2}\right)$$

Maximizes the likelihood of a hypothesis: $logE_{true} \sim N(logE,\ \sigma)$.

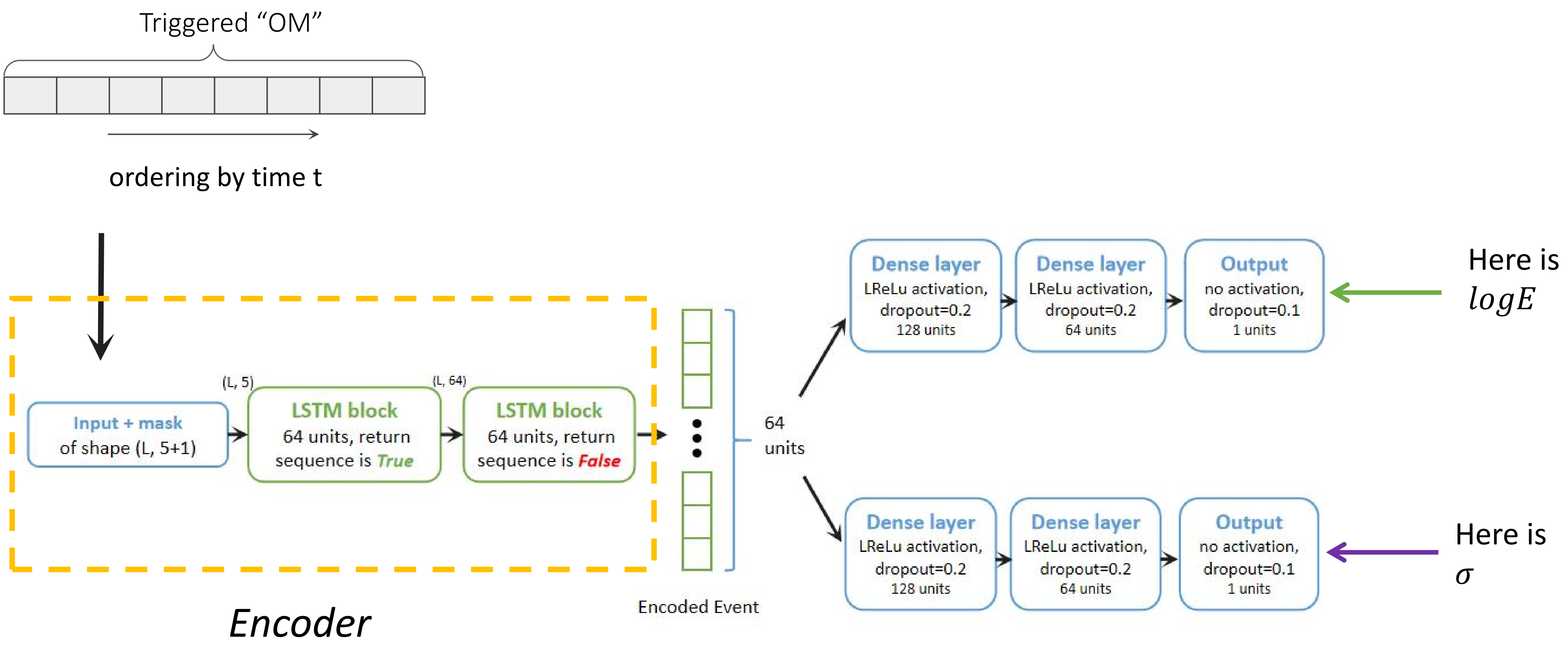
The network



Here is $logE$

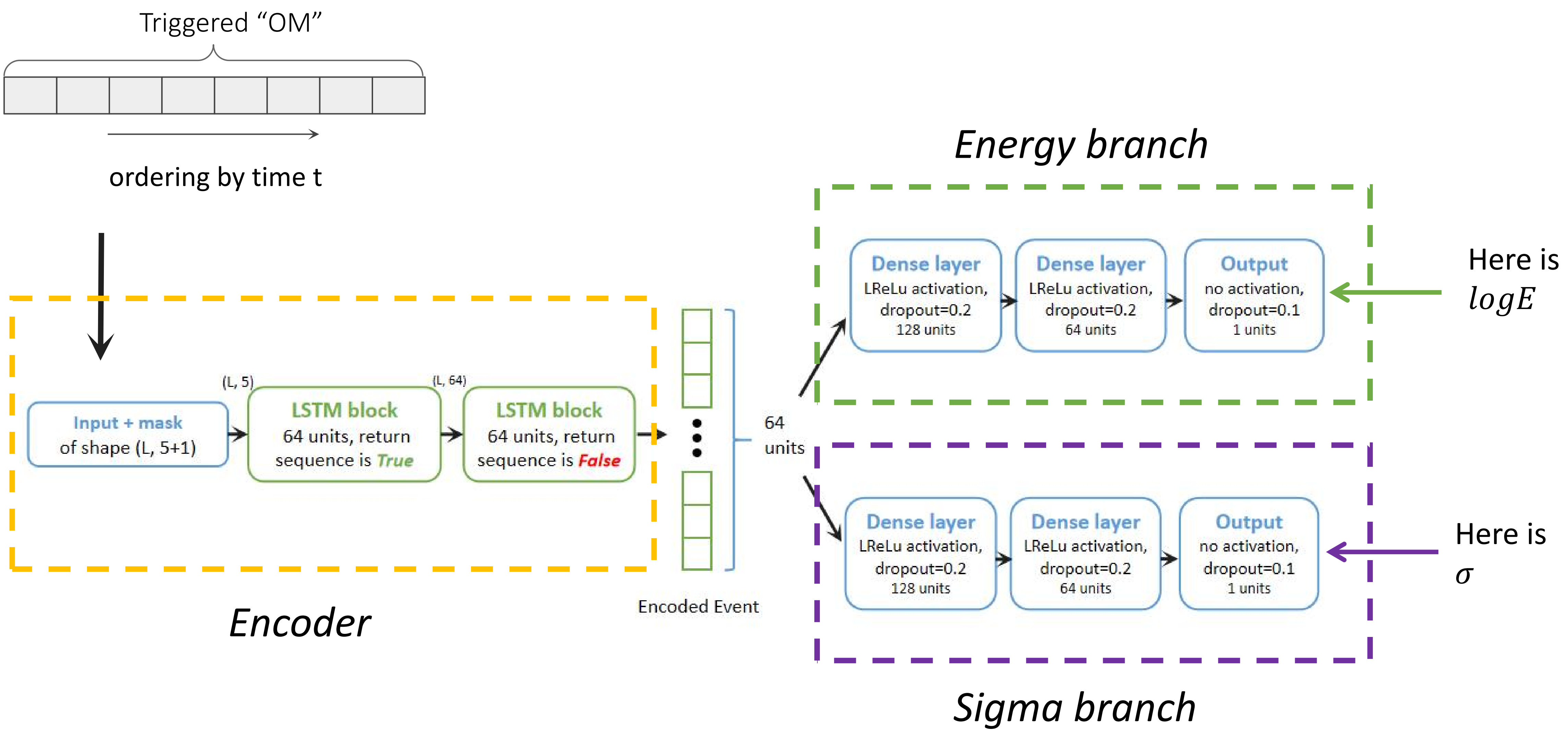
Here is σ

The network



Triggered "OM"
ordering by time t
Input + mask
of shape (L, 5+1)
(L, 5)
LSTM block
64 units, return sequence is True
(L, 64)
LSTM block
64 units, return sequence is False
Encoder
64 units
Encoded Event
Dense layer
LReLu activation, dropout=0.2
128 units
Dense layer
LReLu activation, dropout=0.2
64 units
Output
no activation, dropout=0.1
1 units
Here is $logE$
Dense layer
LReLu activation, dropout=0.2
128 units
Dense layer
LReLu activation, dropout=0.2
64 units
Output
no activation, dropout=0.1
1 units
Here is σ

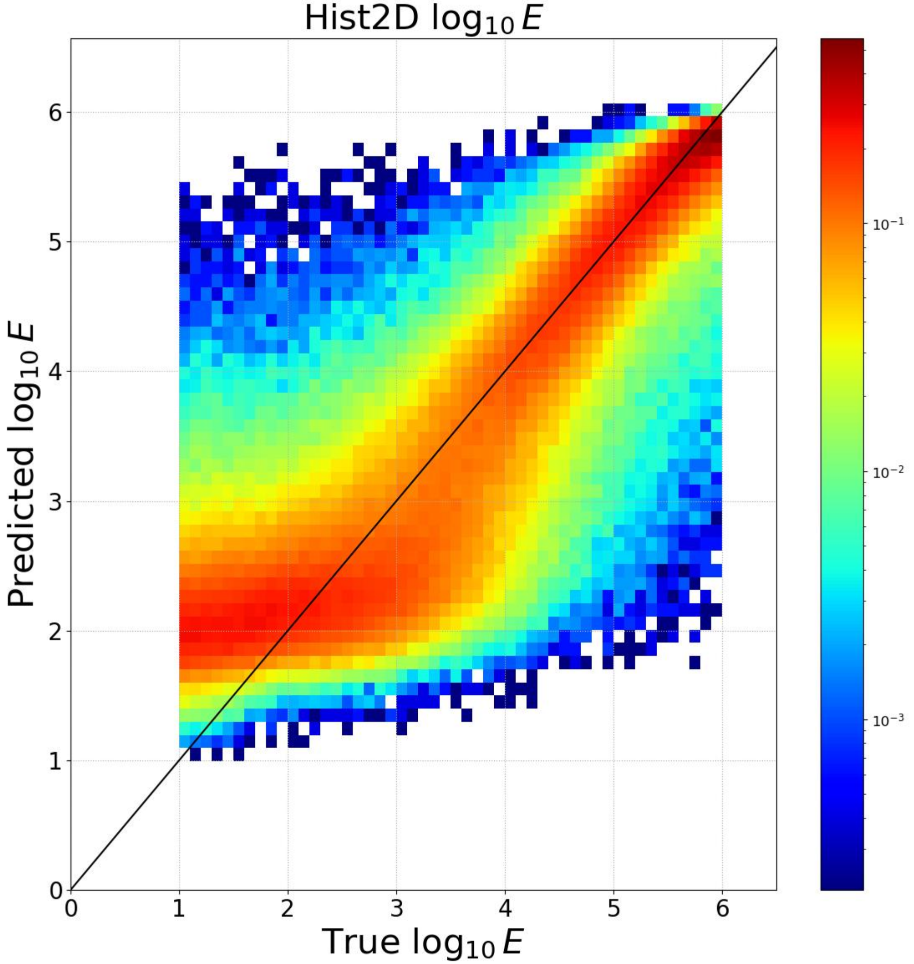
The network



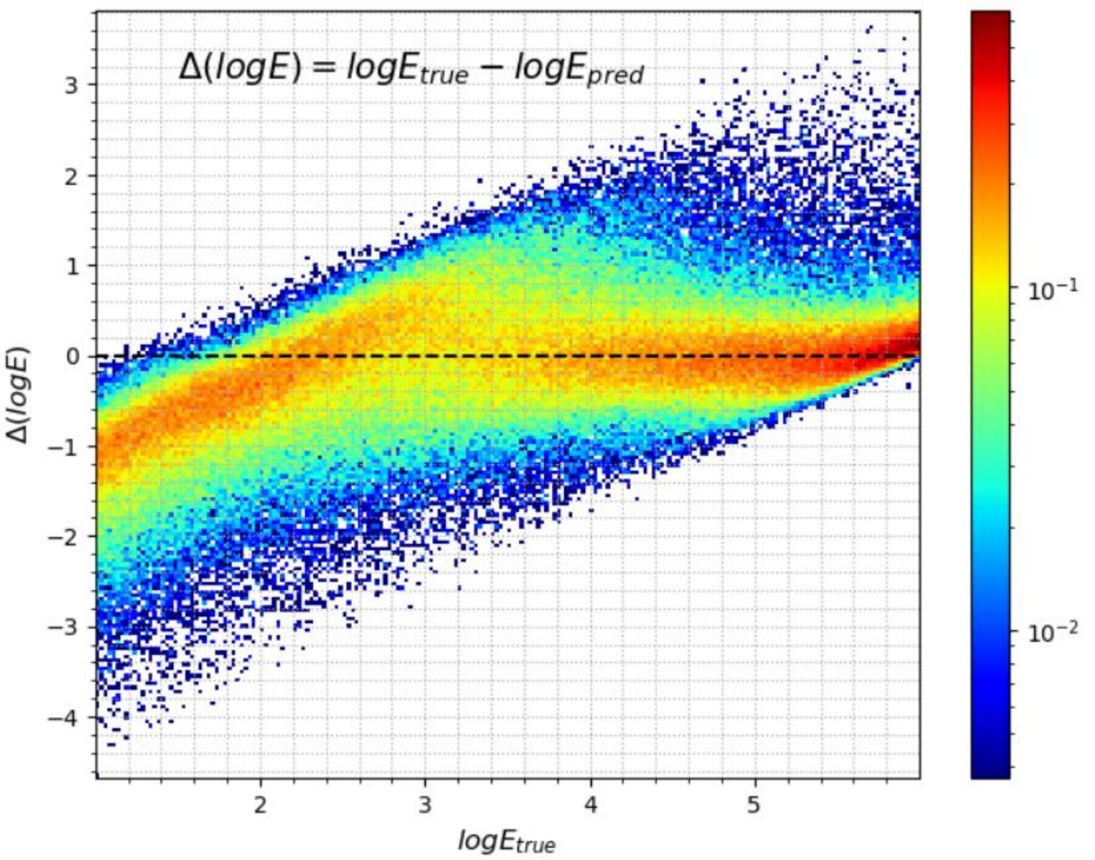
Triggered “OM”
ordering by time t
Energy branch
Here is
$logE$
Input + mask
of shape (L, 5+1)
(L, 5)
LSTM block
64 units, return
sequence is True
(L, 64)
LSTM block
64 units, return
sequence is False
64
units
Dense layer
LReLu activation,
dropout=0.2
128 units
Dense layer
LReLu activation,
dropout=0.2
64 units
Output
no activation,
dropout=0.1
1 units
Dense layer
LReLu activation,
dropout=0.2
128 units
Dense layer
LReLu activation,
dropout=0.2
64 units
Output
no activation,
dropout=0.1
1 units
Here is
σ
Encoder
Encoded Event
Sigma branch

Metrics: energy branch, [E] = [GeV]

$logE$ and $logE_{true}$ correspondence

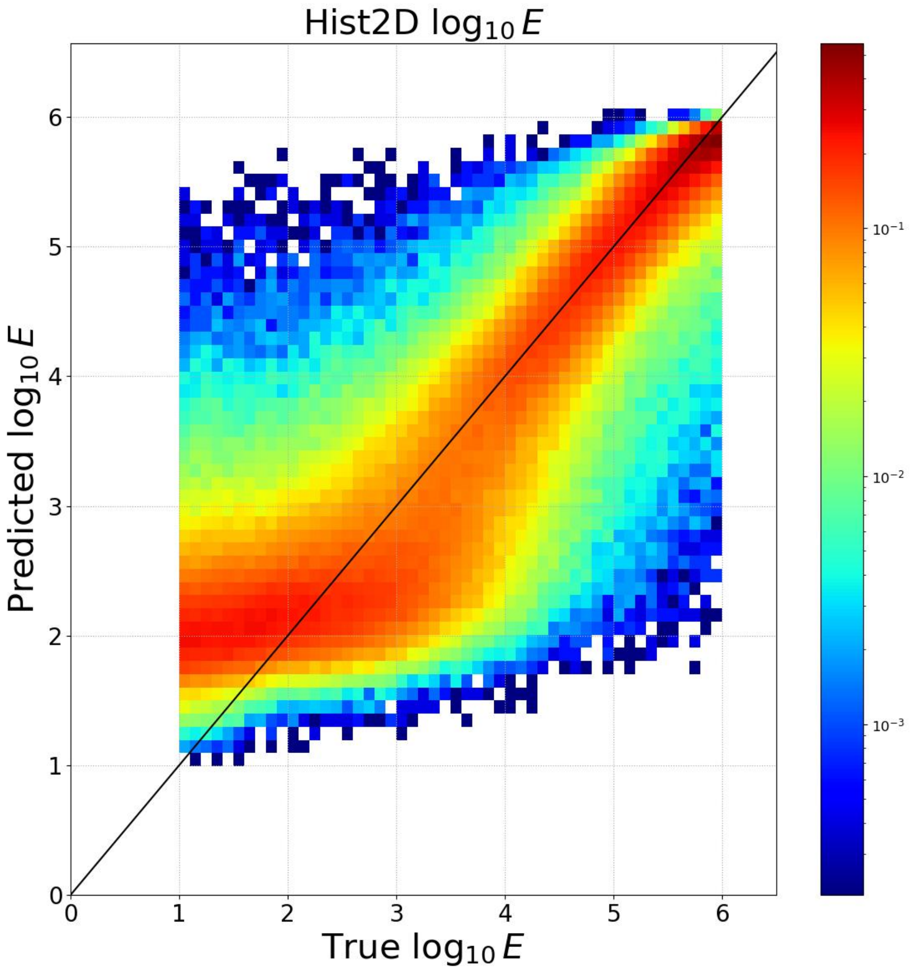


Histogram of $logE$ error

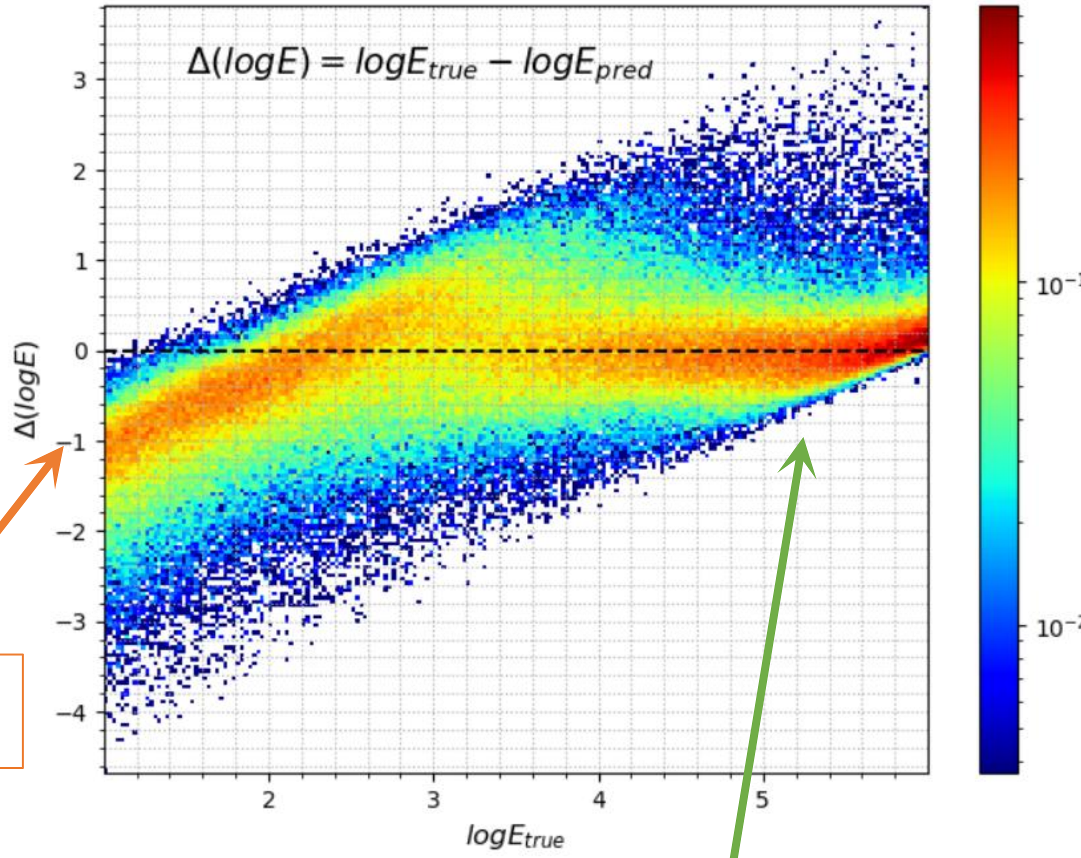


Metrics: energy branch, [E] = [GeV]

$logE$ and $logE_{true}$ correspondence



Histogram of $logE$ error



Increases low true energies

Quality improves with increasing E

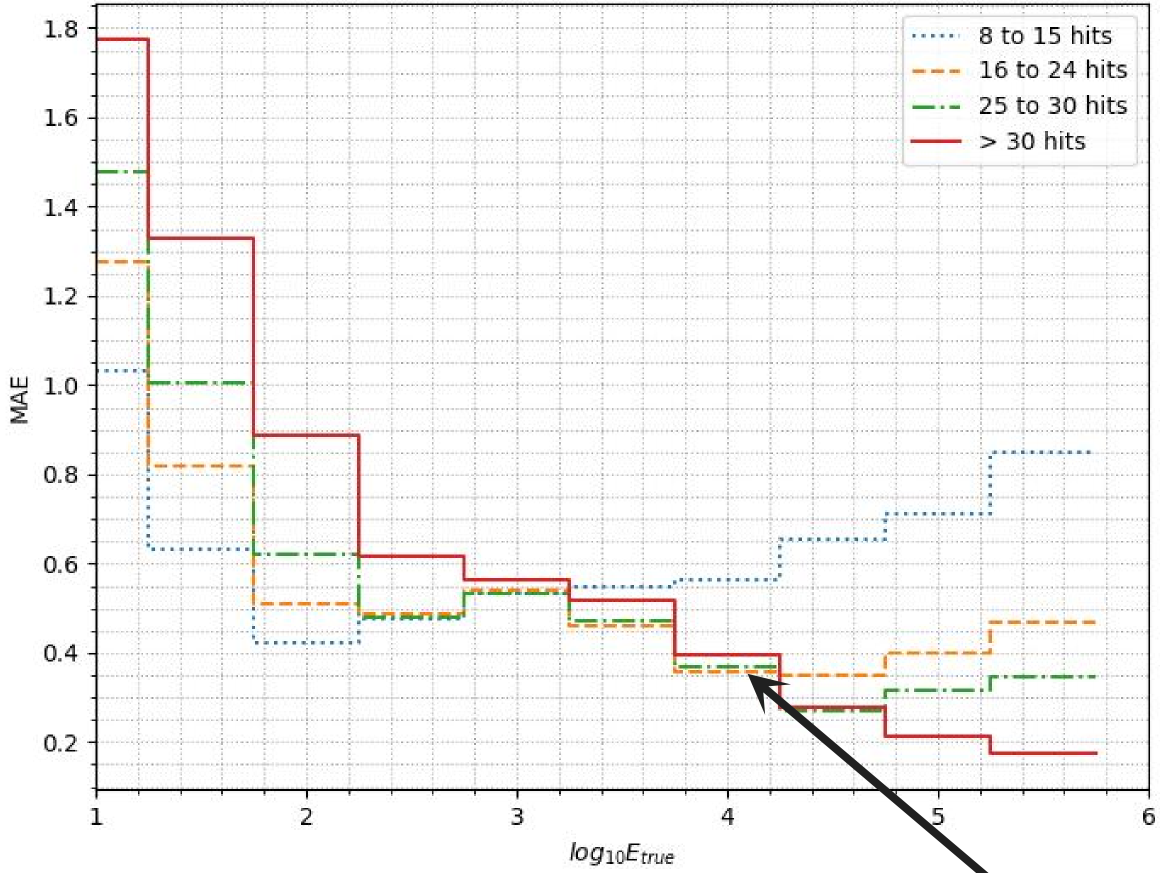
Metrics: energy branch, [E] = [GeV]

Different hits numbers

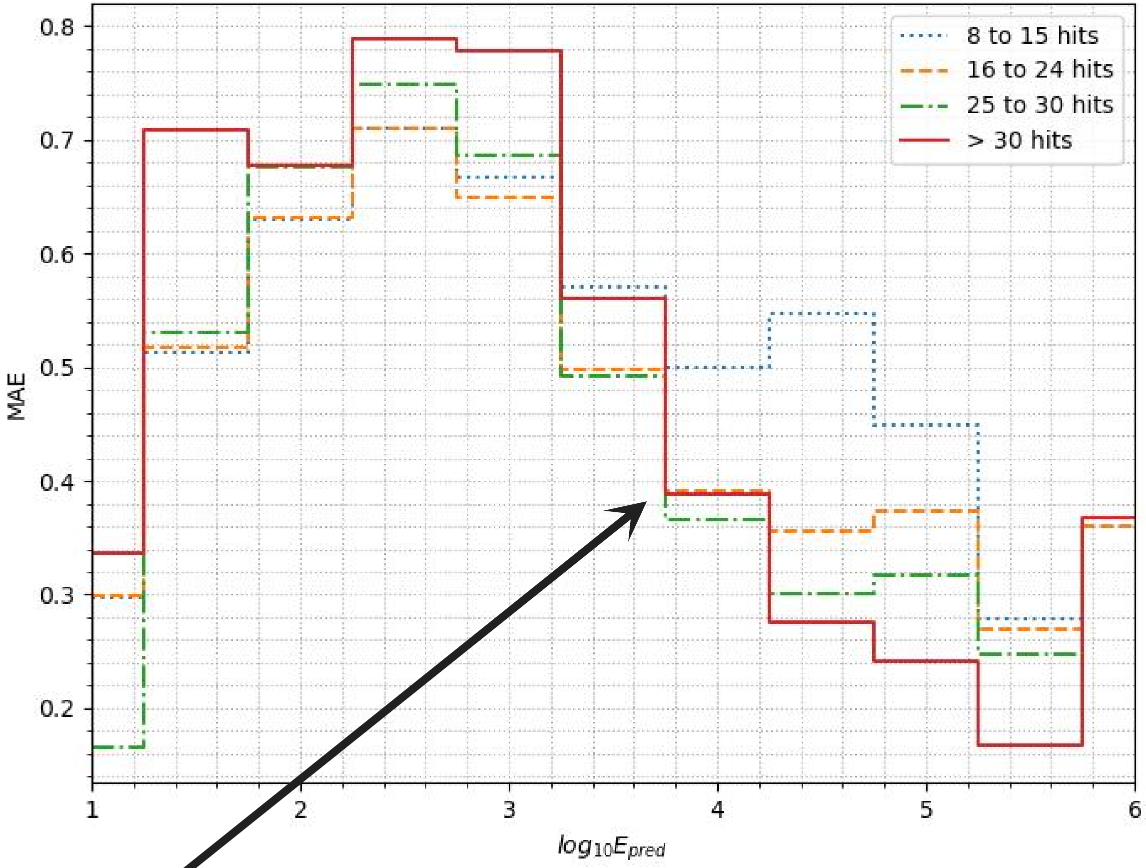
MAE

$$y = \frac{1}{n}\sum |\, logE - logE_{true} \,|$$

MAE vs true Energy



MAE vs predicted Energy



Error factor ~2.5

Metrics: σ branch

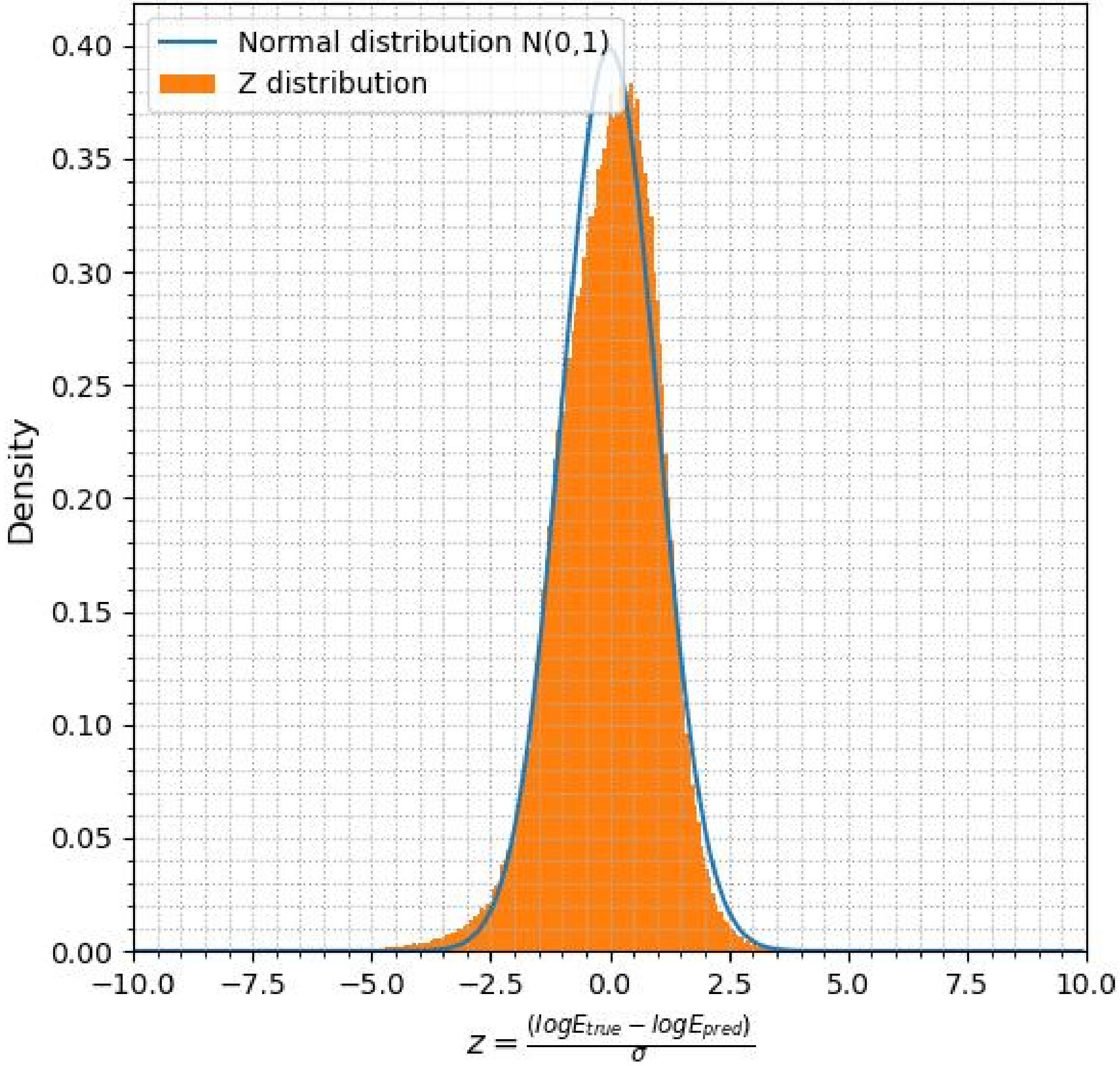
Quality criteria:

1) z = $\frac{logE_{true} - logE}{\sigma} \sim N(0,1)$

2) central quantile $\pm\,\sigma$ = 68%

3) σ and Δ(logE) correlation

Standardized score z distribution



Metrics: σ branch

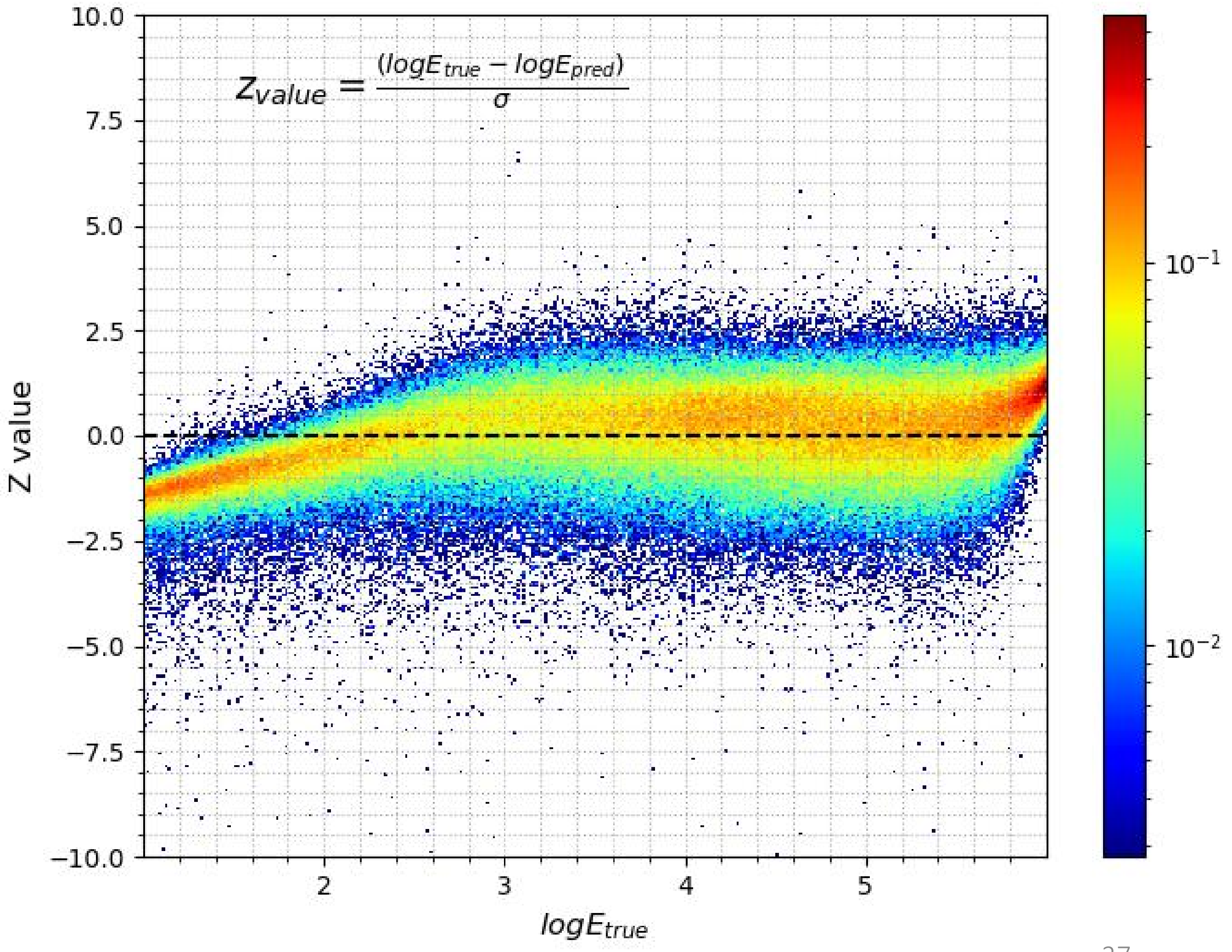
Quality criteria:

1) $z = \frac{logE_{true} - logE}{\sigma} \sim N(0,1)$

2) central quantile $\pm\,\sigma$ = 68%

3) σ and Δ(logE) correlation

Standardized score z 2D distribution



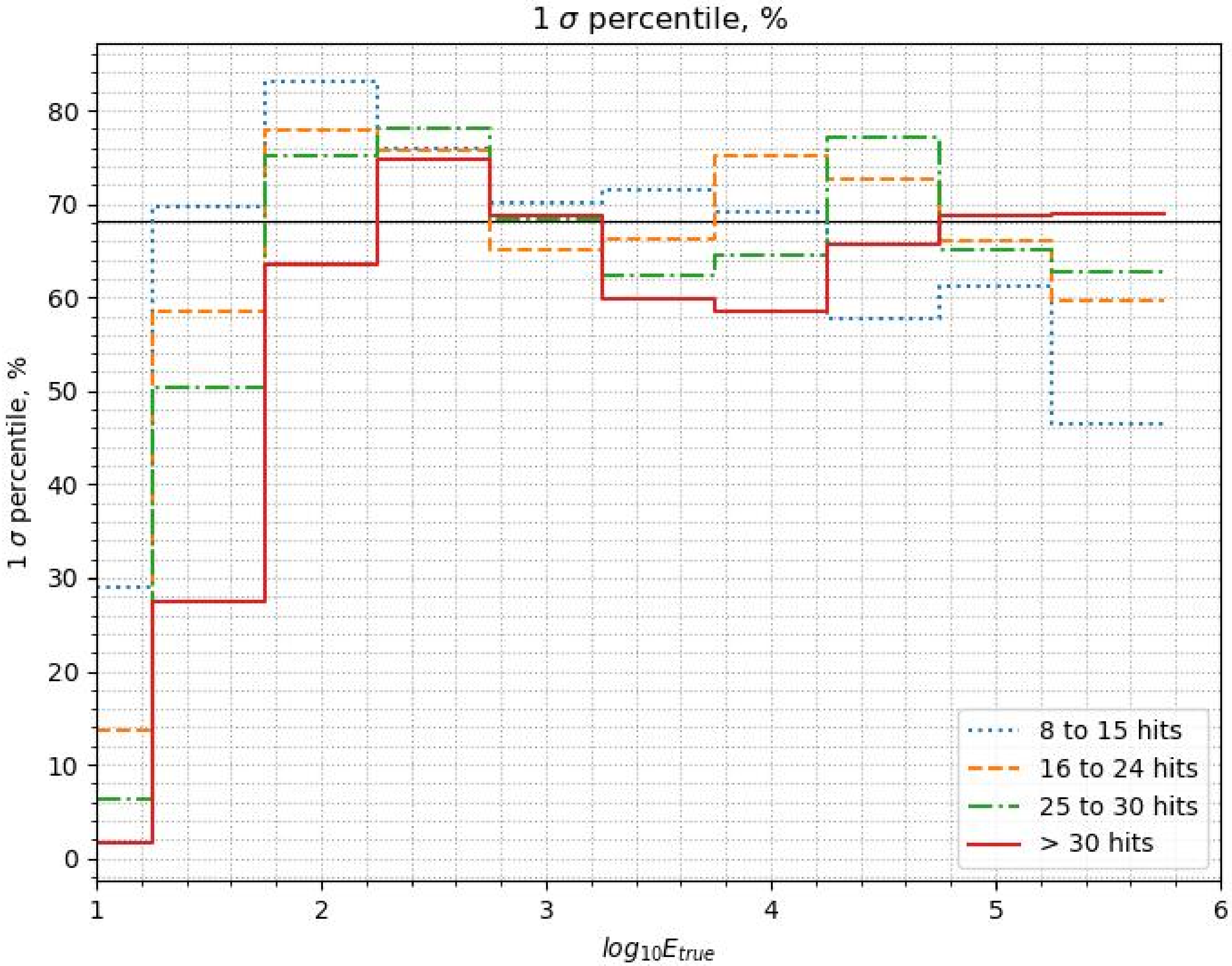
Metrics: σ branch

Quality criteria:

1) z = $\frac{logE_{true} - logE}{\sigma} \sim N(0,1)$

2) central quantile $\pm\,\sigma$ = 68%

3) σ and Δ(logE) correlation



Metrics: σ branch

Quality criteria:

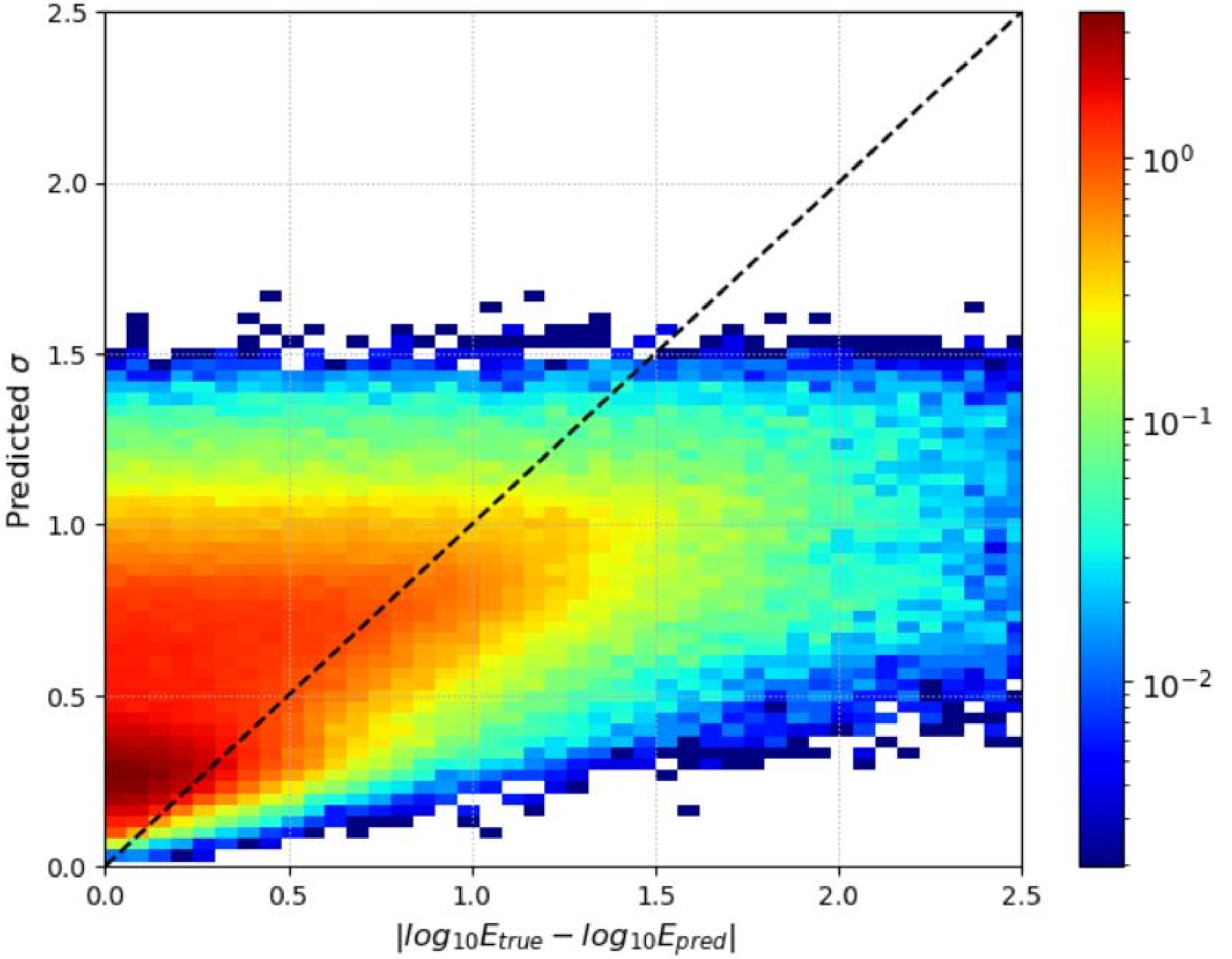
1) z = $\frac{logE_{true} - logE}{\sigma} \sim N(0,1)$

2) central quantile ± σ = 68%

3) σ and Δ(logE) correlation

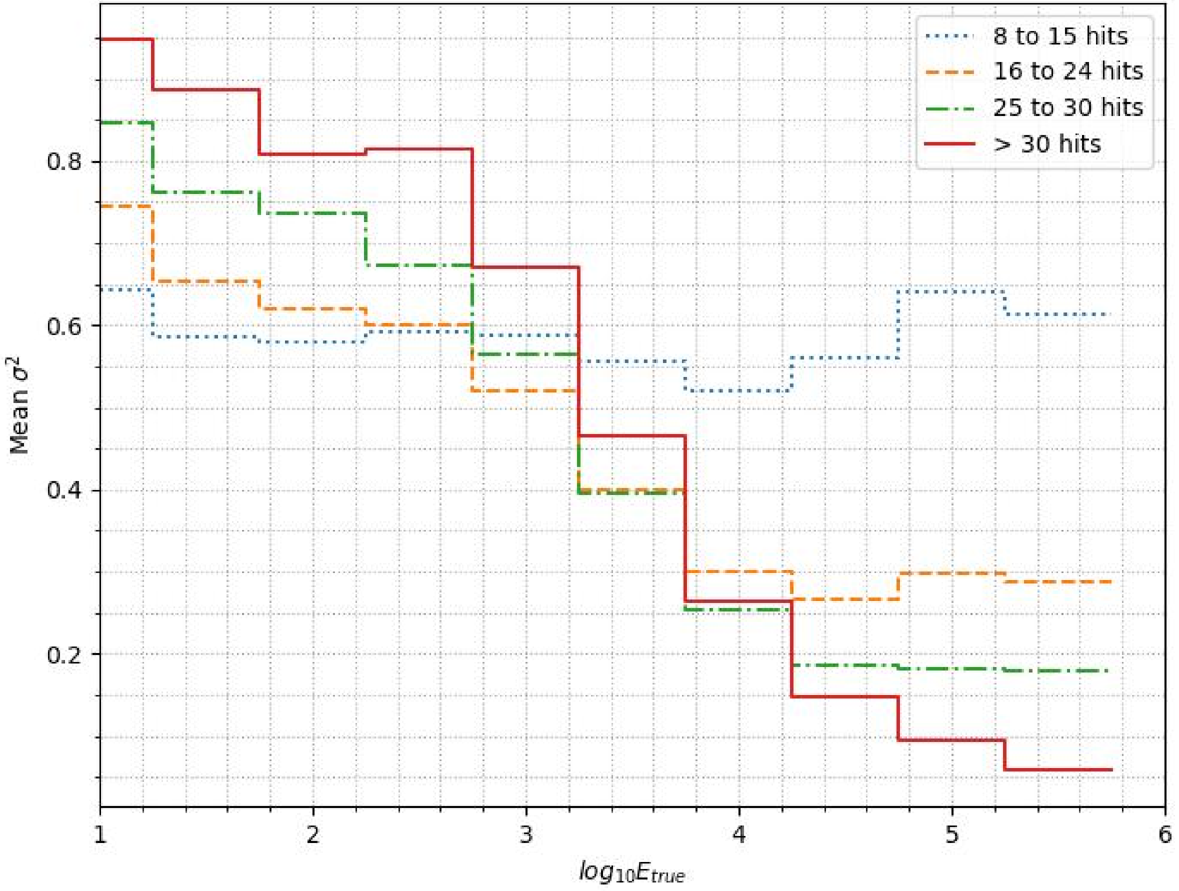
Pearson coefficient = **0.5**
Corresponds to:
$logE_{true} - logE \sim N(0, \sigma)$!

σ vs Δ(logE) 2D histogram

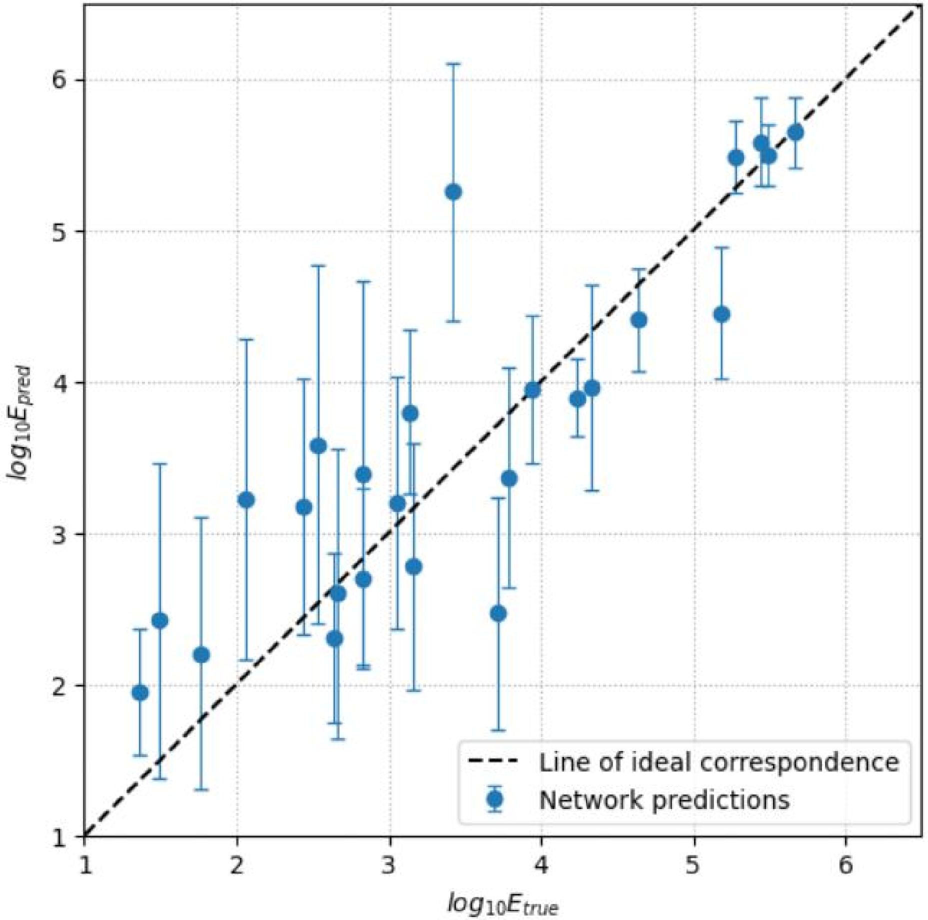


Metrics: σ branch
some more graphs

σ^2 behavior



25 events sample



As energy increases, the predicted error decreases!

Sub-conclusion

Network works fine at **energies ≥1 TeV** and **hits number ≥16**:

- error factor = 3 and less
- predicted sigma corresponds to 1 standard deviation

IV. Conclusion

- Neural networks allow you to:

  - **select 50% of neutrinos**, suppressing the EAS background by 10^6 times

  - **reconstruct E of tracks** no worse than current algorithms and evaluate its own error

- You can apply different cuts (for example, the number of hits) depending on specific tasks

Thanks!

Contacts:

matseiko.av@phystech.edu
t.me/AlbertMac280

Code and models:

github.com/AlbertMatseiko/

- NeutrinoSelection
- NuEnergy

Backup

Общий план применения нейронных сетей

Считываемые данные

↓

Подавление шумов

↓

Выделение нейтринных событий

↓

Реконструкция параметров событий
(энергия, направление)

Общий план применения нейронных сетей

Считываемые данные

↓

Подавление шумов

↓ → стандартная реконструкция

Выделение нейтринных событий

↓ → оценка потока нейтрино

Реконструкция параметров событий
(энергия, направление)

Общий план применения нейронных сетей

Считываемые данные

↓

Подавление шумов

→ стандартная реконструкция

↓

Выделение нейтринных событий

→ оценка потока нейтрино

↓

Предварительная реконструкция ⊕ → Реконструкция параметров событий (энергия, направление)

Технические данные

$\lambda^{eff}_{scatterina} \approx 480\text{м при } 475 \text{ nm}$ $\lambda^{max}_{absorption} \approx 24\text{м}$

Трековые события :точность угла прилета ≈ 0, 25°
Каскадные события: разрешение ≈ 2°

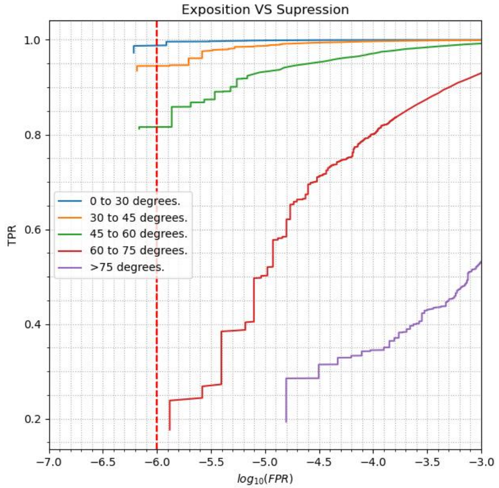
Монте - Карло:

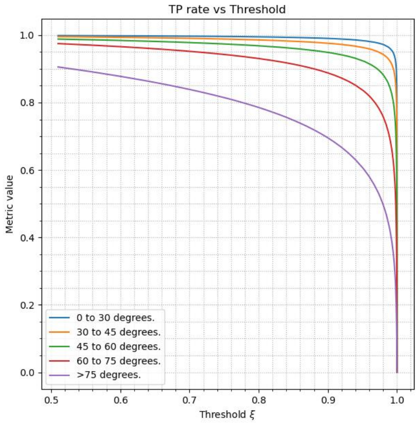
Взаимодействие нейтрино с ядрами : CTEQ4M
(нейтрино с энергиями 10 ГэВ – 100ТэВ
Прилет мюонов: программа CORSIKA 5.7 на модели адронных вз-ий QGSJET
Распространение мюонов до Байкала : MUM v1.3u
Космические лучи: модель на базе KASCADE (240 ГэВ – 20 ПэВ)
Ошибка по времени 5 нс ; 30% по заряду

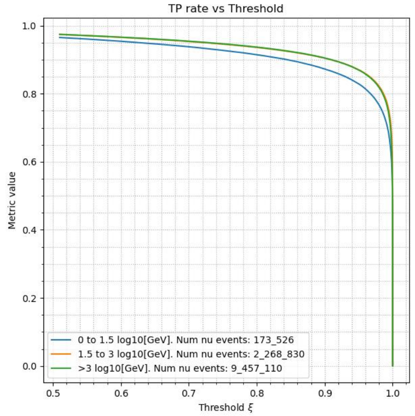
EAS:Nu = 1:1
train: 5556146 events,
test: 465253 events,
val: 22345821 events

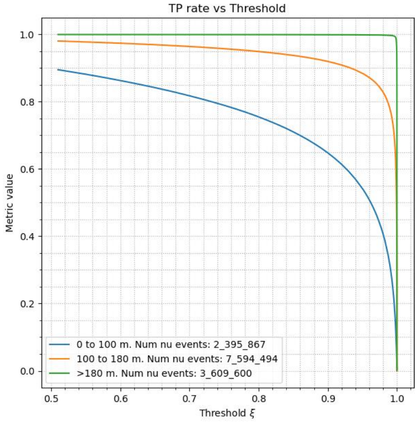
Разные графики

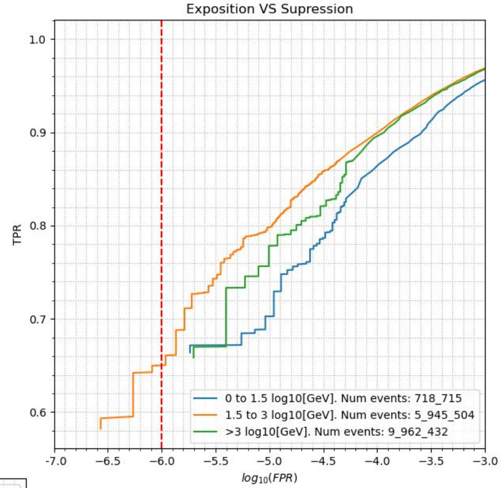
Больше разделения

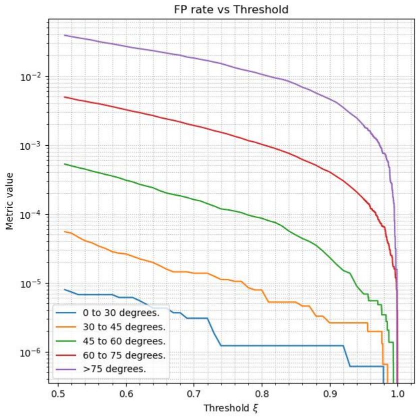


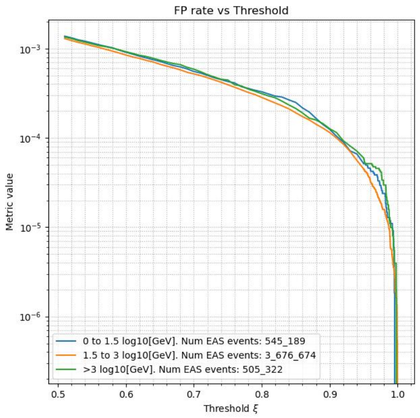


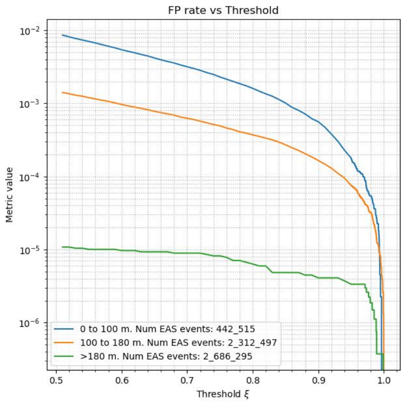


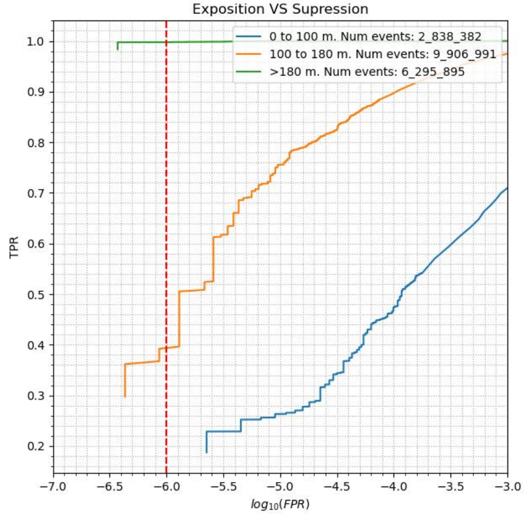




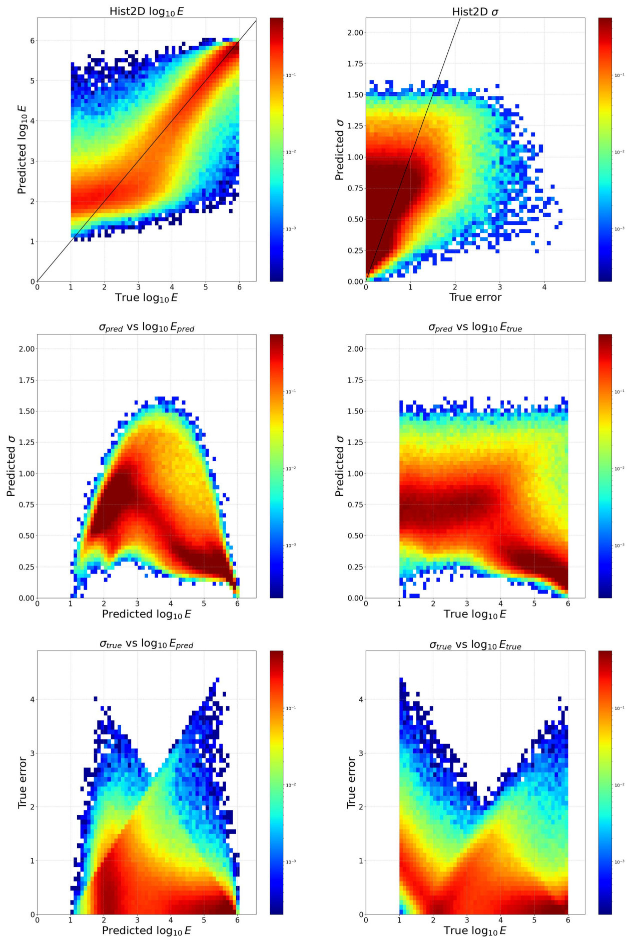


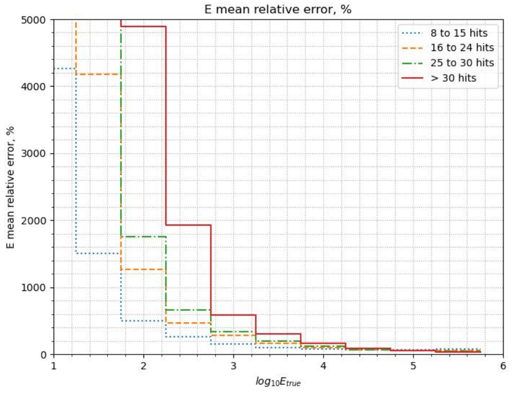


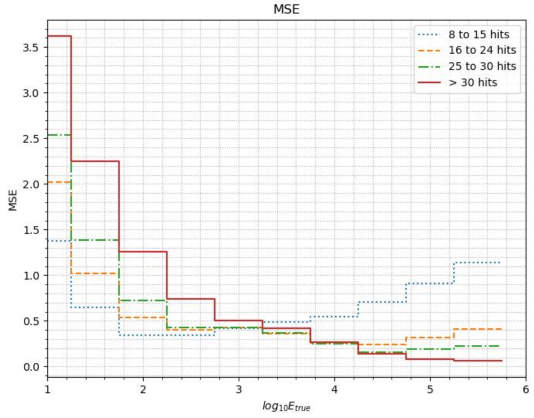


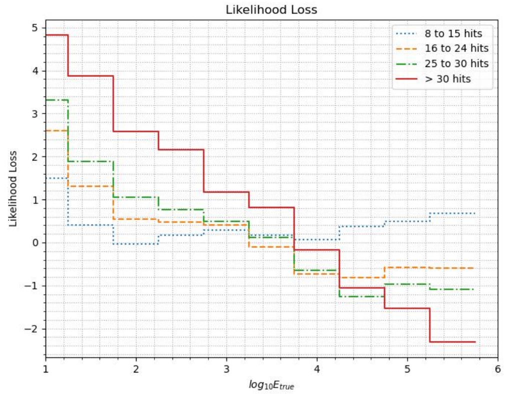


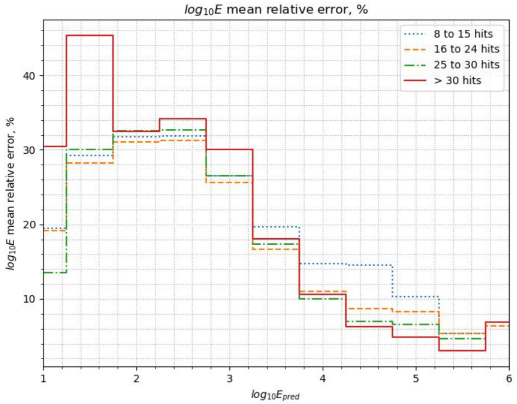
Больше энергии

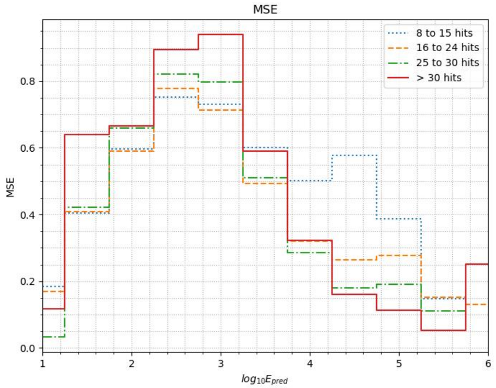


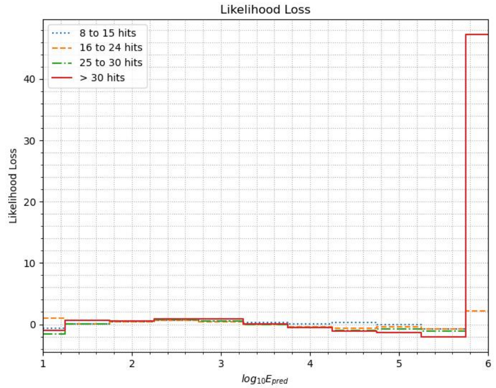






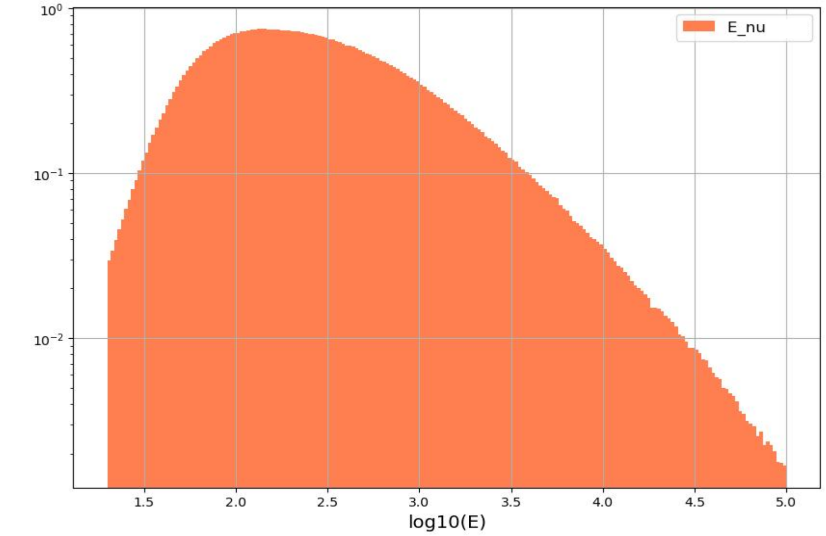


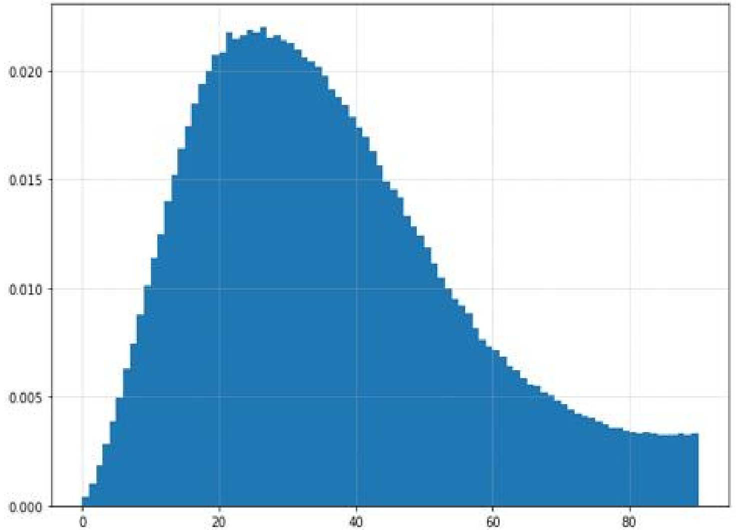




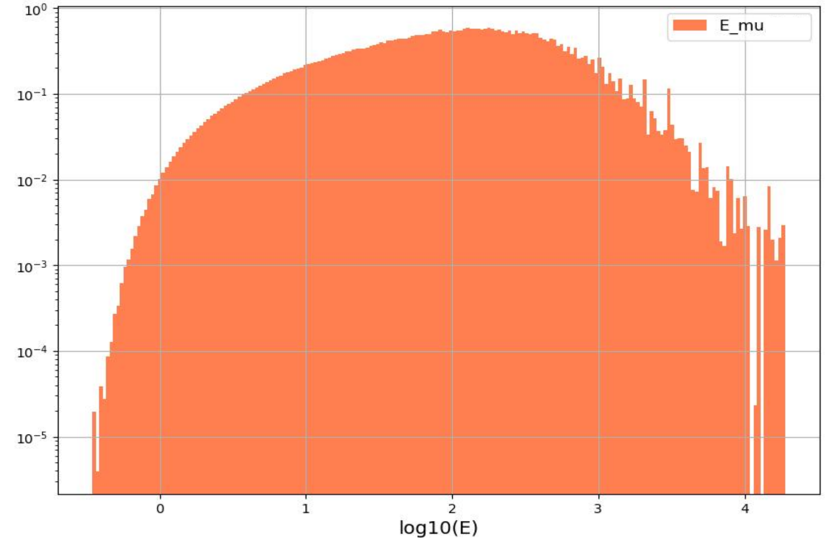
Спектры

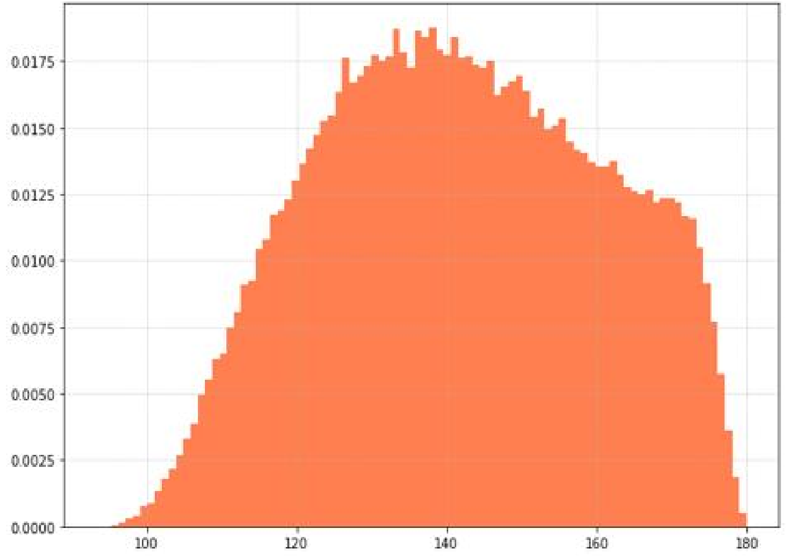
Спектры ν





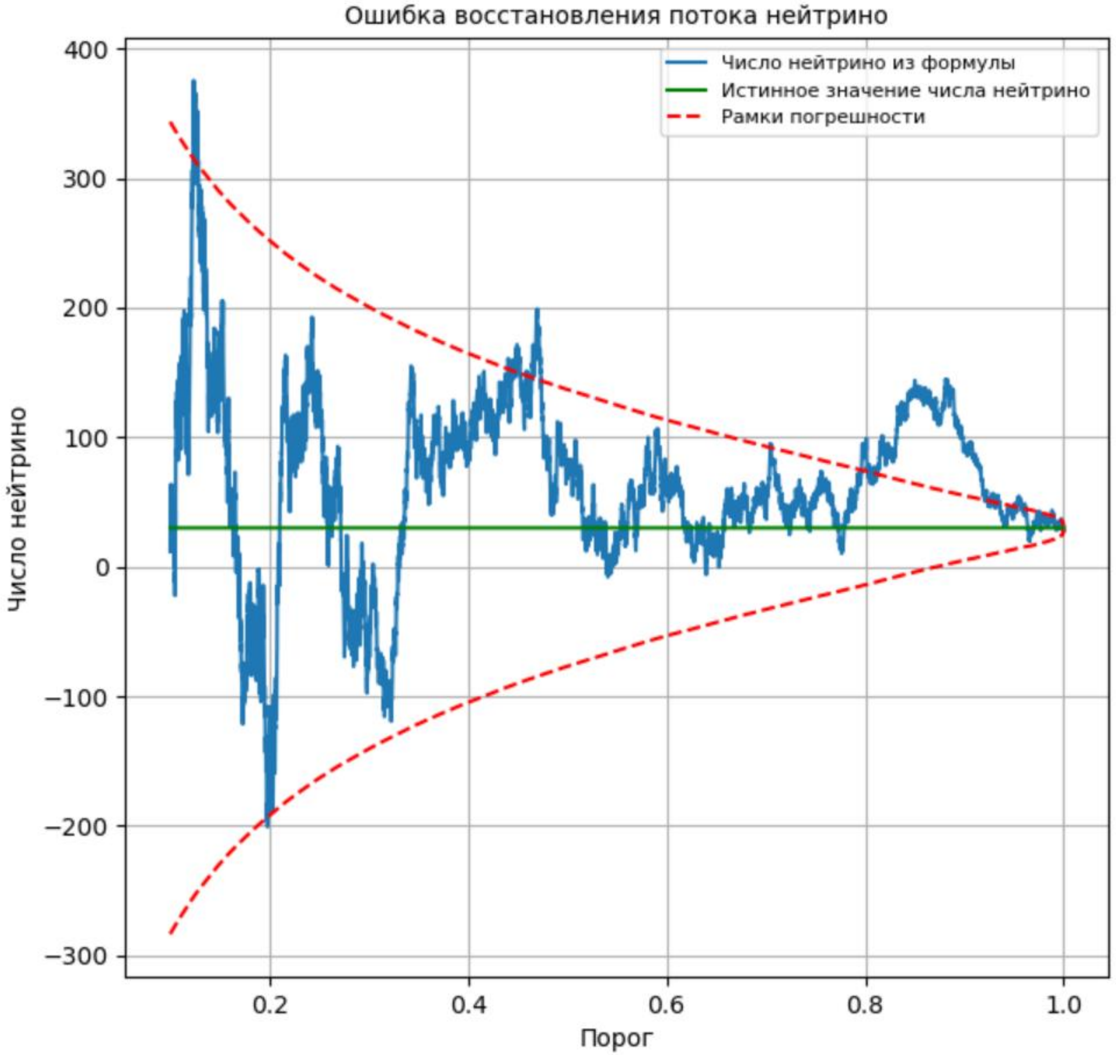
Спектры μ

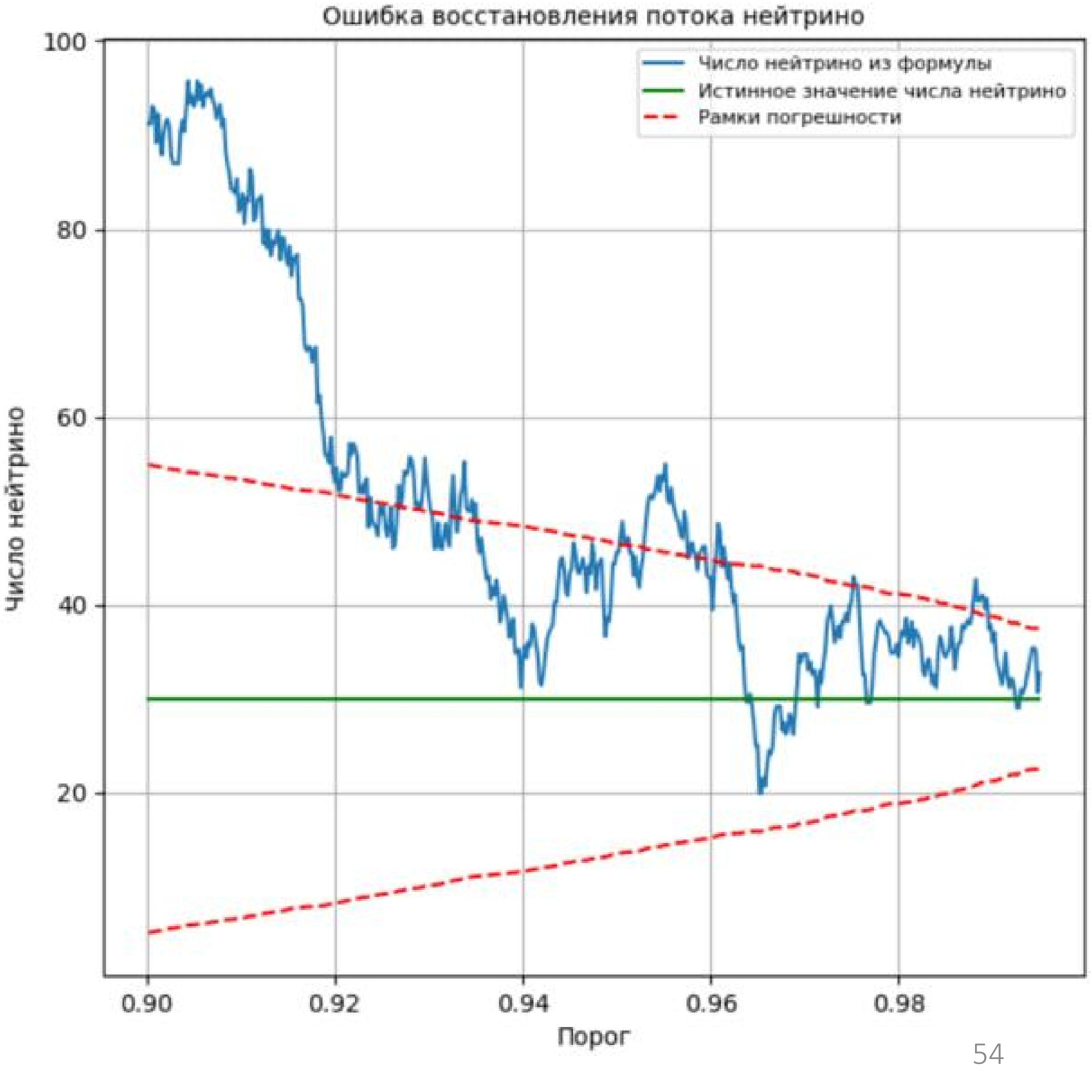




Polar angle

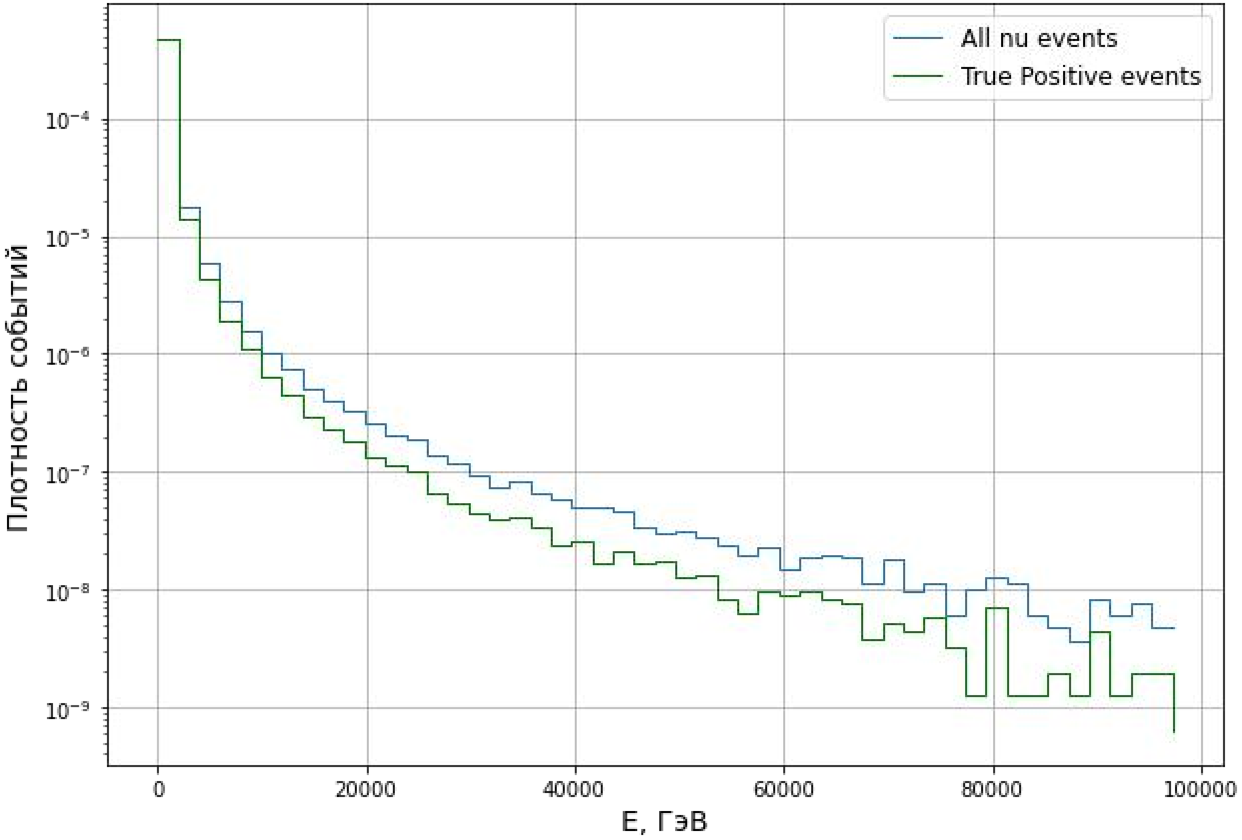
Больше потока



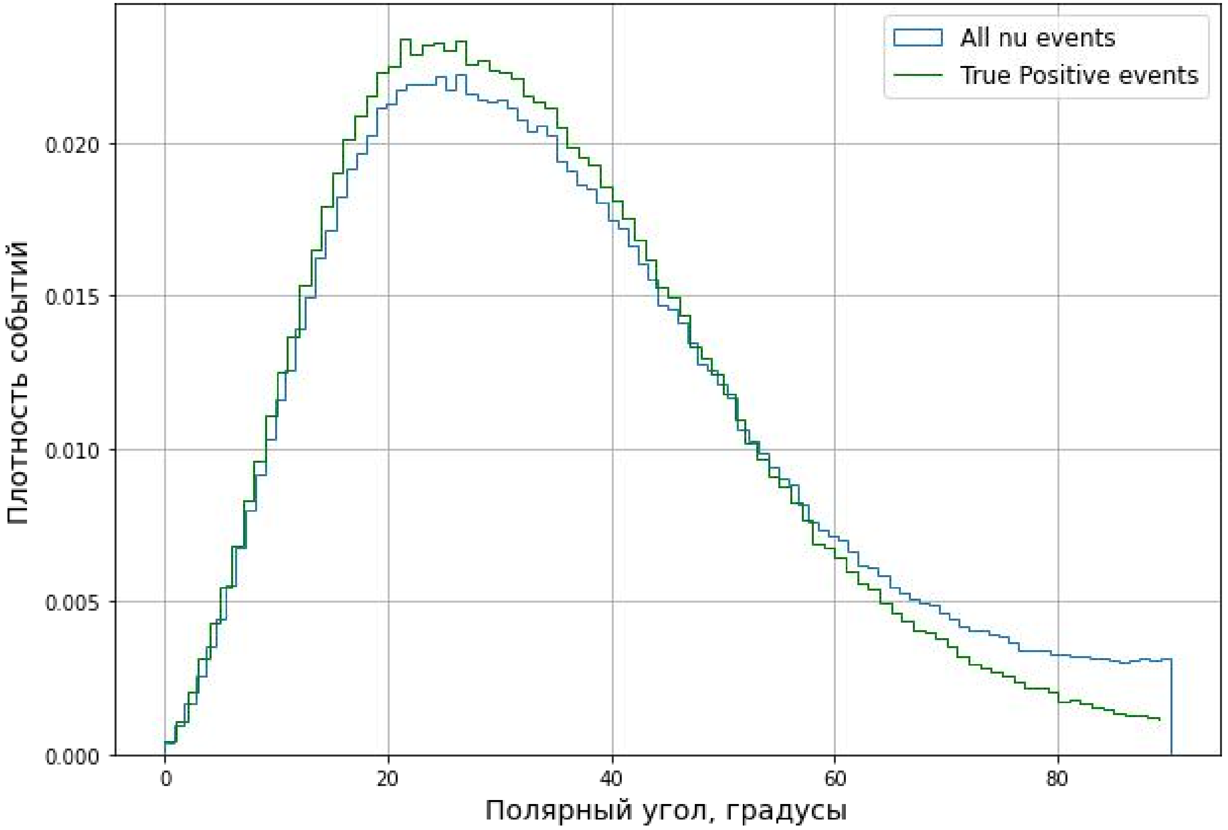


Спектры частиц

Энергия, нейтрино

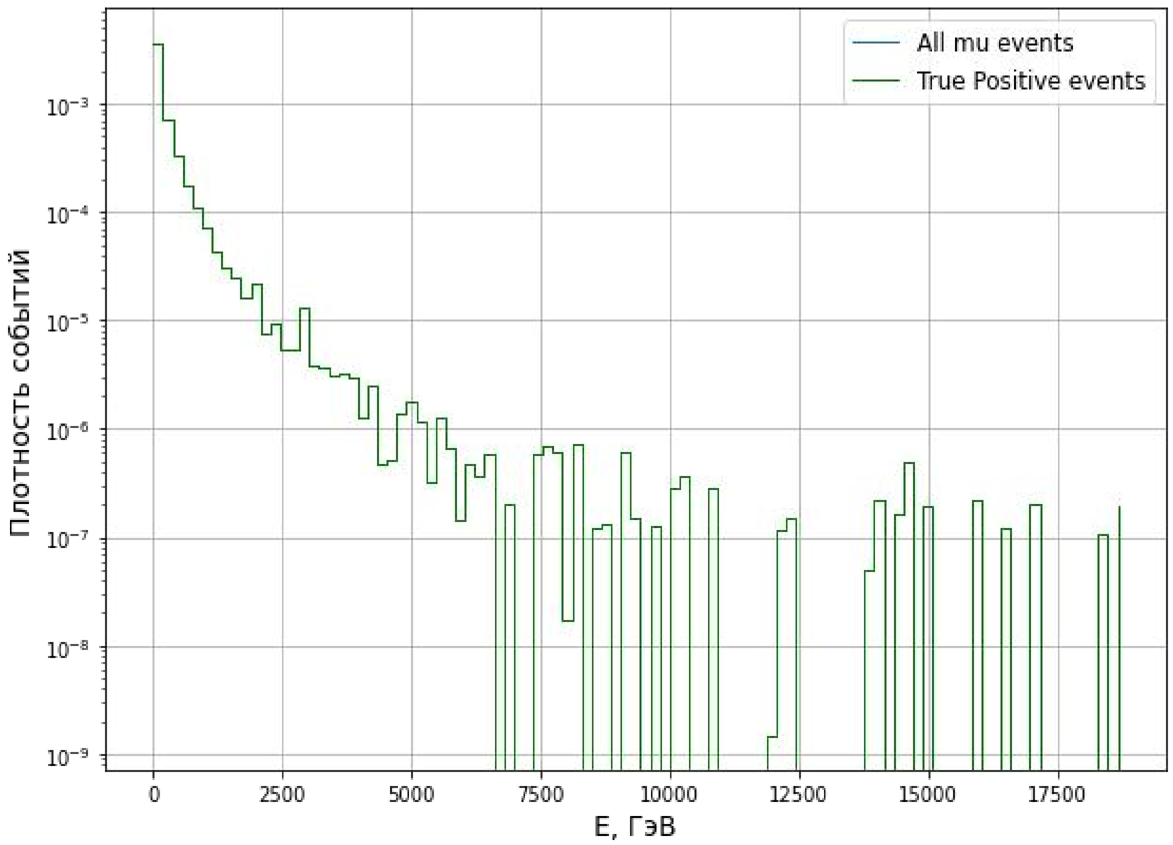


Полярный угол, нейтрино

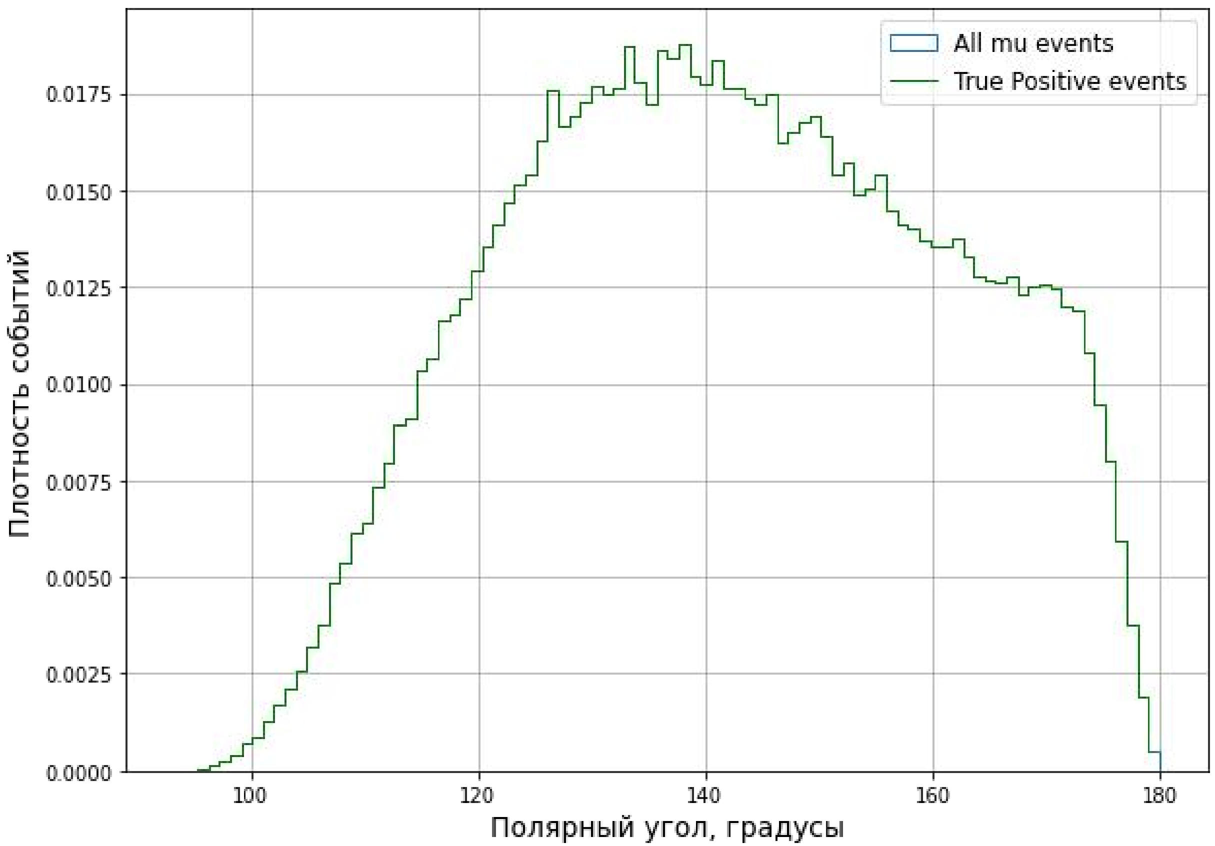


Спектры частиц

Энергия, мюоны

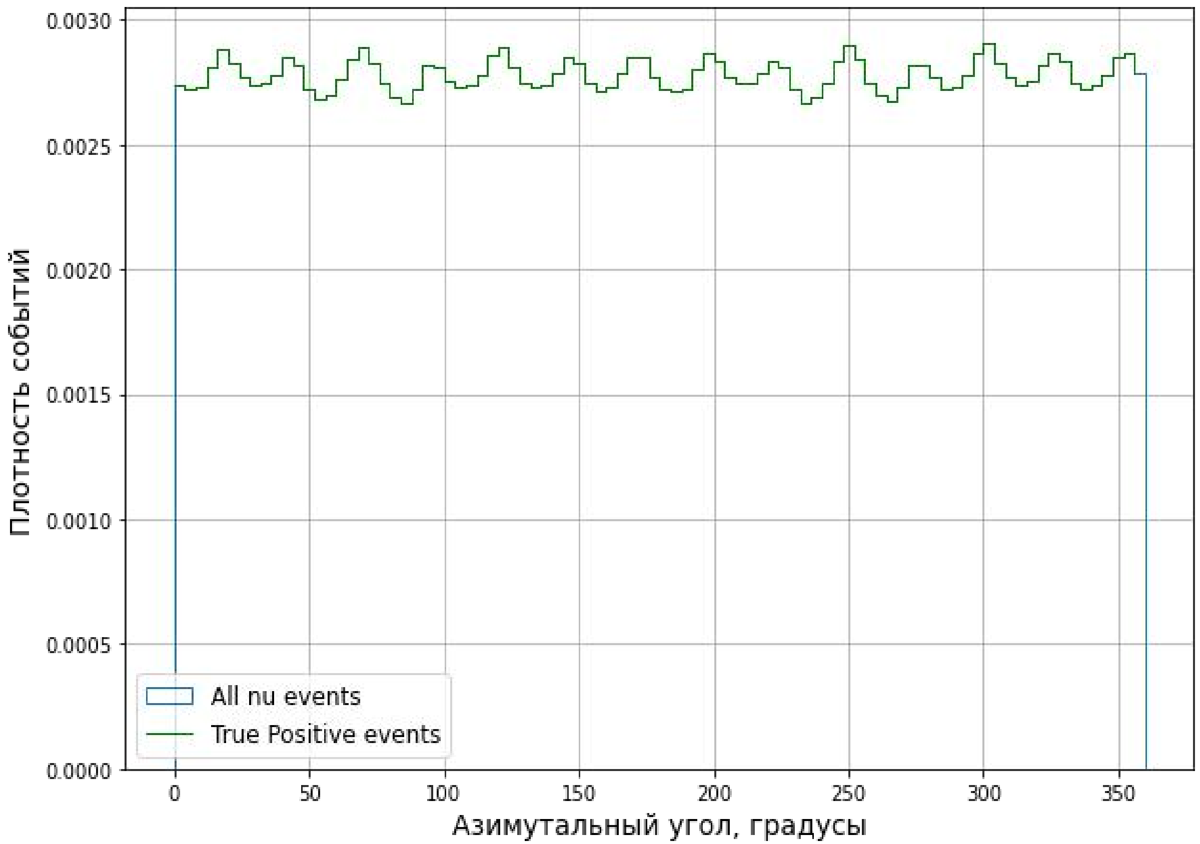


Полярный угол, мюоны

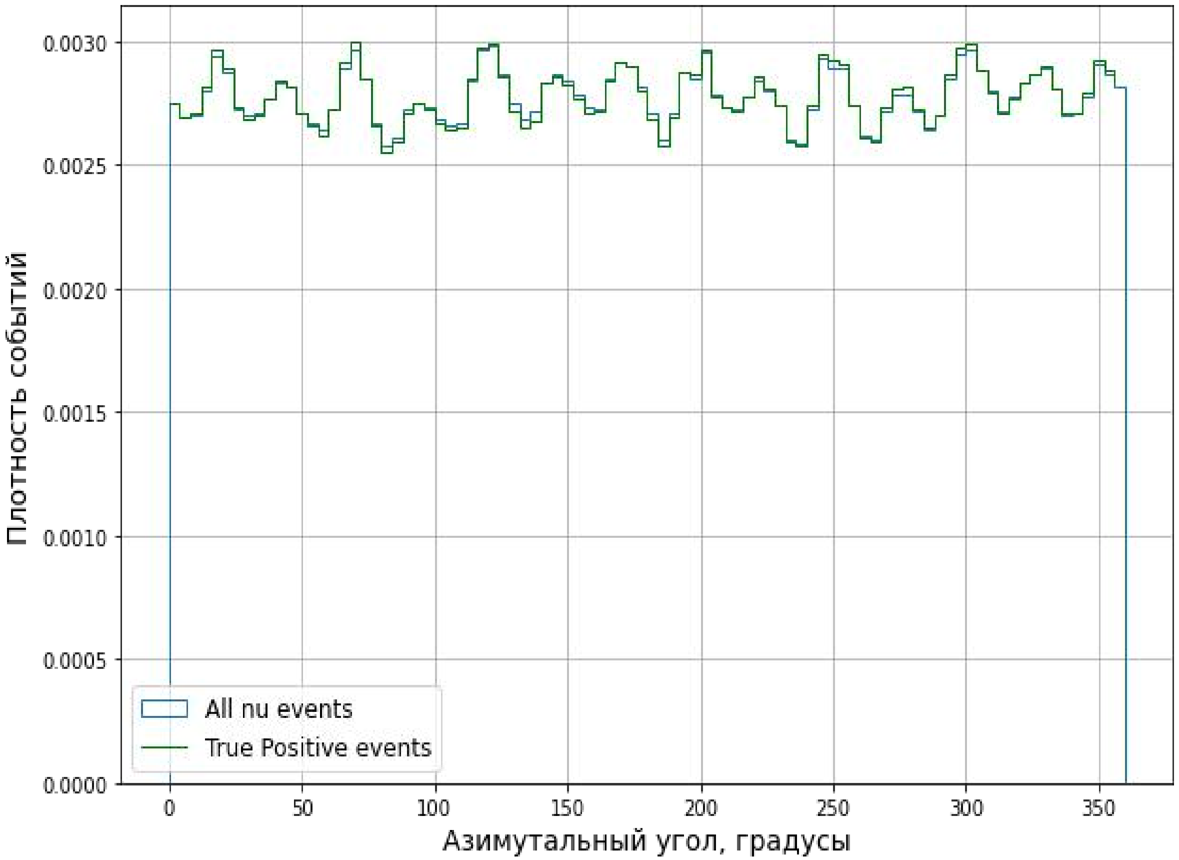


Спектры частиц

Азимутальный угол, мюоны



Азимутальный угол, нейтрино



Разные формулы

Focal loss

$$FL(p_t) = -\alpha_t (1-p_t)^{\gamma} \log(p_t)$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

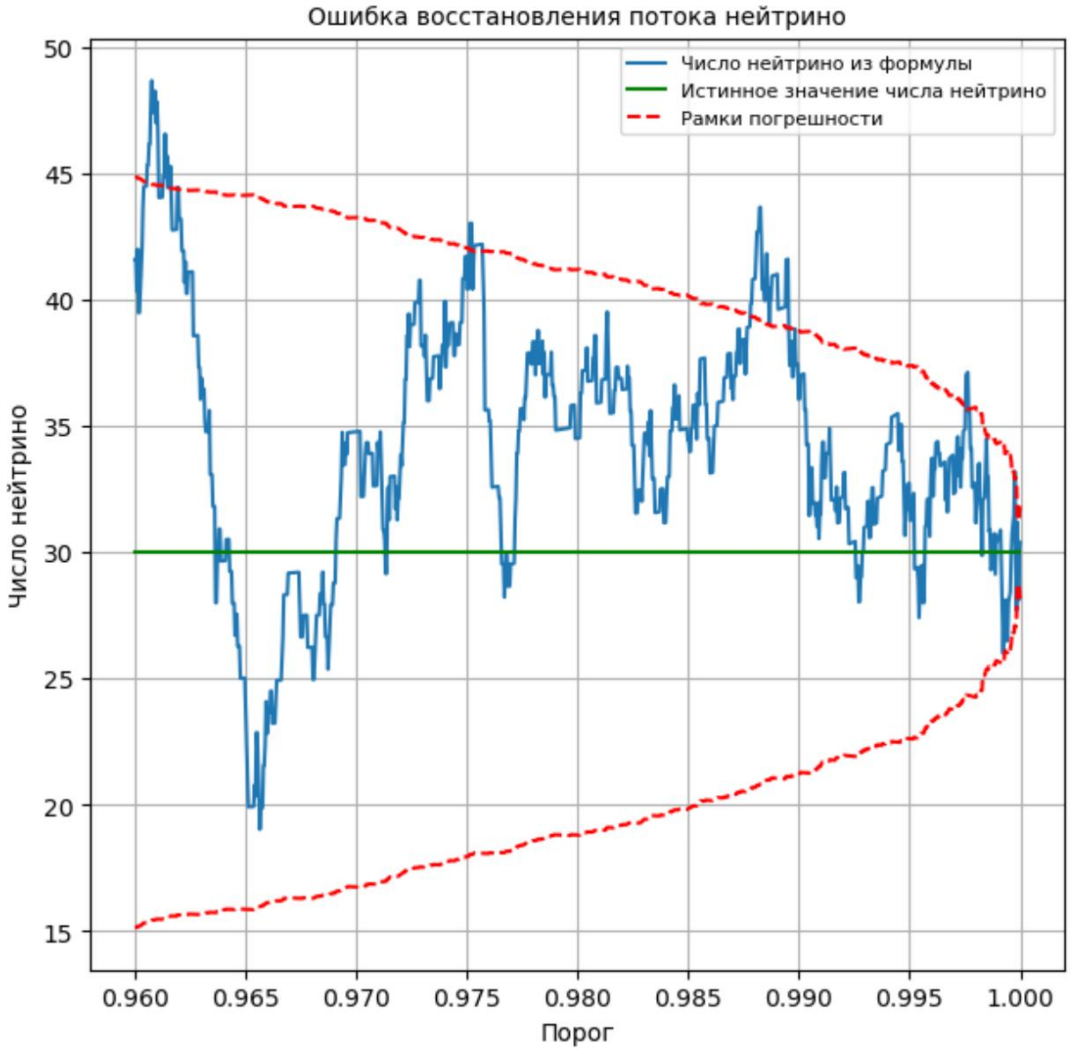
Оценка потока нейтрино (Следует из определний E и S)

$$n_\nu \approx \frac{n(\xi) - S^0(\xi)\ \ n(0)}{E^0(\xi) - S^0(\xi)}$$

ξ - порог классификации.

S^0, E^0 - оценки подавления и экспозиции на тестовом МК наборе данных.

n(ξ) - количество событий правее порога.



Можно оценить ошибку!

Отношение ШАЛ к ν : ~ 100 000

Оценка потока нейтрино

$$n_\nu \approx \frac{n(\xi) - S^0(\xi)\ \ n(0)}{E^0(\xi) - S^0(\xi)}$$

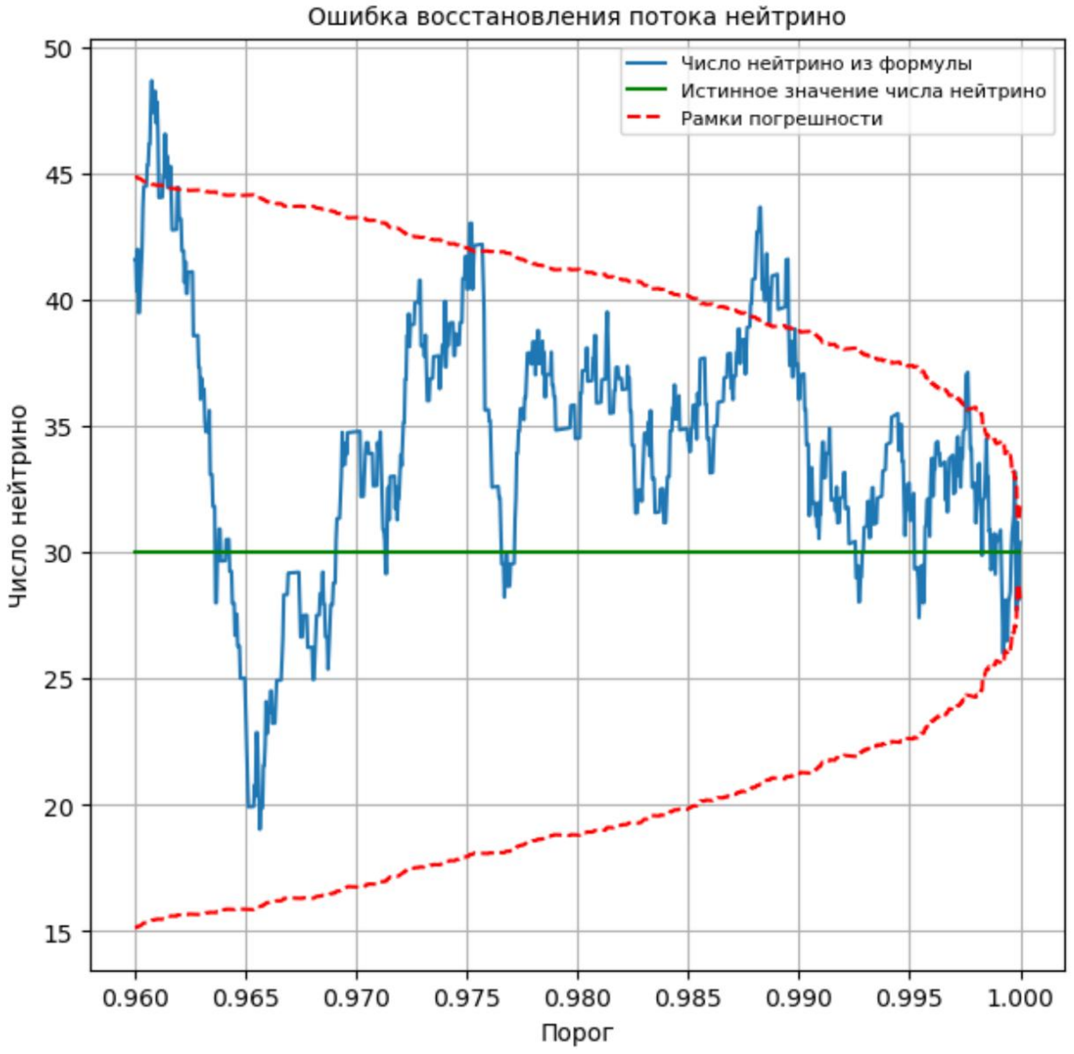
Можно оценить ошибку!

- Возьмём тестовые нейтринные события.

E - параметр биномиального распределения!
По МК оцениваем E^0 с доверительным интервалом.

- ШАЛ события и S^0 - аналогично!

- Считаем погрешность формулы потока.



Отношение ШАЛ к ν : ~ 100 000

ξ - порог классификации.
S^0, E^0 - подавление и экспозиция, оцененные на МК.
n(ξ) - количество событий, правее порога.

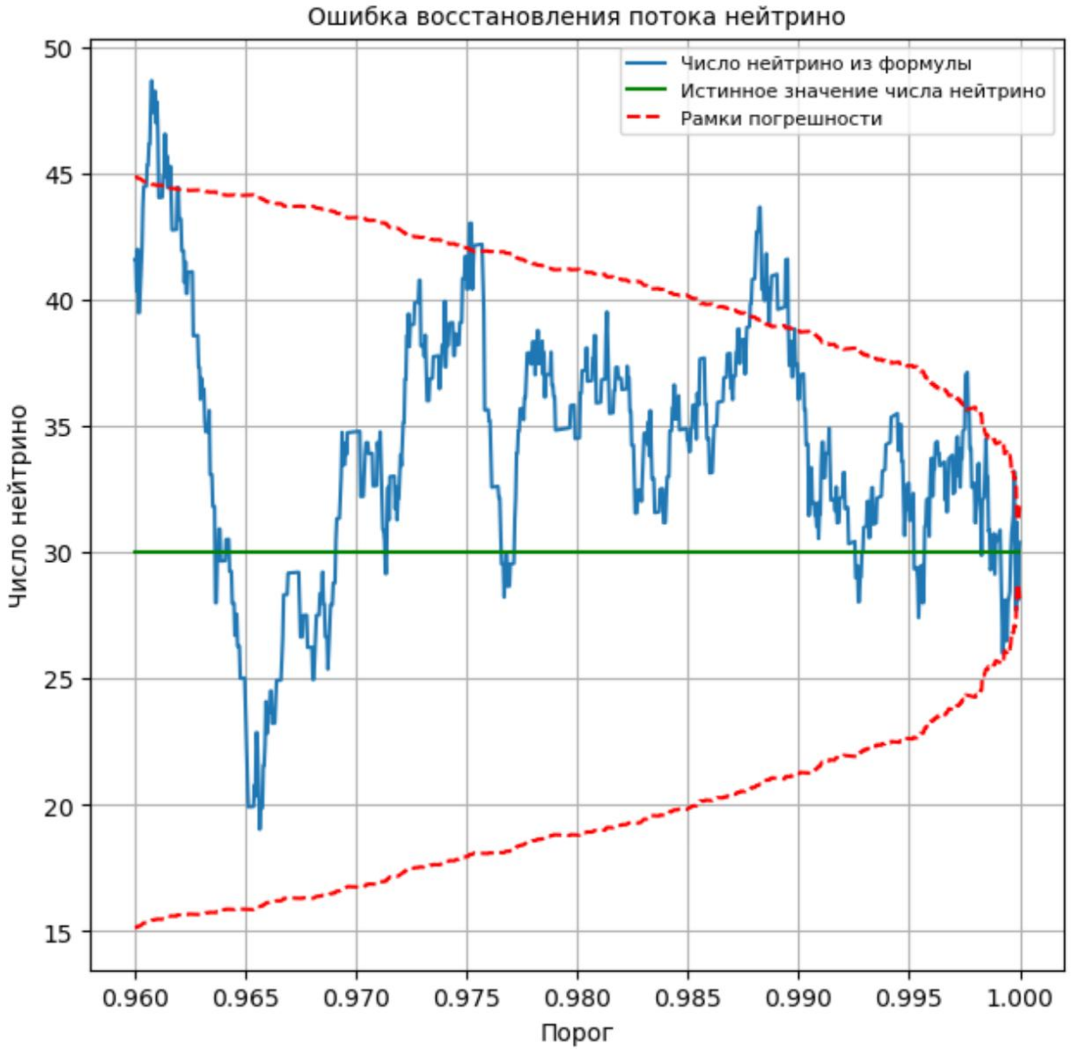
Оценка потока нейтрино

$$n_\nu \approx \frac{n(\xi) - S^0(\xi)\ \ n(0)}{E^0(\xi) - S^0(\xi)}$$

Можно оценивать в 2 режимах:

1) Оценка числа ν-событий в данных

2) Оценка параметра потока ν
(число n(ξ) - Пуассоновская случ. величина)



Отношение ШАЛ к ν : ~ 100 000

ξ - порог классификации.
S^0, E^0 - подавление и экспозиция, оцененные на МК.
n(ξ) - количество событий, правее порога.

Вывод формулы ошибки

Appendix A: Derivation of Eq. (??)

In this section we derive distributions, expected values and dispersions of random values in Eq.(??). Then, using them, we evaluate the error of estimating the number of neutrino events using Eq.(??). In this section, we use P to denote probability of some outcome for a random variable, and M and D for its expected value and dispersion, accordingly.

We start by discussing the properties of random variables in the experimental dataset. Let the latter contains n^0 events in total, n_ν^0 of which are neutrino-induced and $n_\mu^0 \equiv n^0 - n_\nu^0$ are EAS-induced. n_ν^0 and n_μ^0 are random variables distributed according to the Poisson law with parameters ν and μ respectively. Hence

$$P(n_\nu^0 = k) = \frac{\nu^k e^{-\nu}}{k!}\,, \tag{A1}$$

$$P(n_\mu^0 = k) = \frac{\mu^k e^{-\mu}}{k!}\,. \tag{A2}$$

Since n^0 is a sum of n_ν^0 and n_μ^0, it also follows the Poisson distribution,

$$P(n^0 = k) = \frac{(\nu+\mu)^k e^{-(\nu+\mu)}}{k!}\,. \tag{A3}$$

It's expected value and dispersion are:

$$M(n^0) = D(n^0) = \nu + \mu\,. \tag{A4}$$

Let us now address the classification of events by the neural network. A trained neural network can be considered as a black box. As it was discussed in section ??, for a fixed classification threshold ξ, the network classifies a neutrino-induced event correctly with some probability E, and EAS-induced event is identified incorrectly with the probability S. Hence the number of identified true and false neutrino-induced events are independent random variables with binomial distributions:

$$P(n_\nu = k | n_\nu^0 = m) = Bin(m, E)(k)\,, \tag{A5}$$

$$P(n_\mu = k | n_\mu^0 = m) = Bin(m, S)(k)\,. \tag{A6}$$

Here $n_\nu(\xi) \equiv n_\nu$, $n_\mu(\xi) \equiv n_\mu$, and $Bin(m,p)(k)$ stands for the binomial distribution with number of experiments m and success probability p:

$$Bin(m,p)(k) = C_m^k p^k (1-p)^{m-k}\,. \tag{A7}$$

The number of neutrino-induced events identified by the neural network on a test dataset is subject to both of the above-described random processes. Hence the full probability distributions, P_f, of n_ν and n_μ can be obtained by multiplying the corresponding Poisson and binomial distributions,

$$P_f(n_\nu = k) = \sum_{m=0}^{\infty} P(n_\nu = k | n_\nu^0 = m) P(n_\nu^0 = m) \tag{A8}$$

$$P_f(n_\mu = k) = \sum_{m=0}^{\infty} P(n_\mu = k | n_\mu^0 = m) P(n_\mu^0 = m) \tag{A9}$$

Using Eq. (A8), Eq. (A9), Eq. (A1) and Eq. (A2), one can evaluate the expected values and dispersions of n_ν and n_μ:

$$M(n_\nu) = D(n_\nu) = E\nu\,, \tag{A10}$$

$$M(n_\mu) = D(n_\mu) = S\mu\,. \tag{A11}$$

* E-mail: matseiko.av@phystech.edu

† E-mail: ivan.kharuk@phystech.edu

For the random variable n, which is a sum of n_ν and n_μ, one has:

$$M(n) = D(n) = E\nu + S\mu\,. \tag{A12}$$

Now the task is to estimate binomial parameters E and S of the neural network after measurements of n_ν and n_μ on the test dataset. To do this, we use standard formulae:

$$\tilde{E} = \frac{n_\nu(\xi)}{n_\nu^0}, \quad \tilde{S} = \frac{n_\mu(\xi)}{n_\mu^0}. \tag{A13}$$

The errors of these evaluations are calculated using the Clopper–Pearson interval. For example, setting confidence level $1-\alpha$ for $\tilde{E}$, we obtain interval $E_{\min} < E < E_{\max}$, where $E_{\min}$ and $E_{\max}$ are from equations:

$$\frac{\Gamma(n_\nu^0+1)}{\Gamma(n_\nu)\Gamma(n_\nu^0 - n_\nu + 1)} \int_0^{E_{\min}} t^{n_\nu - 1}(1-t)^{n_\nu^0 - n_\nu} dt = \frac{\alpha}{2} \tag{A14}$$

$$\frac{\Gamma(n_\nu^0+1)}{\Gamma(n_\nu+1)\Gamma(n_\nu^0 - n_\nu)} \int_0^{E_{\max}} t^{n_\nu}(1-t)^{n_\nu^0 - n_\nu - 1} dt = 1 - \frac{\alpha}{2} \tag{A15}$$

The variation of $\tilde{E}$ with confidence level $1-\alpha = 0.68$ we will consider to be equivalent to one standard deviation and calculate as

$$\sigma_{\tilde{E}} = (E_{\max} - E_{\min})/2, \tag{A16}$$

For case of S the reasoning is similar:

$$\sigma_{\tilde{S}} = (S_{\max} - S_{\min})/2 \tag{A17}$$

Now we are ready to evaluate the dispersion of N_ξ estimated using Eq. (??). For this purpose, we used the standard formula for the dispersion of a function of random variables,

$$\sigma_{N_\xi}^2 = \sum_v \left(\frac{\partial N_\xi}{\partial v}\right)^2 \sigma_v^2 + 2\sum_{v \neq u} \left(\frac{\partial N_\xi}{\partial v}\right)\left(\frac{\partial N_\xi}{\partial u}\right) Cov_{v,u}\,. \tag{A18}$$

Here, v and u denote arguments of the function N_ξ, which are $n(\xi) \equiv n$, $n(0) \equiv n^0$, $\tilde{E}(\xi)$ and $\tilde{S}(\xi)$; σ_v^2 stands for squared variance of v, and $Cov_{v,u}$ denotes covariance between v and u.

Let us explicitly write out the estimation of the variances and covariances. According to Eq. (A4), $\sigma_{n^0}^2$ can be estimated as

$$\sigma_{n^0}^2 = n^0\,. \tag{A19}$$

Further, from Eq. (A12), one gets

$$\sigma_n^2 = n\,. \tag{A20}$$

Next, $\sigma_{\tilde{E}}^2$ and $\sigma_{\tilde{S}}^2$ can be obtained from Eq. (A16) and Eq. (A17).

Finally, note that there are only two dependent random variables in Eq. (??) — n and n^0. Their covariance can be calculated using Eq. (A3), Eq. (A8) and Eq. (A9),

$$Cov_{n^0,n} = E\nu + S\mu = M(n)\,. \tag{A21}$$

Therefore this covariance can be estimated as n.

By calculating the partial derivatives of N_ξ in Eq. (A18) and substituting the obtained expressions for variances and covariances, we obtain the final result:

$$\sigma_{N_\xi}^2 = \frac{(n - n^0\tilde{S})^2}{(\tilde{E}-\tilde{S})^4}\cdot\sigma_{\tilde{E}}^2 + \frac{(n - n^0\tilde{E})^2}{(\tilde{E}-\tilde{S})^4}\cdot\sigma_{\tilde{S}}^2 + \frac{n + n^0(\tilde{S})^2 - 2n\tilde{S}}{(\tilde{E}-\tilde{S})^2} \tag{A22}$$

Про нейросети

Основы нейронных сетей

Стандартные алгоритмы:

“Программа” - фиксированный, написанный человеком алгоритм решения задачи.

Как выделить признаки?

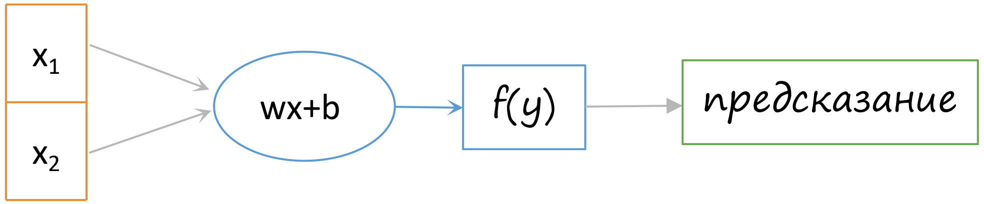
Машинное обучение:

“Программа” - обучающийся на примерах алгоритм выделения оптимальных признаков.

Программы, создающие оптимальные алгоритмы

Основы нейронных сетей

Пример: по координате точки на плоскости предсказать, лежит ли она внутри окружности единичного радиуса.



Предсказание:

[0;1],   0 - внутри, 1 - снаружи

Мера ошибки:

штраф = |предсказание - правда| ≥ 0

Инициализация:

Выбираем случайно w и b

Оптимизация:

Пока возможно улучшение:

1. Для заданной точки, считаем предсказание
2. Считаем величину функции штрафа
3. Методом градиентного спуска изменяем w и b, чтобы минимизировать штраф.
4. Берем следующий “обучающий пример”

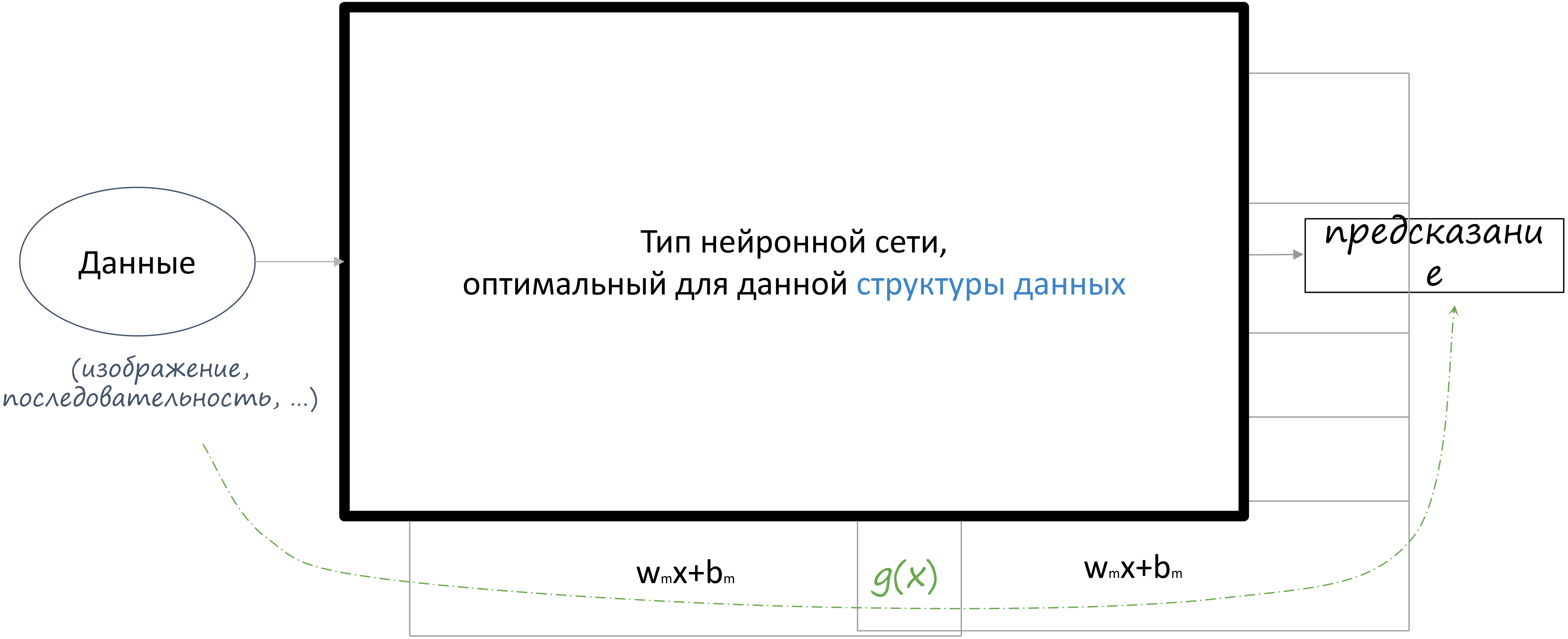
Итог:

Оптимальные значения w и b.

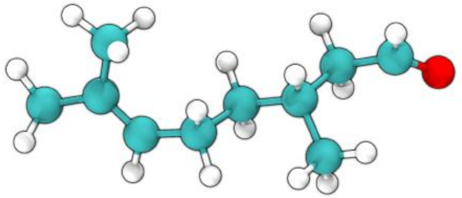
Найденные оптимальные параметры и есть итоговая “программа”.

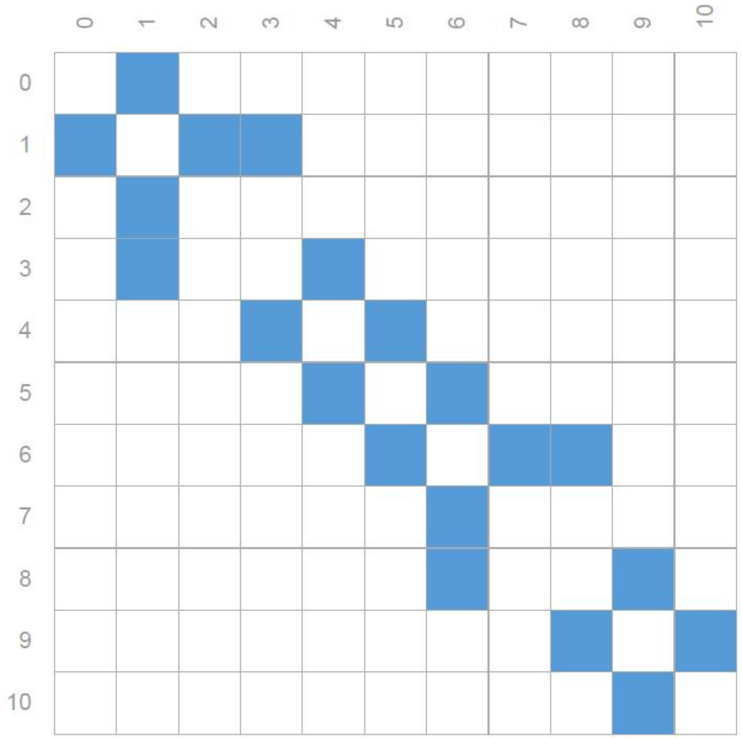
Сила машинного обучения

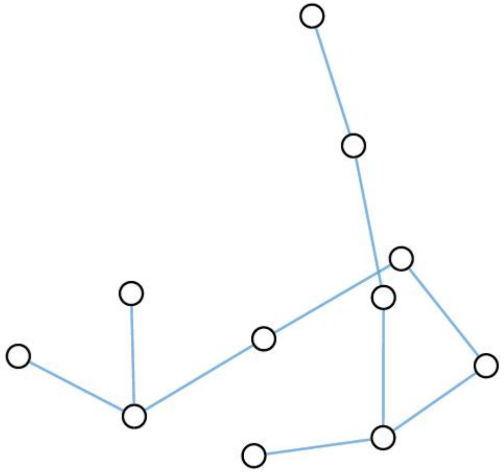
Нейросеть способна аппроксимировать любую функцию



Графовые сети

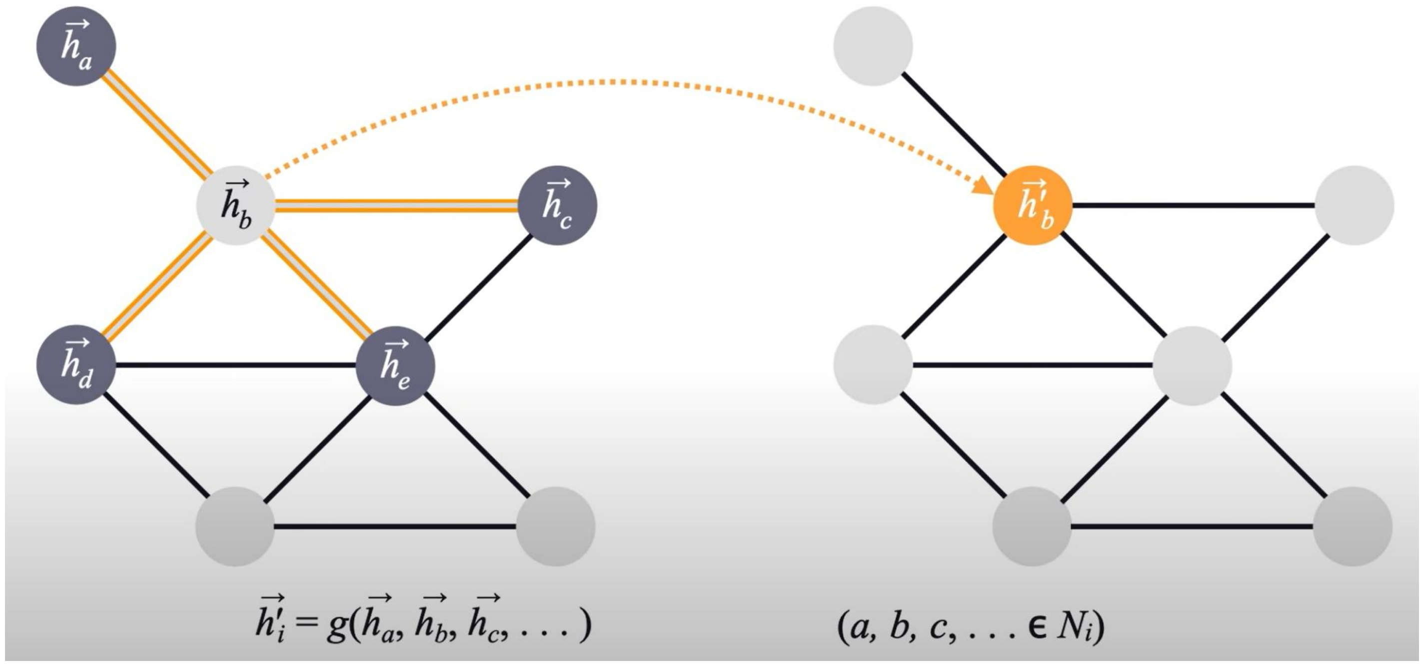






Графы позволяют учитывать более сложные связи в данных

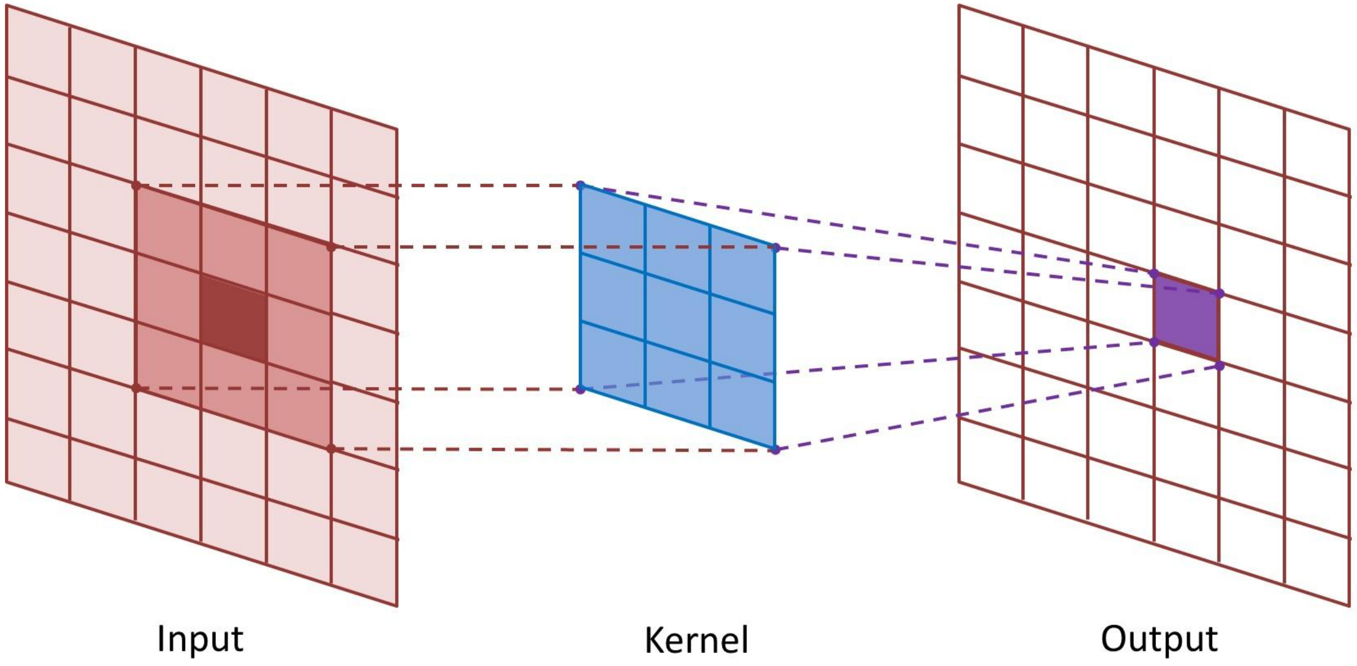
Графовые сети



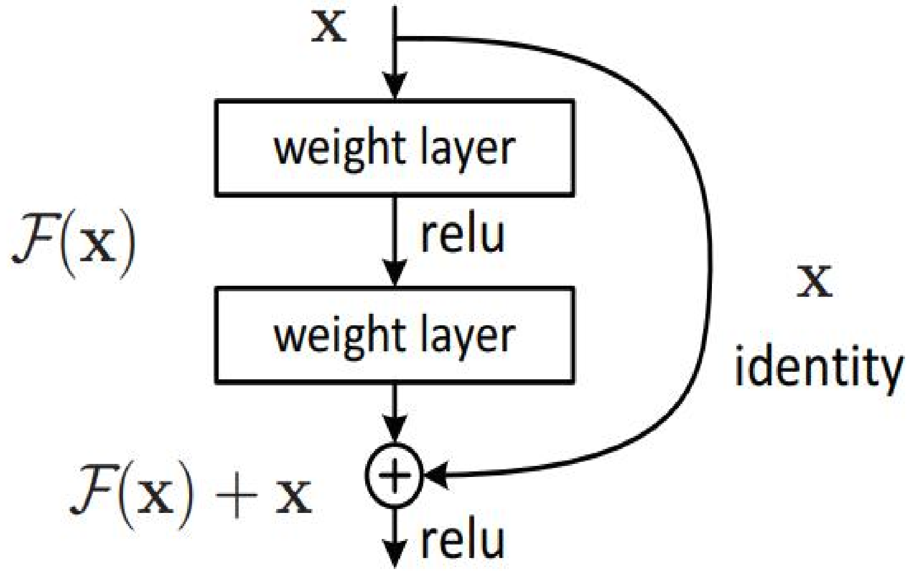
Графовые сети “обновляют” граф: на следующем шаге значение вершины является функцией от а) её соседей, б) связывающих рёбер, в) глобальных агрегированных свойств графа.

Для реконструкции угла прилета, агрегатируется информация со всего графа.

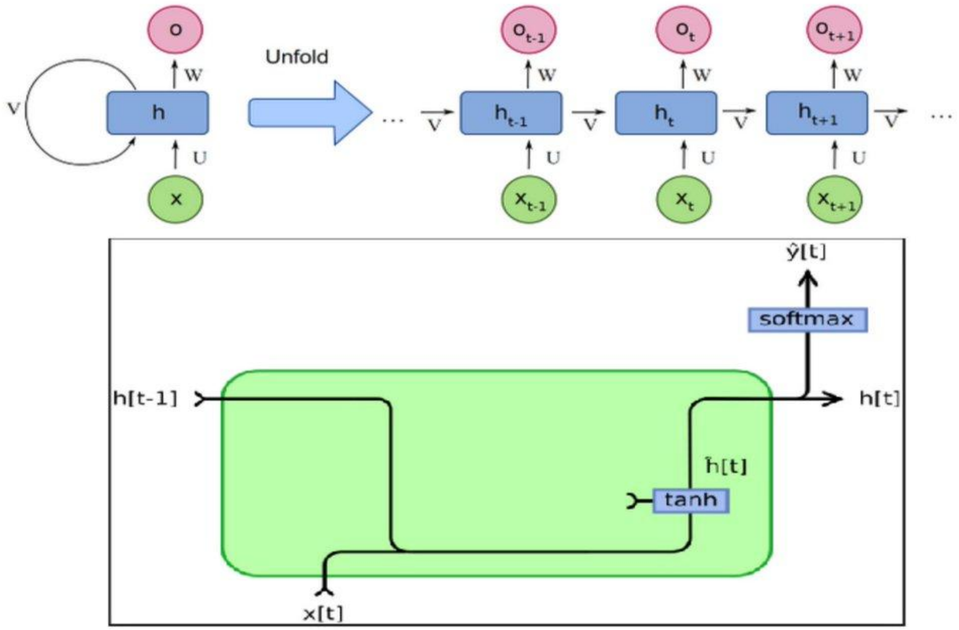
Свёрточная сеть и resnet



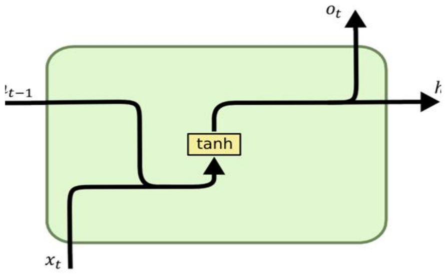
R



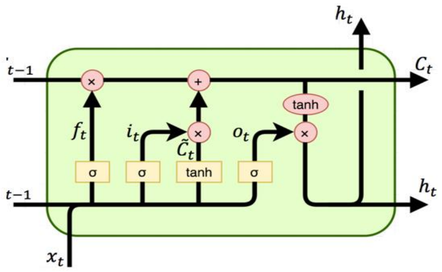
Рекуррентная сеть



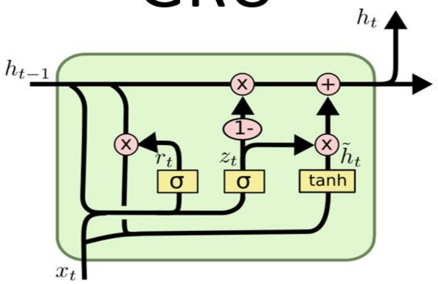
RNN



LSTM



GRU



Про углы

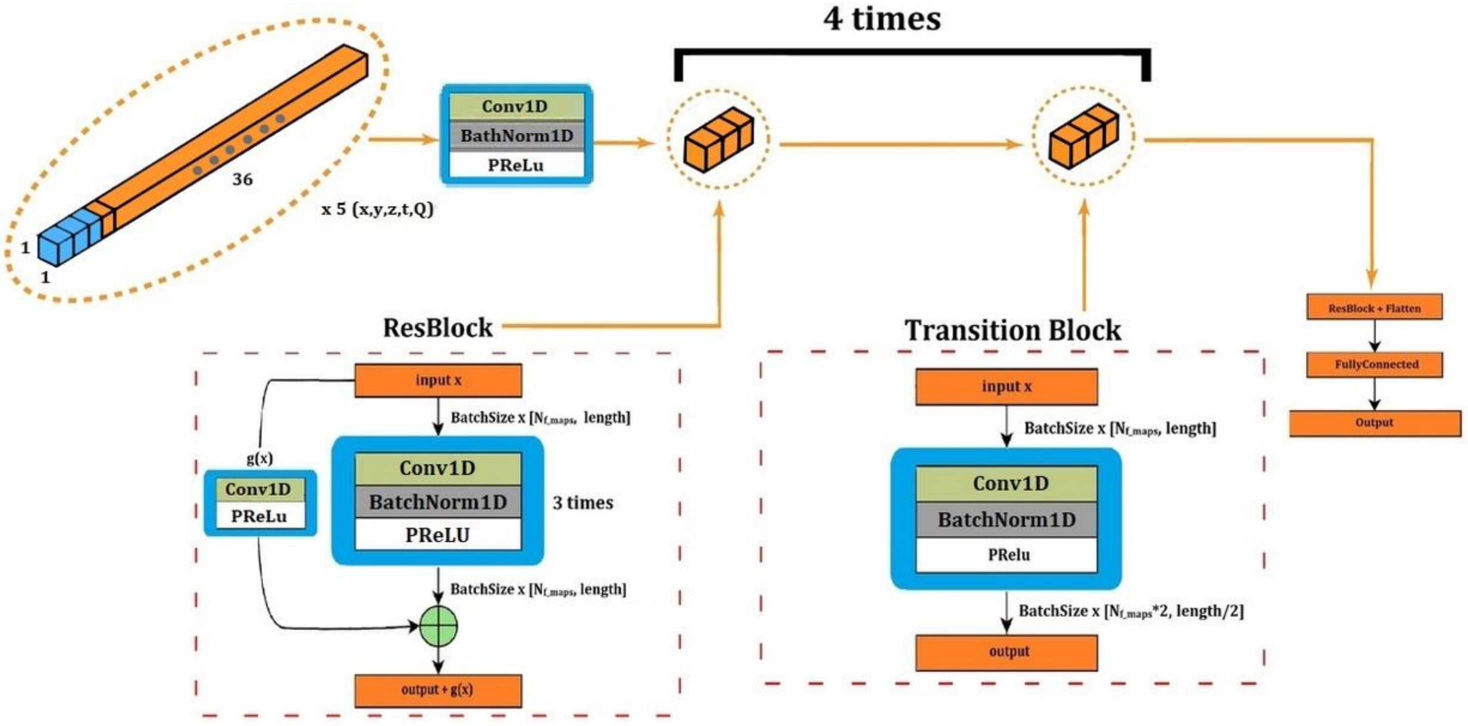
Нейронная сеть

- Используются:

А) сверточная нейронная сеть (CNN) на основе сети ResNet [3]

Б) графовая сверточная нейронная сеть (EdgeGNN[4])

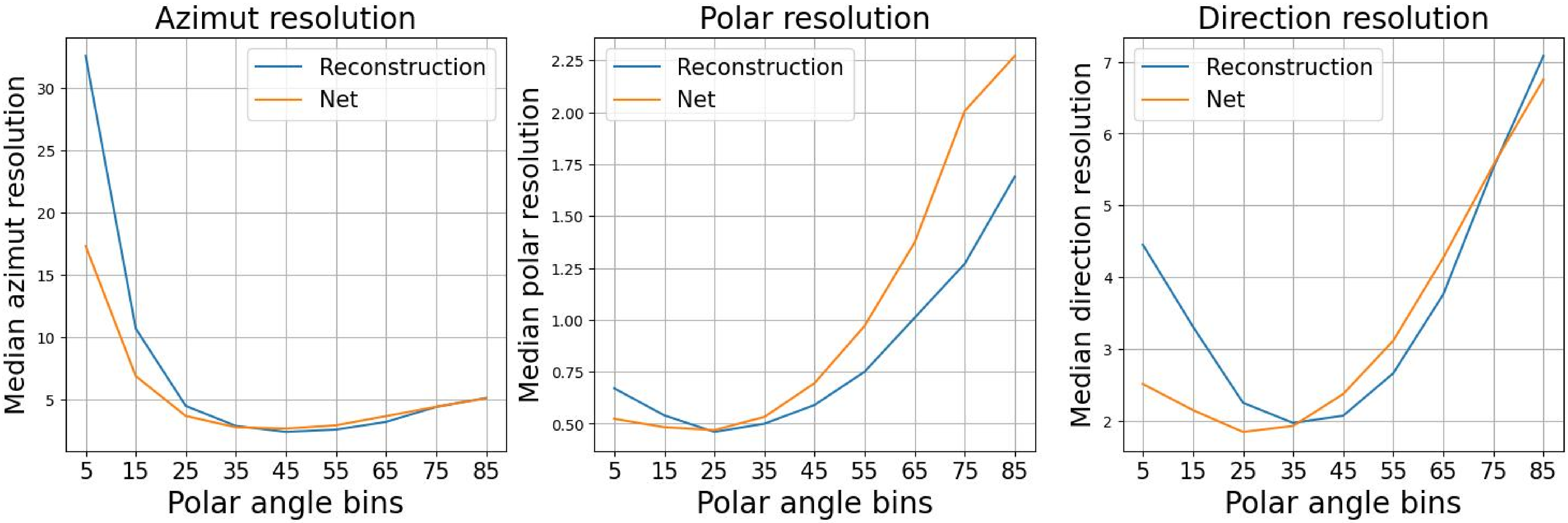
- Работа сети оценивается по медианным угловым разрешениям



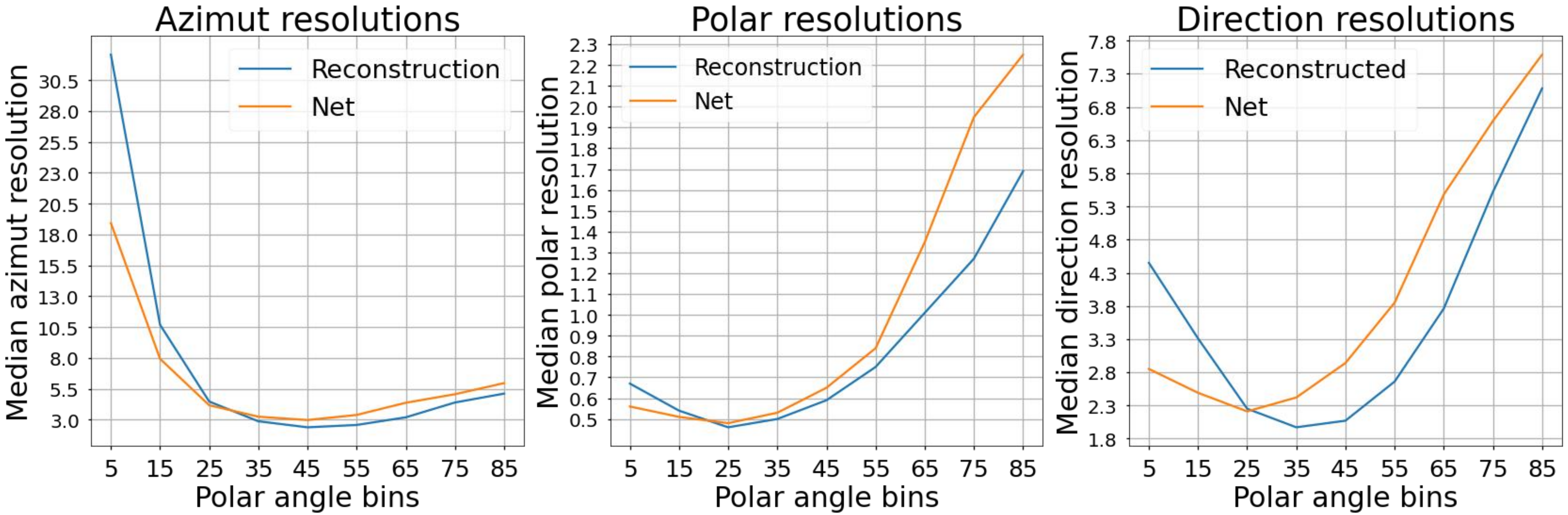
Архитектура CNN

[3] arxiv.org/abs/1512.03385
[4] arxiv.org/abs/1801.07829

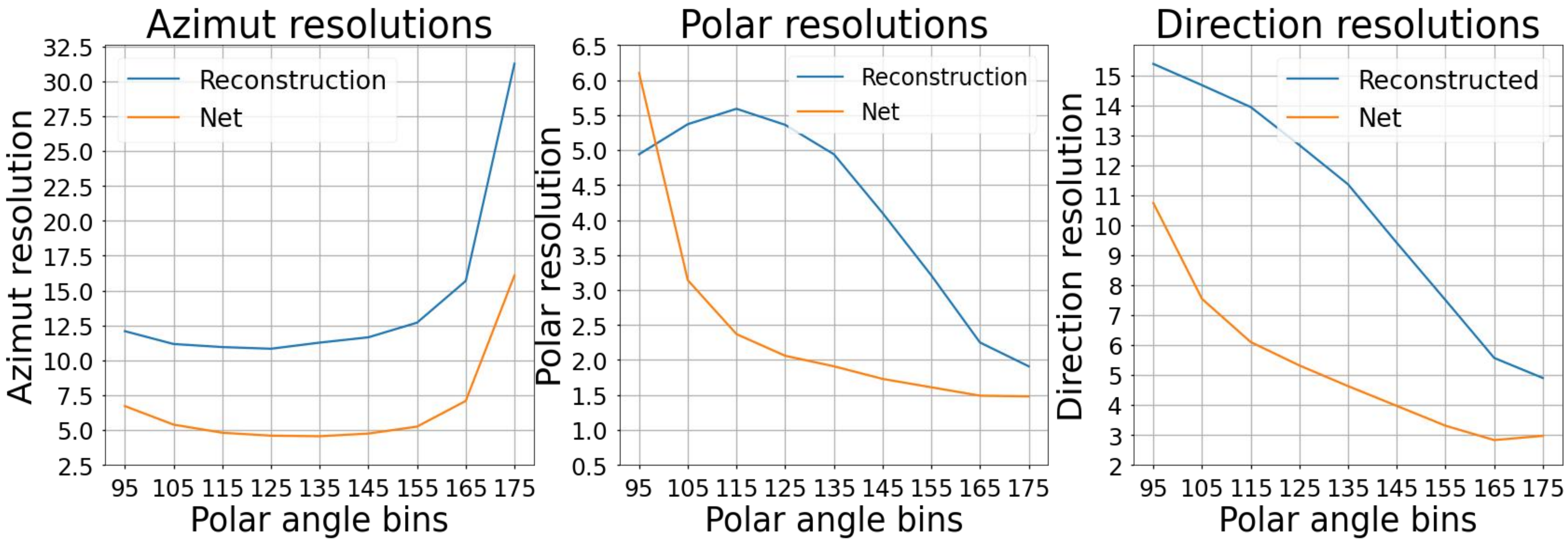
Нейтрино : Угловые разрешения GCN



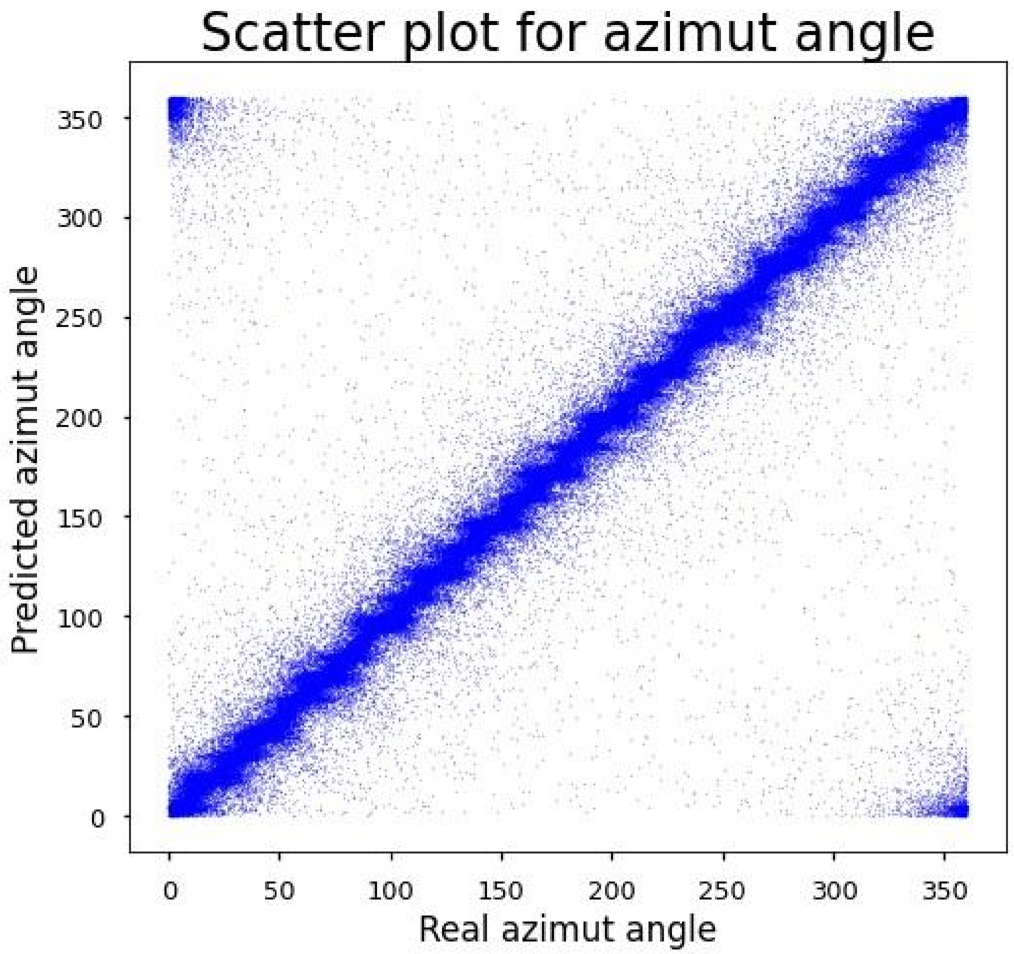
Нейтрино : Угловые разрешения CNN

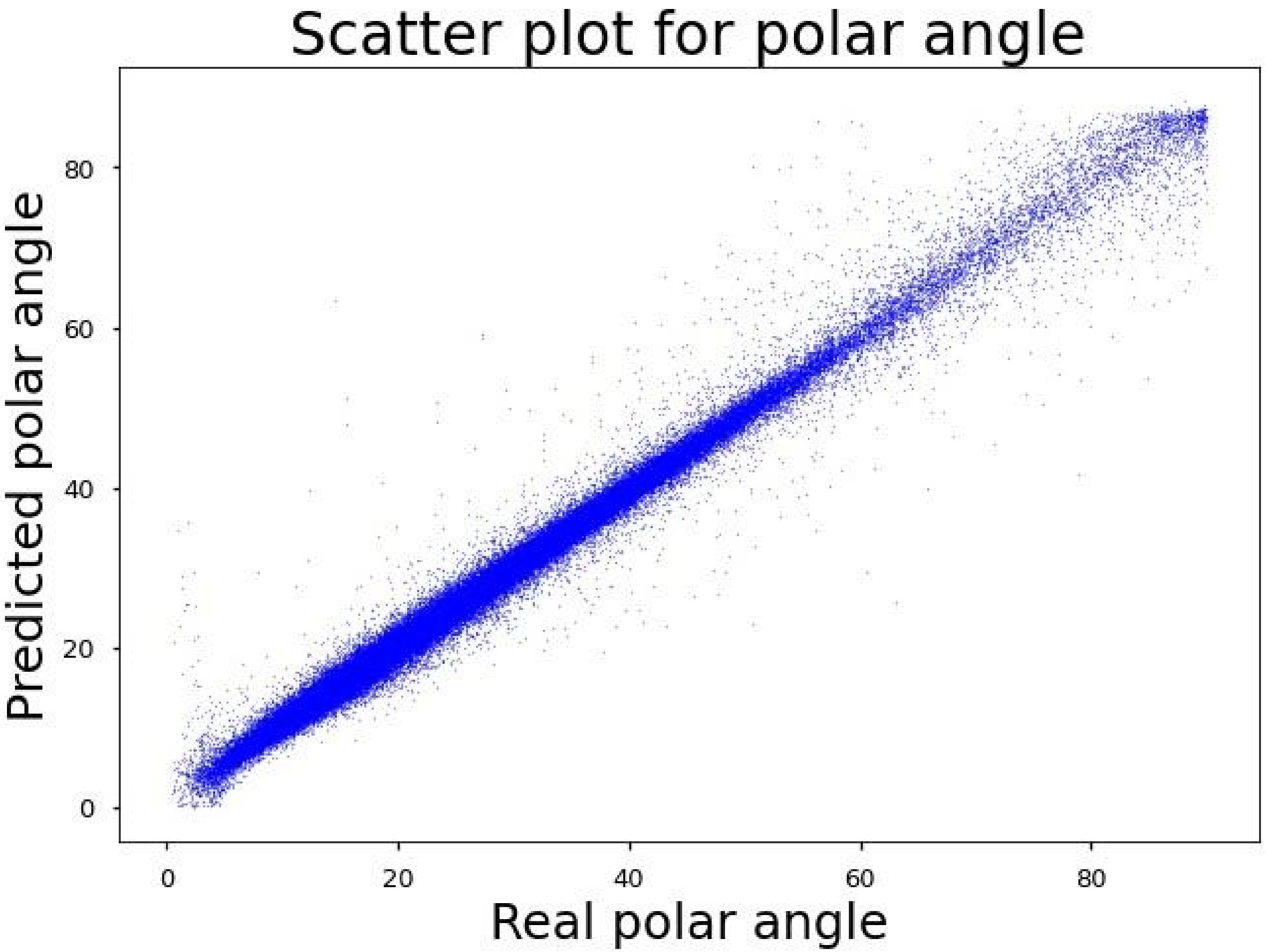


Мюоны : Угловые разрешения



Нейтрино: Восстановленные углы





Мюоны: Восстановленные углы