Extrapolation of NCSM results using machine learning and artificial neural networks

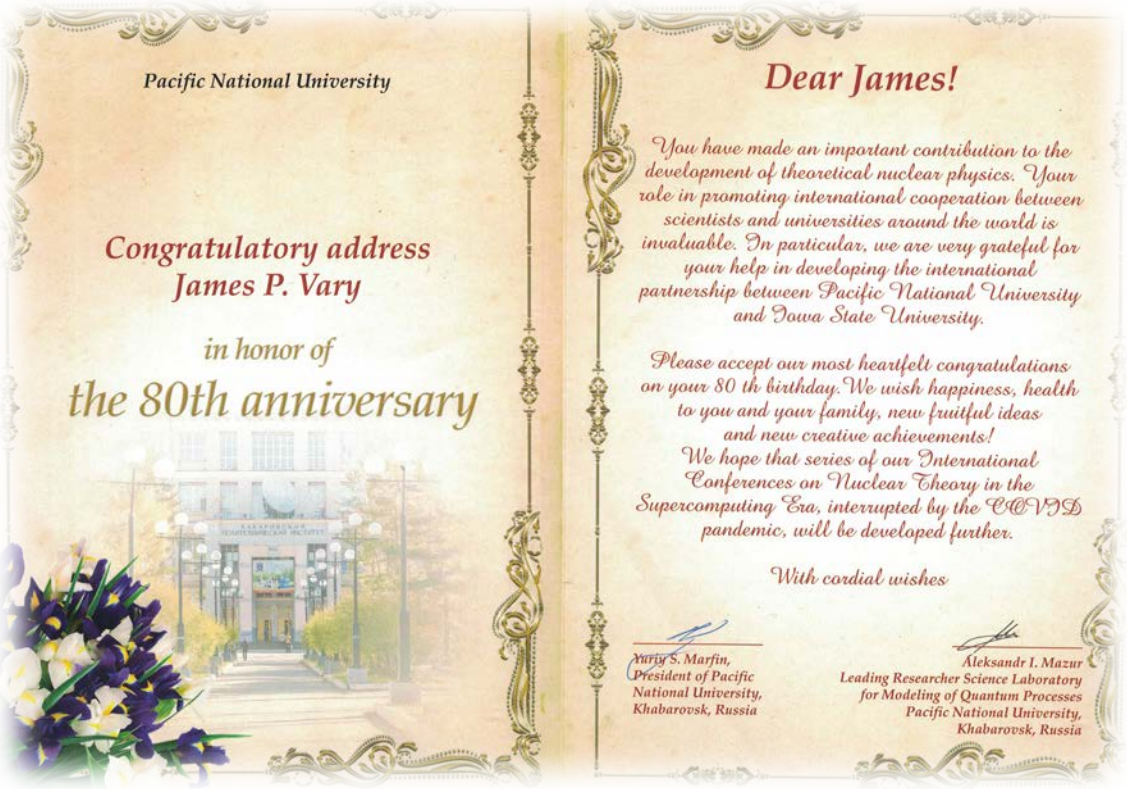
Khabarovsk

A. Mazur, **R. Sharypov, A. Belozerov**
SciLab for Modeling of Quantum Processes,
Pacific National University, Khabarovsk, Russia
A. Shirokov ***SINP Moscow State University, Moscow, Russia***
I. J. Shin ***RISP, Institute for Basic Science, Daejeon, Korea***

Ministry of Science and Higher Education,
project No. 0818-2020-0005
Theor. Phys. and Math. Advancement
Foundation ”BASIS”, project No. № 21-1-1-67-3)

NTSE-2023
IMP CAS
June 5, 2023

Congratulation



Pacific National University

Congratulatory address
James P. Vary

in honor of
the 80th anniversary

Dear James!

You have made an important contribution to the development of theoretical nuclear physics. Your role in promoting international cooperation between scientists and universities around the world is invaluable. In particular, we are very grateful for your help in developing the international partnership between Pacific National University and Iowa State University.

Please accept our most heartfelt congratulations on your 80 th birthday. We wish happiness, health to you and your family, new fruitful ideas and new creative achievements!
We hope that series of our International Conferences on Nuclear Theory in the Supercomputing Era, interrupted by the COVID pandemic, will be developed further.

With cordial wishes

Yuriy S. Marfin,
President of Pacific National University, Khabarovsk, Russia

Aleksandr I. Mazur
Leading Researcher Science Laboratory for Modeling of Quantum Processes
Pacific National University, Khabarovsk, Russia

Ten years ago
The 70th anniversary
ISU, Ames

Motivation

Now machine learning methods have been widely used in nuclear physics. The detailed overview of application of machine learning methods in nuclear physics is presented in the review "Machine Learning in Nuclear Physics" (A. Boehlein et. al, arXiv:2112.02309v2 [nucl-th] 2 May 2022)

In my talk, I will consider only one of the interesting problems - the extrapolating results of variational (NCSM) calculations to large model spaces.
Various methods of extrapolation are known, but all of them do not have a strict theoretical justification. Therefore, the development of new reliable extrapolation methods remains an important task.

The development of extrapolation methods is also interesting from the point of view of the machine learning methods development themselves, because they are usually applied to interpolation problems.

Outlook

Artificial Neural Networks (ANNs)

- Topology
- Hyperparameters

Extrapolation algorithm

- Selection of input data
- Scaling of input data
- Selection of ANNs

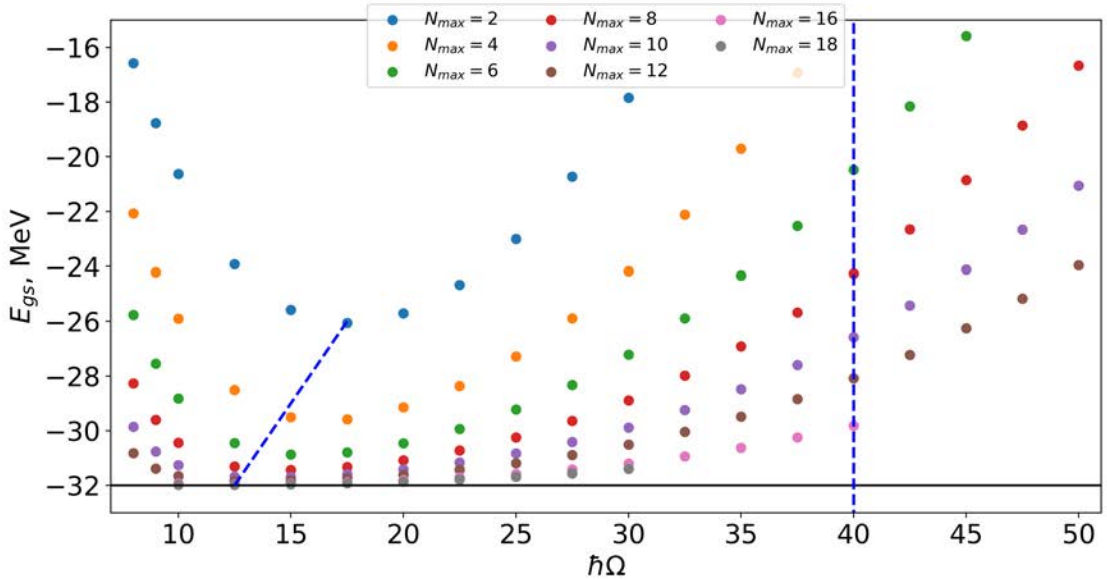
Extrapolation results:

- ^{6}Li: E_{gs}, rms-radii
- ^{6}He, ^{6}Be: E_{gs}

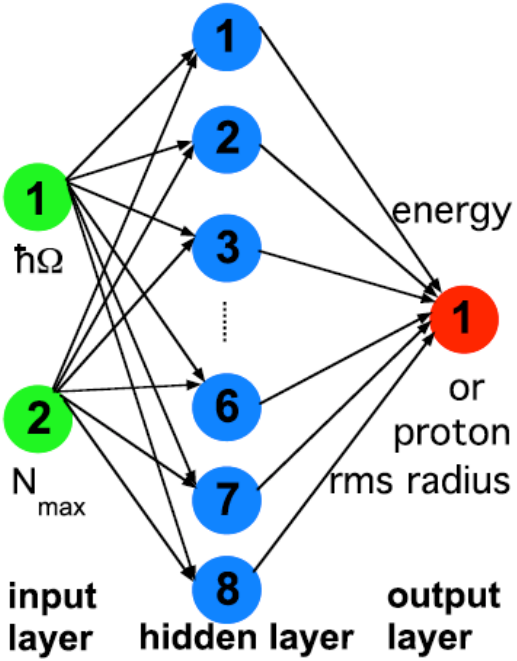
Problem

As input, we have a set of NCSM energies (or other output) in different model spaces N_{max} as function of the $\hbar\Omega$. We need to extrapolate this data to the large N_{max} in order to estimate the binding energy (radius of the nucleus, etc.) and then to compare with experimental data.

^{6}Li, NCSM with Daejeon16



Neural Networks



Sigmoid activation function

ANNs are composed of an input layer, one or more hidden layers, and an output layer. The neurons in the input layer receive the data from outside and transmit the data via weighted connections to the neurons in the first hidden layer, which transmit the data to the next layer. Finally, the neurons in the output layer give the results.

G. A. Negoita et al. "Deep learning: Extrapolation tool for ab initio nuclear theory" **PRC 99**, 054308 **(2019)**

I. Vidana "Machine learning light hypernuclei" (**arXiv:2203.11792v2 [nucl-th]** 14 Feb 2023)

Neural Networks

ANNs consist highly interconnected artificial **neurons**, which are processing units. The **neurons** are connected with each other via adaptive synaptic weights ω_i. The **neurons** collects all the input signals and calculates a net signal as the weighted sum of all input signals.

$$s = \sum_i \omega_i x_i + b$$

Set of **weights** ω_i and **biases** b are ANNs trainable parameters.
Next, the **neuron** calculates and transmits an output signal. The output signal is calculated using an activation function, which depends on the value of the net signal. We use sigmoid activation function

$$f(s) = \frac{1}{1+\exp(-s)}$$

and linear activation function

$$f(s) = s$$

Neural Networks

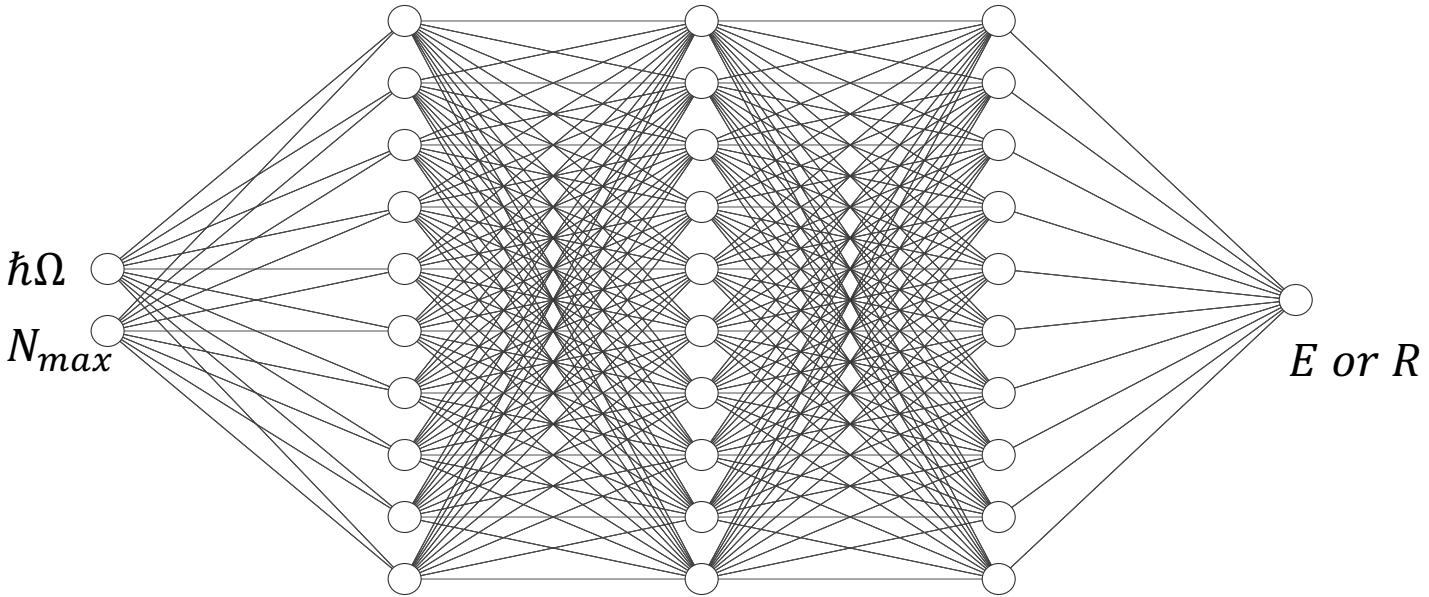
The development of an ANN is a two-step process with training and testing stages. In the training stage, the ANN adjusts its weights until an acceptable error level between desired and predicted outputs is obtained. The difference between desired and predicted outputs is measured by the loss (error) function. A common choice for the **loss function** is mean-square error (**MSE**).

After a single passage of signals through the ANN and obtaining predictions, the loss function is calculated. The weight coefficients ω_i are changed using the **Adam algorithm**. This algorithm adjusts the weight factors ω_i to minimize the loss function. The value by which the weight coefficients change each time is defined by the **learning rate γ**.

The set of coefficients ω_i is updated each time a certain amount of data, set with the batch size B_S, has passed through the ANN. We use $B_S = len(dataset)$. After passing all learning dataset, the ANN predictions are tested on the validation set. The learning process is iterative and is repeated many times. Each passage of the entire dataset through the ANN is called the **epoch**. In our case $\boldsymbol{N_{ep} = 500\,000}$.

Neural Networks

We use more complicated networks with **three hidden layers** of **10 neuron** in each. Besides, we use **different** activation functions.



Trainable parameters: 261

linear activation → sigmoid activation → sigmoid activation → linear activation

Neural Networks

Some feature of our training algorithm.

1. We believe that for extrapolation problems the **last layer** should have a **linear activating function**. This is important because the **output interval should not be restricted**.

   Papers **PRC99(2019)** and **arXiv:2203.11792v2 [nucl-th]** are used **only sigmoids.**

2. We do not have any problems with ANNs **overfitting** even for large epoch numbers. So, we do not divide input data on training, test and validation sets. That is, we use **all the data to train the networks**.
   This is very important due to the limited number of input data.

Overwriting in a simple words: The such networks simply "remembers" the training set but does not generalize trends. So, such networks can not make predictions. Moreover, this networks does not reproduce even the test set.

Extrapolation algorithm

Usually, machine learning is not applied to extrapolation problems because predictions of a single network include randomness due to the initial values of the weight coefficients.

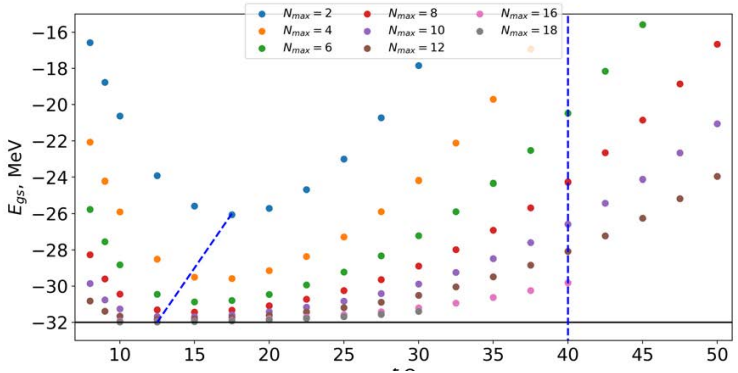
However, if you use an **ensemble** of neural networks with **identical hyperparameters**, but **different initial values of the weight** coefficients, the extrapolation problem can be solved.

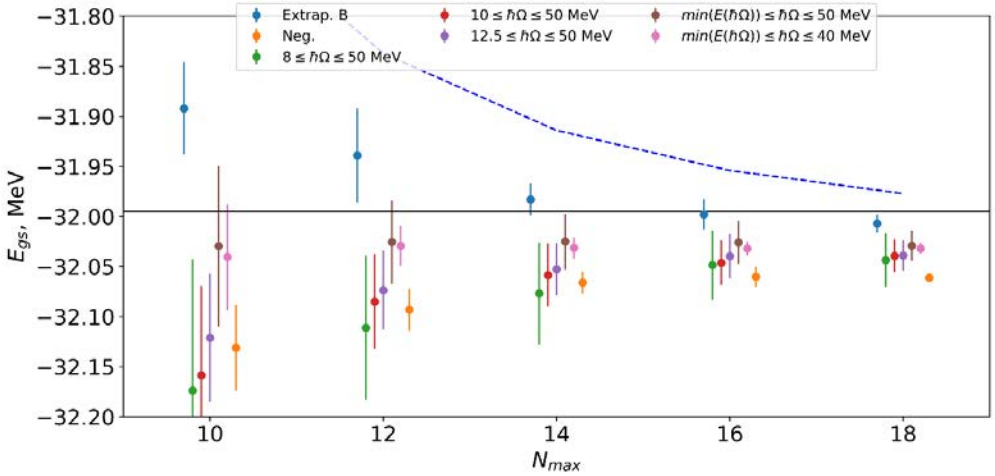
We train **1024 neural networks**.

The training of the ANNs ensemble is carried out according to the above scheme.

Extrapolation algorithm: selection of input data

^{6}Li, NCSM with Daejeon16





The **preliminary selection of data** plays an **important** role.

black line: experiment;
blue dotted: variatonal minimum;
symbols: orange: PRC99(2019) - all data are used;
blue – exponential extrapolation PRC79(2009);
green: our results, **all data**;
Vertical lines – error bars.

Our results with different intervals
pink: $(\min E_{gs}(\hbar\Omega)$ - 40) MeV – the BEST
red: (10 – 50) MeV;
violet: (12.5 – 50) MeV;
Brown: $(\min E_{gs}(\hbar\Omega)$ - 50) MeV.

arXiv:2203.11792v2 [nucl-th] uses data with $\hbar\Omega$ = (28-34) MeV which turns the original data into a set of oblique straight lines, while the minimum of the corresponding curves is not included.

Extrapolation algorithm: scaling of input data

Usually, a standard technique to improve Neural Netvork training is data scaling. Since the sigmoid activating function

$$f(s) = \frac{1}{1+\exp(-s)}$$

at $|s| > 4$ becomes a straight horizontal line, learning in these regions is the worst (the derivative is equal 0). Multiplying by sufficiently small weights can bring the sum of the sigmoid input to the desired range $|s| < 2$, but training with too small weights can also make learning difficult.

We have analyzed the dependence of the predictions of our algorithm on the scaling of the input data (N_{max},$\hbar\Omega$) and output data (E_{gs}).

Results are presented in the table below.

Extrapolation algorithm: scaling of input data

In this table, this dependence is not so strong (although in the worst case, the error is 3 times greater than in the best!). But when analyzing other scaling ranges (for example, $(0.1 \div 0.8)$ for both input and output data) the error increase up to 60 keV.

Input \ Output	$(0 \div 1)$	$(0 \div 2)$	$(0 \div 4)$
$(0 \div 1)$	$-32.033(24)$	$-32.031(10)$	$-32.032(11)$
$(0 \div 2)$	$-32.035(32)$	$-32.031(10)$	$-32.031(11)$
$(0 \div 4)$	$-32.029(24)$	$-32.031(14)$	$-32.033(13)$

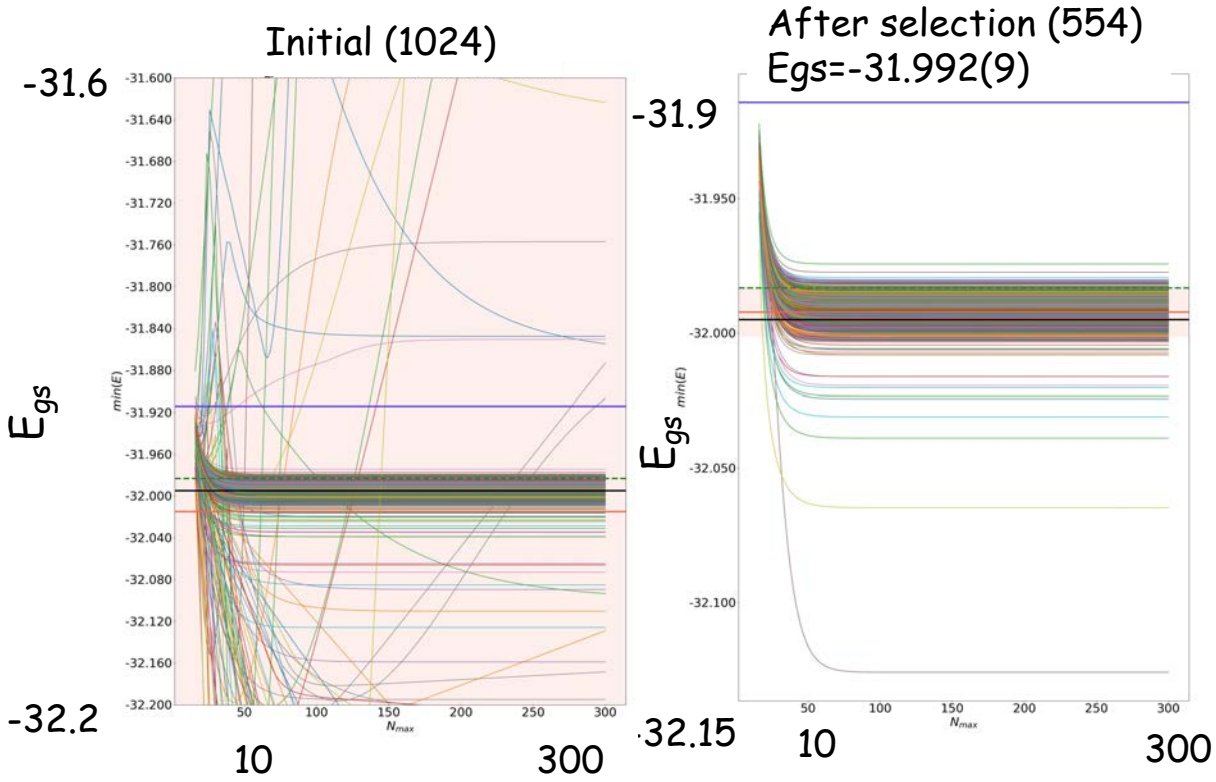
Extrapolation algorithm: selection of ANNs

Not all trained Neural Networks will provide reliable predictions; so, it is important to have criteria to select obviously incorrectly trained networks.

Criteria what we use:

1. "Soft" variational principle (only for energy): Predictions can a little violate variational principle (total violation at large N_{max}=300 ~ 5 keV).

2. No $\hbar\Omega$ dependency of predictions at large N_{max}=300.

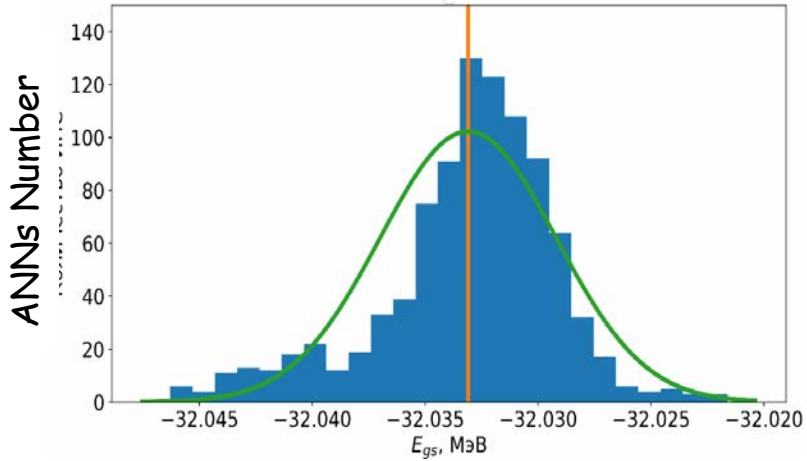


Extrapolation algorithm: selection of ANNs

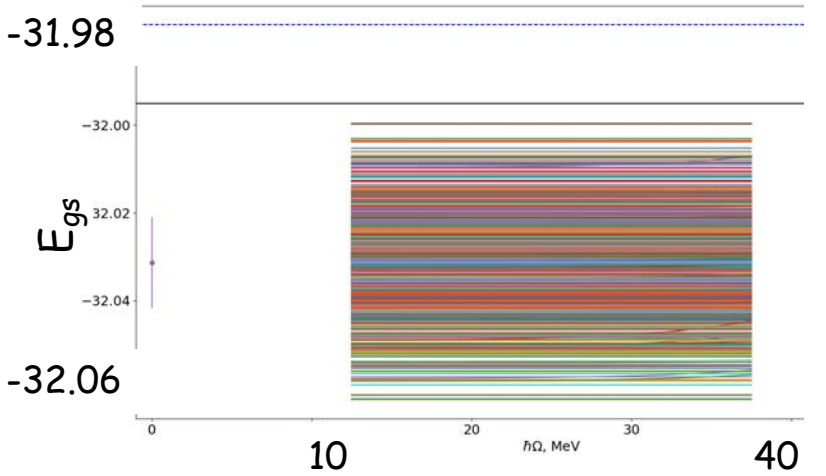
3. 3σ criterion.

The typical distribution results of trained networks can be approximated by gaussoid. The average value of this distribution is associated with the E_{gs} prediction, and the r.m.s. deviation σ is accepted for the prediction uncertainty ΔE. The algorithm described above utilizes the Keras library for the Python.

We neglect networks with results out 3σ interval.



As a **result of selection** from the initial ensemble with **1024** ANNs remains **500-800** networks, which provides a fairly **high statistical significance** of our predictions.

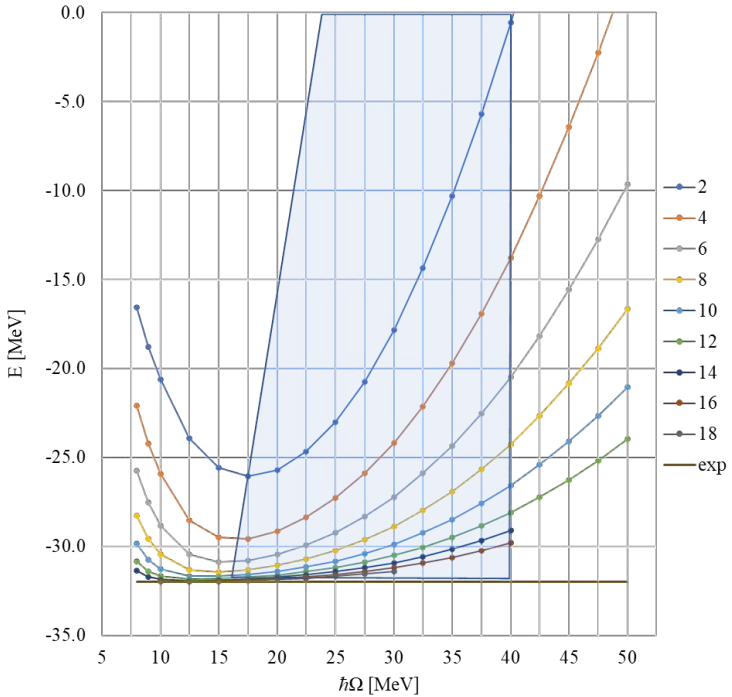


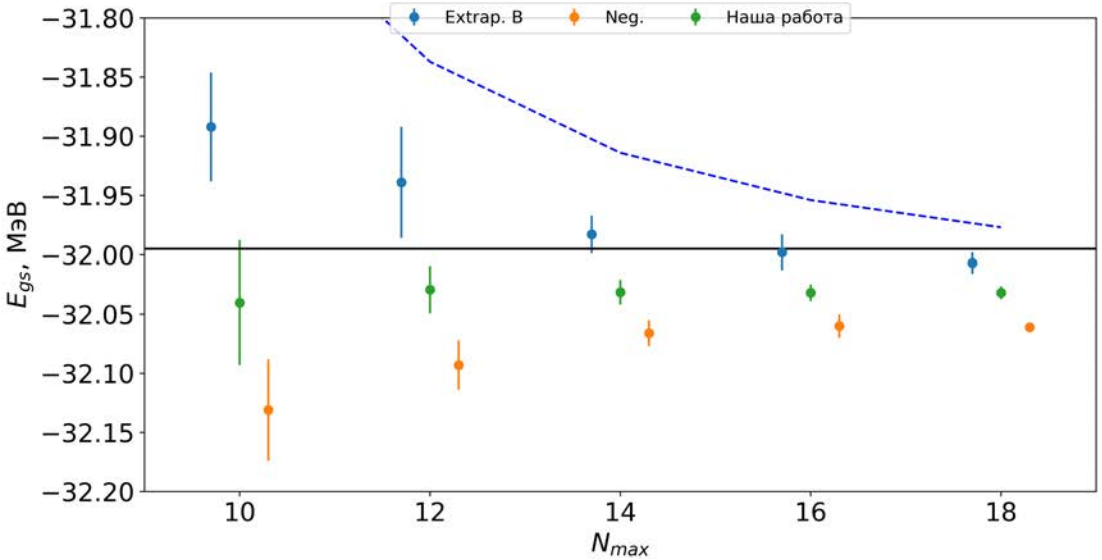
For comparison: in **PRC99(2019)** from **~400 000** trained networks selected only **50** ones.

Results:

^{6}Li, Daejeon16,

$E_{gs\ exp}$ = -31.995 MeV

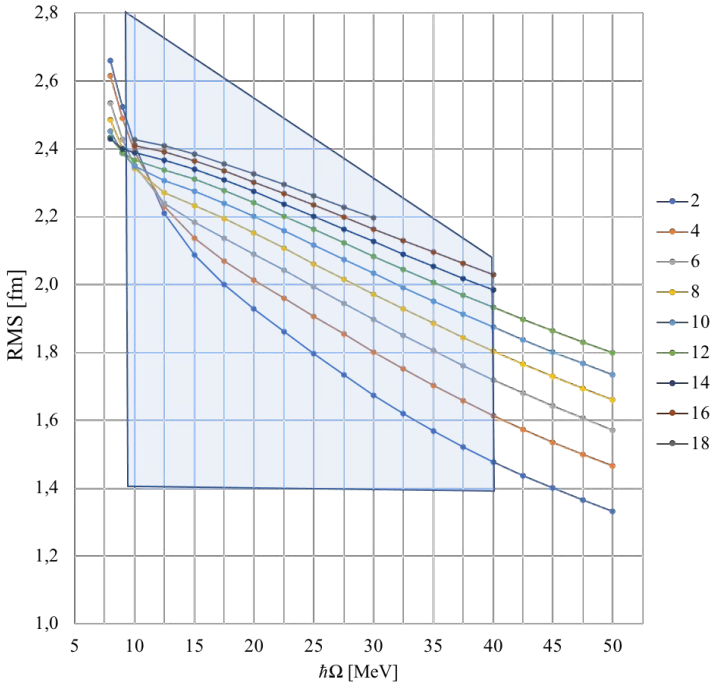


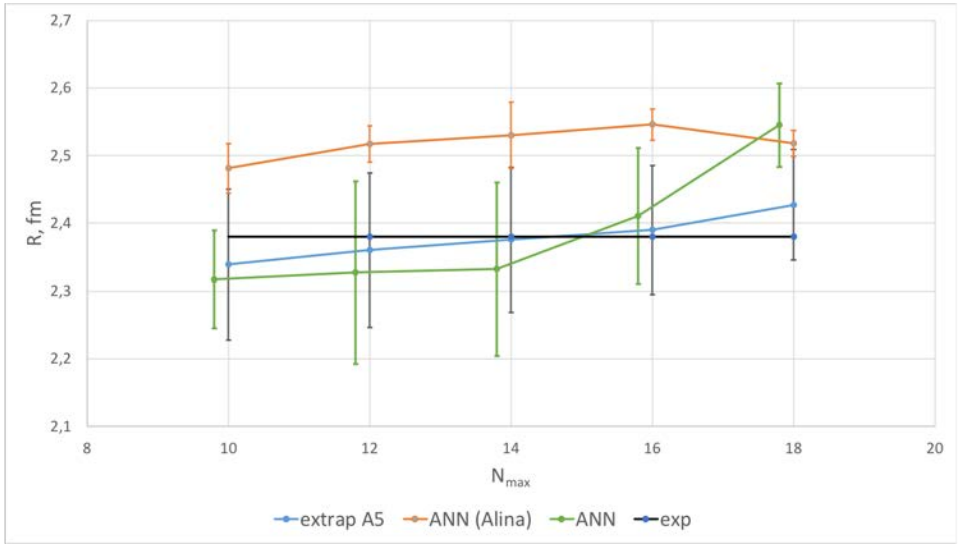


N_{max}	Var. min E_{gs}	exp. extrap. PRC79(2009) $E_{gs}(\Delta E)$	ANN PRC99(2019) $E_{gs}(\Delta E)$	Our results $E_{gs}(\Delta E)$
10	-31.688	$-31.892(46)$	$-32.131(43)$	$-32.040(53)$
12	-31.837	$-31.939(47)$	$-32.093(21)$	$-32.029(20)$
14	-31.914	$-31.983(16)$	$-32.066(11)$	$-32.031(10)$
16	-31.954	$-31.998(15)$	$-32.060(10)$	$-32.032(7)$
18	-31.977	$-32.007(9)$	$-32.061(4)$	$-32.032(5)$

Results:

^{6}Li, Daejeon16,
rms proton point radii
r_{exp} = 2.38(3) fm



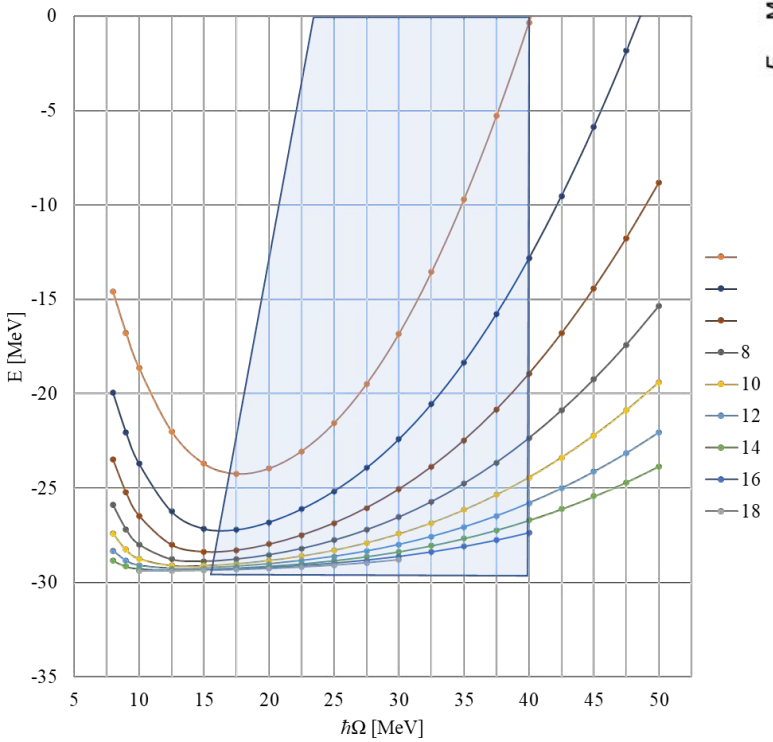


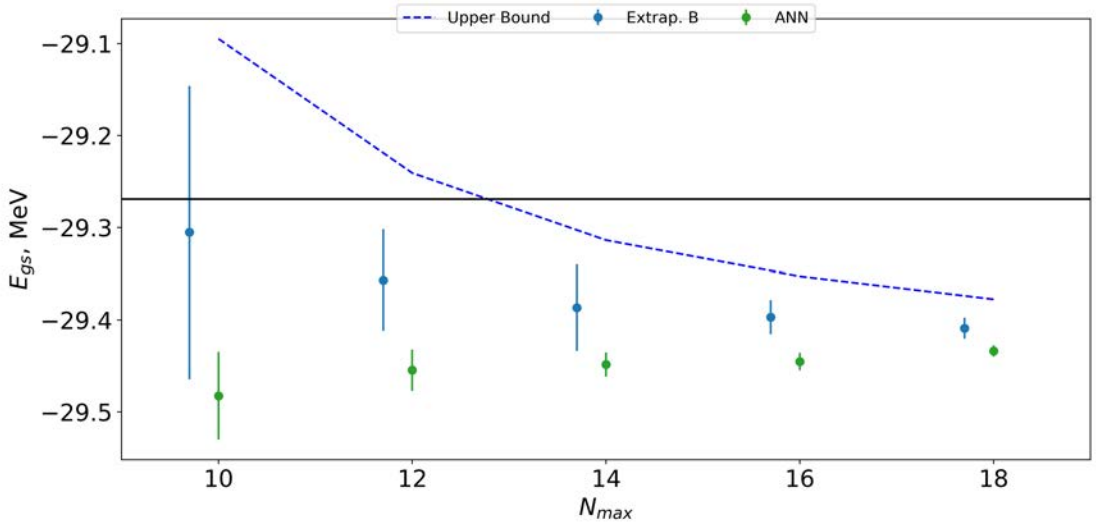
N_{max}	exp. extrap. PRC79(2009) $E_{gs}(\Delta E)$	ANN PRC99(2019) $E_{gs}(\Delta E)$	Our results $E_{gs}(\Delta E)$
10	2.339(111)	2.481(37)	2.317(72)
12	2.360(114)	2.517(27)	2.327(135)
14	2.376(107)	2.530(49)	2.332(128)
16	2.390(95)	2.546(23)	2.411(100)
18	2.427(82)	2.518(19)	2.545(62)

Results:

^{6}He, Daejeon16,

$E_{gs\ exp}$ = -29.269 MeV



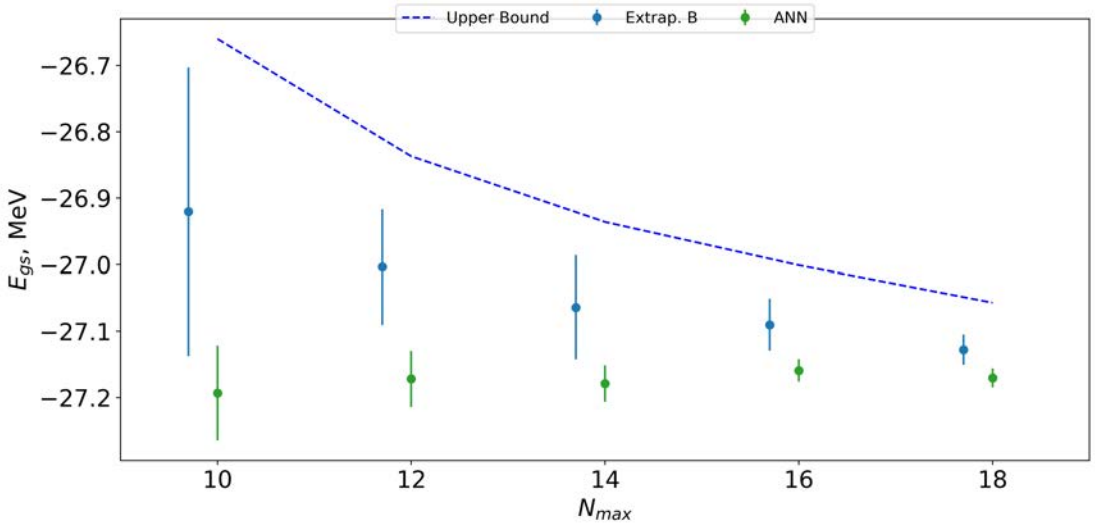


N_{max}	Var. min E_{gs}	exp. extrap. PRC79(2009) $E_{gs}(\Delta E)$	Our results $E_{gs}(\Delta E)$
10	−29.090	−29.305(159)	−29.482(48)
12	−29.241	−29.357(55)	−29.455(23)
14	−29.314	−29.387(47)	−29.448(13)
16	−29.353	−29.397(18)	−29.445(9)
18	−29.378	−29.409(11)	−29.434(6)

Results:

^{6}Be, Daejeon16,

$E_{gs\ exp}$ = -27.697 MeV



N_{max}	Var. min E_{gs}	exp. extrap. PRC79(2009) $E_{gs}(\Delta E)$	Our results $E_{gs}(\Delta E)$
10	−26.660	−26.920(217)	−27.193(71)
12	−26.837	−27.004(87)	−27.172(42)
14	−26.936	−27.064(78)	−27.179(27)
16	−27.001	−27.090(39)	−27.159(17)
18	−27.057	−27.128(23)	−27.170(14)

THANK YOU
for YOUR ATTENTION

THANK TO ORGANIZERS OF NTSE-2023 WORKSHOP for:

1. Invitation;
2. Hospitality;
3. Possibility to present my talk;
4. Excellent excursion;
5. …