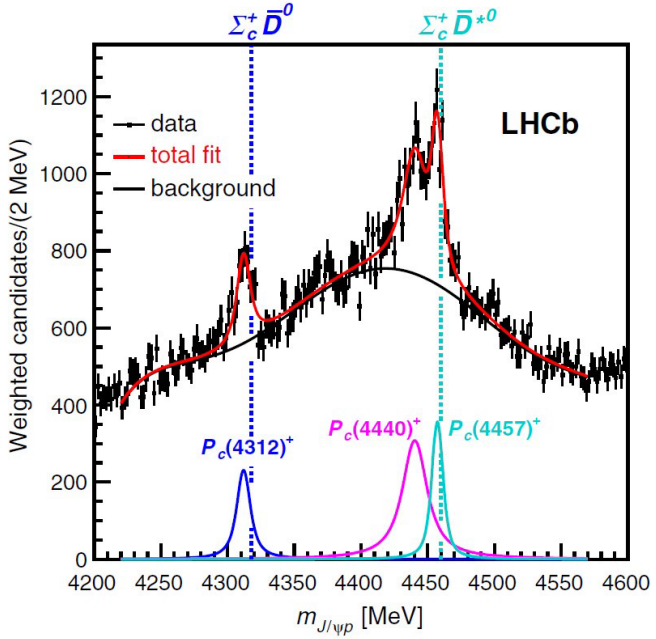

# Probing the pole configuration of scattering amplitude using deep learning

*arXiv:2105.04898 (accepted in PRD)*
*arXiv:2104.14182*

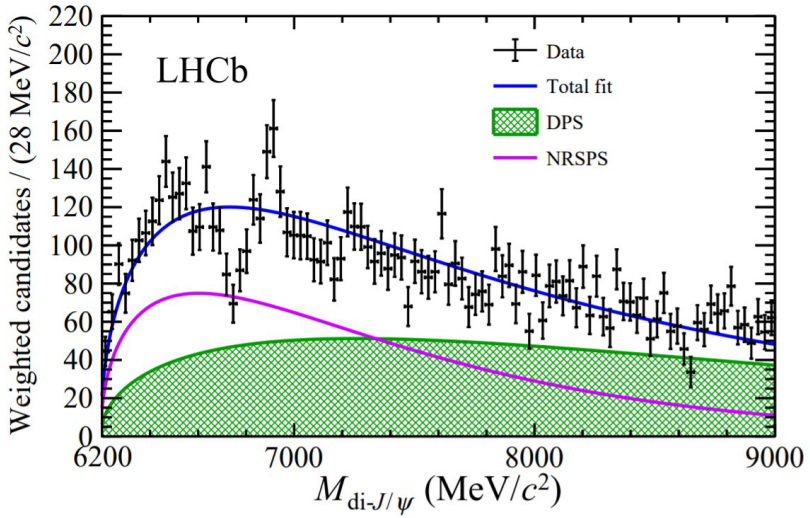
Denny Lane B. Sombillo

In collaboration with:
Yoichi Ikeda (Kyushu University)
Toru Sato (RCNP, Osaka University)
Atsushi Hosaka (RCNP, Osaka University and ASRC, JAEA)

# Motivation and Overview




10.1103/PhysRevLett.122.222001;
arXiv:1904.03947




10.1016/j.scib.2020.08.032;
arXiv:2006.16957

Explanation of the observed near-threshold/ threshold structures

- Threshold cusp
- Molecular state
- Virtual state
- Compact state

How to tell if a near-threshold structure is caused by a physical state?

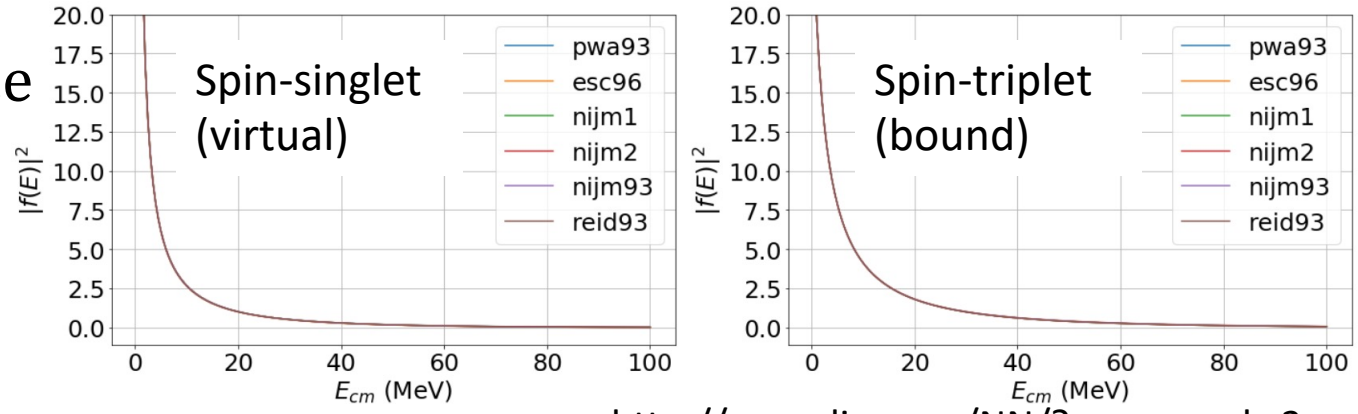
- Phrase the question as a classification problem.
- **Deep learning approach** excels in solving a classification problem.

Deep learning approach: Use of **data** (simulated or real) to **improve the performance** of a **model** in accomplishing a specific **task**.

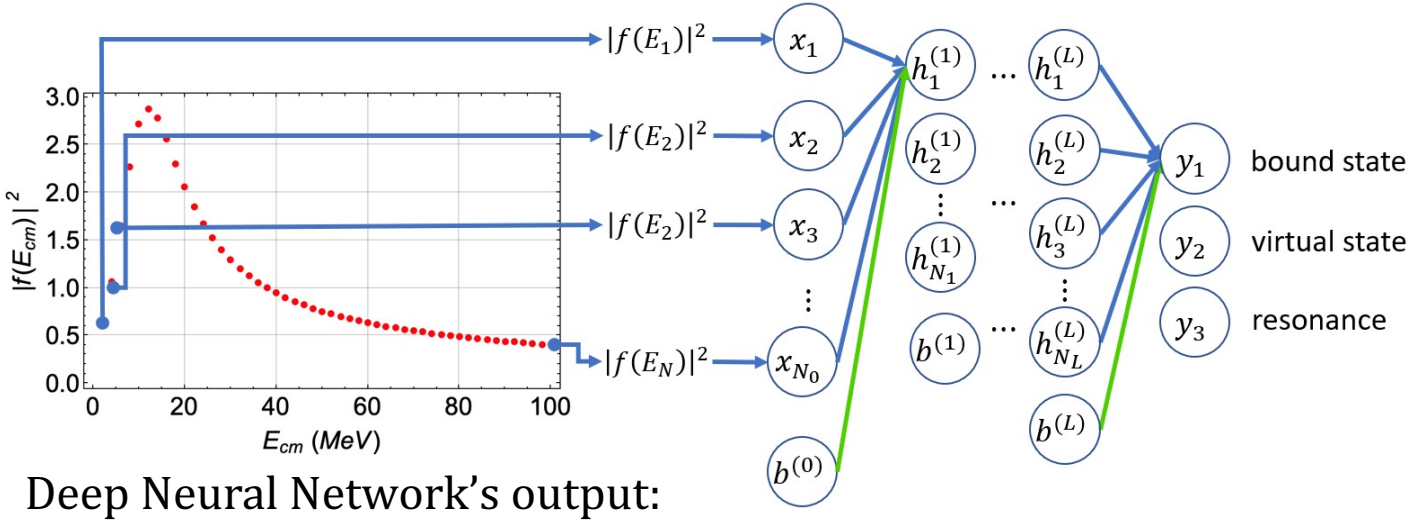
# Motivation and Overview

DNN to distinguish bound-virtual-resonance in single-channel cross-section

DLB Sombillo, Y Ikeda, T Sato, A Hosaka
10.1103/PhysRevD.102.016024; arXiv:2003.10770
https://github.com/sombillo/DNN-for-bound-virtual-classification




http://nn-online.org/NN/?page=nnphs2




Deep Neural Network's output:

| | PWA93 | ECS96 | NijmI | NijmII | Nijm93 | Reid93 |
|---|---|---|---|---|---|---|
| $^1S_0$ | virtual | virtual | virtual | virtual | virtual | virtual |
| $^3S_1$ | bound | bound | bound | bound | bound | bound |

**Proof of principle**

- Generic properties of S-matrix is sufficient to generate the training dataset.
- Threshold enhancements can be studied using data science approach.

# Motivation and Overview

Extension to coupled-channel problem

Complete classification requires a model.

A. M. Badalyan et.al., Phys. Rep. 1982

Information that can be obtained without using a model:

**Pole configuration**: number of nearby poles in each Riemann sheet

pole-counting method

D. Morgan, Nucl. Phys. A, 543,4,1992

two-pole structure of $\Lambda(1405)$

PhysRevC.68.018201; arXiv:0212026

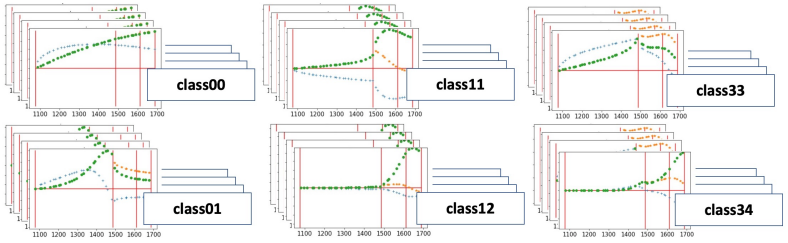
**Pole configuration** can be used to construct an appropriate parametrization.

# Motivation and Overview

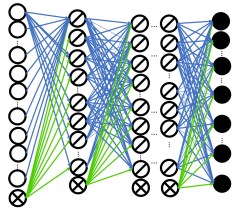
Proposed model-independent deep learning analysis:

## Generating the training dataset

- Use general properties of S-matrix
- Include energy uncertainty

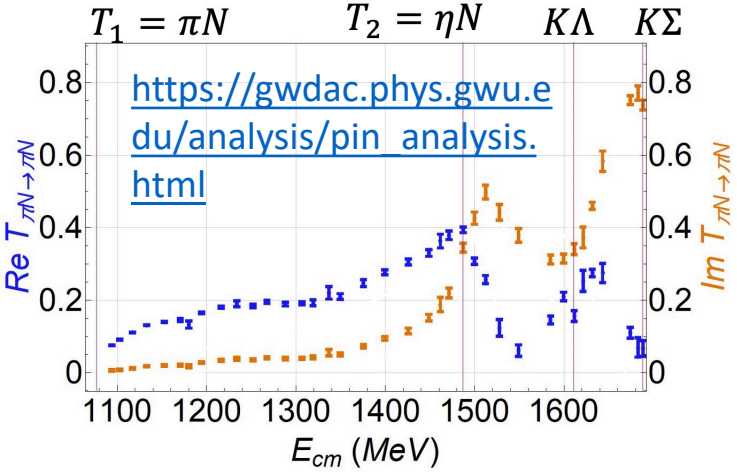


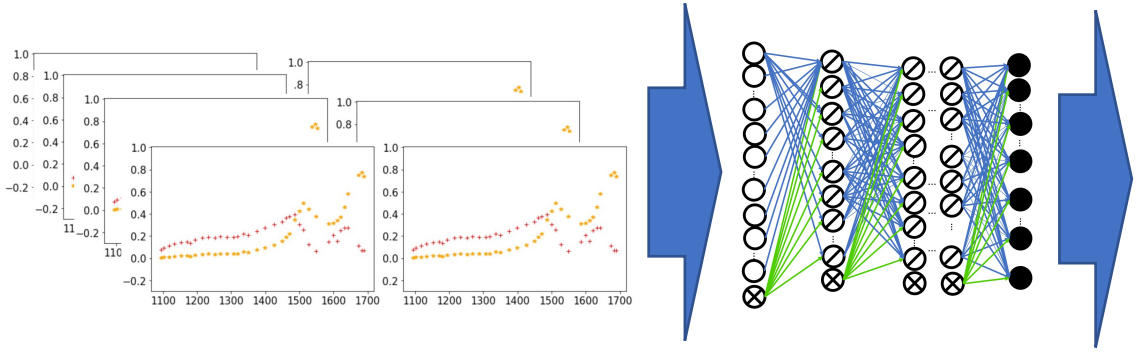

## Optimization of deep neural network (DNN) model



## Apply trained-DNN on experimental data

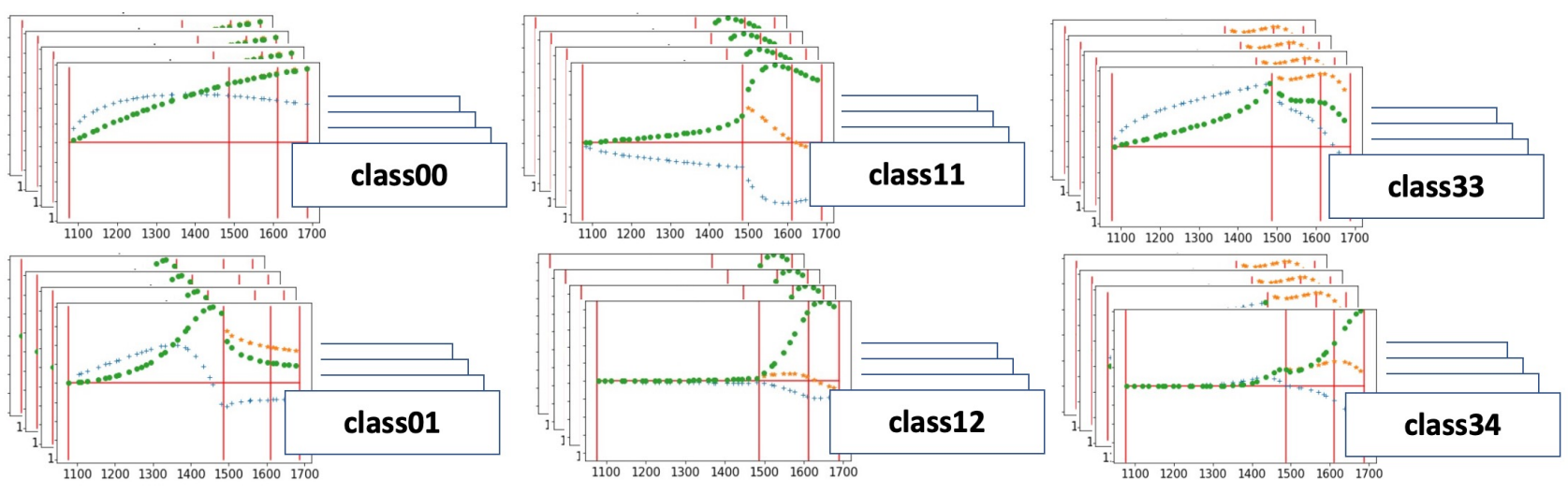
- Use error bars in the data to generate inference amplitudes



Count the number of outcomes

# Generation of training dataset



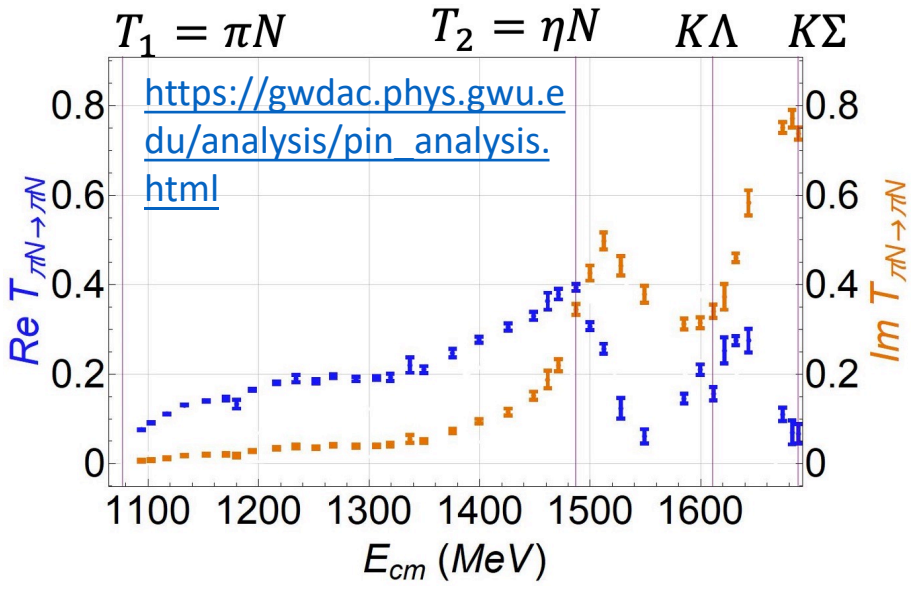
# Generation of training dataset

General form of S-matrix:

- Hermitian below the lowest threshold
- Unitarity
- Analyticity

https://doi.org/10.1063/1.1703698
https://doi.org/10.1098/rspa.1960.0096

$$S_{11}(p_1,p_2)=\prod_m \frac{D_m(-p_1,p_2)}{D_m(p_1,p_2)}; \qquad S_{11}=1+2iT_{11}$$




$$S_{22}(p_1,p_2)=\prod_m \frac{D_m(p_1,-p_2)}{D_m(p_1,p_2)} \qquad S_{22}S_{11}-S_{12}^2=\prod_m \frac{D_m(-p_1,-p_2)}{D_m(p_1,p_2)}$$

The available experimental data will determine the relevant S-matrix element.

Ensure that one $D_m(p_1,p_2)$ will produce one pole (conjugate pair).

# Generation of training dataset

How to control the pole configuration?

- Assign the pole position: $E_{\text{pole}}^{(m)} = E_R^{(m)} \pm iE_I^{(m)}$
- Control the Riemann sheet:

$$D_m(p_1, p_2) = \left[\left(p_1 - i\beta_1^{(m)}\right)^2 - \alpha_1^{(m)2}\right] + \lambda^{(m)}\left[\left(p_2 - i\beta_2^{(m)}\right)^2 - \alpha_2^{(m)2}\right] = 0$$

Absolute values of $\alpha_1^{(m)}, \alpha_2^{(m)}, \beta_1^{(m)}, \beta_2^{(m)}$ are determined by $E_{\text{pole}}^{(m)} = \frac{p_i^2}{2\mu_i} + T_i$

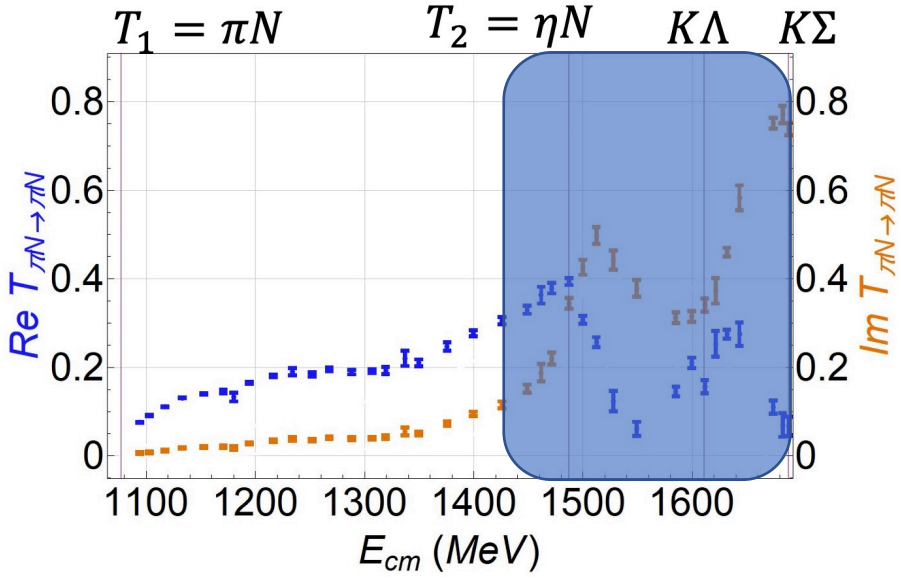
Hermiticity is automatically satisfied.

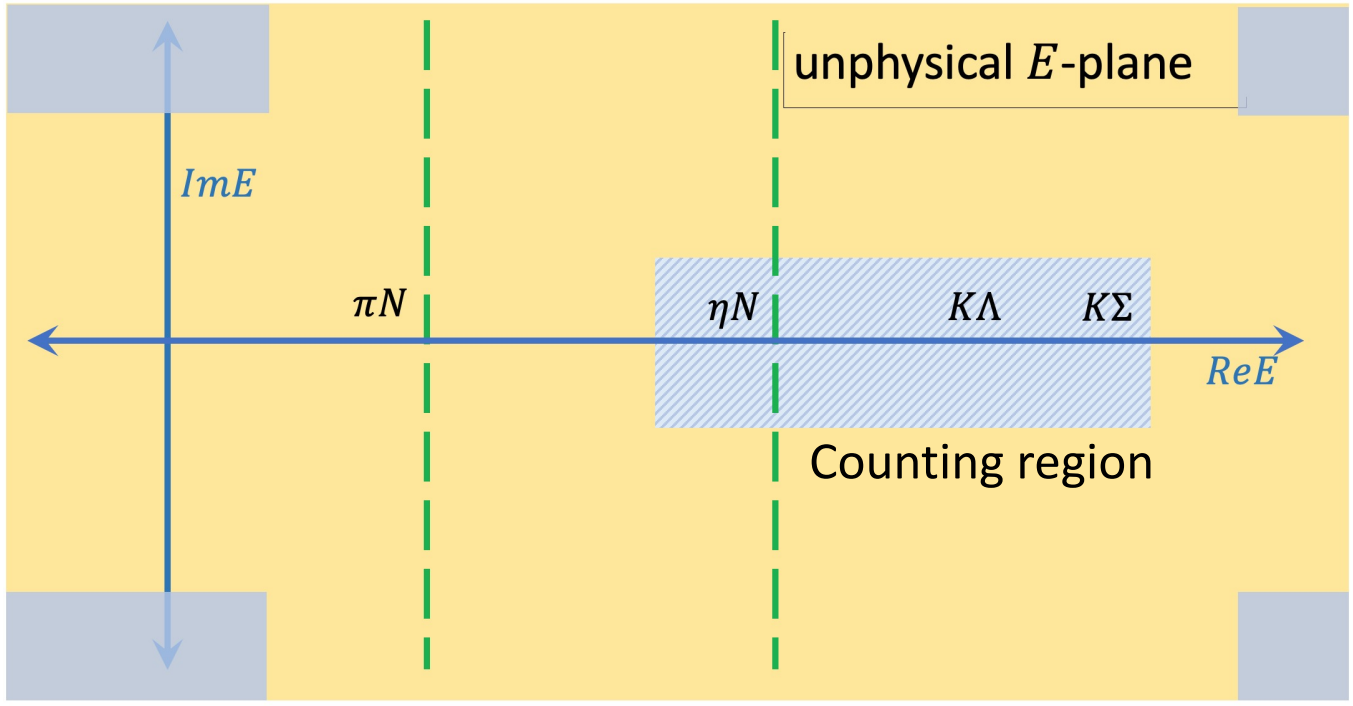
Analyticity – do not choose $\beta_1^{(m)}, \beta_2^{(m)} > 0$ simultaneously

- Use $\lambda^{(m)}$ to ensure that only one pole per $D_m(p_1, p_2)$

# Generation of training dataset

We limit the region of analysis.




Region where poles are generated and counted:

$$\eta N - 50 \le Re\, E_{pole} \le \eta N + 200$$
$$-200 \le Im\, E_{pole} \le 200$$

Region where poles are generated but not counted:

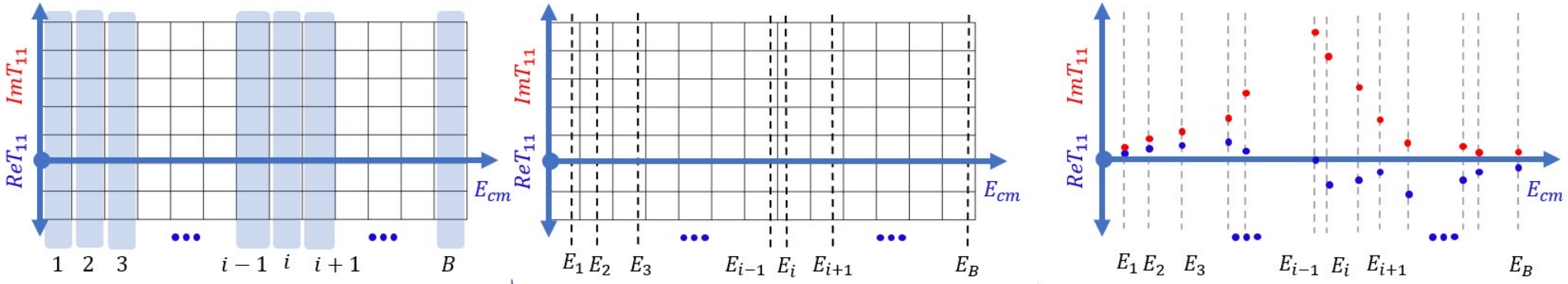
$$Re\, E_{pole} < \pi N - 100 \quad and \quad Re\, E_{pole} > \eta N + 500$$
$$700 < \left| Im\, E_{pole} \right| < 2000$$

$$S_{11}(p_1, p_2) = \prod_j \frac{D_j(-p_1, p_2)}{D_j(p_1, p_2)} \prod_n \frac{D_n(-p_1, p_2)}{D_n(p_1, p_2)}$$

We limit the maximum number of counted poles to 4.

# Generation of training dataset

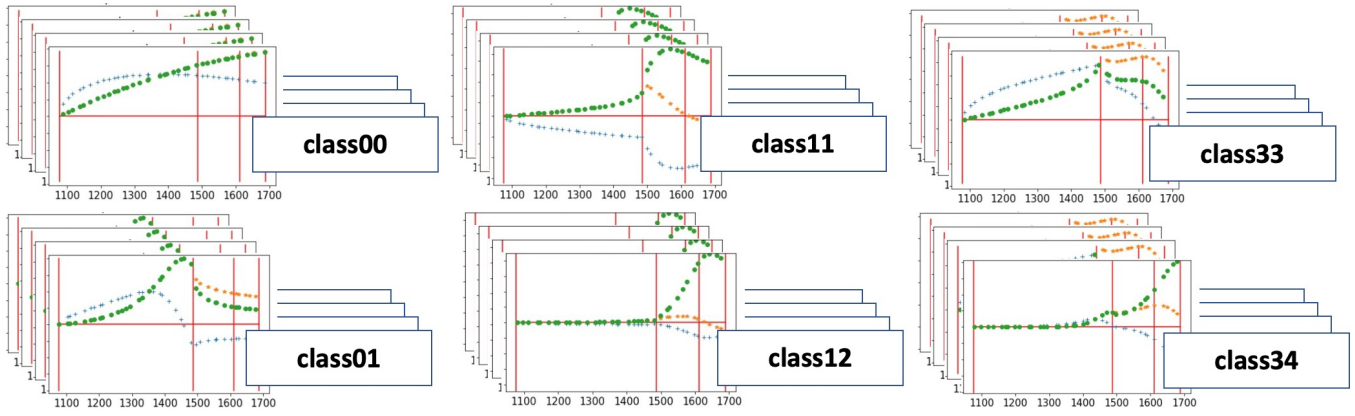
To simulate the limited energy resolution.




(1) Divide the energy range into $B$ bins

(2) Choose a random energy value in each bin

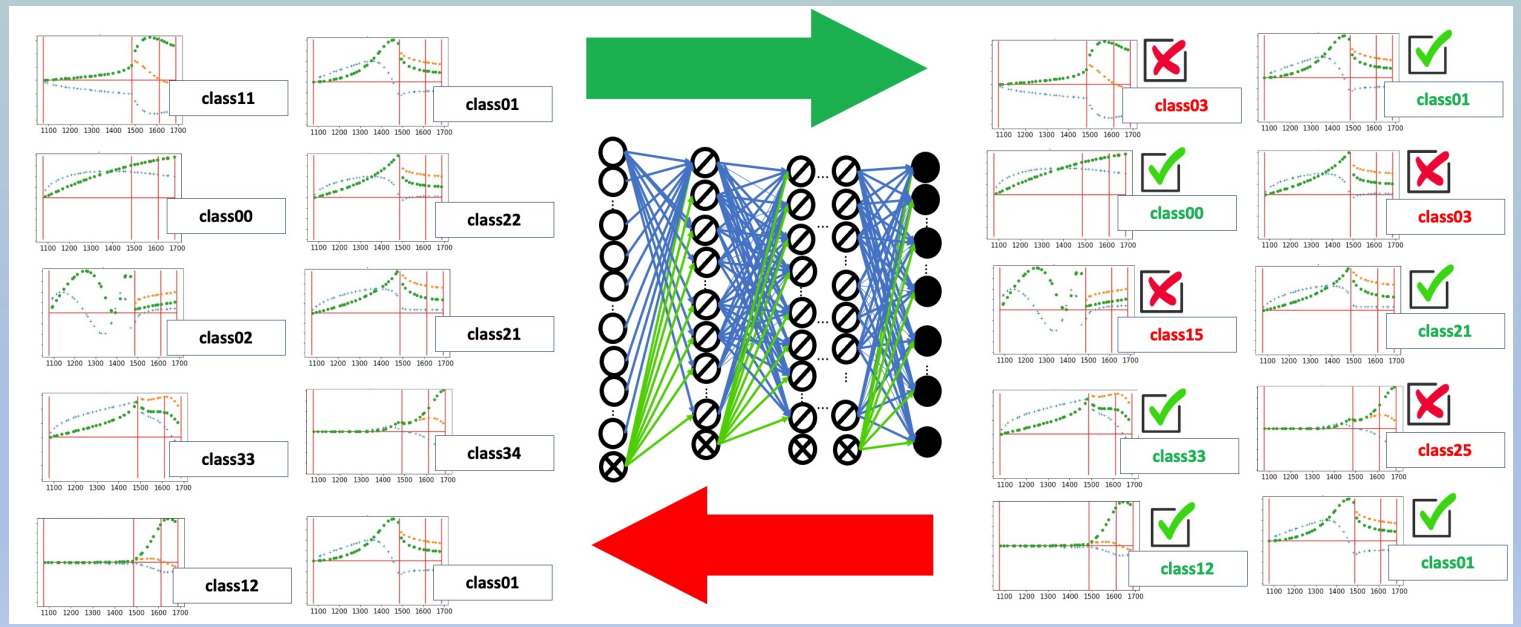
(3) Calculate the amplitude for each random energy value



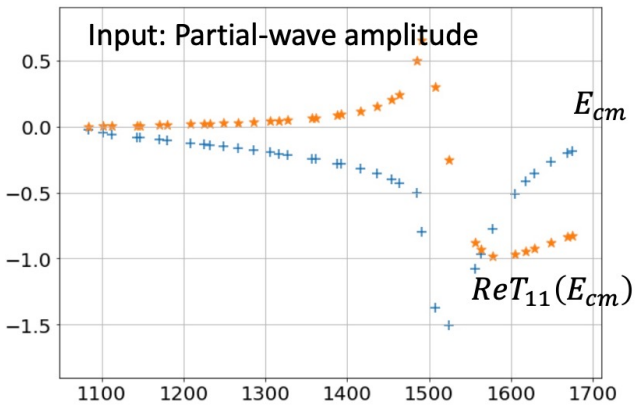

(4) Label each amplitude according to its pole-configuration

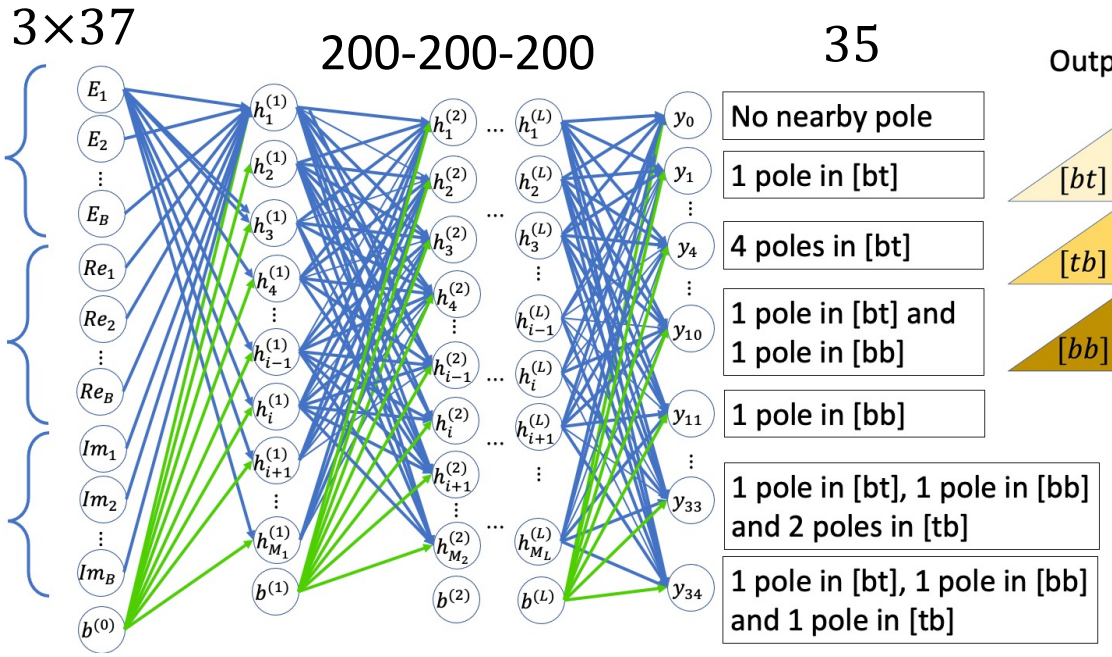
| Label | S-matrix pole configuration |
|---|---|
| 0 | no nearby pole |
| 1 | 1 pole in $[bt]$ |
| 2 | 2 poles in $[bt]$ |
| ⋮ | ⋮ |
| 32 | 1 pole in $[bt]$, 2 poles in $[bb]$ and 1 pole in $[tb]$ |
| 33 | 1 pole in $[bt]$, 1 pole in $[bb]$ and 2 poles in $[tb]$ |
| 34 | 1 pole in $[bt]$, 1 pole in $[bb]$ and 1 pole in $[tb]$ |

# Optimization of DNN model

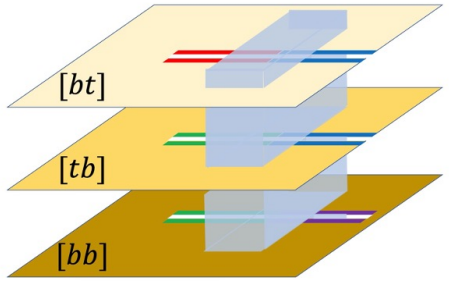


# Optimization of DNN model

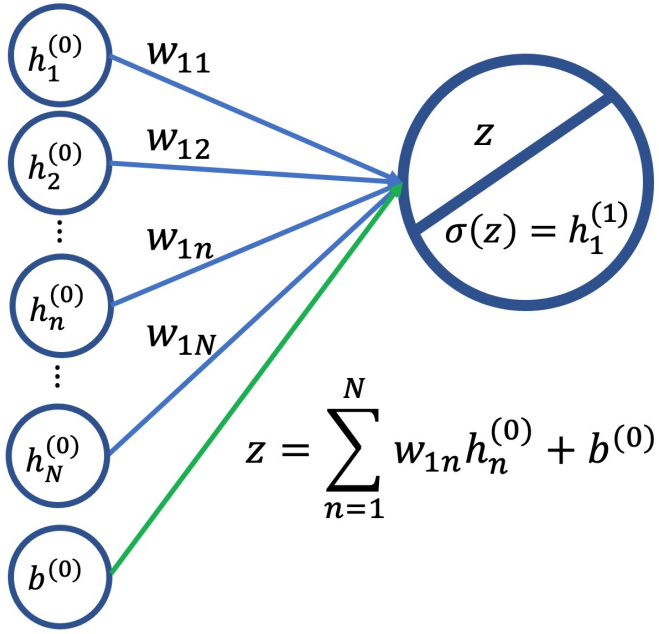



Output: Pole configuration

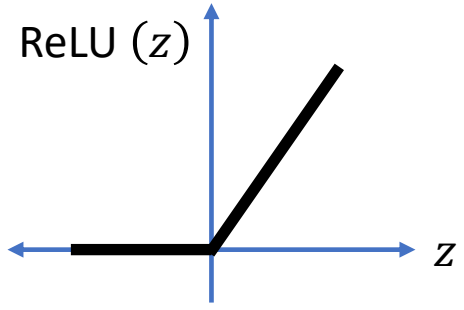



Chainer
https://chainer.org/




$$z = \sum_{n=1}^{N} w_{1n} h_n^{(0)} + b^{(0)}$$

For hidden-layer nodes:



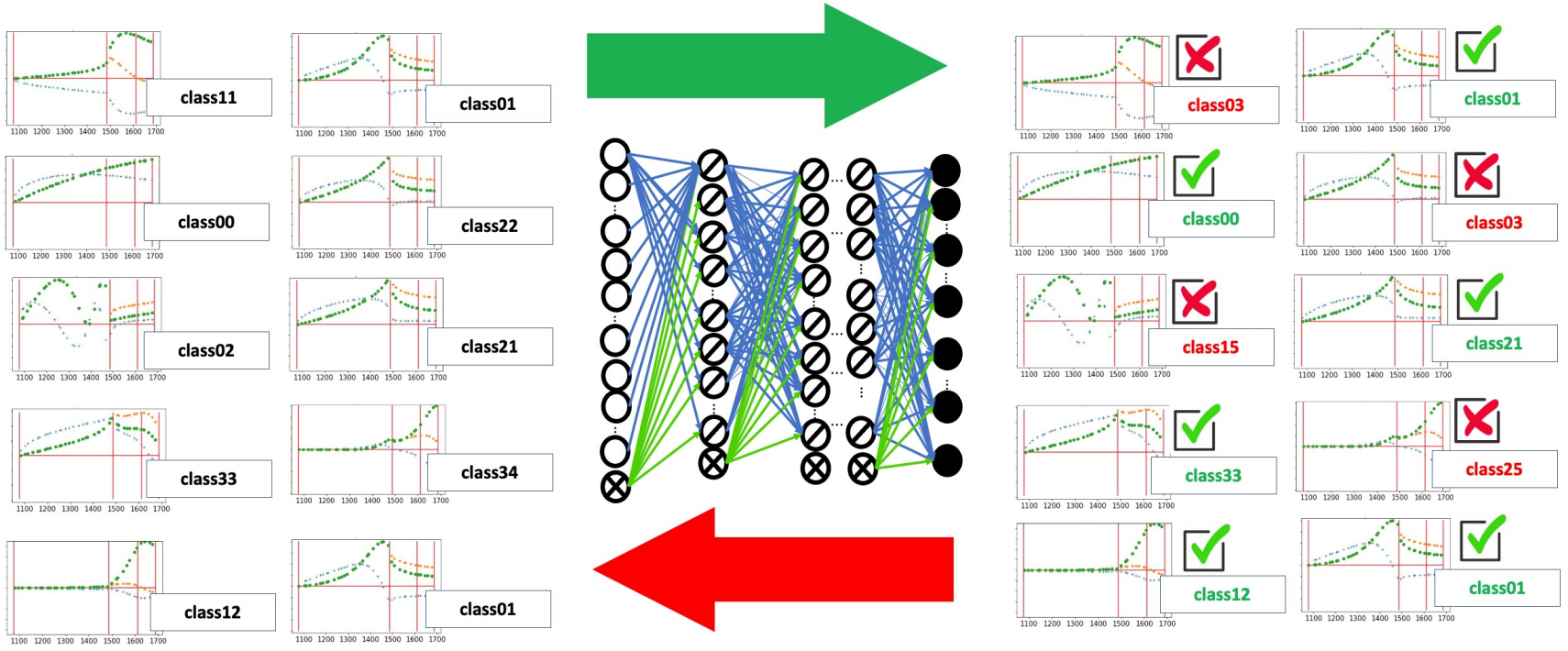

For output nodes:

$$\mathrm{SoftMax}\left(z_n^{\text{output−layer}}\right) = \frac{\exp\left(z_n^{\text{output−layer}}\right)}{\sum_{m=1}^{35} \exp\left(z_m^{\text{output−layer}}\right)}$$

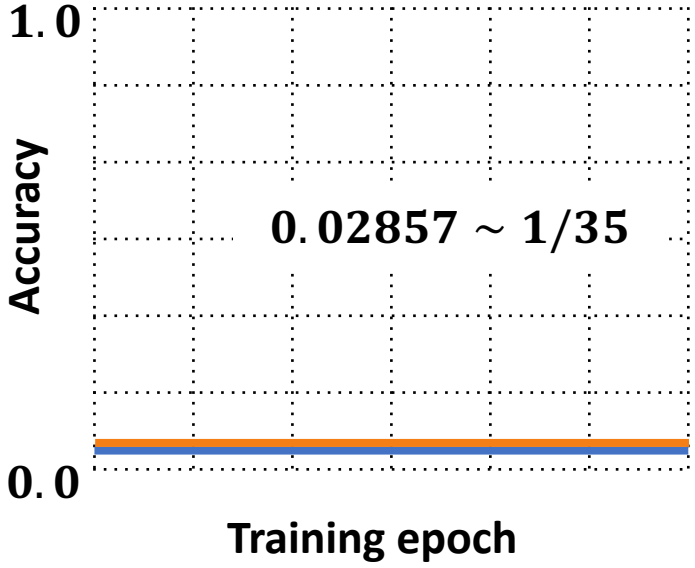
Optimization: Find optimal values of weights and biases.

# Optimization of DNN model

(1) Forward pass
(estimate the cost function)




(2) Backpropagation
(update weights and biases)




The DNN is just guessing.

# Optimization of DNN model

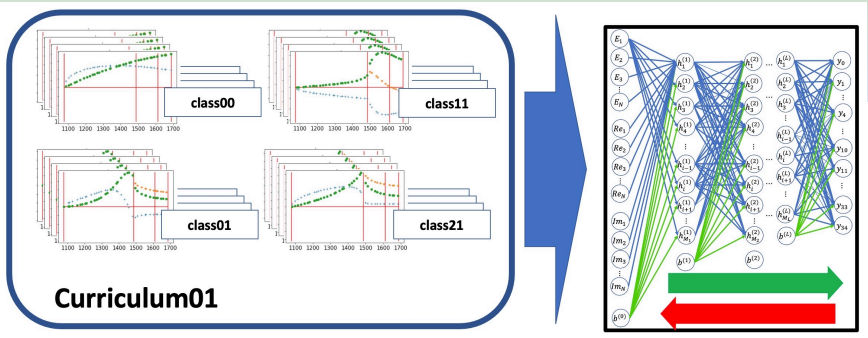
- Start with small easy examples.
  Elman 1993
- Slowly introduce new class until all examples are presented.
- Perform regular training loop

Chosen DNN architecture

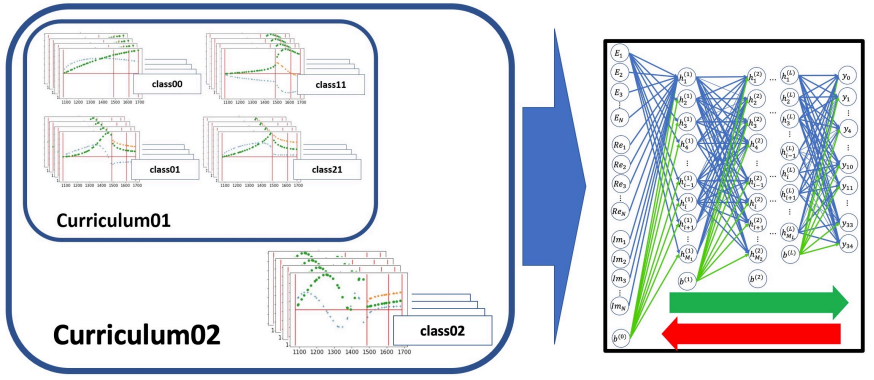
| Layer | Number of nodes | Activation Function |
|---|---|---|
| Input | 111+1 | |
| 1st | 200+1 | ReLU |
| 2nd | 200+1 | ReLU |
| 3rd | 200+1 | ReLU |
| Output | 35 | Softmax |

arXiv:2105.04898
arXiv:2104.14182

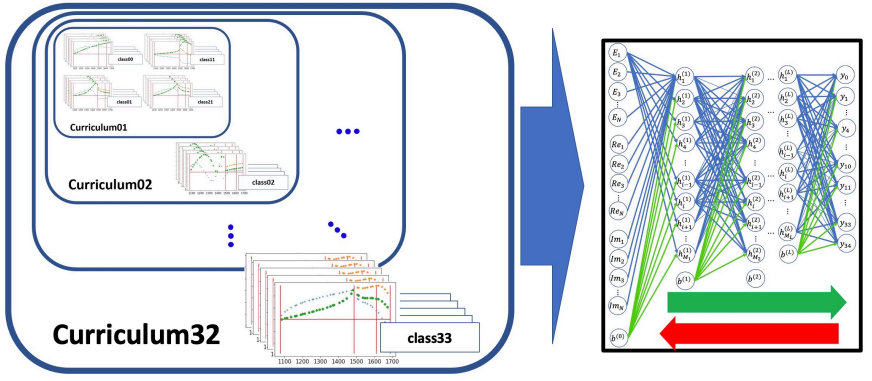
**Curriculum01:**
At-most-**one**-pole classification



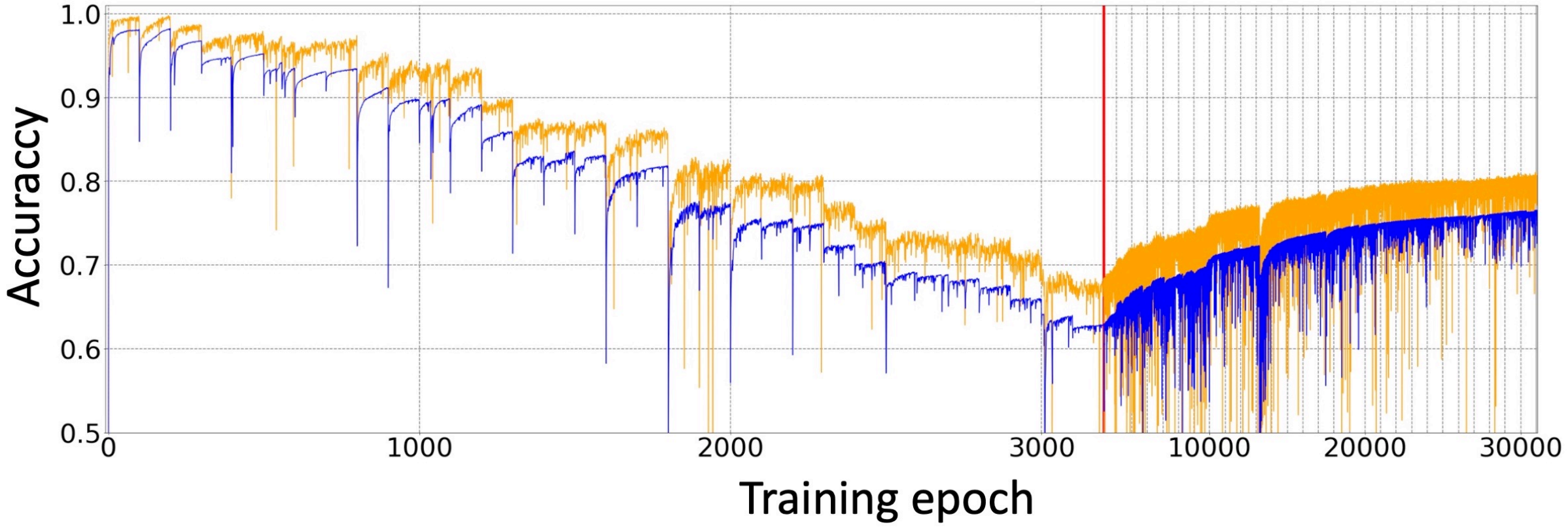

**Curriculum02:**
Curriculum01 +
1 new class in
the **two**-pole
classification set




**Curriculum32:**
Curriculum31 +
last class in
the **four**-pole
classification set



# Optimization of DNN model

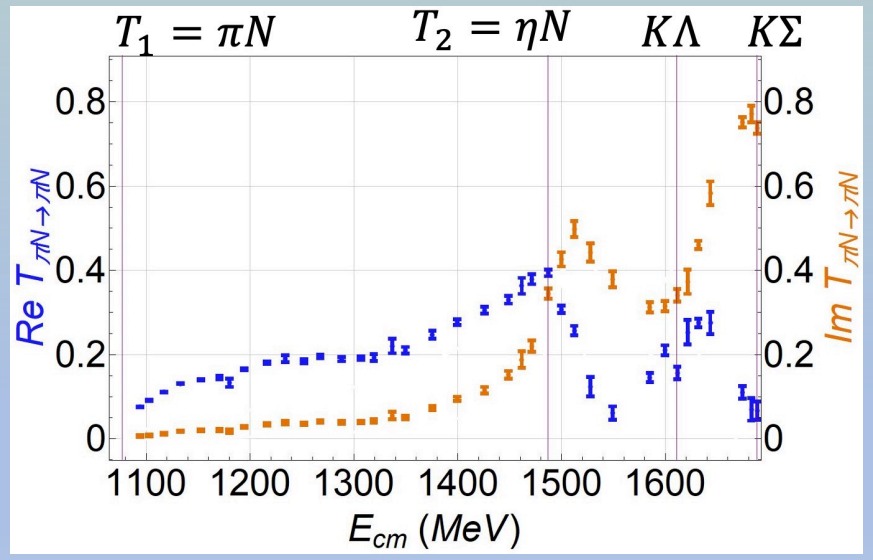


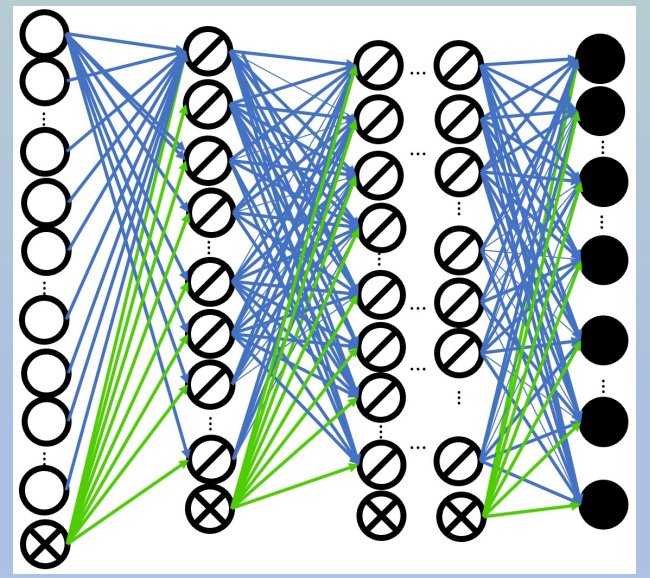

Random model accuracy (wild guessing): 2.86%

Post-curriculum training accuracy: 76.5%
Post-curriculum testing accuracy: 80.4%

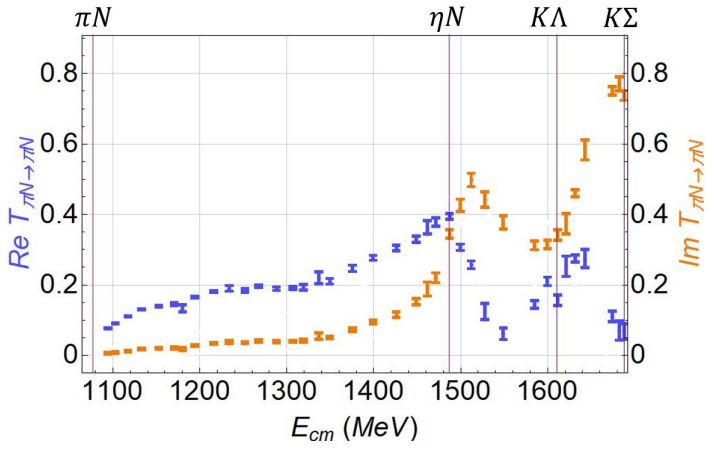
We now have a DNN that can detect up to four poles on any Riemann sheet.

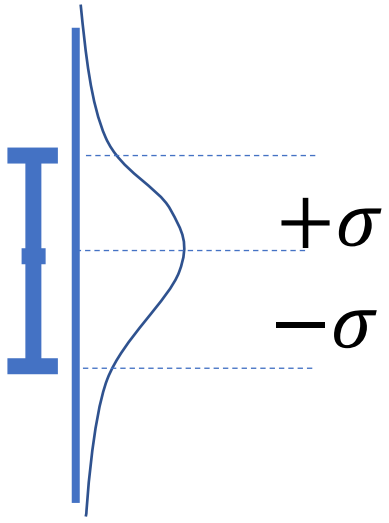
# Inference stage: Application

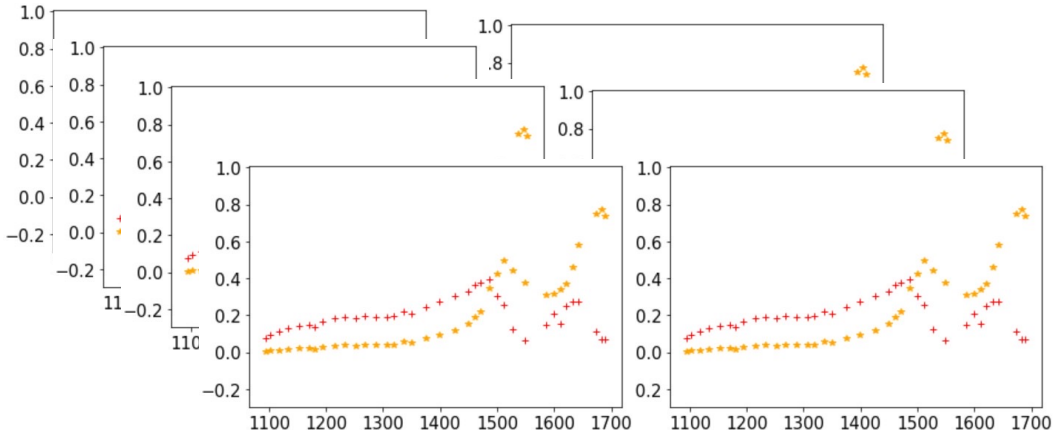


# Inference stage: Application



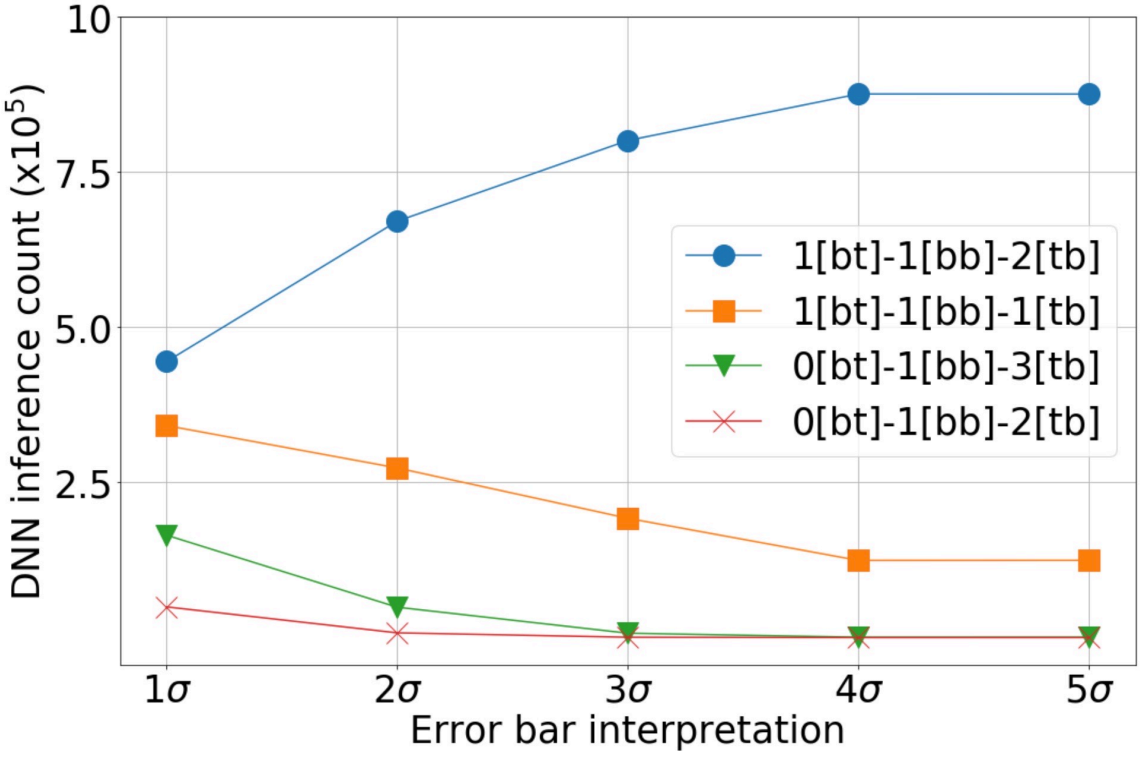
- Pick random points in each error bar
- No model-fitting step is needed
- Generate $10^6$ amplitudes
- Feed to the trained DNN
- Count the number of outcomes

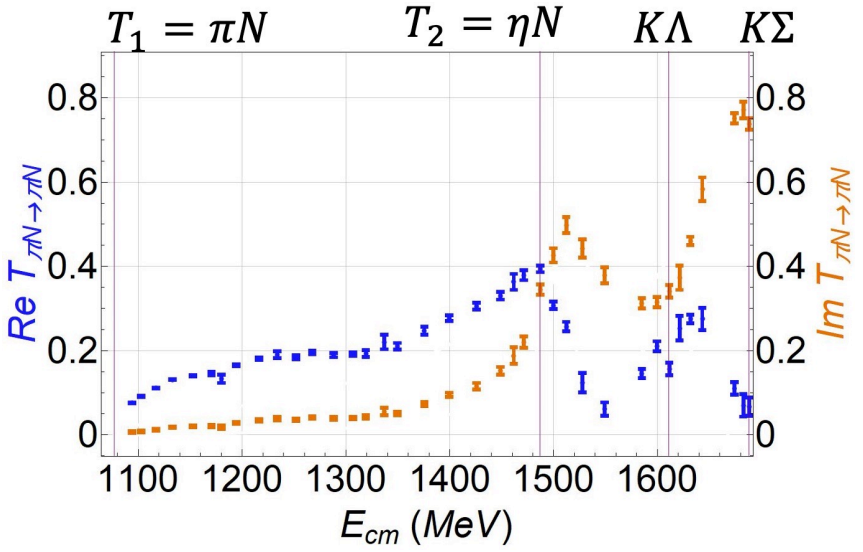
DNN inference on 10^6 amplitudes using 1 error bar = $1\sigma$

- 44.6% 1[bt]-1[bb]-2[tb]
- 34.1% 1[bt]-1[bb]-1[tb]
- 16.4% 0[bt]-1[bb]-3[tb]
- 4.9% 0[bt]-1[bb]-2[tb]

# Inference stage: Application

DNN inference on 10^6 amplitudes using Gaussian distribution for each error bar

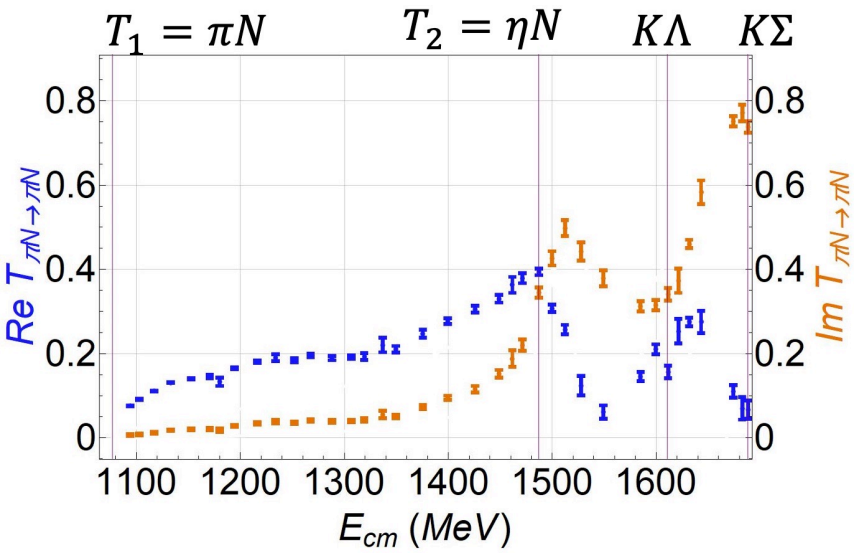



DNN inference on 10^6 amplitudes using uniform distribution for each error bar

- 60.3% 1[bt]-1[bb]-2[tb]
- 30.9% 1[bt]-1[bb]-1[tb]
- 7.5% 0[bt]-1[bb]-3[tb]
- 1.3% 0[bt]-1[bb]-2[tb]

# Inference stage: Application




DNN inference on 10^6 amplitudes using uniform distribution for each error bar

- 60.3% 1[bt]-1[bb]-2[tb]
- 30.9% 1[bt]-1[bb]-1[tb]
- 7.5% 0[bt]-1[bb]-3[tb]
- 1.3% 0[bt]-1[bb]-2[tb]

Guide to parametrization

- Detected [bb] pole
  - Enhancement between $K\Lambda$ and $K\Sigma$ thresholds.
- Detected [bt] pole
  - Far from $\eta N$ threshold but within the counting region
  - Might be shadow of the detected [bb]
- Detected [tb] poles
  - Only poles left to explain $\eta N$ enhancement

# Summary and Outlook

Summary:

- Generate training dataset using the general properties of S-matrix.
- Optimization of DNN model with noisy dataset (energy resolution).
- DNN inference stage
  - No assumed functional form for the amplitude
  - No assumptions made on the detected poles

Use the DNN result to design an appropriate parametrization

# Summary and Outlook

Outlook:

- Multi-DNN analysis
  - Pole configuration
  - Pole positions
- Inclusion of models to trace the pole's origin

- DNN applicable to any two-hadron scattering
- DNN to distinguish resonance and kinematical enhancements

Thank you for listening.