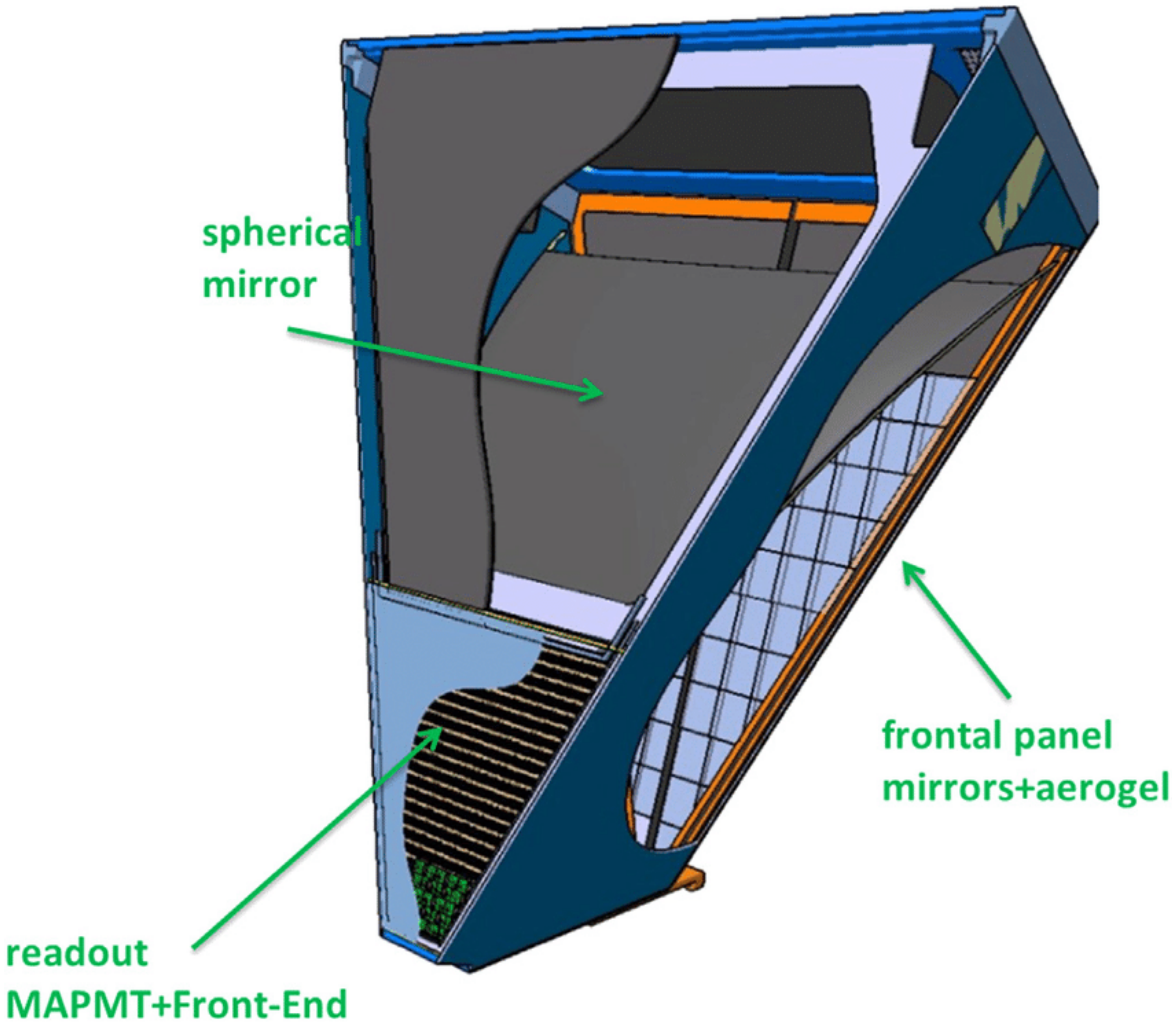
Machine Learning for RICH

Gagik Gavalian (Jefferson Lab)

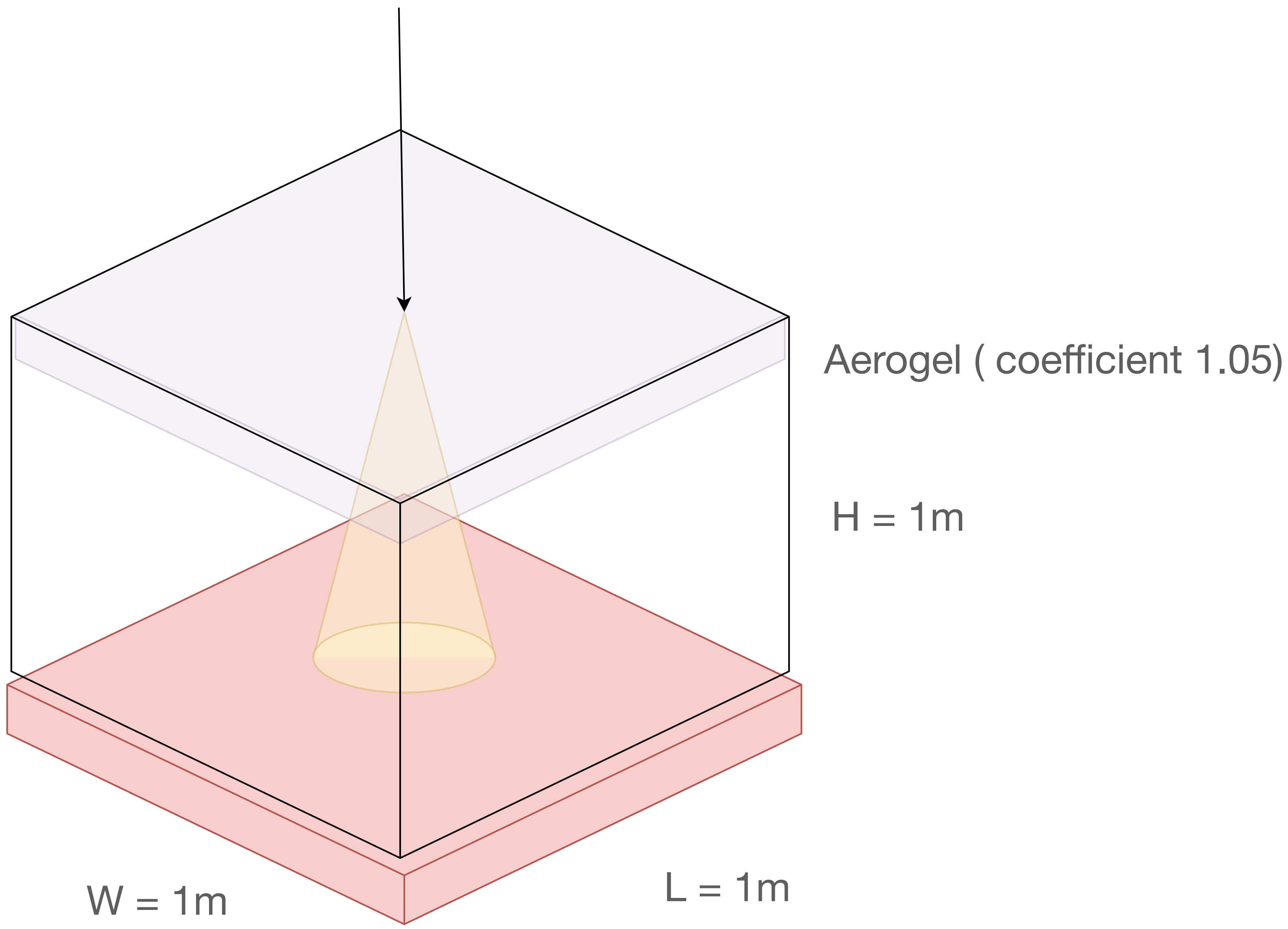
Mini Workshop for Kaon Physics (December 16,2022)

Outline

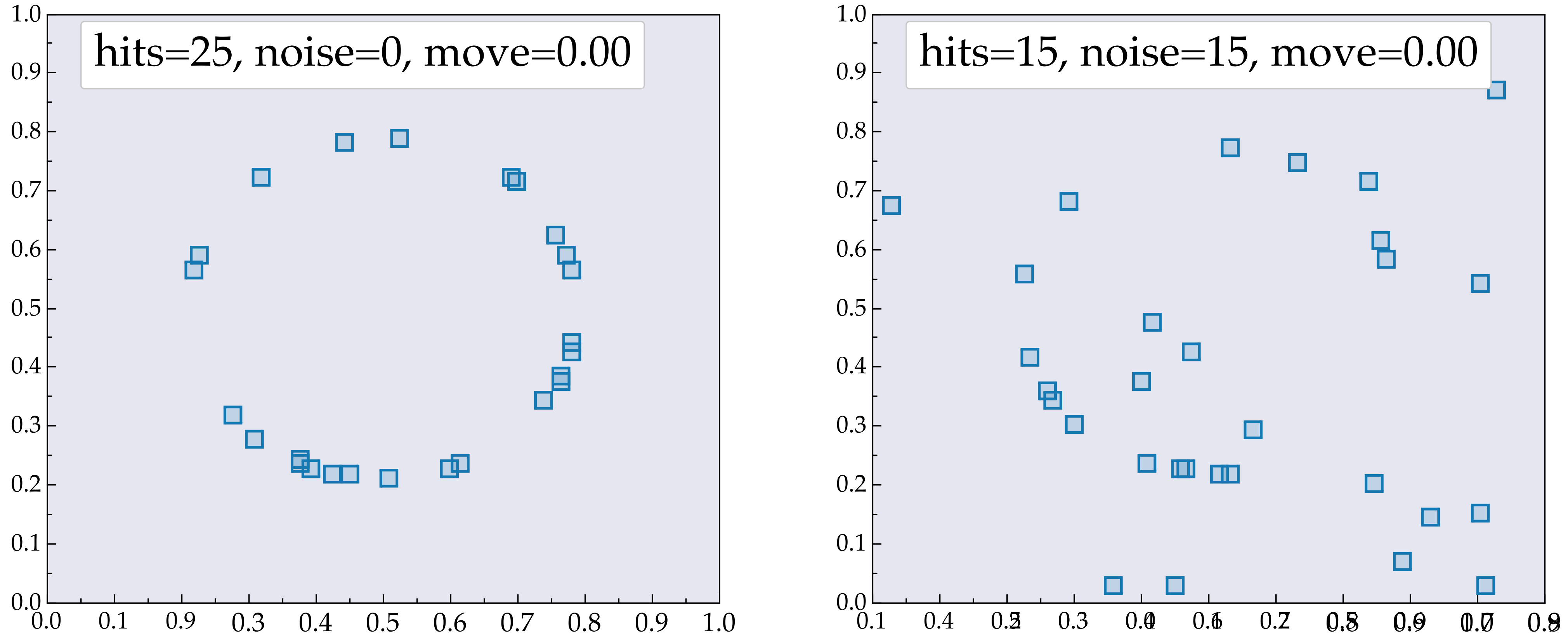
- Preliminary studies
- Neural Network Architecture
- Computational Graph
- Generative Adversarial Networks (GANs)
- Future Plans

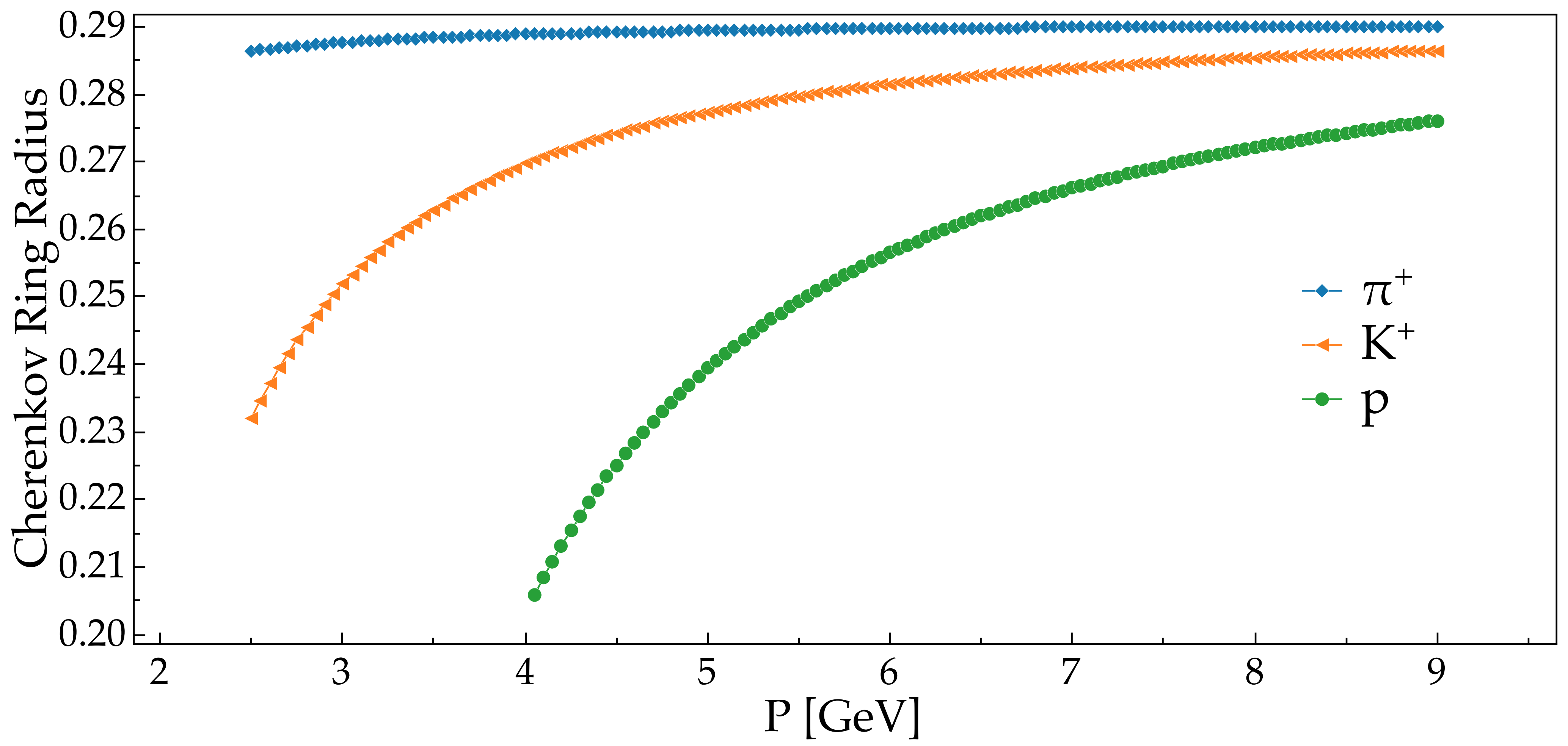


RICH Detector

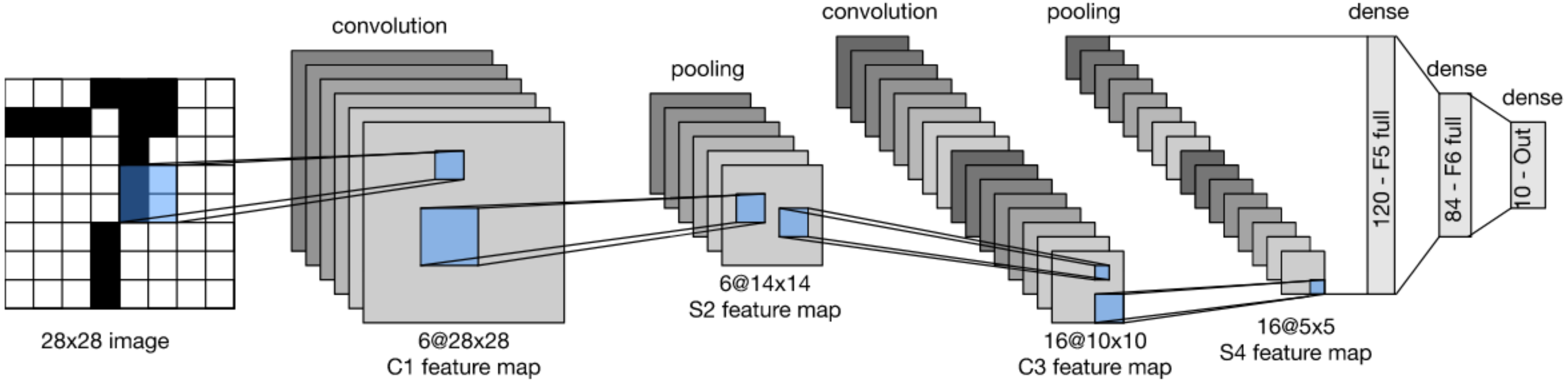


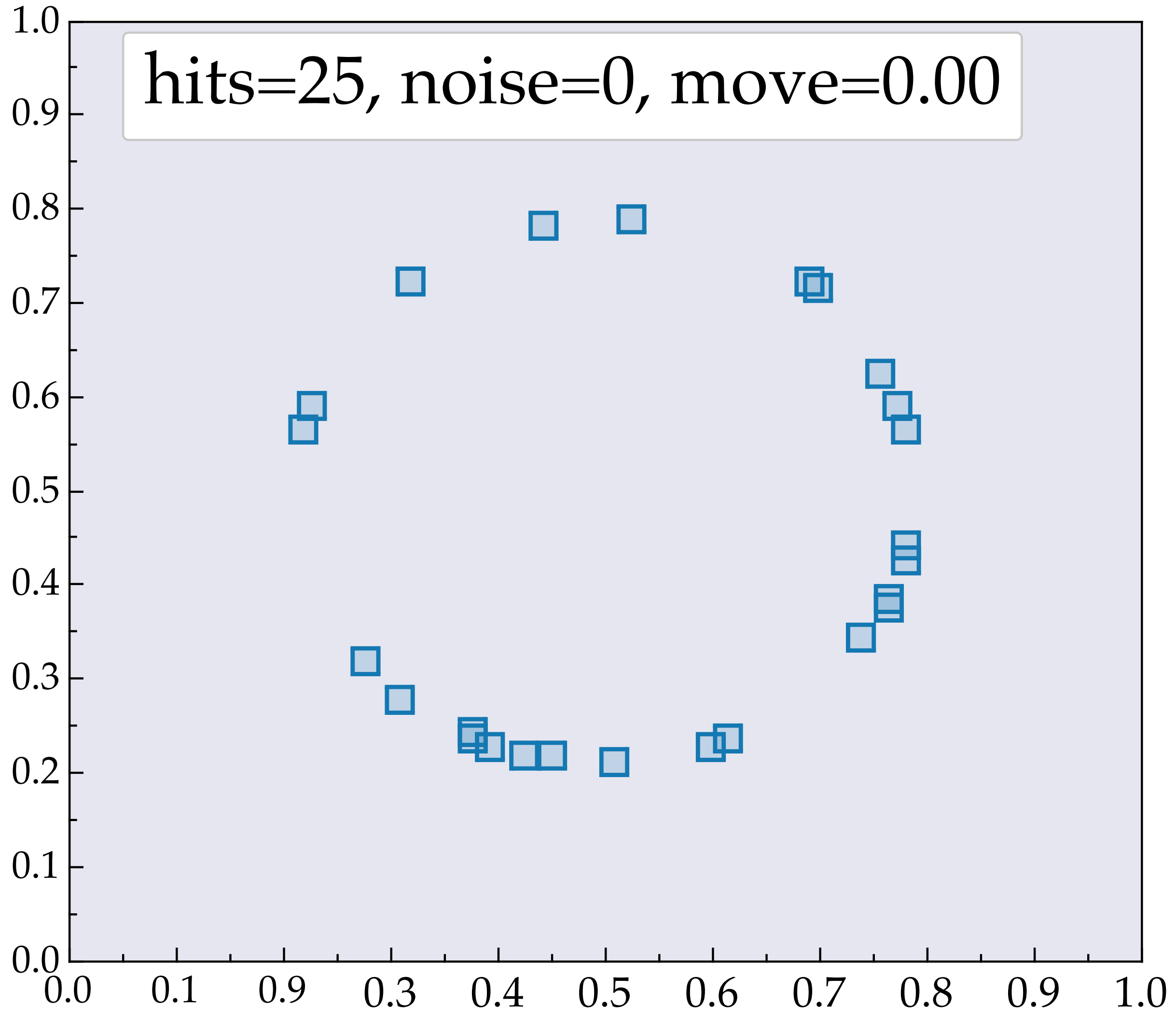
- **Simulation:** Simplified simulation with a square box with aerogel (refraction coefficient 1.05)
- **Photo-electrons**: Generating the Cherenkov ring with a various number of photo-electrons randomly distributed.
- **Noise**: Random levels of noise hits are added to the sample to evaluate network performance
- **Network Architecture:** Convolutional neural network trained on detector images to identify signal from kaon and pion.

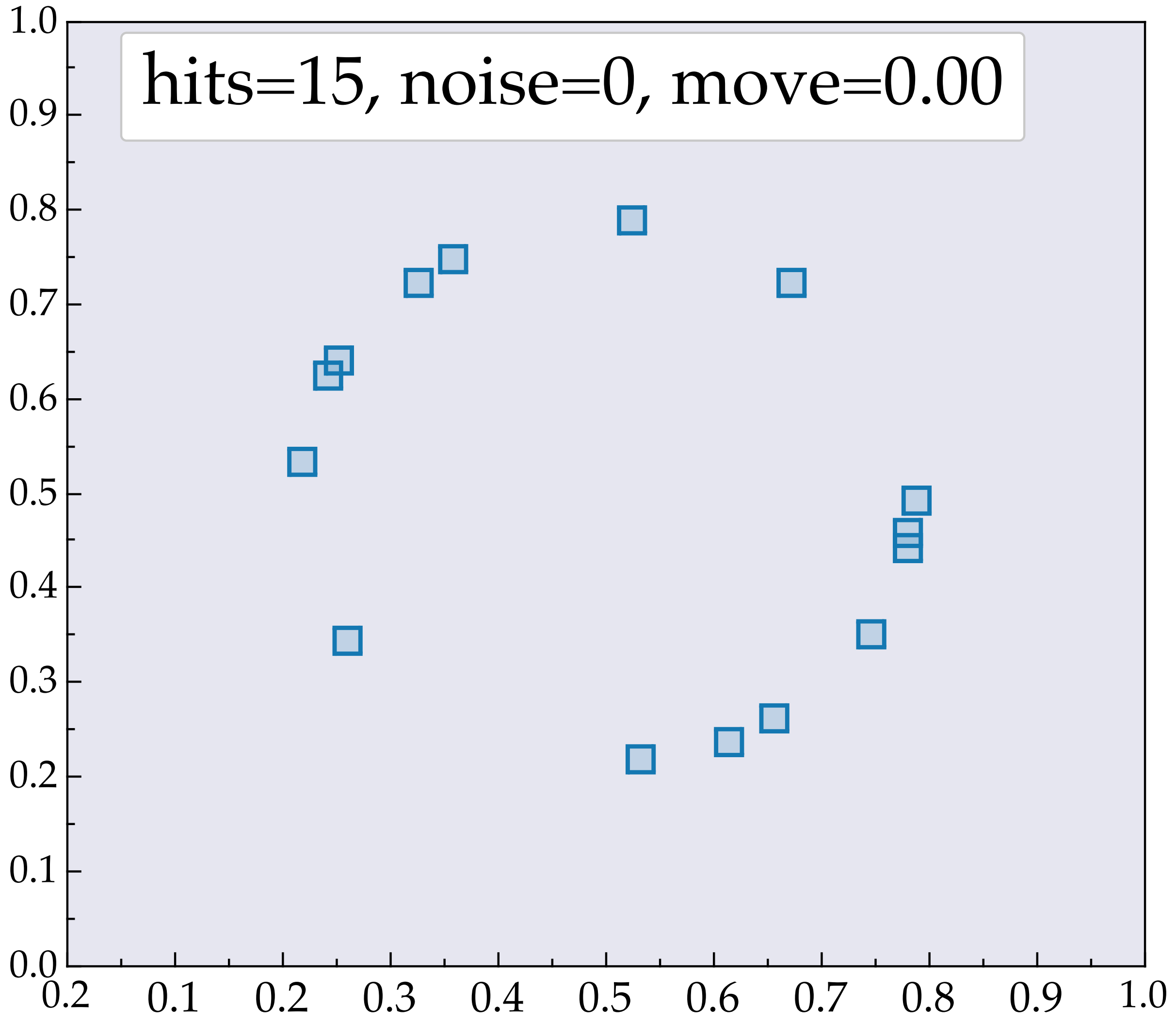


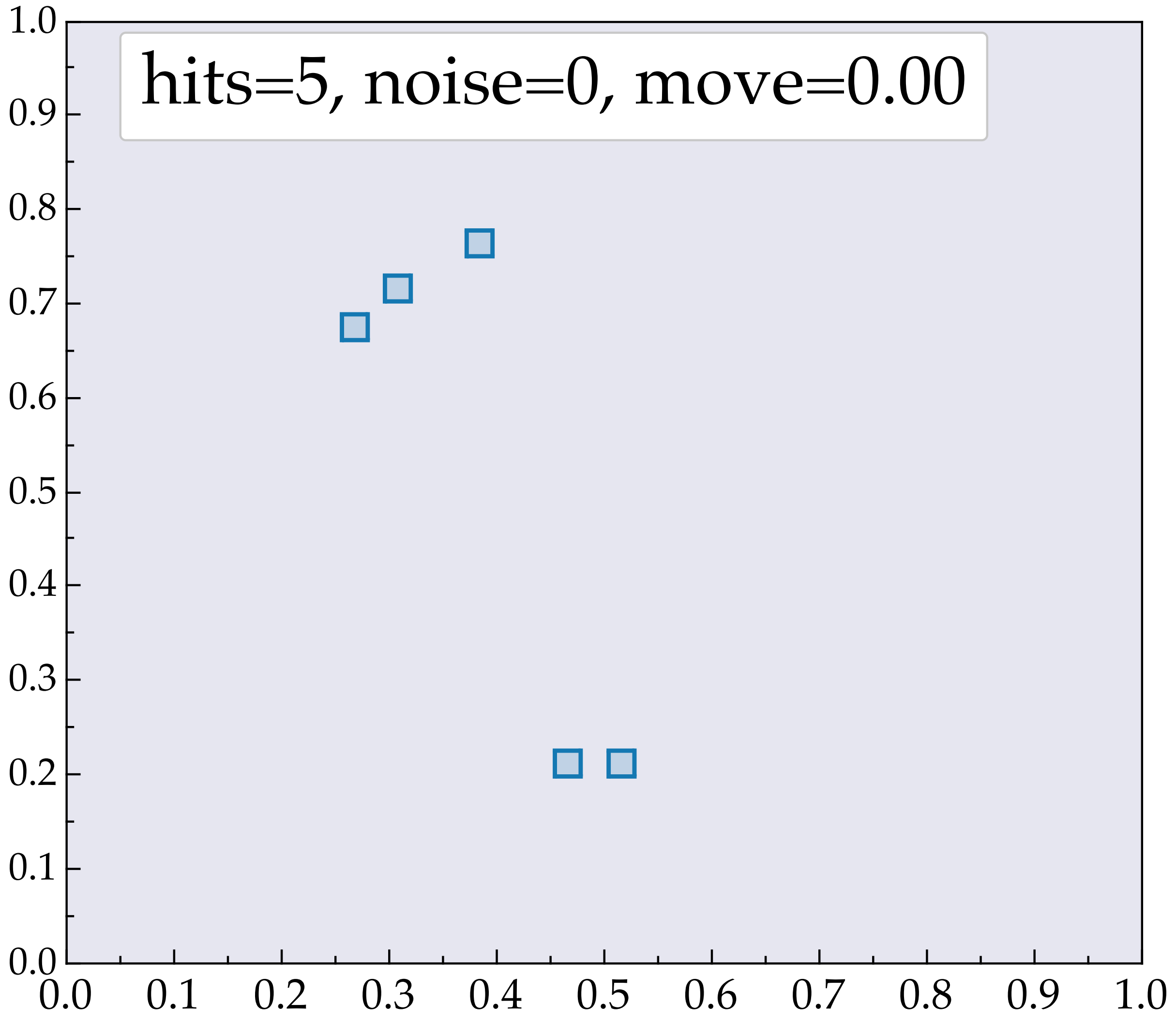


Computational Graph

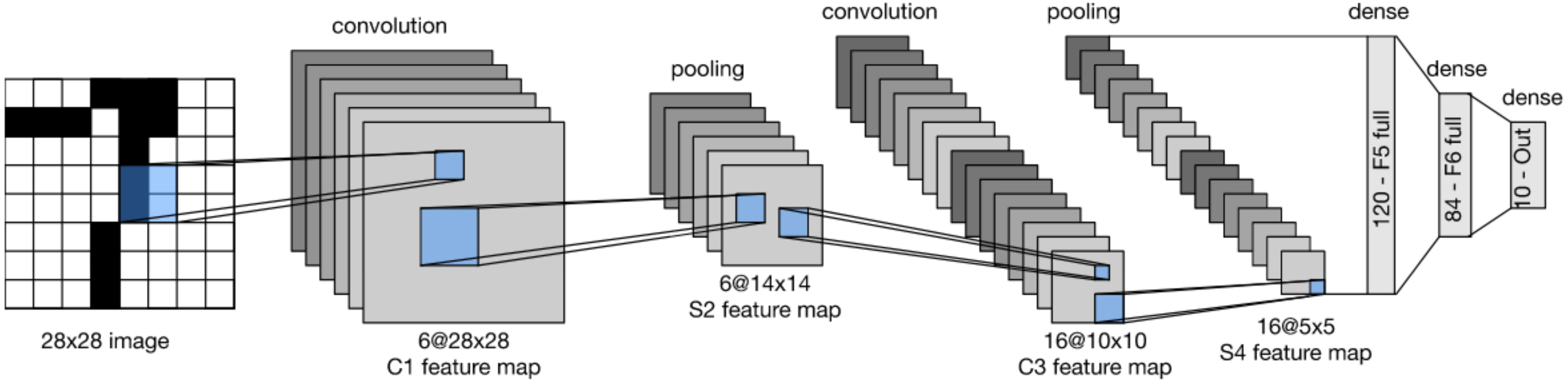


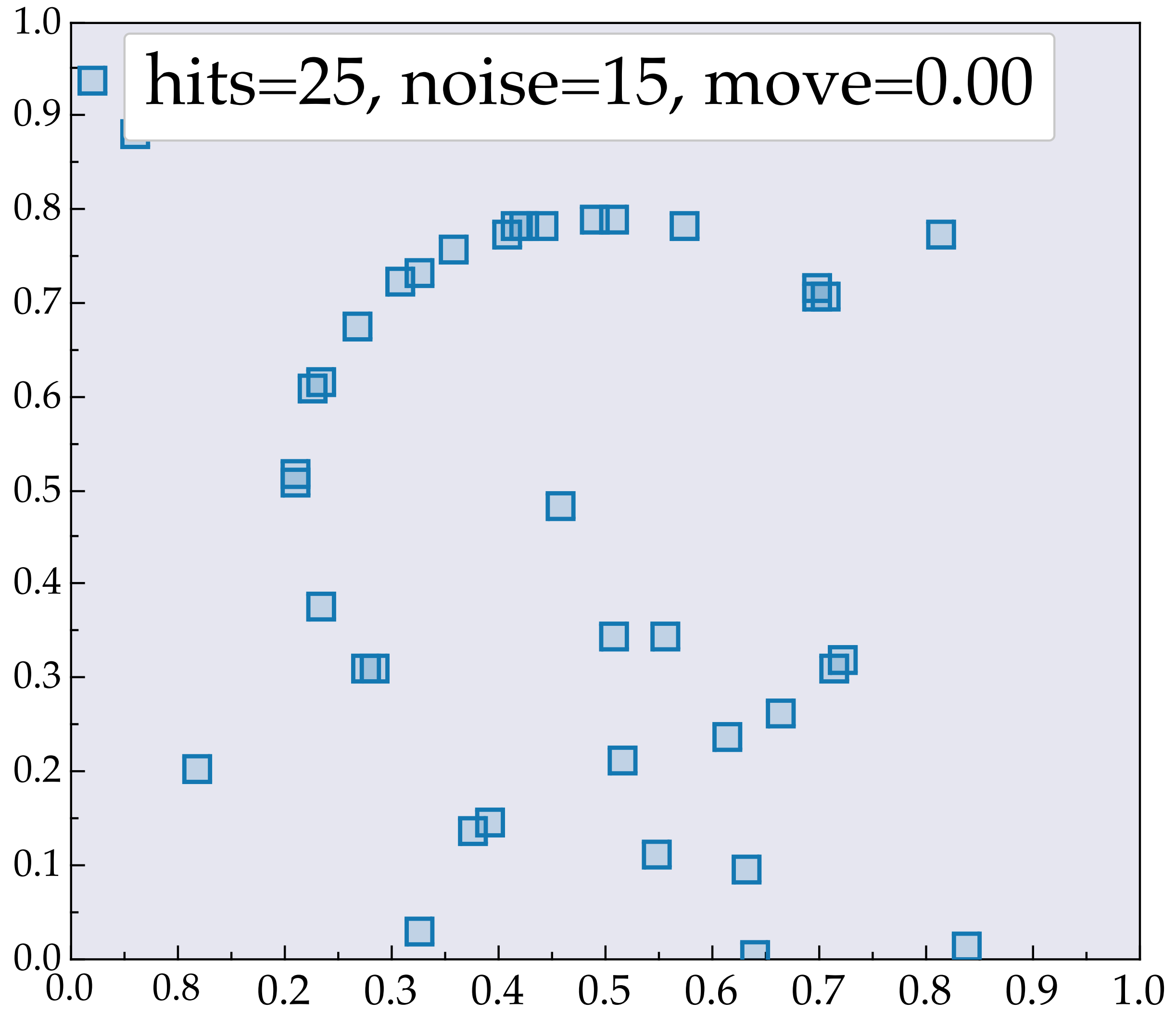


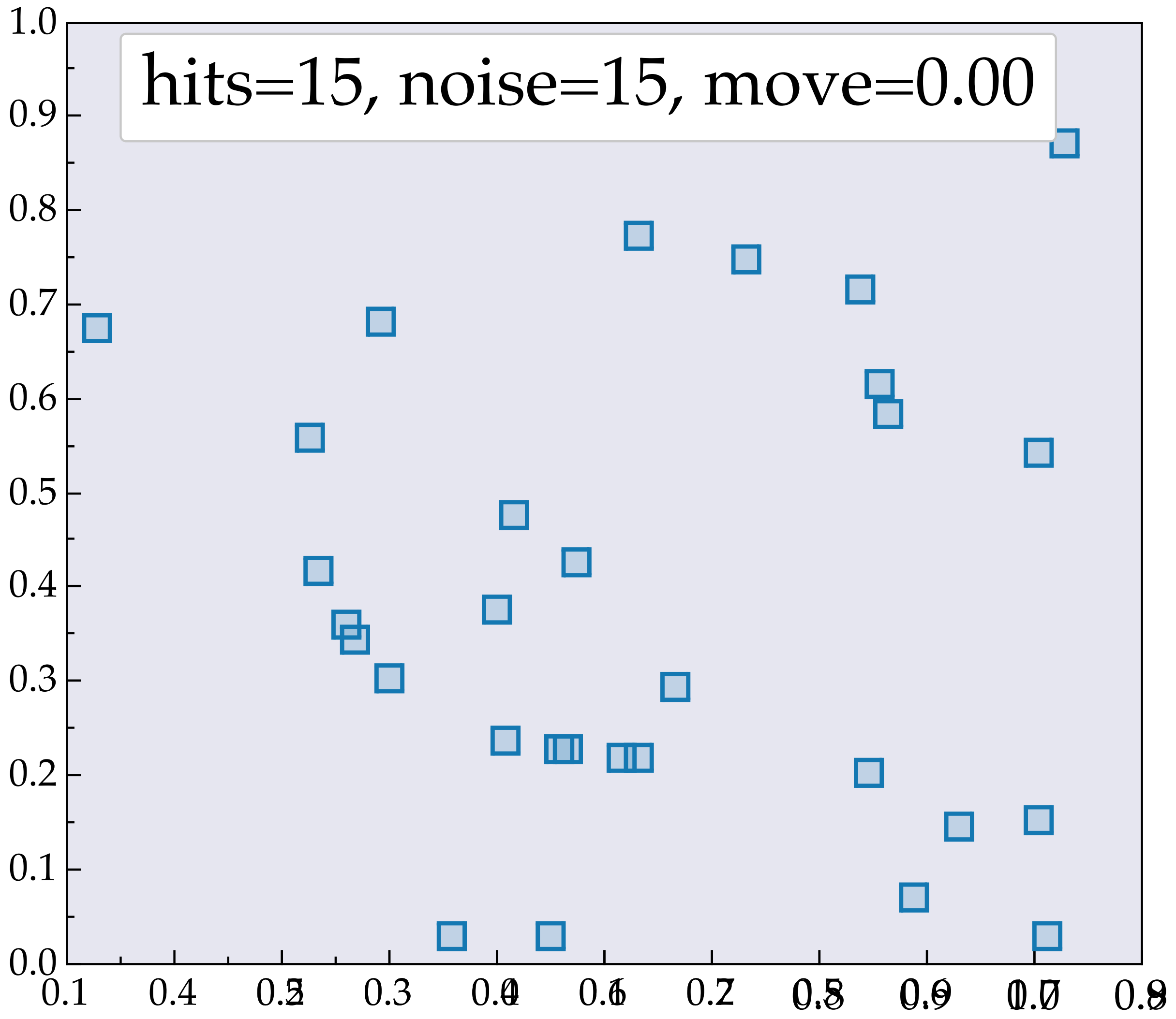


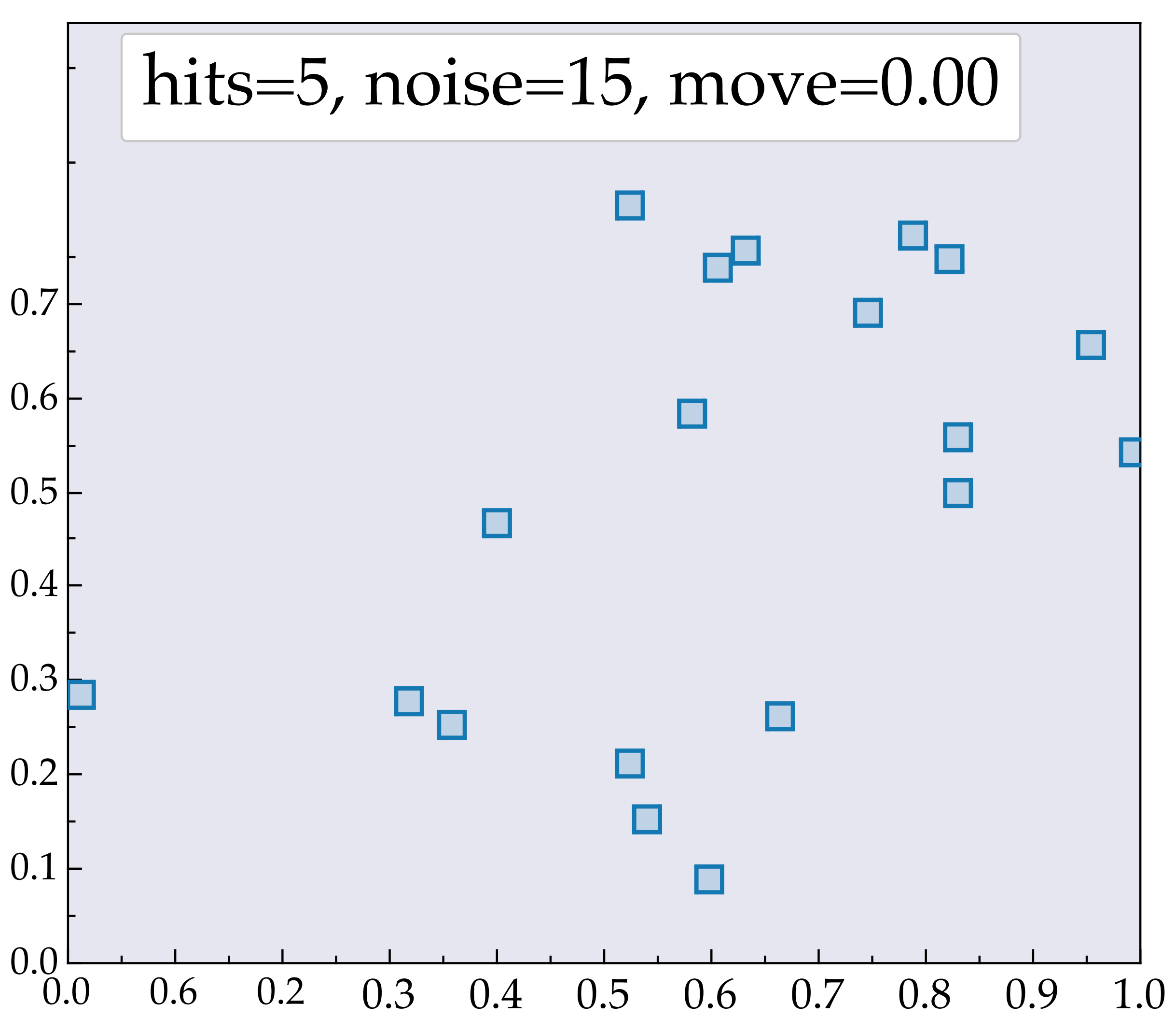


Computational Graph



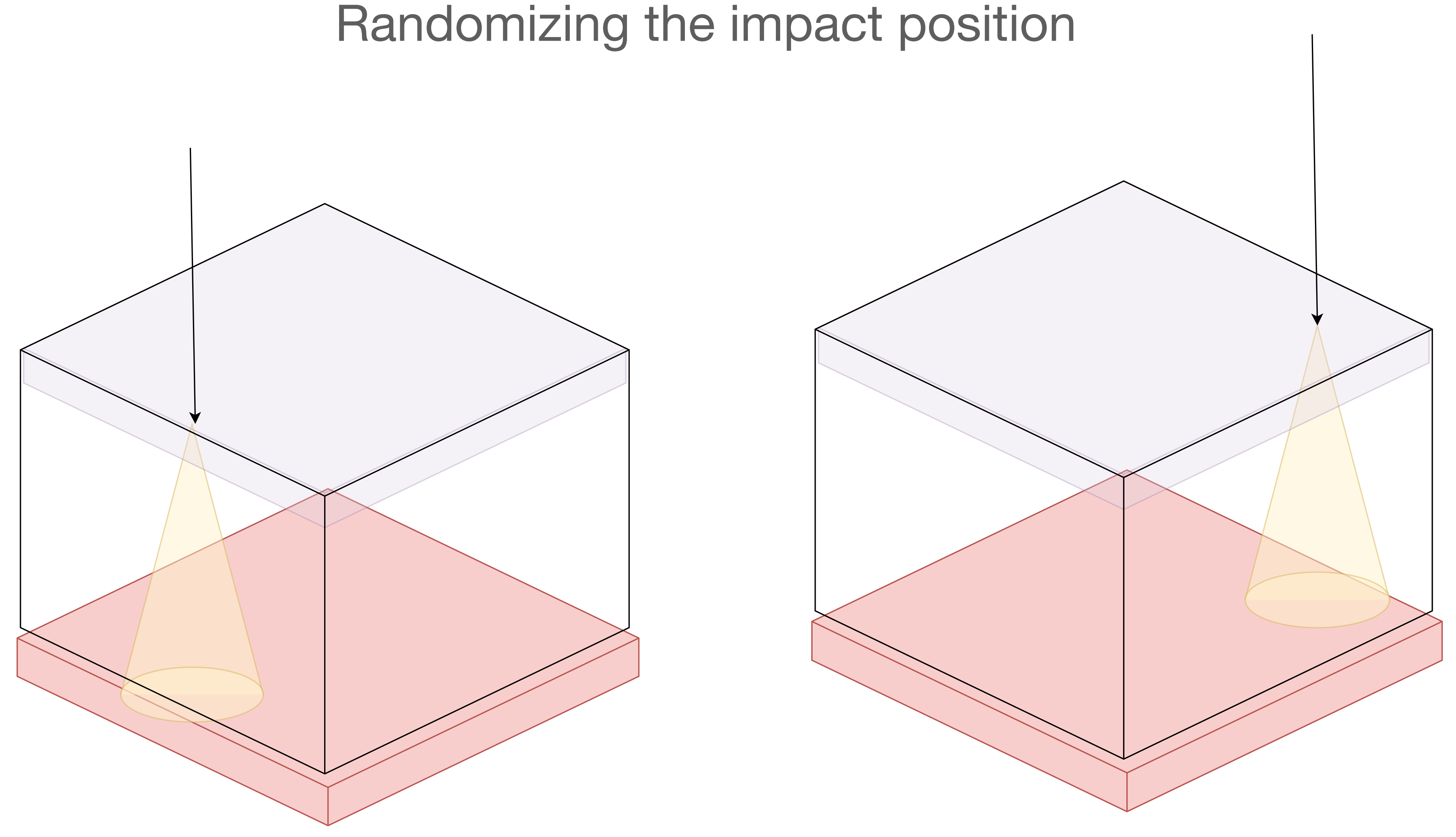




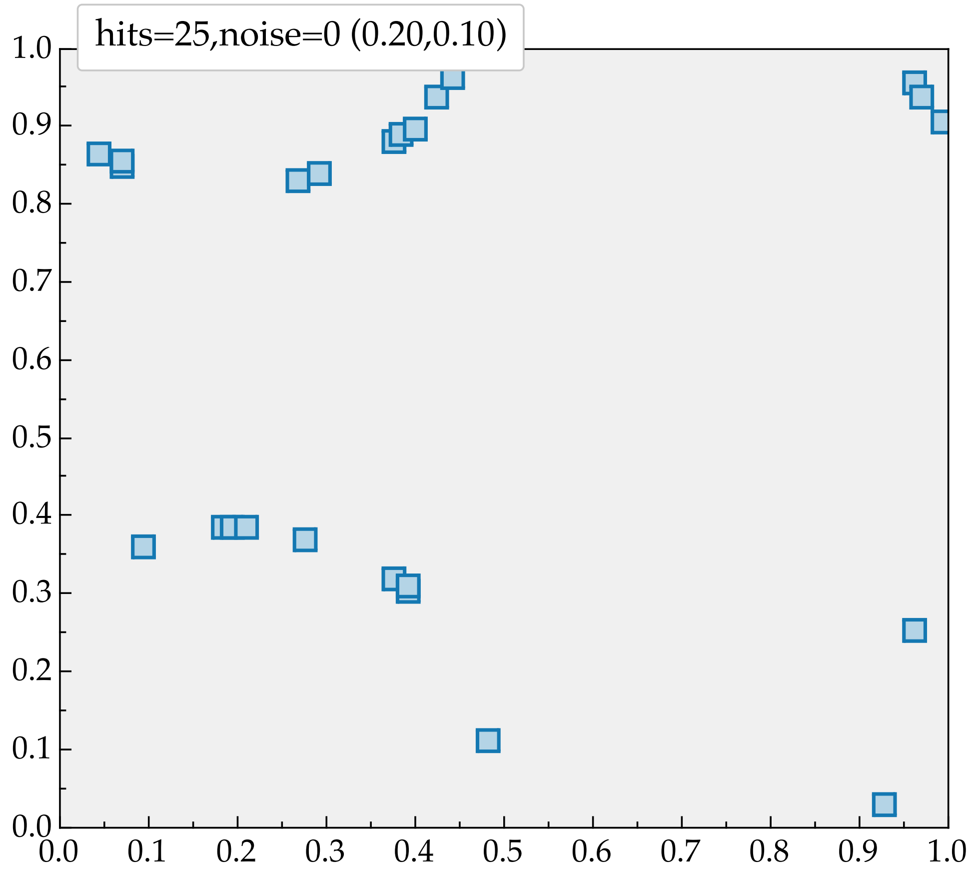


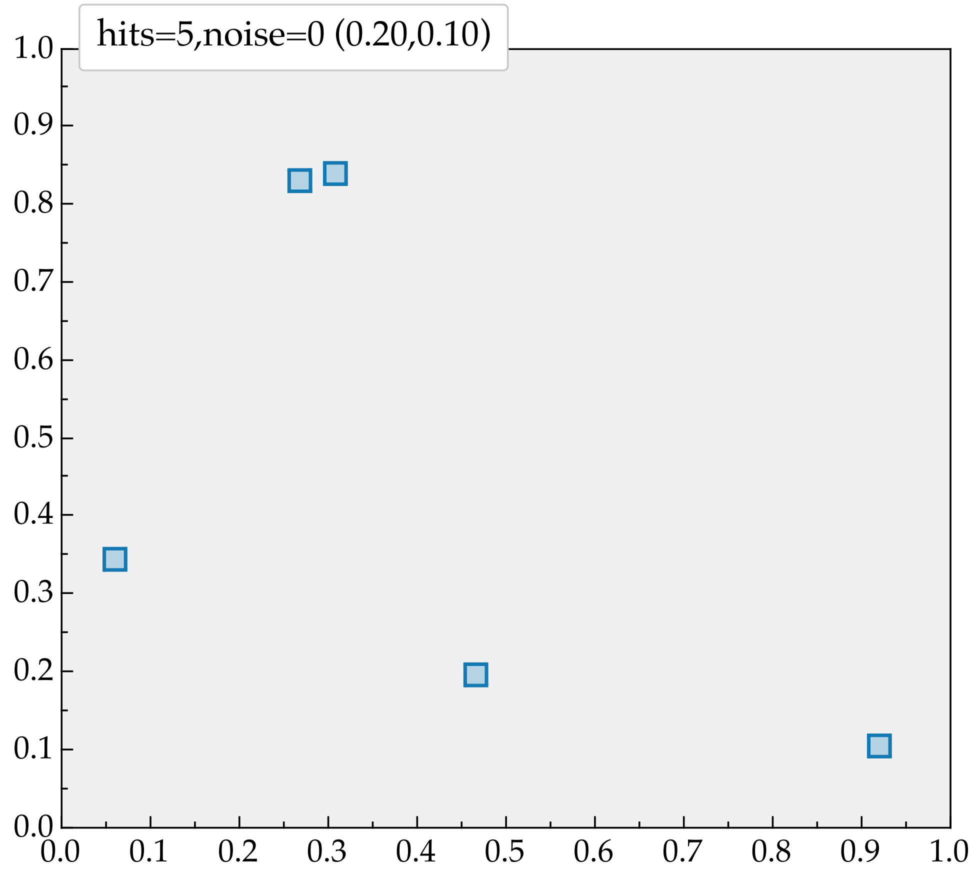
RICH Detector

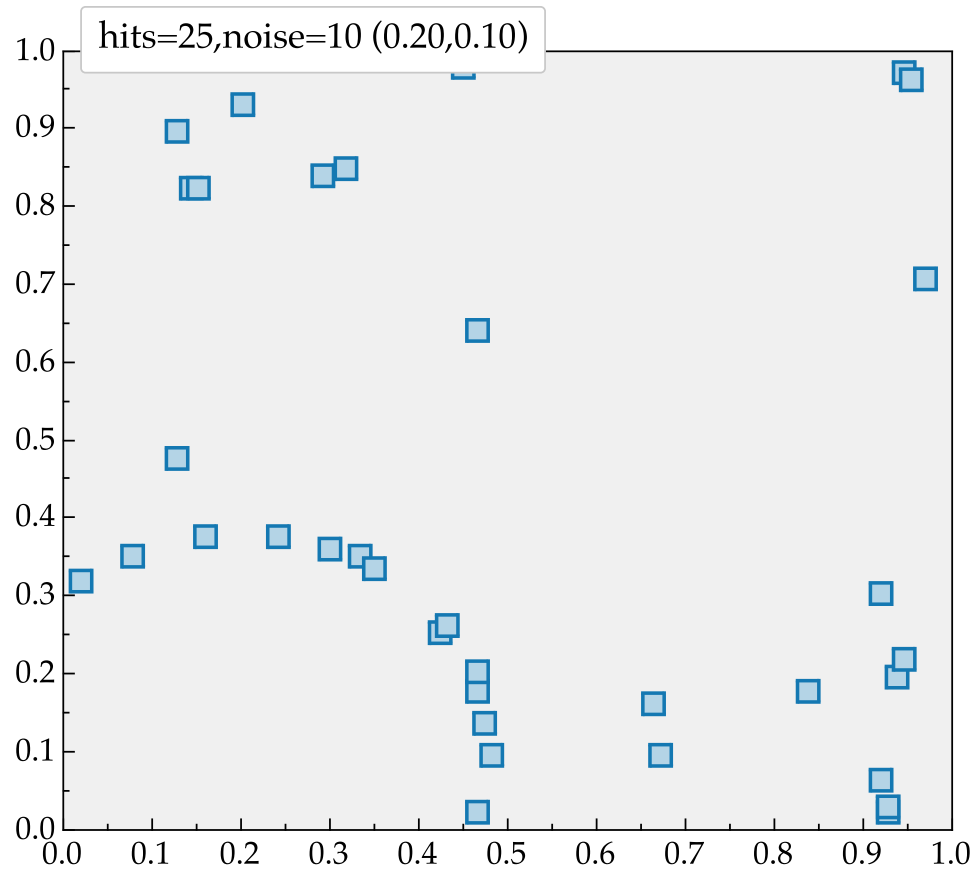
Randomizing the impact position

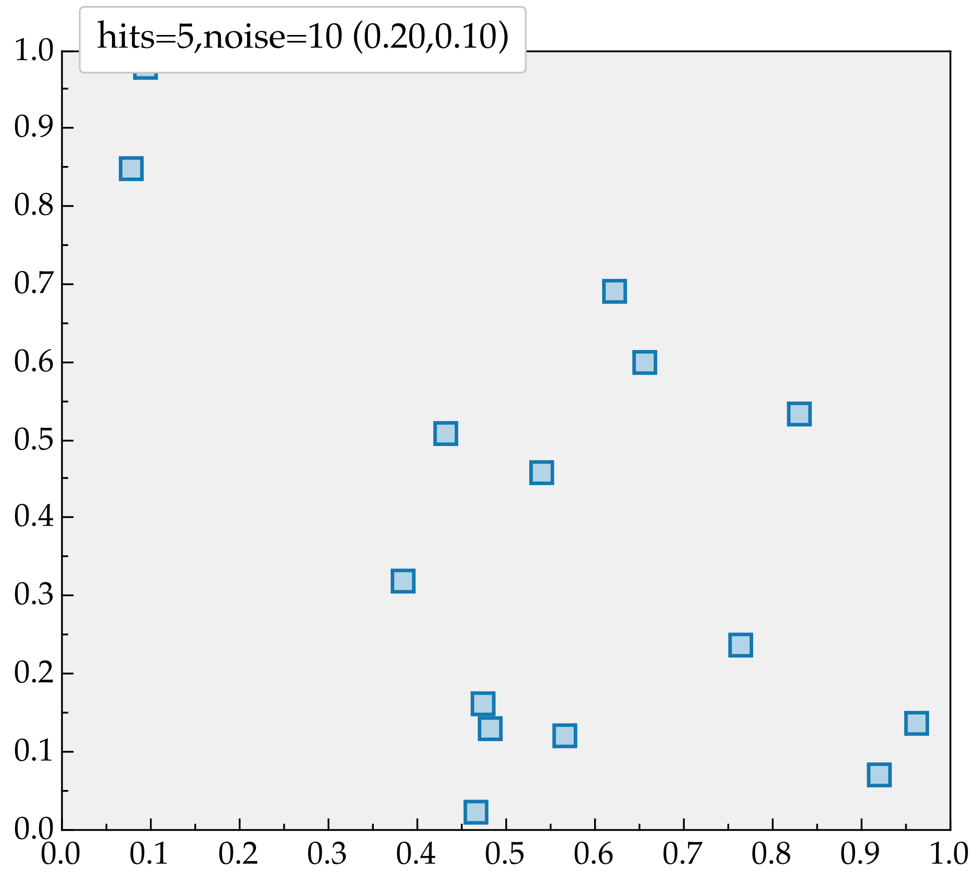


- **Simulation:** Simplified simulation with a square box with aerogel (refraction coefficient 1.05)
- **Photo-electrons**: Generating the Cherenkov ring with a various number of photo-electrons randomly distributed.
- **Noise**: Random levels of noise hits are added to the sample to evaluate network performance
- **Random impact position**: Randomizing the impact position of the particle on the box. Center = (0.5,0.5) + RNDM(-0.4:0.4,-0.4:0.4)
- **Network Architecture:** Convolutional neural network trained on detector images to identify signal from kaon and pion.



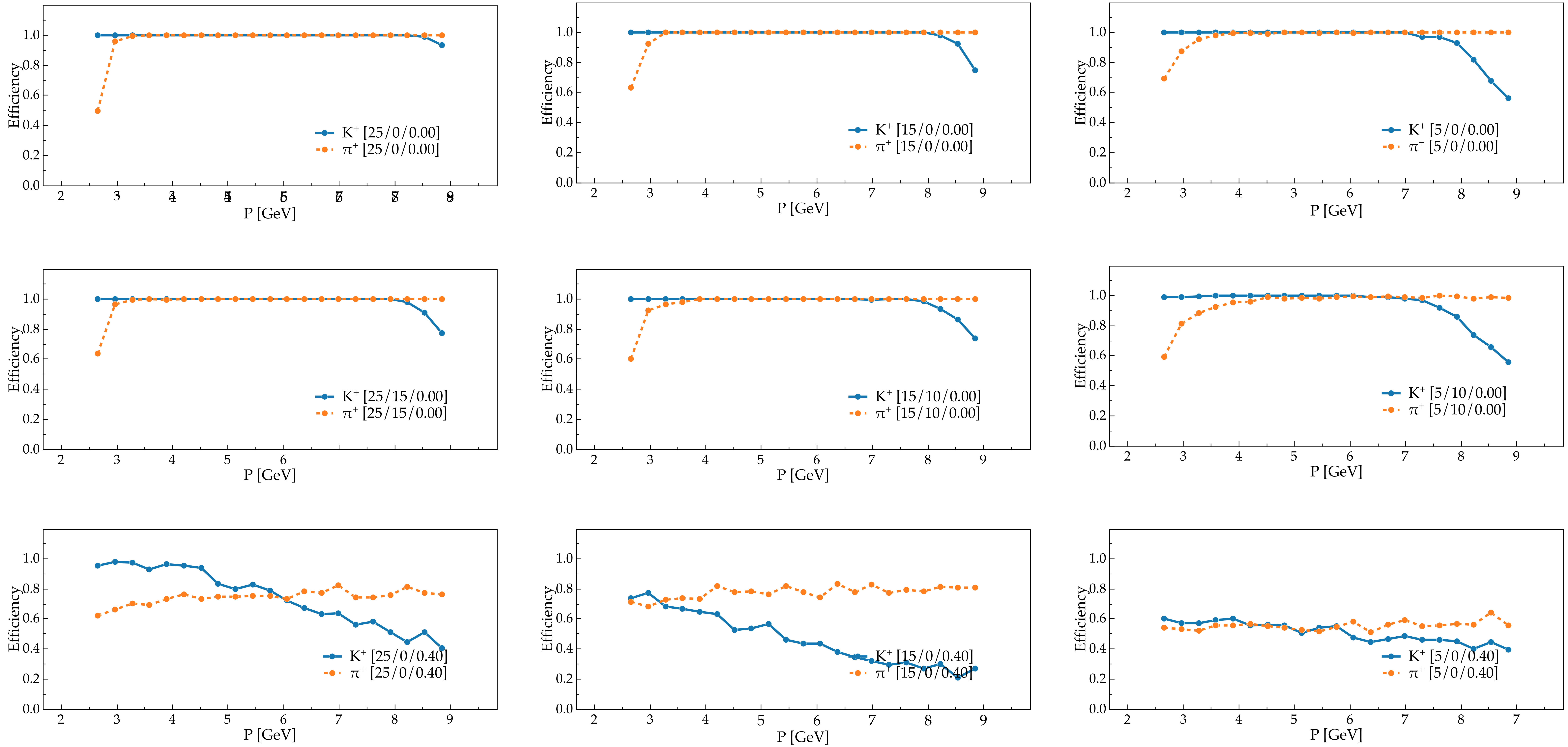


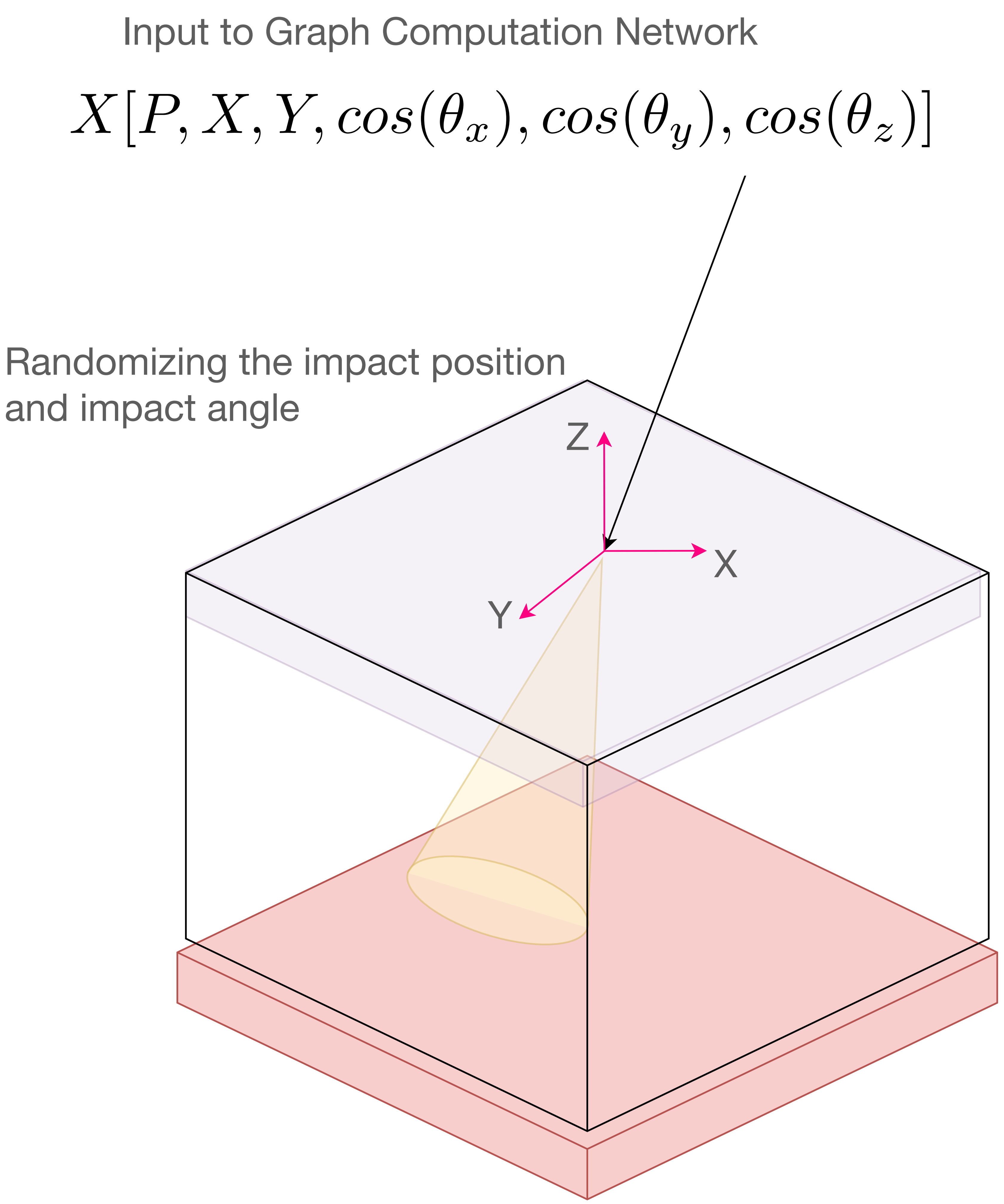


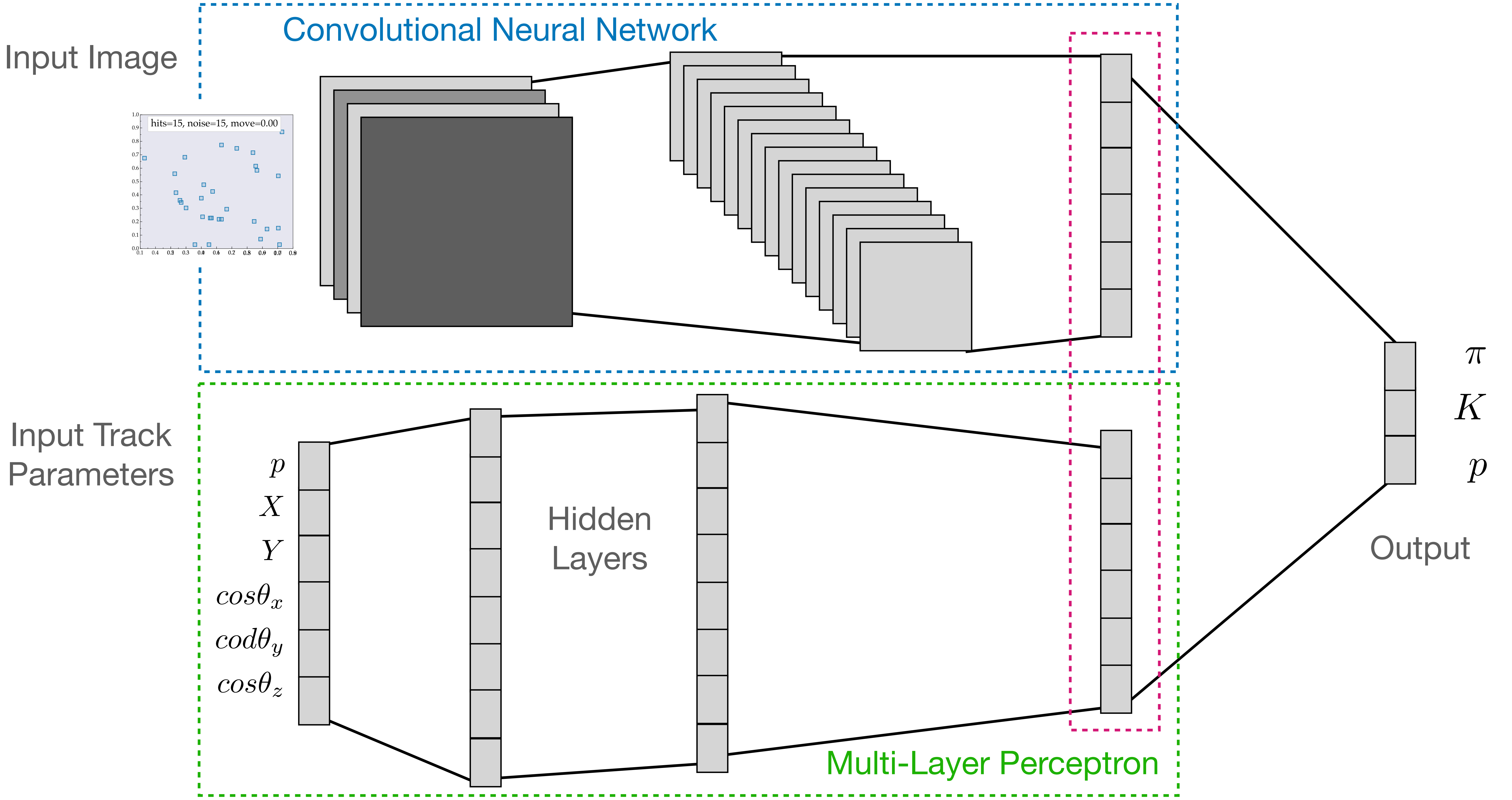


Results

	# photo-electrons 25	# photo-electrons 15	# photo-electrons 5
No Noise Center (0,0)			
Added Noise Center (0,0)			
No Noise Center Random (0,0) +/- 0.4			

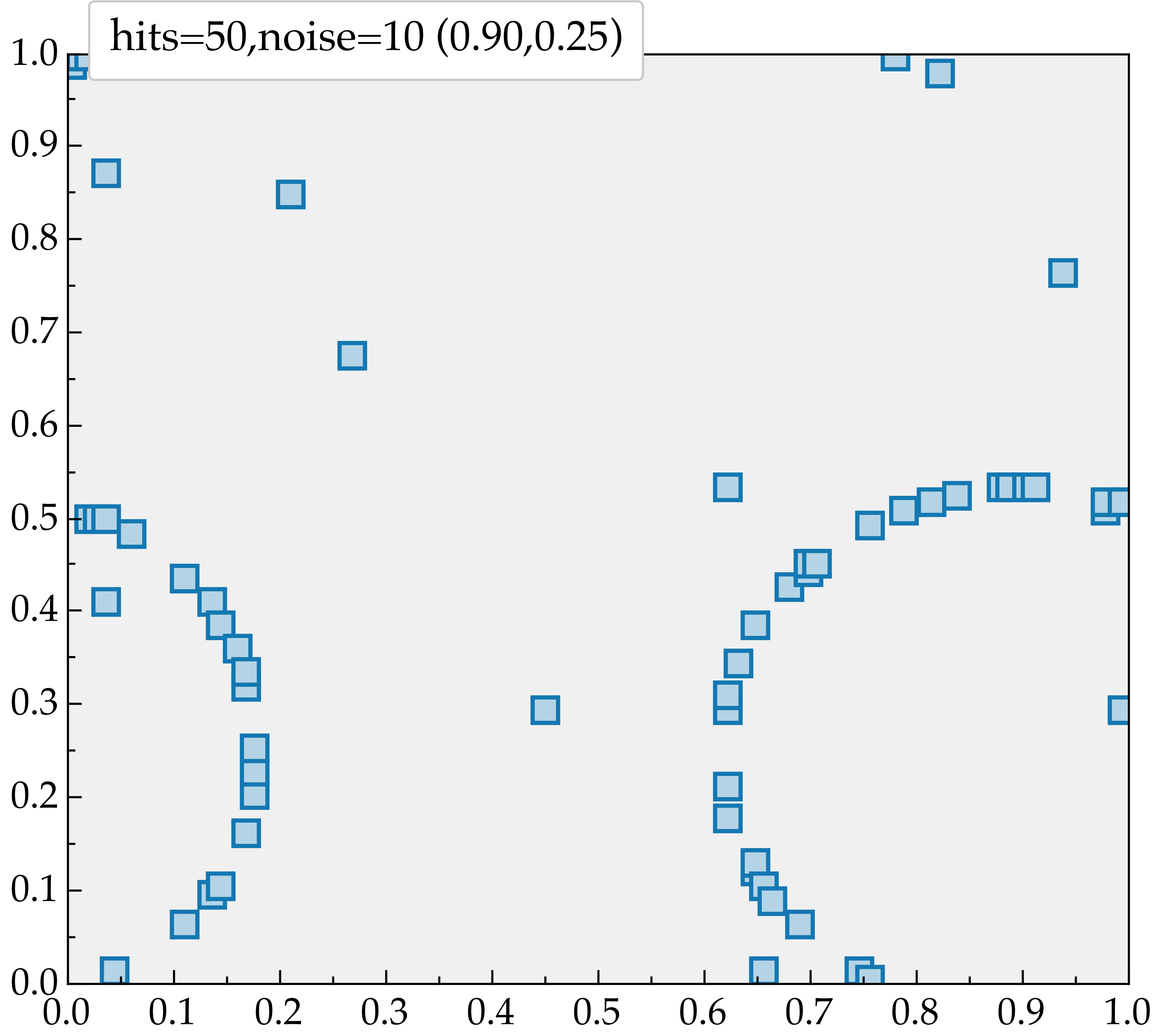






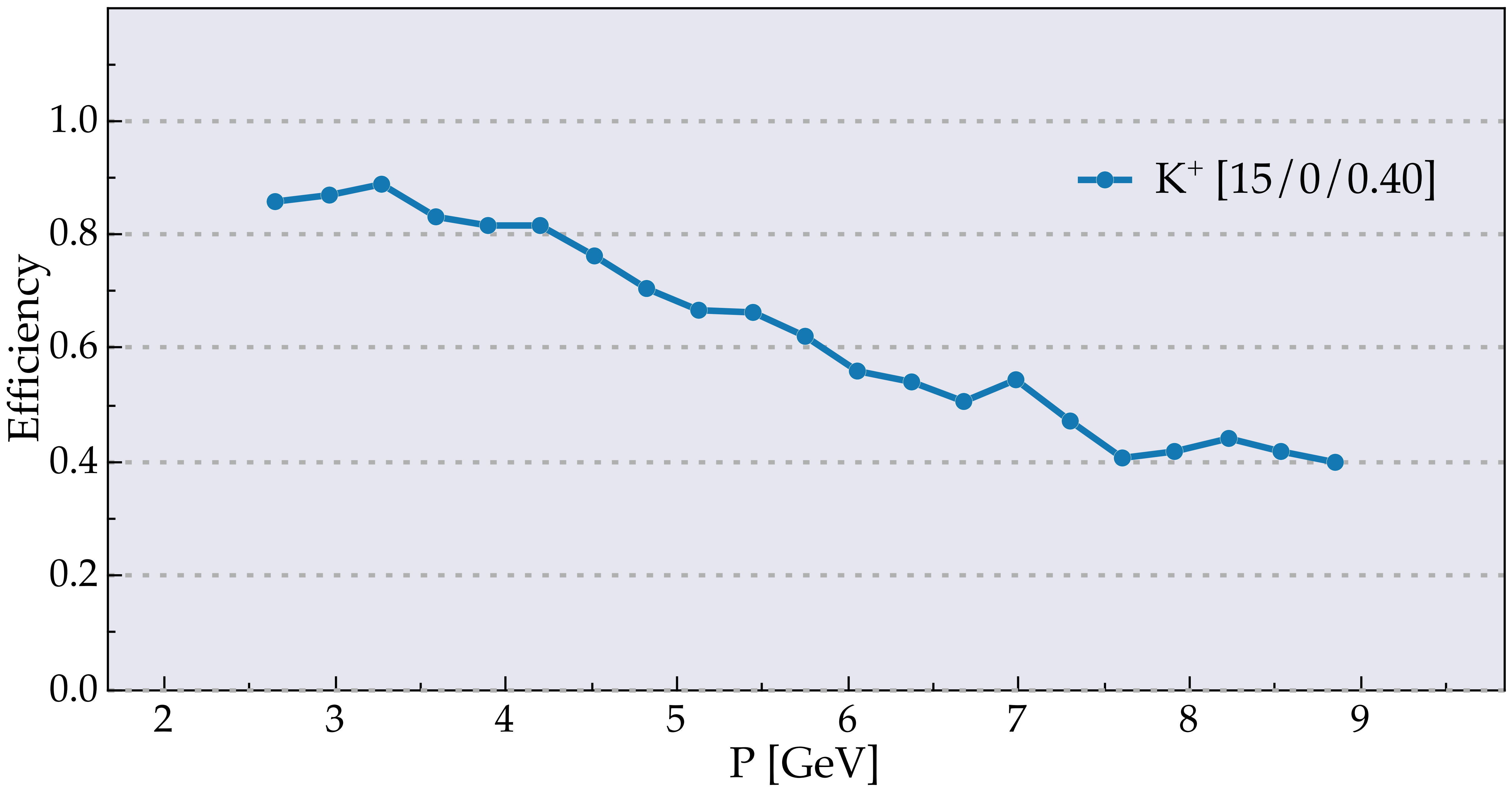
- **Computation Graph:** Consists of two different neural network architectures joined together to produce a classification output.
- **CNN**: Convolutional neural network analyzes the image of the ring from RICH detector
- **MLP**: Multilayer Perceptron provides input parameters for track producing the Cherenkov light.
- **Together** they provide classification for the particle.

- **Rings:** Image produced by particle hitting the box at position (0.90,0.25)
- **Photo-Electrons**: 50 photons are traced on the Cherenkov cone.
- **Noise**: 10 random noise hits are added to the data.



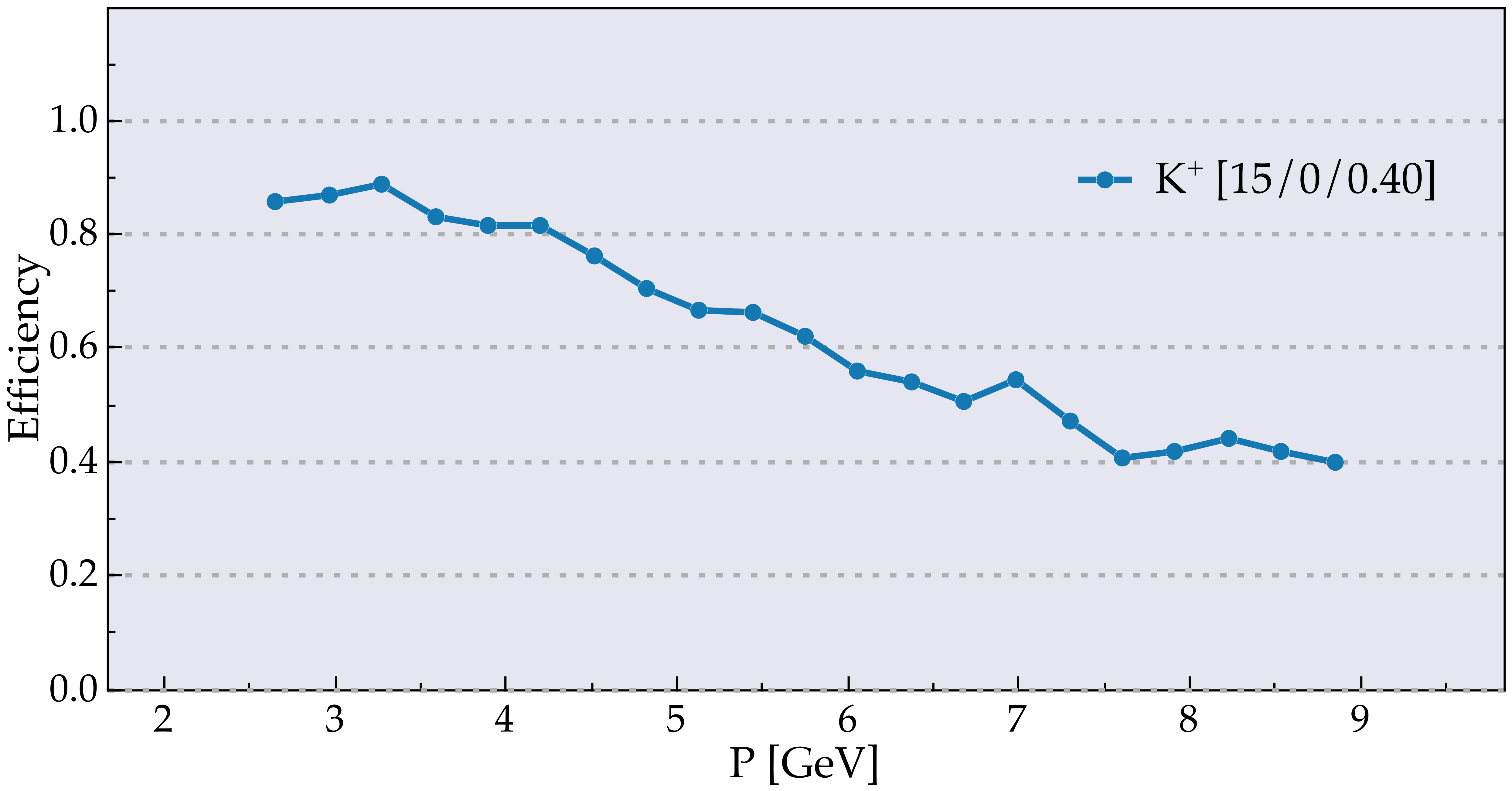
Results from Computational Graph

- **Sample:** Generated random incident position of the incoming particle on the Aerogel, (0.0-0.4) shifted center
- **Training Network**: Computational Graph Networks with Convolutional Network and Multi-Layer perceptron
- **Evaluation**: Kaon sample is evaluated by the network provides kaon probability.



Results from Computational Graph

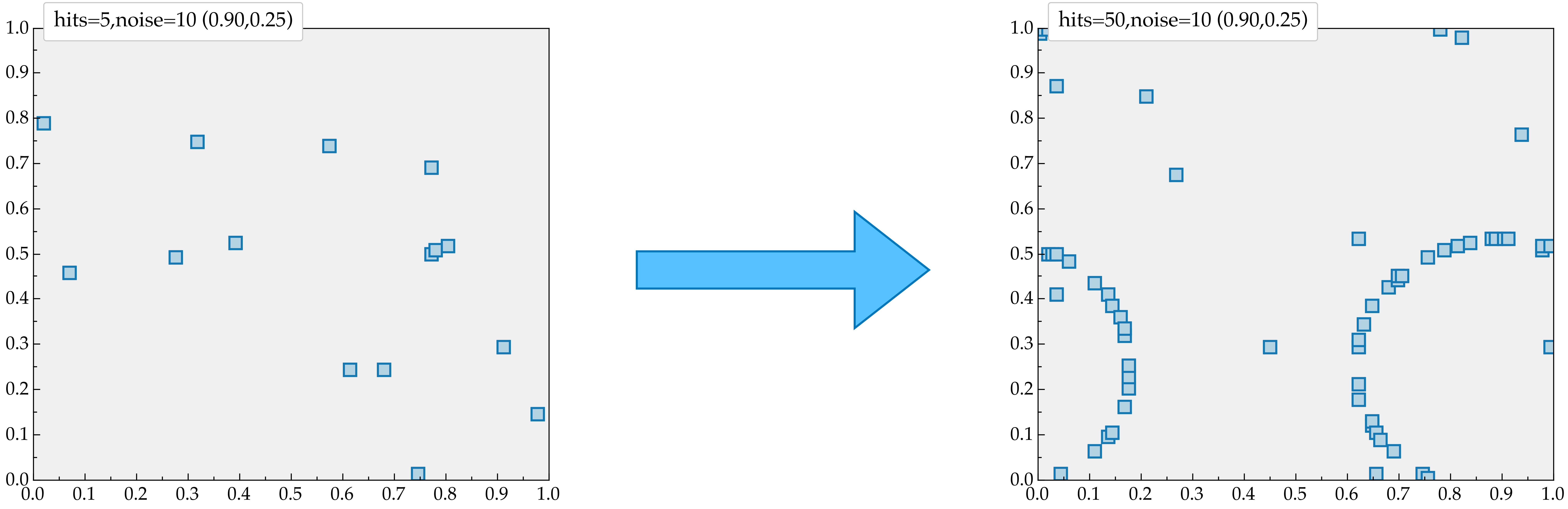
- **Sample:** Generated random incident position of the incoming particle on the Aerogel, (0.0-0.4) shifted center
- **Training Network**: Computational Graph Networks with Convolutional Network and Multi-Layer perceptron
- **Evaluation**: Kaon sample is evaluated by the network provides kaon probability.



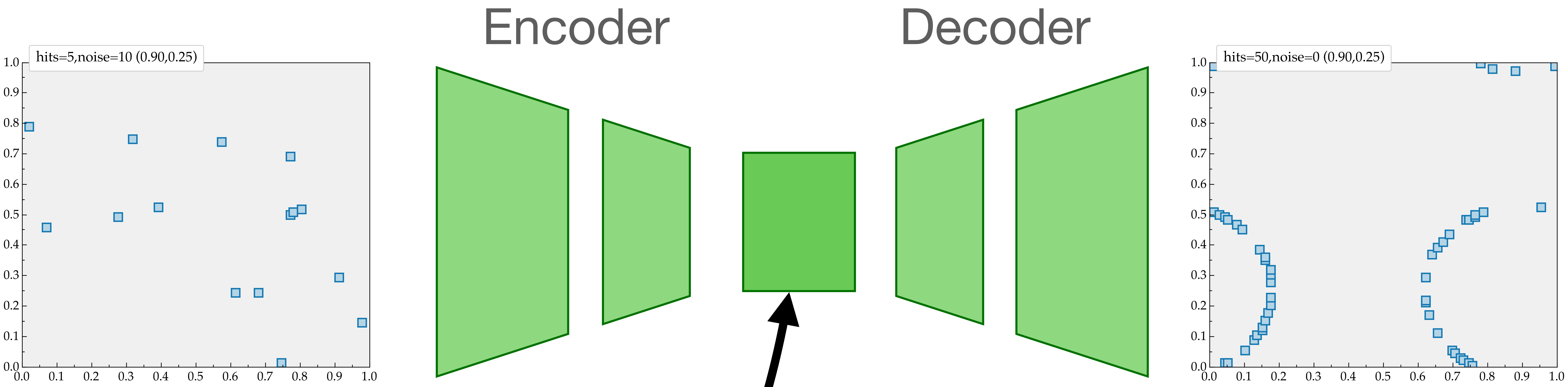
- **Problems:** For more sparse hit patterns (less photo-electrons) the efficiency is worse.
- **Noise**: Noise contributes to more reduction in Kaon identification reduction.
- **Solution**: Try to enhance the signal by removing noise and possibly generating pseudo-hits.

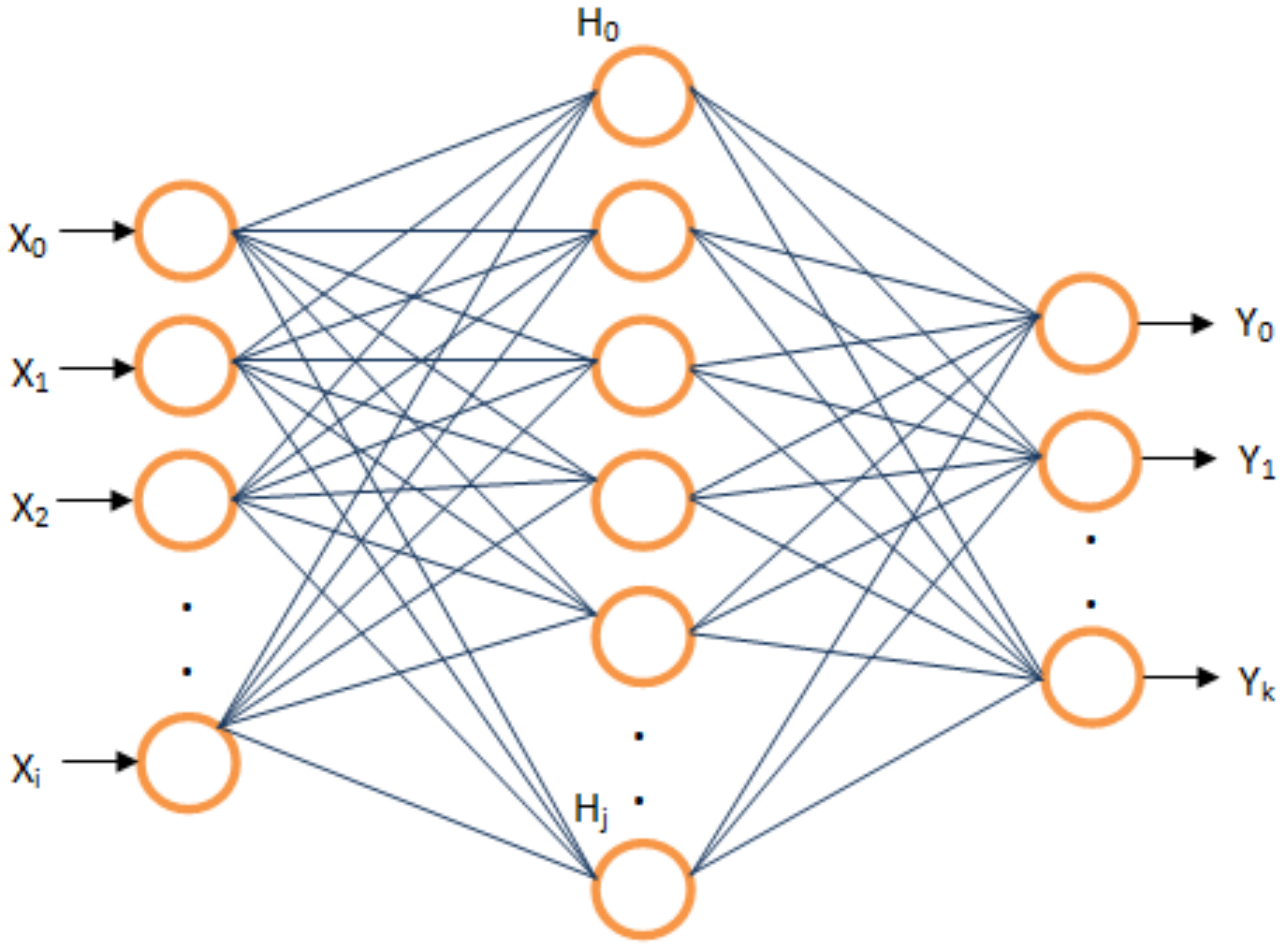
Possible Network Implementation

- **Procedure:** the sparse hit pattern has to be transferred to more populated image.
- **Particles**: For each particle, hypothesis have to be made by creating a new image based on the hit pattern and the input from particle interaction point.
- **Network**: A convolutional auto encoder computational graph will be appropriate, or a Generative Adversarial Network (GAN)



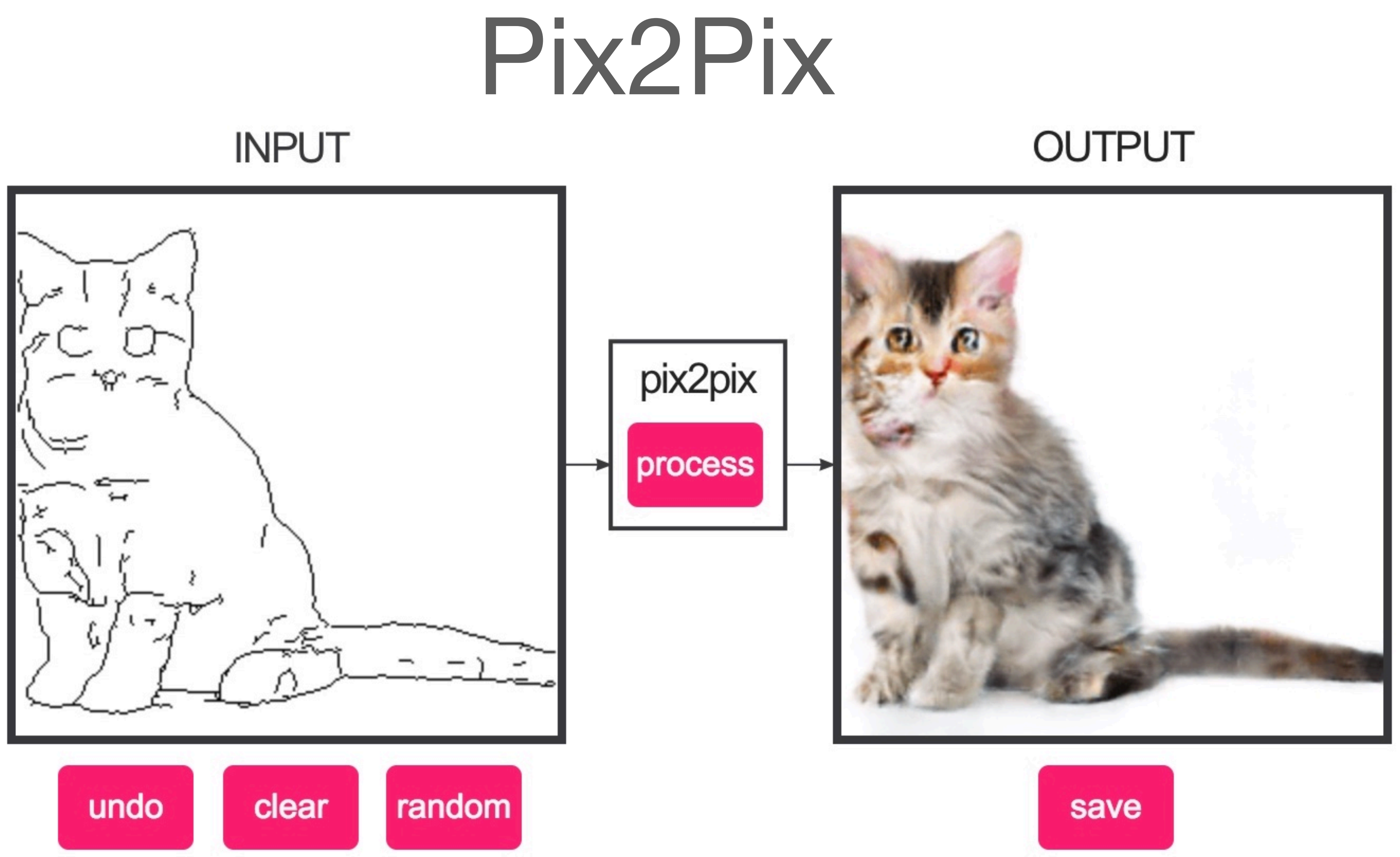
Results from Computational Graph

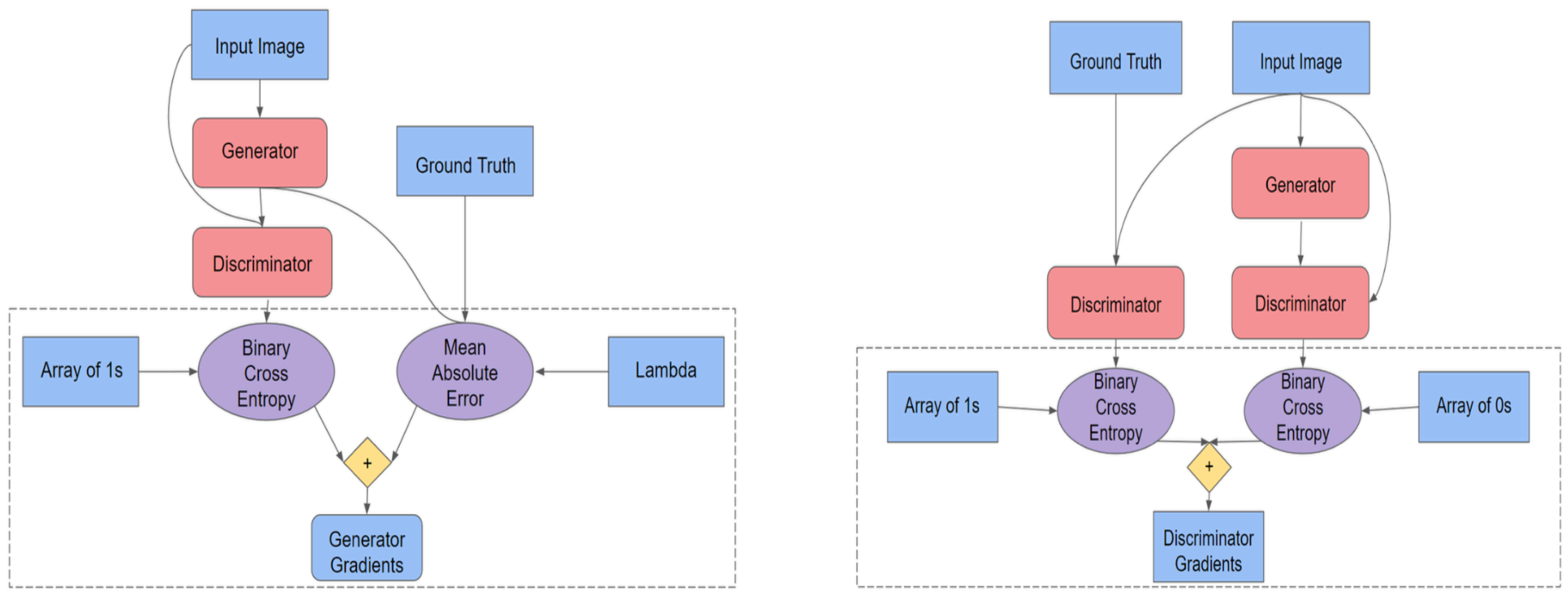




- **Network:** Use computational Graph with MLP and Convolutional Autoencoder.
- **Input to CNN**: The image of hits in RICH detector
- **Input to MLP**: Particle detector intersection parameters and particle momentum and direction.

- **Network:** Use computational Graph with MLP and Convolutional Autoencoder.
- **Input to CNN**: The image of hits in RICH detector
- **Input to MLP**: Particle detector intersection parameters and particle momentum and direction.
- **Output of CNN**: Reconstructed image in RICH of certain particle type
- **Combined Computational Graph:** Determine the particle type from





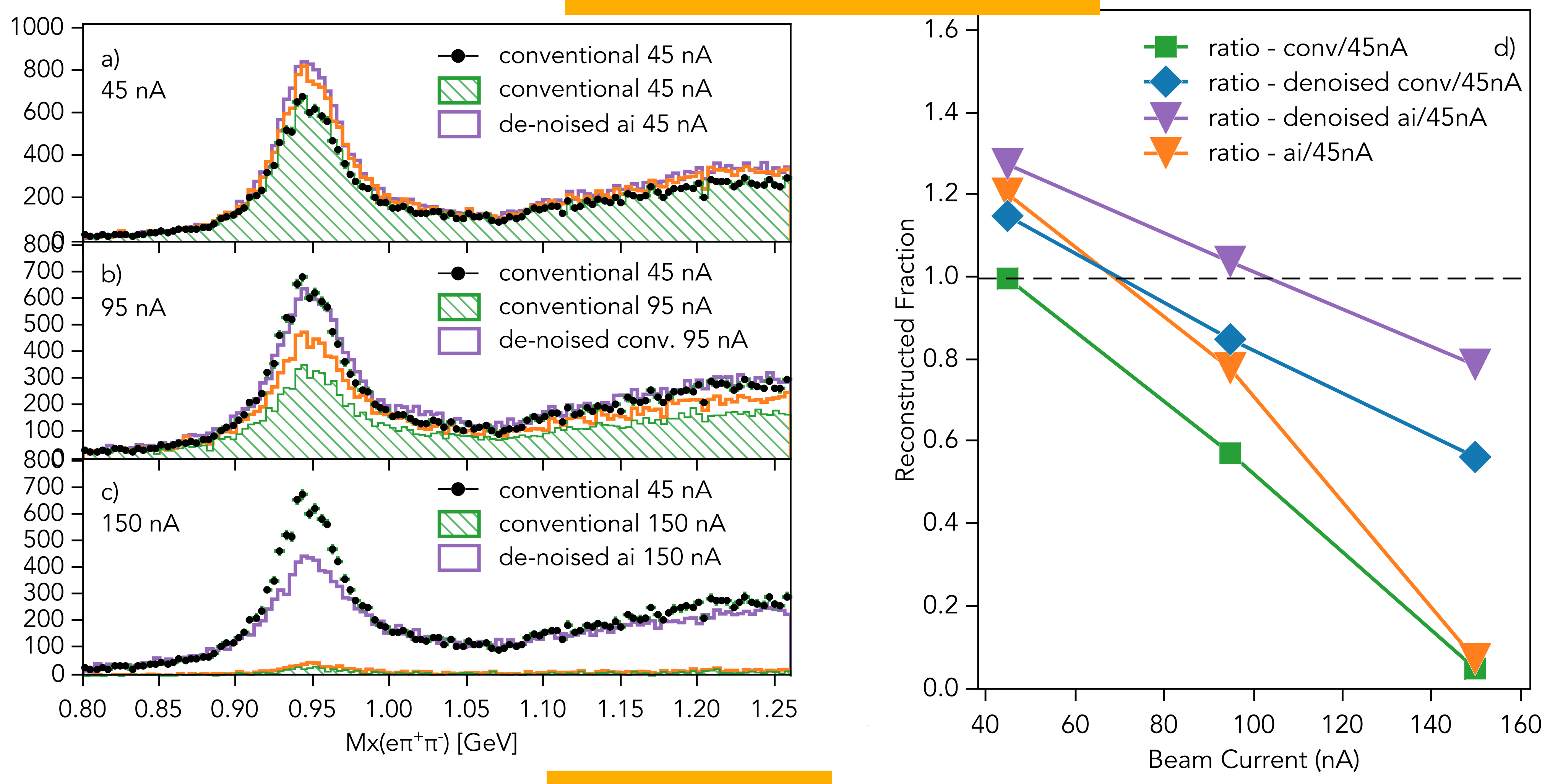
(a) Generator Loop (b) Discriminator Loop

A **generative adversarial network** (**GAN**) is a class of machine learning frameworks

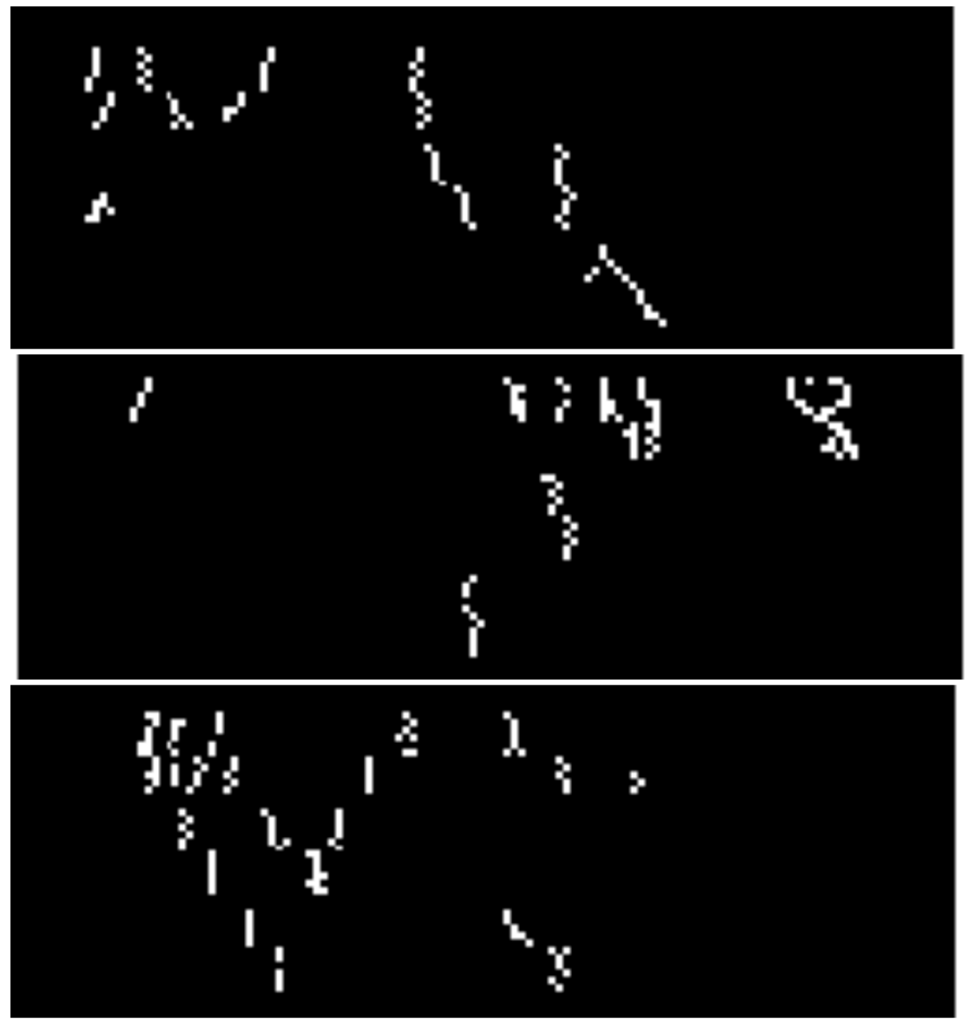
Two neural networks contest with each other in the form of a zero-sum game, where one agent's gain is another agent's loss.

- **Usage:** Generate hits belonging to a track in CLAS12 drift Chambers given all the hits. (Removes noisy hits)

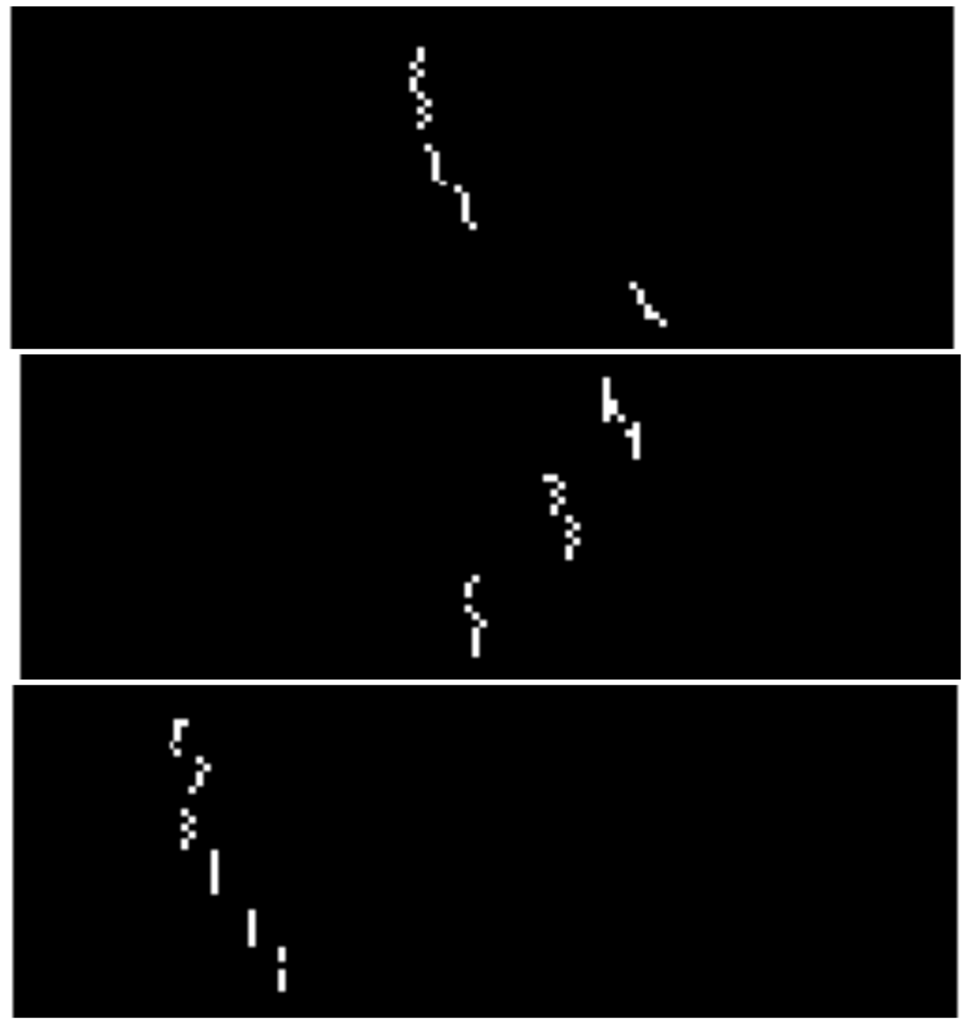
Simulation



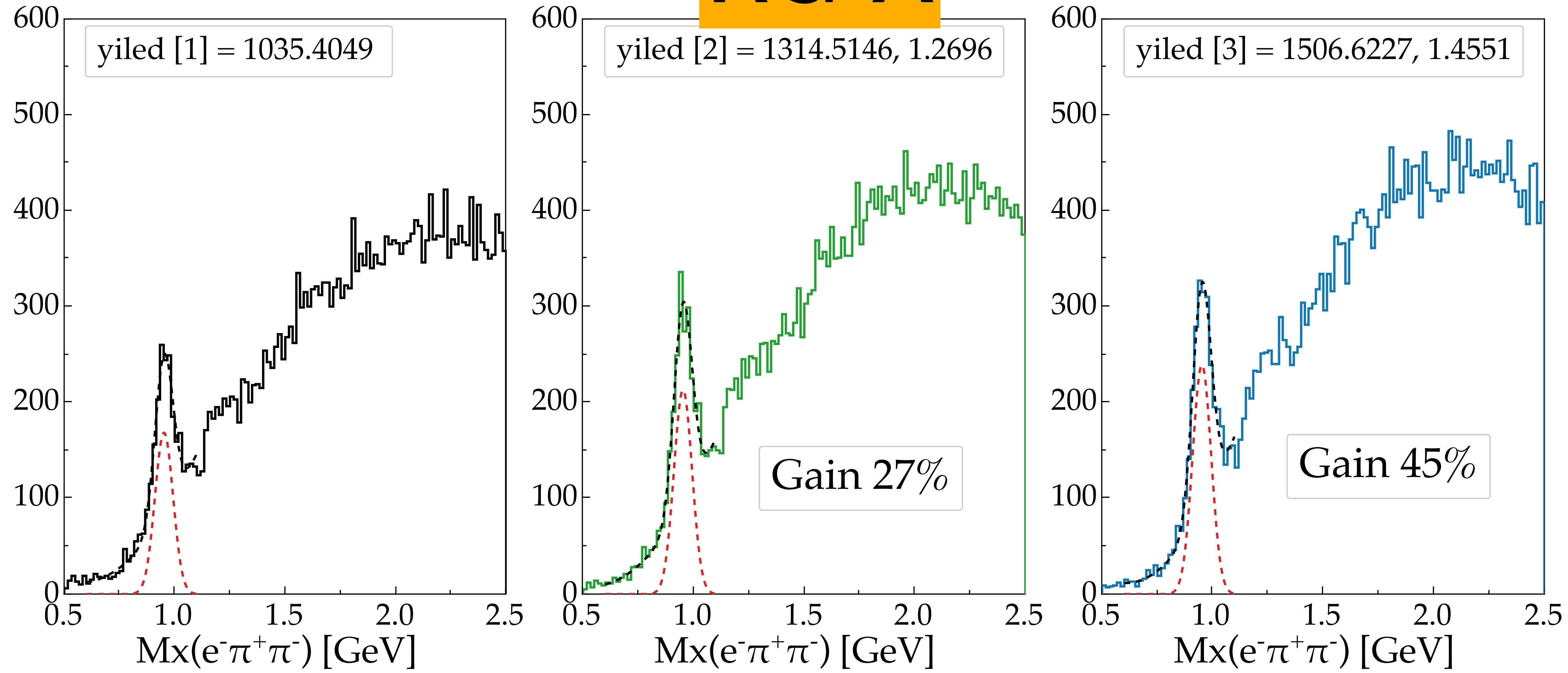
RAW



Pix2Pix



RG-A



Conclusion

- **AI approach:**
- The AI approach is preferred due to simpler implementation and less code base.
  - Need good simulation to produce training sample
- When trained on experimental data AI may reduce the need for alignment.
- Neural Networks trained with Python (Keras/TenserFlow) can be included in the workflow using C++/Java interfaces.

Backup