Machine Learning for Signal Processing in the NEWS-G Experiment

Annabelle Makowski, Supervised by Ryan Martin and Guillaume Giroux

May 27th, 2024

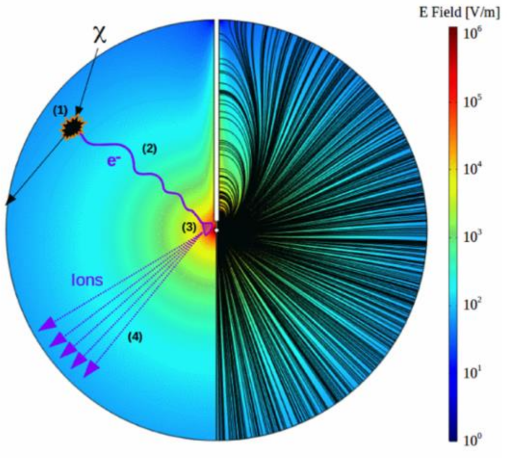
CAP Conference, London, ON

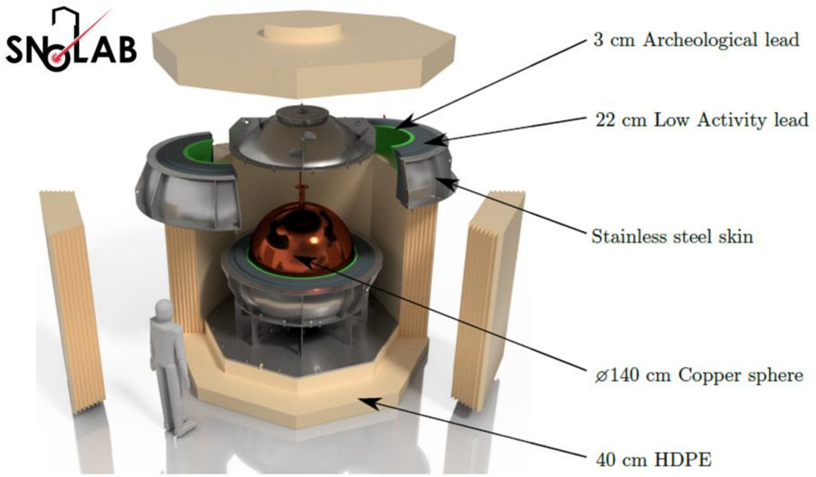
About the NEWS-G Experiment

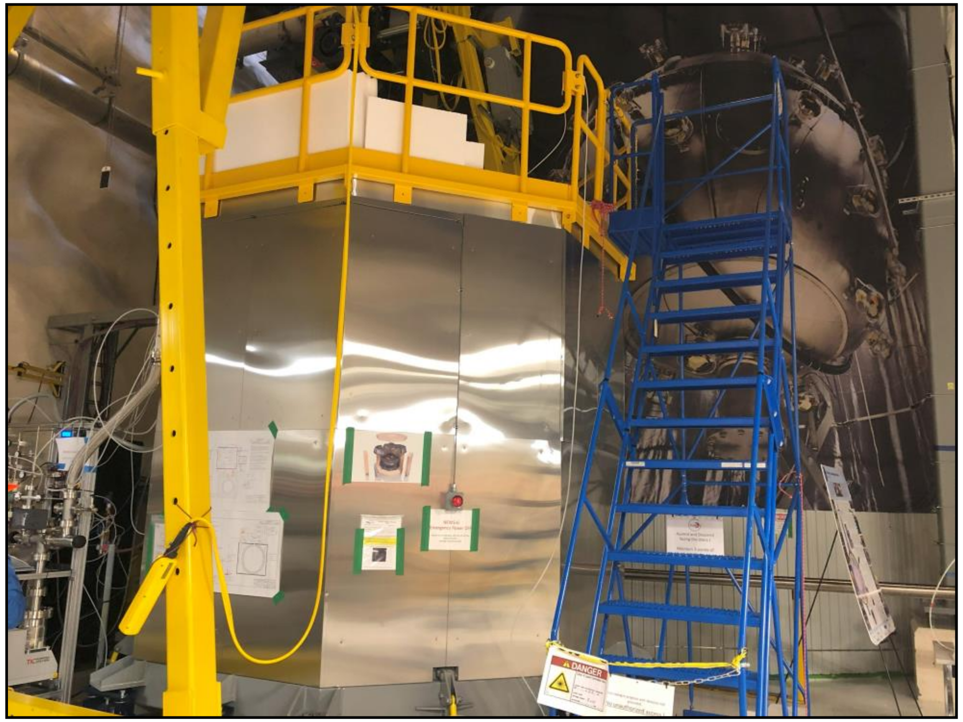
- The NEWS-G experiment uses SPCs (spherical proportional counters) for low-mass dark matter searches.

- Most recent detector S140 (SNOGLOBE) is a 140 cm diameter sphere, located at SNOLAB 2 km underground and is currently being used to take data.

- The sphere is filled with a light gas mixture. Incoming particles cause the gas molecules to ionize, and secondary ions induce a current on the central anode.

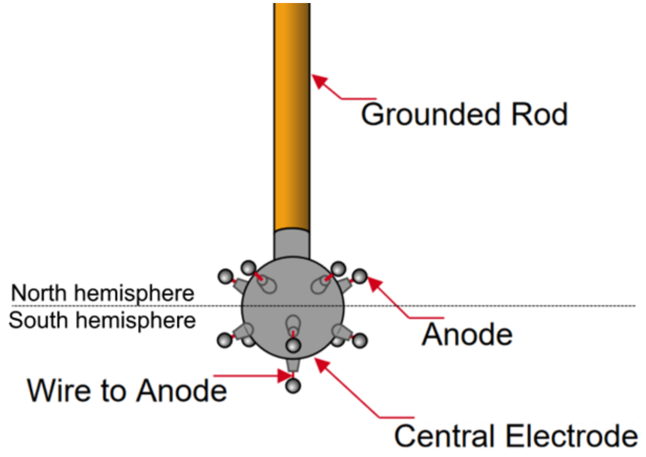


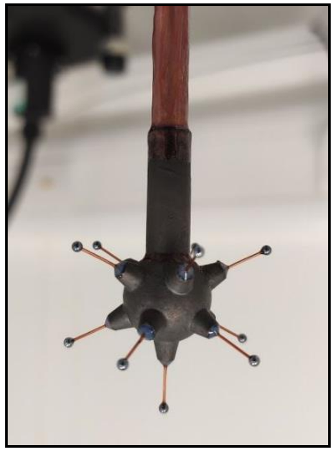


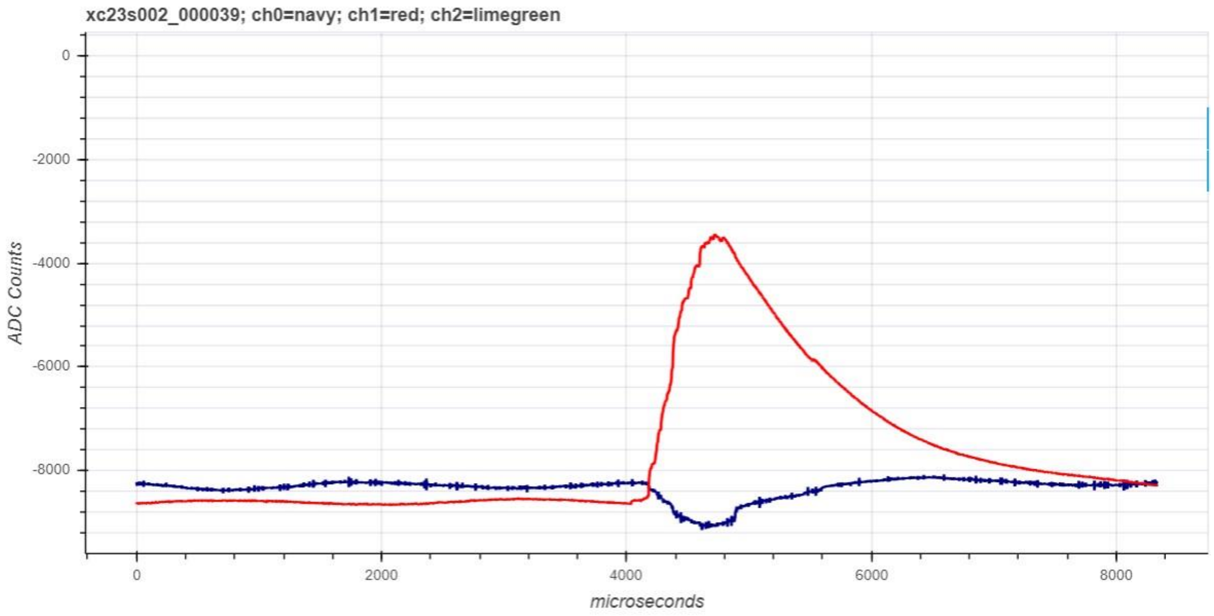


Data Processing

- The central anode sensor is divided into two channels: north and south.
- Signals from the detector are processed by a double deconvolution algorithm, which involves some smoothing.
- Data analysts must identify event populations from their pulse shapes and sizes.
- Neural networks can be applied to improve this processing.







Machine Learning for signal processing

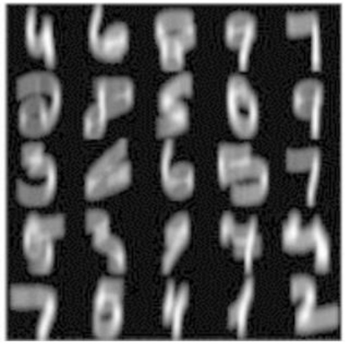
Advantages

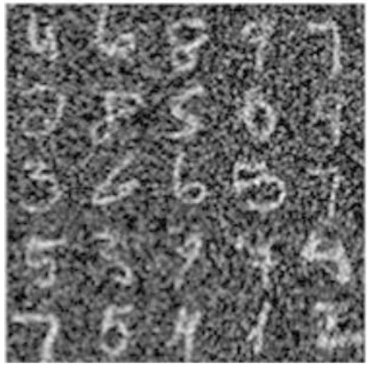
- Would likely be quicker and more efficient than traditional analysis methods.
- Removes human bias of manually selecting processing parameters.
- Any pulse shapes can be fed to the network, the model doesn't have to know what kind of pulses to expect. In this scenario, the data is unlabeled, and this is called unsupervised learning.

Challenges

- Generating appropriate training data can be a challenge. Machine learning models are very sensitive to even slight changes in datasets.
- It can take time to fine-tune the parameters of the network and find an appropriate network structure to use.

Applications of Convolutional Autoencoders





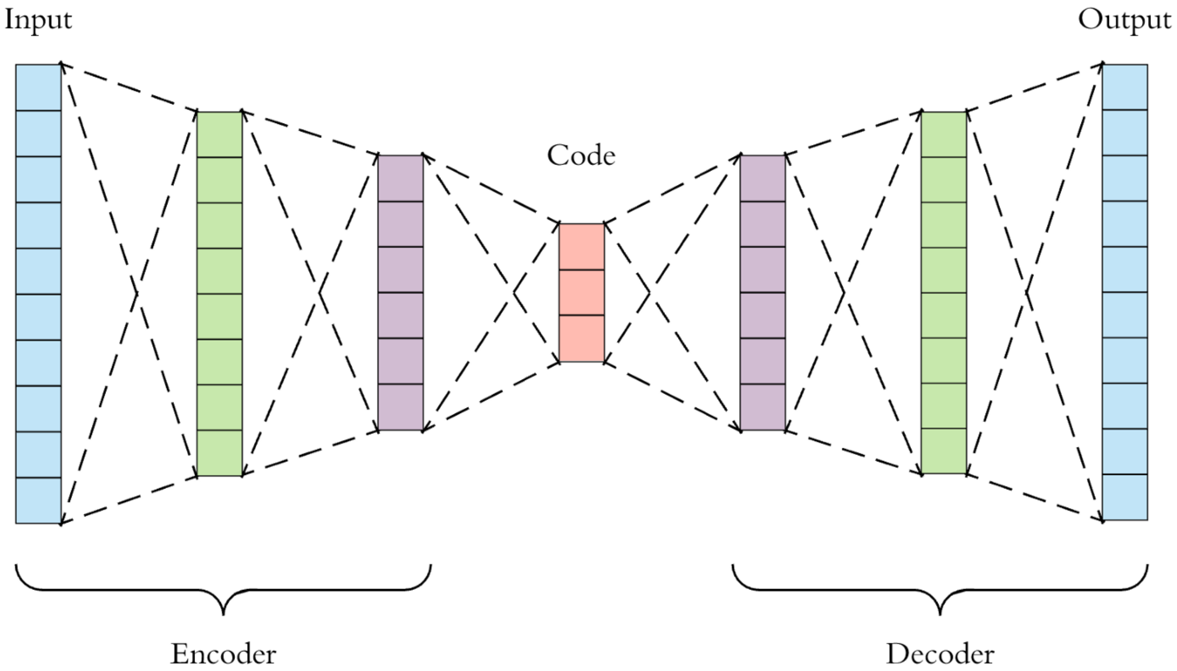
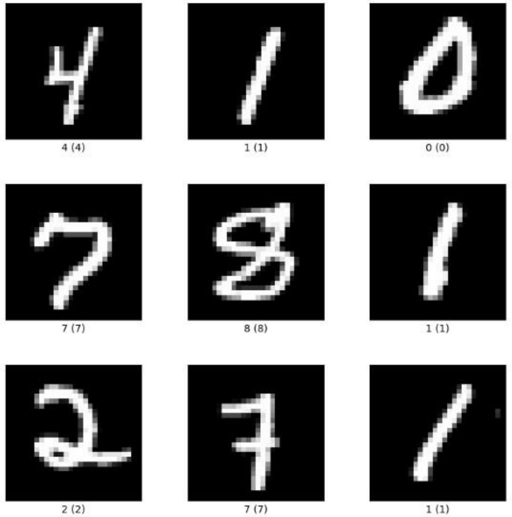


Image denoising



[The MNIST Database of Handwritten Digits]

[Learning Sparse Feature Representations Using Probabilistic Quadtrees and Deep Belief Nets]

Applications of Convolutional Autoencoders

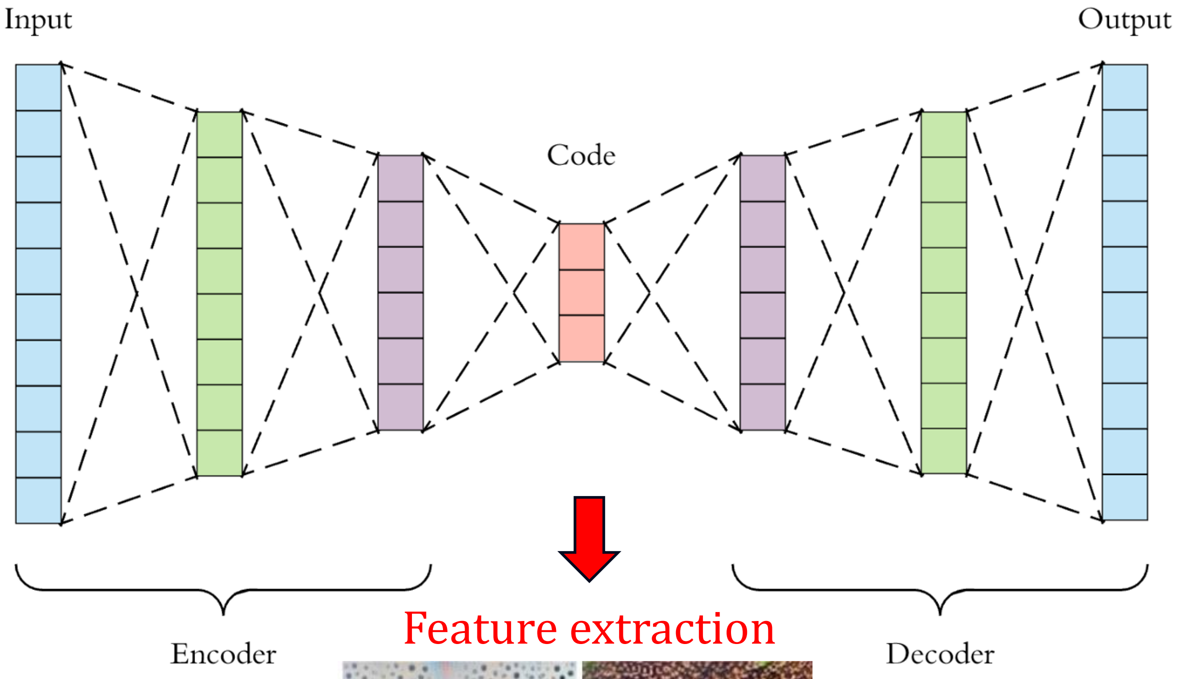
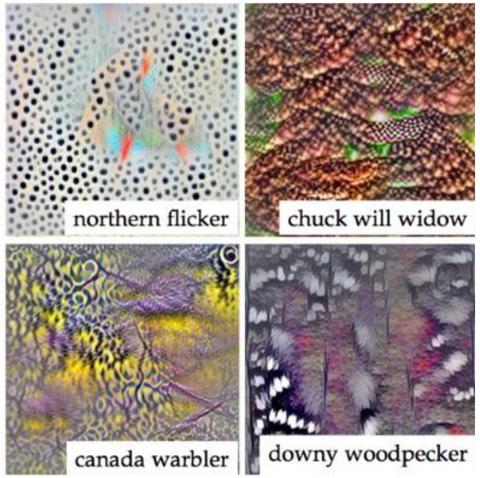


Image Classification, image reconstruction

Northern Flicker

Feature extraction



[Bilinear CNNs for Fine-grained Visual Recognition, Tsung-Yu Lin et al.]

Training the Convolutional Autoencoder Network

Prepare training data

↓

Normalization

↓

Augmentation

Training the Convolutional Autoencoder Network

Prepare training data

Normalization

Augmentation

Clean-to-clean training

Clean pulses are input, the same clean pulses are output

The model learns what clean pulses look like

Training the Convolutional Autoencoder Network

Prepare training data

Normalization

Augmentation

Clean-to-clean training

Clean pulses are input, the same clean pulses are output

The model learns what clean pulses look like

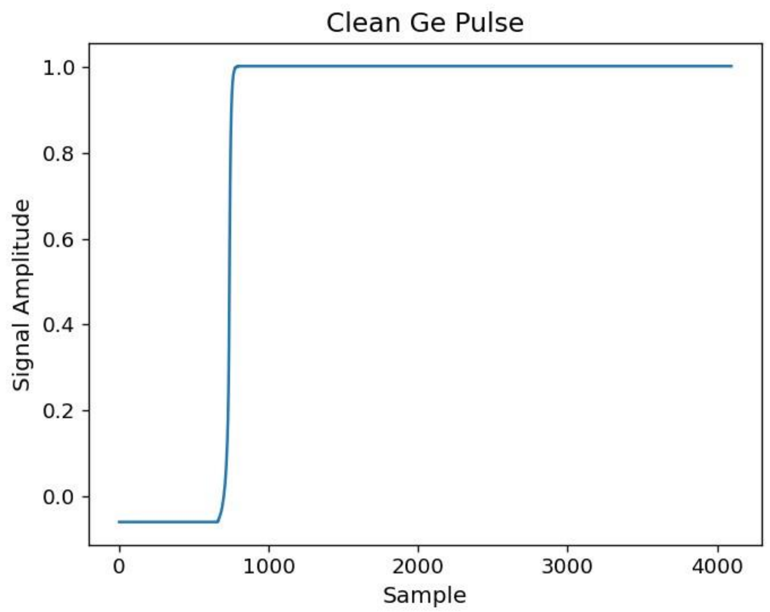
Test the trained model

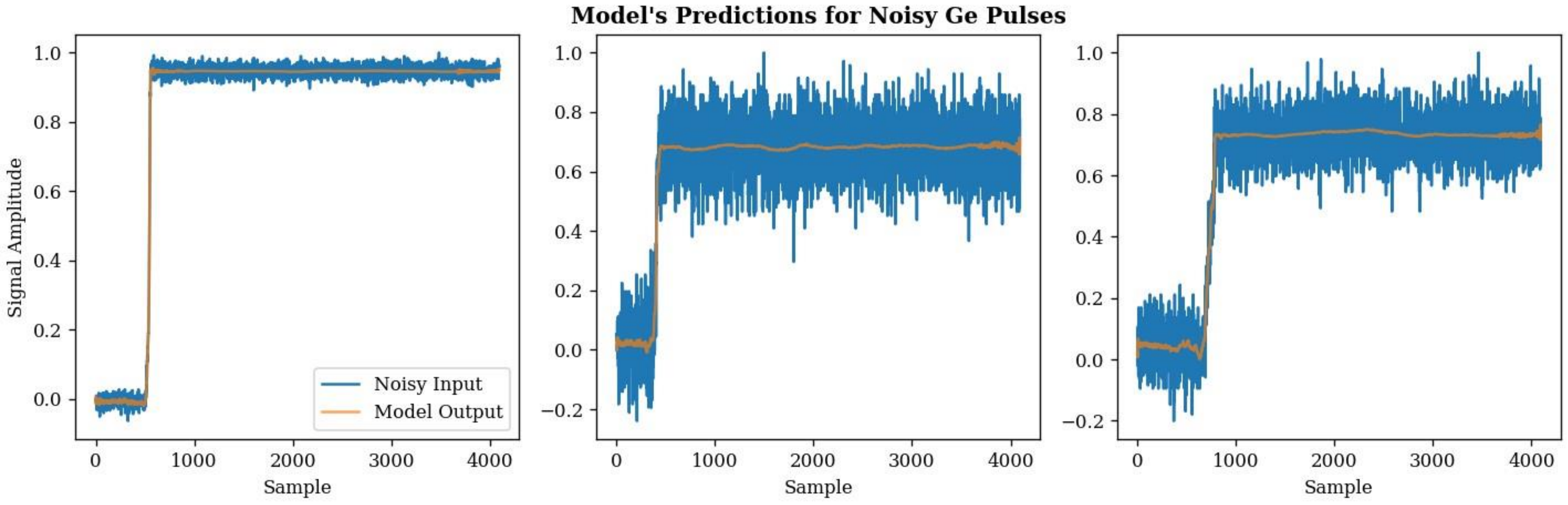
Test on noisy and clean pulses

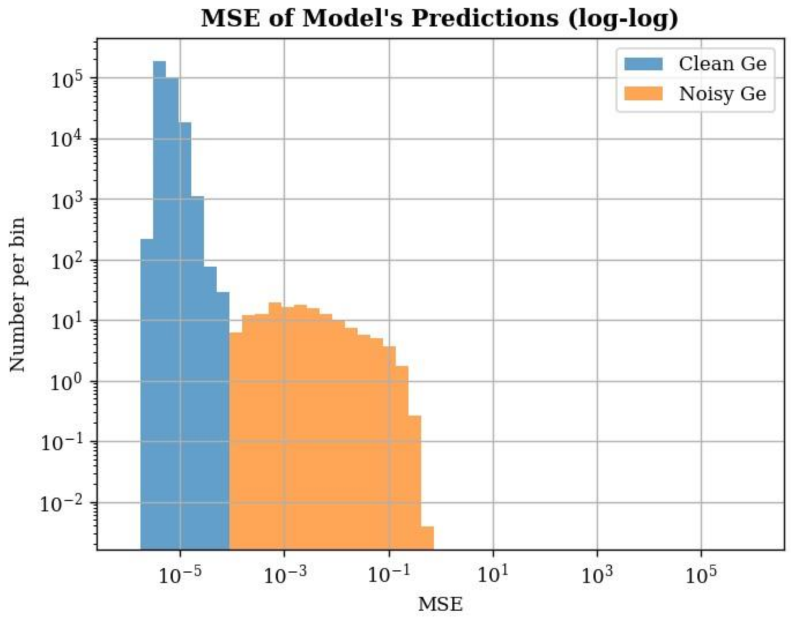
Test on different pulse shapes

Germanium Pulse Training

- As an initial proof of concept, the model trained on clean Ge pulses and tested on noisy Germanium pulses.
    - There is much more of this data-type available for training.
- Simulated clean pulses modeled from a high purity Germanium detector, with real noise added to create noisy pulses.
- Trained on 2.8 million Ge pulses.







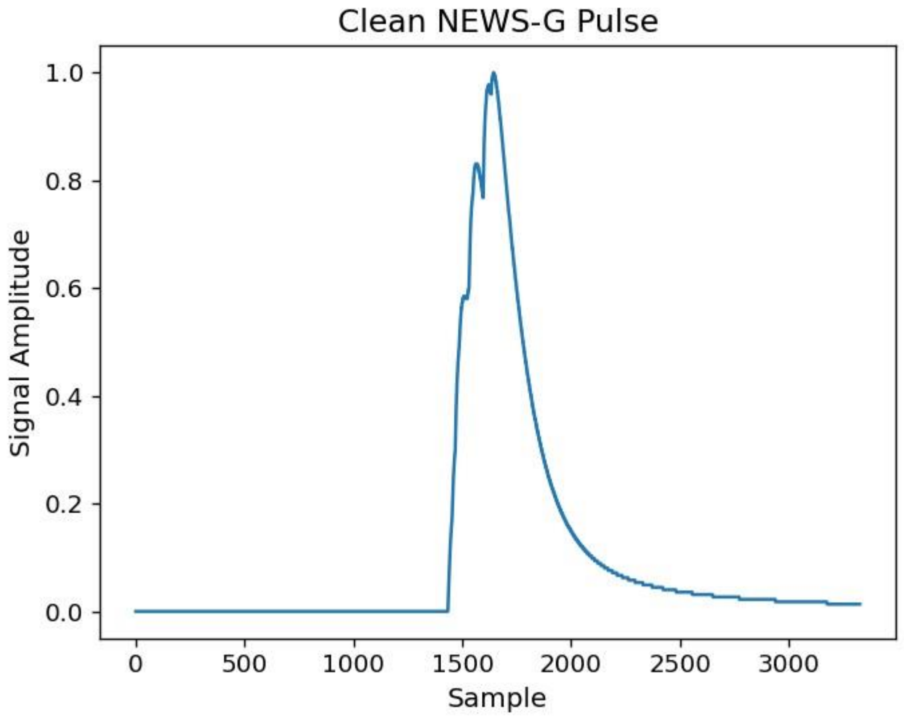
Applying the Model to NEWS-G SPC Pulses

- These pulses have a different shape, characteristic of single electron signals from SPCs.
- Training data generated by simulating an SPC detector, where noisy pulses have real noise added to the clean simulated pulses.

- Training data had to be prepared carefully:

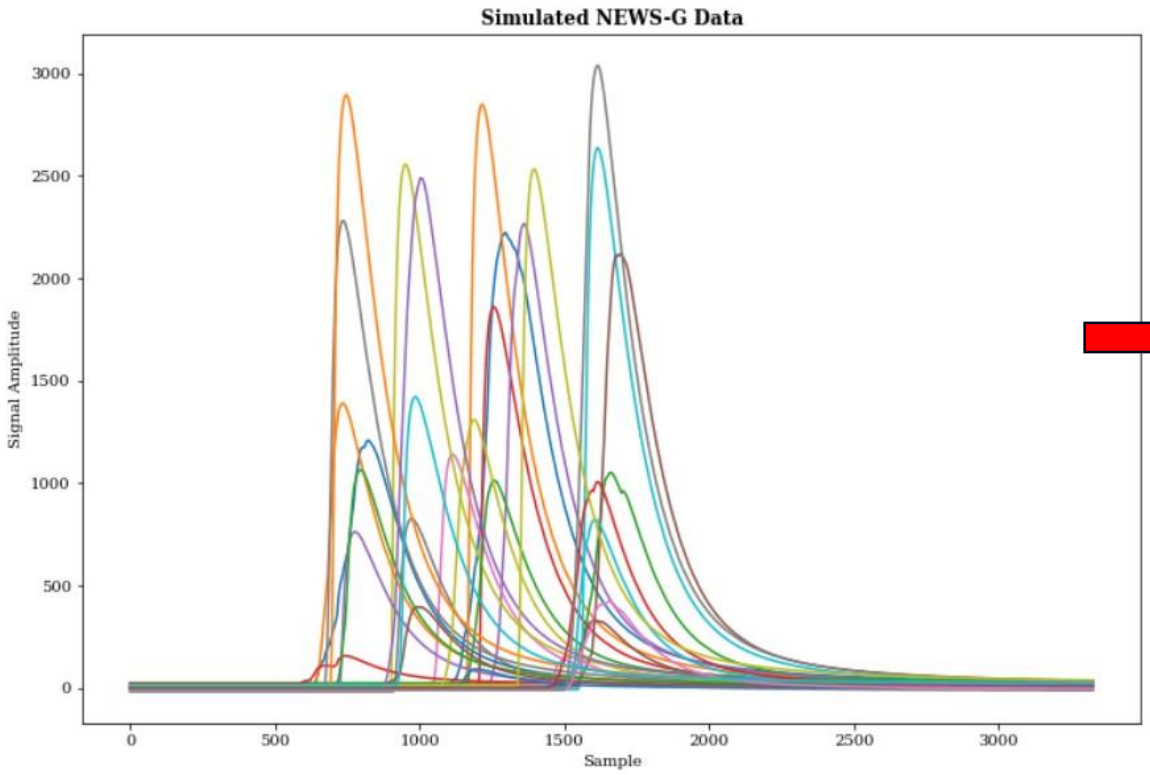
    - If there is too much variation in the baseline of the pulses, the model will not calculate training metrics. In the other extreme, if there is too little variation, the model will not reconstruct the baseline of the pulse correctly.

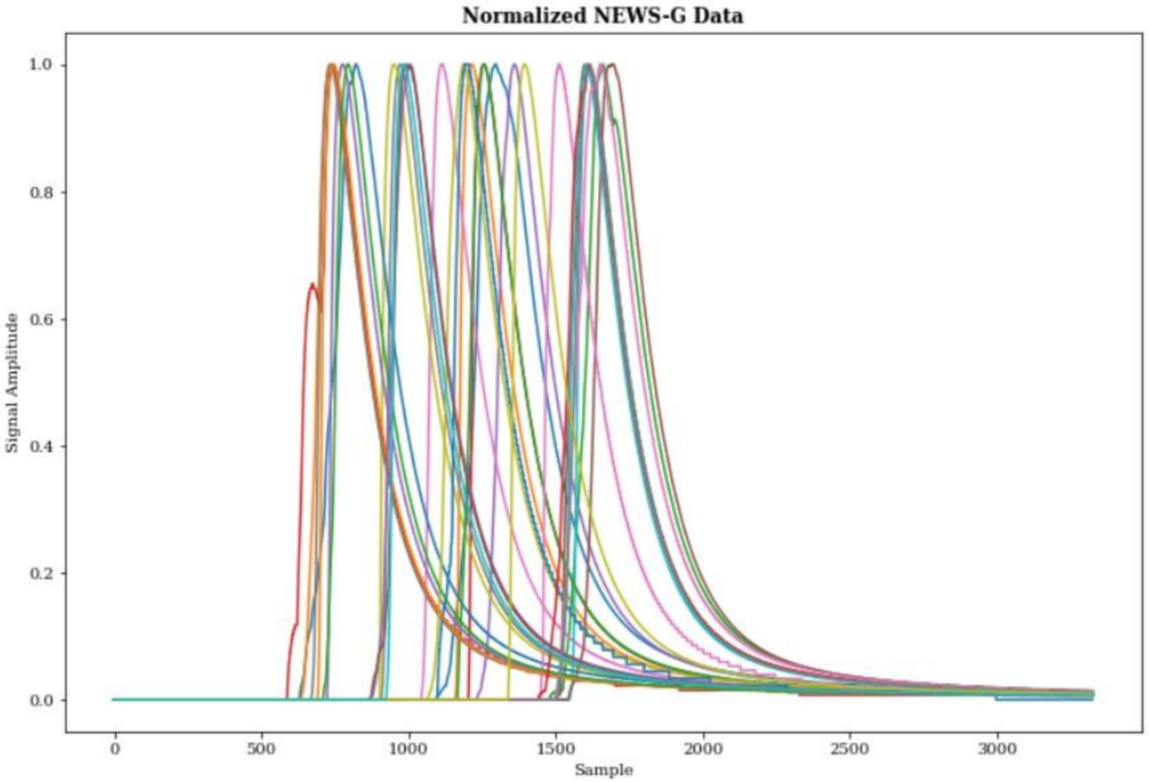
    - Also want the model to not reconstruct electronic noise present in some pulses.



Preparing the NEWS-G Training Data

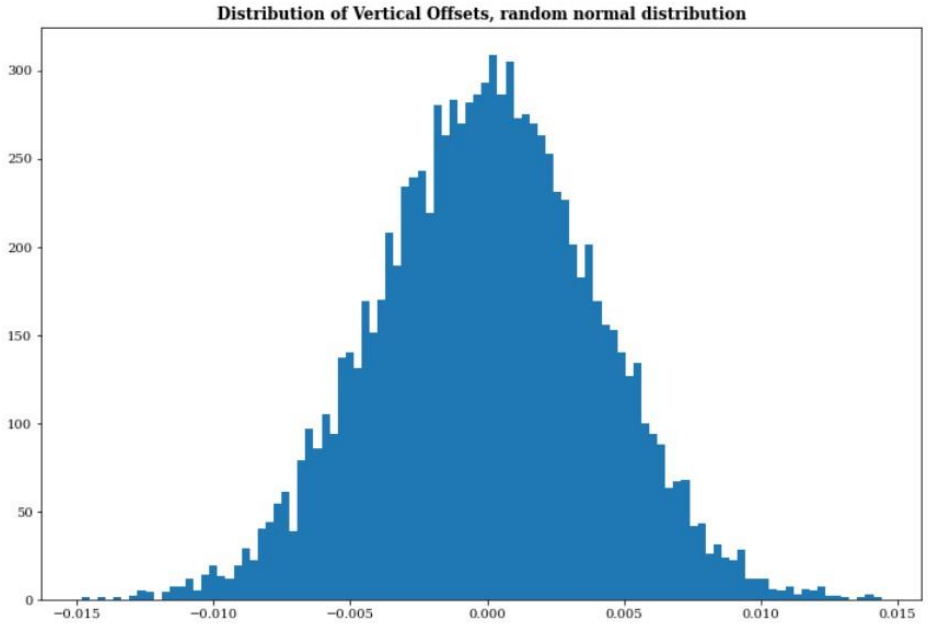
- The pulses from the simulations have a wide range of amplitudes. The model will have trouble with this much variation in the input, so the amplitude of all pulses must be normalized.

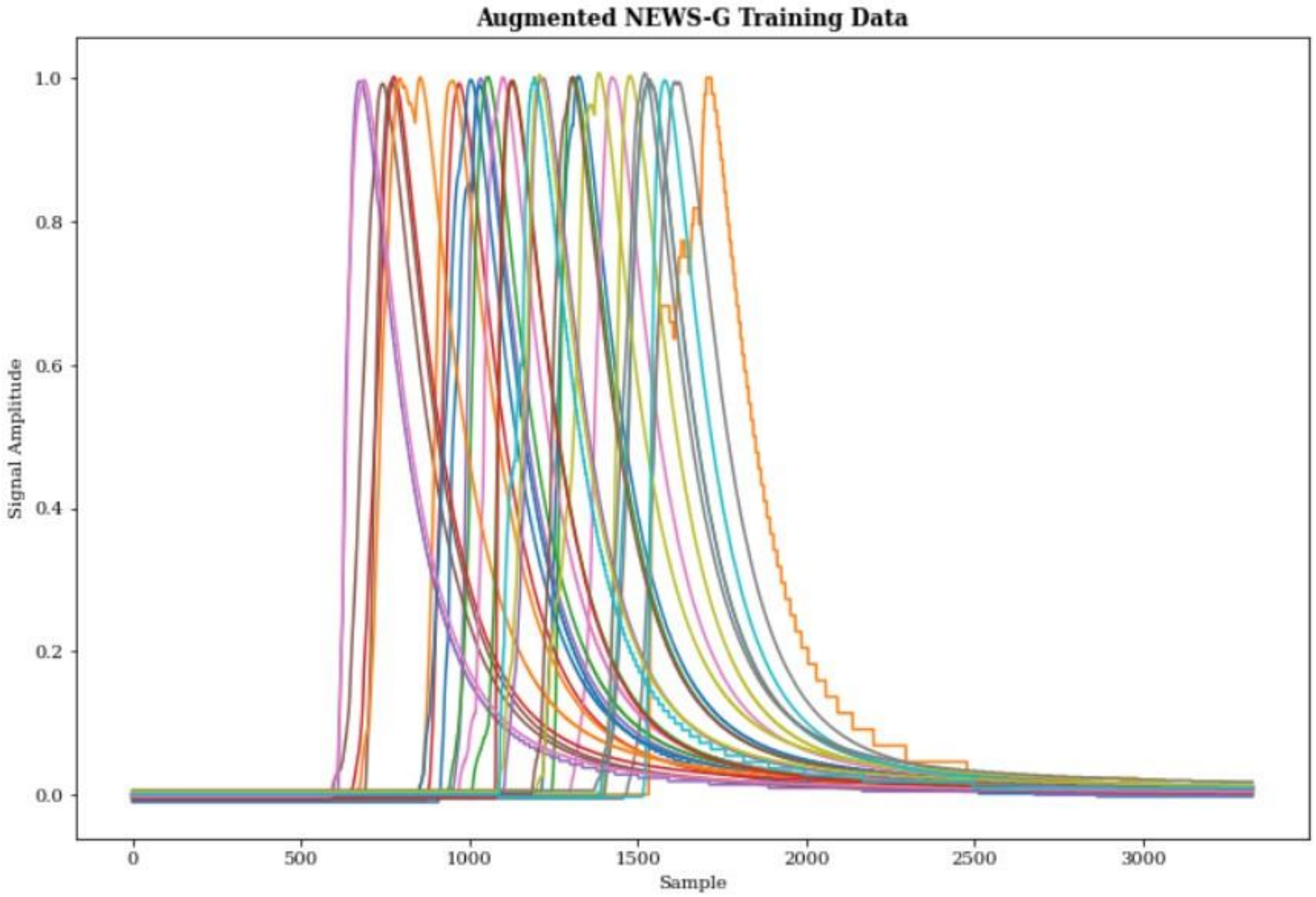




Preparing the NEWS-G Training Data

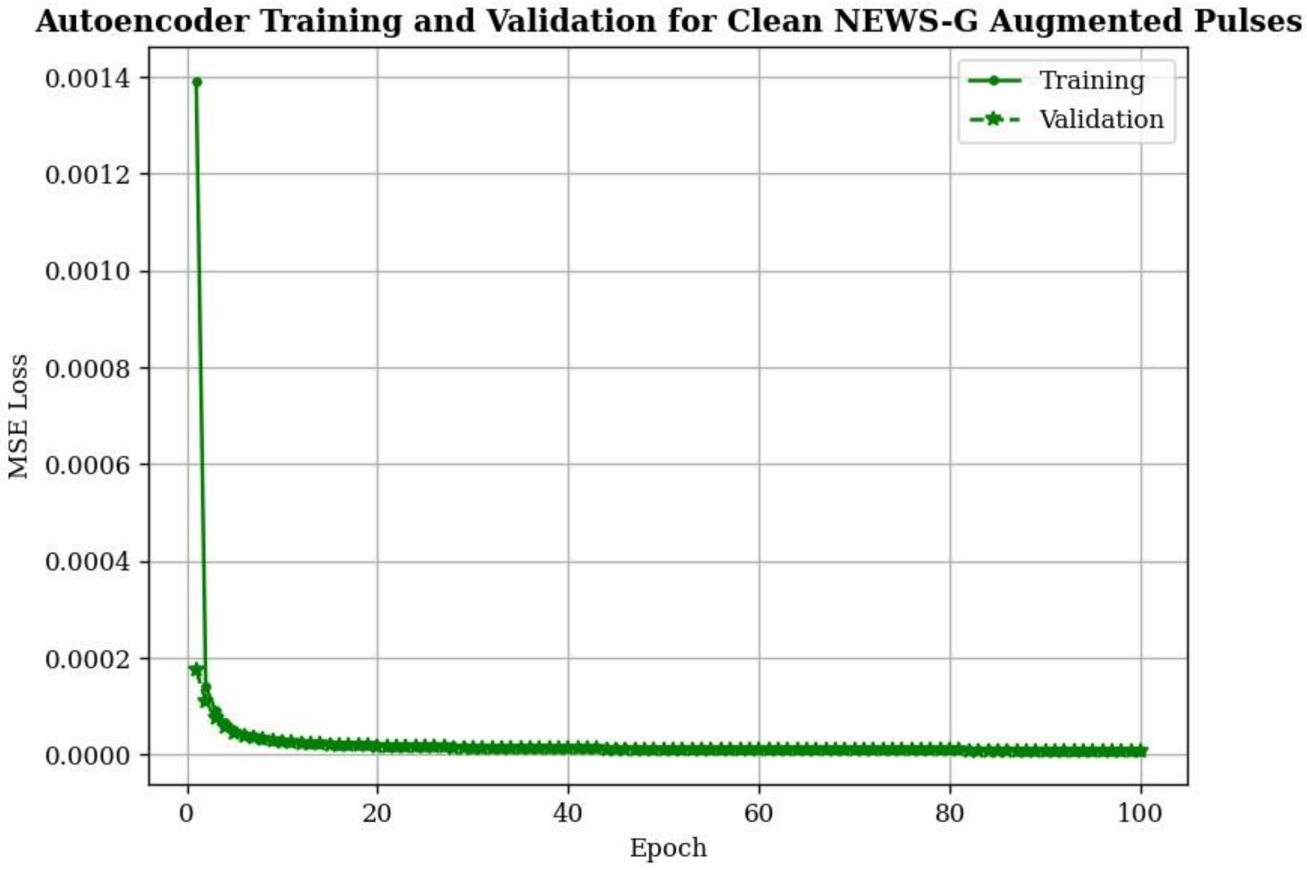
- The normalization removes the variation in vertical shifts. This is applied again via a generated random normal distribution.
- These variations are important to help the model learn what variations to expect in the real data.





NEWS-G Pulse Training Results

- Trained on 700,000 augmented NEWS-G pulses for 100 epochs, takes ~13 hours to complete.
- **Mean Squared Error Loss** – how the model's predictions are measured.
  - A better prediction = less loss.



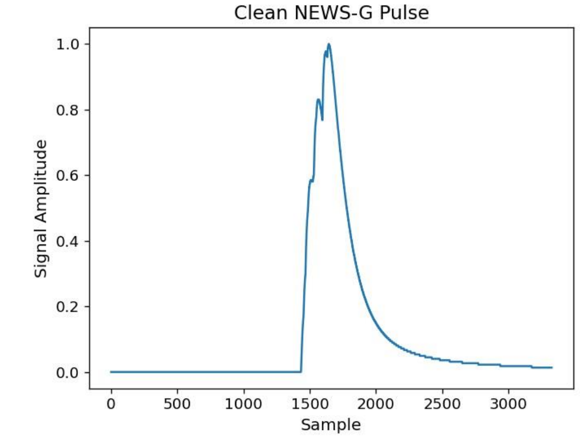
Epoch	MSE Loss	MSE Val Loss
1	1.39E-03	1.75E-04
2	1.41E-04	1.10E-04
3	9.22E-05	7.62E-05
4	6.60E-05	5.64E-05
5	5.10E-05	4.57E-05
...	...	...
96	8.82E-06	8.57E-06
97	8.78E-06	8.53E-06
98	8.74E-06	8.48E-06
99	8.69E-06	8.44E-06
100	8.65E-06	8.40E-06

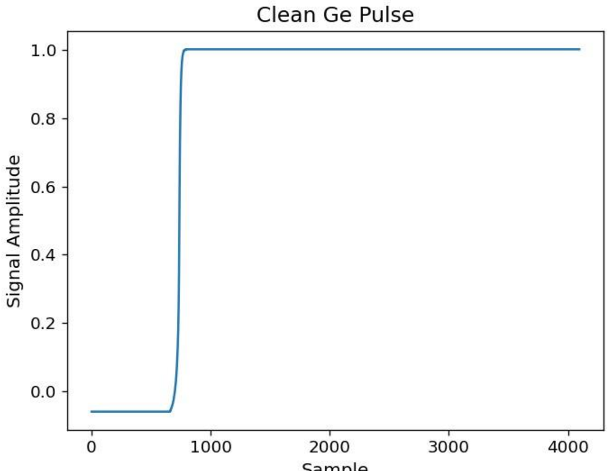
Testing the Model

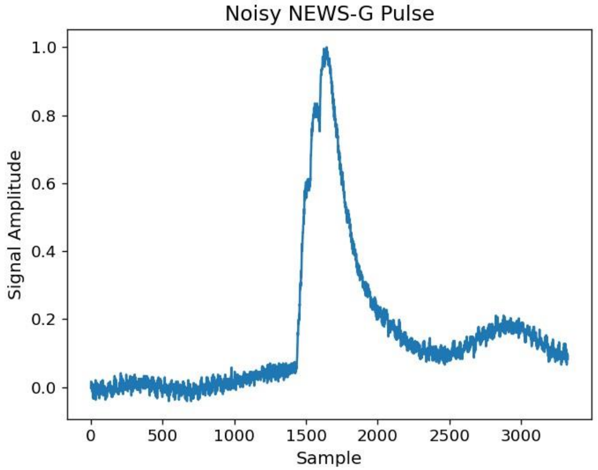
- Have four possible pulse types. Can give all of these to the model for testing. Also use the Ge pulses to see how the model performs on a different pulse shape type than what it trained on.

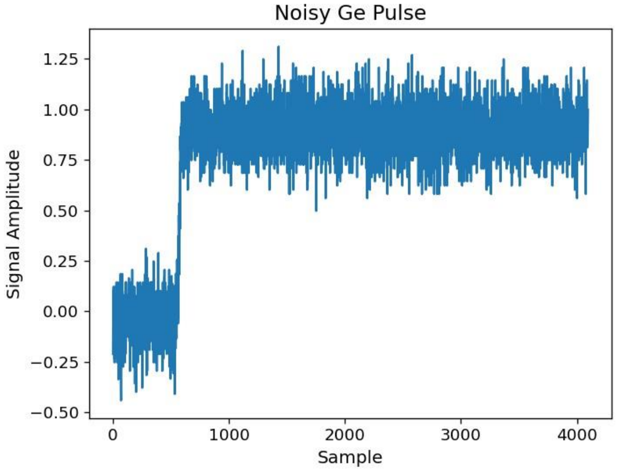
- All test data was normalized to have amplitudes of 0-1. However, this normalization is not always exact.

Test Datasets:
- 6,985 clean Ge pulses
- 6,985 noisy Ge pulses
- 10,000 clean NEWS-G pulses
- 10,000 noisy NEWS-G pulses



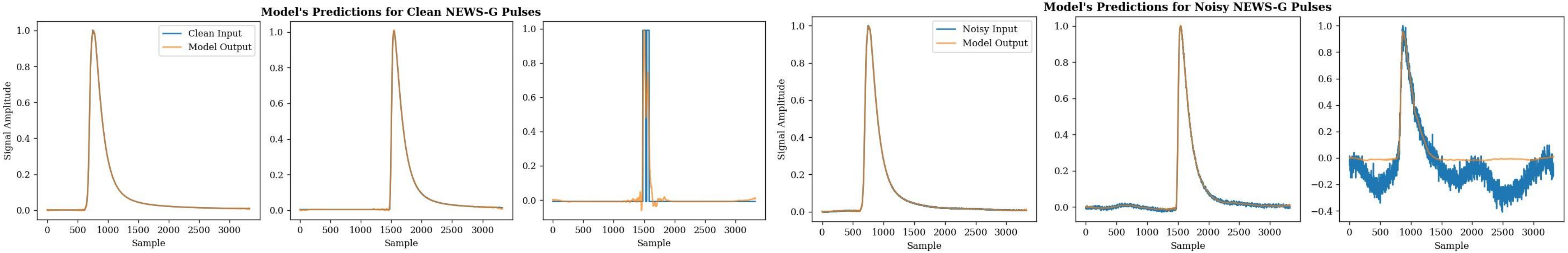




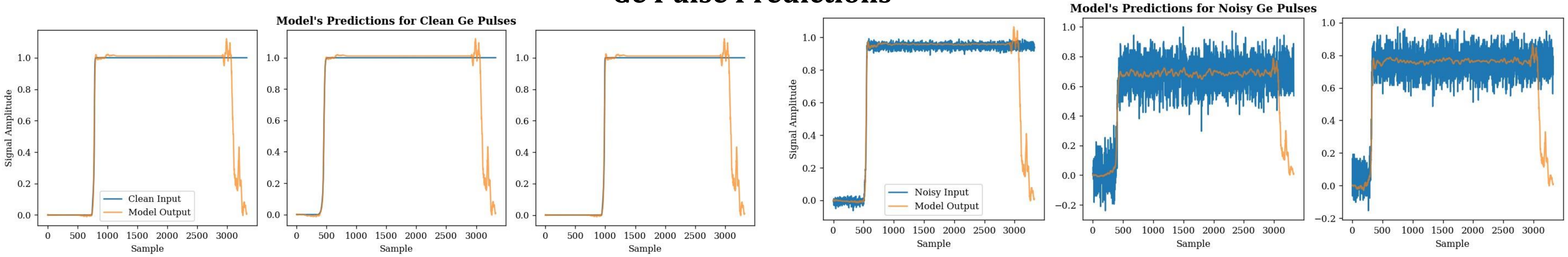


Test Results

NEWS-G Pulse Predictions

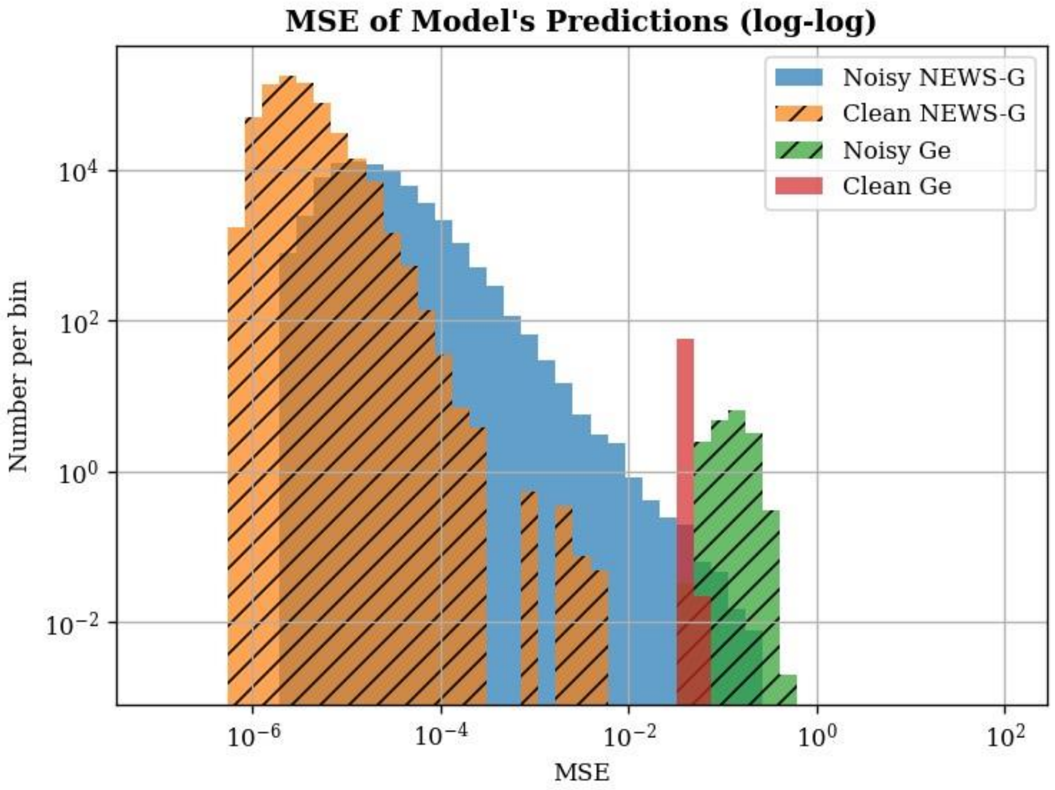


Ge Pulse Predictions



Since this model was trained on clean NEWS-G pulses, it is expected that it would have the best predictions on this pulse type.

Comparing the Model's Outputs



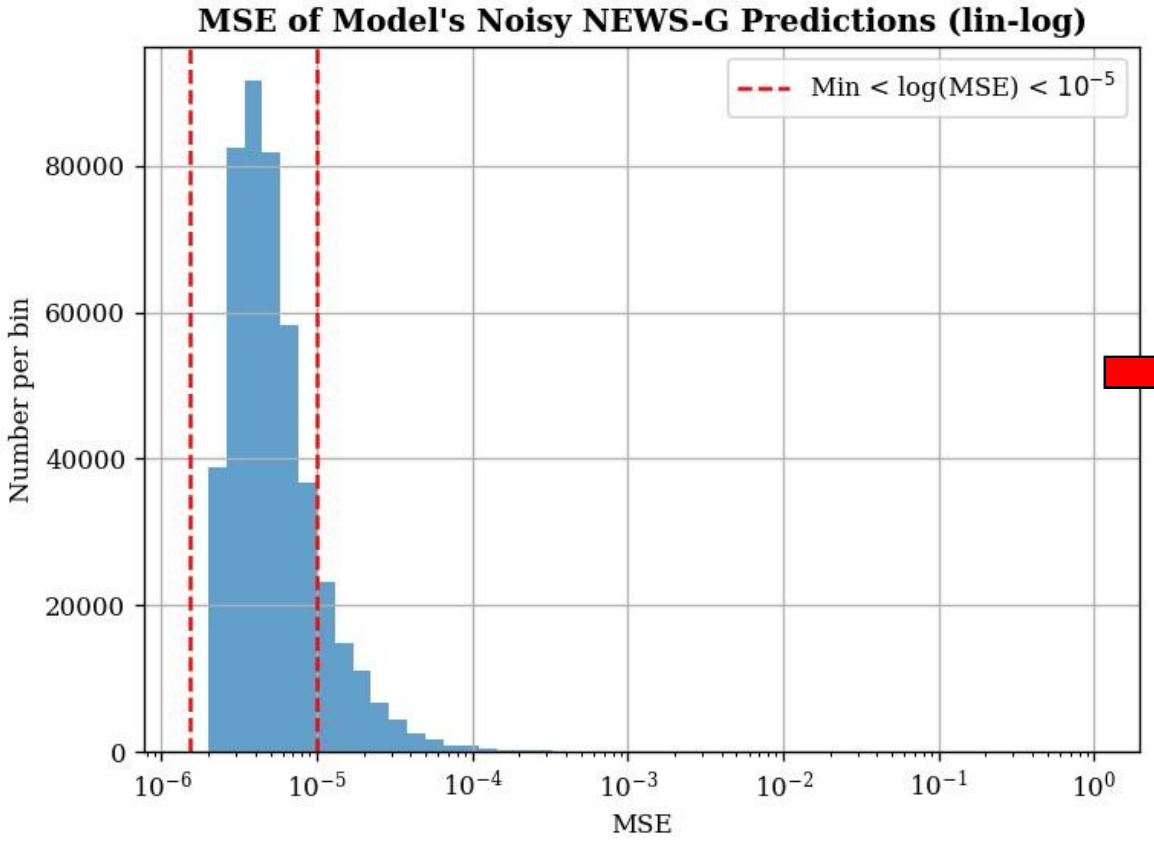
Pulse Type	Min MSE	Max MSE
Clean Ge	4.8978E-02	4.9626E-02
Noisy Ge	2.0556E-02	0.1244
Clean NEWS-G	7.0731E-07	0.0044
Noisy NEWS-G	1.5070E-06	43.8902

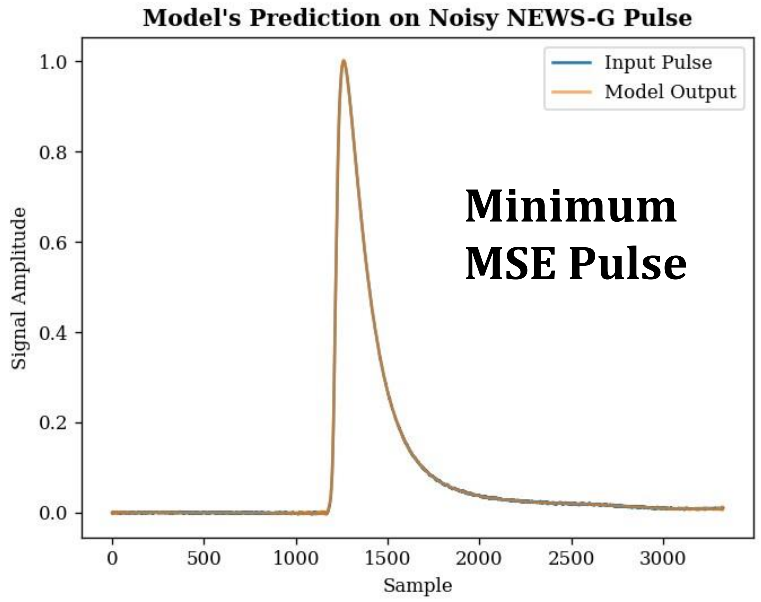
The model had the best predictions on clean NEWS-G pulses, since these are the same pulse type that the model trained on.

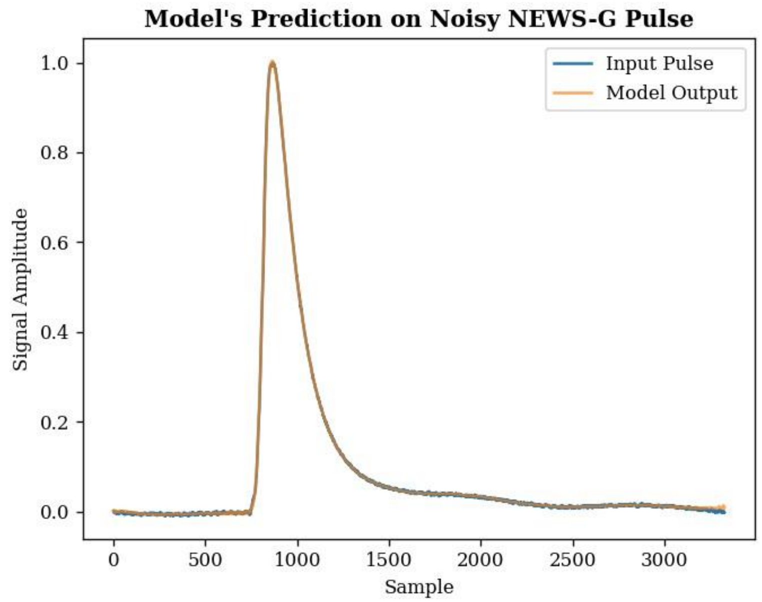
There are clear areas of different pulse shapes in the MSE. This is a good indication that the model could be used to identify different event populations and filter out unwanted pulses in the data.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Clean\ Dataset_i - Model\ Output_i)^2$$

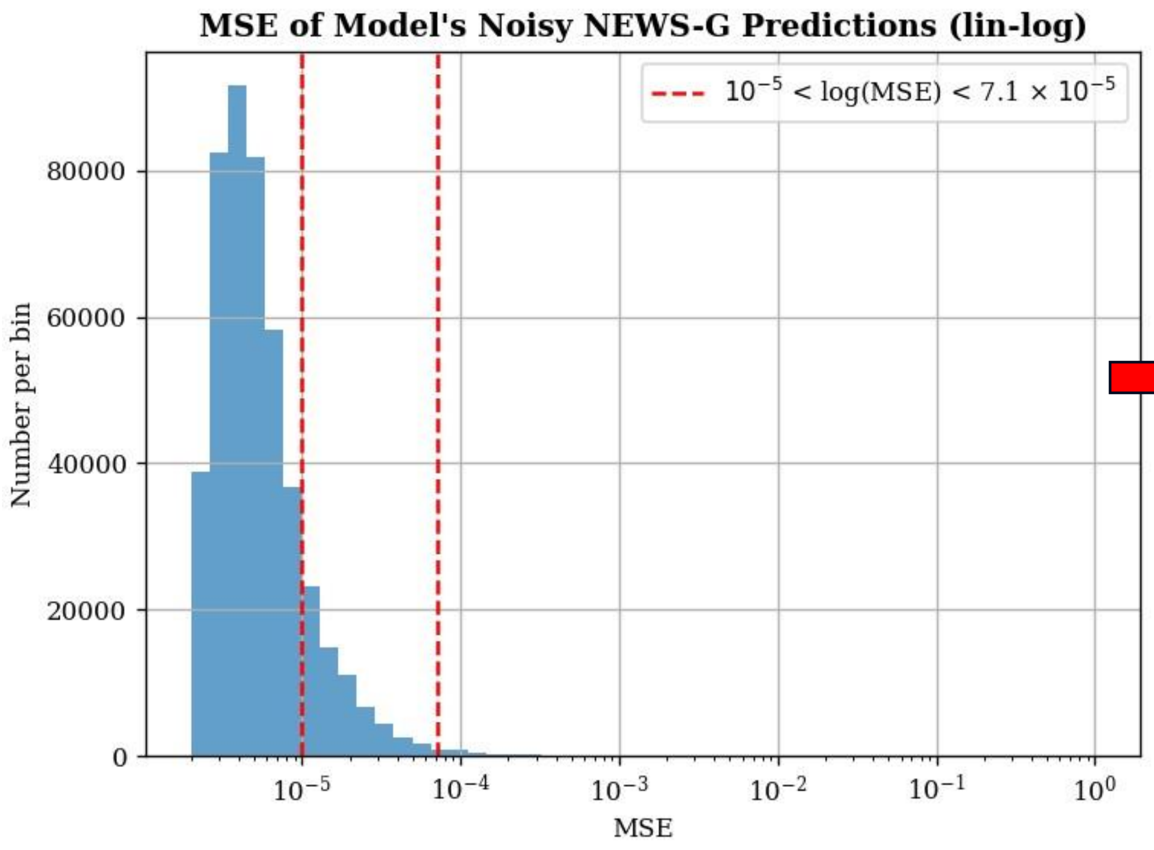
Examining Different Pulse Shapes at Different MSE's

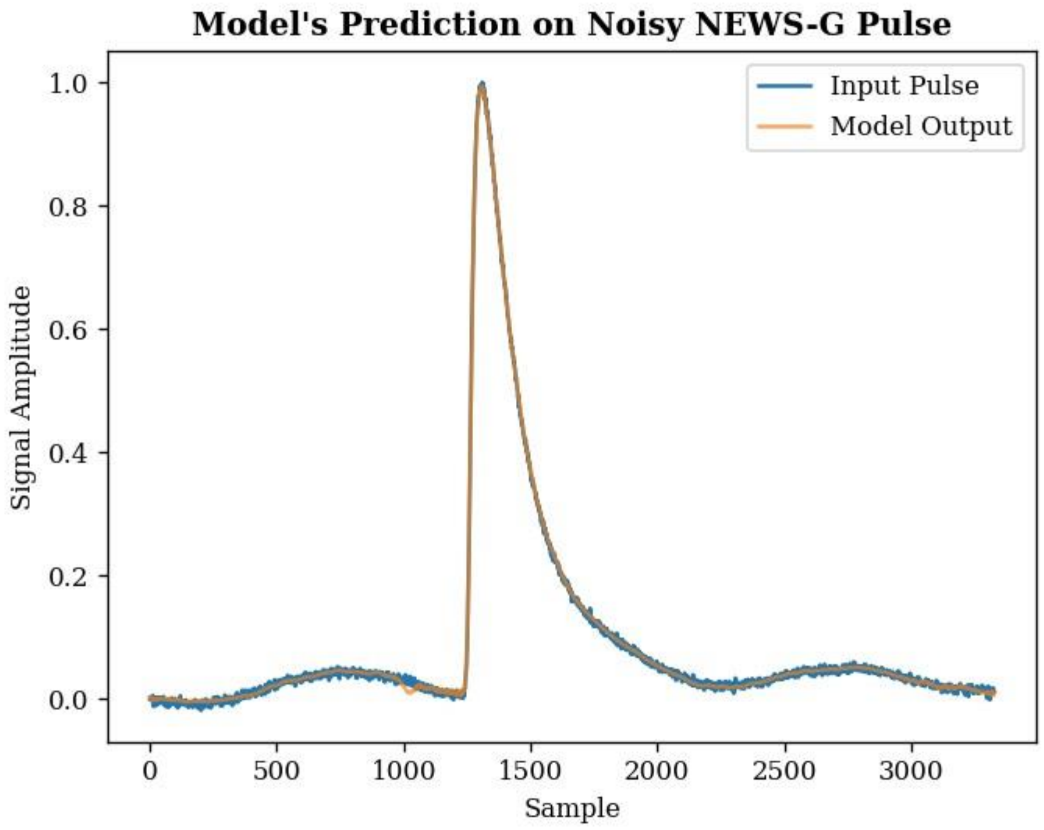




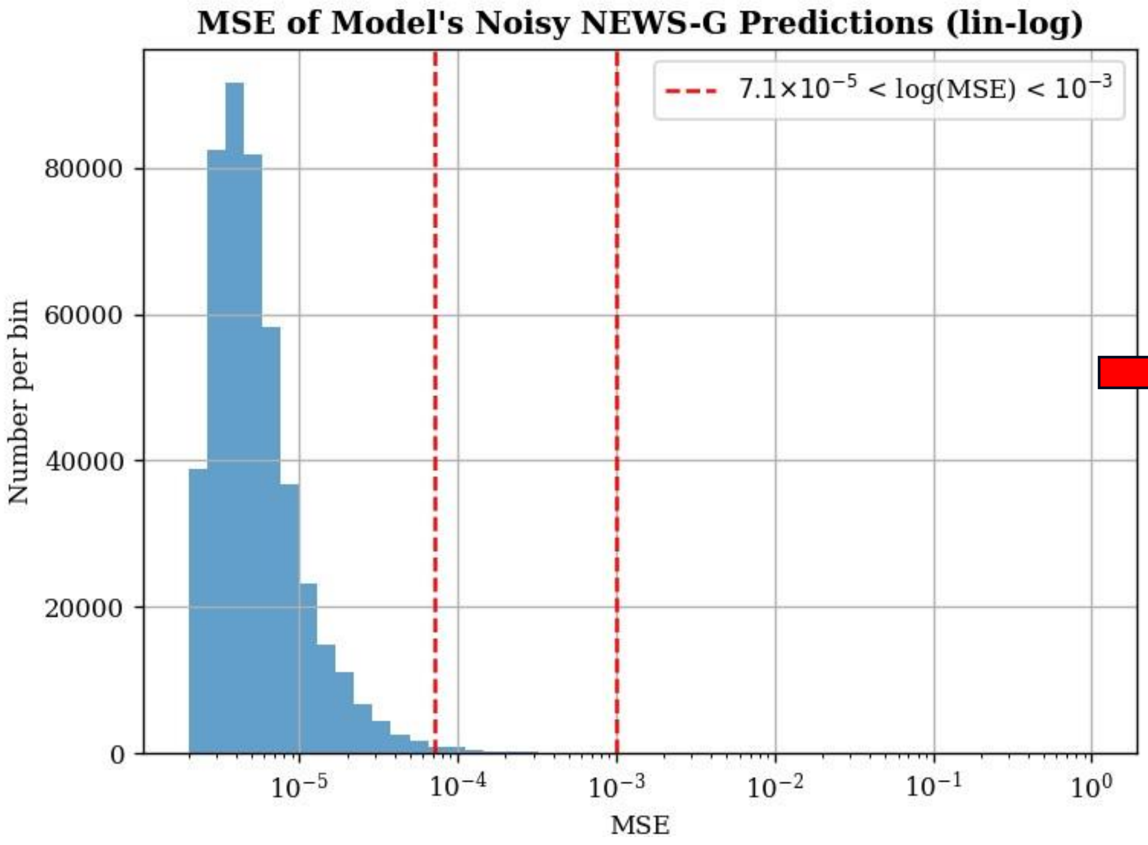


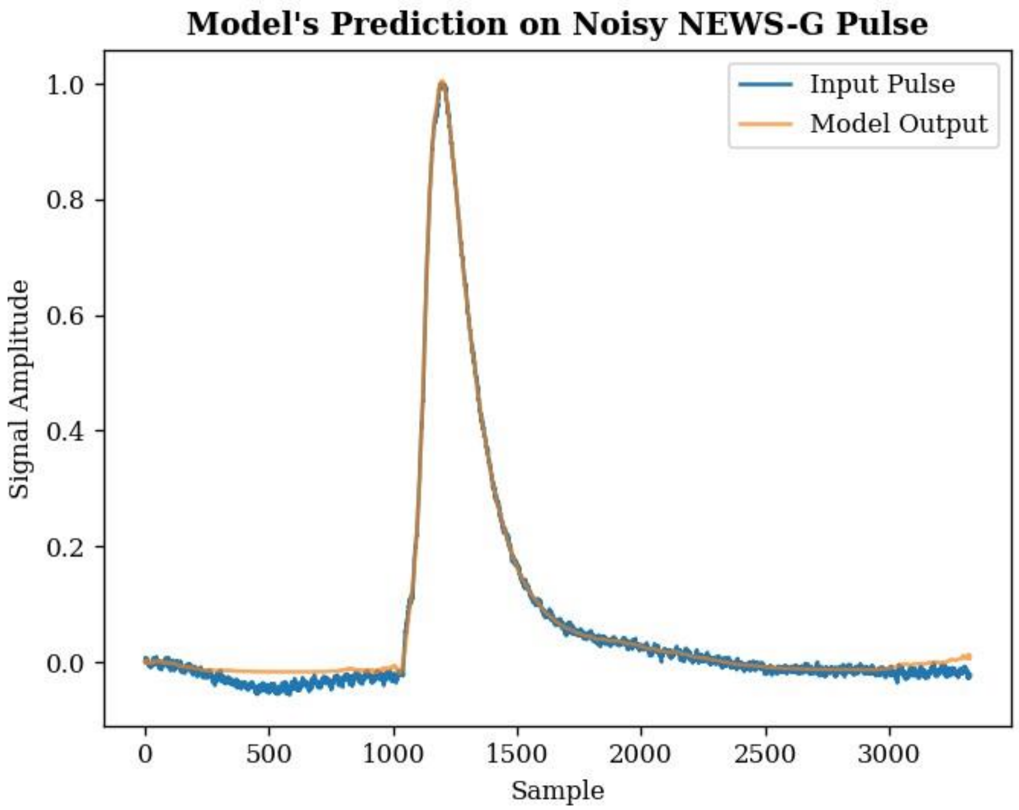
Examining Different Pulse Shapes at Different MSE’s



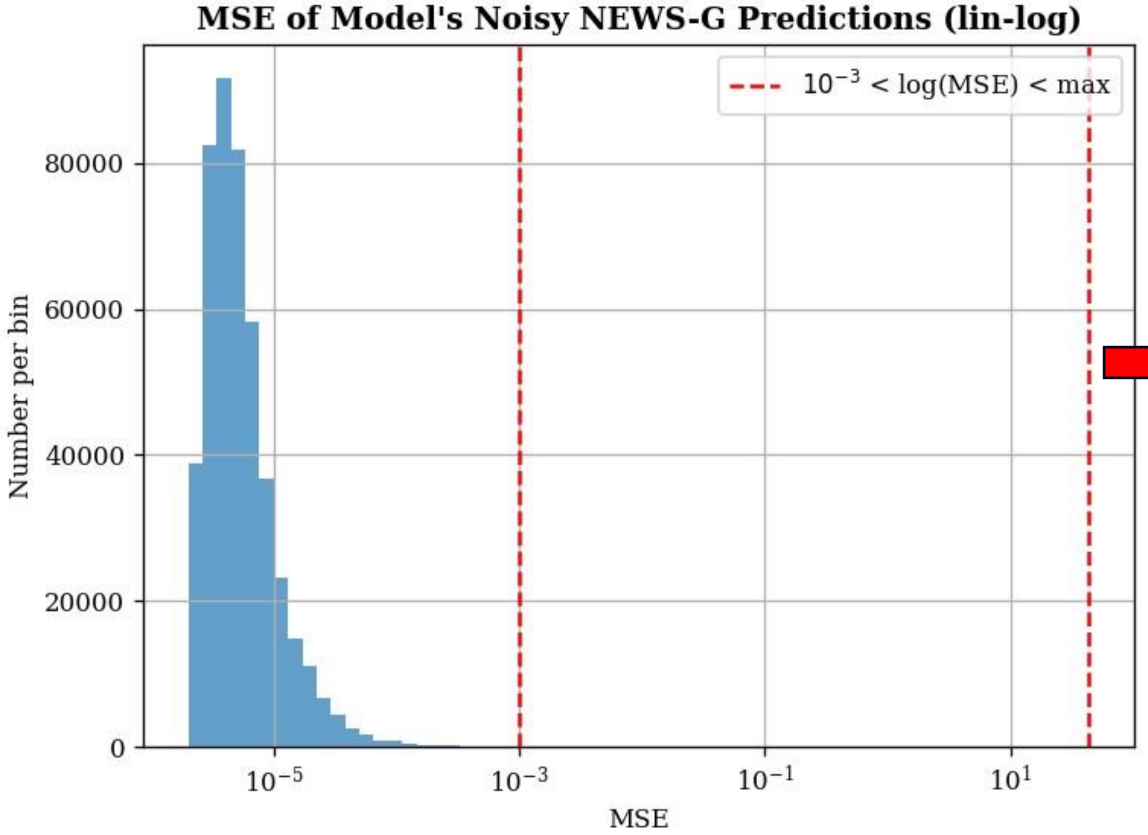


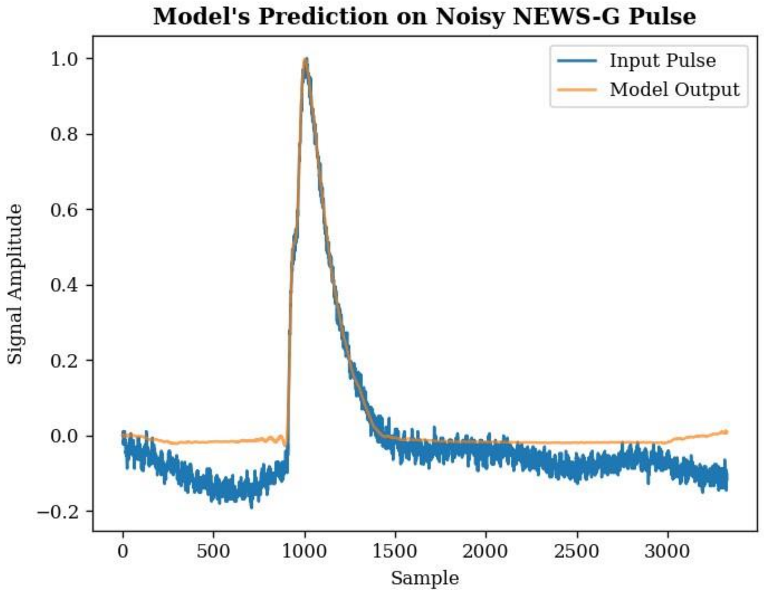
Examining Different Pulse Shapes at Different MSE’s

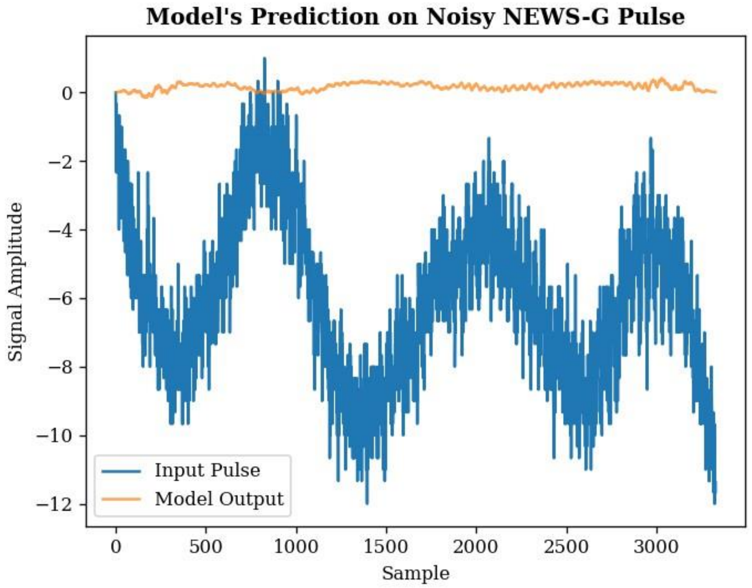




Examining Different Pulse Shapes at Different MSE’s







Maximum MSE Pulse

Conclusions

- The model's predictions heavily rely on the input data being as similar as possible to the data it was trained on.
- This is beneficial for data cleaning and filtering out non-physics pulses, since the model will have worse predictions for pulse types that differ from the training data. The model is also unsupervised, so non-physics pulse shapes do not need to be known ahead of time.
- Once trained on one type of clean data, the model can de-noise that data type much better than other data types.
- The model training may have to be further modified to reduce how noise is affecting the model's predictions.
- The next step is to test the model on real SPC data. Currently these tests were only done using simulated data.

Thank you!

Extra Slides

Model Summary

Layer (type)	Output Shape	Param #
autoencoder_input (InputLayer)	[(None, None)]	0
expand_dims_for_conv1d (Lambda)	(None, None, 1)	0
conv1d (Conv1D)	(None, None, 8)	16
activation (Activation)	(None, None, 8)	0
conv1d_1 (Conv1D)	(None, None, 16)	1168
activation_1 (Activation)	(None, None, 16)	0
average_pooling1d (AveragePooling1D)	(None, None, 16)	0
conv1d_2 (Conv1D)	(None, None, 32)	8736
activation_2 (Activation)	(None, None, 32)	0
average_pooling1d_1 (AveragePooling1D)	(None, None, 32)	0
conv1d_3 (Conv1D)	(None, None, 64)	67648
activation_3 (Activation)	(None, None, 64)	0
average_pooling1d_2 (AveragePooling1D)	(None, None, 64)	0
conv1d_4 (Conv1D)	(None, None, 32)	67616
encoder_output (Activation)	(None, None, 32)	0
conv1d_transpose (Conv1DTranspose)	(None, None, 32)	33824
activation_4 (Activation)	(None, None, 32)	0
up_sampling1d (UpSampling1D)	(None, None, 32)	0
conv1d_transpose_1 (Conv1DTranspose)	(None, None, 64)	67648
activation_5 (Activation)	(None, None, 64)	0
up_sampling1d_1 (UpSampling1D)	(None, None, 64)	0
conv1d_transpose_2 (Conv1DTranspose)	(None, None, 32)	34848
activation_6 (Activation)	(None, None, 32)	0
up_sampling1d_2 (UpSampling1D)	(None, None, 32)	0
conv1d_transpose_3 (Conv1DTranspose)	(None, None, 16)	4624
activation_7 (Activation)	(None, None, 16)	0
conv1d_5 (Conv1D)	(None, None, 1)	17
autoencoder_output (Lambda)	(None, None)	0

Total params: 286,145
Trainable params: 286,145
Non-trainable params: 0