Finanziato dall'Unione europea
NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani
PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Hyperparameter Optimization for Deep Learning Model Using High Performance Computing

INFN
Istituto Nazionale di Fisica Nucleare

Speaker : Muhammad Numan Anwar
On the Behalf of FCC Collaborator: Nicola De Filippis, Marcello Abbrescia, Domenico Diacono, Guang Zhao, Mingyi Dong, Shengsen Sun, Linghui Wu, Francesco Grancagnolo

中国科学院高能物理研究所
Institute of High Energy Physics
Chinese Academy of Sciences

1st AI-INFN User Forum,Bologna 12 June 2024

Outline

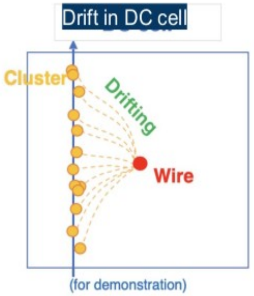
- Cluster Counting
- Simulation based on Garfield ++
- Long Short Term Memory (LSTM) Model for Peak Finding Algorithm
- Convolution Neural Network (CNN) Model for Clusterization Algorithm
- Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources
- Future Planning

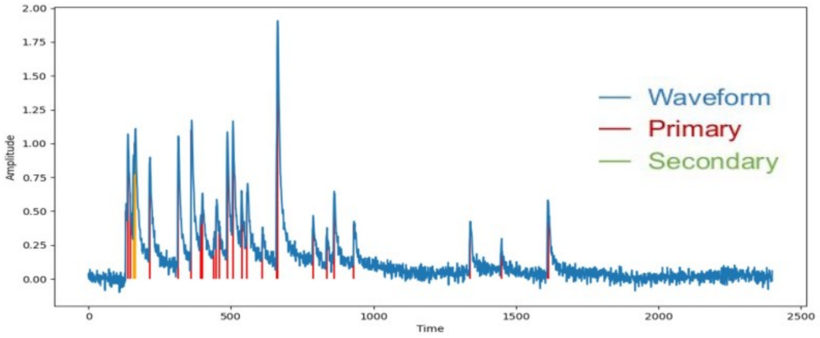
Main Goal of the Talk

- **The main goal of the talk is to train neural network models, such as the Long Short-Term Memory (LSTM) Model and Convolutional Neural Network (CNN) Model, using various hyperparameters like loss functions, activation functions, different numbers of neurons, batch sizes, and varying numbers of epochs etc. These models are trained for a two-step reconstruction algorithm, which involves peak finding and clusterization**
- **For the peak finding algorithm, a trained LSTM model is used to discriminate between ionization signals (primary and secondary peaks) and noise in the waveform, addressing a classification problem**
- **Concurrently, a Convolutional Neural Network model is utilized to determine the number of primary ionization clusters based on the detected peaks, dealing with a regression problem**
- **It should be noted that the trained models (LSTM and CNN) are applied to simulations based on Garfield++**
- **Additionally, I designed a task related to the simultaneous submission of several jobs using local HPC resources in order to train different models, such as long short-term memory models and convolutional neural networks, for classification and clustering tasks**
- **Morever, it requires managing different resources such as memory requests, job duration, and CPU usage etc**
- **At the end, I should select best model based on the different evaluation metrics like F1 score, Precision, Recall, Area under the curve (AUC) and Efficiency*Purity**

Background Knowledge

Cluster Counting in Drift Chambers





- **A charged particle is passed through mixture of gases (90% He and 10% C_4H_{10}) generate**
- **electron-ion pairs causing a read out signal (induce current)**

Task:

- Both primary electrons and secondary electrons contribute to peaks in the waveform
- Find the number of peaks from primary electrons

Two step reconstruction algorithm:

- Peak finding: Find all peaks (primary and secondary) in the waveform
- Clusterization: Determine the primary peaks from the founded peaks in step 1

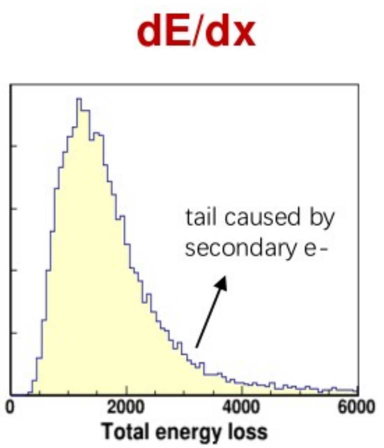
Cluster Counting in Drift Chambers

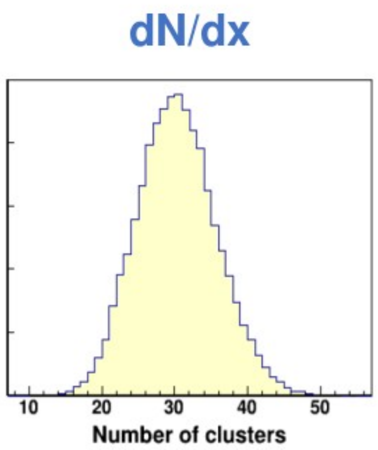
dE/dx: Measure the ionization by total energy loss

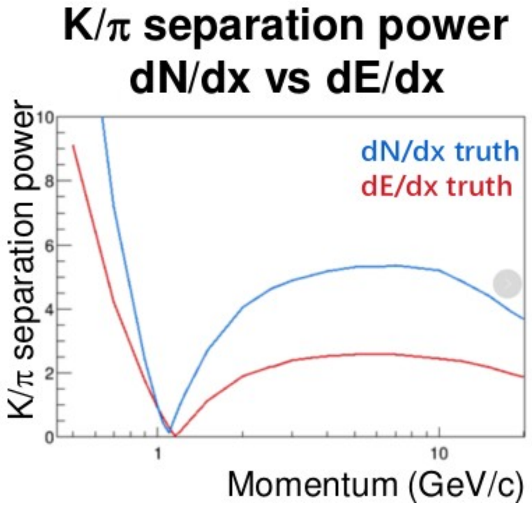
- Landau distributed
- Large fluctuation from many sources

dN/dx: Measure the ionization by number of primary ionizations

- Poisson distributed
- Small fluctuation; Potentially improve the resolution by a factor of 2







Simulation Based on Garfield ++

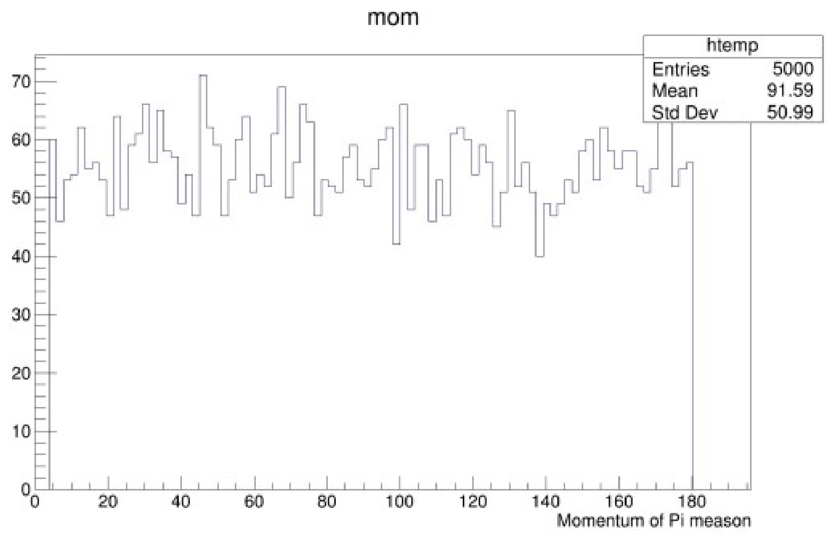
Simulation Based on Garfield ++

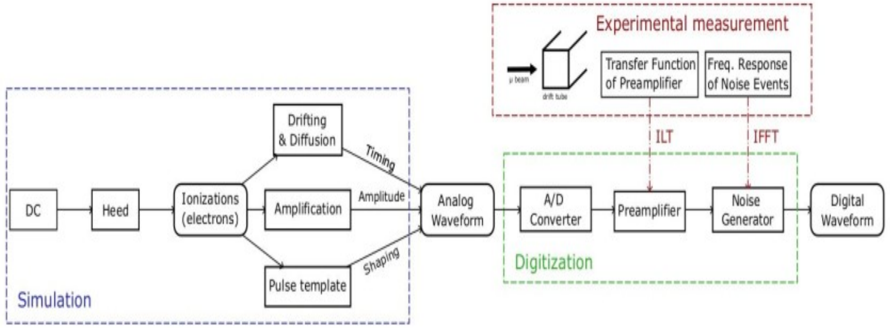
- Pi meson is passed through mixture of gas having 90% He and 10% Isobutane C_4H_{10} by using a geometry of drift tubes mimicking what was used for the beam test at CERN in 2023

- The simulation parameters included a cell size of 1.5 cm, a sampling rate of 1.2 GHz, a time window of 2000 ns, and momentum pi-meson particles ranging from 4 to 180 GeV/c. The simulation was conducted using Garfield++

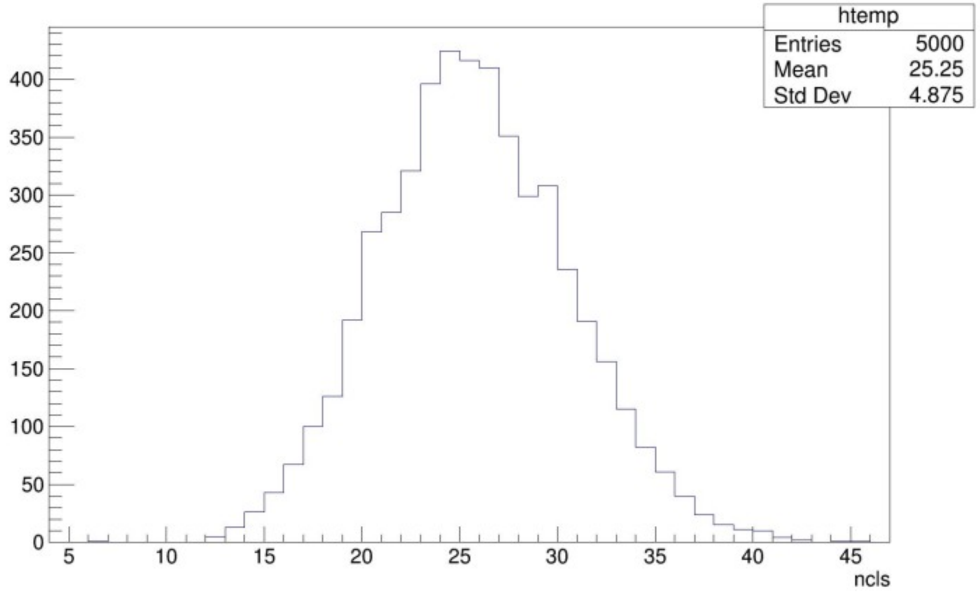
- Following the simulation in Garfield++, I proceeded to plot various results for the study of the cluster counting techniques

- The simulation package creates analog induced current waveforms from ionizations. The digitization package incorporates electronics responses taken from experimental measurements and generates realistic digital waveforms

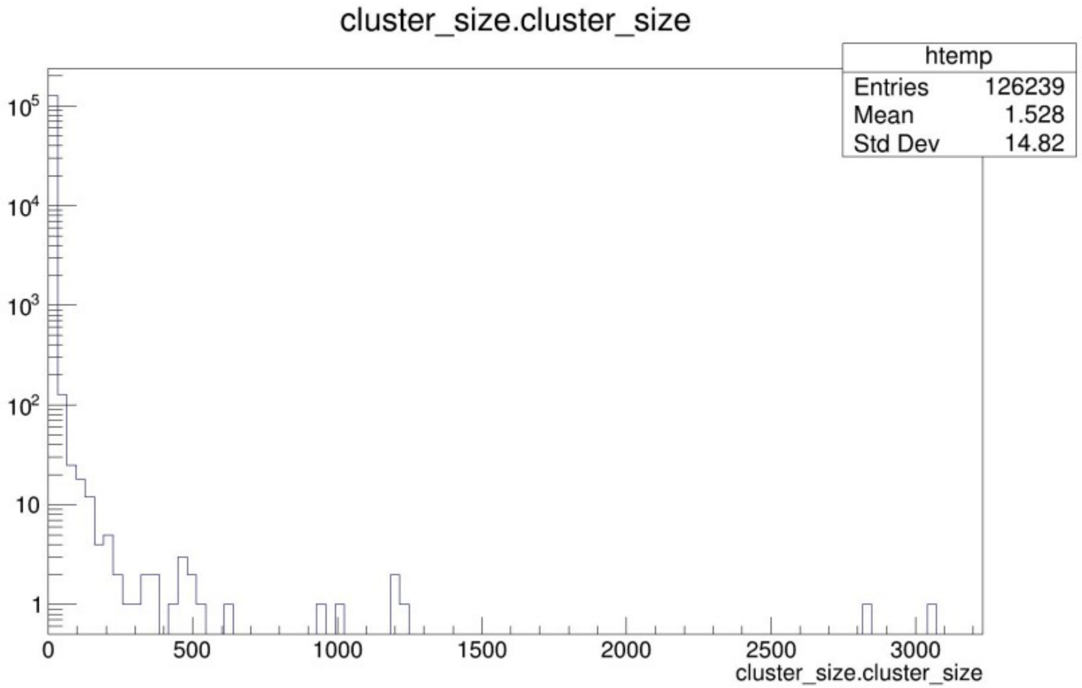




Simulation Based on Garfield ++



The above distribution shows the number of primary ionization clusters with mean value 25.25



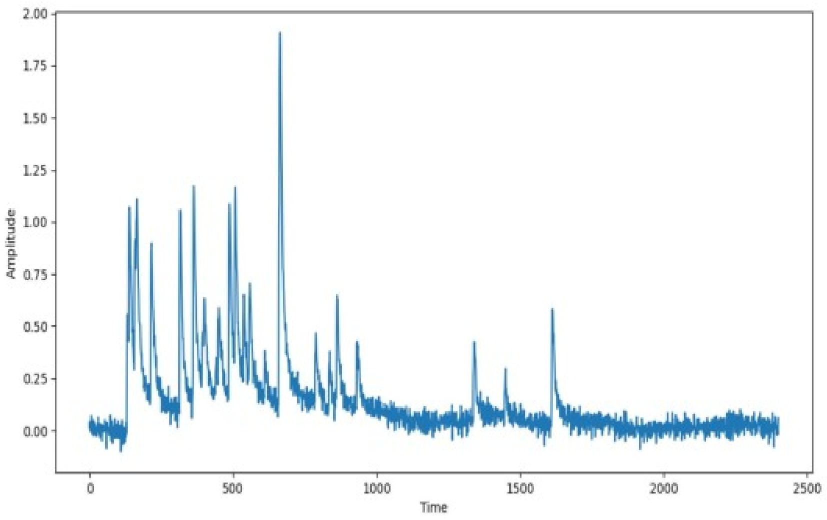
The above distribution shows the number of ionized electrons per cluster with mean value 1.528

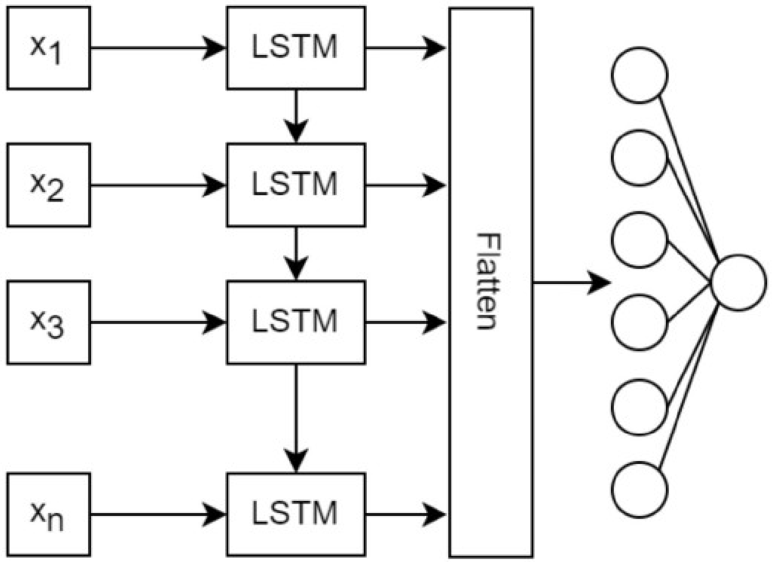
Trained LSTM and CNN Models for Two Step Reconstruction Algorithm Interactively

Two-Step Reconstruction Algorithm

Step1: Peak Finding

Waveform

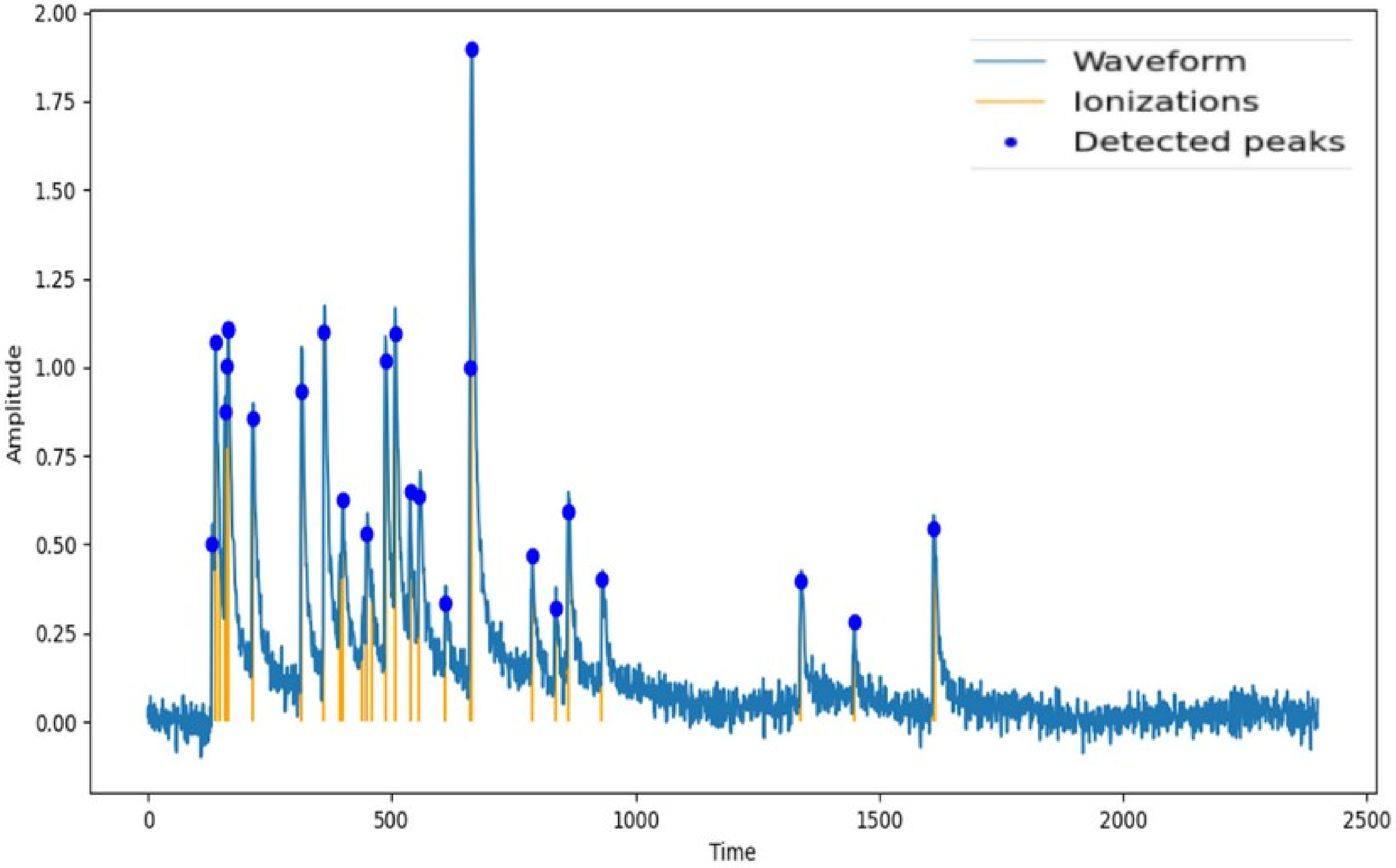




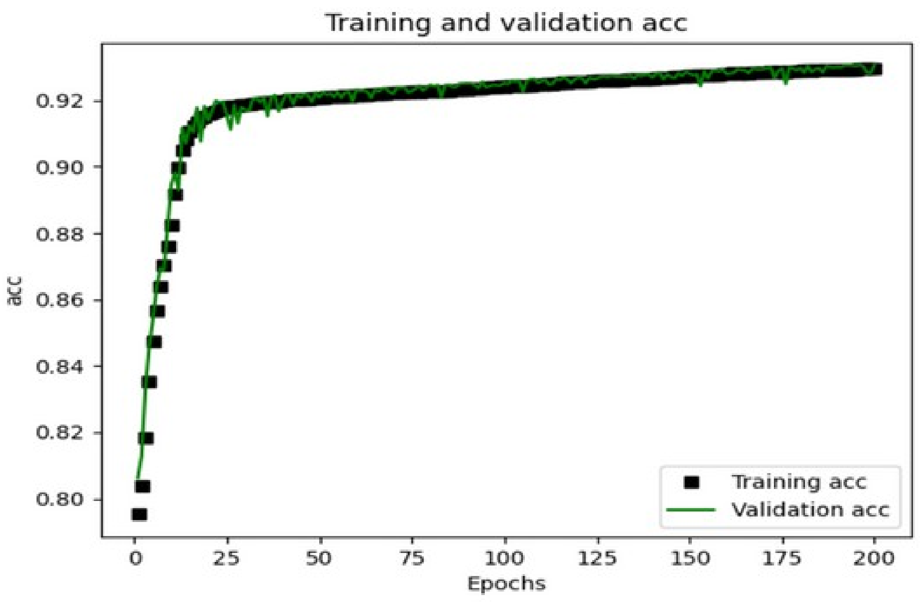
- **A classification problem to classify ionization signals (Primary and Secondary Ionizations) and noises in the waveform by using Long Short Term Memory (LSTM) model**

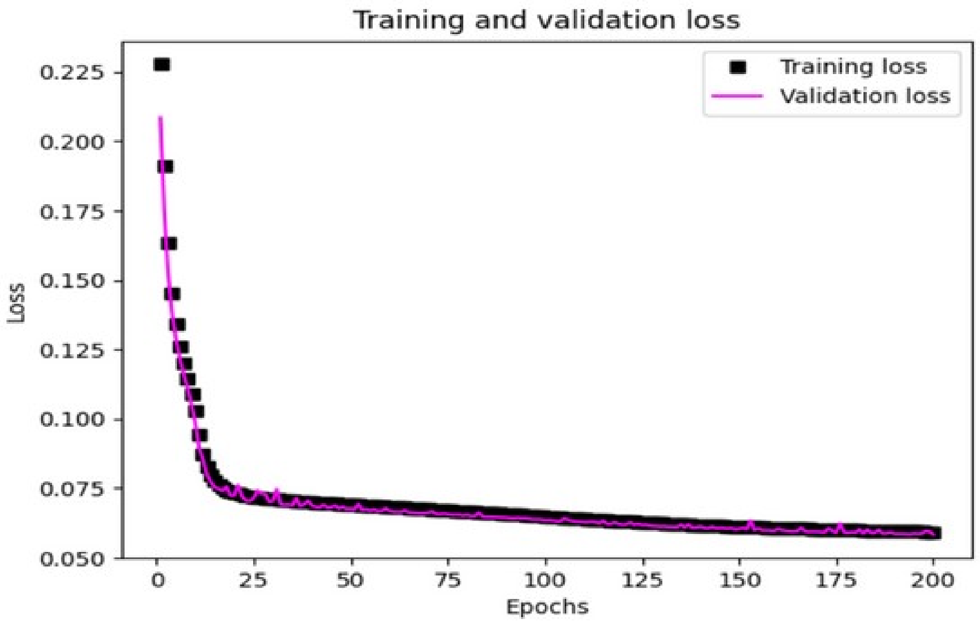
• Labels: Signal or Noise
• Features: Slide windows of peak candidates, with a shape of (15, 1)
• The data of waveform is time sequence data, which is suitable for Long short Term Memory (LSTM) model

Evaluation by Waveform



Performance of the LSTM Model

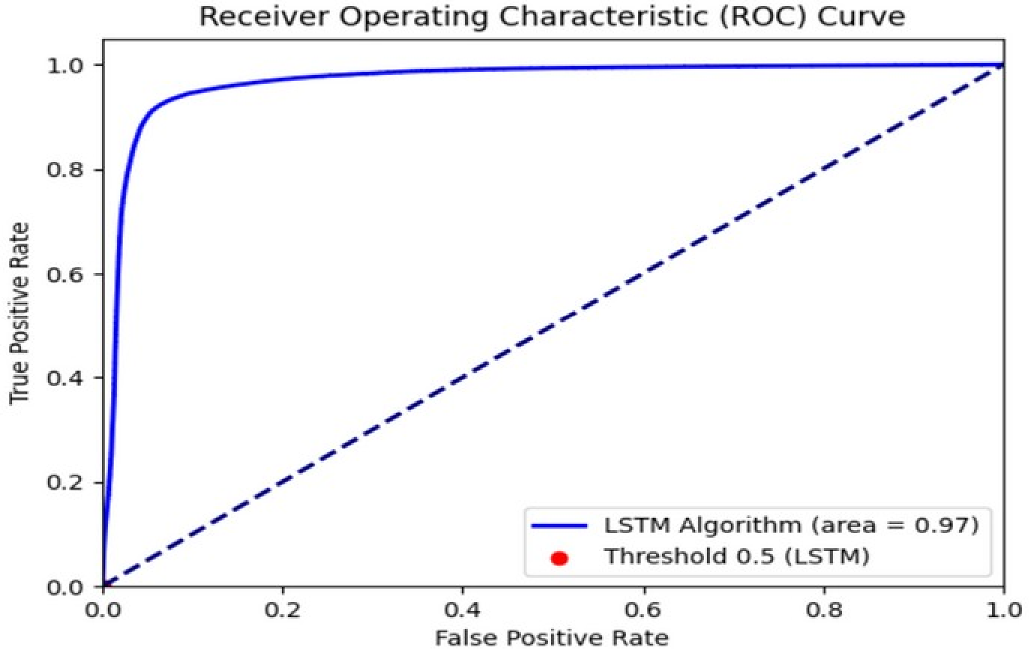




- **The above plot Accuracy VS Epoch show us that the training and validation accuracy increases over the epochs and then it become approximately constant which shows a best trained model**

- **The above plot loss VS epoch show us that the training and validation loss decreases over the epochs and then it become approximately constant which shows a best trained model**

Performance of the LSTM Model

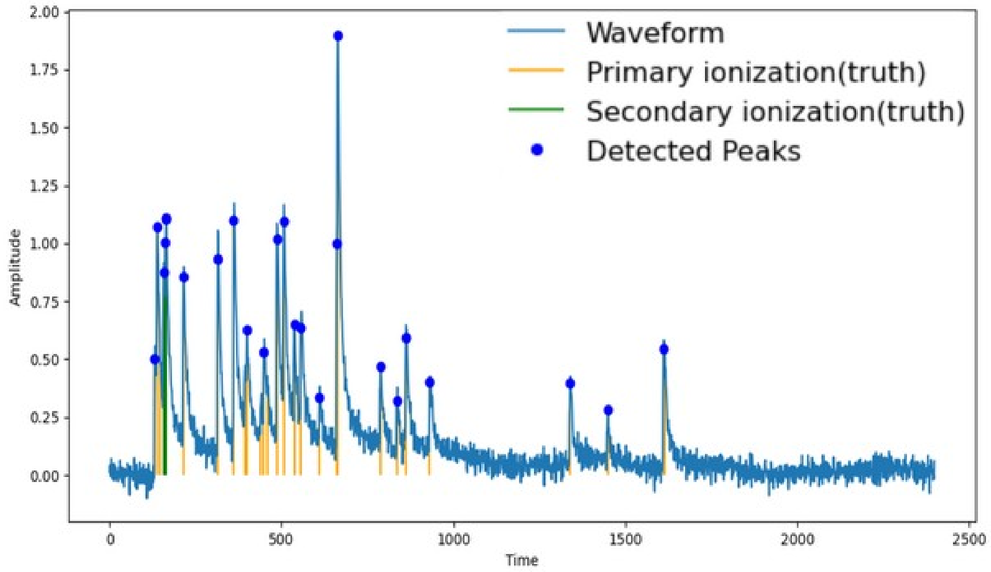


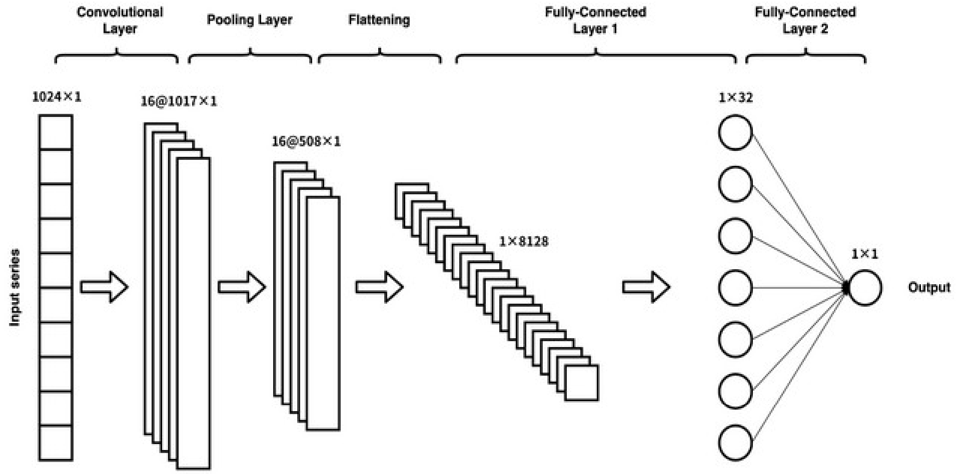
- The above plot show ROC curve for the LSTM model with Area under the curve value 0.97 with threshold value 0.5 which show a best classification to discrimate sugnal from background

TPR = TP/(TP+FN)
FPR = FP/(FP+TN)

		Prediction	
		Sig	Noise
Truth	Sig	TP	FN
	Noise	FP	TN

Step2: Clusterization





- A regression problem to predict Number of primary ionization clusters based on the primary detected peaks by using Convolutional Neural Network (CNN) model

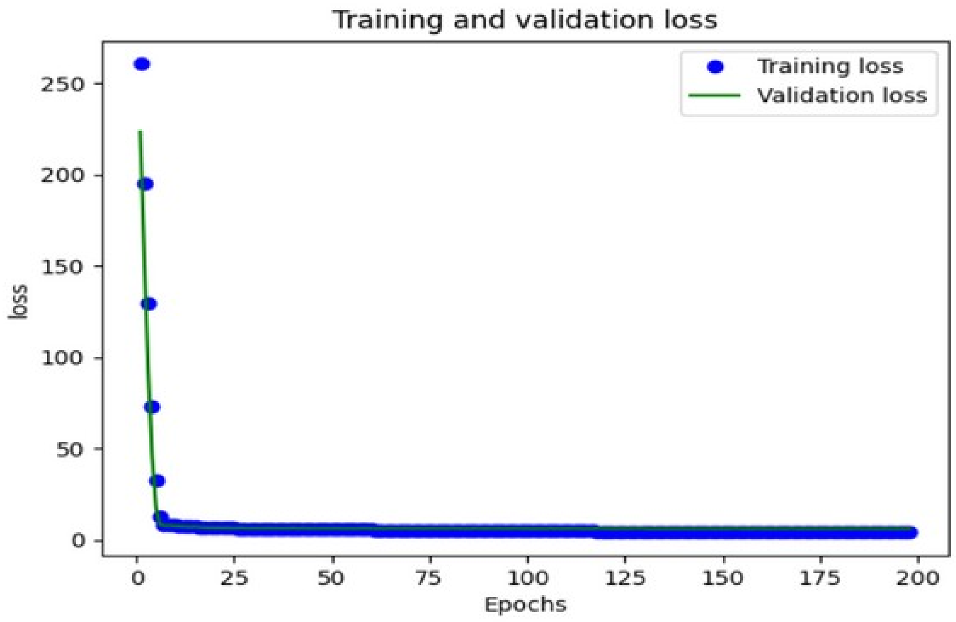
- The peaks found by peak finding algorithm would be training sample of this algorithm

- Labels: Number of clusters from MC truth

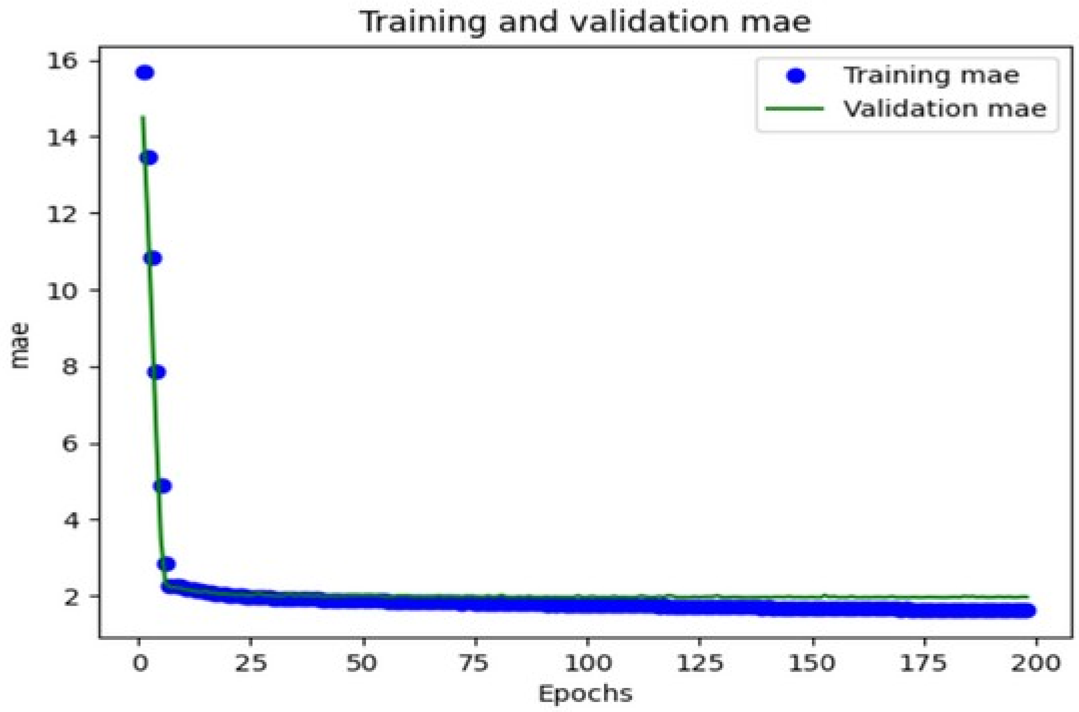
- Features: Time list of the detected times in the previous step encoding in an (1024, 1) array.

- A regression problem

Performance of the CNN Model

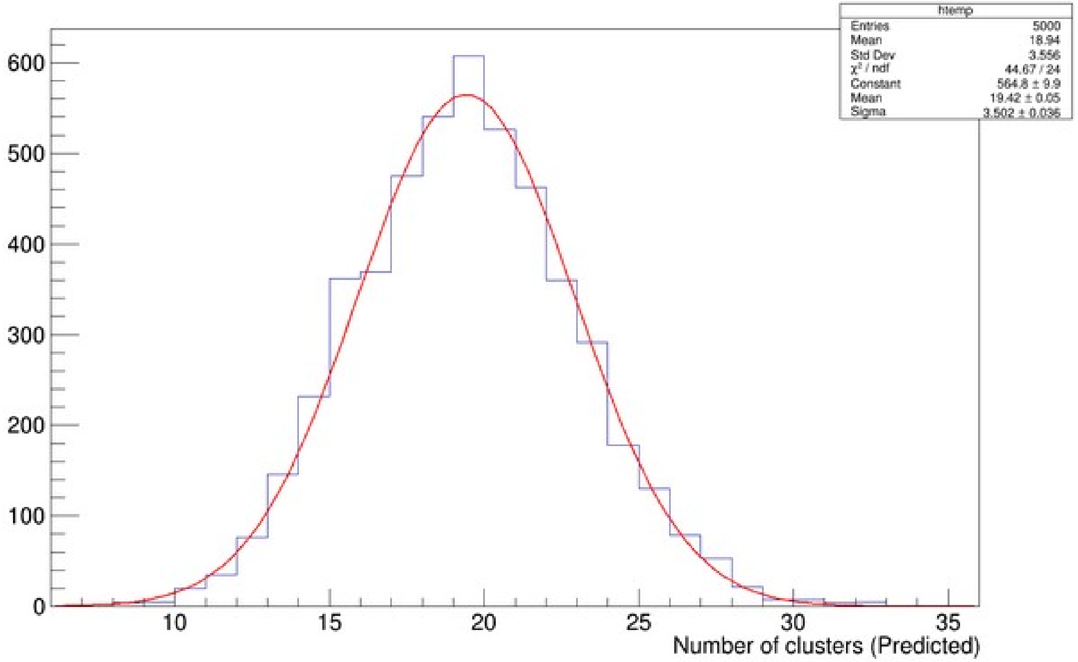


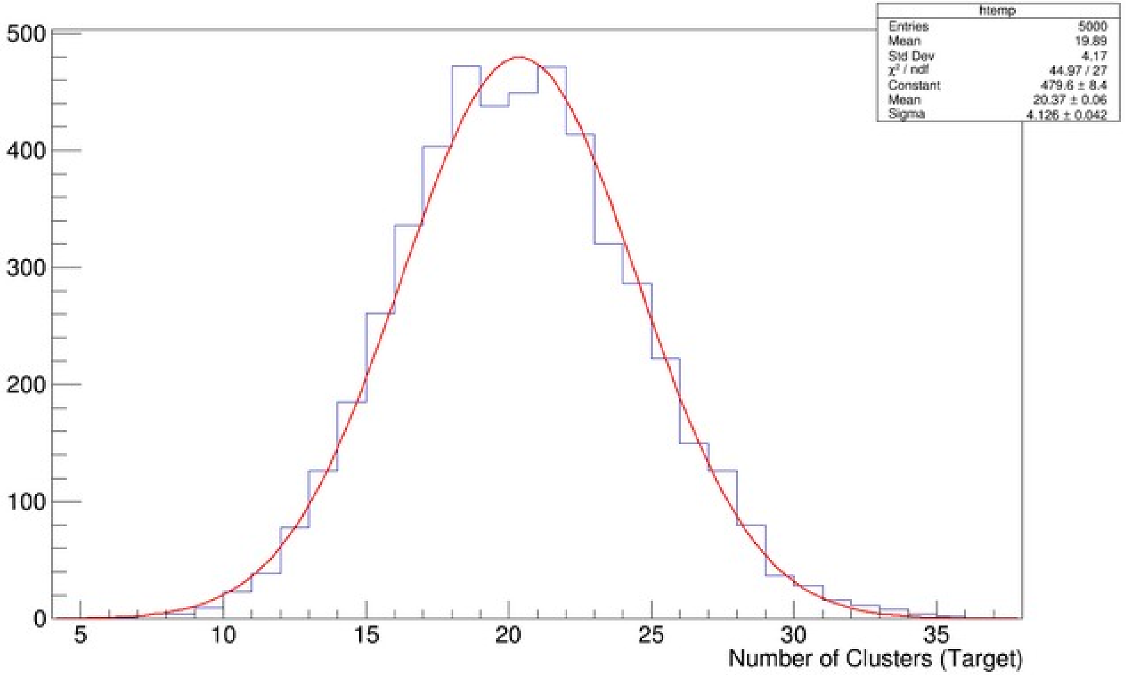
- The above plot loss (MSE) VS epoch show us that the training and validation loss decreases over the epochs and then it become constant shows us a best trained model



- The above mean absolute error VS epoch show us that the training and validation loss decreases over the epochs and then it become constant shows us a best trained model

Performance of the LSTM Model





- **Number of Primary ionized clusters with mean value (18.94) predicted by CNN Model based on the detected primary peaks with mean value (19.89)**
- **Good Gaussian distribution**

Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

- **Currently, I designed a task involving the simultaneous submission of several jobs using local HPC Resources**
- **The purpose of this task is to train Long Short-Term Memory (LSTM) models to classify signals from background, a process known as a classification task.**
- **To achieve this task, I utilized various hyperparameters, including activation functions, optimizer Epochs, batch size, patience, and dropout rates etc**
- **Additionally, I managed different resources such as memory requests, Job duration, and CPU Usage etc**
- **Then, I selected the best model based on different evaluation metrics such as the F1 score, Recall, Efficiency*Purity, and the highest AUC value among all configurations for training the LSTM model**

Training Structure of an LSTM Model Using Different Hyperparameters

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 96)	37632
flatten (Flatten)	(None, 96)	0
dense (Dense)	(None, 128)	12416
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129
dropout_1 (Dropout)	(None, 1)	0
dense_2 (Dense)	(None, 1)	2

Total params: 50,179
Trainable params: 50,179
Non-trainable params: 0

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 32)	4352
flatten (Flatten)	(None, 32)	0
dense (Dense)	(None, 64)	2112
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65
dropout_1 (Dropout)	(None, 1)	0
dense_2 (Dense)	(None, 1)	2

Total params: 6,531
Trainable params: 6,531
Non-trainable params: 0

- **The above screen shots tell us about the structure of LSTM Model with different number of neuron as well as parameters**

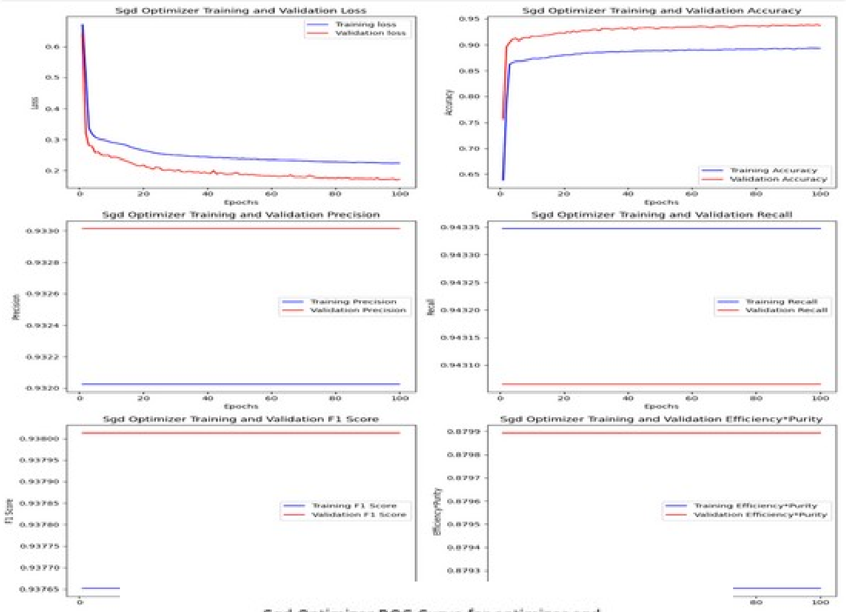
First Trial of Training an LSTM Model Using Various Hyperparameter Optimization Techniques

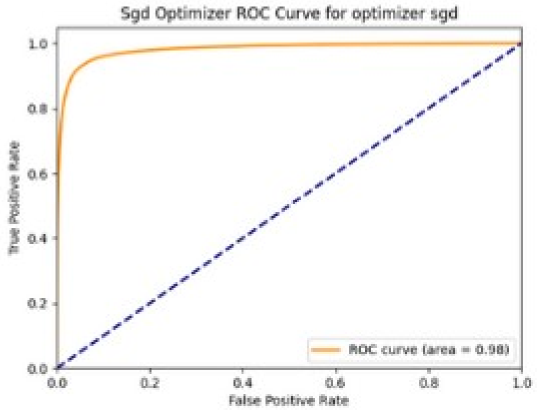
- **Different plots related to the Evaluation metrics such as Loss, Accuracy, F1 Score, Recall, Precision and Effieciency*Purity over the epochs are shown upper right side. This Evaluation metrics tell us about the best model by using different hyperparameters**

- **Mathematical formulae for the different evaluation metrics are:**

✳ **Precision = TP/TP + FP**
✳ **Recall = TP/TP + FN**
✳ **F1 score = 2 *Precision*recall/Precision + recall**
✳ **Accuracy = TP + TN/ TP + TN + FP + FN**

- **Area Under the Curve (AUC) value from Receiver Operating Characteristic (ROC) Curve is 0.98. Higher the value of AUC , best model would be consider fro the classification task**





Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

Optimizer	sgd
Topology	[96, 128, 1]
Bach size	[150]
Dropout Rate	[0.1, 0.1]
Number of Epochs	100
Activation function	Relu, Selu
Train/Validation Split	0.7
Patience	[50]

- The above table shows us different hyperparameters to train LSTM model for the classification task

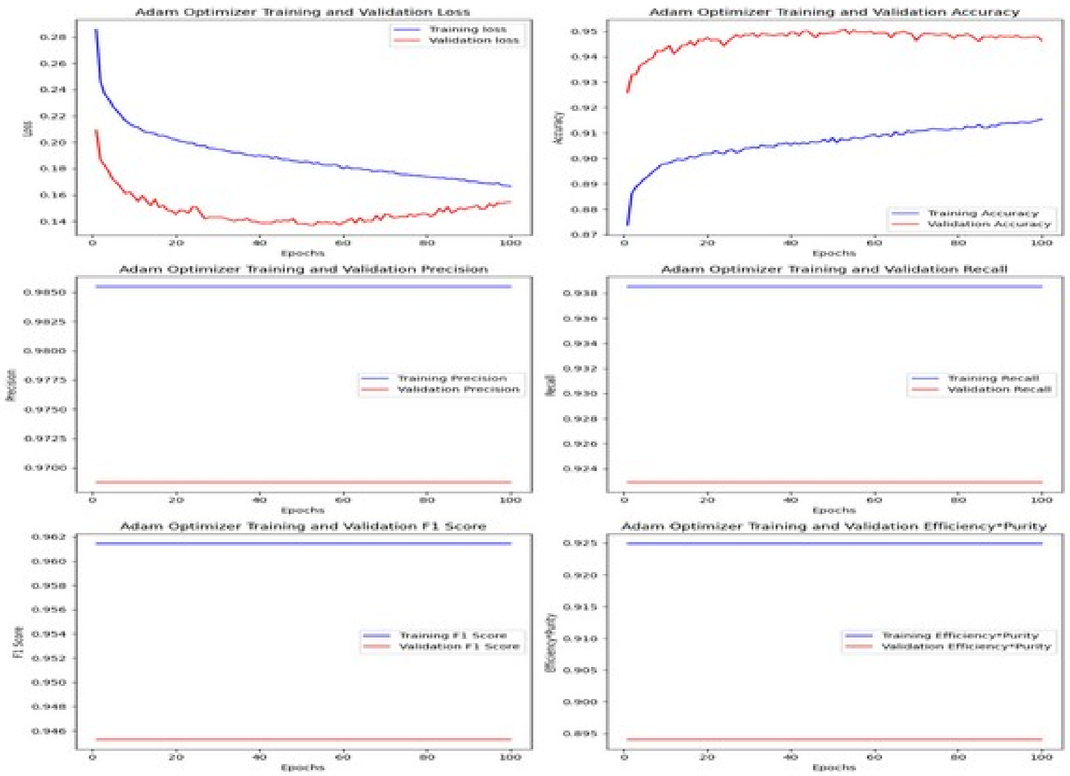
Precision	0.93301
Recall	0.94306
F1 Score	0.9380
AUC Score	0.98
Efficiency*Purity	0.87

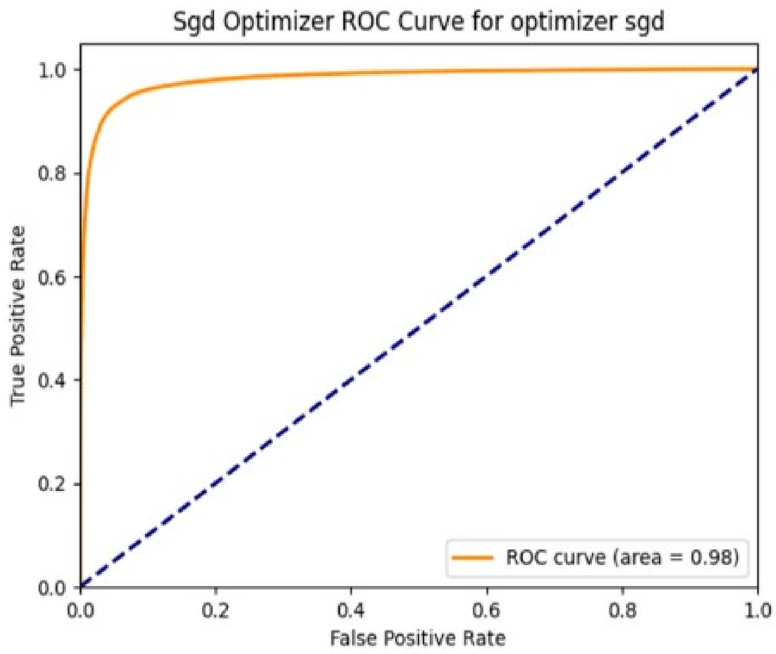
- The above table shows us different evaluation metrics to choose best LSTM model among all configurations

Partionable Resources	Usage	Request	Allocated
CPUS	1.92	4	4
Memory (MB)	925	5000	5120
Run Remote Usage	12 min 59 sec	2hr/job	--

- The above table shows us different HPC Local Resources of the RECAS like CPUS, Memory Usage and Run remote Usage

Second Trial of Training an LSTM Model Using Various Hyperparameter Optimization Techniques





- Different plots related to the evaluation metrics over the epochs are shown above. This Evaluation metrics tell us about the the performance of best model

- Area Under the Curve (AUC) value from Receiver Operating Characteristic (ROC) Curve is 0.98. Higher the value of AUC , best model would be consider fro the classification task

Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

Optimizer	adam
Topology	[96, 128, 1]
Bach size	[150]
Dropout Rate	[0.1, 0.1]
Number of Epochs	50
Activation function	Relu, Selu
Train/Validation Split	0.7
Patience	[50]

- **The above table shows us different hyperparameters to train LSTM model for the classification task**

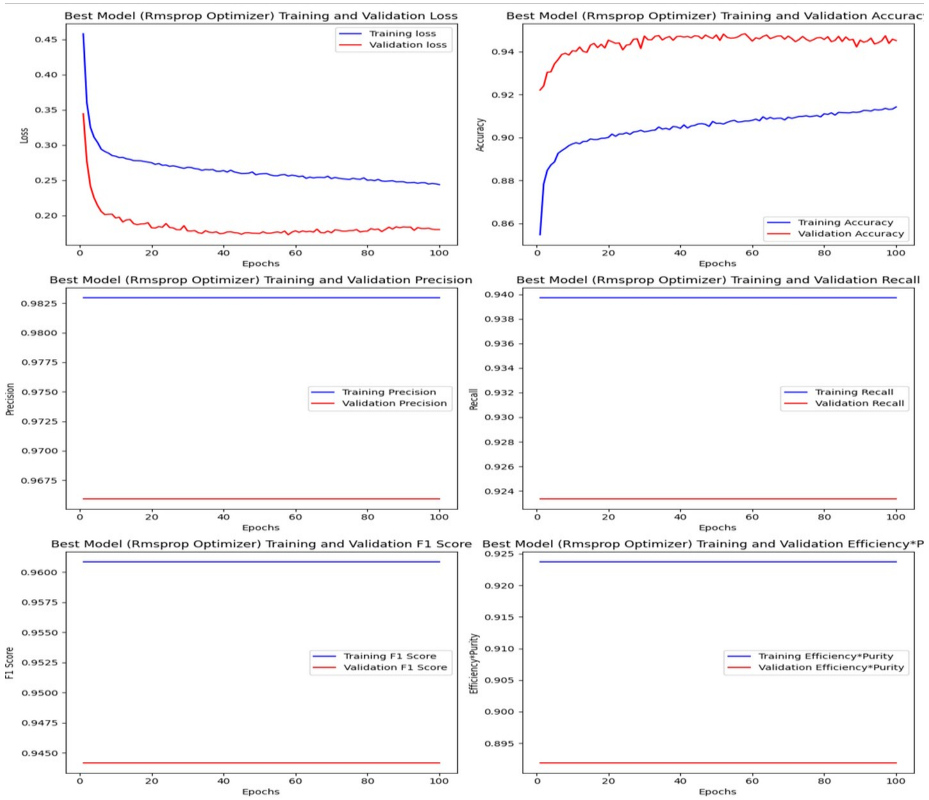
Precision	0.9687
Recall	0.9229
F1 Score	0.9380
AUC Score	0.98
Efficiency*Purity	0.89

- **The above table shows us different evaluation metrics to choose best LSTM model among all configurations**

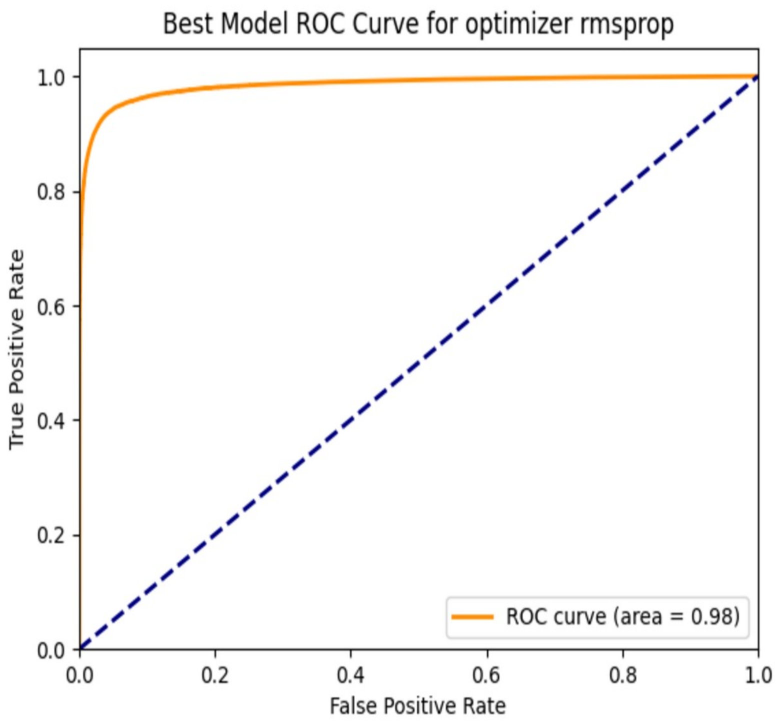
Partionable Resources	Usage	Request	Allocated
CPUS	1.76	4	4
Memory (MB)	830	5000	5120
Run Remote Usage	8 min 39 sec	2hr/job	--

- **The above table shows us different HPC Local Resources of the RECAS like CPUS, Memory Usage and Run remote Usage**

3rd Trial of Training an LSTM Model Using Various Hyperparameter Optimization Techniques



- Different plots related to the Evaluation metrics over the epochs are shown above. This evaluation metrics tell us about the the performance of best model



- Area Under the Curve (AUC) value from Receiver Operating Characteristic (ROC) Curve is 0.98. Higher the value of AUC , best model would be consider fro the classification task

Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

Optimizer	rmsprop
Topology	[96, 128, 1]
Bach size	[150]
Dropout Rate	[0.1, 0.1]
Number of Epochs	100
Activation function	selu, sigmoid
Train/Validation Split	0.7
Patience	[50]

- The above table shows us different hyperparameters to train LSTM model for the classification task

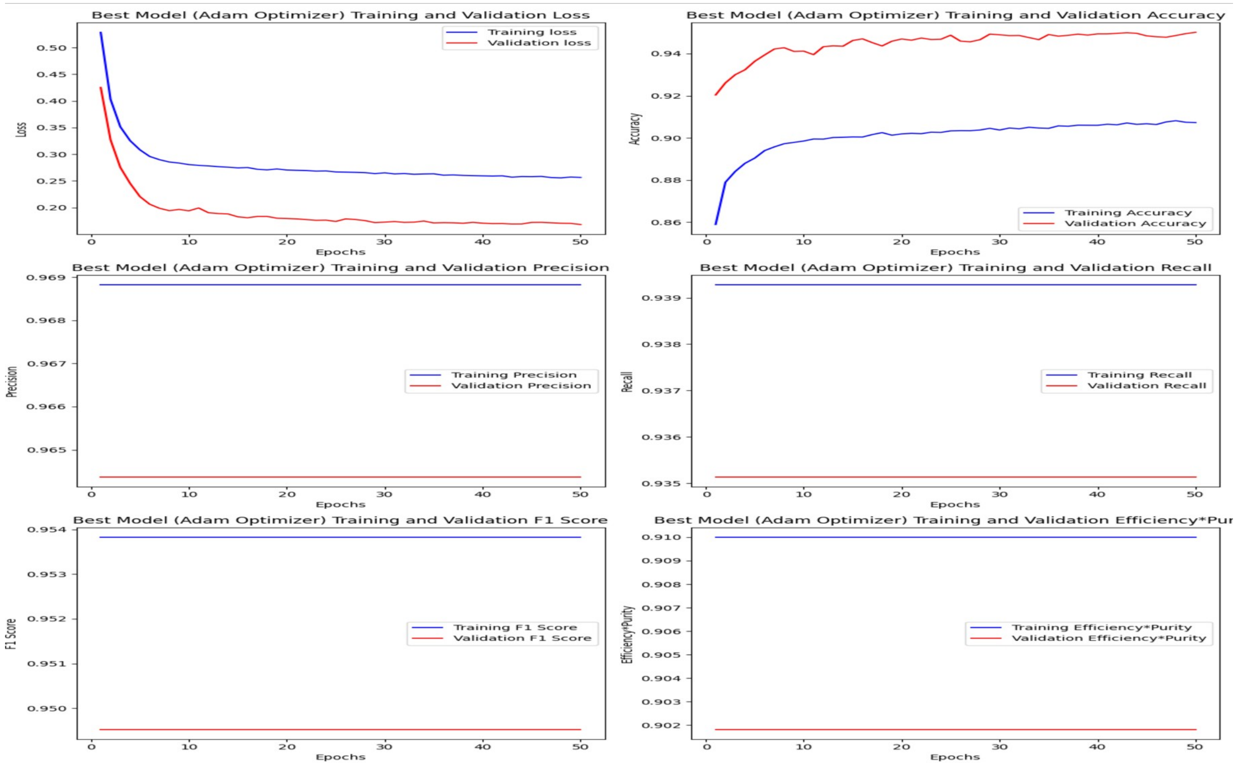
Precision	0.9659
Recall	0.9233
F1 Score	0.9441
AUC Score	0.98
Efficiency*Purity	0.89

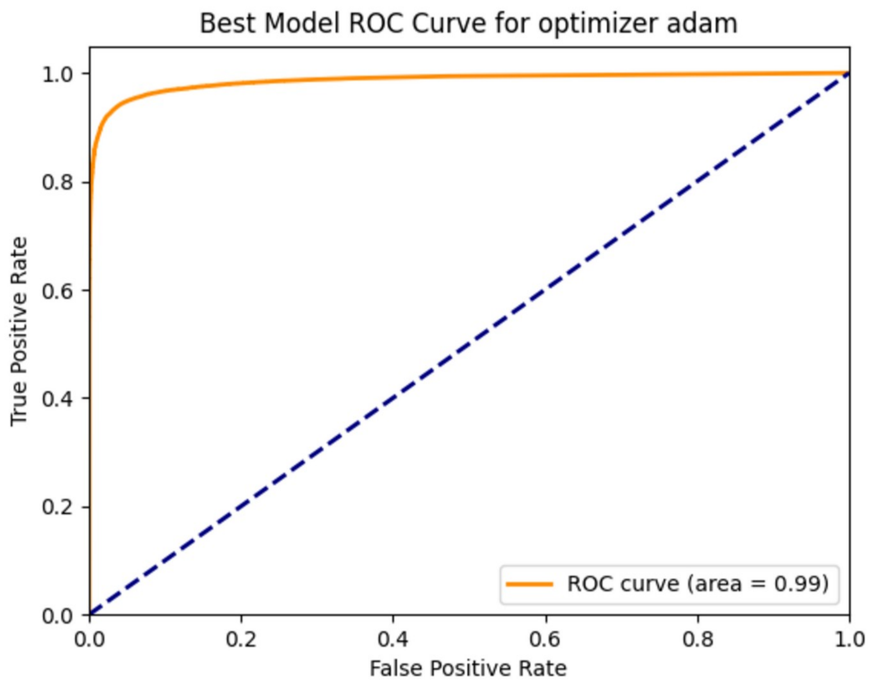
- The above table shows us different evaluation metrics to choose best LSTM model among all configurations

Partionable Resources	Usage	Request	Allocated
CPUS	1.87	4	4
Memory (MB)	791	5000	5120
Run Remote Usage	11 min 4sec	2hr/job	--

- The above table shows us different HPC Local Resources of the RECAS like CPUS, Memory Usage and Run remote Usage

Selected Best LSTM Model Based on the Higher Efficiency* Purity, F1 Score, Precision and Recall Value





- **Different plots related to the Evaluation metrics over the epochs are shown above. This evaluation metrics tell us about the the performance of best model**

- **Area Under the Curve (AUC) value from Receiver Operating Characteristic (ROC) Curve is 0.99. Higher the value of AUC , best model would be consider fro the classification task**

Optimization of Hyperparameters for Long Short-Term Memory Models Using HPC Resources

Optimizer	adam
Topology	[128 96 1]
Bach size	[150]
Dropout Rate	[0.1, 0.1]
Number of Epochs	50
Activation function	Relu, Sigmoid
Train/Validation Split	0.7
Patience	[50]

- The above table shows us different hyperparameters to train best LSTM model for the classification task

Precision	0.9643
Recall	0.9351
F1 Score	0.9495
AUC Score	0.99
Efficiency*Purity	0.90

- The above table shows us different evaluation metrics to choose best LSTM model among all configurations

Partionable Resources	Usage	Request	Allocated
CPUS	1.72	4	4
Memory (MB)	858	5000	5120
Run Remote Usage	9 min 34sec	2hr/job	--

- The above table shows us different HPC Local Resources of the RECAS like CPUS, Memory Usage and Run remote Usage

Summary

- Particle identification (PID) is essential in most particle experiments
- Cluster counting (CC) in gaseous detector is the most promising breakthrough in PID due to potential of 3 times better resolution than traditional method
- I executed the code pertaining to the simulation of particles traversing a gas mixture made out of 90% Helium (He) and 10% Isobutane (C4H10) filling drift tubes with the same geometry of the one used for the beam test at CERN in 2023
- Following the simulation in Garfield++, I proceeded to plot various results for the study of the cluster counting technique
- A two-step reconstruction algorithm involving peak finding (Discriminate signal from background in the waveform) and clusterization (Primary ionization clusters based on the detected peaks) was used in cluster counting techniques
- For the peak finding algorithm, I trained Long Short Term Memory (LSTM) model by using mean square error (MSE) as the loss function, sigmoid and rectified linear unit (ReLU) as activation functions, stochastic gradient descent (SGD) as the optimizer, with a batch size of 250 and 200 epochs interactively
- Concurrently, I trained Convolutional Neural Network (CNN) Model using mean absolute error (MAE) as a metrics, Root mean square propagation as the optimizer, with a batch size of 250 and 200 epochs to determine the number of primary clusters based on the detected peaks, dealing with a regression problem interactively

Summary

- **Additionally, I designed a task related to the simultaneous submission of several jobs using local HPC resources in order to train different models, such as long short-term memory models for classification tasks**

- **To achieve this, I utilized again various hyperparameters, including activation functions, optimizers, epochs, batch size, patience, and dropout rates etc**

Then, I selected the best model based on different evaluation metrics such as the precision (0.9643), F1 score (0.9495), Recall (0.9351), Efficiency*Purity (0.90), and the highest AUC (0.99) value among all configurations for training the LSTM model

Future Planning

- I would repeat the above task for all the possible hyperparameters and then select the best model based on F1 score, Recall. Efficeincy*Purity and highest AUC value among all different configurations for training the LSTM model

- Then I would apply LSTM Model to the simulation based on the garfield++ in order to classify signal from background in the waveform for the peak finding algorithm

- Similarly I would repeat the above logic to choose the best Convolutional Neural Networks (CNN) model in order to find out number of primary ionized cluster based on the detected primary peaks known as regression problem.

- Then, I will apply the trained models (LSTM & CNN) to the real beam test data to classify signals from noise in the waveform and determine the number of primary clusters based on the detected peaks


Thank you

Background Slides

Meaning of different Hyperparameters

- **SGD (Stochastic Gradient Descent): SGD is an optimization algorithm used to minimize a function by iteratively moving towards the minimum value in small steps.**
- **Batch Size: Batch size refers to the number of training samples used to train a model in one iteration.**
- **Dropout: Dropout is a regularization technique used to prevent overfitting in neural networks. It works by randomly setting a fraction of the input units to 0 at each update during training time, which helps to make the model robust by preventing it from relying too heavily on any one feature.**
- **Dropout Rate: The dropout rate is the fraction of the input units in a neural network that are set to zero during training.**
- **Epochs: An epoch is a single pass through the entire training dataset**
- **Activation Function: An activation function in a neural network is a mathematical operation that is applied to the input coming into a neuron. It decides whether a neuron should be activated or not, helping to add non-linearity to the model, which allows it to learn more complex patterns.**
- **Patience: Patience is a hyperparameter often used in conjunction with early stopping during training.**

Meaning of different Hyperparameters

- <u>Adam Optimizer:</u>
- **Adam stands for "Adaptive Moment Estimation." This optimizer combines two other optimizers—RMSprop and SGD with momentum. It keeps track of an exponentially decaying average of past gradients (similar to momentum) and an exponentially decaying average of past squared gradients (similar to RMSprop).**
- **Essentially, Adam adjusts the learning rate for each parameter, combining the benefits of having momentum (which helps to propel the optimizer towards the right direction and smooth out updates) and scaling the gradient by the square root of the recent average of squared gradients. This helps Adam adapt its learning rates based on the properties of data and make it effective and stable in practice..**
- <u>RMSprop Optimizer</u>:
- **RMSprop stands for "Root Mean Square Propagation." It was developed to resolve issues where the learning rates either diminish too quickly or too slowly. This optimizer adjusts the learning rate for each parameter by dividing the gradient by a running average of its recent magnitude.**
- **RMSprop makes adjustments to the learning rate during the training process. It does this by keeping track of the average of past squared gradients, and using this average to scale the gradient. This helps in a more balanced and faster convergence, as it prevents the learning rate from becoming too large or too small for certain weights in the network.**

Meaning of Evaluation metrics

- Precision: Precision is a metric that measures the accuracy of the positive predictions made by a model. In other words, it is the ratio of correctly predicted positive observations to the total predicted positives. It answers the question: "Of all the items labeled as positive, how many actually belong to the positive class?"

- Recall: Recall (also known as sensitivity) is the metric that measures the ability of a model to find all the relevant cases within a dataset. It is the ratio of correctly predicted positive observations to all observations in the actual positive class. It answers the question: "Of all the actual positives, how many were identified correctly?

- F1 Score: The F1 Score is the harmonic mean of precision and recall. It is a way to combine both precision and recall into a single measure that captures both properties. This score is particularly useful when you need to balance precision and recall and there is an uneven class distribution (large number of actual negatives).

- AUC Score (Area Under the Curve): The AUC score is used with the ROC curve (Receiver Operating Characteristic curve), which plots the true positive rate (recall) against the false positive rate at various threshold settings. The AUC score represents the degree or measure of separability achieved by the model. It tells how much the model is capable of distinguishing between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s.

Meaning of Evaluation metrics

- **Efficiency*Purity: This is a product of two metrics, Efficiency and Purity, which is often used in fields like particle physics but can also be applied in other areas of classification.**

- **Efficiency: It is similar to recall. It measures the proportion of actual positives that are correctly identified.**

- **Purity: This metric is similar to precision. It measures the proportion of positive identifications that were actually correct.**

- **When combined (Efficiency*Purity), this product provides a single measure that reflects how many of the selections were correct (purity) and how many of the correct cases were selected (efficiency). This can be particularly useful in scenarios where it's crucial not only to identify positive cases accurately but also to cover as many of them as possible without introducing too many errors.**

- **The acronym "CPU" stands for Central Processing Unit. It is the primary component of a computer that performs most of the processing inside. A CPU executes instructions from a computer program by performing the basic arithmetic, logical, control, and input/output (I/O) operations specified by the instructions.**

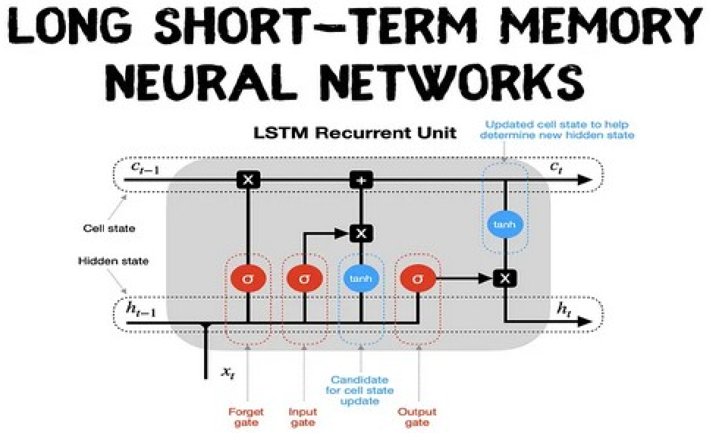
- **When you mention using CPU from RECAS resources, it suggests that you are utilizing CPU computing power provided by the RECAS (REsources for Cloud federated Access Services) project or a similar computing resource. RECAS typically offers infrastructure and computing resources to support scientific research, where CPUs are used to process tasks, run simulations, or analyze data. These resources are often shared or distributed across a network, allowing users to leverage powerful computational capabilities remotely.**

ACCURACY and LSTM

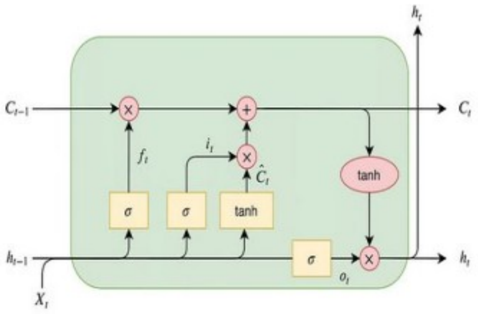
- The accuracy is defined as the ratio between the number of correct predictions to the total number of predictions
- Accuracy values range between 0 and 1. Obviously an accuracy values near to 1 means that our model fits well the datasets

$$Accuracy = \frac{True_{positive} + True_{negative}}{True_{positive} + True_{negative} + False_{positive} + False_{negative}}$$

- Forget Gate: This gate determines what information from the previous cell state should be forgotten or retained.
- Input Gate: It controls what new information should be stored in the cell state.
- Output Gate: This gate defines the output of the LSTM cell, considering the current input and the updated cell state



Long SHort Term Memory (LSTM)



$$f_t = \sigma(W_f \cdot [h_{t-1}, X_t] + b_f)$$

[redacted]

$$i_t = \sigma(W_i \cdot [h_{t-1}, X_t] + b_i)$$

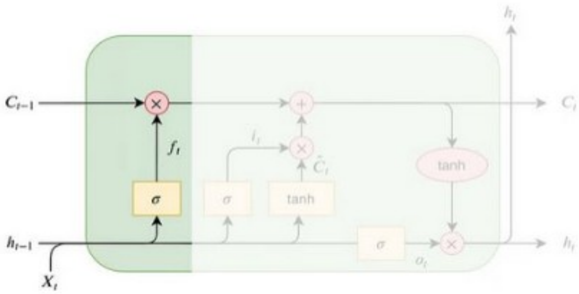
$$o_t = \sigma(W_o \cdot [h_{t-1}, X_t] + b_o)$$

$$\hat{C}_t = \tanh(W_C \cdot [h_{t-1}, X_t] + b_C)$$

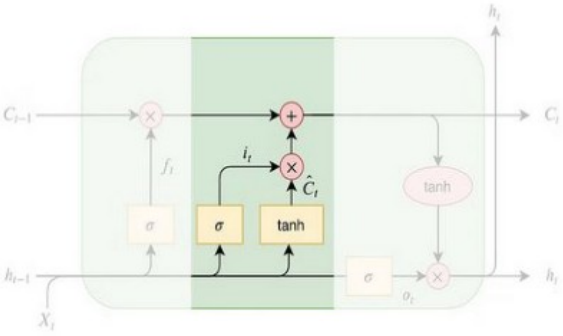
$$C_t = i_t \cdot \hat{C}_t + f_t \cdot C_{t-1}$$

Forget Gate

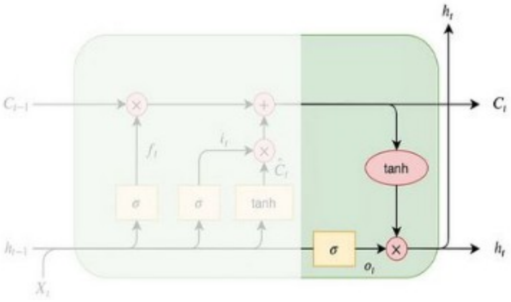
state and the new input data.



Input Gate



Output Gate



EXAMPLES of LOSS FUNCTIONS

- Mean Squared Error(MSE)/ Quadratic Loss/ L2:

$$MSE(y^{(i)}, y^{(i)}_{pred}) = \frac{\left(y^{(i)} - y^{(i)}_{pred}\right)^2}{n}$$

- Mean Absolute Error (MAE)/ L1 Loss:

$$MAE(y^{(i)}, y^{(i)}_{pred}) = \frac{\left|y^{(i)} - y^{(i)}_{pred}\right|}{n}$$

- Mean Bias Error (MBE):

$$MBE(y^{(i)}, y^{(i)}_{pred}) = \frac{\left(y^{(i)} - y^{(i)}_{pred}\right)}{n}$$

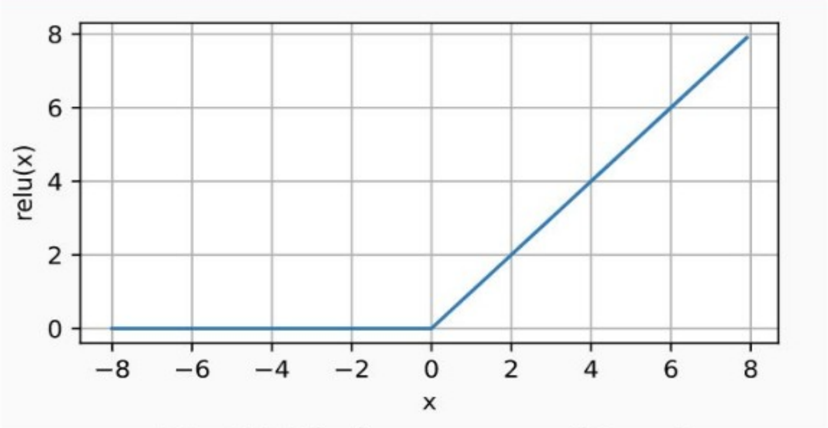
NUMBER OF EPOCHS

- **Epoch**: In terms of artificial neural networks, an epoch refers to one cycle through the full training dataset
- Number of epochs is a delicate choice:
  - ❑ A large number of epochs can induce our model to an overfitting problem
  - ❑ Too small number of epochs can lead to an under fitting problem
- To avoid a wrong choice we can use the ' EarlyStopping', also implemented by Keras:
  - ❑ It allows to stop the training when a monitor (set by us and tipically the loss function) has stopped improving.

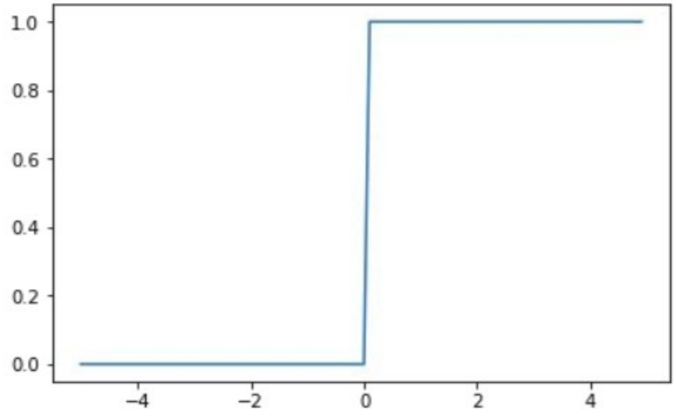
RECTIFIED LINEAR UNIT (RELU)

- One the most popular non-linear activation function is the REctified Linear Unit (ReLU)
- It provides a non-linear transformation and returns the max value between the input x (the argument) and 0
- The ReLU function is also differentiable in as given below:

$$\frac{dReLU(x)}{dx} = \begin{cases} 0 & x \le 0 \\ 1 & x > 0 \end{cases}$$



$$ReLU(x) = max(0, x)$$



SCALED EXPONENTIAL LINEAR UNIT (SELU)

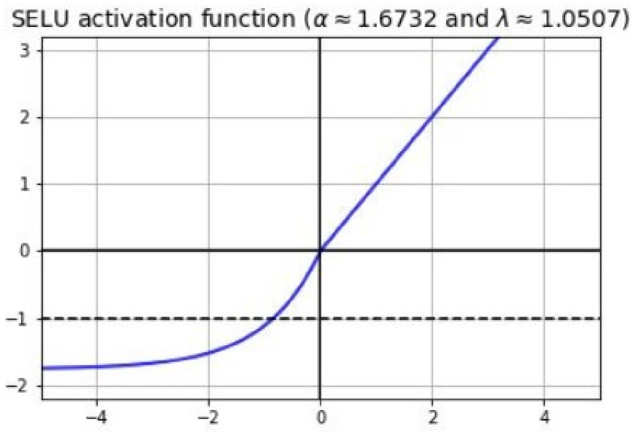
- Another choice is the Scaled Exponential Linear Unit (SELU)

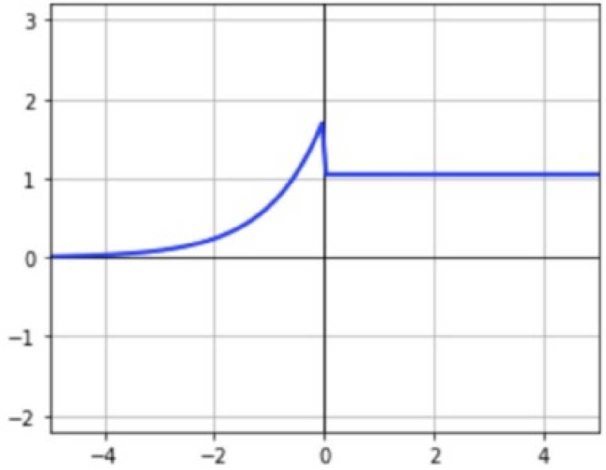
- The functions depends on two parameters and the equation is the following:

$$SELU(x) = \lambda \begin{cases} \alpha(e^x - 1) & x \leq 0 \\ x & x > 0 \end{cases}$$

- The function is not differentiable in zero

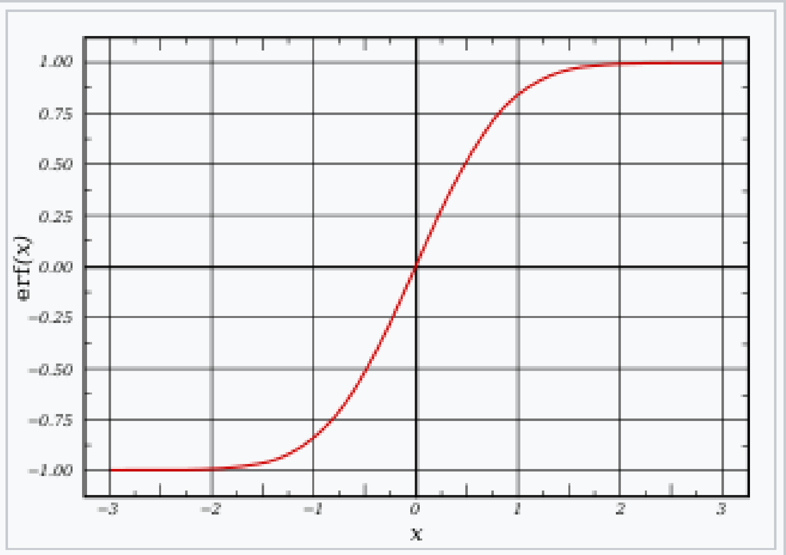
$$\frac{dSELU(x)}{dx} = \lambda \begin{cases} \alpha e^x & x \leq 0 \\ 1 & x > 0 \end{cases}$$



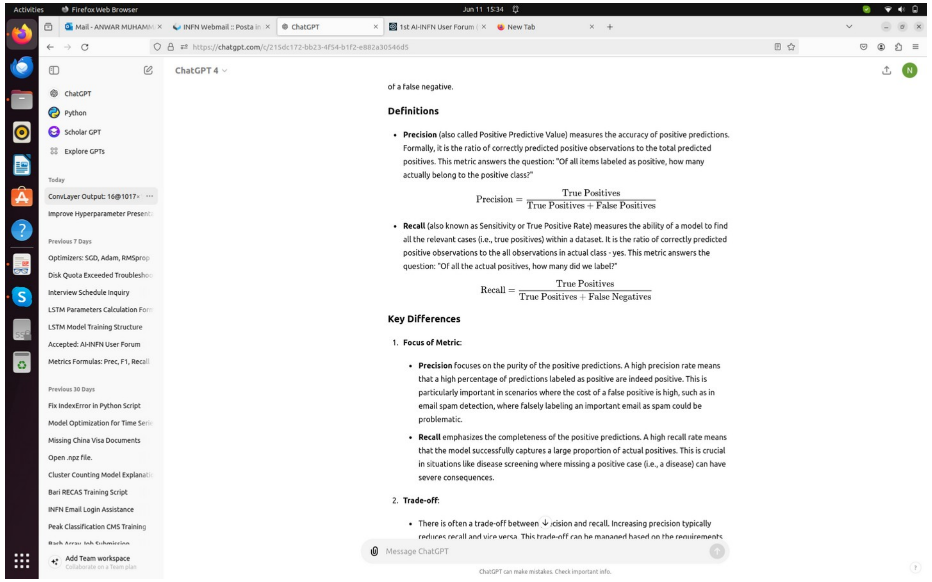


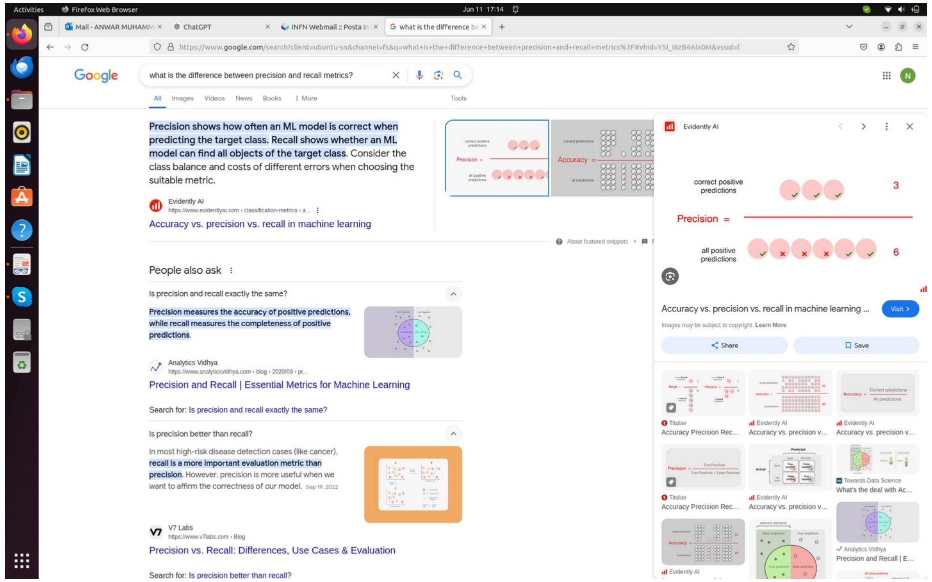
Sigmoid Function

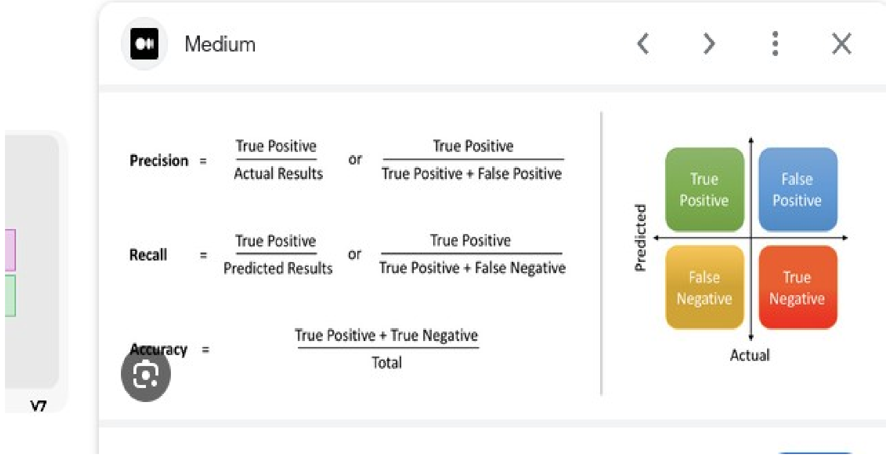
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



- **The sigmoid function outputs a value between 0 and 1, making it especially useful for models where you need to predict probabilities that vary between these two limits. The function is S-shaped, providing a smooth gradient as xxx increases or decreases. This characteristic is particularly beneficial during the optimization phase of training a model, as it provides a clear path toward minimizing the loss**.


ChatGPT 4
of a false negative.
Definitions
Precision (also called Positive Predictive Value) measures the accuracy of positive predictions. Formally, it is the ratio of correctly predicted positive observations to the total predicted positives. This metric answers the question: "Of all items labeled as positive, how many actually belong to the positive class?"
Precision = True Positives / True Positives + False Positives
Recall (also known as Sensitivity or True Positive Rate) measures the ability of a model to find all the relevant cases (i.e., true positives) within a dataset. It is the ratio of correctly predicted positive observations to all observations in actual class - yes. This metric answers the question: "Of all the actual positives, how many did we label?"
Recall = True Positives / True Positives + False Negatives
Key Differences
1. Focus of Metric:
Precision focuses on the purity of the positive predictions. A high precision rate means that a high percentage of predictions labeled as positive are indeed positive. This is particularly important in scenarios where the cost of a false positive is high, such as in email spam detection, where falsely labeling an important email as spam could be problematic.
Recall emphasizes the completeness of the positive predictions. A high recall rate means that the model successfully captures a large proportion of actual positives. This is crucial in situations like disease screening where missing a positive case (i.e., a disease) can have severe consequences.
2. Trade-off:
Message ChatGPT
ChatGPT can make mistakes. Check important info.


what is the difference between precision and recall metrics?
Precision shows how often an ML model is correct when predicting the target class. Recall shows whether an ML model can find all objects of the target class. Consider the class balance and costs of different errors when choosing the suitable metric.
Accuracy vs. precision vs. recall in machine learning
People also ask
Is precision and recall exactly the same?
Precision measures the accuracy of positive predictions, while recall measures the completeness of positive predictions.
Precision and Recall | Essential Metrics for Machine Learning
Is precision better than recall?
In most high-risk disease detection cases (like cancer), recall is a more important evaluation metric than precision. However, precision is more useful when we want to affirm the correctness of our model.
Precision vs. Recall: Differences, Use Cases & Evaluation
Evidently AI
correct positive predictions
Precision =
all positive predictions
Accuracy vs. precision vs. recall in machine learning ...


Medium
Precision = True Positive / Actual Results or True Positive / True Positive + False Positive
Recall = True Positive / Predicted Results or True Positive / True Positive + False Negative
Accuracy = True Positive + True Negative / Total
Predicted
True Positive
False Positive
False Negative
True Negative
Actual