Preliminary report:

Auto Picking using Deep Learning

Prepared by:

Arham Zakki Edelo

Outline:

Overview of Phase Picking
Deep Learning in General
Deep Learning for Phase Picking
Methodology
Data Processing Prior to Training
Evaluation of Trained Model Performance
Results of Predictive Model Implementation
Comparison with Manual Picking
Stochastic Accuracy Improvement
A One Day Automatic Monitoring Test
Discussion, Model Limitations and Future Improvements
References

Overview of Phase Picking

To satisfy this fundamental earthquake location equation (Havskov dan Ottemoller, 2010):

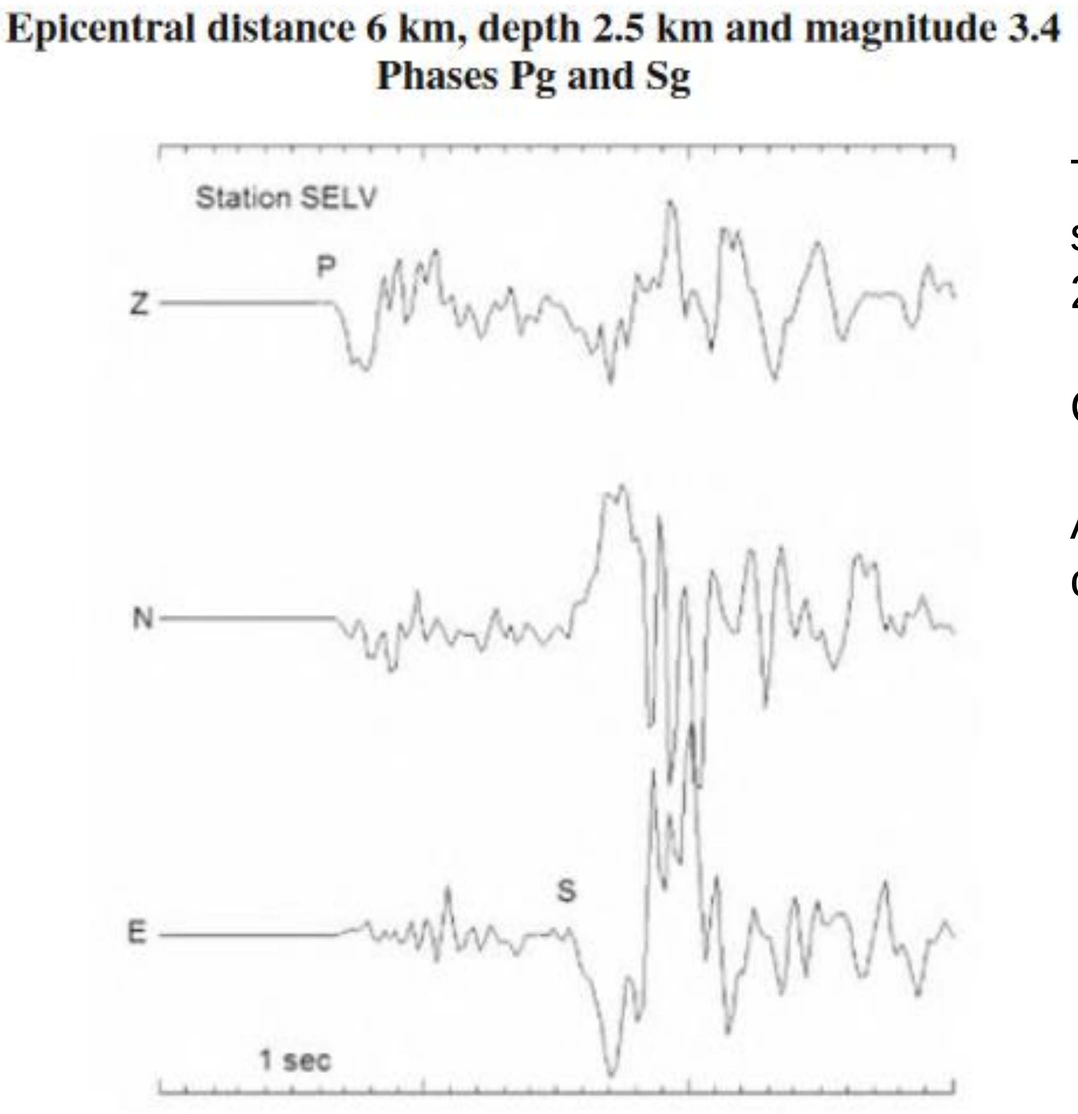
$$t_i^{arr} = \frac{\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}}{v} + t_0$$

$$r_i = t_i^{obs} - t_i^{arr}$$

We need a precise definition of t_i^{obs} (observed arrival time) at every station i.

This only can be achieved through wavefrom analysis, which involves determining the onset of P-phase and S-phase to provide robust constraints for the hypocenter solution.

Overview of Phase Picking



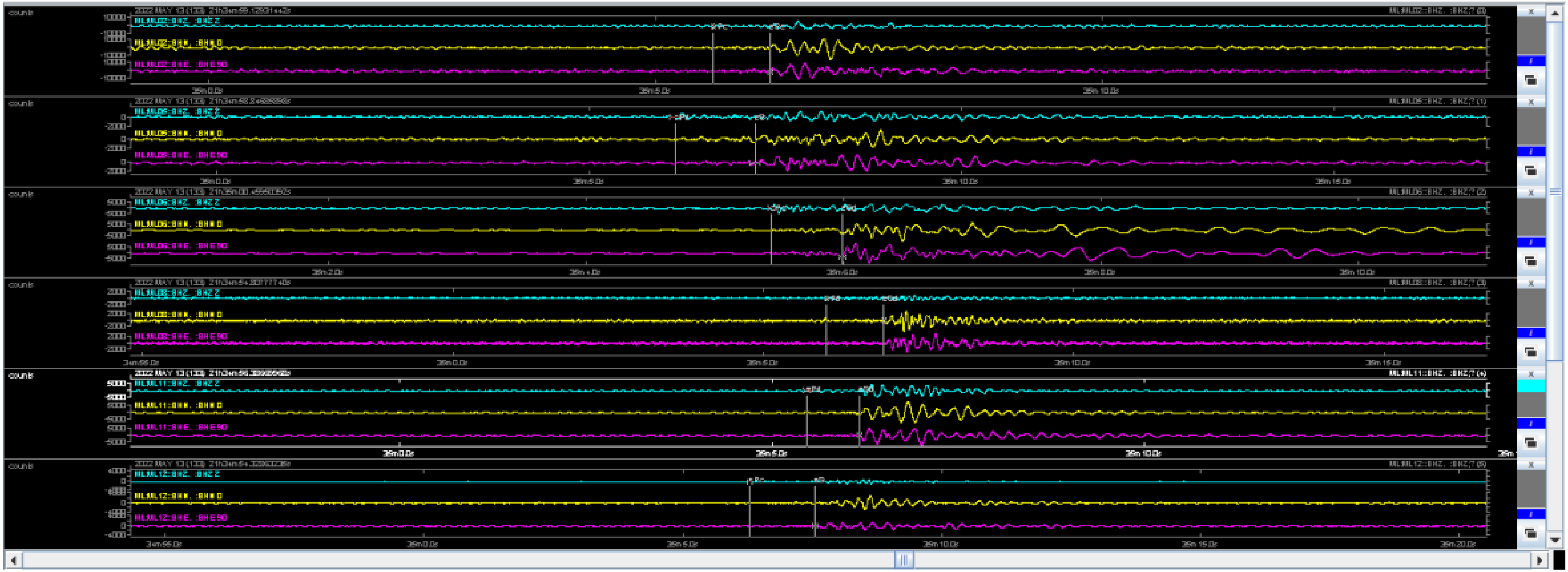
Source : Havskov and Ottemoller, 2010

Typical of local earthquake waveform recorded on seismogram 3 components (Havskov dan Ottemoller, 2010).

Clear onset of P-phase on vertical or Z component.

And clear onset of S-phase on horizontal or N/E component.

Overview of Phase Picking

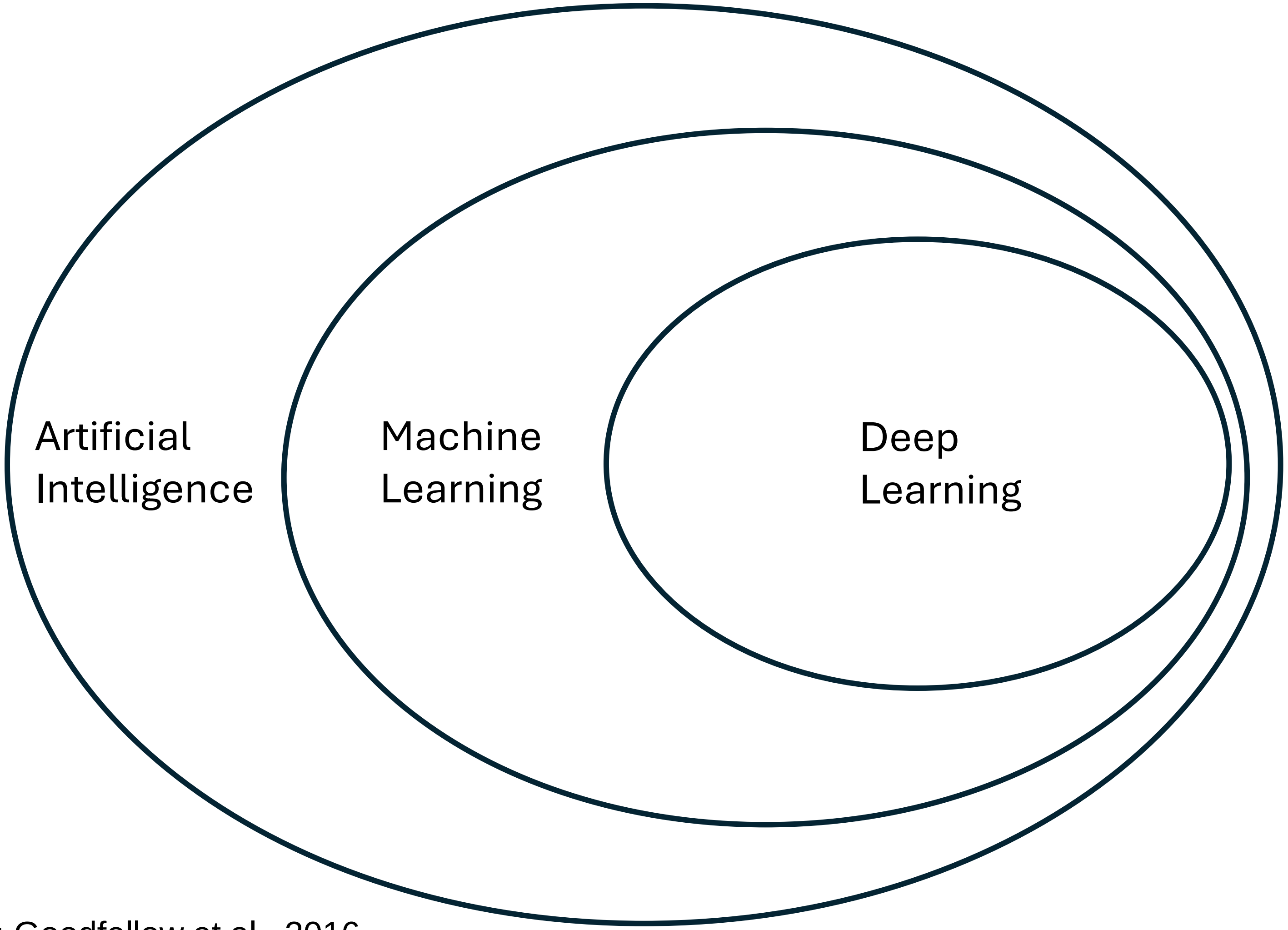


Phase picking on multiple stations with actual MEQ monitoring data.

More stations & more phases ➔ Better hypocenter solution.

Deep Learning in General

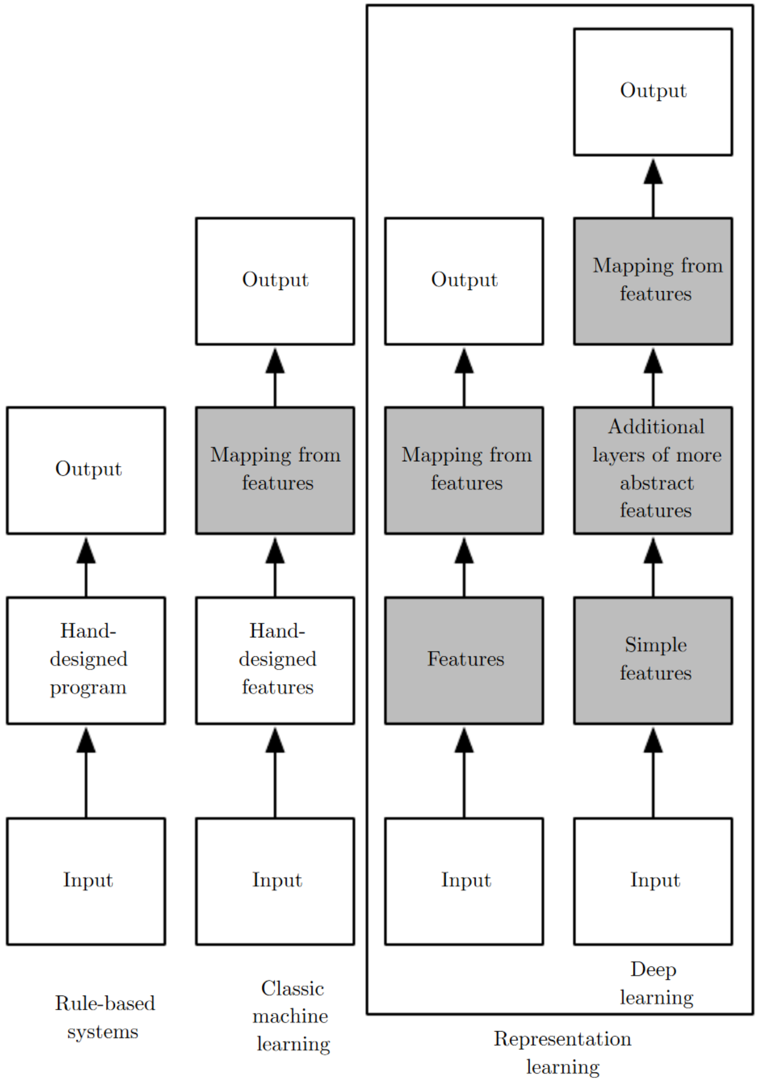
Deep learning is particular kind of machine learning that achieves great power and flexibility by learning to represent the **world nested hierarchy of concepts**.



Source : Goodfellow et al., 2016

Deep Learning in General

How the different parts of an AI system relate to each other within different AI discipline.



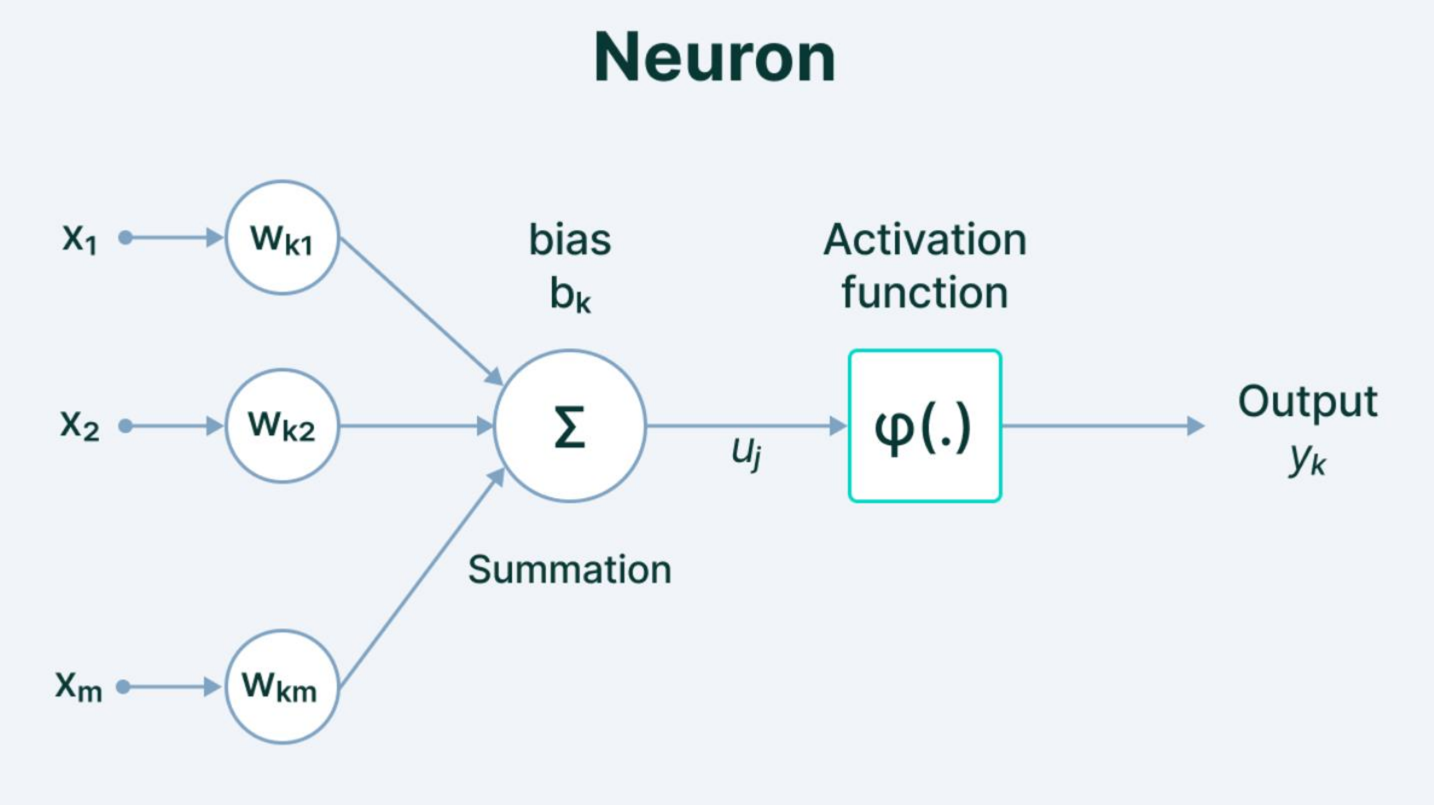
Source : Goodfellow et al., 2016

Deep Learning = Machine Learning + Representation Learning

Deep learning is a set of method that allows machine to perform automatic discovery of powerful features from raw data (Deng dan Yu, 2014).

Deep Learning in General

The architecture of neural networks usually consist of few essential parts:

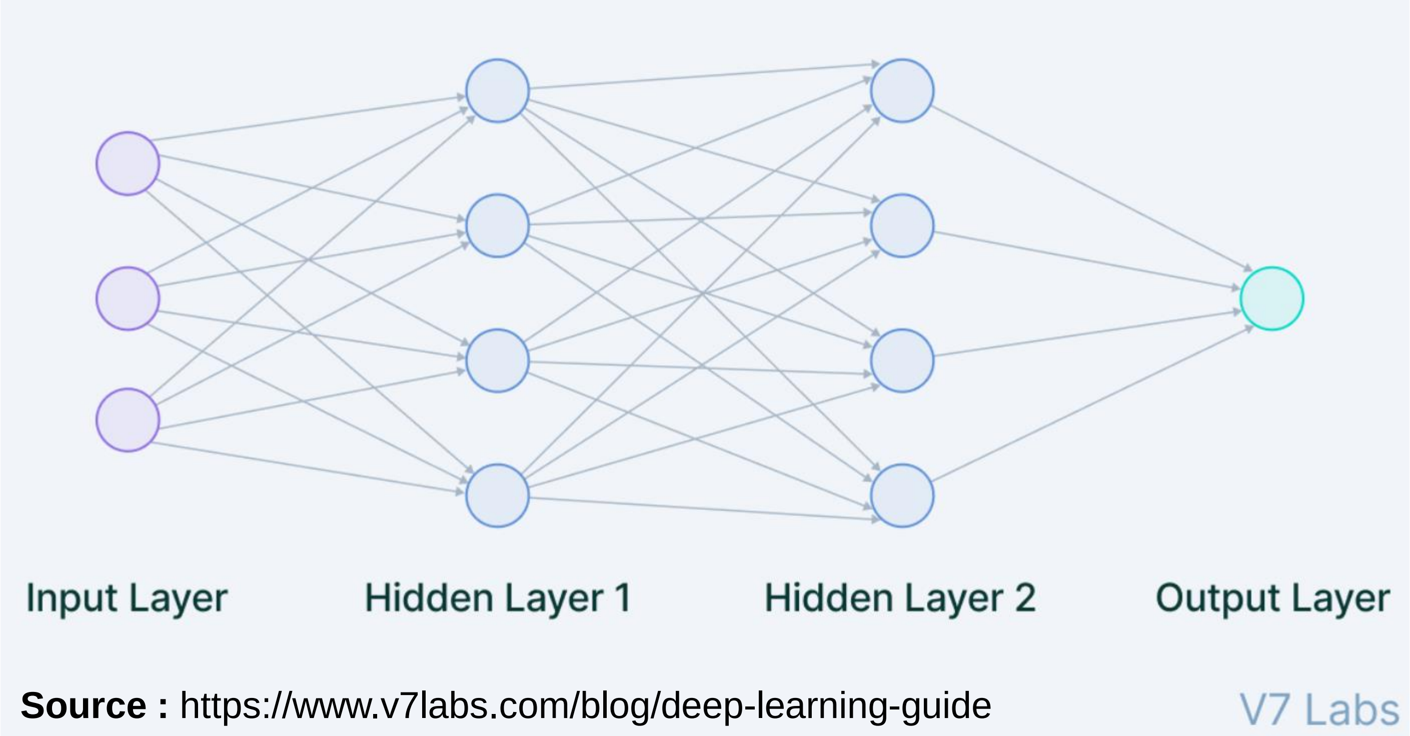


Source : https://www.v7labs.com/blog/deep-learning-guide

- **Input :** set of features that are fed into the model for learning process.
- **Weight :** Its main function is to give importance to those features that contribute more towards the learning.
- **Transfer function :** The job of the transfer function is to combine multiple inputs into one output value so that the activation function can be applied. It is done by a simple summation of all the inputs to the transfer function.
- **Activation function :** It introduces non-linearity in the working of perceptrons. Without this function the output would just be a linear combination of input values and would not be able to introduce non-linearity in the network.
- **Bias :** It shifts the value produced by the activation function. The roles is similar to ther role of a constant in a linear function .

Deep Learning in General

And when there are multiple layers stacked together it is called multi-layer neural network.



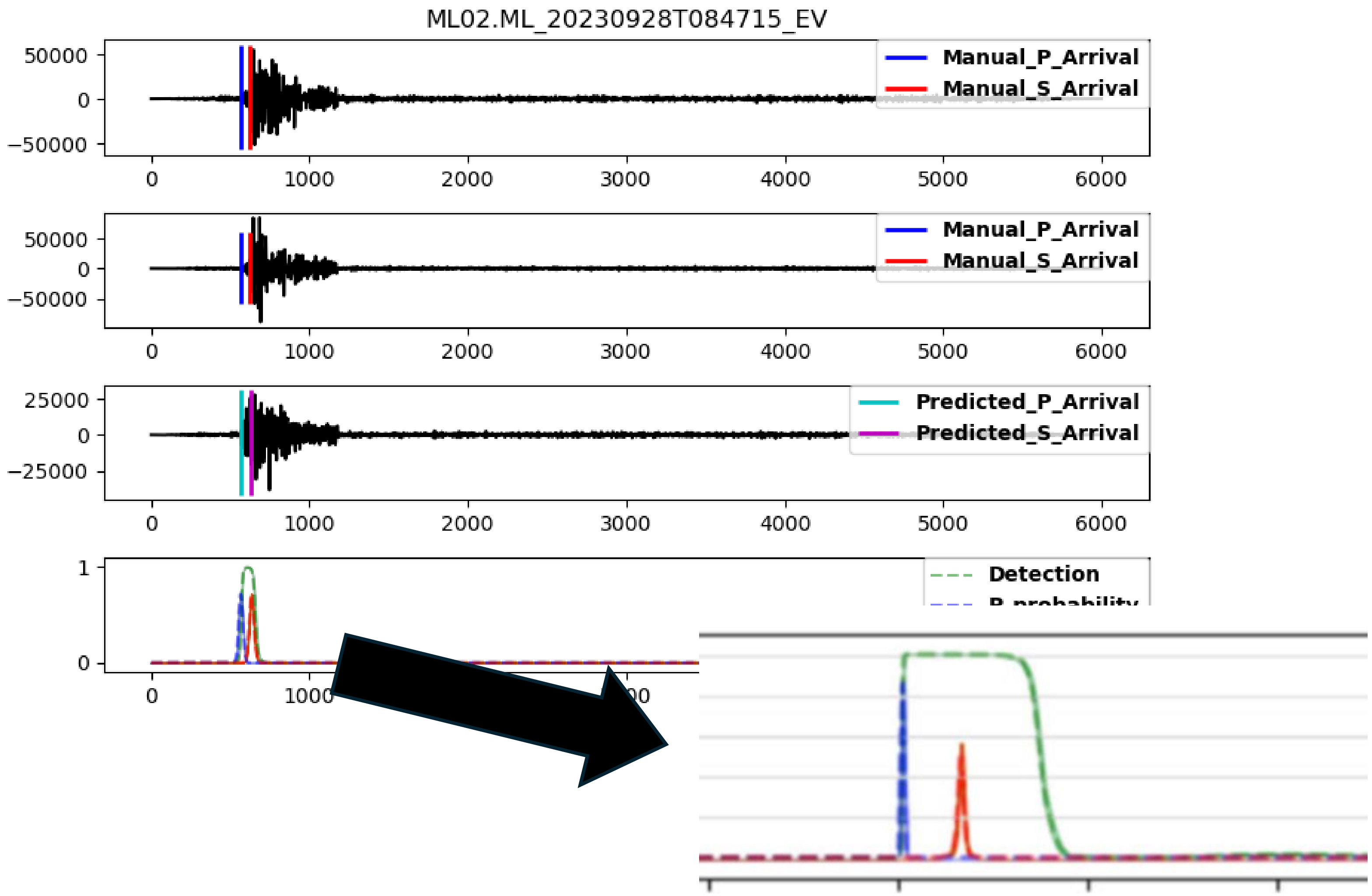
Source : https://www.v7labs.com/blog/deep-learning-guide

The multi-layer neural network consist of :

- **Input layer :** The only visible layer in NN architecture (CSV file input).
- **Hidden layers :** The intermediate layer that do all the computations and extract the features from the data.
- **Output layer :** The layer where we get the final result.

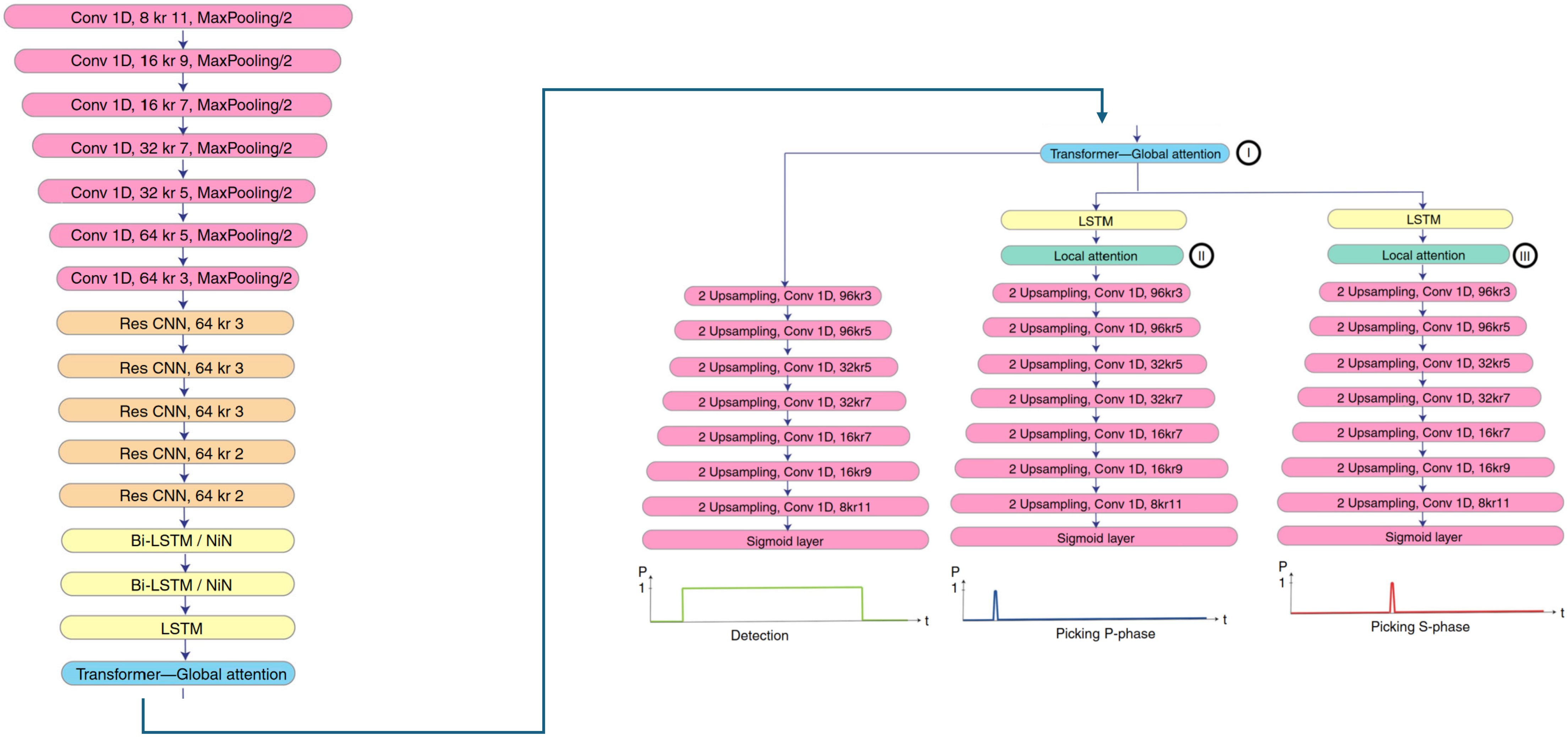
Deep Learning for Phase Picking

EQTransformer ➔ Deep learning models for phase picking, designed to simultaneously detect earthquakes and identify the onset of P-phase and S-phase by hierarchical attention mechanism.



Deep Learning for Phase Picking

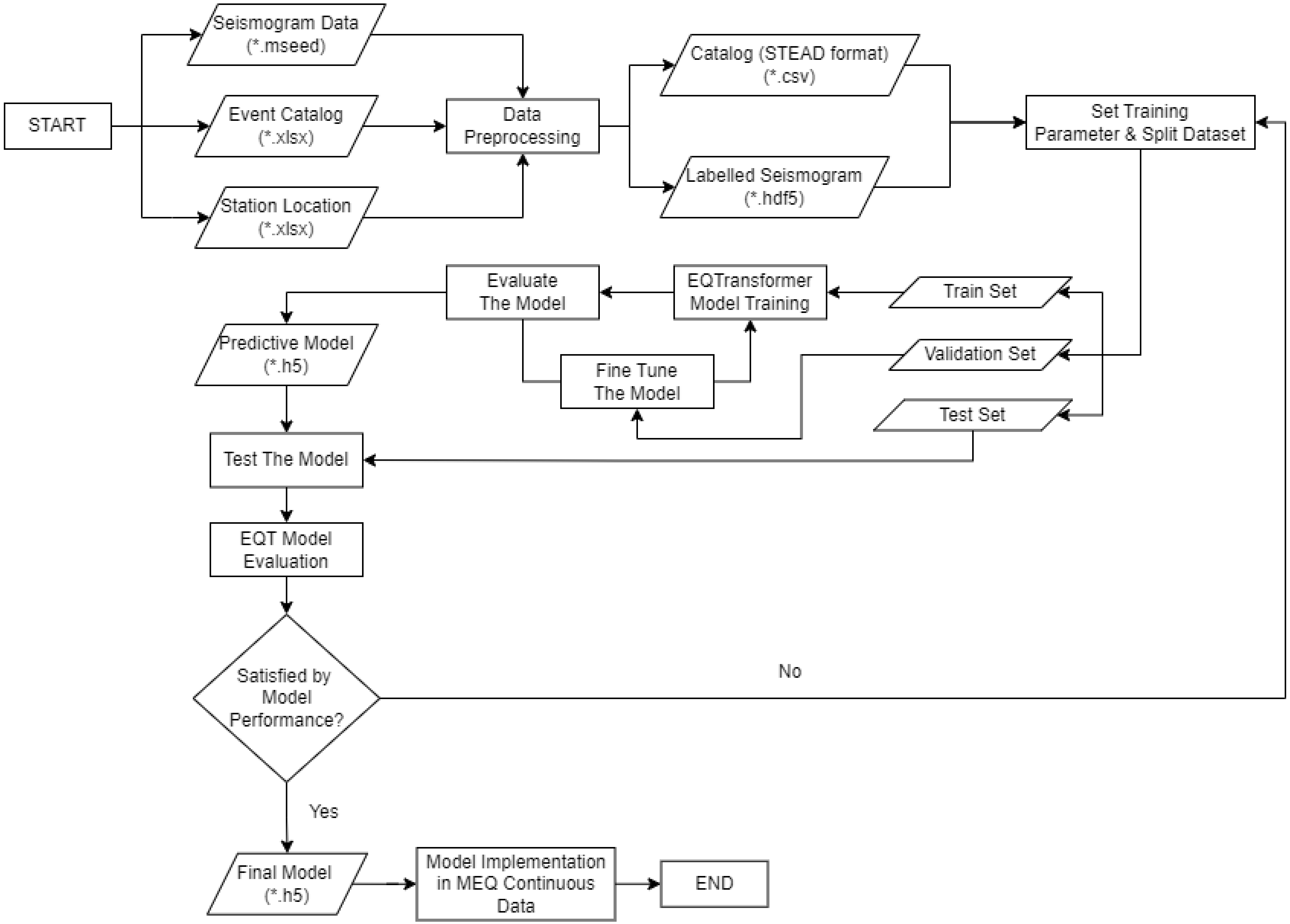
EQTransformer Network Architecture ➔ Consist of one very-deep encoder and three separate decoders composed of 1D convolutions, bi-directional and uni-directional long-short-term memories (LSTM).



Source : Mousavi et al., 2020

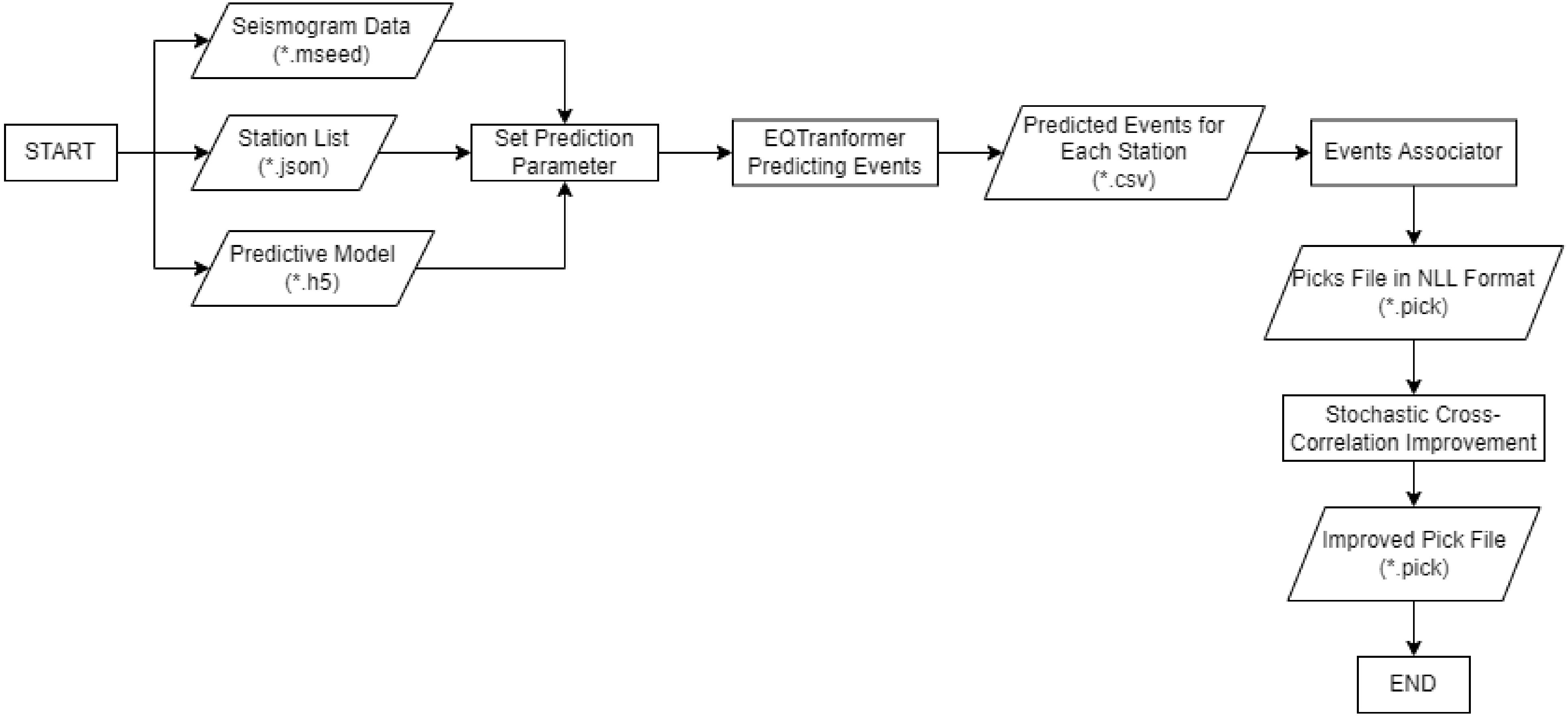
Methodology

EQTransformer Model Training Flowchart



Methodology

EQTransformer Model Implementation for Predicting Earthquake Flowchart



Data Processing Prior to Training

- The data used to train our model is exclusively from the SEML area. In the future, we plan to include the SERD dataset as well.
- Reformat the SEML MEQ routine Catalog to Stanford Earthquake Dataset (STEAD) format and save it in csv format, the full dataset consist of total 5,737 microearthquakes from 15 stations available from 815 events and total 2,360 of noises data.
- Seismogram conditioning:
  - ➢ Trim the seismogram to 60 seconds duration, covering event's coda.
  - ➢ Isolate the event trace and create synthetic noise to manipulate the empty trace .
  - ➢ Apply filter with bandpass filter 1 – 45 Hz.
  - ➢ Resample the data to 100 samples per second.
  - ➢ Apply demean and detrending to the seismogram data.
  - ➢ Taper with 5% cosine taper for each tip.
  - ➢ Transform the data into a 6000x3 dimensional array, since the seismogram has 3 components.
  - ➢ Save and label the data with all necessary tags for EQTransformer training in HDF5 format.

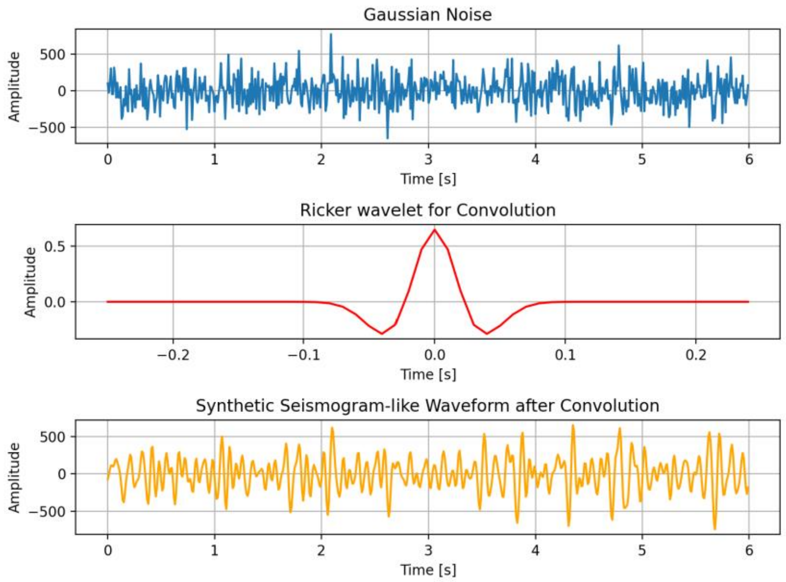
Data Processing Prior to Training

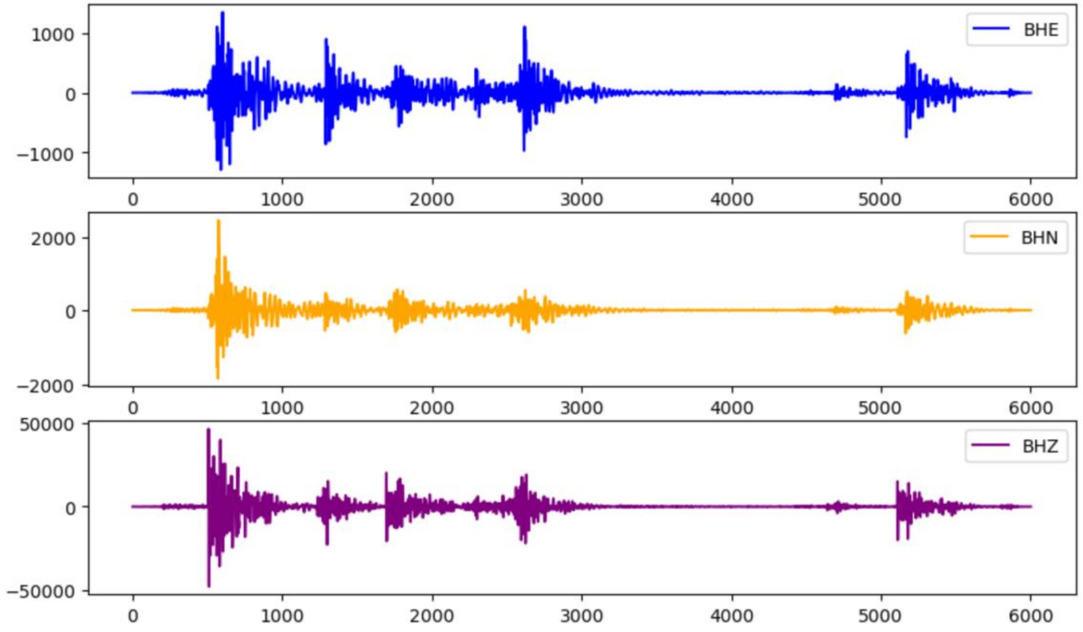
Labels assigned to the seismogram data:

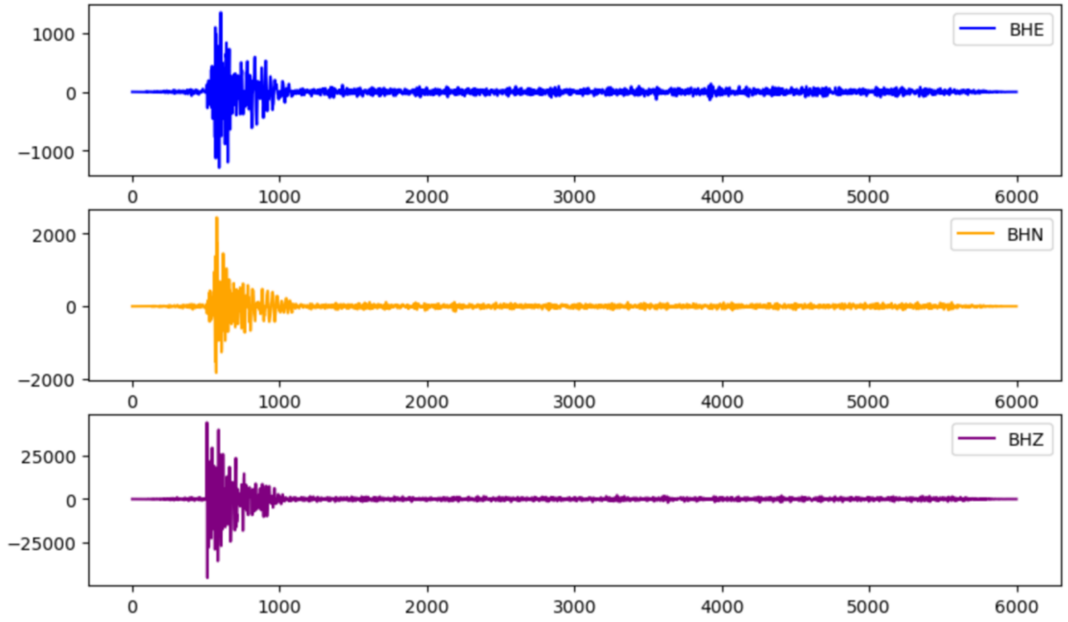
```
back_azimuth_deg
coda_end_sample
network_code
p_arrival_sample
p_status
p_travel_sec
p_weight
receiver_code
receiver_elevation_m
receiver_latitude
receiver_longitude
receiver_type
s_arrival_sample
s_status
s_weight
snr_db
source_depth_km
source_depth_uncertainty_km
source_distance_deg
source_distance_km
source_error_sec
source_gap_deg
source_horizontal_uncertainty_km
source_id
source_latitude
source_longitude
source_magnitude
source_magnitude_author
source_magnitude_type
source_mechanism_strike_dip_rake
source_origin_time
source_origin_uncertainty_sec
trace_category
trace_name
trace_start_time
```

Data Processing Prior to Training

Isolate the event and fill the empty trace with convoluted Gaussian noise



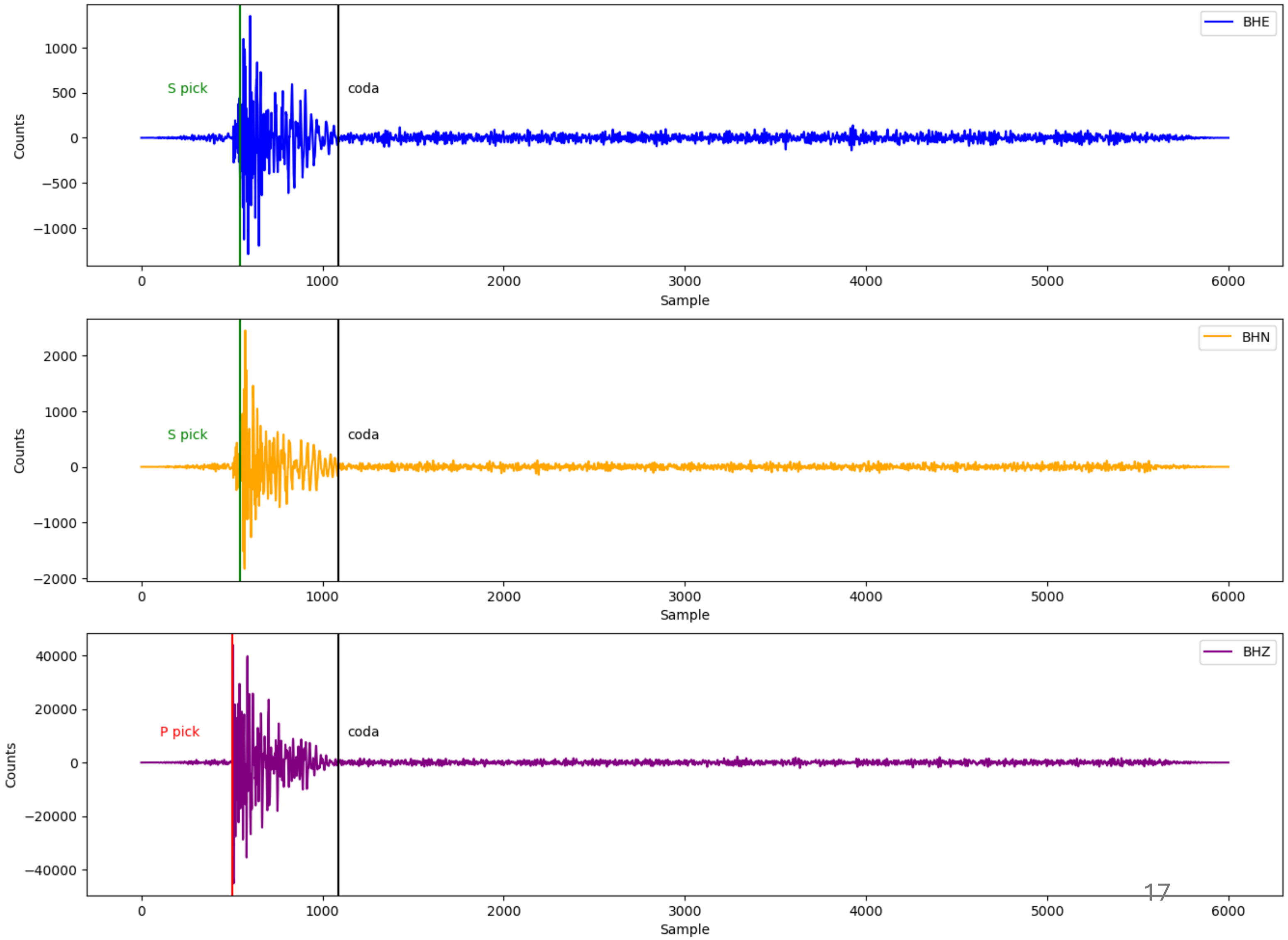




Data Processing Prior to Training

Seismogram data with full label and simple plot with the P, S, and Coda.

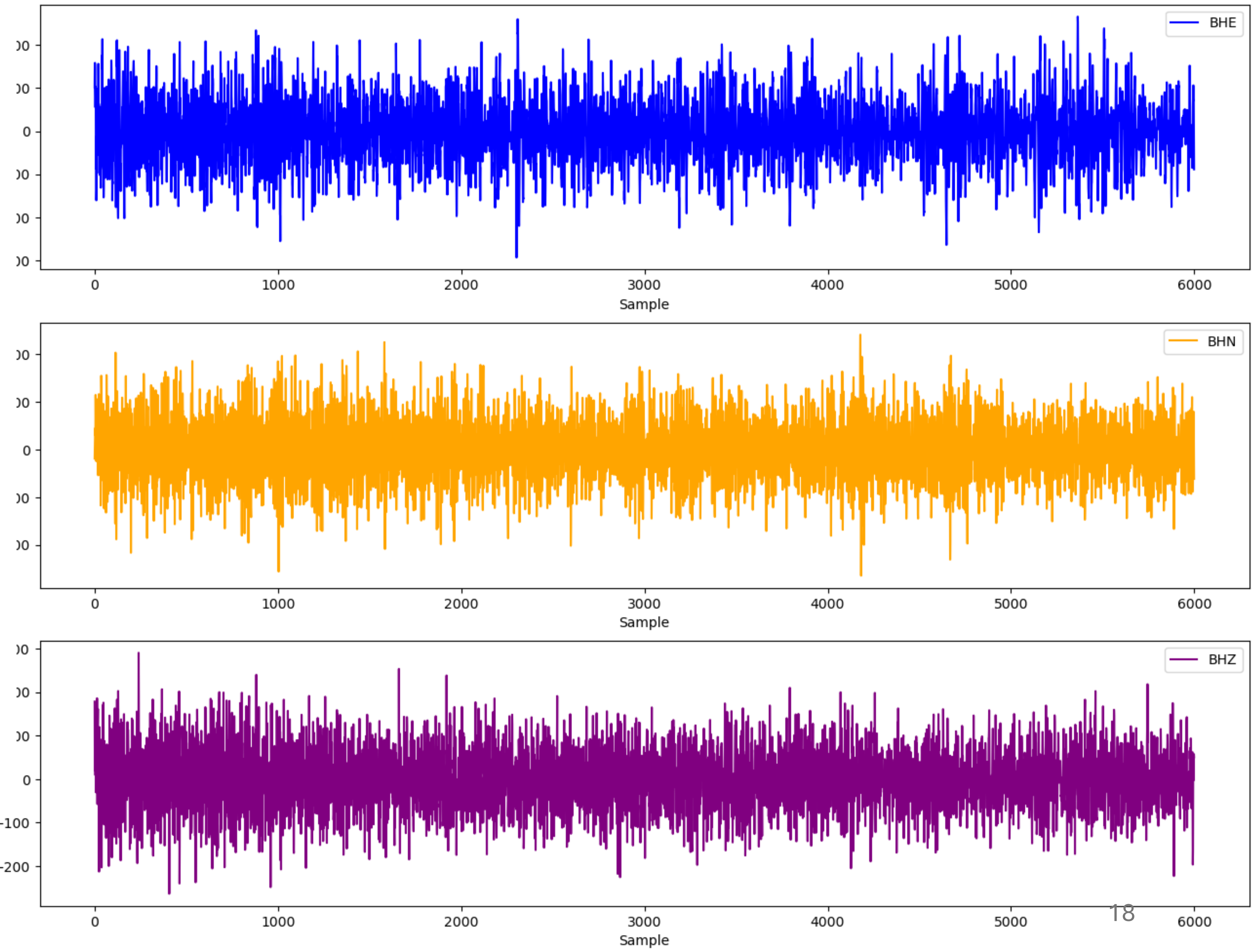
```
back_azimuth_deg 342.7742321006944
coda_end_sample [[1087]]
network_code ML
p_arrival_sample 503
p_status manual
p_travel_sec 0.548567
p_weight 0.98
receiver_code ML06
receiver_elevation_m 1396
receiver_latitude -1.635350219
receiver_longitude 101.1419969
receiver_type BH
s_arrival_sample 545
s_status manual
s_weight 0.95
snr_db [21.64694782 23.36168946 -5.90079015]
source_depth_km 0.76442
source_distance_km 1.8757342470675313
source_id 3115
source_latitude -1.6461647
source_longitude 101.1453288
source_magnitude 0.489299014216566
source_magnitude_type mw
source_origin_time 2023-09-14 16:24:13.489999
source_origin_uncertainty_sec 0.015119
trace_category earthquake_local
trace_name ML06.ML_20230914T162409_EV
trace_start_time 2023-09-14 16:24:09
```



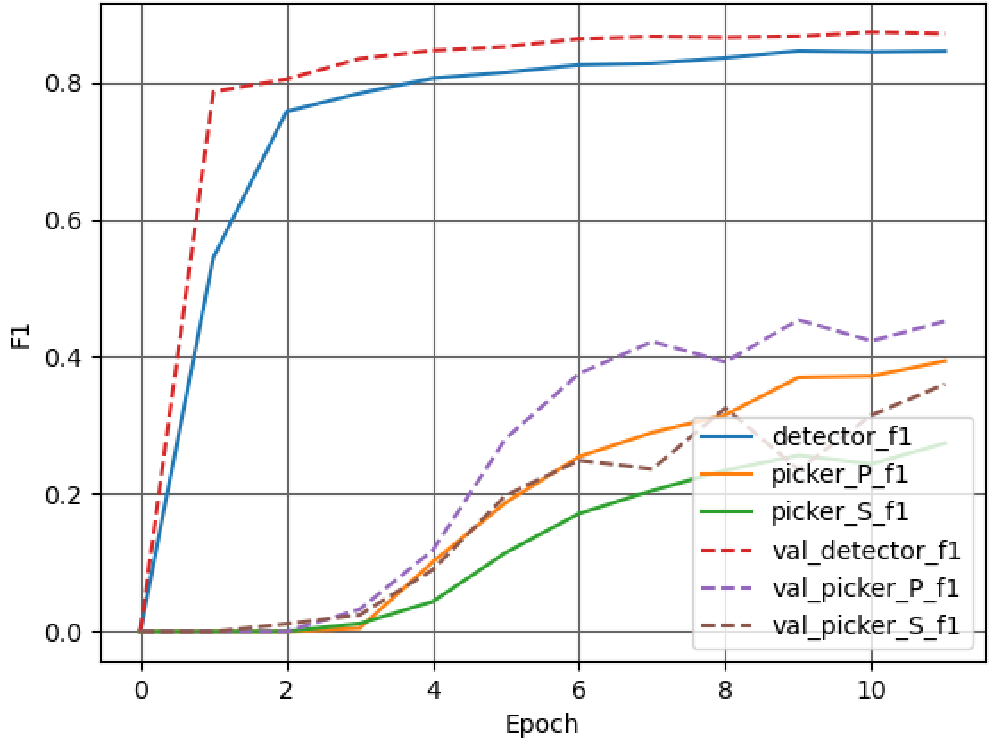
Data Processing Prior to Training

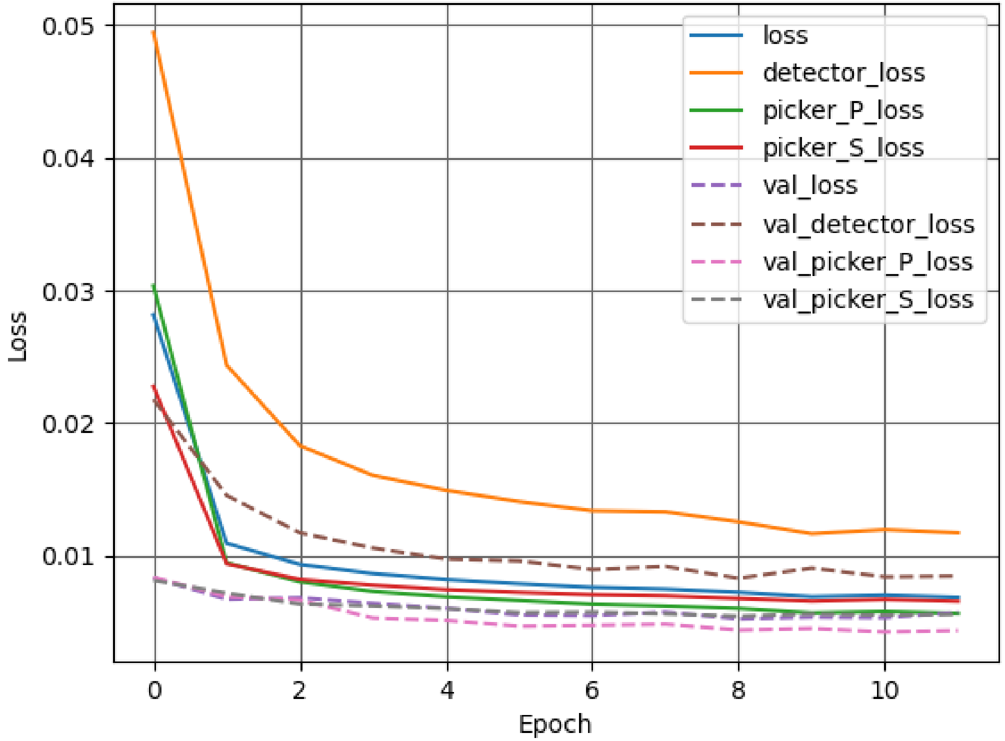
Noise data and the full labels.

```
back_azimuth_deg
coda_end_sample
network_code ML
p_arrival_sample
p_status
p_travel_sec
p_weight
receiver_code ML09
receiver_elevation_m 1270
receiver_latitude -1.618465946
receiver_longitude 101.138134
receiver_type BH
s_arrival_sample
s_status
s_weight
snr_db
source_depth_km
source_distance_km
source_id
source_latitude
source_longitude
source_magnitude
source_magnitude_type
source_origin_time
source_origin_uncertainty_sec
trace_category noise
trace_name ML09.ML_20240707T111500_NO
trace_start_time 2024-07-07 11:15:00
```



Evaluation of Trained Model Performance



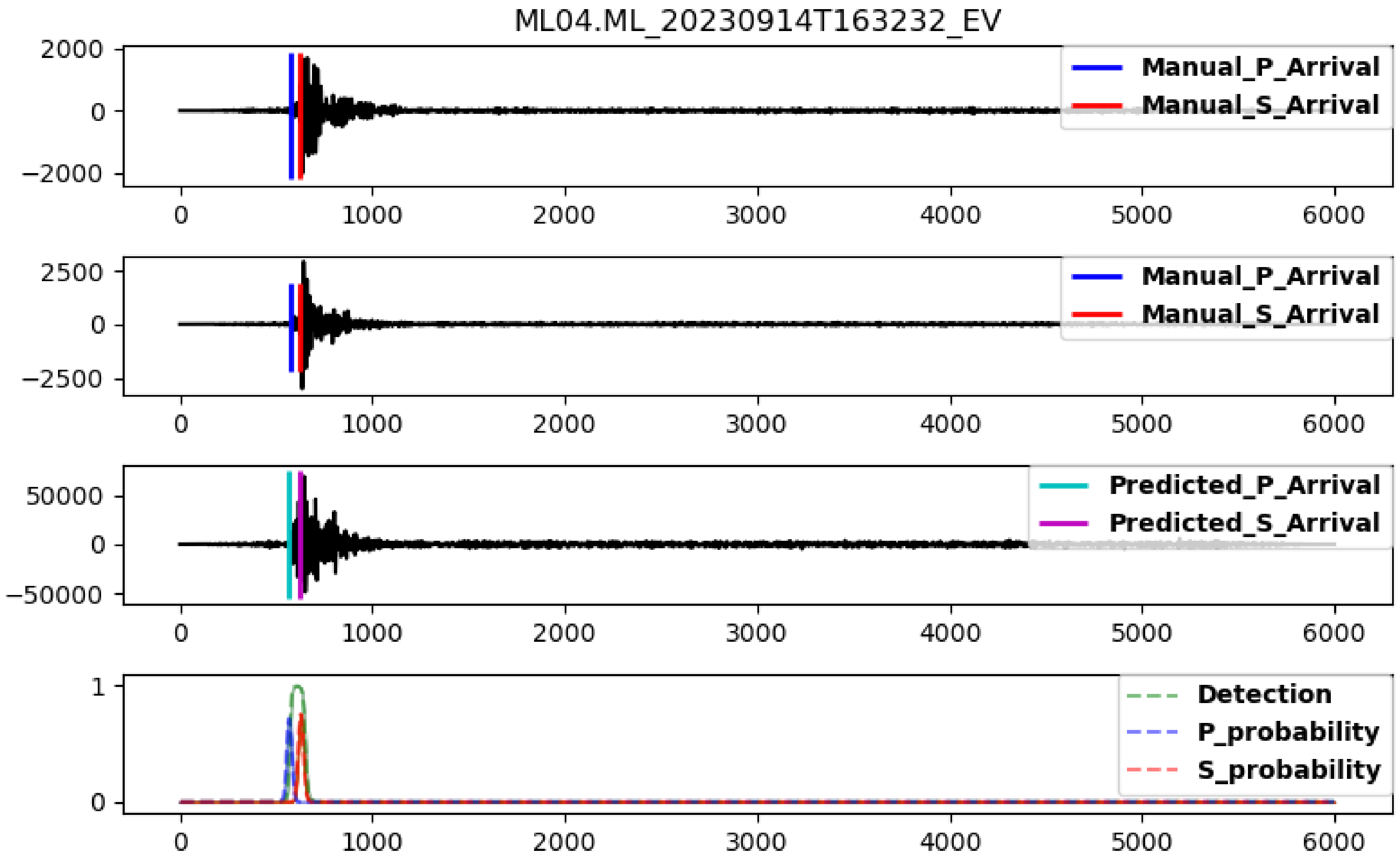


Implication: Based on this performance, the trained model performs excellently in detecting earthquake occurrences but struggles to accurately identify the P and S phase arrival times in real continuous seismogram data due to its complexity. This, expectedly, leads to false positives, where the model incorrectly identifies non-earthquake events as potential earthquakes.

Possible Cause : The amount of current dataset is not big enough, hindering the model's ability to generalize ability. Additionally, the quality of synthetic noise used when isolating the event trace outperforms the data from very small magnitude earthquakes (Mw ≤ 0), potentially affecting the model's generalization ability..

Results of Predictive Model Implementation

Example 1

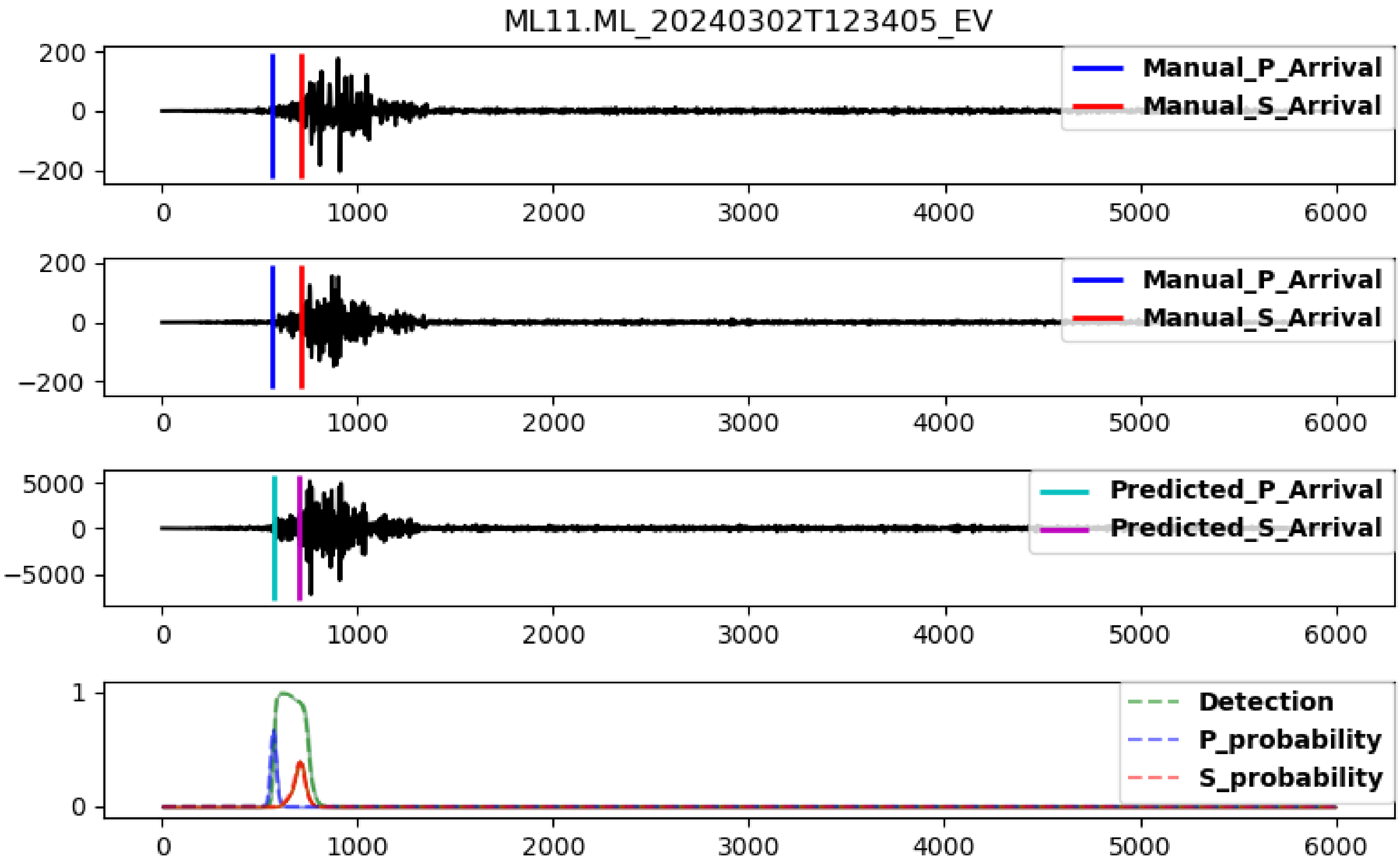


P_error : + 0.03 s
S_error : - 0.02 s

Detection_probability: 0.812
P_probability : 0.717
S_probability : 0.747

Results of Predictive Model Implementation

Example 2

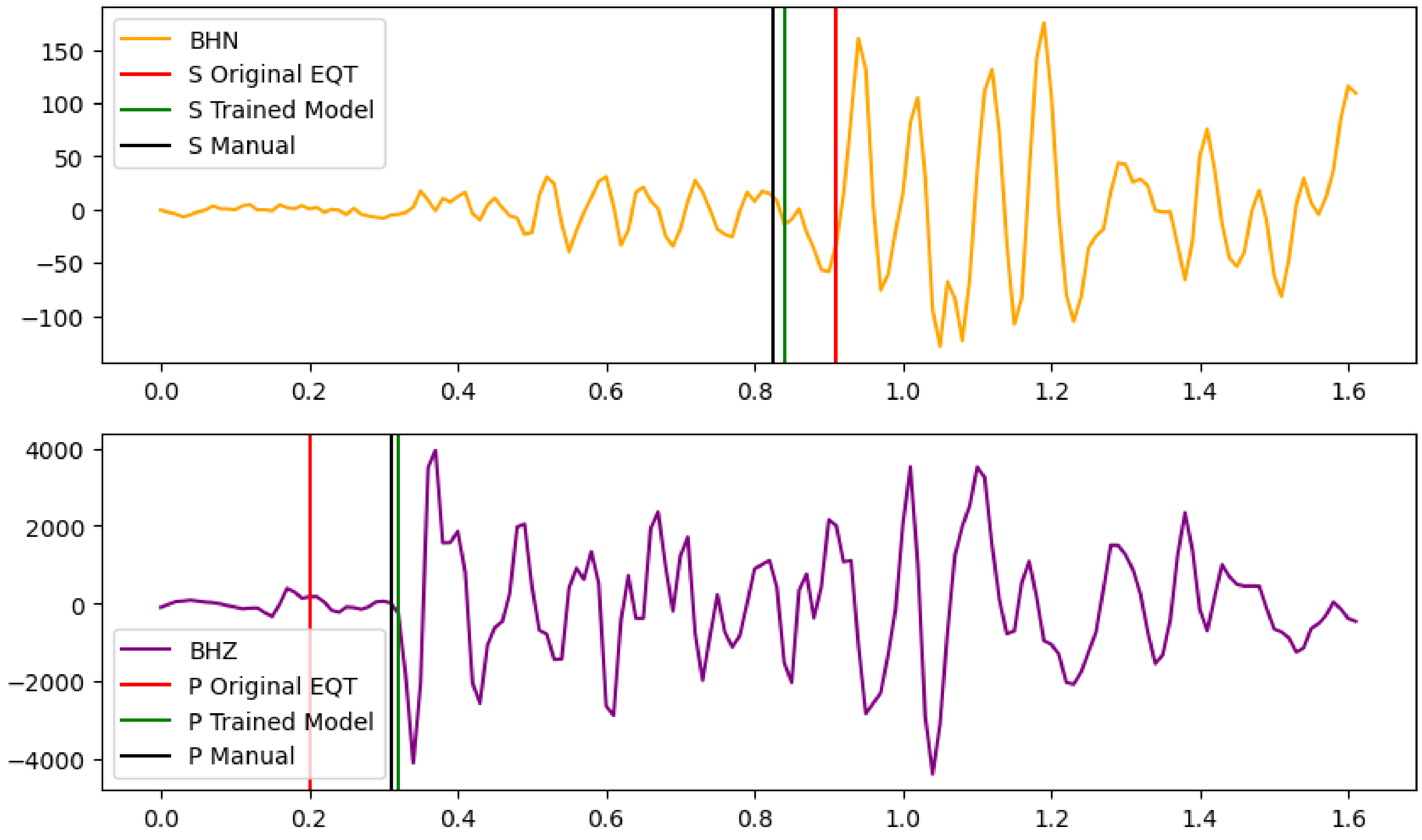


P_error : - 0.01 s
S_error : + 0.06 s

Detection_probability : 0.836
P_probability : 0.673
S_probability : 0.385

Comparison with Manual Picking

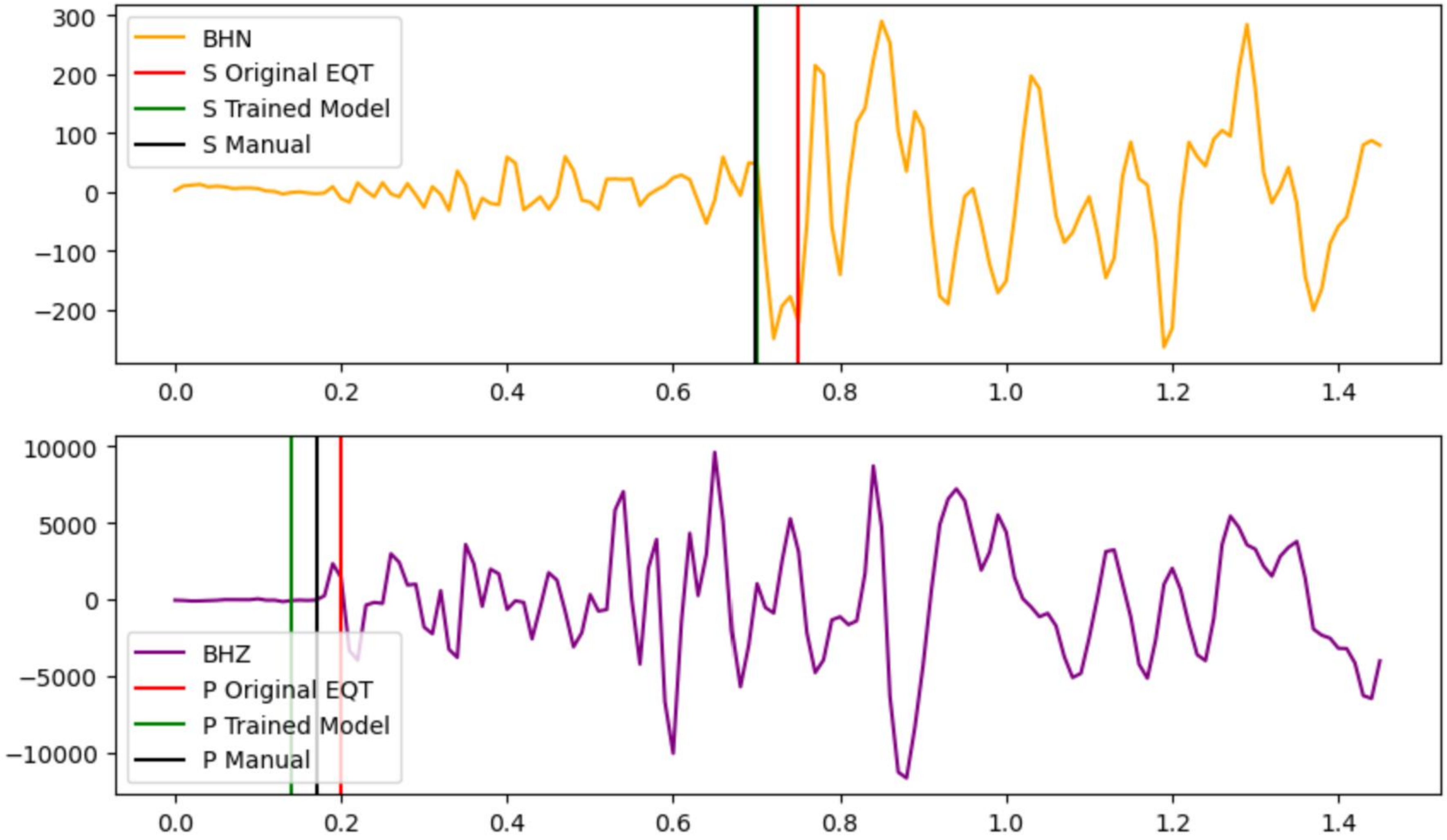
Example 1



In this first example, the locally trained model outperformed the original EQTransformer model and closely matched the manual picking results

Comparison with Manual Picking

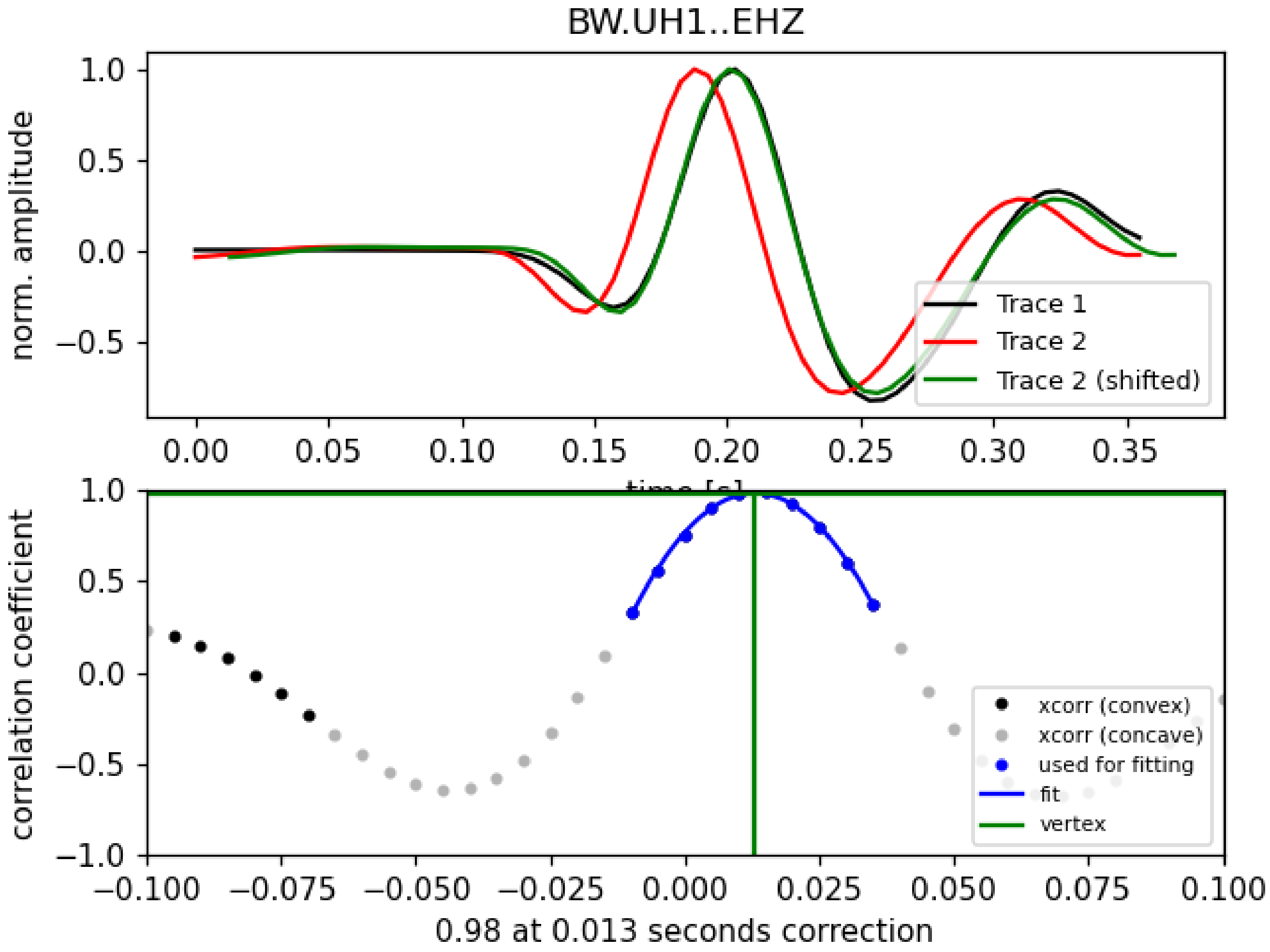
Example 2



In this second example, both the locally trained model and the original EQTransformer slightly deviated from the manual pick for the P arrival. However, for the S arrival time, the locally trained model provided a better solution and perfectly matched the manual pick.

Stochastic Accuracy Improvement

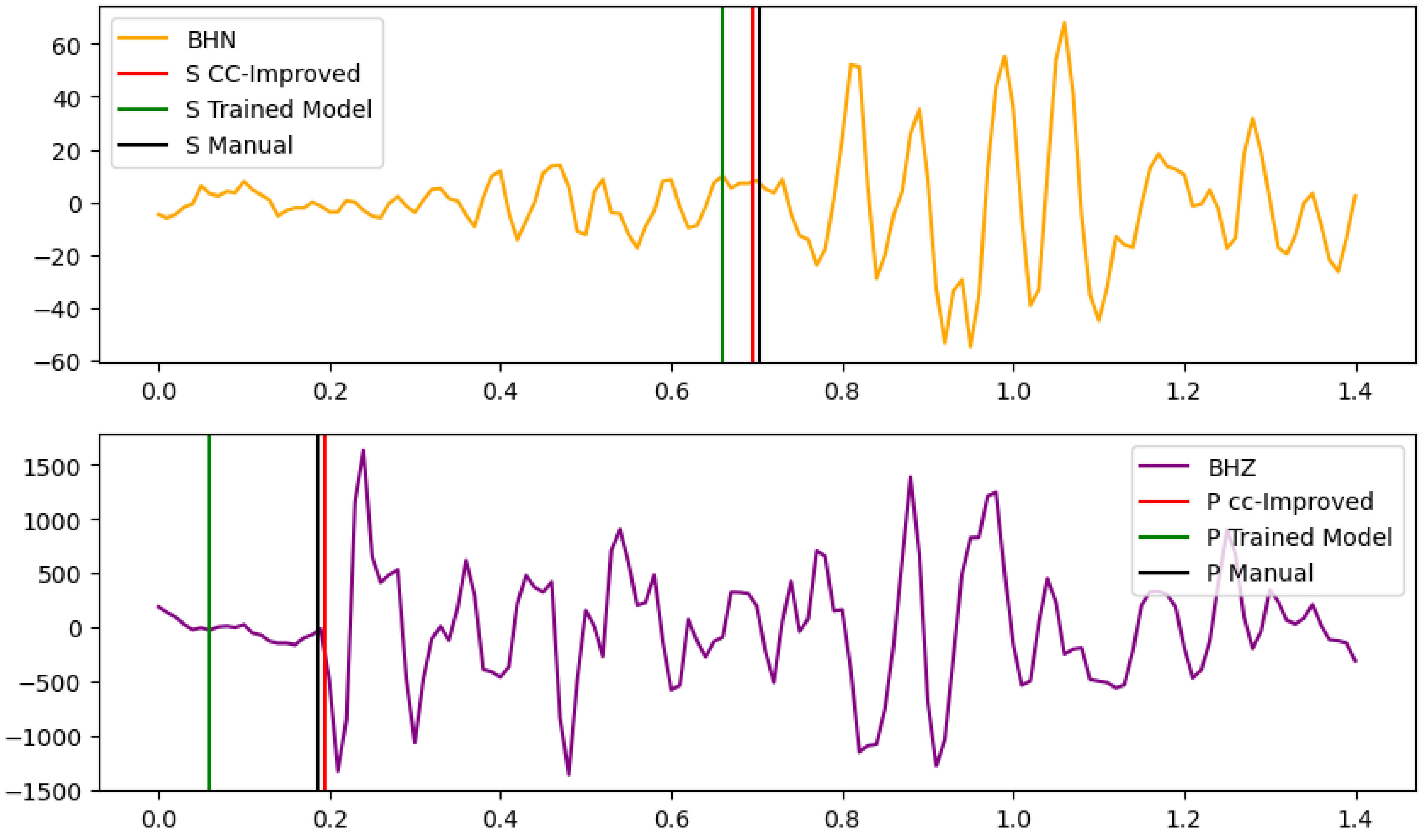
To achieve even better accuracy in arrival time for both the P and S phases, a stochastic cross-correlation correction can be applied. This is possible since we already have hundreds of manually picked events in our dataset, which have demonstrated high accuracy in the studied region.



Using the random search algorithm, we perform cross-correlation 50 times for each phase to find the best fit. The best fit is defined as the automatically picked phases that achieve the highest correlation coefficient with the random dataset, as waveforms from the same station and region tend to be similar. The correction time is then applied to the automatically picked phases.

Stochastic Accuracy Improvement

Example of Cross-correlation Improvement Results



The stochastic cross-correlation correction successfully improved the accuracy of our locally trained model. Even a slight improvement can have a significant impact on the final result of our hypocenter distribution.

A One Day Automatic Monitoring Test

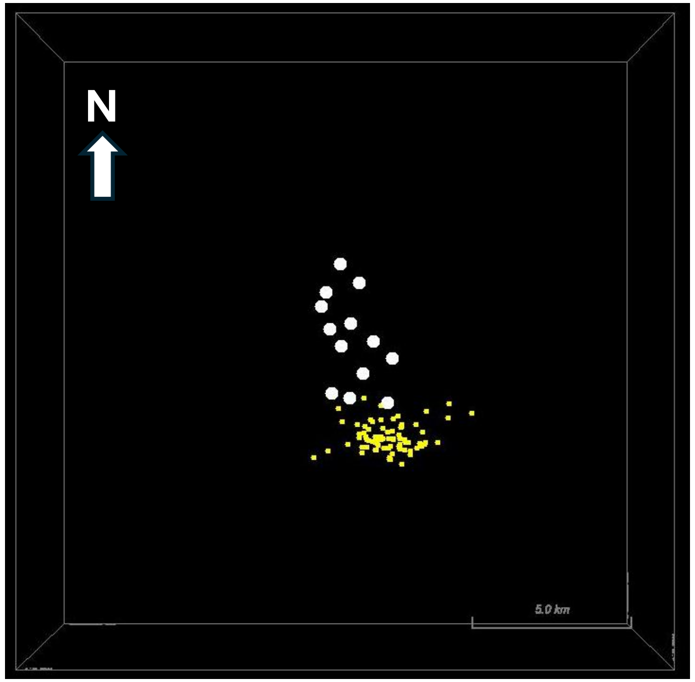
One full day of monitoring data was used to test the performance of our locally trained model. The data, collected on September 14, 2023, included recordings from 14 active stations with the detail result as below:

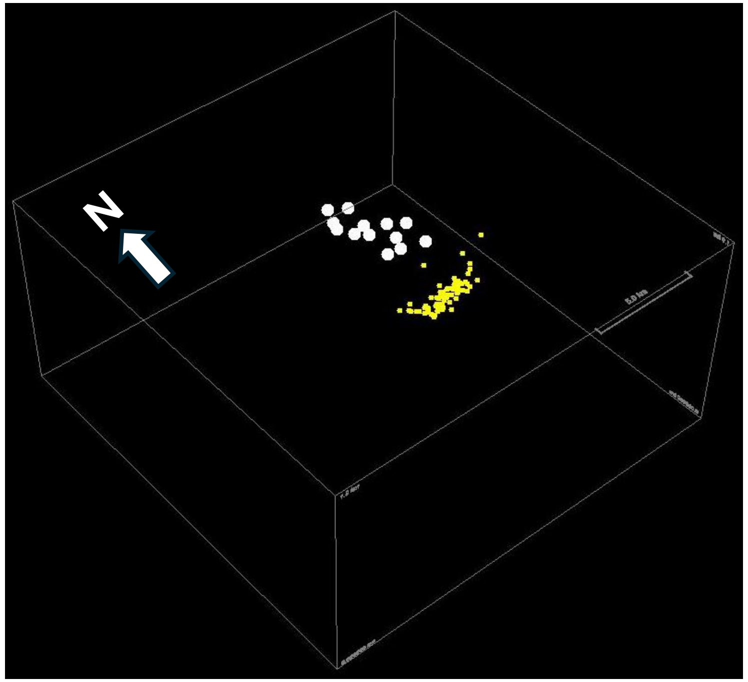
- **Locally trained model** succeeded in detecting and determining the P and S phases for 71 events, with a total of 866 phases.
- **EQTransformer original Model** only able to detect 41 events, with a total of 528 phases.
- In comparisSucceeded in detecting and determining the P and S phases for 71 events, with a total of 870 phases.on, **manual picking** at the same day succeeded to detect 77 event, with a total of 1170 phases.

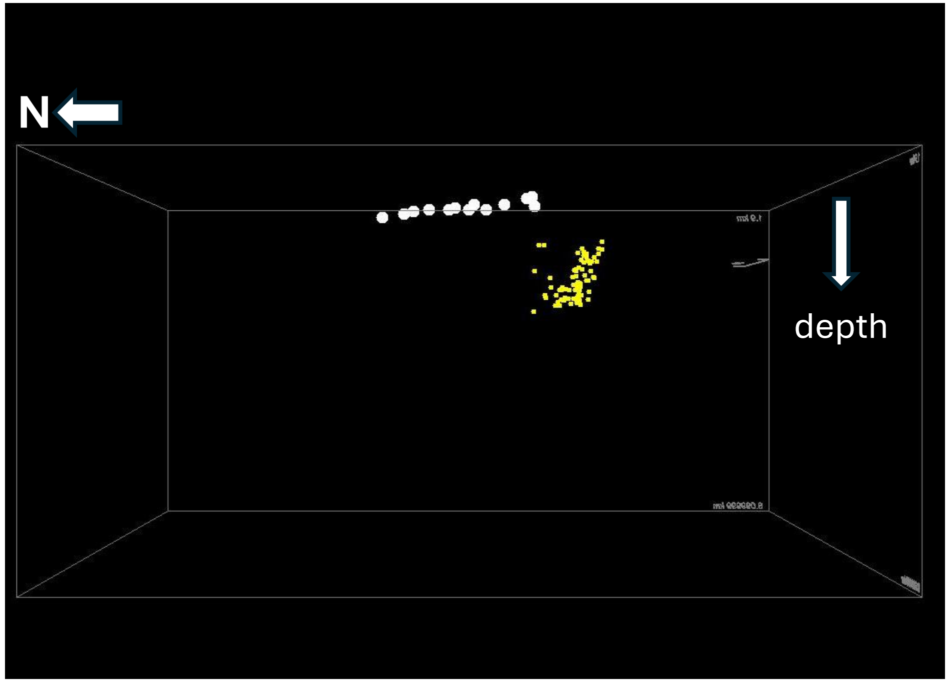
A One Day Automatic Monitoring Test

Hypocenter distribution, in comparison with manual routine procedure.

1. Manual Picking

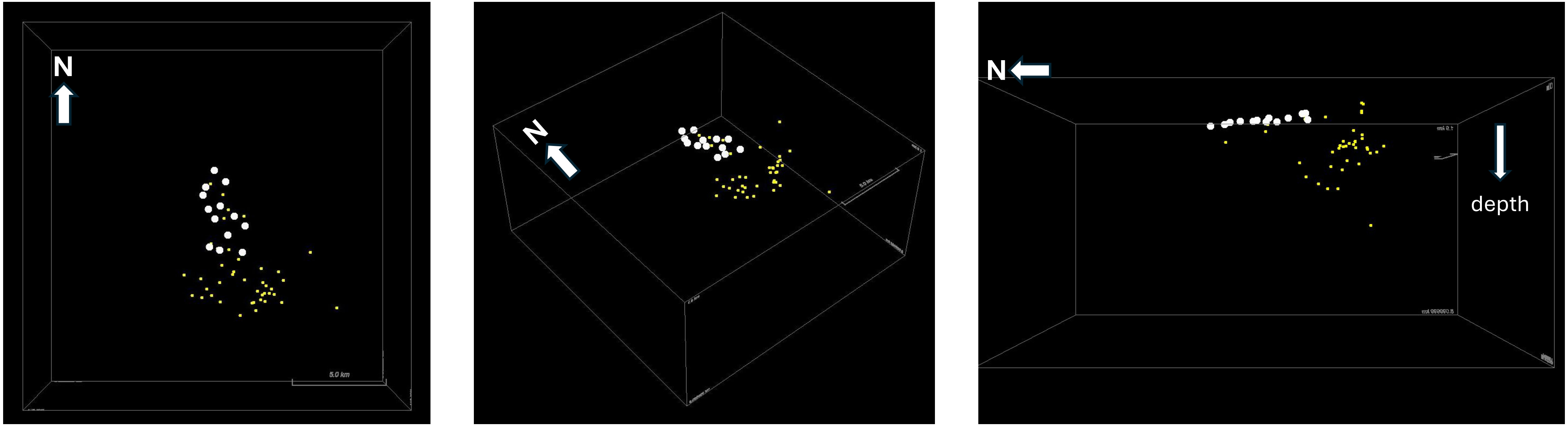






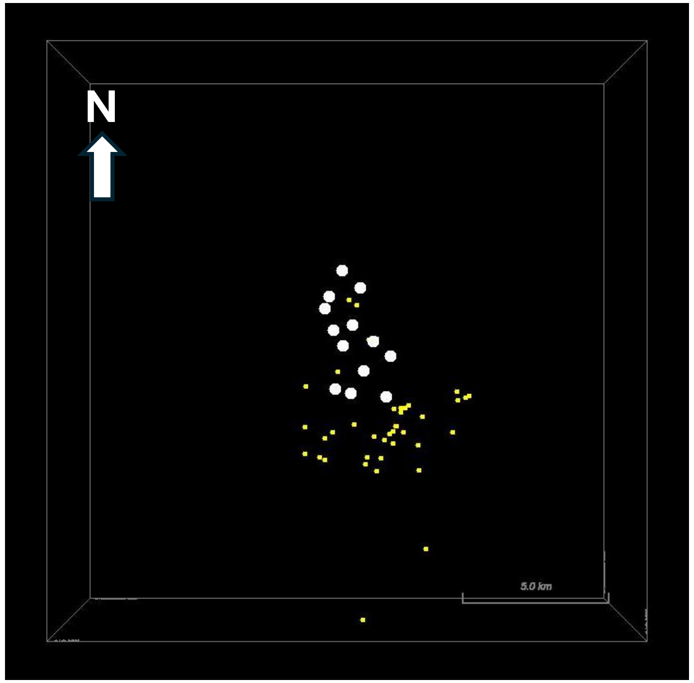
A One Day Automatic Monitoring Test

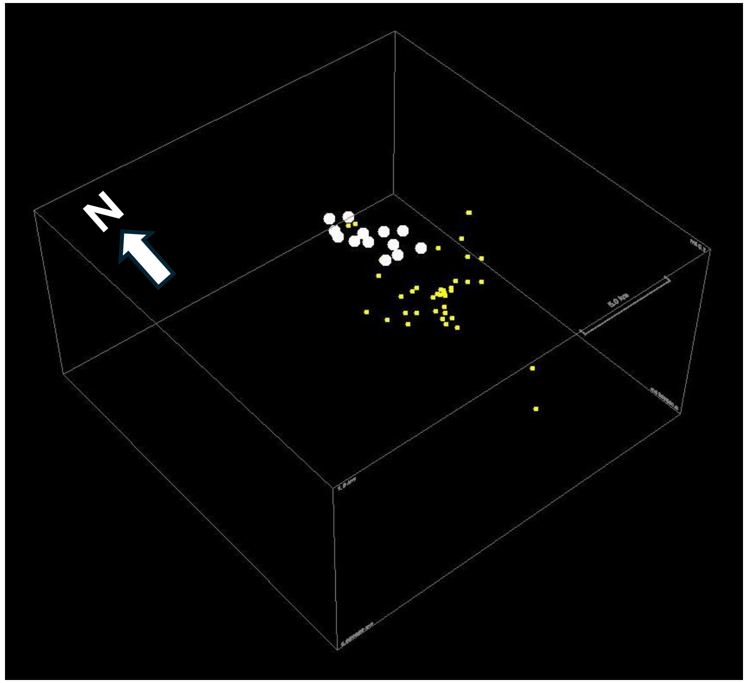
2. Original EQTransformer Model (without cross-correlation improvement)

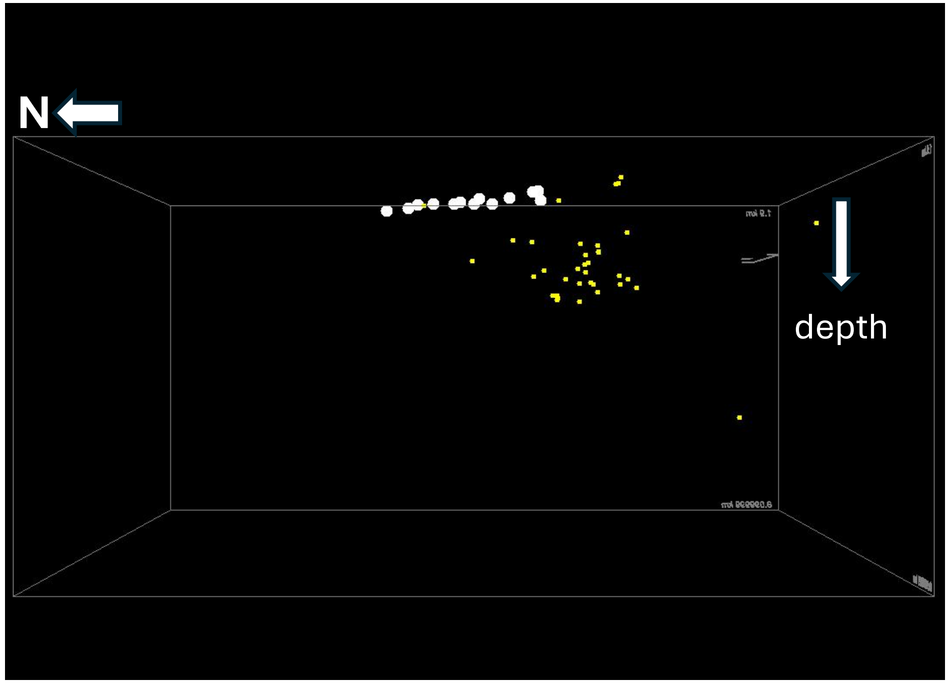


A One Day Automatic Monitoring Test

3. Original EQTransformer Model (with cross-correlation improvement)

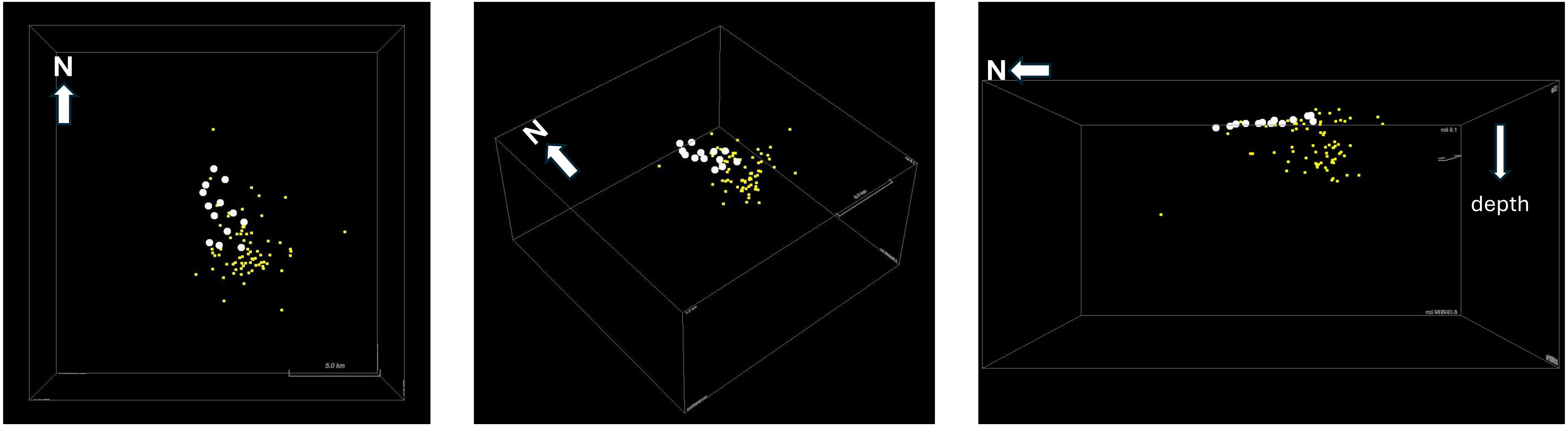






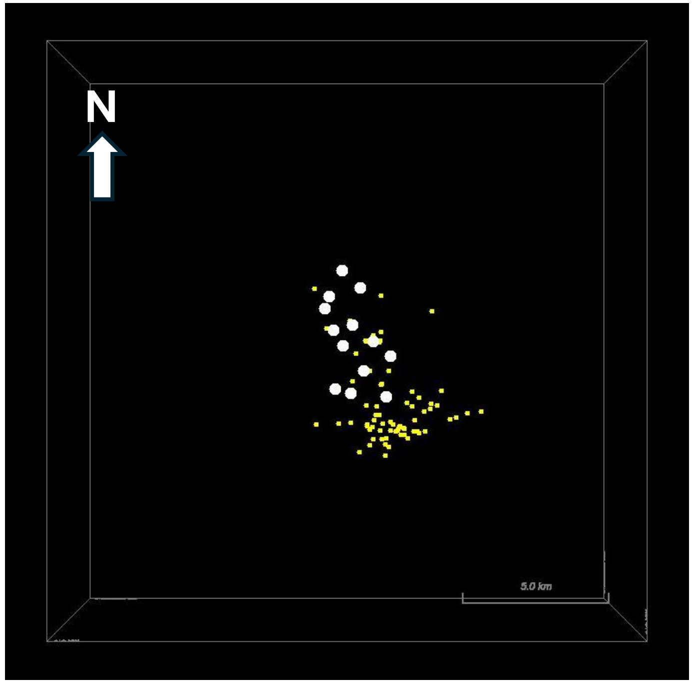
A One Day Automatic Monitoring Test

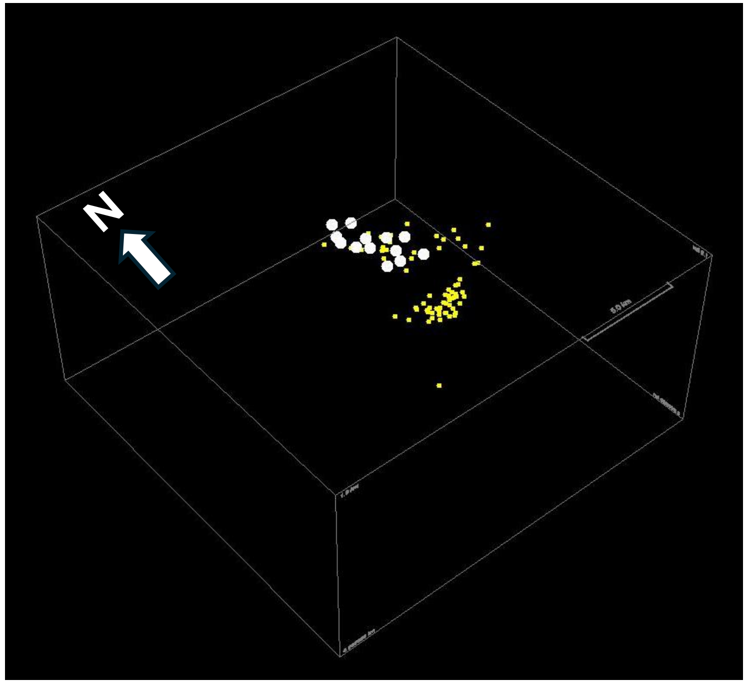
4. Our Locally Trained Model (without cross-correlation improvement)

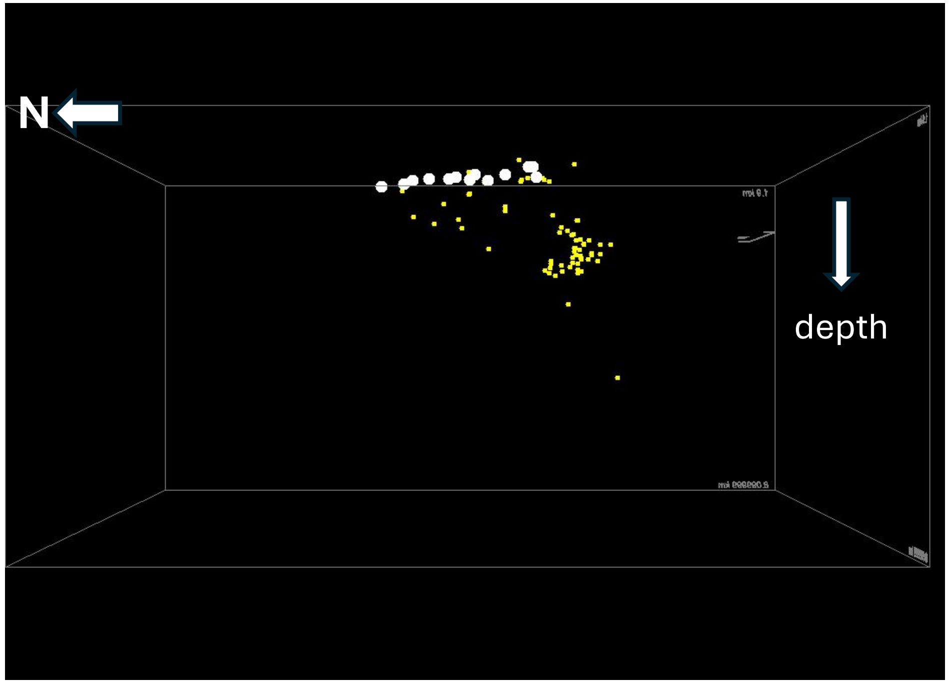


A One Day Automatic Monitoring Test

5. Our Locally Trained Model (with cross-correlation improvement)





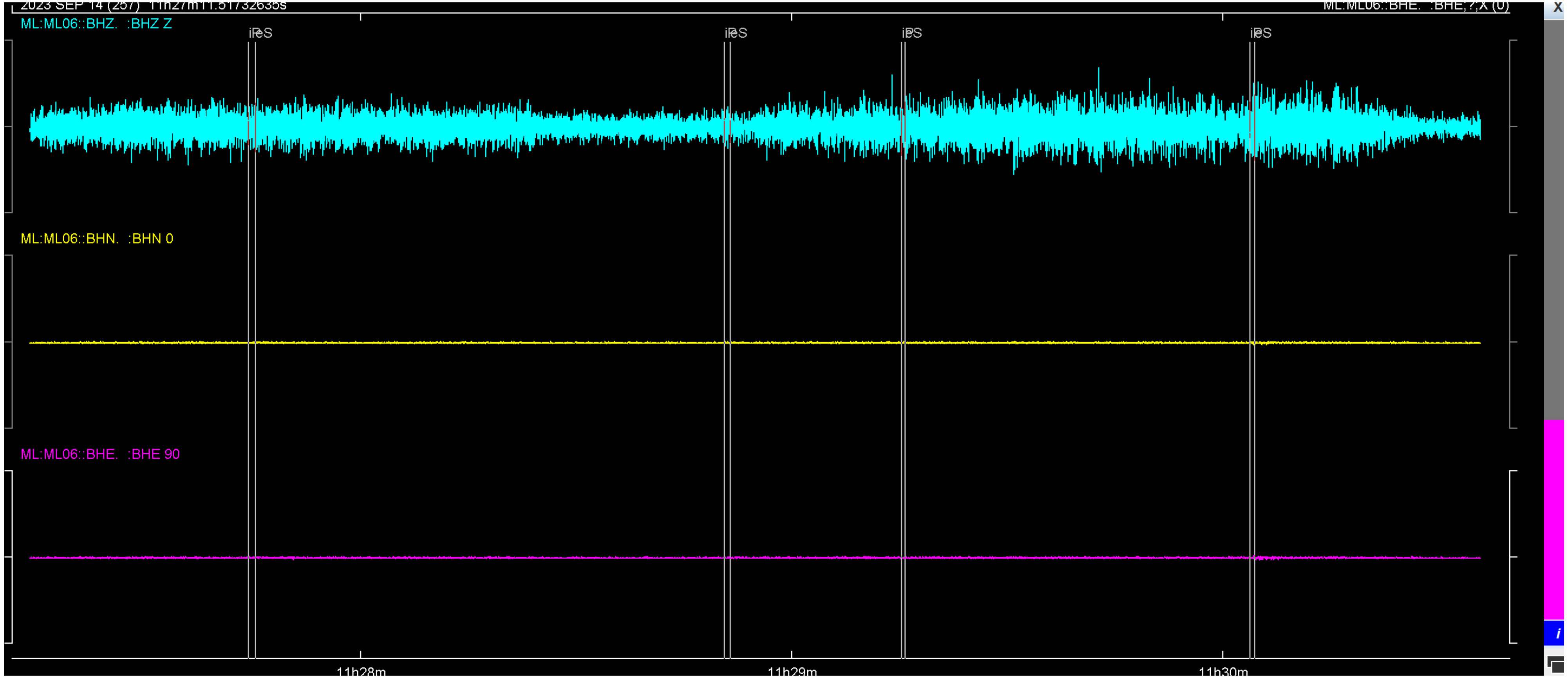


Discussion, Model Limitations and Future Improvements

- Our locally trained model, based on the EQTransformer deep learning architecture, has demonstrated promising reliability and accuracy for daily routine monitoring. However, as indicated by the training performance, there are still noticeable false positives.
- The stochastic cross-correlation correction has significantly improved our hypocenter distribution, but this method continues to struggle with minimizing false positives, as some falsely picked events may not even be earthquakes in the first place.
- To address this challenge, we need to enhance our trained model. A larger and more diverse dataset is required, as the current model has been trained only on data from the SEML area. Combining this with the SERD dataset in the future will likely improve both sensitivity and accuracy.
- Additionally, to fully automate the monitoring of microearthquakes (MEQs) in the future, we will need to design a detailed and robust pipeline that can handle real-time monitoring effectively.

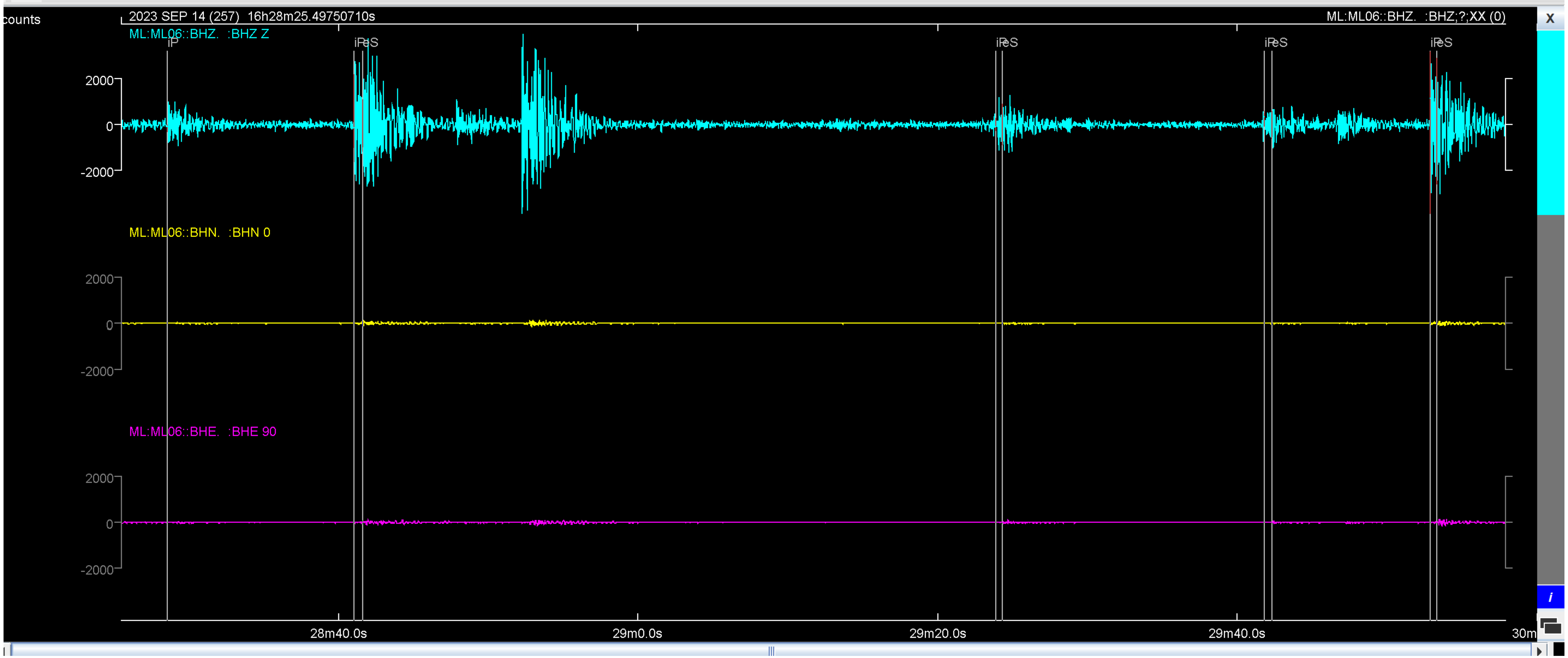
Discussion, Model Limitations and Future Improvements

- **Model limitation example 1** : False positive picks, to improve this we need a greater variety of labelled noise data for training.



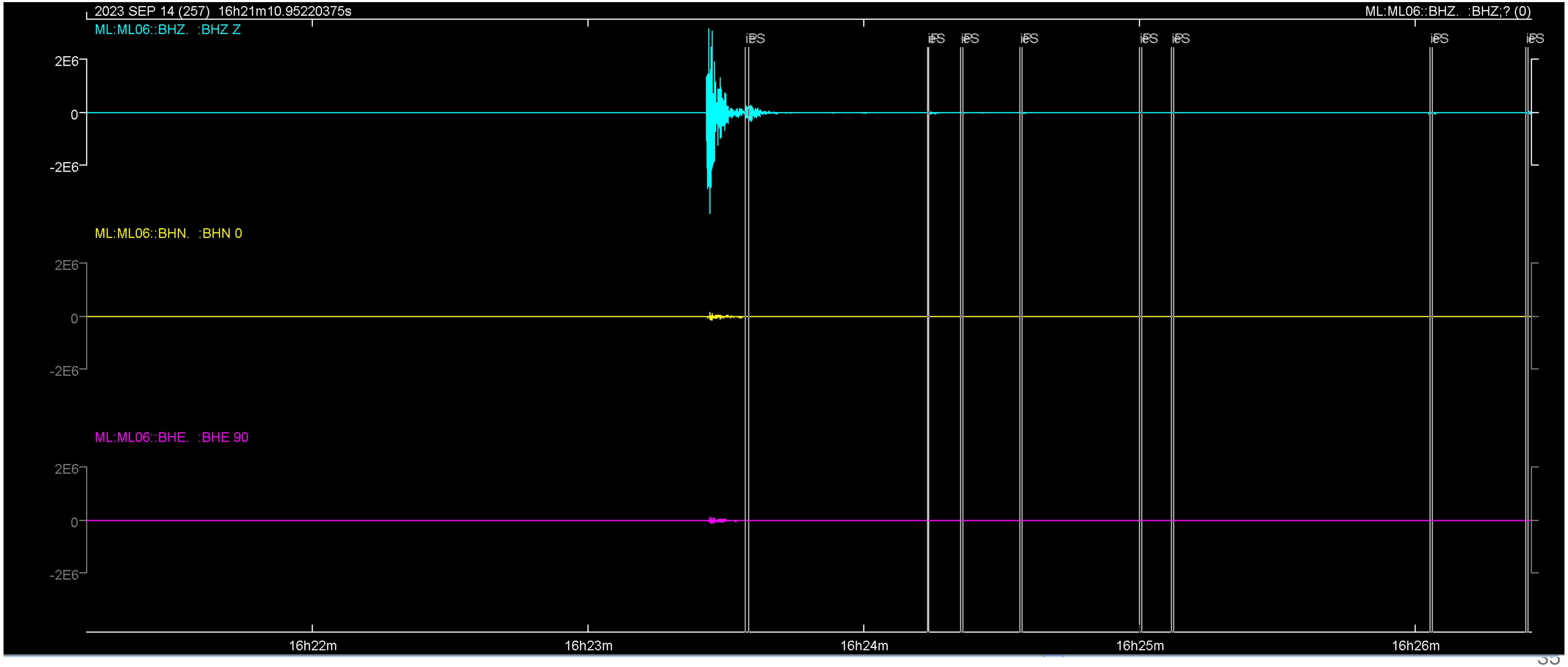
Discussion, Model Limitations and Future Improvements

- **Model limitation example 2** : The model still struggles to detect earthquakes that are closely grouped with other earthquakes.



Discussion, Model Limitations and Future Improvements

- **Model limitation example 3** : The model ignores earthquakes with significantly larger amplitudes than the surrounding smaller earthquakes. This phenomenon is an inherent characteristic of the model, as we lack high-amplitude events in the training dataset.



References

Deichmann, N. dan Garcia-Fernandez, M. (1992), Rupture geometry from high-precision relative hypocentre locations of microearthquake clusters, Geophysical Journal International, 110 (3), 501-517. http://doi.wiley.com/10.1111/j.1365-246X.1992.tb02088.x

Deng, L. dan Yu, D. (2014), "Deep Learning: Methods and Applications", Foundations and Trends® in Signal Processing, Vol.7, No.3–4, hal. 197–387. http://doi.org/10.1561/2000000039.

Goodfellow, I., Bengio, Y. dan Courville, A. (2016), Deep Learning, Adaptive computation and machine learning, The MIT press, Cambridge, Mass.

Havskov, J. dan Ottemoller, L. (2010), Routine Data Processing in Earthquake Seismology, Springer Netherlands, Dordrecht. http://doi.org/10.1007/978-90-481-8697-6.

Mousavi, S.M., Ellsworth, W.L., Zhu, W., Chuang, L.Y. dan Beroza, G.C. (2020), "Earthquake Transformer—an Attentive Deep-Learning Model for Simultaneous Earthquake Detection and Phase Picking", Nature Communications, Vol.11, No.1, hal. 3952. http://doi.org/10.1038/s41467-020-17591-w.