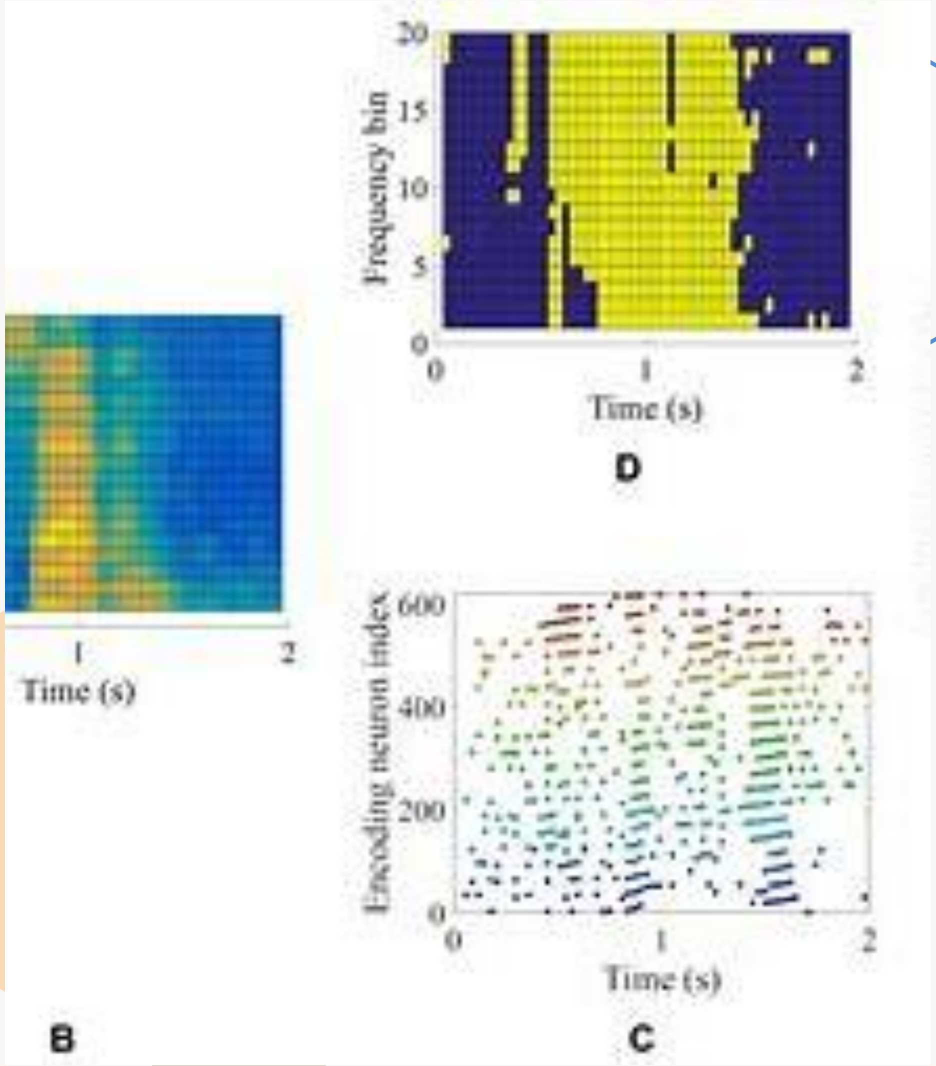
Signal Processing & Spike Detection

Mrigank Maharana

https://github.com/mkorkrish/NeuralPrograms



Combining traditional signal processing with machine learning

Program 1

- Objective
- Data Stimulation
- Machine Learning Classification
- Spike Stimulation
- Signal Filtering
- Label Generation
- Feature Extraction
- Data Splitting

Program 1

Objective

Process a simulated neural signal, detect spikes, and visualize the results.

Use signal processing and machine learning for spike detection.

Data Stimulation

FUNCTION: ACQUIRE_DATA

SIMULATES A SINUSOIDAL WAVEFORM.

ACTS AS A SIMPLE NEURAL SIGNAL.

NOISE INTRODUCTION:

RANDOM NOISE IS ADDED TO MIMIC REAL-WORLD NOISY RECORDINGS.

Spike Simulation

ARTIFICIAL "SPIKES" ARE INTRODUCED AT RANDOM INTERVALS.

THESE SPIKES REPRESENT NEURAL EVENTS OF INTEREST.

Signal Filtering

USE BUTTERWORTH LOW-PASS FILTER.

FUNCTION: FILTER_DATA

REDUCES HIGH-FREQUENCY NOISE AND RETAINS MEANINGFUL SIGNAL COMPONENTS.

Feature Extraction

Use Short Time Fourier Transform (STFT).

Function: extract_features

Converts time domain signal to frequency domain.

Offers both frequency and temporal characteristics.

Key parameters:

nperseg (Number of data points per segment)

noverlap (Number of overlapping points)

Label Generation

Based on the artificial spikes.

Function: generate_labels

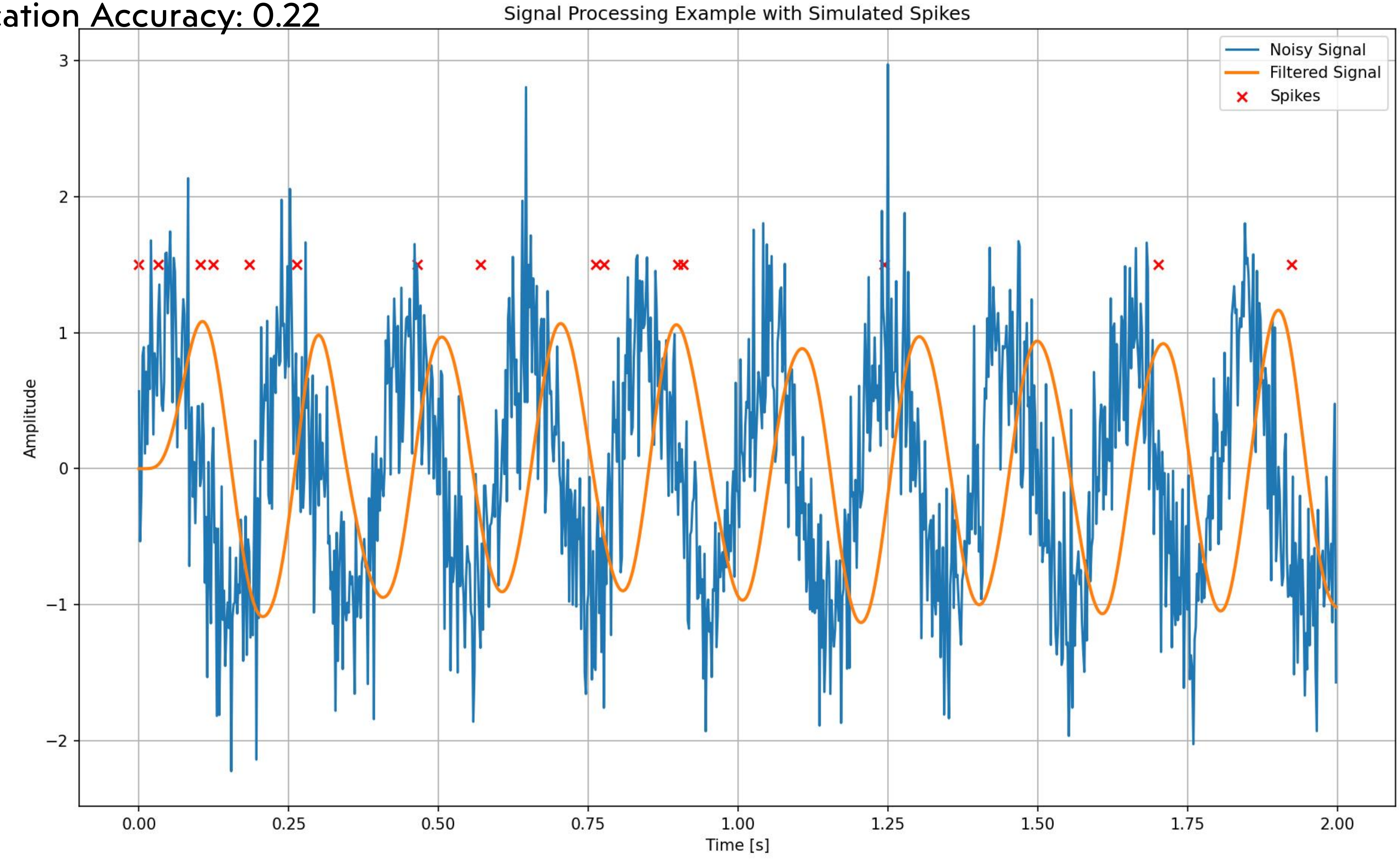
Generates labels for each STFT segment indicating the presence or absence of a spike.

Machine Learning Classification

Model: Logistic Regression
Binary classification: Spike or No Spike.
Model is trained on training set and evaluated on the test set.
Metric: Classification Accuracy

Features shape: (30,)
Labels shape: (30,)
Classification Accuracy: 0.22

Visualization



Combining traditional signal processing with machine learning

Program 1.1

Objective

Data Stimulation

Visualization

Neural Network Training

Spike Stimulation

Bandpass Filtering

Spike Labeling

Neural Network Architecture

Data Splitting

Objective

Program 1.1

Process a noisy sinusoidal signal.

Use bandpass filtering and a neural network for a demonstration of spike detection

Data Stimulation

FUNCTION: ACQUIRE_DATA

GENERATES A SINUSOIDAL WAVEFORM.

REPRESENTS A NEURAL SIGNAL OR ANY TIME-SERIES DATA.

INTRODUCE NOISE:

RANDOM NOISE IS ADDED TO MIMIC REAL-WORLD VARIATIONS.

Bandpass Filtering

Filtering to retain frequencies of interest and reject others.

Function: butter_bandpass and bandpass_filter

Filters the signal between 1.0Hz and 10.0Hz.

Neural Network Architecture

A simple feedforward neural network for "spike" detection.

Input layer: 32 neurons, ReLU activation

Hidden layer: 16 neurons, ReLU activation

Output layer: 1 neuron, Sigmoid activation

Compiled with Adam optimizer and binary cross-entropy loss.

Spike Labeling

Here, the "spike" labels are generated as:

1 if the original signal is positive.

0 otherwise.

Note: This is a simplistic approach just for demonstration purposes.

Data Splitting

70% TRAINING DATA, 30% TESTING DATA.

USE TRAIN_TEST_SPLIT FROM SKLEARN.

Neural Network Training

70% TRAINING DATA, 30% TESTING DATA.

USE TRAIN_TEST_SPLIT FROM SKLEARN.

Training Log of the Neural Network Model

```
Epoch 1/10
22/22 [==============================] - 1s 15ms/step - loss: 0.6958 - accuracy: 0.5343 - val_loss: 0.6824 - val_accuracy: 0.6367
Epoch 2/10
22/22 [==============================] - 0s 5ms/step - loss: 0.6719 - accuracy: 0.6629 - val_loss: 0.6732 - val_accuracy: 0.6267
Epoch 3/10
22/22 [==============================] - 0s 5ms/step - loss: 0.6590 - accuracy: 0.6643 - val_loss: 0.6639 - val_accuracy: 0.6267
Epoch 4/10
22/22 [==============================] - 0s 5ms/step - loss: 0.6454 - accuracy: 0.6657 - val_loss: 0.6563 - val_accuracy: 0.6233
Epoch 5/10
22/22 [==============================] - 0s 4ms/step - loss: 0.6322 - accuracy: 0.6671 - val_loss: 0.6511 - val_accuracy: 0.6267
Epoch 6/10
22/22 [==============================] - 0s 5ms/step - loss: 0.6226 - accuracy: 0.6657 - val_loss: 0.6490 - val_accuracy: 0.6267
Epoch 7/10
22/22 [==============================] - 0s 6ms/step - loss: 0.6165 - accuracy: 0.6714 - val_loss: 0.6488 - val_accuracy: 0.6133
Epoch 8/10
22/22 [==============================] - 0s 4ms/step - loss: 0.6122 - accuracy: 0.6686 - val_loss: 0.6491 - val_accuracy: 0.6133
Epoch 9/10
22/22 [==============================] - 0s 4ms/step - loss: 0.6100 - accuracy: 0.6700 - val_loss: 0.6497 - val_accuracy: 0.6133
Epoch 10/10
22/22 [==============================] - 0s 4ms/step - loss: 0.6082 - accuracy: 0.6714 - val_loss: 0.6508 - val_accuracy: 0.6267
```

Neural Network Training Overview:

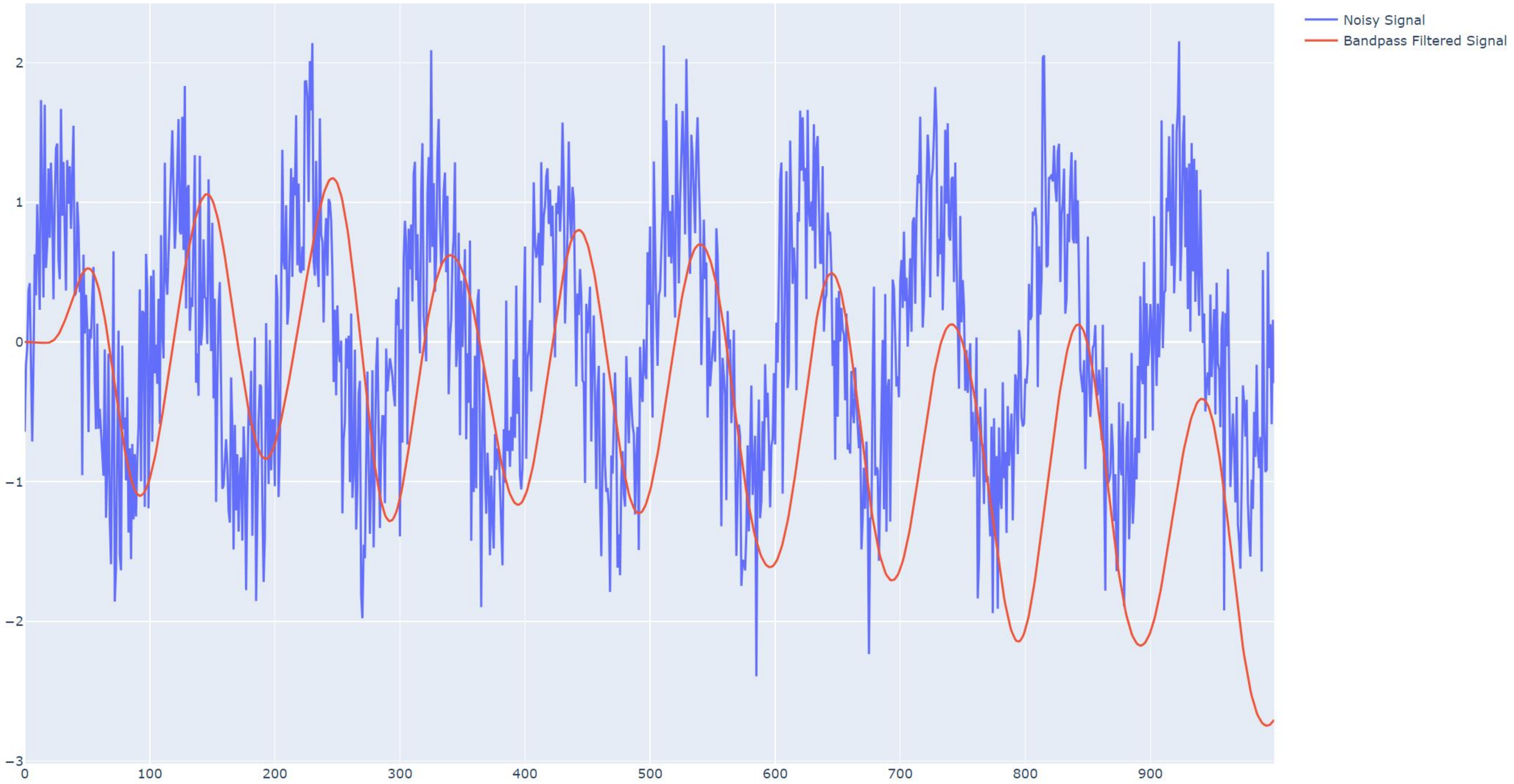
- Epoch: One full training cycle on the dataset. Model trained over 10 epochs.
- Batches: Data divided into 22 batches per epoch for training.
- Loss: Measures prediction error. Lower is better.
- loss: Training data error.
- val_loss: Validation data error.
- Accuracy: Percentage of correct predictions.
- accuracy: Training data accuracy.
- val_accuracy: Validation data accuracy.

Neural Network Training Overview Key Takeaways:

1. Training Progress: Model's loss decreases, indicating it's learning.
2. Overfitting Signs: Validation loss increases after the 6th epoch while training loss decreases. Model might be memorizing training data.
3. Accuracy Plateau: Model's validation accuracy is steady, hinting it might not generalize well to new data.

Neural Network Training Overview Next Steps :

Early Stopping: Halt training when validation doesn't improve.
Regularization: Implement techniques like dropout to prevent overfitting.
Review Model: Simplify architecture if too complex.
Adjust Learning Rate: Optimize for better learning speed and results.



Noisy Signal
Bandpass Filtered Signal
2
1
0
−1
−2
−3
0
100
200
300
400
500
600
700
800
900

Conclusion

The fusion of traditional signal processing with machine learning offers potent tools for time-series data analysis. While this demonstration highlighted a basic approach, the underlying principles can be broadened to cater to more intricate datasets and tasks. This synergy underscores the program's capability in deciphering and managing noisy signals, showcasing the transformative potential of merging classical techniques with modern algorithms.




Thank you

Mrigank Maharana

[Mk's Portfolio Page - Home (mk-maharana.web.app)](mk-maharana.web.app)

https://github.com/mkorkrish/NeuralPrograms

www.linkedin.com/in/mrigank-maharana-67a07020a