Open Seizure Detector: A User-Specific Convolutional Neural Network for Seizure Detection from Accelerometer Measurements

Graham Jones, 03/01/2026

Background

The OpenSeizureDetector seizure detection system utilises a wrist worn watch to collect accelerometer data at 25 Hz and heart rate data at 0.2 Hz. It uses a deterministic algorithm to identify seizures and raise alarms [1]. The system has a good seizure detection reliability based on user collected data of over 80% [2], but does suffer from false alarms from normal daily activities which involve repetitive movements (such as brushing teeth or washing dishes).
To help with addressing the false alarm issue, the project introduced a 'Data Sharing' system to allow users to mark alarms as either false alarms or real seizures, and the data collected is made available to researchers as the Open Seizure Database [3]. The intention is that the Open Seizure Database will be used to develop improved seizure detector algorithms that can be incorporated into the Open Seizure Detector system to improve performance and reduce false alarm rates.

This note describes the use of the Open Seizure Database to train a deep learning model that can be incorporated into the system.

Model Choice

Earlier attempts (by the author) to train a model have suffered from over-fitting which has meant that the models do not generalise well to un-seen data, resulting in poor performance. Attempts to address the over-fitting issue by simplifying the model have resulted in models which have not performed any better than the simple deterministic algorithm that is currently in use.

A recent paper lead by Spahr et.al [4] reported extremely good results for convulsive seizure detection using a relatively small number of seizures for training and testing, by using a deep (14 layer) 1D convolutional neural network. The extremely good results were somewhat surprising based on our previous experience given the complexity of the model and the small number of seizures used for training and testing. This implies that the use of ReLU activation and batch normalisation between the layers, and saving the model based on both True Positive Rate (TPR, Sensitivity) and False Positive Rate (FPR, FAR) was avoiding the over-training that we had experienced previously.

Therefore a model of similar size and depth to that described in [4] was constructed using pyTorch, and incorporated into the OpenSeizureDatabse nnTraining toolchain [5]. The model accepts a 1D vector of 30 seconds of data (=30 x 25 Hz = 750 samples), where the data is the vector magnitude of the acceleration in G. It produces a binary classification output (seizure or non-seizure)

Data Selection and Processing

For this initial investigation of the model only a subset of Version 1.9 of the OpenSeizureDatabase was used. It was decided to use only a single user, who has contributed a significant proportion of the seizure events in the database, and also has generally included comments describing the seizure to allow us to assess the implications of any detection failures. User '39' was selected, which resulted in 102 seizure events being used, of which 37 were labelled as 'Tonic-Clonic' seizures. 38 of the seizures were recorded using the PineTime [6] watch, with the others a Garmin watch.

The dataset also included 326 'Normal Daily Activity' events (~40 hours), and around 240 'false alarm' events which had generated an alarm from the Open Seizure Detector algorithm and had

been specifically labelled (those marked 'unknown' were ignored in case they were a genuine seizure that had not been noticed).

The data was processed using the 'runSequence.py' script from the Open Seizure Database tools [5], which included several processing steps:

- Filter the data for the required user, and exclude 'unknown' events.
- Split the data using nested k-fold validation (3 outer folds and either 3 or 5 inner folds)
- Flatten the .json files into .csv files that included one 5 second 'data point' per row (=125 samples at 25 Hz).
- (the 'extract features' tool was called, but only the simple acceleration measurements were included, no calculated features were added).
- Some validation of the data was carried out to check for duplicates (overlapping 5 second sequences) or missing data. Missing data was replaced with zeroes, but later processing filters out data points with very low movement, so these zero data points have no effect.
- Either noise or phase augmentation was applied
  - Noise augmentation replaces each datapoint with several copies, with gaussian noise added to each measurement
  - Phase augmentation produces several copies of each data point, starting at a different position in the acceleration data window.
- Before training the data for each event was concatenated into a continuous acceleration measurement sequence, and 30 second windows selected from it for training or testing.

Training

The runSequence.py toolchain [5] is configured using a text configuration file to specify the data to be selected and filtered, the augmentation to be applied and model parameters. The k-fold structure is specified on the command line.

The detail of the model is included in a separate python file that includes functions to build the model and process the data from the 5 second acceleration data windows in the database into the format required by the model. The 14 layer 1D CNN model is defined in 'deepEpiCnnModel_torch.py' [5].

The model learning rate was varied during training using a method similar to that described in [4], but it was found that it was not necessary to train over 50000 samples as described in the paper because the requirements for saving an improved model were not being met after about 20000 samples – therefore the model was only trained over 2000 samples. The model was saved if:

- Both TPR (Sensitivity) and FPR improved, or
- FPR improved and TPR did not fall by more than 20% AND TPR remained above 0.25.

The TPR threshold of 0.25 was selected based on observation that a datapoint level TPR of 0.25 tended to give an event beased TPR of over 80%, which is comparable to the performance achieved by the deterministic OpenSeizureDetector algorithm.

The model was trained over several different runs with different model parameters. These included the level of augmentation applied, whether to use 3 or 5 inner folds in the k-fold validation.

On completion of the training run the best model of the 3 or 5 inner folds was assessed against the independent test dataset kept back in the outer fold. The TPR and FPR values for the three outer folds were then averaged and the standard deviation calculated to give a measure of the variability (and hence stability) of the resulting models.

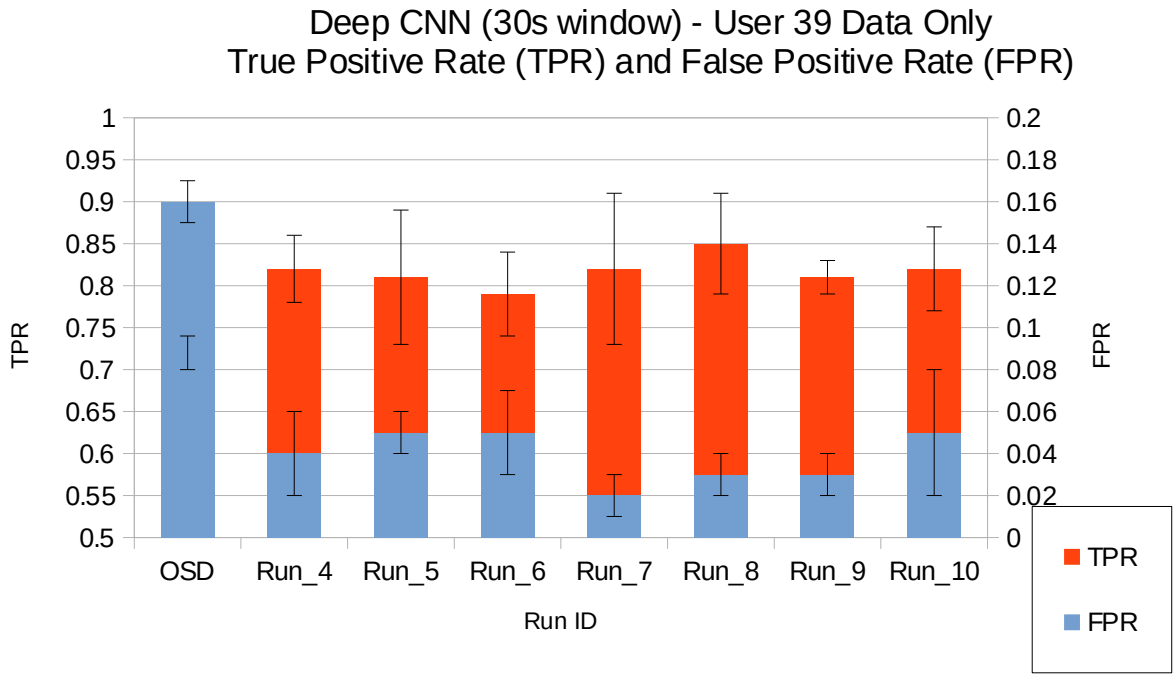
Results

Seven training runs were performed (labelled Run4 to Run10) as shown in Table 1 below. The numerical values are plotted in Figure 1 for easier comparison.

Table 1: Results of Training Runs with Various Parameters Varied

Run	Description	TPR	Stdev (TPR)	FPR	Stdev (FPR)
OSD	Deterministic OpenSeizureDetector Method for comparison	0.72	0.02	0.16	0.01
Run_4	3x3 kFold, phaseAug 25 step (1s), noiseAug 5 x 30 mg	0.82	0.04	0.04	0.02
Run_5	3x5 kFold, phaseAug 5 step (0.2s), noiseAug 5 x 30 mg	0.81	0.08	0.05	0.01
Run_6	3x3 kFold, phaseAug 5, noiseAug 5 x 30 mg	0.79	0.05	0.05	0.02
Run_7	3x3 kFold, No Augmentation	0.82	0.09	0.02	0.01
Run_8	3x3 kFold, phaseAug None, noiseAug 5 x 30 mg	0.85	0.06	0.03	0.01
Run_9	**3x3 kFold, phaseAug None , noiseAug 10 x 30 mg**	**0.81**	**0.02**	**0.03**	**0.01**
Run_10	3x3 kFold, phaseAug 25 step (1s), noiseAug None	0.82	0.05	0.05	0.03

Figure 1: Results of Training Runs with Various Parameters Varied

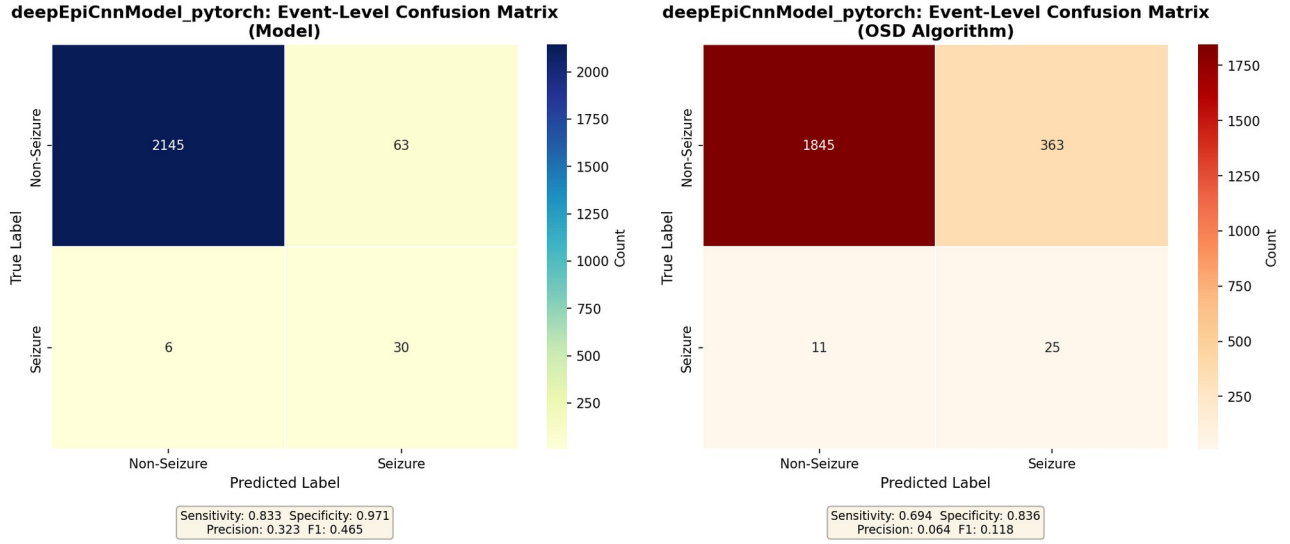


It can be seen from the data that the False Positive Rate (FPR) is significantly lower for the CNN model runs than for the determinstic Open Seizure Detector algorithm (labelled 'OSD' in Figure 1). The True Positive Rate also shows an improvement increasing from around 72% to around 82% depending on the model parameters.

When the uncertainty in the TPR results is taken into account (shown by the 1 standard deviation error bars in Figure 1) it can be seen that all 7 CNN model runs TPR lie within the same uncertainty band, but that Run 9 has the lowest uncertinty in TPR and FPR. This implies that this model is showing the best generalisation to unseen data.

Further examination of the best Run 9 model (outer fold 2, inner fold 2), shows that we have an event based confusion matrix as shown in Figure 2, where it is clear that the number of false positives is significantly reduced compared to the OpenSeizureDetector deterministic algorithm.

Figure 2: Run 9 Confusion Matrix (with comparison against the deterministic OSD Algorithm



It should be noted that most of the 6 false negatives (out of 36 in the test dataset) from the testing of the model were not serious seizures that we expect to detect. For example the descriptions include 'no movement of watch arm', 'possibly prelude to this morning's other ones', 'very short, not sure if it was a real seizure'. Only one (event 151658) was labeled as 'partial, left arm flapping' which we might have expected to detect. None of the 6 undetected seizures were classified as 'Tonic-Clonic' seizures. If the 5 smaller seizures are discounted, the detection reliability would be considered to be very high.

Despite these good results from purely independent test data, examination of the training data shows that it is quite noisy with the sensitivity (TPR) not improving significantly during training, as shown in Figure 3 below. The more conventional accuracy/loss plot is shown in Figure 4. The noise in the validation results during training may be a result of the learning rate being too high, so future processing will experiment with a reduced learning rate, and possibly a single parameter for optimisation and model selection (such as maximising TPR/FPR), rather than looking for reducing FPR with a TPR cross-check.

Figure 3: Variation of TPR and FPR during model training (run 9)

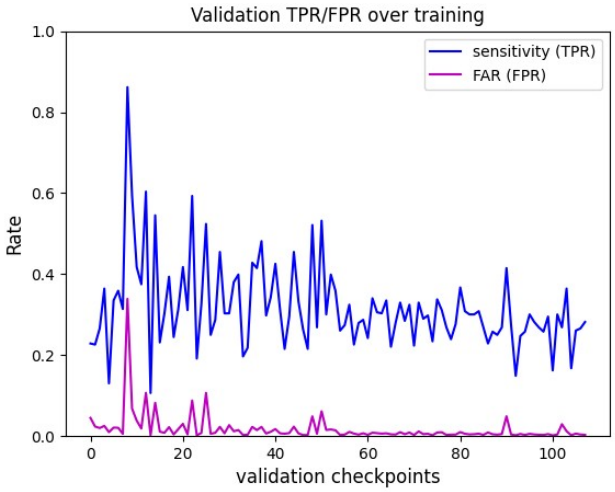
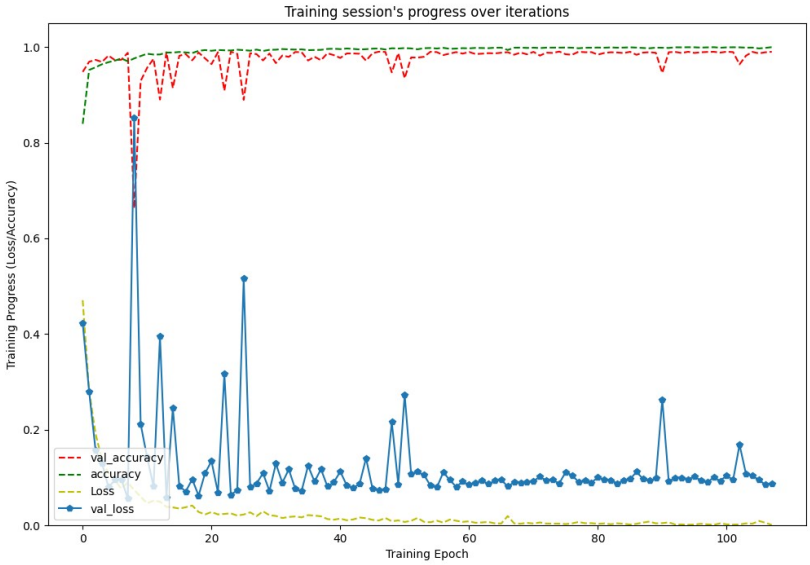


Figure 4: Accuracy / Loss Plot during Training (Run 9)



Conclusion

Based on the low variation between outer k-fold results for Run 9, and the comparable TPR and FPR to the other runs, it is proposed to incorporate the best Run 9 model (10x noise augmentation and no phase augmentation) into the OpenSeizureDetector android app for real-world testing.

Future training will consider reducing the learning rate to try to reduce the noisiness of the validation results during training which should improve the model stability and reproducibility.

References

[1] OpenSeizureDetector: Seizure Detection Algorithm; https://www.openseizuredetector.org.uk/?page_id=455

[2] OpenSeizureDetector: PineTime performance report, July 2025; https://openseizuredetector.github.io/pages-user/PineTime_Performance_Report.pdf.

[3] Pordoy, J; "The Open Seizure Database", 2023; https://www.techrxiv.org/users/692829/articles/682772-the-open-seizure-database-facilitating-research-into-non-eeg-seizure-detection.

[4] Spahr et. al. "Deep learning–based detection of generalized convulsive seizures using a wrist-worn accelerometer"; Epilepsia Volume 66, Issue S3; September 2025; https://onlinelibrary.wiley.com/doi/full/10.1111/epi.18406.

[5] Open Seizure Database nnTraining2 toolchain; https://github.com/OpenSeizureDetector/OpenSeizureDatabase/tree/main/user_tools/nnTraining2.

[6] Open Seizure Detector: PineTime Watch; https://openseizuredetector.github.io/pages-user/pine-time-installation.html

[7] Open Seizure Detector: Garmin Seizure Detector; https://www.openseizuredetector.org.uk/?page_id=1128.