CS 74: Project Final Writeup
Github

Epileptic Dogs: Advanced Seizure Prediction

Taylor Neely & Jack Terwilliger
January 15, 2015

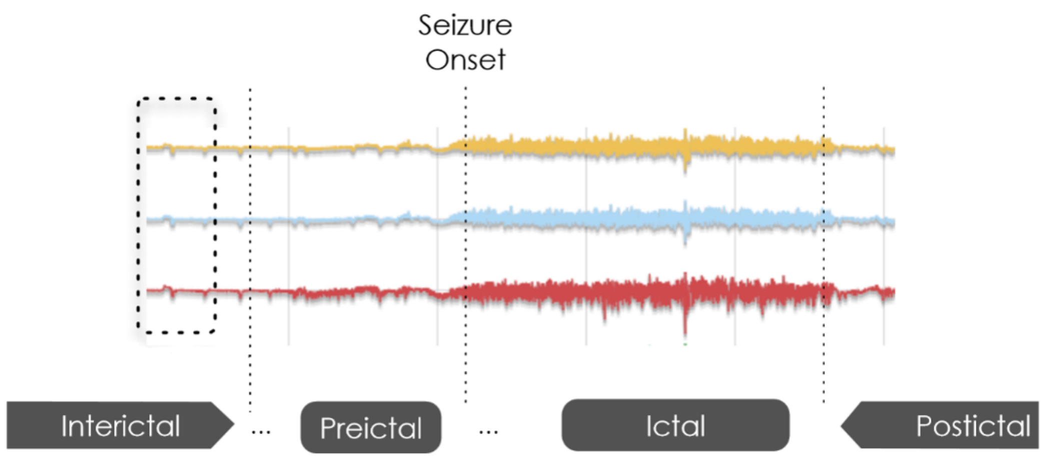
Introduction

Epilepsy is a neurological disorder defined by random, spontaneous seizures. Those afflicted with epilepsy suffer from persistent stress and anxiety due to the nature of the seizures.

> "It's the variability that really makes it so stressful. You never know when it is going to be chaos again and you'll have one. Just because this morning is terrific doesn't mean tonight is going to be terrific, either behavior-wise, medication-wise, or any otherwise. So it is the unpredictability of it that is really nerve-racking to live with." - Murray, Seizure 2 (1993) 167-178

While medication and surgery can, to some degree, alleviate the symptoms of epilepsy, these treatments fail to accommodate all patients. Of the 1% of the world's population suffering from epilepsy, treatment is ineffective on about 30% of these people [1]. Unforeseen seizures can put epileptics at risk during everday activities such as biking and driving.

We hope to delevop an accurate seizure prediction system to improve the daily life and reduce anxiety of those with epilepsy. Electroencephalography (EEG) is used to record brain waves in order to characterize brain states. Using this data, we will use a neural network to classify the brain into two states: interictal (between seizures) and preictal (before seizure). By predicting a preictal state, we can successfuly warn of an oncoming seizure.



Data

All of our data is provided by the American Epilepsy Society through Kaggle.com. It consists of 5 dogs and 2 human patients. For our project we will only look at the dog data, as it is more managable is size and also offers more diversity. Each dog has about 500-1500 training segments associated with it. Each segment is 10 minutes long, recorded at a frequency of 400Hz. There are between 15 to 16 recording channels in each segment. Due to the nature of the disorder, most segments are interictal and defined by a week after and before any seizures. Preictal segments can be anywhere from 1:05 to 0:05 before onset of a seizure.

Methods

Due to the nature of EEG and human biology, it is near impossible to build a robust, reliant seizure prediction system that can function across mulitple patients. There is too much variation in the placement and condition of electrodes in EEG and in the nature of an individual's seizures to train a broad system. Therefore, we have built a flexible neural network that can be easily trained and customized to fit an individual patient's needs [2].

> "there is not any system usable by patients allowing them to predict a coming seizure and to take action to preserve his (her) safety and privacy, improving substantially his (her) social integration. This is probably because most of the researchers look for a general method and algorithm that would work for every patient. And although several authors propose methods to which they claim a high performance, the considered performance criteria is only partial, neglecting other parts of the problem that prevent them to be used in a clinical environment."
> -V. Kurkova et al. (Eds.): ICANN 2008, Part II, LNCS 5164, pp. 479-487, 2008.

Our approach is to first reduce the working size of our data by generating simple and unique features that can characterize each segment. Our project scenario is for realtime seizure prediction, so total time from features to prediction is constrained. As time passes from when data is sampled, the prediction window narrows leaving the patient less prepared in case of a seizure. Also, in a mobile device, this process should be computationally cheap. Of course, training the neural network has no time constraints, as training will not be a part of the onboard system. Generated features will be passed into a feedforward neural network trained on backpropagation to learn the appropriate weights required to predict a patient's seizure.

Features

Feature generation was a difficult process for us. Becuase the number of channels in each dog acts as a multiplier, it was important to pick methods that could reduce each channel into a scalar value. From each channelX(time*frequency) segment, we generated three main types of feautures: spectral edge, spectral power, and average power. All of these features require the time-domain EEG waves to be turned into frequency-domain[3]. To do this, we use the fast fourier transform availible in MATLAB, which is defined by:

$$X(k) = \sum_{j=1}^{N} x(j) w_N^{(j-1)(k-1)} \tag{1}$$

$$x(j) = (1/N) \sum_{k=1}^{N} X(k) w_N^{-(j-1)(k-1)} \tag{2}$$

where N is the length of the input vectors. From there we calculate the power spectrum by squaring the absolute value of the frequency-domain. The spectral edge frequency is defined as the frequency at which 75-90% of the power spectrum lies behind. In our case, we found 90% to generate a diverse range of frequency values. The signal power is simply the max power found within a certain range. Ranges are defined according to specific brain waves: theta (0-4Hz), delta (4-8Hz), alpha (8-16Hz), beta (16-32Hz), and gamma (32-100Hz)[4]. Average power is similar to signal power, as it looks at each brain wave region separately. Instead of max power, average power is simply the average power within a defined frequency range. While basic, these features provide enough information for characterization of our data. All features are normalized independently before input into the neural network (See Figure 1).

We attempted to use cross-correlation, spectral entropy, wavelet transformations, and effective correlation dimension to further characterize the data [5]. However, these methods were typically computationally expensive and of high dimmension, or, in the case of wavelets, very difficult to implement. We also tried breaking each segment into further segments, or windows, but keeping these windows within the same feature set. While easily increasing number of features, this splitting did little to increase the number of unique features. As a result, training only took more time with no improvement, and in a few cases made error worse.

NEURAL NETWORK

To analyze our features we used an artificial neural network. Our current ANN implementation is flexible to allow us to accomodate for variances in patient data such as the number of EEG channels. We have included a bias term, weight momentum, and weight decay functionality to allow our neural network to be more generalizable and flexible.

Each layer generates output as follows:
Let $\varphi(x)$ be the activation function, x^l be per-layer input, $\hat{o}^l$ be net input per neuron, W^l be layer weights, and $\hat{b^l}$ be the bias term.
output is:

$$y = \varphi(\hat{o} + \hat{b^l})$$

$$\hat{o} = x^{l-1} W^l$$

Our backpropagation implementation is as follows:
Let $\varphi'(x)$ be the activation function derivative δ be propogated error, λ be momendum, and γ be decay
Change to synapse weights in each layer is:

$$\Delta W_t^l = -\alpha \delta^l \hat{o}^l + \lambda \Delta W_{t-1}^l - \gamma \Delta W_{t-1}^l$$

$$\Delta \hat{b} = -\alpha \delta^i$$

$$d^i = \begin{cases} D^L e, & outputlayer \\ D^i W^{i+1} \delta^{i+1}, & hiddenlayer \end{cases}$$

$$e = \frac{1}{2}(\hat{y} - \hat{t})^2, \text{where t is the target output vector}$$

$$D = \begin{bmatrix} \varphi'(o_1) & 0 & \cdots & 0 \\ 0 & \varphi'(0_2) & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \varphi'(o_n) \end{bmatrix}$$

We currently use a three-layer (input, hidden, output) feedforward neural network trained using backpropagation. The number of hidden nodes is determined by following the general rule of thumb: $\#hidden = (\#input + \#output)/2$ [6]. The activation function for every level is the logistic sigmoid function. To classify ictal state, we round the output to either one or zero, thus our output is [1] for preictal state, and [0] for interictal state.

To deal with the inbalance in the size of our classes, before every epoch, we weight the learning rate, α, for each class as a percentage of the total the number of interictal and preictal segments, respectively. In choosing a proper learning rate, we trained on a single dog over 500 epochs and plotted error on varying learning rates (See Figure 2). In the end, we settled with a learning rate of 0.3.

Decay and momentum were chosen in a similar manner. Effects of momentum and decay are slight, but higher momentum allows for faster initial descent while decay acts to counterbalance and prevent movement away from minima. We opted for a momentum value of 0.85 and decay value of 0.05 based on literature [7][8], which performs slightly better than control over 500 epochs (See Figure 3).

Implementation Details

With the exception of the builtin function fft() for Fast Fourier transformations, all code for feature generation and the neural network were hand implemented. Feature generation can all be run through read_gen.m, which allows selection of raw data and desired features, outputting a matrix of features and vector of classes. Training and construction of the neural network is done through eeg_train.m, which takes training data and trials, learning rate, momentum, and decay parameters, and outputs the error for each trial as well as the trained neural network. This function can also take test data and spit out sensitivity and false positive rates (FPR) over each trial. cross_val.m provides N-fold cross validation on a given training set, and outputs sensitivity and FPR averaged over all folds.

Results

We plotted error over 200 epochs for each dog, both with and without a hidden layer. While the hidden layer performed better for dogs 1 and 4 (See Figure 5), we were surprised to find that Dog 3 had similar performance in both cases and that Dogs 2 and 5 performed better without the hidden layer (See Figure 4). This suggests that each individual is unique and validates our goal for per-subject customization in this project.

Two important measures of success for our model are sensitivity and false postitive rate (FPR). Sensitivity indicates how well our model is a classifying preictal states as preictal. We want this to be as high as possible so that no unexpected seizures will occur. FPR measures how often we classify an interictal state as preictal. While we want to minimize this to prevent unnecessary stress of patients, we prioritize maximizing sensitivity. Because cross validation over a high number of epochs is very time consuming, we were only able to generate generalized data (i.e. 1 hidden layer with half as many nodes as features and learning rate, momentum, and decay set at 0.3, 0.85 and 0.05 respectively). Figure 6 plots sensitivity and FPR for each dog after 1000 epochs. While sensitivity is reasonably high (>70%), FPR was higher than desired in a few cases. We believe this to be a consequence of our time-sensitive generalized approach, and feel that both of these numbers can be improved with further customization of the neural network on each dog.

We included results of randomized assignment using Matlab's $rand$ function as a comparison to our ANN predictions. If the output of $rand$ fell below the fraction of preictal segments, we assigned that segment as preictal. Otherwise, the segment was labeled as interictal. Results from this test were as expected, with senstivity and FPR both remaining very low (See Figure 7).

By plotting sensitivity and FPR over time, we were able to get a better idea of how our neural network trained (See Figure 8). Surprisingly, prediction after the first epoch is always all 1s (preictal). We believe this to be a consequence of the way we initialize our weights, which are randomly initialized. Although we did not have time to implement an improved weighting system, adjusting weights so that prediction begins with 0s (interictal) would most likely greatly improve training, due to the uneven class sizes.

SUMMARY

Overall, we are satisfied with our results. We were able to show that by customizing the neural network for each subject, we can significantly improve training and therefore predictions. Additionally, even using generalized parameters, our classification system is capable of producing high sensitivity while keeping false positive rates relatively low. Given more time, we feel it is feasible to reach sensitivities above 90% while keeping false positive rates below 10%. While training is time consuming, and even moreso for cross validation, we kept with our goal of keeping feature generation and class prediction well within the prediction window.

REFERENCES

[1] Thurman, DJ, et al. ILAE Commission on, Epidemiology (September 2011). "Standards for epidemiologic studies and surveillance of epilepsy." Epilepsia. 52 Suppl 7: 2-26.

[2] Howbert JJ, et al. (2014) Forecasting seizures in dogs with naturally occurring epilepsy. PLoS One 9(1):e81920.

[3] Chestnutt, Catherine. "Feature Generation of EEG Data Using Wavelet Analysis." Thesis. Texas Tech University. May, 2012.

[4] Kolev V, BaÅ§ar-Eroglu C, Aksu F, BaÅ§ar E. (1994). "EEG rhythmicities evoked by visual stimuli in three-year-old children." Int J Neurosci. 75(3-4):257-70.

[5] Maiwald, Thomas, et al. "Comparison of three nonlinear seizure prediction methods by means of the seizure prediction characteristic." Physica D: nonlinear phenomena 194.3 (2004): 357-368.

[6] Heaton, Jeff. "Introduction to Neural Networks for Java." 2nd Edition. Heaton Research, Inc. 1 October, 2008.

[7] http://en.wikibooks.org/wiki/Artificial_Neural_Networks/Neural_Network_Basics

[8] http://ann.thwien.de/index.php/Multilayer_perceptron

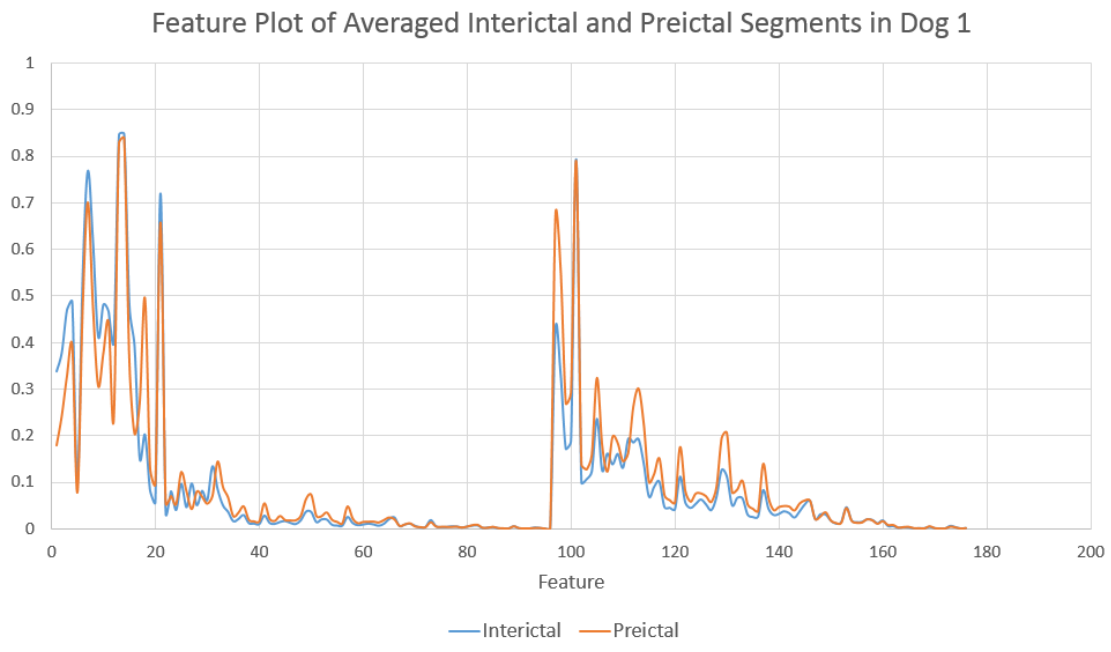


Figure 1: Plot of all features for interictal and preictal segments, averaged over all respective segments in Dog 1. First 16 features are for spectral edge on each channel. Features 17-86 are for spectral power. Features 87-176 are for average power.

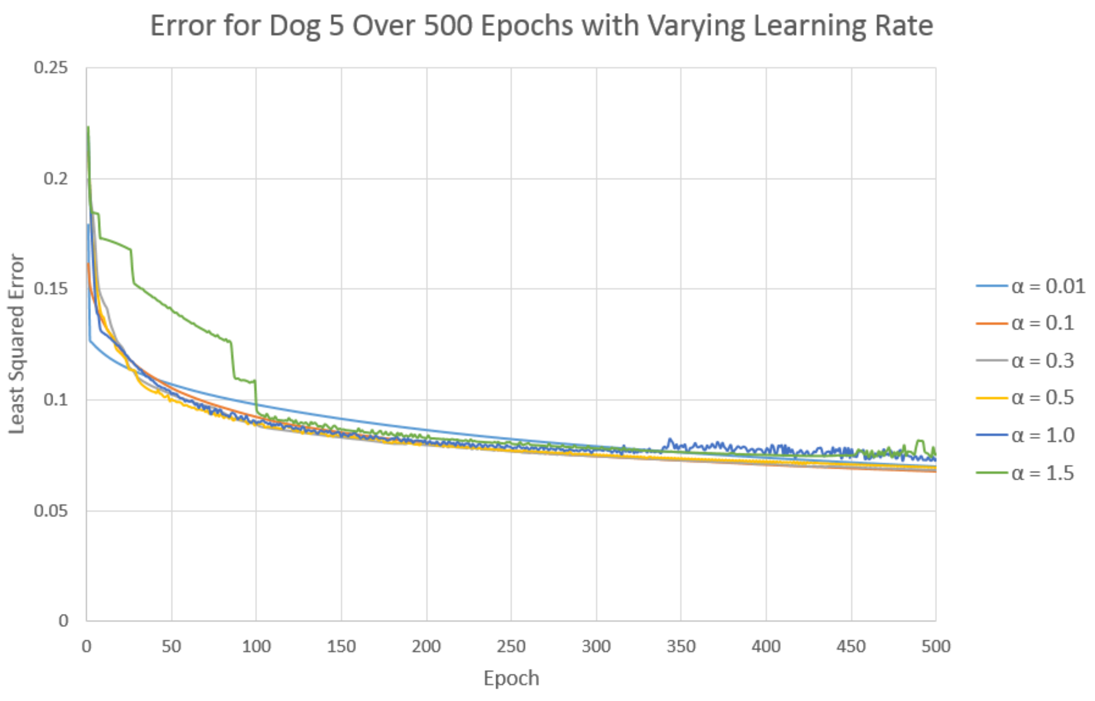


Figure 2: Plot of least squared error on Dog 5 over 500 epochs. Learning rate is varied, with 0.3 performing the best.

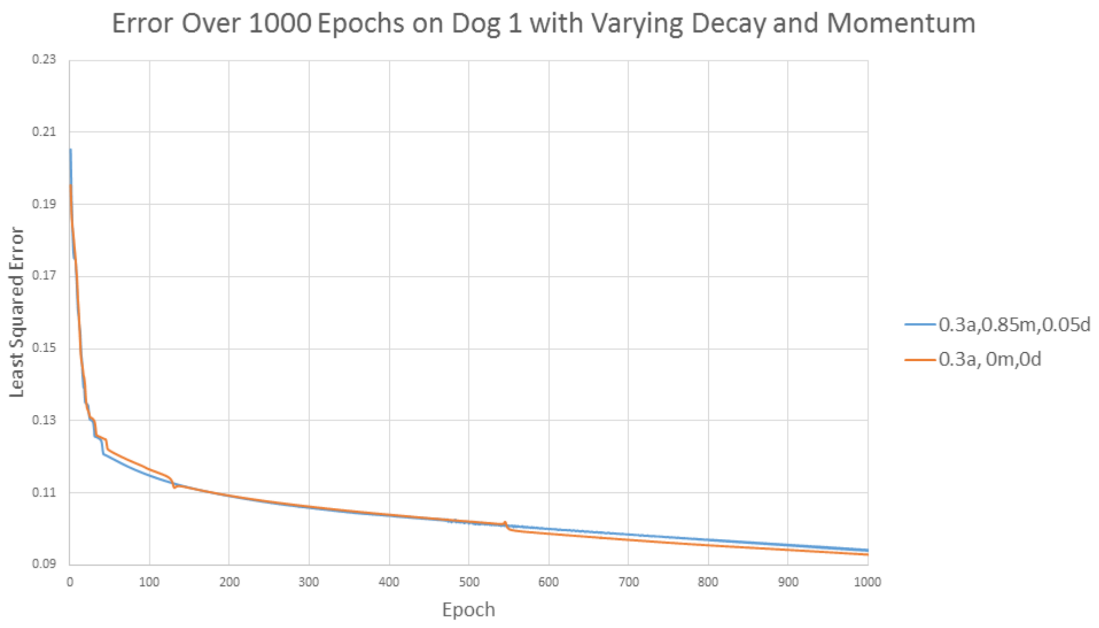


Figure 3: Plot of least squared error on Dog 1 over 1000 epochs. With learning rate fixed at 0.3, decay and momentum are varied from both 0 to 0.85 and 0.05, respectively.

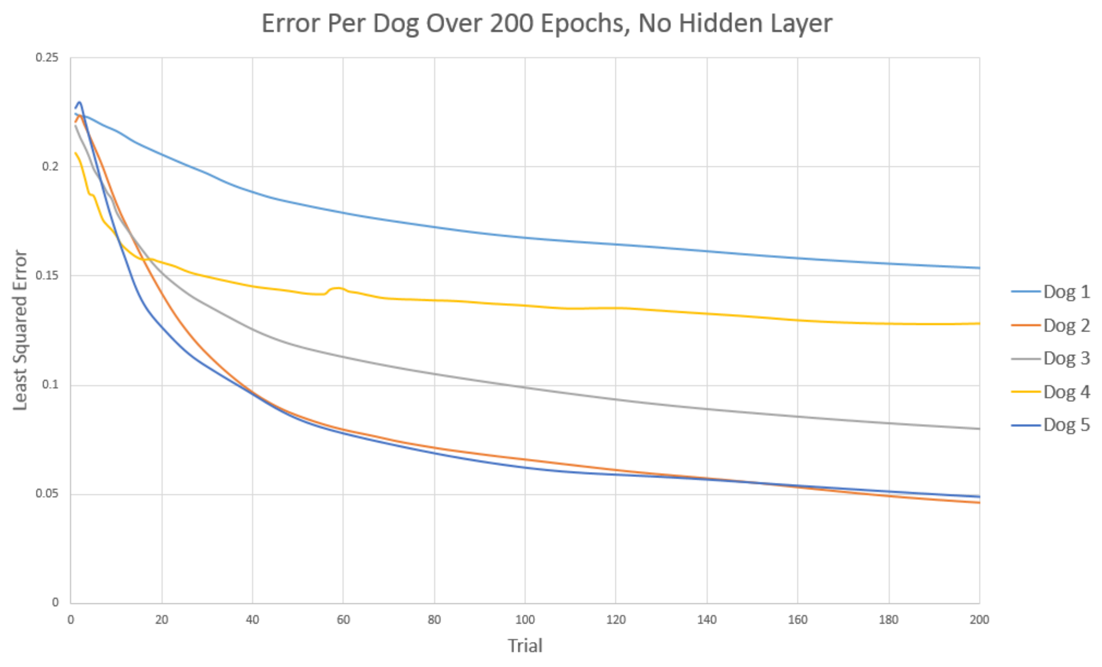


Figure 4: Plot of least squared error on all dogs over 200 epochs. With learning rate, momentum, and decay fixed at 0.3, 0.85, and 0.05 respectively. No hidden layers are used.

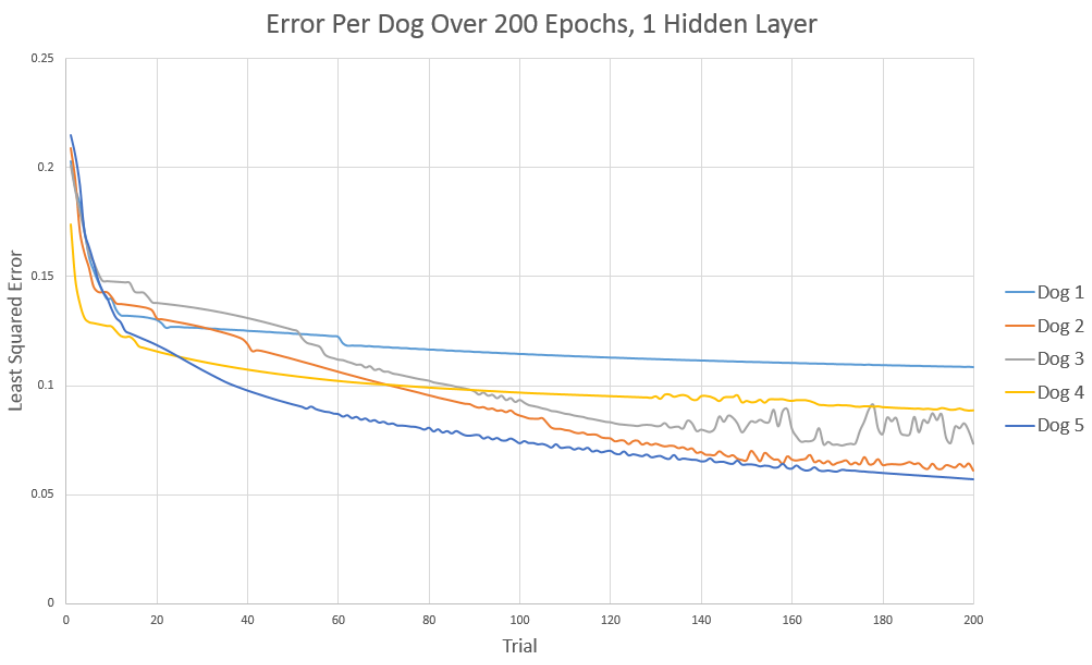


Figure 5: Plot of least squared error on all dogs over 200 epochs. With learning rate, momentum, and decay fixed at 0.3, 0.85, and 0.05 respectively. 1 hidden layer is used with the number of nodes equalling half the number of features.

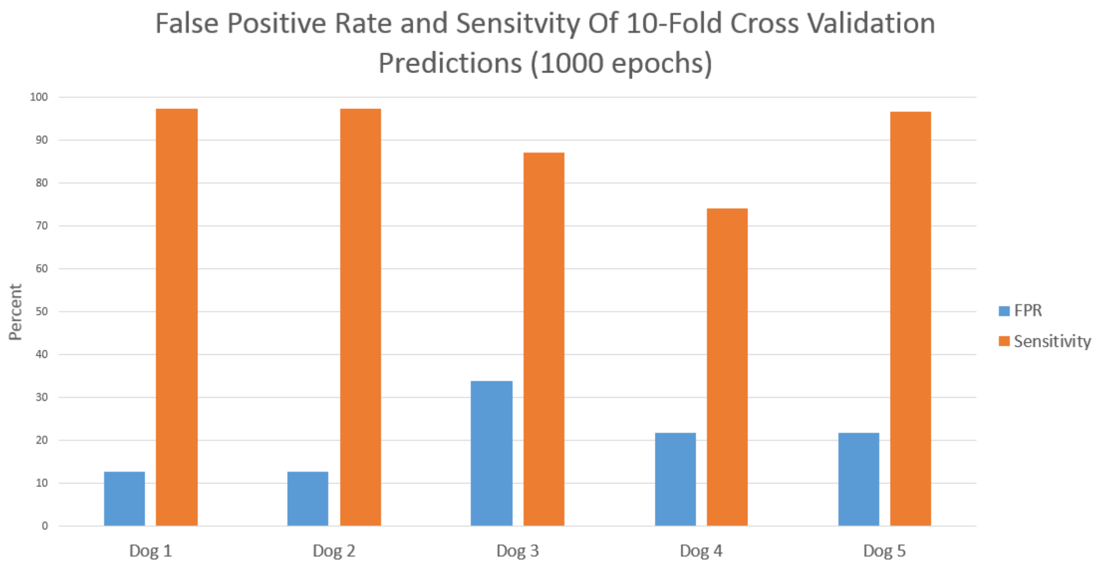


Figure 6: 10-fold cross validation resulting in final sensitivity and false postitive rates for each dog after 1000 epochs. There is 1 hidden layer and learning rate, momentum, and decay are set at 0.3, 0.85 and 0.05 respectively.

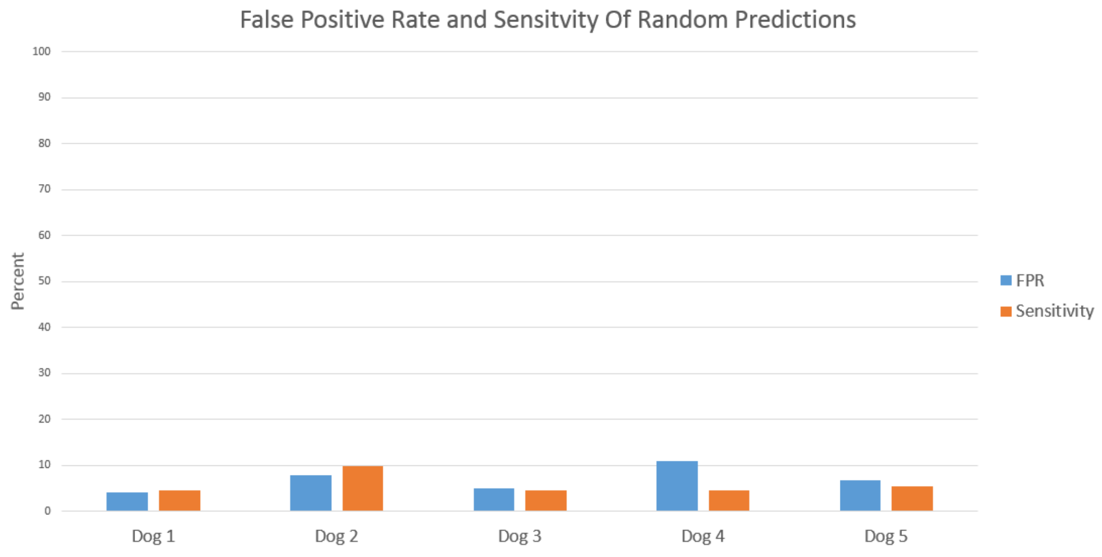


Figure 7: Random assignment of class labels using the matlab *rand* function. Results were averaged over 10 runs. Makes use of prior assumptions about class sizes

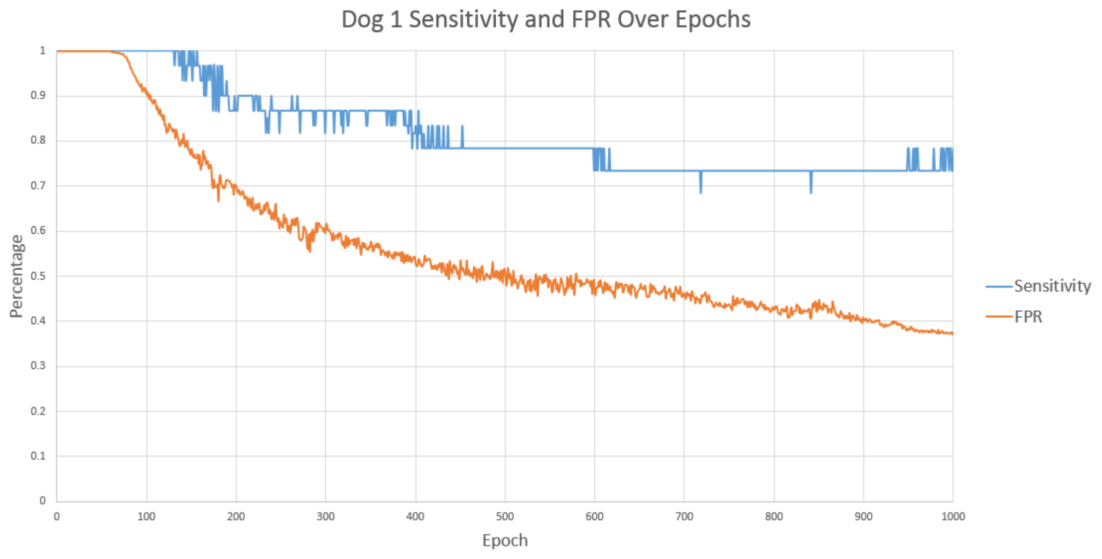


Figure 8: 10-fold cross validation resulting in sensitivity and false postitive rates for each dog over 1000 epochs. There is 1 hidden layer and learning rate, momentum, and decay are set at 0.3, 0.85 and 0.05 respectively