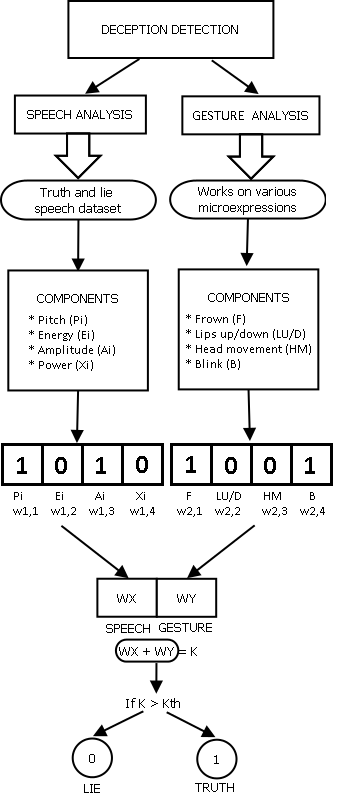
For this model, we use the real-time courtroom truth and lie dataset (2015) consisting of 153 videos of truth and lie labelled videos of a courtroom.

Our deception detection model comprises of two important parts: the speech analysis and the gesture analysis.

In the speech analysis, we consider the components such as the amplitude of voice, pitch, energy etc. and in the gesture analysis we consider various facial micro expressions such as blinking, frowning, lip movement etc. Then we perform one-hot encoding for each of these parameters to convert into 0 and 1 binary format suitable for coding.



*Figure 1: Process flow of deception detection*

For this model, we use the real-time courtroom truth and lie dataset (2015) consisting of 153 videos of truth and lie labelled videos of a courtroom.

Our deception detection model consists of two main parts: the speech analysis and the gesture analysis.

In the speech analysis, we consider the components such as the amplitude of voice, pitch, energy etc. and in the gesture analysis we consider various facial micro expressions such as blinking, frowning, lip movement etc. Then we perform one-hot encoding for each of these parameters to convert into 0 and 1 binary format suitable for coding.

We concatenate the given parameters into 2 vectors each: WX and WY, the speech vector and the gesture vector respectively. We combine these two vectors to perform a matrix ‘K’, which is then compared with the threshold matrix ‘KTH’ that tells whether the input video is a truth or lie.

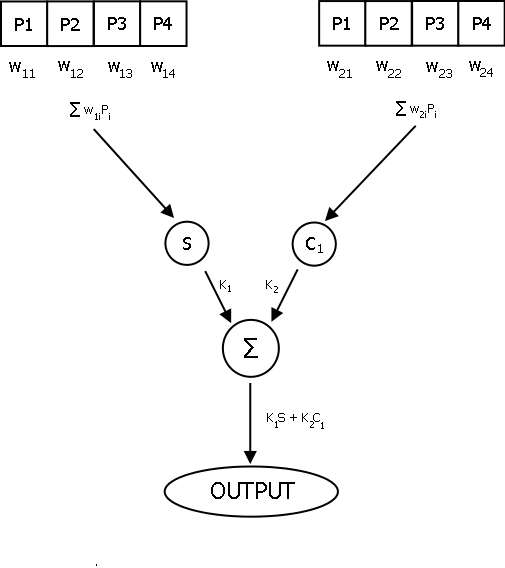
Note: We compute the hyper parameter ‘KTH’ by training the dataset.

Flow Chart Description for ASD

This model is similar to deception detection in many ways, but it also involves some more significant additions to it.

In the speech analysis, we can consider the components such as the amplitude of voice, pitch, energy etc. which are the same as in case of deception detection and in the gesture analysis we consider various facial micro expressions such as blinking, frowning, lip movement etc. Subsequently, we perform one-hot encoding for each of these parameters to convert into 0 and 1 binary format suitable for coding.

Next, the parameter values are multiplied with the weights allotted to each parameter of either the speech or gesture analysis. The speech parameters are given weights as ‘w1,i’ and gesture parameters are given as ‘w2,i’. For allotting weight to the parameters, we propose a weight training algorithm that would be using a neural network with zero bias to find the optimal value of the weight using iterations of forward and backward propagation.



*Figure 2: Process flow for autism detection*

**Weight allocation algorithm using Neural Networks**

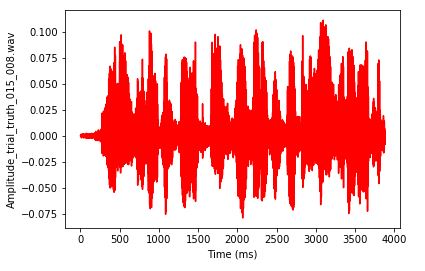
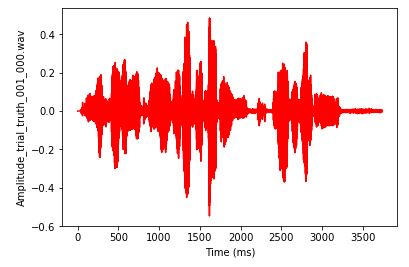
We concatenate the given parameters into 2 vectors each: S and C1, the speech vector and the gesture vector respectively. We combine these two vectors to perform a matrix ‘∑’, which is then compared with the threshold matrix which helps in identifying whether the video snippet consists of a patient suffering from ASD. We use the Adaline neural network to train the weight allocation process used in both gesture and speech analysis. The Adaptive Linear Neural Net has been shown as follows:

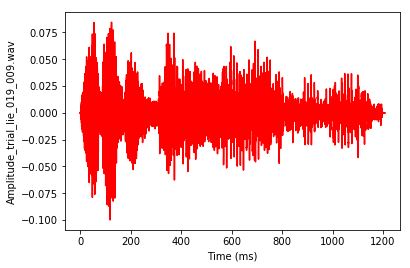
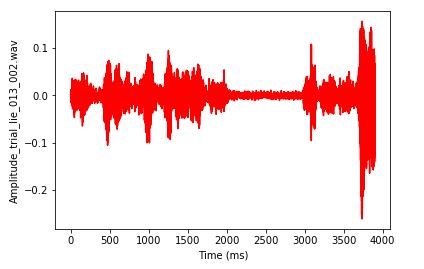


*Figure 3: Weight allocation framework*

Standard Deviations within parameters

We take mean of each parameter measurement for our sample subjects and compare it with our original measurements, in doing so we form a method for measuring standard deviation of the error. The Standard deviations form a Gaussian distribution.



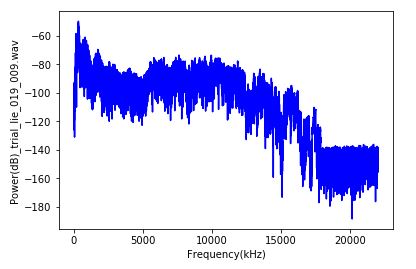
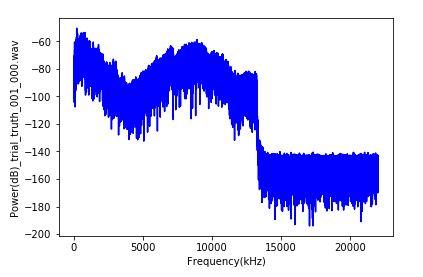
(a) (b) 

(c) (d)

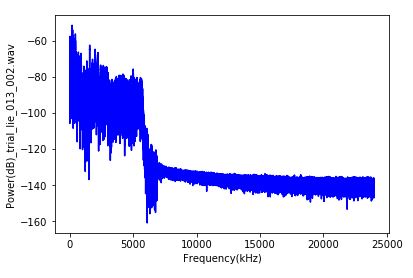
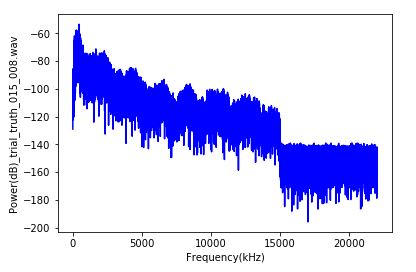
*Figure 4: Amplitude vs. frequency curve: (a) and (b) represents the amplitude based on truth samples. (c) and (d) represents the amplitude of lie samples*

The standard deviation of the sample mean is equivalent to the standard deviation of the error in the sample mean with respect to the true mean, since the sample mean is an unbiased estimator. Next, we calculated the following parameters we mentioned above on several courtroom truth and lie videos. These were videos of length ranging from 4 minutes to 15 minutes. We sampled these videos into short clips for our analysis using Python libraries. Next, we trained the model parameters on 500 video samples and calculated the mean of means.

The figures 4 and figure 5 show the plot of amplitude vs. frequency and power (in dB) vs. frequency for both the truth and lie samples. As it can be observed from the plot, the lie samples vary significantly in amplitude as well as in power as compared to the truth counterparts.



(a) (b)



(c) (d)

*Figure 5: Power (in dB) vs. frequency: (a) and (b) represents the power based on truth samples. (c) and (d) represents the power of lie samples.*

Digital association and weighing

Having obtained the parameters, next step is digital association.

We represent our training videos in a binary vector form, thus in a way we digitize our training videos which help us in other formulations, and we describe this process with a term called Digital Association. Here is a snippet of how this process is represented:



*Figure 6: Matrix with truth and lie data set for digital association*

**Note:** As an improvement over the previous model, we have incorporated a supervised learning based Adaline neural network to train our weight allocation mechanism. All the allotted weights have been trained under the weight training algorithm mentioned in the previous section of the literature.

Combining speech and gesture

Zero bias

***G***

**Decision Block**

*Output*

*W1*

**∑**

*W2*

***S***

*Figure 7: Neural network model for deception detection*

Where, *G* = Gesture analysis final value

*S* = Speech analysis final value

*W1*= Weight assigned to gesture value

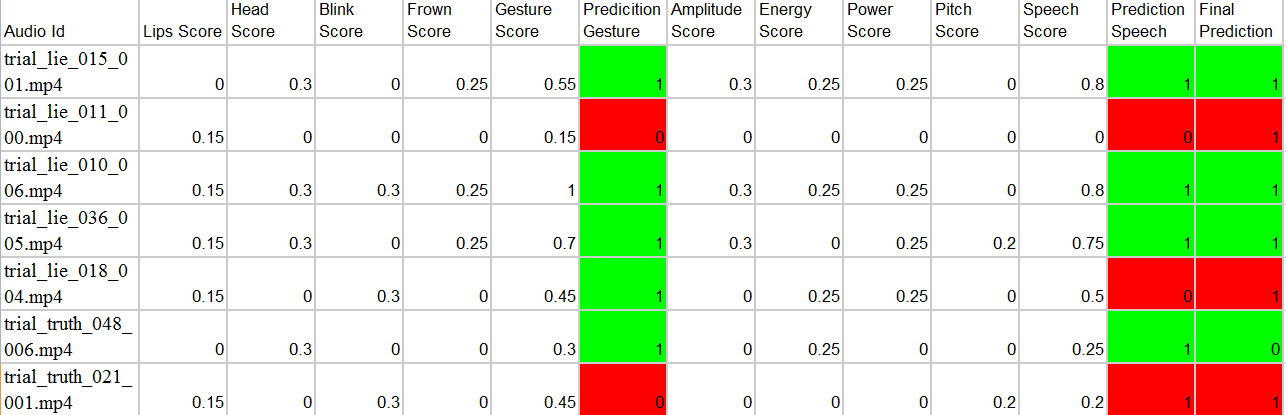
*W2* =Weight assigned to speech value

and, W1 + W2 = 1.

Using the above MP neuron we can compute the final output value, the output value is compared to the threshold, if it is found greater than the threshold we term it a lie else we classify it as a truth case. The comparison to the threshold is done within a decision block which compares the output value with a pre-defined threshold value.

**Results and Conclusion**

Deception Detection

****

*Figure 8: Final predictions for deception detection*

Upon application of the test data set values on our defined model we finally obtain prediction results as shown in table 7.1. The red color indicates error in prediction whereas green indicates that prediction matches the truth or reality.

Table 7.1 shows a sample of 10 videos over which our predictive algorithm has been applied to showcase the accuracy obtained by our model.

We have trained our model on a sample of **146 lie** video snippets along with **114 truth** video snippets. The testing has been done on **294 lie and 230 truth** video snippets.

The accuracy obtained upon testing is: **67.07%**