

# Speech Emotion Recognition System using both Spectral and Prosodic Features

Nevin Augustine<sup>1</sup>, C.R. Srinivasan<sup>2</sup> and Kevin Richards<sup>3</sup>

<sup>1,3</sup>MIT Manipal Manipal University, Karnataka

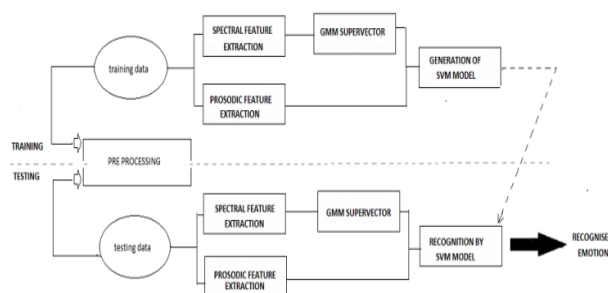
<sup>3</sup>Dept. of Electronics and communication Engg. College of Engineering Trivandrum

E-mail: <sup>1</sup>nevin.augustine@manipal.edu, <sup>2</sup>cr.srinivasan@manipal.edu

**Abstract**—In this paper, we propose an emotion recognition system from speech signal using both spectral and prosodic features. Most traditional systems have focused on spectral features or prosodic features. Since both the spectral and the prosodic features contain emotion information, it is believed that combining spectral features and prosodic features will improve the performance of the emotion recognition system. Therefore, we propose to use both spectral and prosodic features. For spectral features, a GMM super vector based SVM is applied. For prosodic features, a set of prosodic features that are clearly correlated with speech emotional states and SVM is also used for emotion recognition. The combination of both spectral features and prosodic features is done and an SVM is trained using the combined feature vector. The emotion recognition accuracy of our experiments allow us to explain which features carry the most emotional information and why. It also allows us to develop criteria to class emotions together. Using these techniques we achieved high emotion recognition accuracy.

## 1. INTRODUCTION

Recognition of emotion of a person from his speech signals plays important role in many applications. Although emotion detection from speech is a relatively new field of research, it has many potential applications. In human-computer or human-human interaction systems, emotion recognition systems could provide users with improved services by being adaptive to their emotions. In virtual worlds, emotion recognition could help simulate more realistic interaction. The body of work on detecting emotion in speech is quite limited. The emotion recognition accuracy of our experiments allow us to explain which features carry the most emotional information and why. It also allows us to develop criteria to class emotions together. Using these techniques we achieved high emotion recognition accuracy. The work aims to detect emotions from speech in real time. Five emotions are taken into consideration for this project namely, angry, happy, normal, sad and surprise. A layout of the proposed system is as shown below:



## 2. FEATURES OF A SPEECH SIGNAL:

For effective classification and hence effective result, a feature that can easily distinguish between the emotions is needed. Speech features can be mainly divided into two – spectral and prosodic. In the proposed work we considered both spectral and prosodic features as a combined study has not been done before. The literature review mainly pointed to features like MFCC, Rasta PLP, LPC etc. Out of many the following features were selected and studied: MFCC, Rasta PLP, LPC, Pitch, Loudness, Moments.

## 3. COLLECTION OF EMOTIONAL DATA

The performance of an emotion classifier relies heavily on the quality of the database used for training and testing and its similarity to real world samples (generalization). Speech data used for testing emotion recognition can be grouped under three categories depending on the way the speech signal is captured. The first method uses actors to record utterances, where each utterance is spoken with multiple feigned emotions. The speakers are usually given the time to imagine themselves into a specific situation before speaking. The data used for this work comes from real time recording from normal people.

The audio recordings and corresponding transcripts were collected with proper care as to reduce to zero the effect due to noise and other attenuations. Each utterance file represents one utterance by one speaker expressing one emotion. The recordings consist of normal speakers reading a series of utterances spanning five emotional categories under study. The utterances were mainly numbers like “two fifty three”. Numbers were chosen because they are neutral in nature and hence emotions can be expressed easily. Along with numbers a lot of emotional sentences like “oh my god”, “what are you doing” etc. are used. The maximum duration was fixed around 3 seconds. A total of 20 speakers which included both males and females gave their utterances as samples. The number of utterances that belong to each emotion category is 200 thus bringing the total number of utterances to 1000. The recordings were recorded with a sampling rate of 44100Hz and encoded in 16 bit- mono format.

#### 4. FEATURE EXTRACTION

Feature extraction is the process of calculating a compact parametric representation of speech signal features which are relevant for speech recognition. In speaker independent speech recognition, a premium is placed on extracting features that are somewhat invariant to changes in the speaker. So feature extraction involves analysis of speech signal. Broadly the feature extraction techniques are classified as temporal analysis and spectral analysis technique. In temporal analysis the speech waveform itself is used for analysis. In spectral analysis spectral representation of speech signal is used for analysis. Features are primary indicator of speaker’s emotional state. Here in feature extraction process some features are extracted like Mel Frequency Cepstral Coefficient (MFCC), pitch, PLP, RASTA-PLP, loudness etc. Feature extraction process can be divided into two steps: spectral feature extraction and prosodic feature extraction.

##### 4.1 Spectral feature extraction

###### MFCC

MFCCs are the most widely used spectral representation of speech in many applications, including speech and speaker recognition. Statistics relating to MFCCs also carry emotional information. For this project, MFCC is extracted from the speech waveform using a modified algorithm for MFCC calculation implemented in the AUDITORY toolbox for speech processing in MATLAB as function `melcepst()`. For each 25ms frame of speech, thirteen standard MFCC parameters are calculated by taking the absolute value of the STFT, warping it to a Mel frequency scale, taking the DCT of the log-Mel spectrum and returning the first 13 components. The 13 dimensional MFCC vector forms the first thirteen components of the feature vector.

###### Linear predictive coding

LPC coefficients are extracted in MATLAB using function `proclpc()`. This function is inbuilt in the AUDITORY toolbox

for speech processing in MATLAB. The coefficients are by default 13 dimensional. These 13 dimensional LPCC is then appended to the MFCC feature vector at the end. Now we have a 26 dimensional feature vector

###### RASTA-PLP

Another popular speech feature representation is known as RASTA-PLP, an acronym for Relative Spectral Transform - Perceptual Linear Prediction. PLP was originally proposed by Hynek Hermansky as a way of warping spectra to minimize the differences between speakers while preserving the important speech information. RASTA is a separate technique that applies a band-pass filter to the energy in each frequency sub-band in order to smooth over short-term noise variations and to remove any constant offset resulting from static spectral coloration in the speech channel. RASTA-PLP coefficients are extracted using the `rastaplp()` function in auditory toolbox. The model order is set as 13. So there are 13 coefficients in all. These coefficients are appended to the end of the present feature vector.

###### GMM Supervector generation

A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system, such as vocal-tract related spectral features in a speaker recognition system. GMM parameters are estimated from training data using the iterative Expectation Maximization (EM) algorithm or Maximum A Posteriori (MAP) estimation from a well-trained prior model.

##### 4.2 Prosodic feature extraction

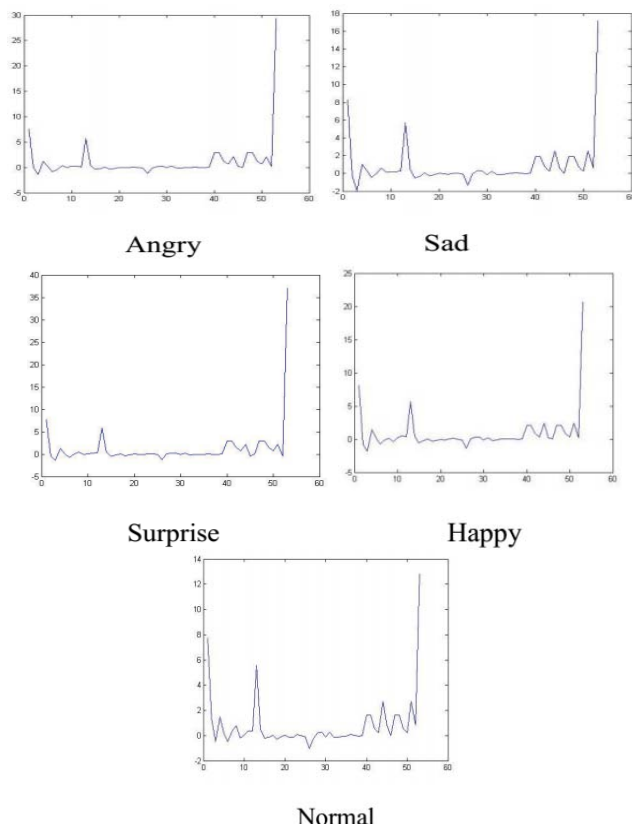
###### PITCH

Statistics related to pitch conveys considerable information about emotional status. For this project, pitch is extracted from the speech waveform using a modified version of the RAPT algorithm for pitch tracking implemented in the VOICEBOX toolbox. Using a frame length of 50ms, the pitch for each frame was calculated and placed in a vector to correspond to that frame. The various statistical features are extracted from the pitch tracked from the samples. We use minimum value, maximum value, range and the moments- mean, variance, skewness and kurtosis. We hence get a 7 dimensional feature vector which is appended to the end of the 39 dimensional supervector obtained from the GMM.

###### Loudness

Loudness is extracted from the samples using DIN45631 implementation of loudness model in MATLAB. The function `loudness()` returns loudness for each frame length of 50ms and also one single specific loudness value. Now the same minimum value, maximum value, range and the moments-

mean, variance, skewness and kurtosis statistical features are used to model the loudness vector. Hence we get a 8 dimensional feature vector which is appended to the already obtained 46 dimensional feature vector to obtain the final 54 dimensional feature vector. This vector can now be given as input to the SVM.



The figures above shows the feature vector plots for each of the five emotions of a single sample. It can be seen that they show considerable variations. Next step is to group together the samples of each emotional category and make an instance matrix that can be fed as input to LIBSVM. Thus we get a 800 x 54 matrix as the instance matrix for training and 200 x 54 matrix for testing. This completes the feature extraction process.

## 5. SVM CLASSIFIER

Classification includes a broad range of decision-theoretic approaches to the identification. For the classifier to properly classify the emotions it should be trained properly. The training involves the following steps: 1. Collection of corpus emotion data 2. Feature extraction 3. Preparation of database 4. Training Classification analyses the numerical properties of various speech features and organizes data into categories. Classification algorithms typically employ two phases of processing: training and testing. In the initial training phase, characteristic properties of typical speech features are isolated

and, based on these, a unique description of each classification category, i.e. training class, is created. In the subsequent testing phase, these feature-space partitions are used to classify speech features.

The description of training classes is an extremely important component of the classification process. In supervised classification, statistical processes based on an a priori knowledge of probability distribution functions or distribution-free processes can be used to extract class descriptors. Unsupervised classification relies on clustering algorithms to automatically segment the training data into prototype classes. In either case, the motivating criteria for constructing training classes are that they are: 1. Independent, i.e. a change in the description of one training class should not change the value of another, 2. Discriminatory, i.e. different speech features should have significantly different descriptions, and 3. Reliable, all speech features within a training group should share the common definitive descriptions of that group. A convenient way of building a parametric description of this sort is via a feature vector, where  $n$  is the number of attributes which describe each speech feature and training class. This representation allows us to consider each speech feature as occupying a point, and each training class as occupying a sub-space (i.e. a representative point surrounded by some spread, or deviation), within the  $n$ -dimensional classification space. Viewed as such, the classification problem is that of determining to which sub-space class each feature vector belongs.

Support vector machines (SVMs) are a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The standard SVM is a non-probabilistic binary linear classifier, i.e. it predicts, for each given input, which of two possible classes the input is a member of. Since an SVM is a classifier, then given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other.

## 6. CLASSIFIER TRAINING AND CLASSIFICATION

Classification includes a broad range of decision-theoretic approaches to the identification. All classification algorithms are based on the assumption that the speech in question depicts one or more features and that each of these features belongs to one of several distinct and exclusive classes. The classes may be specified a priori by an analyst (as in supervised classification) or automatically clustered (i.e. as in unsupervised classification) into sets of prototype classes, where the analyst merely specifies the number of desired categories. For the classifier to properly classify the emotions it should be trained properly. The training involves the following steps: 5. Collection of corpus emotion data 6. Feature extraction 7. Preparation of database 8. Training Classification analyses the numerical properties of various

speech features and organizes data into categories. Classification algorithms typically employ two phases of processing: training and testing. In the initial training phase, characteristic properties of typical speech features are isolated and, based on these, a unique description of each classification category, i.e. training class, is created. In the subsequent testing phase, these feature-space partitions are used to classify speech features. The description of training classes is an extremely important component of the classification process. In supervised classification, statistical processes based on an a priori knowledge of probability distribution functions or distribution-free processes can be used to extract class descriptors. Unsupervised classification relies on clustering algorithms to automatically segment the training data into prototype classes. In either case, the motivating criteria for constructing training classes are that they are: 4. Independent, i.e. a change in the description of one training class should not change the value of another, 5. Discriminatory, i.e. different speech features should have significantly different descriptions, and 28 6. Reliable, all speech features within a training group should share the common definitive descriptions of that group. A convenient way of building a parametric description of this sort is via a feature vector, where  $n$  is the number of attributes which describe each speech feature and training class. This representation allows us to consider each speech feature as occupying a point, and each training class as occupying a sub-space (i.e. a representative point surrounded by some spread, or deviation), within the  $n$ -dimensional classification space. Viewed as such, the classification problem is that of determining to which sub-space class each feature vector belongs. MATLAB version of LIBSVM is used for training and testing. The database consisting of 54 dimensional feature vectors of 1000 utterances in five emotions is the data we have. The data is first split into training and testing sets. We used 800 utterances for training and the rest 200 for testing equally split among the five emotions. The following is the procedure to be followed when LIBSVM is used in training. The scaling process done should be same on both training and testing data. 1. Transform data to the format of an SVM package 2. Conduct simple scaling on the data 3. Consider the RBF kernel 4. Use cross-validation to find the best parameter  $C$  and  $\gamma$  5. Use the best parameter  $C$  and to train the whole training set 6. Test For training 800 samples are taken proportionally 175 from each emotional category to form a  $800 \times 54$  matrix. This is the testing instance matrix. Next a label vector containing the numbers corresponding to the type of emotion in the 800 sample instance matrix is created. This matrix is then used to train an SVM which outputs a model which is further used for testing. The test vector is a  $200 \times 54$  matrix. It is given as input to `svmpredict()` function which predicts the corresponding labels

## 7. REAL TIME IMPLEMENTATION

Real time implementation involves detecting the emotion from speech sample recorded in real time. The MATLAB GUI tool guide is used for creating the GUI.

A graphical user interface (GUI) is a pictorial interface to a program. A good GUI can make programs easier to use by providing them with a consistent appearance and with intuitive controls like pushbuttons, list boxes, sliders, menus, and so forth. The GUI should behave in an understandable and predictable manner, so that a user knows what to expect when he or she performs an action. For example, when a mouse click occurs on a pushbutton, the GUI should initiate the action described on the label of the button. A graphical user interface provides the user with a familiar environment in which to work. This environment contains pushbuttons, toggle buttons, lists, menus, text boxes, and so forth, all of which are already familiar to the user, so that he or she can concentrate on using the application rather than on the mechanics involved in doing things. However, GUIs are harder for the programmer because a GUI-based program must be prepared for mouse clicks (or possibly keyboard input) for any GUI element at any time. Such inputs are known as events, and a program that responds to events is said to be event driven. The three principal elements required to create a MATLAB Graphical User Interface are: 1. Components. Each item on a MATLAB GUI (pushbuttons, labels, edit boxes, etc.) is a graphical component. The types of components include graphical controls (pushbuttons, edit boxes, lists, sliders, etc.), static elements (frames and text strings), menus, and axes. Graphical controls and static elements are created by the function `uicontrol`, and menus are created by the functions `uimenu` and `uicontextmenu`. Axes, which are used to display graphical data, are created by the function `axes`. 2. Figures. The components of a GUI must be arranged within a figure, which is a window on the computer screen. In the past, figures have been created automatically whenever we have plotted data. However, empty figures can be created with the function `figure` and can be used to hold any combination of components.

3. Callbacks. Finally, there must be some way to perform an action if a user clicks a mouse on a button or types information on a keyboard. A mouse click or a key press is an event, and the MATLAB program must respond to each event if the program is to perform its function. For example, if a user clicks on a button, that event must cause the MATLAB code that implements the function of the button to be executed. The code executed in response to an event is known as a call back. There must be a callback to implement the function of each graphical component on the GUI. MATLAB GUIs are created using a tool called `guide`, the GUI Development Environment. This tool allows a programmer to layout the GUI, selecting and aligning the GUI components to be placed in it. Once the components are in place, the programmer can edit their properties: name, colour, size, font, text to display, and so

forth. When guide saves the GUI, it creates working program including skeleton functions that the programmer can modify to implement the behavior of the GUI. When guide is executed, it creates the Layout. The large white area with grid lines is the layout area, where a programmer can layout the GUI. The Layout Editor window has a palate of GUI components along the left side of the layout area. A user can create any number of GUI components by first clicking on the desired component, and then dragging its outline in the layout area. The top of the window has a toolbar with a series of useful tools that allow the user to distribute and align GUI components, modify the properties of GUI components, add menus to GUIs, and so on. The basic steps required to create a MATLAB GUI are: 1. Decide what elements are required for the GUI and what the function of each element will be. Make a rough layout of the components by hand on a piece of paper. 2. Use a MATLAB tool called guide (GUI Development Environment) to layout the Components on a figure. The size of the figure and the alignment and spacing of components on the figure can be adjusted using the tools built into guide. 3. Use a MATLAB tool called the Property Inspector (built into guide) to give each component a name (a "tag") and to set the characteristics of each component, such as its color, the text it displays, and so on. 4. Save the figure to a file. When the figure is saved, two files will be created on disk with the same name but different extents. The fig file contains the actual GUI that you have created, and the M-file contains the code to load the figure and skeleton call backs for each GUI element. 5. Write code to implement the behavior associated with each callback function.

## 8. SPEECH EMOTION DETECTOR GUI

The GUI is as shown below in the figure. The basic components used are 3 pushbuttons and an axes. The pushbuttons are: 1. RECORD: for recording the speech in real time 2. DETECT: for detecting the emotion of the recorded speech. 3. RESET: for resetting the GUI to initial state at anytime. The figure below shows the initial state of the GUI. Only the record and reset pushbuttons are active and the other detect button is inactive or off.



On pressing the record button the recording starts. The recording is done with the help of inbuilt windows recorder by using the function `wavrecord()` in MATLAB. The recorded speech is now stored in a variable. After the recording is done the record button goes off and the detect button turns active. Reset button remains the same. On pressing the detect button the main program is called and the emotion detection program is run. Features are extracted from the speech recorded and a feature vector is formed which is fed as a testvector to the SVM for testing. After testing a label is predicted by the SVM. The corresponding emotion is displayed with the help of a smiley in the axes using `imshow()` function. The figure below shows the GUI in the case of a happy speech sample.



Speech emotion detector GUI

## 9. RESULT AND CONCLUSION

On testing with the test vectors an accuracy is output by LIBSVM. It was found that on using RBF kernel though it gave 87% accuracy on cross validation the accuracy went down considerably below 50% on testing. Using trial and error methods it was found that the linear kernel gave maximum accuracy on testing with cost  $C=275$ . Hence linear kernel was used in further experiments. The testing with the final optimum parameters gave an accuracy of 67.826%. The table below shows the accuracy distributed among the five emotions. The highest accuracy rate is found to be for anger. And the lowest for happiness. Normal and surprised were found to have reasonable detection accuracy. Sad was found to be the second least detectable emotion. The results obtained on real time implementation was not as satisfactory. Recognition was good for angry and surprise. But the other three emotions had an uncertainty associated with it. It can be accounted to the fact that recording software used for recording speech training samples could not be used from MATLAB using the GUI. Also noise can be significant in real time. The training samples were taken in a noise free environment. Experimental results shows that the combination of both spectral features and prosodic features yields significant accuracy in detection of speech emotion recognition, over using only spectral and

prosodic features. To further improve the emotion recognition performance, it might be interesting to integrate other novel types of features, like articulatory feature, which has been proven to be an effective feature for speech recognition and speaker identification.

EMOTION	ACCURACY (%)
ANGER	76.67
HAPPINESS	51.7
NORMAL	75.33
SAD	62.1
SURPRISE	73.33

## REFERENCES

- [1] Cross-lingual speechemotionrecognitionssystem based on a three-layer model for human perception Elbarougy,R.;Akagi,M.Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific
- [2] Improving speechemotionrecognitionssystem for a social robot with speaker recognitionJuszkiewicz, L. Methods and Models in Automation and Robotics (MMAR), 2014 19th International Conference On
- [3] A study on the consistency of human perception and machine recognition of an emotional corpus Tsang-Long Pao ;Ren FuLuo;Yu-JiLiSystems, Man, and Cybernetics (SMC), 2012 IEEE International Conference
- [4] Emotional Speech Analysis on Nonlinear Manifold Mingyu You ; Chun Chen ; Jiajun Bu ; Jia Liu ; Jianhua TaoPattern Recognition, 2006. ICPR 2006. 18th International Conference on
- [5] Comparing Feature Sets for Acted and Spontaneous Speech in View of Automatic Emotion Recognition Vogt, T. ; Andre, E.
- [6] Motion Recognition in Spontaneous Speech Using GMMs Daniel Neiberg, Kjell Eleniusand Kornel Laskowski
- [7] Improving Automatic Emotion Recognition from Speech via GenderDifferentiationThurid Vogt, Elisabeth Andre ,Multimedia Concepts and Applications Group Augsburg University, Germany.