2016

Paralinguistic Speech Recognition: Classifying Emotion in Speech with Deep Learning Neural Networks

Eli Ridley Segal
Bard College

Paralinguistic Speech Recognition:

Classifying Emotion in Speech with Deep Learning Neural Networks

Senior Project submitted to

The Division of Science, Mathematics and Computing

of Bard College

by

Eli Segal

Annandale-on-Hudson, New York

May 2016

C O N T E N T S

Abstract

Emotion recognition in speech using deep learning begins as a problem of translating raw auditory data into an informationally rich feature set that can be trained on by a neural network

and, ideally, result in a machine learning system capable of accurately classifying the paralinguistic content of speech. We performed feature extraction using Praat, a tool for phonetic analysis, and obtained a variety of harmonic, intensity, and spectral characteristics that together formed the basis for the training vectors in our machine learning system.

While a number of different machine learning approaches have proved successful, there has been a strong resurgence in the application of so-called ‘deep learning’ systems to machine learning problems due to the striking degree of success that has been achieved with them using modern hardware [8]. After empirically validating a network architecture and learning parameters we trained six neural networks for the problem of emotion classification. We used six independent networks, each with the same network architecture and learning parameters, because this allowed us to obtain empirical data about the relative efficacy of different training features. Our six training sets represent the pairwise extraction of different subsets of features from a larger pool of features ranging from spectral and energy characteristics to periodicity. This allowed us to paint a more general picture of which facets of human speech are most relevant to and indicative of emotionality.

Introduction

Anger, boredom, anxiety, joy and many other emotional states are manifest in our speech. Such affect represents highly semantic content and presents a unique challenge with substantial applications. The frustration present in someone’s request, reluctance under the surface of someone’s response, happiness in someone’s greeting; such emotional inflections are far from superfluous in the complicated process of parsing meaning from speech. The automatic classification of emotion may have applications in smart human-computer interaction and could

conceivably be a tool for the indexing of multimedia. Textual sentiment analysis within the field of natural language processing has proved to be a valuable area of research and emotion recognition in speech may be a natural extension of this.

Emotion recognition is a related problem to that of speech recognition with the defining difference being that speech recognition is concerned primarily with the classification of the individual lexical items, the words, of an utterance. By contrast, emotion recognition pays no heed to the discrete entities of our expressions and is instead concerned with the paralinguistic qualities — the inflections, tonality, emphasis, prosody, etc. — which together constitute a kind of emotional profile for an utterance. It is the problem of deriving high level affective states on the level of utterances from such low level features of a speech signal as pitch and intensity. While undoubtedly still a nascent field, speech recognition has far-reaching applications and research on it stems from disciplines as varied as psychology, artificial intelligence, pattern recognition, and, in both current research and our own work, is a promising area of research where high success rates and classification accuracies have been achieved.

Before we begin to assess some of the prior literature it is worth mentioning that we will refer to “classification accuracy” frequently and by this we mean the number of correct classifications divided by the total number of utterances. While it is highly intuitive to understand what a correct classification is, a predicted emotion lining up with the correct emotion, it is important to note that this is still a step removed from the actual output of many emotional classifiers. In the case of neural networks, both with our own

Our study in particular was concerned with the relative success and efficacy of particular features and subsets of features, as well as various methodologies for preprocessing those features, in classifying emotion. This is a contrast between our work and much of the

surrounding literature, where generally the success of the network is the primary goal. Such studies in the early years of emotion recognition made strong progress using Gaussian mixture models and Hidden Markov models to learn a distribution of low-level features which were taken in tandem with a Bayesian classifier or the maximum likelihood principle to derive a classification of emotion. Later, studies began to train "universal background models [on the low-level features] and then generated supervectors for SVM classification," which is reminiscent of work done in speaker identification" [9].

Despite the many and varied approaches to emotion recognition a common refrain noted in Xiao's, *et al. Hierarchical Classification of Emotional Speech*, wherein they express that one of the "main difficulties for an efficient speech emotion classification reside in complex emotional class borders leading to necessity of appropriate auditory feature selection" [3], a point which is further emphasized in Han, Yu and Tashev's article *Speech Emotion Recognition Using Deep Neural Network and Extreme Learning Machine,* "Speech emotion recognition is a challenging problem partly because it is unclear what features are effective for the task" [9]. The thematically consistent obstacle of feature selection motivates our own research and is why we were concerned not with achieving state-of-the-art classification accuracy but rather with the relative efficacy of various feature sets.

In addition to the high variety of classifiers and permutations of those classifiers, an analysis of previous approaches also highlights the lack of a literature-consistent definition of speech emotions or emotion classes. In [10] Polzin and Waibel examined speech segments from English movies, making use of classical prosody and spectral features, and classified according to 3 general emotion sets: sad, angry, and neutral. In their study an accuracy of 64% was

achieved which, interestingly, they note is quite close to human level classification accuracy, based on their studies.

Dimitrios, *et al.* [11][12] made use of the Danish Emotional Speech database wherein speech segments are labeled according to 5 emotional classes: anger, happiness, neutrality, sadness, and surprise. Classifying over the DES set on 5 different emotions, [11] achieved an accuracy of 54%. Prior to Dimitrios *et al.* the work of McGilloway *et al.* [13] similarly had 5 separate emotion classes, however rather than 'surprise,' [13] had 'fearful' as their 5th class.

These studies demonstrate the flexibility and lack of consistent definition for emotional classes. In our own work we decided to expand on the classes used by Polzin and Waibel, as they are all negative and are thus decidedly unrepresentative of the emotional spectrum, but to also disclude the 'surprise' and 'fearful' emotional classes as they appear inconsistently throughout the literature and, further, are peculiarly difficult to obtain informative data for. As such we make use of four total emotional classes: anger, sadness, happiness, and neutrality. The three former emotions triangulate a wide emotional spectrum while also maintaining a general lack of overlap as each are in many ways poles of their respective emotional hemispheres. Neutrality is a contrast to this as it can neither be said to be a distinct emotion nor does it sustain the lack of commonality — as we will see, neutrality and sadness share many of their spectral and energy related characteristics — however we felt it was important to allow for the *lack* of powerful emotional affect in speech as well because in our day to day lives and interactions there is a not insubstantial amount of neutral speech.

In order to actually recognize these emotions though, we must first process the audio signal into emotion-relevant features, and then further into trainable vectors for a neural network. Within the surrounding literature it is a general rule to provide a discussion, and ideally

justification, of the features chosen for training. However, as our work is devoted in large part to the empirical justification of such features, it would be counterproductive to have a priori justifications for the features we seek to justify. It is for this reason that, rather than obtaining several specific, handcrafted features (as has often been done), such as median duration plateaus at pitch minima [H [6]], we instead examine more varied and abstracted features. Specifically we obtain matrices of pitch, spectrogram, intensity, and harmonicity which together constitute the body of utterance-level features which we pull from, pairwise, to examine their relative efficacy. As Huang, *et al.* [2] note, "the quality of feature extraction directly [affects] the accuracy of speech emotion recognition," and our project seeks to bring out and clarify the relationships between features and the overall ability of the network to accurately classify.

In *Emotion Recognition From Expressions in Face, Voice, and Body,* Banziger and Scherer proposed that fundamental frequency correlated directly to emotional content. Further, the work of Huang, *et al.* [2] indicates that the curve for pitch variance is also meaningful in expressing emotion, for example the pitch variance curve for 'sad' is generally decreasing while that of 'anger' plateau's, and 'happy' tends to slope upwards. Our own work demonstrated a relationship between pitch and the ability to distinguish between anger and happiness, as well as the correlation between intensity and both sadness and neutrality. The results from our work indicate a relationship between intensity and sadness/neutrality, and also between pitch and happiness/anger. The pairwise network classifications demonstrated significant disparities between the performance of each feature subset, with all spectrogram networks consistently outperforming non-spectrogram trained networks. On average the networks achieved higher classification accuracies than current state of the art models but the limited size of our data set, as

well as the self-imposed controls we placed on that data, together provide a possible explanation for the high performance.

Data and Methods

Data

There were several different corpora for emotional speech obtained throughout the course of this project. The University of Southern California has a number of emotional speech research labs which provided us with the following databases: Interactive Emotional Dyadic Motion Capture (IEMOCAP),[1] Electromagnetic Articulography (EMA),[2] as well as the Modeling Creative and Emotive Improvisation in Theatre Performance[3] database.While each of this datasets was created with needs and intentions distinct from our own, they all provide, in some form, labeled emotional speech data. A common dataset used within the emotion recognition literature is the Berlin Database of Emotional Speech (BES), which is referenced positively in both [2] and [14]. However, the BES contains only utterances spoken in German and, though it is plausible that emotional inflections cross language borders, the phonetic and articulatory differences between German and English present a potential for inapplicability that dissuaded our use. All three of the USC datasets provide usable English utterances labeled with emotional

[1] C. Busso, M. Bulut, C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. Chang, S. Lee, and S. Narayanan, "IEMOCAP: Interactive emotional dyadic motion capture database," Journal of Language Resources and Evaluation, vol. 42, no. 4, pp. 335-359, December 2008.

[2] Sungbok Lee, Serdar Yildirim, Abe Kazemzadeh and Shrikanth S. Narayanan, An articulatory study of emotional speech production, in Proceedings of InterSpeech, pages 497-500, 2005

[3] C. Busso, M. Bulut, C.C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J.N. Chang, S. Lee, and S.S. Narayanan, "IEMOCAP: Interactive emotional dyadic motion capture database," Journal of Language Resources and Evaluation, In press, 2008

classes, however both the IEMOCAP dataset and the Emotive Improvisation dataset are composed of unique utterances with various speakers — which would be ideal were we attempting to meet or surpass the current state of the art systems in emotion classification — however, as our particular permutation of this problem is concerned primarily with the effectiveness of individual features relative to one another and *not* the effectiveness of the network as a whole relative to other classifiers, we ultimately opted to use the EMA database because of the controlled environment for feature observation that it facilitates.

The EMA database is constructed to model the articulography of the face whilst speaking as a function of emotional inflection. Due to the radical differences in articulatory movements from one expression to the next, Lee, *et al.* constructed a dataset wherein there are four utterances made for every unique expression, corresponding to four emotional classes. The variation expressed across multiple unique sentences is thusly mitigated and we speculate this will also result in more learnable features as, for a given expression, the only contrast between each of four utterances will be the emotional inflection. The database contains, in total, 70 unique expressions, spoken with four different emotional inflections, by three different speakers. This means there are a total of 840 individual utterances contained in the EMA corpus, however in the interest of further control for feature analysis we decided to limit the number of speakers to one which yields the 280 utterances of our training set.

Feature Extraction

To classify emotion in speech there are a number of important problems which must first be accounted for. The rough shape of these obstacles is:

1) Signal processing and feature extraction
2) Feature preprocessing and training vector creation

3) Implementation and training of neural network

4) Analysis of results of trained network on testing set

Each of these is highly cursory so let us begin to unpack. We begin the process of emotion detection at the source, the auditory signal. We find our initial material in the Electromagnetic Articulography database for the study of expressive speech, courtesy of the SAIL Laboratory at the University of Southern California. As mentioned in the previous section, the EMA database contains 680 utterances spoken with 4 different emotional inflections. Thus our source data takes the form of WAV files for individual utterances that are spoken in 4 different target emotions. To process the waveforms we made use of the phonetic analysis tool Praat.[4] We used Praat for a number of reasons, it is an industry and scholarly standard which is convenient for both contextual comparison and the availability of documentation. Moreover though, Praat has its own scripting language that accompanies the software, this allowed us to perform large scale audio analysis while maintaining the specificity necessary for feature extraction that's tailored for the training of neural networks. Praat has a wide range of different audio processing capabilities which allow for the analysis of periodicity as well of spectral, harmonic, and energy related characteristics. In order to obtain the broadest possible feature range not precluded by combinatorial explosion we chose the following signal features for extraction: pitch, intensity, harmonicity, and spectrogram. For every utterance in our data set we used Praat scripts to extract these four features and then utilized Praat's 'To Matrix' functionality to further manipulate the data into readable matrices such that they would be ready for pre-processing. A highly condensed version of the Praat script for feature extraction can be seen below.

[4] Boersma, Paul & Weenink, David (2016). Praat: doing phonetics by computer [Computer program]. Version 6.0.17, retrieved 21 April 2016 from http://www.praat.org/

```
for i from 1 to numWaveforms$:
      selectObject: i
      To Pitch: 0.05, 75, 600
      To Harmonicity (cc): 0.05, 75, 0.1, 1
      To Spectrogram: 0.005, 5000, 0.05, 20, "Gaussian"
      To Intensity: 100, 0.05, "yes"
endfor
for i from featureStart$ to featureEnd$:
      selectObject: i
      To Matrix
      name$ = extractLine$ (obj$, " ")
      shortened$ = name$ - right$ (name$, 6)
      Save as headerless spreadsheet file: shortened$ + num$ + [".pitch",
                              ".harmonicity", ".intensity", ".spectrogram"]
endfor
```

The Praat scripting language allows the user to determine their desired features and feature parameters in advance and then perform the extraction on a large scale. In our case a consistent time step of 0.05 seconds was chosen as it provides both us with the sensitivity to subtle temporal shifts that we hypothesize to be relevant to and indicative of emotional inflection, while also yielding matrices with only one to two hundred elements which means processing and training can be accomplished within reasonable timeframes. Other feature parameters, such as the floor and ceiling frequency for pitch, 75Hz and 600Hz respectively, were chosen according to widely accepted values in the literature of speech recognition. The adult male voice ranges from a fundamental frequency of 85Hz to 180Hz while that of the adult female ranges from 165Hz to 255Hz,[5] which, in addition to the natural variability across different voices, is why we chose the range 75-600. The second for loop in the code above utilizes Praat's capacity for vectorization to transform large matrices into vectors and then save them to simple spreadsheets so that we can perform further processing using a different, more flexible language such as Python with NumPy.

[5] Titze, I.R. (1994). Principles of Voice Production, Prentice Hall (currently published by NCVS.org) (pp. 188),

Once we created the feature matrices for our entire data set we now had all the raw information necessary to begin collating our training vectors. All preprocessing work for the project was done using Python which allowed us to rapidly prototype ideas while also having the computational efficiency to perform semi-large scale mathematical operations because of NumPy — though it is worth noting that this was possible largely because of the small size of our primary data set and were we working with data on the scale of hundreds of thousands or even just tens of thousands then Python would likely not be appropriate any longer, even with NumPy.

Preprocessing

While dozens of programs were created in the course of this project, there were ultimately several important factors in our preprocessing work that took shape in the form of six primary programming tasks:

1. Reading the features from our spreadsheet files into usable data structures
2. Performing operations on these matrices to derive all training features
3. Obtain normalized training vectors
4. Obtain all training vector sets corresponding to every feature pair
5. Write the training vector set for each feature pair to different pattern files that can be learned on by the neural network
6. Read the results corresponding to each network trained on a different feature pair
7. Process results matrices into accuracy scores and confidence intervals

I will forego explanation of steps 1, 5, and 6 because though each was not without their own difficulties, these tasks were ultimately technical in nature and merely represent the scripting

hurdles necessary to interface with JavaNNS, our neural network simulator, and not the more substantive challenges endemic to the problem of emotion recognition in speech.

Beginning with the matrices obtained from our feature extraction, now appropriately conformed to usable data structures within Python, we needed to reduce the large feature matrices into low level descriptors for the training of our neural networks. The reason for this reduction comes from the significant disparities in matrix length that are a result of the speaker's propensity to take more or less time in articulating any given utterance depending on which emotion they were emulating. As was previously mentioned, the time step that we used for feature extraction was 0.05 seconds, which means that the length of matrices resulting from an utterance that was 3 seconds long versus 10 seconds long varied by a factor of 3.33 (60 elements vs. 200 elements). As Huang, *et al.* note in [2]

> "Speech time construction refers to the emotion speech pronunciation differences in time. When people express different feelings, the time construction of the speech is different. Mainly in two aspects, one is the length of continuous pronunciation time, the other is the Average rate of pronunciation.One is the length of continuous pronunciation time and the other is the average rate of pronunciation"
>
> *A Research of Speech Emotion Recognition Based on Deep Belief Network and SVM*

The wide temporal range of our data precluded raw, unprocessed input because the number of input nodes in a network must remain constant throughout training. That being said, this is not without issues, as Amer, et al. note, "A standard approach to solving this problem involves extracting framewise low-level descriptors (LLDs) from the audio signal and then using functional features to aggregate these features over the utterance level.. A major drawback of

these approaches is that they ignore the temporal dynamics of the phenomena both within and across utterances" [1].

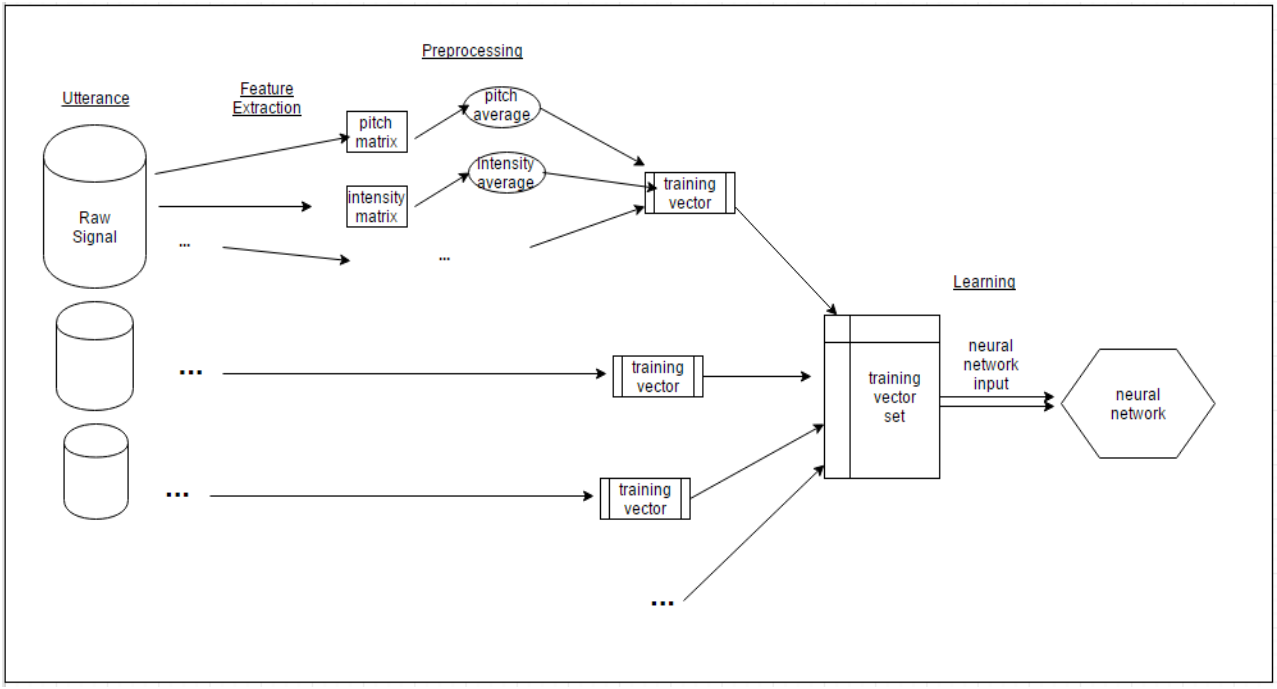
There are a number of alternative approaches to data processing that we considered as a means to retain the rich temporal information of our extracted features:

1) Determining the average utterance length and then adjusting the time step length for all extracted features depending on the current utterance length would allow us to, in a sense, stretch or compress each feature matrix to conform to the average matrix size, thus enabling fully consistent matrix sizes and so too applicability to the neural network.

2) By determining the maximum utterance length we could 'pad' all non-average utterances with null values on either end such that the matrix of every feature extracted would have the same length.

Unfortunately however, there is no uncomplicated solution. 1) literally skews the data and while there is a possibility that such warping would not be relevant to the network's representation of emotion, it is far more likely that 'speeding up' or 'slowing down' utterance level locutions would be highly impactful to the network's ability to accurately classify. 2) avoids this problem by maintaining a consistent time step between utterances but does so at a high cost; the wide temporal range means that the maximum utterance length is likely several standard deviations from the average and would thus cause the majority of utterances to be padded with almost as much useless (null) data as substantive data. We concluded that each of these approaches would likely be more detrimental than beneficial relative to the reductive but unmanipulated utterance-level, aggregate features. For this reason we chose to use utterance level features for the training of our networks.

There are a great variety of possible candidates for the utterance level features, pitch minima, intensity maxima, harmonic means, spectral medians, word length, pause length, rate of syllables spoken, &c. However we wanted to leverage a limited number of inputs against the range of possible feature combinations while maintaining a controlled, consistent environment for their analysis. This resulted in an input layer containing two nodes, where each node represents one feature (in [pitch, intensity, harmonicity, spectrogram]) and each is taken as the overall average of that extracted feature matrix for that particular utterance. For a less convoluted explanation of this workflow see the diagram below.



For each feature, we extract a matrix from an utterance, and from that matrix we compute the mean. This results in four different values, each representing the average of a particular feature for a particular utterance. Rather than supplying the network with all of these values simultaneously though, we hope to evaluate network performance as a function of the features input which can be accomplished by providing only two of these values (features) at any given

time as the network input. Two nodes may seem too constrained an input to reliably classify emotion, and indeed our own expectation entering into this project was that that would be the case, however the goal of our research was less to create the most sophisticated possible neural network and achieve state-of-the-art level classification accuracy, and more to see how a network trained on features x and y performs relative to one trained on w and z, as well as x and z, and so on. Thus having two input nodes serves two primary functions here: Pairwise feature comparison and clarity regarding feature efficacy.

Using only two nodes allows us to perform pairwise comparisons between features across the entire feature space. I.e. with 'average intensity,' 'average pitch,' and 'average harmonicity' as features we are able to efficiently demonstrate which feature subsets perform most effectively in the classification of our emotions, whereas a larger quantity of input nodes carries the burden of either the inability for comparison (because the nodes fully encompass the feature space and hence leave no room for comparison) or the combinatorial explosion of exhaustively training on larger feature subsets. For example, were we to have 10 features and use 4 of them as input in any given network, we would need to create and test *hundreds* of networks to exhaustively compare over the feature space. Further the use of two nodes means that our actual results — whether in the form of error over time, connection weights, or overall accuracy in classifying — provide us with a far more controlled indication of the predictive power of particular features and particular feature combinations.

A final step taken before the data can be provided to the neural network as input is normalization. The actual values for such features as average pitch or intensity are relatively massive taken in the context of our randomly initialized network weights and the fine tuned adjustments that learning makes to them. Providing values of such radically different magnitudes

to the network is, as my advisor put it, like using a sledgehammer when we need tweezers. We did not initially take the time to normalize the input for this project and when we began to train the networks, the consequences were seen immediately. The provided input would create such drastic weight changes that they would become stuck in local minima, unable to escape in spite of any number of training cycles, essentially halting learning altogether. Fortunately normalization was performed by obtaining z-scores for all input vectors by determining the overall mean of that feature in the training set, and then the standard deviation, which finally allows each value to be converted to a representation of its relationship to the mean of the set. Specifically, the error of a particular value from the mean is divided by the standard deviation to obtain the z-score for that value:

$$Z_s = (s - m) / d$$

Where the z-score for s, Z_s is given by s — the mean, m, divided by the standard deviation, d.

Neural Network

Having established the form of our input, the next step is creating actual network. Over the course of the project a fairly diverse array of different tools for the creation of neural networks were learned, Tensor Flow, Theano, and JavaNNS foremost among them. Tensor Flow and Theano are Python compatible computational libraries for the efficient (due to outsourcing complex computations to objective-C) computation and manipulation of multi-dimensional arrays and are thus used for the representation of machine learning problems and specifically the simulation of neural networks. Ultimately though the number of network architecture parameters that warranted empirical testing, in addition to the variety of possible learning algorithms, the parameters of those algorithms, the fact that at all times we were working with six different

networks (six feature subsets), and finally the sheer magnitude of actual network instantiations necessary due to the variation caused by random initialization, all led us to JavaNNS for the ability to rapidly construct networks, vary parameters and architecture, and instantly initialize without losing data on previous network iterations (for example, being able to run five different network inits/trainings all on one error graph for easy analysis).

The learning rate for a network determines the speed at which the network weights converge on a solution. A higher learning rate can speed up the training process, making the weights converge more quickly, but only up to point as too large a learning value can cause the network to become unstable and oscillate erratically as it approaches a solution. For this reason we chose a learning rate of 0.2, which is small enough that learning does create network instability but large enough that the time taken for training is manageable. However, this is only a partial solution as it ignores the problem of local minima in training. Naturally the goal of the learning process is to minimize error, this would mean that the overarching goal of training is to converge on a *global* minima but the reality is that with a learning step of only 0.2 it is quite plausible for the network to become 'stuck' in a local minima, oscillating within it without the capacity to surpass the increase in error necessary to breach the minima. A common solution within machine learning generally and also specifically within the literature of emotion recognition is to introduce a momentum parameter. Here we see that the ***i***-th correction for weight $\mathbf{W_k}$ is given by:

$$\Delta W_k(i) = -\alpha \frac{\partial E}{\partial W_k} + \mu \Delta W_k(i-1)$$

Where ($\partial \mathbf{E} / \partial \mathbf{W_k}$) is the variation of the loss with respect to k, and □ is our momentum parameter, such that 0 < □ < 1. The equation allows for the weight to be modified by the previous weight update, where momentum * (previous weight update) is added to the current

weight update. Another means for reducing the influence of local minima comes from the seminal *Neural Network Learning and Expert Systems* (Gallant, 1993), Gallant found that randomly initializing the weights in our network to small positive and negative values, in our case ranging between -1 and 1 attenuates the problem local minima. Due to this random initialization though, we introduce a level of intrinsic variability into the network which must be accounted for. As such, for the empirical validation of our network architecture and learning parameters all data represents an average across at least 5 network iterations.

The training and analysis of the networks takes place over three different data sets: Training set, cross validation set, testing set. The training set constitutes 71% of our overall data set, while both the cross validation set and the testing set are 14.5% of the data. Rather than throw everything at the network we utilize a cross validation set because it insures that we are learning not just the character of a particular dataset but a broader representation of the kinds of patterns that data set exemplifies. Trained without cross validation a network will become overly specified to its input, learning in gory detail the intricacies of one particular data set and classifying that dataset exceptionally well. Cross validation allows learning to take place on the training set while referencing that learning against an independent set, thus allowing training to cease when the error against the cross validation set begins to increase instead of decrease. This allows us to maintain generalizability throughout learning and results ultimately in far greater accuracy when classifying over the test set, despite the fact that a network trained with cross validation will have greater error on the training set.

In our case, using JavaNNS, we are able to provide the testing set and once learning has halted, can provide each utterance in the testing set to the network and evaluate the probabilistic distribution that the network outputs. The evaluation itself takes the node containing the

maximum value from the network output and checks whether this lines up with the correct classification. For example, anger is represented by the one-hot vector [1, 0, 0, 0] and if the largest value (most highly predicted) network output is also the first element of that set then we have a correct classification.

In order to build an effective network for a learning problem we must first determine a network architecture that is appropriate to our inputs, feature pairs, and our outputs, four emotions. As discussed in the introduction, the machine learning problem of emotion classification has had a wide range of different approaches and, even specifically within the domain of neural network approaches, there is no consistent definition or convention for most effective architecture or learning methodology. It is of further importance that, while our emotion classifier falls generally into the domain of emotion recognition, our this is nonetheless unique insofar as our emotion classes, pairwise feature inputs, and data gathered all collectively engender a fully distinct permutation of the classic problem. As such, we felt it was necessary to obtain a variety of empirical data on the many potential neural network structures and parameters for our learning problem. To begin with we performed extensive testing on a number of possible architectures, varying the number of nodes within the hidden layers, as well as the number of hidden layers themselves. A selection of this data can be seen in the table below.

Table 1. Resulting classification accuracy from various network architectures all trained on the pitch-intensity feature pair.

	Network Architectures	**Accuracy (Top 1)**	**Accuracy (Top 2)**
(1)	**2 - 2 - 4**	53.85%	92.31%

(2)	**2 - 4 - 4**	61.53%	89.74%
(3)	**2 - 8 - 4**	58.97%	89.74%
(4)	**2 - 4 - 4 - 4**	53.85%	97.44%
(5)	**2 - 4 - 8 - 4 - 4**	43.49%	87.18%

*Note that the number of input and output nodes never changed, this data is a representation of the effect that varying # of hidden layers and # of nodes in those layers has on accuracy

Clearly in our case more layers does not ultimately effect a more accurate representation of the input. We see that beyond one hidden layer the network's most highly predicted emotion lines up with the correct emotion only ~54% and ~43% of the time for 2-hidden layer and 3-hidden layer networks respectively, whereas a single hidden layer containing four nodes achieves an accuracy of over 60%. In addition we can see that one hidden layer containing only two nodes, (1), does not provide the network with enough free parameters for effective representation and results in ~54% accuracy, while too many nodes can also dilute the network effectiveness, as is the case in the ~59% accuracy of (3). For these reasons we chose to use a single hidden layer containing four nodes for all of our networks. Since each network is trained on a different feature pair and each network's outputs represent the same four emotional classes, there will be a consistent architecture for every network. This means that the only variable from one network iteration to the next is the input, and as our input takes the form of feature pairs, this allows for a more controlled environment and hence a more incisive view of the relative efficacy of different feature subsets.

As we now possess the general schematic for our network we can determine the proper learning parameters for the network and then begin to train on our feature pairs.

Table. Accuracy of Network Classifications by # of Training Epochs

# Training Cycles	Top 1 (Init)	Top 2 (Init)		Top 1 (One Net)	Top 2 (One Net)
100	64.10%	92.31%		56.41%	89.74%
200	69.23%	92.31%		61.54%	89.74%
1000	71.79%	89.74%		56.41%	97.44%
2000	71.79%	92.30%		61.53%	87.18%
10000	61.54%	89.74%		61.54%	89.74%
20000	66.66%	89.74%		61.53%	89.74%

100000	66.66%	89.74%		58.97%	94.87%

* The “Init” data refers to networks that were (randomly) initialized for every subsequent epoch test, whereas “One Net” refers to data gathered from a single network at different intervals of its training

Based on the above data gathered it is clear that effective training is certainly taking place between 100 and 1,000 training cycles but that somewhere in the 10,000 to 20,000 range of cycles the networks are no longer evolving in their classification of emotion. We can see further evidence of this in the error graphs for 5 randomly initialized networks training on various numbers of cycles. It is for this reason that moving forward we will use 5,000 training cycles as the standard for our exploration of pairwise feature sets.

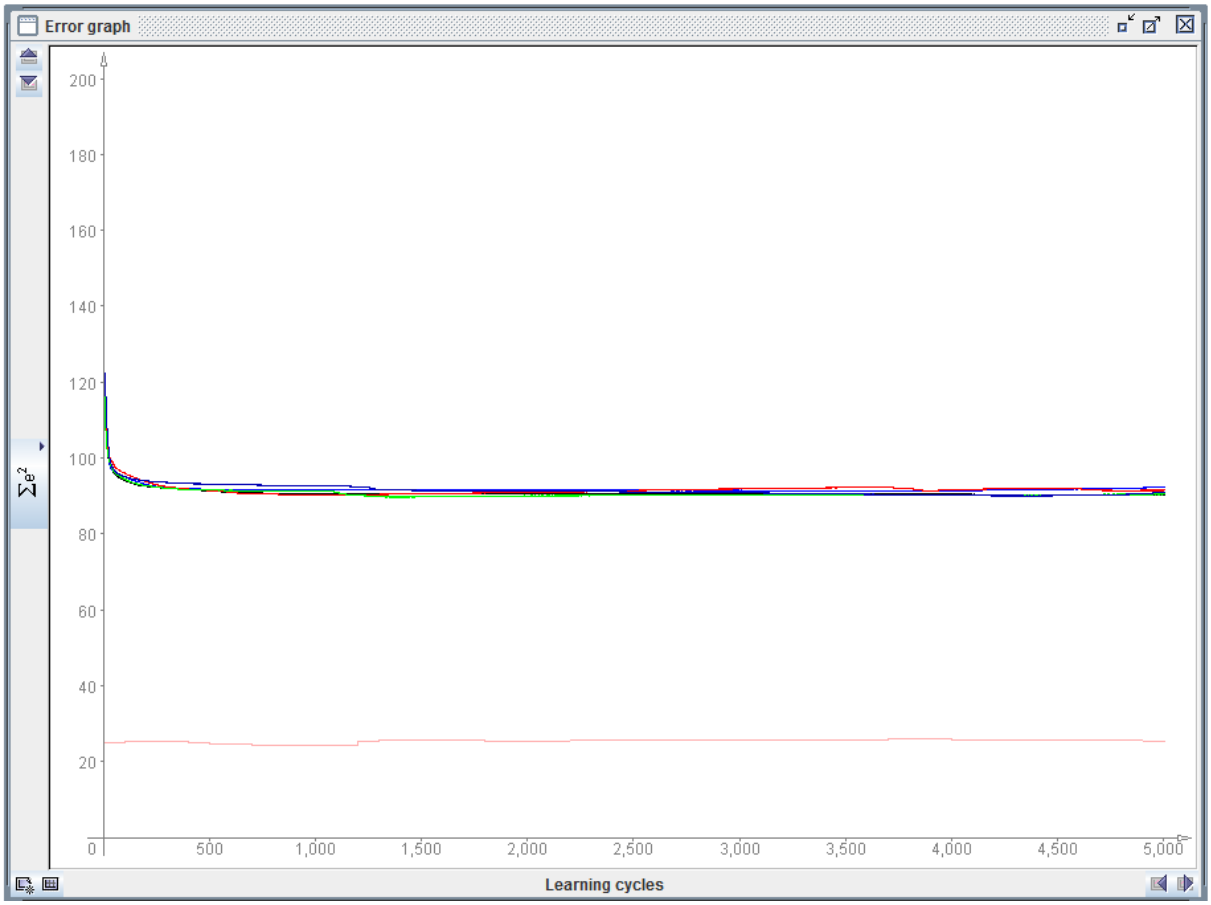
Following extensive observation over a variety of parameter intervals for our backpropagation with momentum algorithm we obtain the following parameters for effective learning:

Step Width	0.2
Momentum	0.5
Flat Spot Elimination	0.1
Max Propagated Error	0.1

Results and Limitations

Having now established our the length of our training and the parameters for our backpropagation method, we may now begin to analyze the actual data. For each pairwise feature set we create 5 separately initialized networks. This is because, due to the fact that every network we create is randomly initialized (all weights are set to a random number between -1 and 1) which means that the networks themselves will vary from one another based on their starting states alone. Below we can see the error graph for our 5 separately initialized networks learning

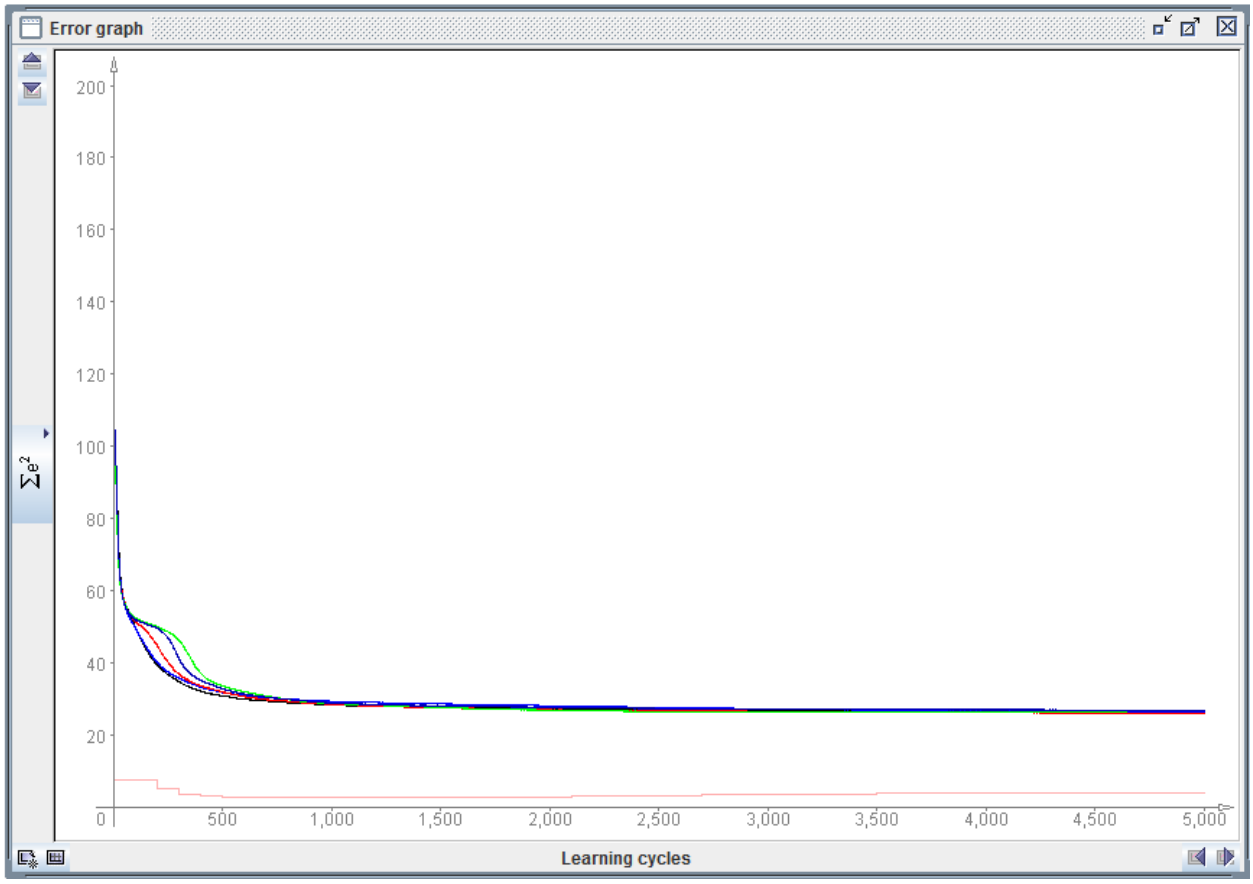
on the Pitch-Intensity feature pair, where each network is displayed in a different color.



The x-axis represents the number of learning cycles and hence begins at 0 and ends at when 5,000 cycles are reached. The y-axis represents the sum-squared error of the network. Due to the number of training cycles the graph is somewhat condensed and the variation is thus less apparent, however it is clear that the error plot of our 5 networks do not follow identical lines. For this reason we obtain our accuracy measures for each feature pair based on an averaging of the accuracies for 5 separate initializations and trainings of the network. We can see this in the table below:

Initialization #	Accuracy (Top 1)	Accuracy (Top 2)
1	64.10%	89.74%
2	66.66%	87.18%
3	64.10%	92.31%
4	64.10%	89.74%
5	61.54%	89.75%

The above table yields an overall accuracy for networks trained on the pitch-intensity feature pair of 64.10% for the top 1 and 89.74% for the top 2. Which of course means that 64% of the time pitch and intensity trained networks produced the correct classification of the emotions in our test set, while 89.7% of the time they made the correct classification in one of the top 2 outputs. Interestingly when we trained our network on the harmonicity-spectrogram feature set we observed a far lower average error:



*Note that the scale is the same here as in the previous error graph for the sake of comparison

This error graph provides further evidence for the extreme disparity in feature set efficacy and hence also the tremendous importance of choosing appropriate features for the training of an emotion classifying network. Unsurprisingly, given the above graph, the networks resulting from the harmonicity-spectrogram feature pair also radically outperformed the pitch-intensity networks in terms of classification accuracy: over 5 separate initializations the average top 1 accuracy was 92.31% while extending our bounds to the top 2 yields average accuracy of 94.87%. These numbers are extremely high and will warrant further discussion later on. Having touched on pitch-intensity and harmonicity-spectrogram, let's now examine the performance of all 6 of the feature pairs.

	Accuracy (Top 1)	**Accuracy** (Top 2)
Pitch - Intensity	64.10%	89.74%
Pitch - Harmonicity	58.97%	87.17%
Pitch - Spectrogram	89.74%	98.72%
Intensity - Harmonicity	71.49%	100.0%
Intensity - Spectrogram	83.34%	100.0%
Harmonicity - Spectrogram	92.31%	94.87%

From these results we can see some clear relationships and distinctions between features as the various pair accuracies were by no means homogenous. Another way of viewing this data that provides a different lens into feature efficacy is to plot the features as axes and view the accuracy as a relation between the feature pairs, an easy way to see this is in another table:

Table indicates Top 1 accuracy for various feature pairings

	Pitch	**Intensity**	**Harmonicity**	**Spectrogram**
Pitch	-	64.10%	58.97%	89.74%
Intensity	64.10%	-	71.49%	83.34%

Harmonicity	58.97%	71.49%	-	92.31%
Spectrogram	89.74%	83.34%	92.31%	-

We can do the same thing for the Top 2 accuracies of the feature pairs:

	Pitch	**Intensity**	**Harmonicity**	**Spectrogram**
Pitch	-	89.74%	87.17%	98.72%
Intensity	89.74%	-	100.0%	100.0%
Harmonicity	87.17%	100.0%	-	94.87%
Spectrogram	98.72%	100.0%	94.87%	-

It is also undoubtedly of interest how these features perform independently from one another. Thus we also tested the classification accuracy of networks trained on only one of these features at a time. However, to maintain the constancy of the network architecture we used both the average *and* the standard deviation of the feature in question such that we maintained the two input node structure while only testing on one feature. The results follow:

	Accuracy (Top 1)	**Accuracy** (Top 2)
Pitch	62.5%	90.0%
Intensity	77.5%	97.5%

Harmonicity	50.0%	72.5%
Spectrogram	97.5%	100.0%

As in our pairwise network accuracies, the spectrogram trained network significantly outperformed all other features. These results are consistent with the findings of Amer, *et al.* [1], who speculated that a simple low level feature such as a spectrogram can provide rich feature representations for emotion classification if utilized well. Their research with deep temporal models confirmed this and demonstrated that with their hybrid model and spectrogram feature extraction they were able to improve upon current state-of-the-art results. Our own work reinforced the importance of spectral features for a sophisticated representation and notably performed classifications with 97.5% accuracy, getting only one of the forty utterances in the testing set incorrect. To see how these results actually look, here is a selection of the network results classifying over a forty utterance data set containing only novel, unseen utterances. In order to see classifications over the full range of emotion we have randomly selected a single utterance for each emotion from our results table for the spectrogram trained network.

ANGRY				
Network Input	0.20376	0.20856		
True Output	1	0	0	0
Network Output	0.99841	0.00159	0	0
SAD				
Network Input	-0.94172	-0.94015		
True Output	0	1	0	0
Network Output	0.04379	0.80803	0.00277	0.21329
HAPPY				
Network Input	-0.79378	-0.79034		
True Output	0	0	1	0
Network Output	0.08863	0.09455	0.9015	0
NEUTRAL				
Network Input	-0.99135	-0.99185		
True Output	0	0	0	1
Network Output	0.00135	0.1074	0	0.89332

The network input is a vector of only two values where the first represents the average spectrogram value across the utterance and the second value represents the standard deviation from that average (SEM). The second row for each utterance contains a one-hot vector representing our desired output; we choose [1, 0, 0, 0] to represent ‘anger’ therefore that distribution is what we intend the network to learn from all of the anger-associated input. The network output is normalized as a softmax probabilistic distribution over which the maximum output is designated as the network’s Top 1 prediction. The spectrogram feature only failed to correctly classify a single utterance from the test set and, in this case, the second largest output was correct.

Interestingly, when we look at the results for intensity, all but one instance of an incorrect Top 1 prediction occurred when attempting to classify sadness and in almost every case the network took the sad utterance to be a neutral one. This is not particularly surprising, we even noted earlier that the inclusion of ‘neutral’ as an emotional class could be potentially problematic due to the speculated commonalities between it and sadness; however, these results ground the

speculative and show us specifically that the space of commonality between sadness and neutrality is localized to the intensity of the utterance. We can see another interesting relationship develop in the network trained on pitch: In all but two instances of incorrect Top 1 predictions occurred when the network was given a happy utterance and in every one of these cases the network instead predicted anger. Here we illuminate an overlap between happiness and anger that is local to the utterance' pitch.

The single-feature trained networks have more clearly defined relationships, as seen with pitch and anger/happiness and intensity with sadness/neutrality. The networks trained on feature pairs are of interest as well, though the substance of the relationship the pairs and emotions is less clearly circumscribed. To briefly touch on some of our more interesting findings though, of all of the different networks only one, that trained on pitch and harmonicity, misclassified 'anger' — every single classification over the testing set made by each of the other networks correctly identified anger. Strangely, the network trained on pitch and intensity had the most fully incorrect classifications (that were in neither the first or second most highly predicted emotions for the given utterance) and further, each instance of these fully incorrect classifications occurred on a 'happy' utterance that was predicted by the network to be 'sad.' As we have observed independent from one another, neither pitch nor intensity had any apparent proclivity for the misclassification of happy as sad or vice-versa, yet together they undoubtedly do. This demonstrates the important fact that a network trained on multiple features which do not independently carry a particular bias may very well come together to form one.

Future Work and Limitations

Overall, the results obtained from both the pairwise feature trained networks as well as the single feature trained networks outperform current state of the art models by a significant margin. The networks trained on pitch-intensity, pitch-harmonicity, and intensity-harmonicity were the only networks that scored below current models, [2], achieving 64%, 59% and 71% Top 1 accuracy respectively, though still outperforming than such early models as McGiloway [13], where a classification accuracy of 55% over only three emotion classes was achieved. The networks trained on the spectrogram had outstanding accuracy, often classifying every utterance in the test set within the Top 1 or Top 2. This being said, while the pitch-spectrogram network (90% Top 1 accuracy) and the spectrogram-harmonicity network (92% Top 1 accuracy) improved on the 86.5% Top 1 accuracy of.Huang, *et al.*'s *A Research of Speech Emotion Recognition,* we must reiterate that emotion recognition problems and distinct and varied.

Our own problem classifed over a small data set and the testing data set was also of limited size, as such the results obtained are not effectively comparable to those of networks trained on larger data sets. Further, the reduction of our own emotion recognition problem to a single speaker as well as our desire to control for articulatory variation from one unique expression to the next by using only the (EMA) data wherein each unique expression is spoken with each emotional class, together create a more controlled and learnable environment. This was beneficial for our own research as it clarified the relationships between feature sets and emotions, but it does come at the cost of being unrealistic as an actual, generalized emotion recognition system. That is to say, our networks achieved very high performance by learning a sophisticated representation of *one person's* emotional speech patterns, as opposed to the less accurate but more generalized systems of [1] and [2] which have learned a representation of general human patterns for emotional speech.

These limitations are by no means debilitating however, as state-of-the art accuracy is not the goal of this project. That being said, the representation learned of a single speaker can not necessarily be seen as representative beyond that individual. Therefore we hope, in future work, to broaden both the utterance sample size as well as the number of speakers, while being able to use the data obtained here, in a more controlled environment, as a baseline indication of feature performance down the line. By expanding the sample size and number of speakers we would also likely have enough data to allow for more complicated feature spaces, more network inputs, and the network architectures to support them. A larger feature space would both allow for greater breadth, covering more of the possible range of features, while also facilitating enhanced specificity where features could be designed to fit the specific needs of the problem. It is unavoidable in such feature analysis that, to exhaustively characterize the relationships of features in a feature space, there will be combinatorial explosion. This makes truly sophisticated networks (trained, for example, on 68 features as in Zhao, *et al*.'s *Hierarchical Classification*) unfeasible for full analysis — using [3]'s 68 features as the feature space and selecting a subset of 10 of them at any given time for training would necessitate the individual training of 290 billion networks for exhaustive comparison.

REFERENCES

[1]. Amer, M. R., Siddiquie, B., Richey, C., & Divakaran, A. (2014). Emotion detection in speech using deep networks. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. doi:10.1109/icassp.2014.6854297

[2]. Huang, C., Gong, W., Fu, W., & Feng, D. (2014). A Research of Speech Emotion Recognition Based on Deep Belief Network and SVM. *Mathematical Problems in Engineering, 2014*.

[3]. Xiao, Z., Dellandrea, E., Dou, W., & Chen, L. (2007). Automatic Hierarchical Classification of Emotional Speech. *Ninth IEEE International Symposium on Multimedia Workshops (ISMW 2007)*. doi:10.1109/ismw.2007.4475985

[4]. Schuller, B., Valstar, M., Eyben, F., Mckeown, G., Cowie, R., & Pantic, M. (2011). AVEC 2011–The First International Audio/Visual Emotion Challenge. *Affective Computing and Intelligent Interaction Lecture Notes in Computer Science,* 415-424. doi:10.1007/978-3-642-24571-8_53

[5]. Ramirez, G. A., Baltrušaitis, T., & Morency, L. (2011). Modeling Latent Discriminative Dynamic of Multi-dimensional Affective Signals. *Affective Computing and Intelligent Interaction Lecture Notes in Computer Science,* 396-406. doi:10.1007/978-3-642-24571-8_51

[6]. Siddiquie, B., Khan, S., Divakaran, A., & Sawhney, H. (2013). Affect analysis in natural human interaction using Joint Hidden Conditional Random Fields. *2013 IEEE International Conference on Multimedia and Expo (ICME)*. doi:10.1109/icme.2013.6607590

[7]. Glodek, M., Tschechne, S., Layher, G., Schels, M., Brosch, T., Scherer, S., . . . Schwenker, F. (2011). Multiple Classifier Systems for the Classification of Audio-Visual Emotional States. *Affective Computing and Intelligent Interaction Lecture Notes in Computer Science,* 359-368. doi:10.1007/978-3-642-24571-8_47

[8]. Litman, D., & Forbes, K. (n.d.). Recognizing emotions from student speech in tutoring dialogues. *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No.03EX721)*. doi:10.1109/asru.2003.1318398

[9]. Han, K., Yu, D., & Taschev, I. (2014). Speech Emotion Recognition using Deep Neural Network and Extreme Learning Machine. *Interspeech*.

[10]. Dellaert, F., Polzin, T., & Waibel, A. (n.d.). Recognizing emotion in speech. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*. doi:10.1109/icslp.1996.608022

[11]. Ververidis, D., Kotropoulos, C., & Pitas, I. (n.d.). Automatic emotional speech classification. *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*. doi:10.1109/icassp.2004.1326055

[12]. Ververidis, D., Kotropoulos, C., Automatic speech classification to five emotional states based on gender information, *Proceedings of 12th European Signal Processing Conference*, pp.341–344, September 2004, Austria.

[13]. McGilloway, S., Cowie, R., Douglas-Cowie, E., Gielen, C.C.A.M., Westerdijk, M.J.D., & Stroeve, S.H. Approaching automatic recognition of emotion from voice: a rough benchmark, *Proceedings of the ISCA workshop on Speech and Emotion*, pp. 207-212, 2000, Newcastle, Northern Ireland.

[14]. Kienast, M., & Sendlemeier, W. (n.d.). Acoustical Analysis of Spectral and Temporal Changes in Emotional Speech. *ISCAA*.