FIA 2020/22
XII CONGRESSO/CONGRESO IBEROAMERICANO DE ACÚSTICA
XXIX ENCONTRO DA SOCIEDADE BRASILEIRA DE ACÚSTICA - SOBRAC
Florianópolis, SC, Brasil

Speech Feature Extraction for Emotion Recognition Using Machine Learning

dos Santos, A. N.[1]**; dos Reis, V. A.**[2]**; Masiero, B. S.**[3]

[1,2,3]Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas (Unicamp), Campinas, SP, Brasil
[1] a264372@dac.unicamp.br, [2] v261503@g.unicamp.br, [3] masiero@unicamp.br

Abstract

In machine learning (ML) supervised classification problems, it is often beneficial to crunch down the input data, mapping it to an initial pre-processing stage, to improve the performance of pattern recognition systems, such as artificial neural networks (ANNs). But the success of this kind of approach relies on a viable choice of features to best represent the essence of the input data. In this scenario, most ML sound applications make use of acoustic models, but, regarding *machine hearing*, this works better if some basic properties of the human hearing are considered, such as the variable widths of cochlear critical bands with respect to frequency. In this work, some techniques of *cepstral* feature extraction are investigated, based on linear predictive coding (LPC), which can model the source-filter behavior of glottal speech, and short-time Fourier transform (STFT), using triangular and *gammatone* shaped filter banks to warp the short-time power spectrum into different scales of frequency based on human auditory perception. In addition to it, a dimensional reduction of these representations is further accomplished by applying a discrete cosine transform (DCT). Subsequently, these feature extraction techniques are applied to a speech-only portion of an emotion recognition data-set, serving as a front-end to an ML model. This back-end consists of a shallow and fully connected architecture, known as linear classifier, which has just one input layer and one output layer, and is regularized via the least squares (LS) method. Finally, each model obtained is compared for accuracy. In our validation experiments, the *gammatone cepstral* coefficients (GTCC) was the most prominent front-end, with 55% overall accuracy. However, in our test experiments, the Mel frequency *cepstral* coefficients (MFCC), linear predictive *cepstral* coefficients (LPCC) and Bark frequency *cepstral* coefficients (BFCC) features resulted in improved accuracy, with 98%, 94% and 94%, respectively.

Keywords: cepstral features, speech emotion recognition, linear predictive coding, artificial neural networks, regularized least-squares.

PACS: 43.60.Ek, 43.60.Lq, 43.60.Np.

1. INTRODUCTION

Apart from end-to-end (E2E) approaches, i.e., when raw data is presented to a deep learning (DL) model, which, in turn, learns feature representations within layers of abstraction, machine learning (ML) models based on artificial neural network (ANN) architectures usually require a front-end. In this scenario, digital signal processing (DSP) techniques are often employed to crunch down the input data, extracting underlying features using carefully engineered methods of analysis and/or synthesis, that are tailored to the type of signals under consideration. For audio classification problems, e.g., music information retrieval (MIR), speaker recognition (SR), speech emotion recognition (SER) etc., there are several feature extraction methods available on various programming language libraries and toolboxes, e.g., MATLAB, Python, R etc., based on both time- and frequency-domain representations, pre-processing, data augmentation, etc. All of which serves to better represent a data-set to an ANN architecture, that will, in turn, learn a training paradigm, i.e., recognize patterns in a supervised, semi-supervised or even in an unsupervised manner [1–3].

Figure 1 illustrates a flowchart, in which the left-hand column represents the front-end (data preparation), and the right-hand column represents the so-called back-end (data modeling).

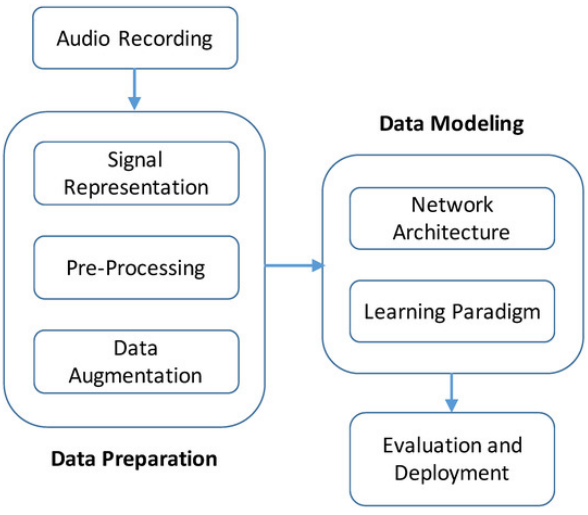

Figure 1: Typical flowchart of an ML-based audio classifier [4].

It's arguable that the scheme illustrated in Figure 1 is best suited for shallow architectures, since deeper models can learn feature representations directly from its layers' transformations. However, DL models often lack *explainability*, which may be an important factor, depending on the application being considered. On the other hand, feature engineering offers more control over what's happening to a data-set when being pre-processed, which also makes it more convenient for pre-processing smaller and/or poor-quality data-sets [5].

That being the case, this work studies a practice to evaluate the impact of using different computational features, separately and/or concatenated, as front-end to an audio classifier. Thereunto, we chose to work with a speech-only portion of an emotion recognition data-set, the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS), which comprises a small set of only 1440 speech samples, appurtenant to 8 classes of emotions [6].

Since SER deals with the classification of speech signals according to affection, it's pertinent to use features based on human speech, e.g., the linear predictive coding (LPC) source-filter model of glottal speech, and human hearing, e.g., short-time signal representations warped into frequency scales that are inspired by the variable widths of cochlear critical bands. After choosing a set of features, carefully analyzing their computational algorithms, and applying a dimensionality reduction technique to spare computational cost, they're employed as a pre-processing stage for an ML model, which is based on a shallow ANN architecture, known as linear classifier, thus enabling to evaluate their effectiveness based on model performance.

The remainder of this paper is organized as follows: Section 2 describes the main theoretical foundations of this work, Section 3 details the development of this work, Section 4 presents results with some discussion, and, finally, Section 5 presents pertinent considerations to conclude this study.

2. FUNDAMENTALS

In this section, the main theoretical foundations behind the speech- and auditory-based features

used in this work are presented. To exemplify each algorithm's steps, an audio sample taken from the Open Speech Repository [7] is used, which comprises a 3.5s clip of a female person speaking at a sampling frequency of 8kHz.

Figure 2 illustrates a time-frequency (T-F) representation of this audio clip.

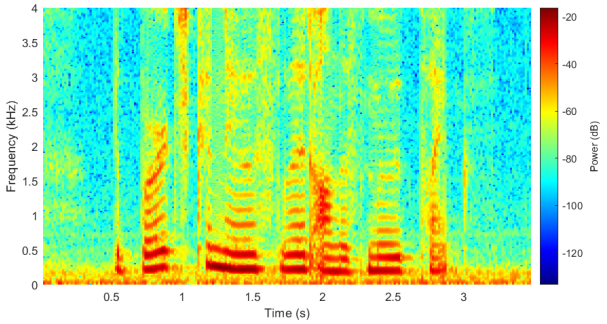

Figure 2: Audio clip's T-F representation.

To extract underlying features from this audio clip, a pre-processing stage is required by both speech- and auditory-based features. Since voiced sounds naturally have a steep roll-off in the high frequency region, a pre-emphasis filter can be used to promote balance in the spectrum, by emphasizing the higher frequency components using

$$H(z) = 1 - \alpha z^{-1}, \tag{1}$$

where $H(z)$ is a transfer function in the z-domain, and α is a value used to control the filter's slope, which is usually set between 0.4 and 1.0 [8, 9].

Figure 3 illustrates a (T-F) representation of the same audio clip, after pre-emphasis.

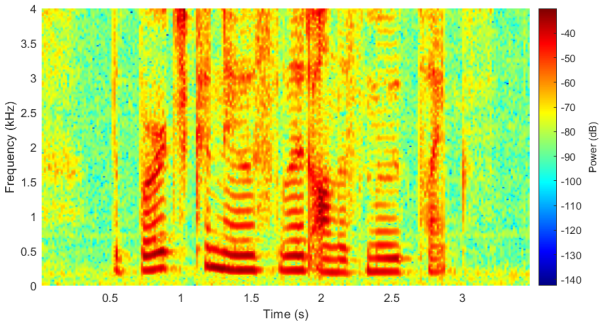

Figure 3: Audio clip's T-F representation, after pre-emphasis.

Since the characteristics of voiced signals can be considered fixed for quasi-stationary time interval periods, framing is also needed, with overlapping hops to smooth the transition between each frame. Moreover, a time-window function is also required, to attenuate the spectral leakage caused by signal truncation [8, 9].

This process, known as short-time analysis, is illustrated in Figure 4, where a bell-shaped function, e.g., a Hanning window, is multiplied, point-by-point, by a pre-emphasized audio waveform, generating windowed segments at each overlapping hop.

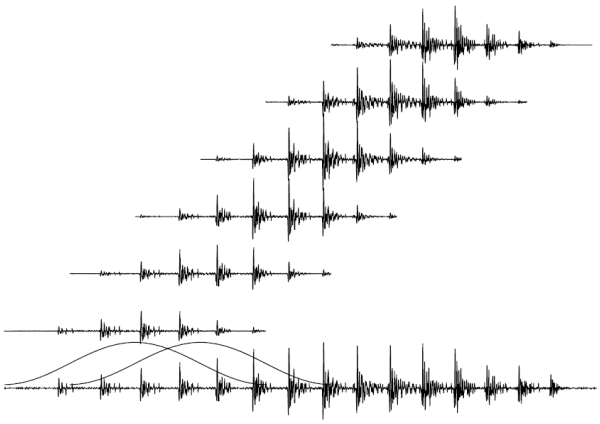

Figure 4: Short-time analysis scheme [5].

All spectrograms in this section were computed using a 25 ms Hamming window, with 10 ms overlaps, and a 512 points, one-sided, short-time Fourier transform (STFT). After this aforementioned pre-processing stage, both speech- and auditory-based feature extraction techniques are readily employed, as discussed next.

2.1 Speech-based features

LPC is a widely used method of audio signal processing that can be used to represent spectral envelopes via a linear predictive model, where a discrete-time sample can be estimated by a linear combination of the previous samples, in the form of an infinite impulse response (IIR) filter, according to

$$H(z) = \frac{G}{A(z)} = \frac{G}{1 - \sum_{m=1}^{p} a_m z^{-m}}, \tag{2}$$

where $H(z)$ is a transfer function in the z-domain, G is the filter gain, a_m is a set of auto-regressive (AR) linear coefficients, and p is the filter order. After analyzing the original signal, it becomes possible to synthesize it by means of its

frequency response. Hence, this approach can be used to model the source-filter filter behavior of glottal speech, under the assumption of stationarity. The basic idea is that, given a speech sample, the LPC analysis can be used to achieve an approximate model of the vocal tract, in the form of an all-pole system, as in Equation 2. The LPC synthesis of voiced sounds can be accomplished by using an impulse train as input signal to $H(z)$, which mimics glottal pulse. For unvoiced sounds, random noise can be used as input signal [10].

Of all algorithms available to compute G and $A(z)$, the auto-correlation method is a popular one, and was used in our experiments. It explores the Toeplitz structure of the input signal's auto-correlation matrix, and uses the Levinson-Durbin recursion to reduce computational cost, as described in [11, 12].

Figure 5 illustrates the LPC filter coefficients computed via this auto-correlation analysis, for the same audio clip as before, while Figure 6 illustrates a T-F representation of the LPC synthesis obtained via the computed $H(z)$.

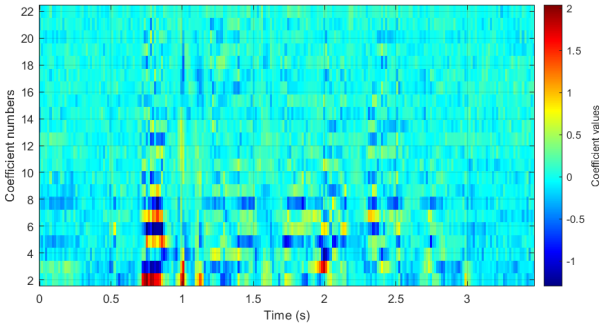

Figure 5: Audio clip's LPC analysis coefficients.

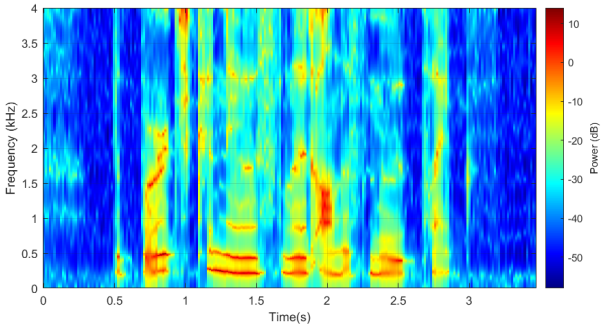

Figure 6: Audio clip's LPC synthesis T-F representation.

Since the LPC analysis coefficients are highly correlated, which could be problematic for ML applications, this feature representation is further transformed into the *cepstral*-domain, which is an extension of the spectral-domain, that was originally proposed by [13].

The basic idea is that, just as a convolution operation in time-domain becomes a simple multiplication in the frequency-domain, it can further become a simple addition in the *quefrency*-domain. The terms *cepstrum* and *quefrency* are derived from the english words spectrum and frequency, with its prefixes shuffled for distinction [5].

The linear predictive coding *cepstral* (LPCC) coefficients can be derived from a set of non-linear recursive equations:

$$c_0 = \ln G^2, \tag{3}$$

$$c_m = a_m + \sum_{k=1}^{m-1} \frac{m-k}{m} a_k, \quad 1 \leq m \leq p, \tag{4}$$

$$\text{and } c_m = \sum_{k=1}^{p} \frac{m-k}{m} a_k, \quad p < m < n, \tag{5}$$

where c denotes the *cepstral* coefficients, G is the filter gain, a denotes the LPC analysis coefficients, and p is the LPC filter order [14].

Figure 7 illustrates the LPCC coefficients computed using this set of equations.

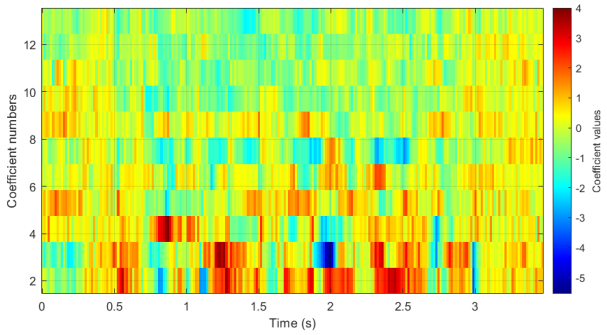

Figure 7: Audio clip's LPCC coefficients.

2.2 Auditory-based features

When a linear frequency scale spectral feature is warped into an auditory-based frequency scale, e.g., Mel, Bark, Equivalent Rectangular Bandwidth (ERB) etc., it becomes an auditory-based feature.

The STFT is a popular spectral feature for both

ML and DL applications, and is obtained by applying an N-point discrete Fourier transform (DFT) to each windowed segment of a short-time analysis, as described before. To warp an STFT representation into different auditory-based frequency scales, filter-banks are required, with each filter's center frequency (CF) spaced using a different scale.

The Mel scale of frequency is approximately linear below 1 kHz, and logarithmic above that. Conversion from physical frequency to Mel is possible using

$$f_{Mel} = 2595 \times \log_e \left(\frac{f_{Hz}}{700} \right), \qquad (6)$$

where f_{Mel} denotes the perceived frequency, f_{Hz} denotes the physical frequency, 2595 is a scaling factor, and 700 the break frequency [5, 8, 9].

Alternatively, the Bark scale of frequency is obtained using

$$f_{\text{Bark}} = 6 \times \sinh^{-1} \left(\frac{f_{\text{Hz}}}{600} \right), \qquad (7)$$

where f_{Bark} denotes the perceived frequency, f_{Hz} denotes the physical frequency, 6 is a scaling factor, and 600 the break frequency [15].

Similarly to Equation 6, the ERB scale of frequency is obtained by replacing the scaling factor and break frequency values using

$$f_{ERB} = 24.7 \times \log_e \left(\frac{f_{Hz}}{228.8} \right). \qquad (8)$$

Moreover, the Mel- and Bark-based filter-banks use triangular shaped filters, while the ERB-based filter-bank uses *gammatone* shaped filters, which are considered a better approximation of the human auditory system.

The *gammatone* filter impulse response can be described as

$$g(t) = at^{n-1} e^{-2\pi bt} \times \cos\left(2\pi f_c t + \varphi\right), \qquad (9)$$

where t is time in seconds, a is an amplitude factor, n is the filter order, f_c is the filter's center frequency, b is the filter's bandwidth, and φ is a phase factor [5].

Figure 8 illustrates three filter-banks, each one designed using one of the aforementioned scales of frequency, and with 40 channels (or bands).

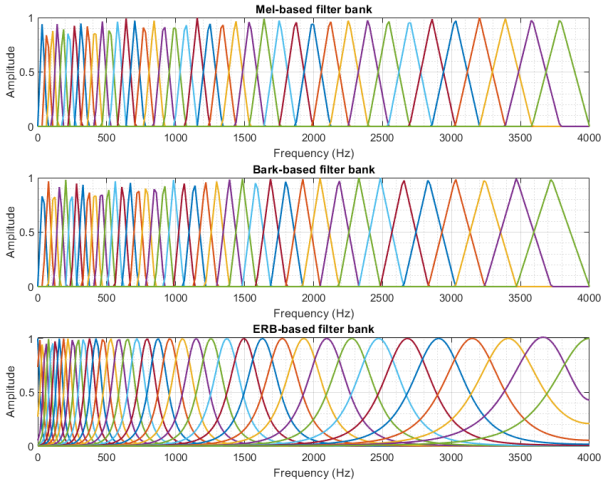

Figure 8: Mel-, Bark- and ERB-based filter banks.

By multiplying the spectrogram in Figure 3 by each of these different filter-bank designs, their respective Mel-, Bark- and ERB-based spectrograms are obtained, as illustrated in Figure 9.

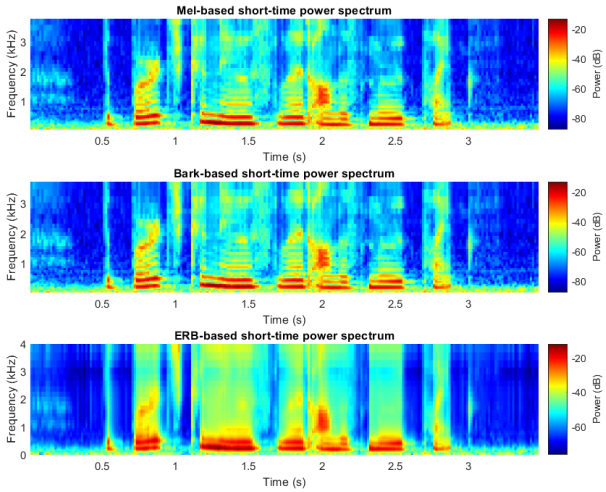

Figure 9: Mel-, Bark- and ERB-based spectrograms.

Since these T-F representations illustrated in Figure 9 are highly dimensional, a type-II DCT is applied to decorrelate them, while also reducing their dimensionality.

This further transforms these Mel-, Bark- and ERB-based spectrograms into what are know as Mel-frequency *cepstral* coefficients (MFCC), Bark-frequency *cepstral* coefficients (BFCC), and *Gammatone cepstral* coefficients (GTCC).

Usually, only 8 to 13 *cepstral* coefficients are kept after this transformation, because higher or-

der coefficients only represent really fast changes in speech signals, which carries little relevant information [8, 9].

Figure 10 illustrates these representations.

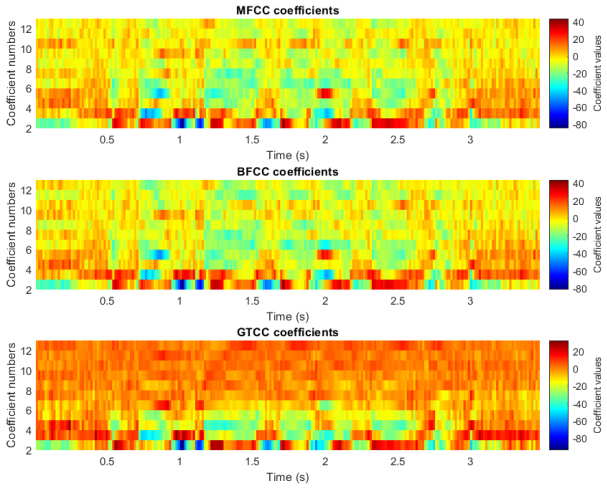

Figure 10: MFCC, BFCC and GTCC coefficients.

Ultimately, these LPC, LPCC, MFCC, BFCC and GTCC coefficients were used as front-end in our experiments, with $\alpha = 0.97$, $Win\_Length = 20ms$, $Overlap = 10ms$, $DFT\_Size = 1024$, $p = 23$, according to the *de facto* used dataset.

Figure 11 illustrates this whole process of feature extraction.

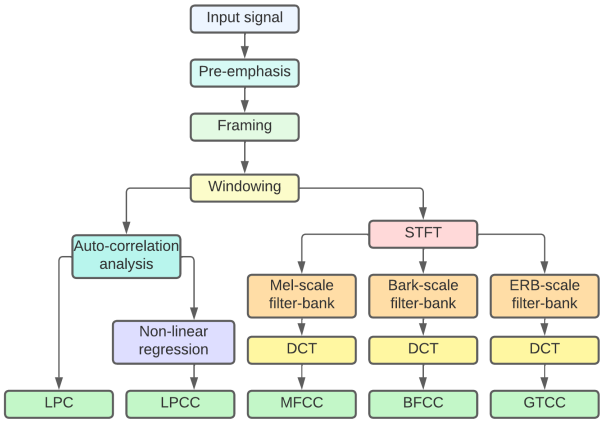

Figure 11: Complete feature extraction flowchart.

3. DEVELOPMENT

This experiment was developed upon programming and numeric computing platform MATLAB (version 2020a), and RAVDESS, a well known data-set of emotional speech and song.

Its speech-only portion comprises 1440 samples performed by 24 actors (12 male and 12 female), vocalizing 2 lexically-matched statements ("*kids are talking by the door*" and "*dogs are sitting by the door*") in a neutral North American accent.

Speech emotions include calm, happy, sad, angry, fearful, surprise, and disgust expressions.

Each expression is pronounced at 2 different levels of emotional intensity (normal and strong), with an additional neutral expression, resulting in a total amount of 60 trials per actor [6].

Data exploration evinced that each audio file varied its duration in seconds, however, they were approximately 3 s long in majority. Thus, all files were fixed at this duration.

For audio files shorter than 3 s, zeros were padded, at the beginning, and at the end of each time-vector.

For audio files longer than 3 s, it was possible to cut off samples, at both the beginning and at the end, because they simply represented silence.

Since each audio file was originally recorded at a sampling frequency of 48 kHz, with 16-bit resolution, and in .WAV format, each one became a $1 \times 144,000$ row vector, which were all organized as a $1,440 \times 144,000$ input matrix X.

Moreover, a $1,440 \times 8$ target matrix S was created, with one-hot encoding. From the extracted LPC, LPCC, MFCC, BFCC and GTCC features, only 12 coefficients were kept, which resulted in a dimensional reduction of 55.6 times less data points, i.e., from $1,440 \times 144,000$ to $1,440 \times 2,587$.

The back-end of this application consisted of a fully-connected neural network (FCNN) architecture with only one input layer, and one output layer.

To match the input data, the input layer had to be of size $2,587$, and the output layer of size 8, to match its pertaining number of emotions.

To regularize the linear classifier, i.e., to avoid over-fitting, the least squares (LS) method was employed.

Figure 12 illustrates a not-to-scale schematic of this FCNN model.

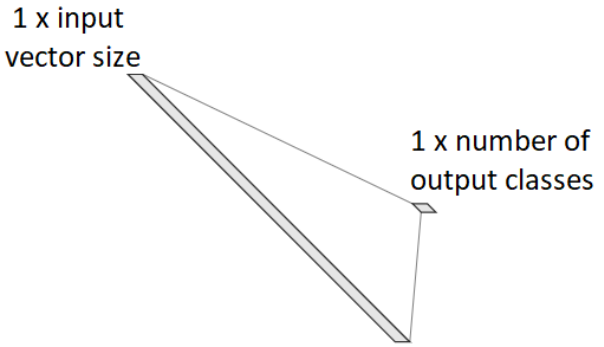

Figure 12: Not-to-scale, back-end ANN architecture.

Since the aim of the linear classifier is to compute a matrix of synaptic weights, that will produce an estimation of the output classes when transposed and multiplied by the input data, the enclosed solution is in the form

$$W = \left(X^T X + \lambda I\right)^{-1} X^T S \qquad (10)$$

where W is the synaptic weights matrix, X is the input data matrix, T denotes matrix transposition, S is the one-hot encoded target matrix, λ is a regularization coefficient that will punish the model for increasing the norm of matrix W, and I is an identity matrix [16–18].

To find the regularization coefficient λ, mean squared error (MSE) was used as loss function, and a broad search for its best value was first conducted between 10^{-10} and 10^{10}, when training on 60% of the input data and validating on another 20%.

Figure 13 illustrates this broad search when using LPC as a sole front-end, limited between 10^{-3} and 10^{3} for better visualization.

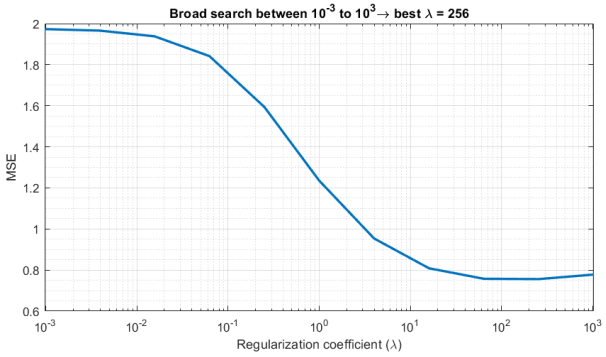

Figure 13: Broad search for λ using LPC as sole front-end, and MSE as loss function.

Since the best λ value, i.e, with lowest MSE, was around 10^2, a narrow search was then conducted around this value, resulting in a better approximation of λ's optimal value.

Figure 14 illustrates this narrow search when using LPC as a sole front-end.

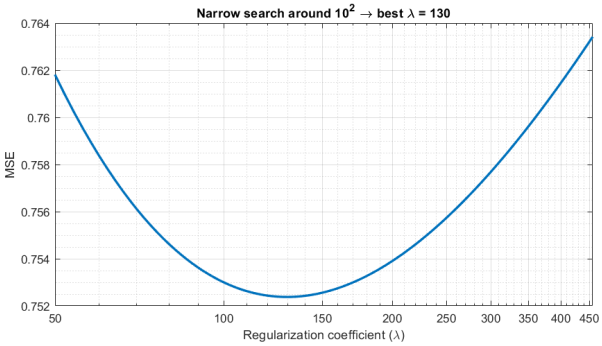

Figure 14: Narrow search for λ using LPC as sole front-end, and MSE as loss function.

To validate whether this regularization would produce the best accuracy, 2 similar broad and narrow searches for λ were also conducted, but using the overall classification accuracy as metric, when training on 60% of the input data and validating on another 20%.

Figure 15 illustrates this broad search when using LPC as a sole front-end, limited between 10^{-3} and 10^{3} for better visualization.

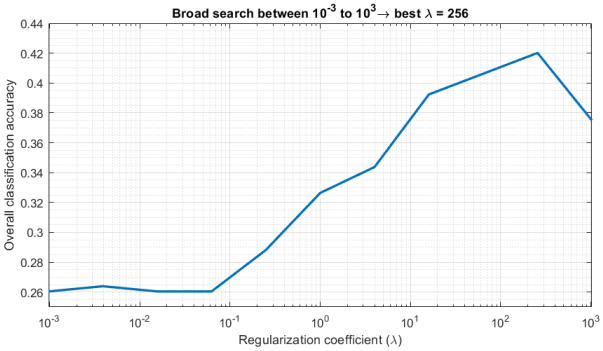

Figure 15: Broad search for λ using LPC as sole front-end and overall accuracy as metric.

Since the best λ value, i.e, with highest overall classification accuracy, was also around 10^2, a narrow search was then conducted around this value, resulting in a better approximation of λ's optimal value.

Figure 16 illustrates its narrow search counterpart when using LPC as a sole front-end.

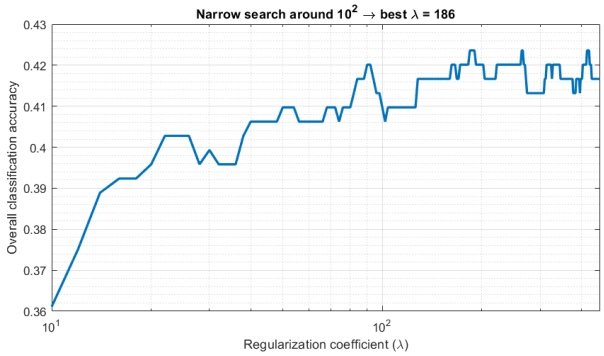

Figure 16: Narrow search for λ using LPC as sole front-end and overall accuracy as metric.

This procedure was repeated for each possible model, i.e., each combination of front-end and back-end. Between training and validation, a random shuffle with normal distribution was performed on the data-set.

Ultimately, the λ value that produced the highest overall classification accuracy was chosen for each front-end and back- end combination, and then compared for validation and test accuracy, as discussed in the next section.

4. RESULTS AND DISCUSSIONS

After obtaining λ for each front-end and back-end combination, each model variant was put up to test, using the remaining 20% of the input data, which was never used during training or validation. Test performance was assessed using confusion matrices, to compute not only overall accuracy, but also accuracy per class, and misclassifications. Figure 17 illustrates the confusion matrix for the LPC model variant, i.e., using LPC features as front-end, and the regularized linear classifier as back-end. The main diagonal (in green shade) shows the accuracy per class (in terms of both samples and %), while the other cells (in salmon color shade) shows the misclassifications. The numbers on the sides of the matrix represent each class, being 1 = neutral, 2 = calm, 3 = happy, 4 = sad, 5 = angry, 6 = fearful, 7 = disgust, 8 = surprise. Matrix's columns represent true classes (target), whilst rows represent the predicted ones (output). At the end of columns, rows and the main diagonal, an overall accuracy % is presented in a green font, and an overall misclassification % in a red font.

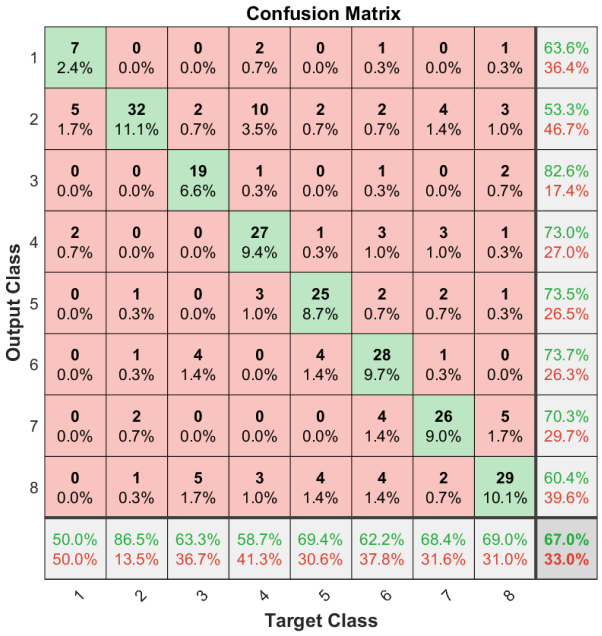

Figure 17: Confusion matrix for the LPC model variant.

This procedure was then repeated for every model variant, and the GTCC model variant was ranked the best performer (with with 55% overall validation accuracy and 84% overall test accuracy), followed by the MFCC model variant (with with 53% overall validation accuracy and 98% overall test accuracy), the BFCC model variant (with with 52% overall validation accuracy and 94% overall test accuracy), the LPCC model variant (with with 46% overall validation accuracy and 95% overall test accuracy), and the LPC model variant (with with 42% overall validation accuracy and 67% overall test accuracy). However, different combinations of front-end features were also tested, to investigate whether it would produce even better results. Figure 18 illustrates these results in a descending rank.

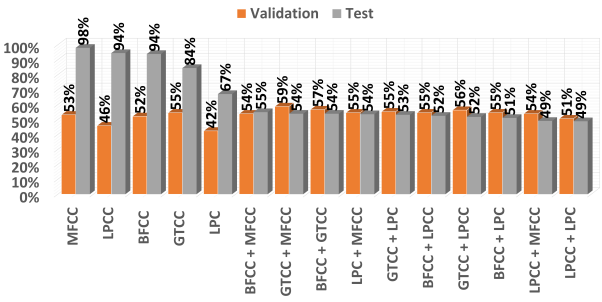

Figure 18: Overall validation and test accuracy for each model variant.

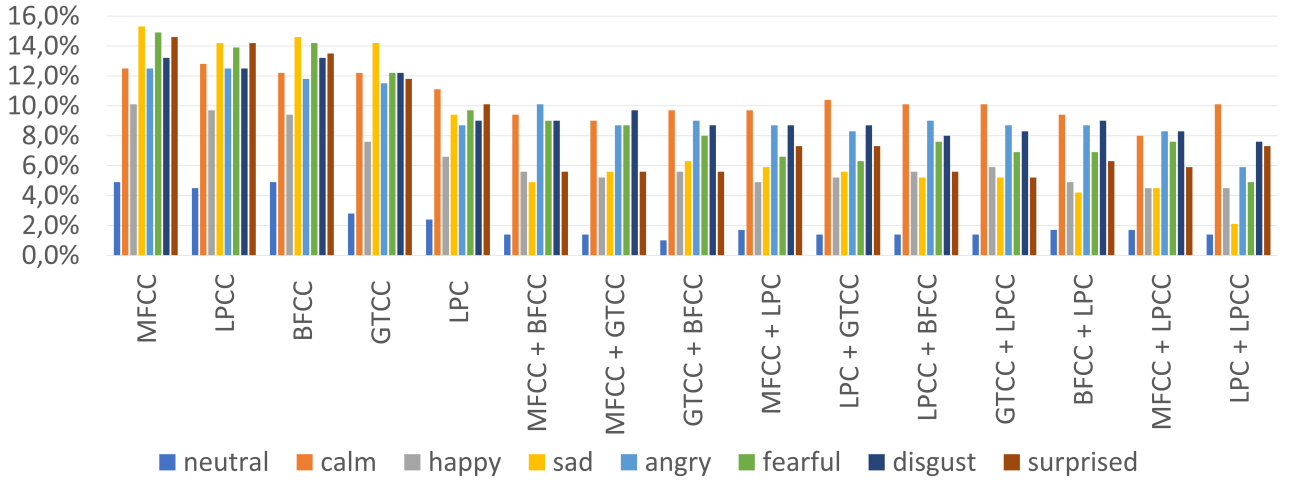

Figure 19: Per class test accuracy for each model variant.

From Figure 18, it's evident that, even though the GTCC model variant was the most prominent one in the validation process, some models had an accuracy leap in the test process. By ranking their performances in a descending order, it's noticeable that, combinations of 2 different feature extraction methods were not as fruitful as sole ones.

By analyzing the overall test accuracy rank, it's clear that the MFCC model variant performs better than any other, followed by the LPCC and BFCC model variants, which indicates that both speech- and auditory-based features are fit for speech emotion recognition, since they model not only the way a person speaks, but also the way another person listens to them speaking.

Figure 19 illustrates the test accuracy per class for each model variant studied, revealing that, for most models, the calm expression is most easily recognizable. As for the top 3 models, they tend to favor calm, sad, fearful and surprised expressions.

RAVDESS was released in 2018. In contrast with recent studies, Xu et al. [19] achieved a 77.8% test accuracy in 2021, by using an MFCC-based front-end with a back-end that combines a convolutional neural network (CNN) architecture with an attention mechanism called head-fusion. Also in 2021, Kwon [20] achieved a 95% test accuracy by combining a spectrogram-based front-end with a hybrid CNN and long short-term memory (LSTM) back-end. And, in 2022, Puri et al. [21] achieved a 98% test accuracy by using a front-end that combines MFCC, chromagram, and Mel frequency spectrograms with a CNN-based back-end. Our work was carried out in 2020, but is only seeing the light of day in 2022, due to the COVID-19 pandemic.

Ultimately, it's important to mention that this dataset is far from being a general representation of the human population. Since it's samples were acquired from subject's with same nationality, lacking accent diversity, ethnical diversity, gender diversity, different languages, disability inclusion etc., when put to test with samples different from it's own *corpus*, it will hardly produce exciting results, such as in Figures 18 and 19.

5. CONCLUSION

In this paper, a practice to evaluate the impact of using different computational features, separately and/or concatenated, as front-end for an audio classifier was proposed.

Since focusing on speech emotion recognition (SER), a set of speech- and auditory-based features were studied, carefully analysing their algorithms, and exemplifying each of their application steps.

To evaluate whether they contribute more or less in solving the audio classification problem at hand, a simple FCNN was employed, comprising only one input layer and one output layer. Model regularization was performed by the least squares (LS) method, with no backpropagation.

For each model variant, i.e., for each combination of front-end and back-end, overall and per class accuracy tests were performed, in a feature-ablation manner, per say. This ultimetely reavealed that both speech- and auditory-based features are fit to SER, and the MFCC, LPCC and BPCC model variants were the most prominent in our experiments.

However, this does not evince that the FCNN model used is best fitted to SER, but rather indicates that further experiments with larger datasets and more complex ANN architectures are demanded, since it's a non-linear problem.

Finally, the SER problem, at its own, remais open.

6. ACKNOWLEDGMENTS

This work was partially supported by the São Paulo Research Foundation (FAPESP), grant #2019/22795-1. The opinions, hypothesis and conclusions or recommendations expressed in this material are the authors' responsibilities, and not necessarily reflect FAPESP's views.

REFERENCES

[1] Simon Haykin. *Neural networks and learning machines, 3/E*. Pearson Education India, 2009.

[2] Eric J Humphrey, Juan P Bello, and Yann LeCun. Feature learning and deep architectures: New directions for music informatics. *Journal of Intelligent Information Systems*, 41(3):461–481, 2013.

[3] V Kishore Ayyadevara. Pro machine learning algorithms. *Berkeley: Apress*, 2018.

[4] Jakob Abeßer. A review of deep learning based methods for acoustic scene classification. *Applied Sciences*, 10(6), 2020.

[5] Richard F Lyon. *Human and machine hearing*. Cambridge University Press, 2017.

[6] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018.

[7] The open speech repository. URL https://bit.ly/3zhJVOH.

[8] K Sreenivasa Rao, V Ramu Reddy, and Sudhamay Maity. *Language identification using spectral and prosodic features*. Springer, 2015.

[9] Haytham Fayek. Speech processing for machine learning: Filter banks, mel-frequency cepstral coefficients (mfccs) and what's in-between. 2016. URL https://bit.ly/3wVFhEg.

[10] Juan I Godino-Llorente, Santiago Aguilera-Navarro, and Pedro Gómez-Vilda. Lpc, lpcc and mfcc parameterisation applied to the detection of voice impairments. In *Sixth International Conference on Spoken Language Processing*, 2000.

[11] Lawrence R Rabiner. *Digital processing of speech signals*. Pearson Education India, 1978.

[12] Behnam Fazlali and Mohammad Eshghi. A pipeline design for implementation of lpc feature extraction system based on levinson-durbin algorithm. In *2011 19th Iranian Conference on Electrical Engineering*, pages 1–5. IEEE, 2011.

[13] Bruce P Bogert. The quefrency alanysis of time series for echoes; cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. *Time series analysis*, pages 209–243, 1963.

[14] Panos E Papamichalis. *Practical approaches to speech coding*. Prentice-Hall, Inc., 1987.

[15] Abel Herrera and Fernando Del Rio. Frequency bark cepstral coefficients extraction for speech analysis by synthesis. *The Journal of the Acoustical Society of America*, 128(4):2290–2290, 2010.

[16] Richard Bronson. *Schaum's Outline of Matrix Operations*. McGraw Hill Professional, 1988.

[17] Gilbert Strang. *Linear algebra and its applications*. Belmont, CA: Thomson, Brooks/Cole, 2006.

[18] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[19] Mingke Xu, Fan Zhang, and Wei Zhang. Head fusion: improving the accuracy and robustness of speech emotion recognition on the iemocap and ravdess dataset. *IEEE Access*, 9:74539–74549, 2021.

[20] Soonil Kwon. Optimal feature selection based speech emotion recognition using two-stream deep convolutional neural network. *International Journal of Intelligent Systems*, 36(9):5116–5135, 2021.

[21] Tanvi Puri, Mukesh Soni, Gaurav Dhiman, Osamah Ibrahim Khalaf, Ihtiram Raza Khan, et al. Detection of emotion of speech for ravdess audio using hybrid convolution neural network. *Journal of Healthcare Engineering*, 2022, 2022.