A University of Waterloo White Paper
August 2023

Audio Analysis in R: A Deep-Dive into Emotion Recognition

UMAID MUHAMMAD ZAFFAR
UWATERLOO ID: 2096345
DATASCI701
SUPERVISED BY: MICHAEL JOHN DAVIS

Table of Contents

Executive Overview

In the rapidly evolving landscape of machine learning and audio processing, the capability to recognize emotions from audio data has transformative potential. This study embarked on an expedition to understand and optimize this capability, using the R language as our primary analytical tool. Using R, we transformed raw audio data into insightful features that resonated with emotion-related cues. Three pivotal machine learning models - SVM, KNN, and XGBoost - were employed, with a particular emphasis on the art and science of feature selection.

Preliminary findings underline the distinct preferences of each model; KNN favored lower-dimensional data, SVM thrived with a richer feature set, and XGBoost, while promising, posed challenges that warrant further exploration. Crucially, the study illuminated the importance of specific features, notably BFCC, spectral contrast, and RMSE, in deciphering emotions from audio. However, it's salient to mention that certain features like torretz and chroma, known for their pertinence in emotion recognition, remained outside our grasp due to R's limitations. This revelation not only emphasizes the challenges faced but also underscores the potential enhancements the R ecosystem could introduce.

In essence, this expedition not only charted the territories of emotion recognition in R but also illuminated potential pathways for future explorations and enhancements.

Introduction

Emotion, as an intricate facet of human experience, influences interactions, decision-making processes, and the very nuances of daily communication. Historically, emotional studies were largely observational and qualitative. However, technological advancements have shifted the focus to quantitative and objective measures, fuelling the evolution of emotion recognition through auditory and visual cues [14]. This transition marked a significant move from basic sentiment analysis of text to a profound understanding of spoken emotional subtleties [4].

Today, emotion recognition finds itself pivotal across various sectors. From potential breakthroughs in mental health diagnostics to refining user experiences in media and reshaping human-computer interactions, its applications seem boundless [1][2][3]. Nevertheless, the challenges are tangible. Cultural variances, linguistic subtleties, and the inherent complexity of emotions challenge the precision of existing tools, calling for refined algorithms and technologies [15].

Python, with its rich libraries and AI capabilities, has largely spearheaded these advancements. In contrast, the capabilities of R, primarily celebrated for its statistical prowess, remain untapped in this domain. This whitepaper navigates this uncharted territory, exploring the potential of R in emotion recognition from

audio signals. Readers can anticipate insights into methodologies, results, and future prospects, culminating in a comprehensive perspective on harnessing R for audio-based emotion analytics.

Background

R, originating in 1993, has matured as a comprehensive tool for statistical computing and graphics, earning its reputation as a stalwart in data analytics and research realms [5]. Over the years, R's expanding package ecosystem has catered to a myriad of applications, ranging from intricate statistical analyses to machine learning, and notably, audio analysis.

Central to R's escalating acclaim among researchers and data analysts are its innate proficiencies in statistical modelling, machine learning, and data visualization. These attributes render it a formidable tool for intricate tasks like audio analysis, and more specifically, emotion recognition [6][7]. Packages in R like "*seewave*" and "*tuneR*" provide researchers with refined capabilities for intricate manipulation, comprehensive analysis, and intuitive visualization of multifaceted audio data [8][9].

Automated Emotion Recognition & The Rise of SER

Within the expansive domain of Automated Emotion Recognition (AER) [16], Speech Emotion Recognition (SER) holds a distinctive position, intricately interweaving paradigms from Automatic Speech Recognition (ASR) [17][18][19][20]. This connection can be seen in the common signal types and similar feature extraction methods, which are then enhanced by various machine learning techniques. Deep Learning (DL) methodologies, originally recognized for their prowess in Natural Language Processing (NLP) [21], find a critical application in SER, attributed to the shared sequential structure of data.

In the historical arc of SER, techniques like Mel Frequency Cepstral Coefficients (MFCC) [22][28]—primarily associated with ASR—have found successful integration into emotion recognition frameworks, significantly enhancing classification and pattern identification capabilities. The innovation extends beyond with the adoption of DL architectures such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), both originating from ASR and NLP, to address SER challenges.

Future Prospects & Applications

Projecting into the future, SER's significance becomes increasingly apparent, promising groundbreaking strides in domains like Human Computer Interaction (HCI), Human Machine Interaction (HMI), and Human Robot Interaction (HRI) [23]. With the looming wave of the Internet of Things (IoT), the power of SER in affective computing promises to transform how we interact with ubiquitous ambient intelligence [24]. Moreover, the clinical implications of SER are undeniable, as illustrated by its potential in automated depression detection through subtle emotional cues in speech [25].

Trends & Evolution

The momentum in SER research has only accelerated in recent years, primarily propelled by the advent and refinement of deep learning methodologies. TDNNs and X-Vector based architectures have crystallized as benchmark methodologies, pushing the frontiers of emotion recognition from speech [17][20][21]. The synergy of classical audio signal processing with deep learning has sharpened emotion prediction capabilities [17]. Contemporary research also beckons towards multi-modal emotion recognition, integrating audio with visual markers, aiming for a comprehensive insight into human emotional expression [4][11][15][17]. Mirroring broader AI trends, the incorporation of transfer learning and pre-trained models has elevated model efficiency without compromising on accuracy [26]. While the research trajectory holds promise, challenges—primarily model generalization across cultural and linguistic diversities—remain.

Classical Algorithms in SER

While the allure of deep learning is undeniable, a substantive research corpus in SER remains rooted in classical machine learning. Emphasis on robust feature extraction, harnessing techniques like MFCC, chroma features, spectral contrast, and tonnetz, ensures a meticulous capture of loudness, pitch, and frequency nuances [22][28]. The probing into prosodic features, which envelop pitch, tempo, and volume, unveils the intricate emotional tapestry woven within speech [27][28]. Classical algorithms, spanning Support Vector Machines (SVMs), Random Forests, to Gradient Boosting Machines, have consistently demonstrated commendable accuracies when paired with these feature sets [10]. These methods not only ensure interpretability but also serve as potent alternatives in contexts where deep learning might be unfeasible [29].

Audio Emotion Recognition in R

The niche of emotion recognition in R, nested within the broader landscape of audio analysis, intricately marries machine learning with advanced signal processing techniques. The objective: To extract, decipher, and subsequently classify latent emotions embedded in the cadence of human speech. By focusing on nuanced auditory cues - the rise and fall in pitch, the subtle shifts in tempo, or the undulating volumes, this cutting-edge technology paints a richer, more authentic portrait of a speaker's underlying emotions.

While R's foray into emotion recognition is relatively nascent, initial investigations appear promising. Zaman et al. (2021), for instance, embarked on a journey through R's labyrinthine capabilities for emotion recognition. Their findings elucidated how R, equipped with its potent fusion of statistical modelling and machine learning, can emerge as a formidable contender in the pursuit of decoding human emotions [10].

Comparison with Python

On the flip side, Python, with its expansive libraries and a robust community, has prominently spearheaded emotion recognition research in recent years [11][12][13]. Its repertoire, enriched by powerful packages like "Librosa" for audio processing and giants like "Tensorflow" and "PyTorch" for machine learning, has undeniably set high benchmarks in the field [13]. Although R does offer analogous tools, like "audio", “tuner”, “seewave”, "kerasR", and "mxnet", the direct translation of specific audio features available in Librosa to R's context remains a challenge. Researchers venturing into R's domain may occasionally find themselves navigating these gaps, resorting to custom solutions or intricate workarounds to emulate Python's functionality.

Methodology

Introduction

This section elucidates the methods employed to explore the capacities of the R programming language in the realm of emotion recognition from audio signals. It furnishes comprehensive details on data sourcing, preprocessing techniques, feature extraction methodologies, Exploratory Data Analysis using PCA, model selection and training, and the ensuing evaluation metrics.

Data Collection

The data used in this study was sourced from CREMA-D, a publicly available repository comprising of 7,442 original clips from 91 actors of 6 distinct emotions (Anger, Disgust, Fear, Happy, Neutral, and Sad) and four different emotion levels (Low, Medium, High, and Unspecified)., each sampled at a 16 kHz frequency. These clips were from 48 male and 43 female actors between the ages of 20 and 74 coming from a variety of races and ethnicities (African, America, Asian, Caucasian, Hispanic, and Unspecified). Actors spoke from a selection of 12 sentences. For the purpose of this study, emotion levels were not considered.

Data Preprocessing:

The initial step in data preprocessing involved trimming each audio clip to remove starting and ending silences. This was accomplished by applying the **noSilence()** method from the *seewave* package to filter out very low frequencies typically associated with silence. After this trimming, background noise was removed using *seewave’s* **rmnoise()** method, ensuring a cleaner and more consistent set of audio samples for further analysis.

Feature Extraction

Fundamental Frequency (F0) Analysis:

Definition: The fundamental frequency, often denoted as F0, represents the base frequency in speech arising from the periodic oscillations of the vocal cords. It provides an insight into the perceived pitch of a speaker's voice. Shifts in F0 play a pivotal role in the tonal quality of spoken words, influencing distinctions such as emotional tone, questioning versus stating, and vocal emphasis.

Choice of feature: Opting for the fundamental frequency as a salient feature is driven by its capability to mirror the emotional contours of speech, making it a robust indicator for vocal emotion recognition tasks.

R implementation: In R, two prominent methods were explored for computing F0: the **FF()** function from the *tuneR* package and the **fund()** function from the *seewave* package, with both being rigorously tested for their efficacy.

1. FF(): For each wave-format audio file, a periodogram was generated, which subsequently served as the input for the **FF()** function. This function yields a distinct fundamental frequency corresponding to each periodogram.
    a. Input: A `Wspec` object encompassing the periodograms.
    b. Output: A `numeric` value representing the fundamental frequency associated with each respective periodogram.

    Advantages:

    - Provides a concise and clear-cut representation of the speech's foundational frequency.
    - Ideal for datasets with minimal temporal variations or when a broad-stroke F0 metric is adequate.
    - Simplifies the representation, making it easy to analyze and compare.

    Use Cases:

    - Suited for applications that require a rapid, holistic estimate of an audio clip's fundamental frequency.
    - Especially valuable in scenarios emphasizing computational speed and efficiency due to its streamlined calculation process.
2. fund(): For each wave-format audio file, the **fund()** function processes it by creating windows of the audio and computes the F0 for each window. As a result, the function produces a two-column

matrix: the first column denotes time (in seconds), while the second illustrates the F0 value (in kHz).

a. Input: A `Wave` object.
b. Output: A two-column `numeric` matrix representing the time in seconds and frequency in kHz.

Statistical Descriptors of F0: Various statistical measures, encompassing the mean, median, standard deviation, minimum value, and maximum value were extracted. Additionally, the 25th and 75th quantiles of the F0 distribution were determined using the `quantile()` function.

Advantages:

- Delivers an in-depth, granular perspective on the temporal evolution of the fundamental frequency.
- Enables the capture of subtle speech variations that can be pivotal for intricate analyses, including fluctuations due to intonation, stress patterns, and emotional context.

Use Cases:

Essential for applications necessitating the observation of dynamic speech shifts, such as in emotion detection, speech synthesis, or during prosodic speech analysis.

Zero Crossing Rate (ZCR) Analysis:

Definition: Zero Crossing Rate (ZCR) denotes the frequency at which a signal changes its sign, i.e., transitions between positive and negative values. Within speech audio analysis, ZCR can provide insights into noise attributes and helps differentiate between voiced and unvoiced segments, also shedding light on the speech's tempo.

Choice of feature: The ZCR is especially relevant for emotion recognition because variations in the rate of zero crossings can correspond to changes in the speech's intensity, speed, or tone. Moreover, high ZCR values often denote noisier segments or voiceless speech, while lower values can indicate voiced segments. Given that emotional states can influence speech speed and intonation, ZCR can be a valuable feature for such analysis.

R implementation: The ZCR for each audio sample was calculated directly using the *seewave* **zcr()** function.

- Input: A `Wave` object.
- Output: A` numeric` value representing the Zero Crossing Rate.

Statistical Descriptors of ZCR: To provide more nuanced insights into the ZCR's characteristics across audio files, additional statistical measures were derived. These included the computation of both the mean and variance of the ZCR.

Spectral Properties:

Definition: Spectral attributes reveal how power is allocated among a signal's various frequency elements. In other words, how energy is spread out over different pitches in a sound. These characteristics depict the balance, shape, and distribution of frequencies and help us understand the 'color' or timbre of a sound, much like how we'd describe the texture or feel of a fabric.

Choice of feature: Spectral characteristics in speech provide intricate insights into its distinct features. The spectral centroid, for instance, can indicate the perceived brightness or sharpness of a sound. Spectral skewness and flatness offer perspectives on the sound's tonal attributes. Meanwhile, spectral entropy

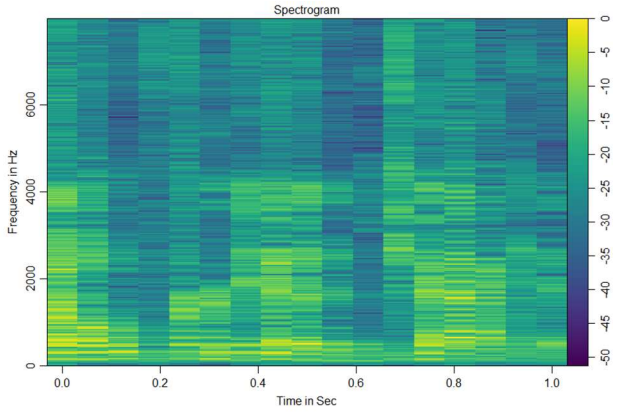

Figure I: Spectrogram of a sample audio

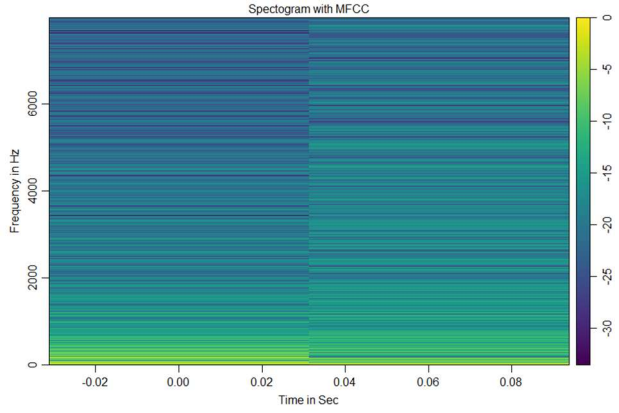

Figure II: Spectrogram with Mel Frequency Cepstral Coefficients of a sample audio

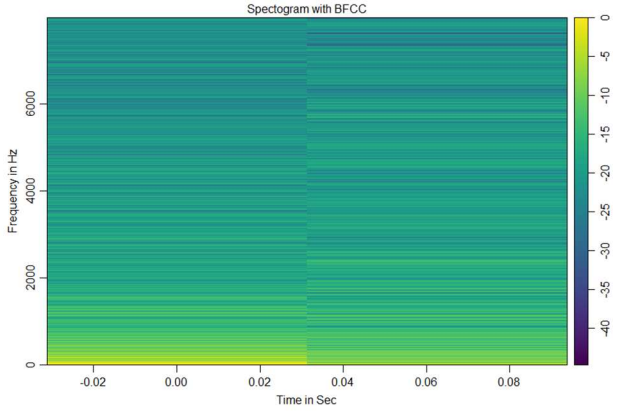

Figure III: Spectrogram with Bark Frequency Cepstral Coefficients of a sample audio

gives us an understanding of the unpredictability or diversity within the pitch variations. Collectively, these spectral attributes provide a clearer view of the nuances in speech's frequency components, making them valuable for applications such as emotion detection. The spectrogram of a sample audio is displayed in Figure I.

R implementation: Spectrograms of audio signals were derived using the *seewave* **spec()** function.

- Input: A `Wave` object.
- Output: A two-column `numeric` matrix representing the frequency in Hz and amplitude.

Specific Spectral Properties: To extract a comprehensive set of spectral characteristics, the *seewave* **specprop()** function was utilized.

- Input: A `Wave` object.
- Output: A named list of spectral characteristics.

The features determined in our study include:

1. Spectral centroid: Represents the "center of gravity" of the spectrum.
2. Spectral skewness: Measures the asymmetry of the spectrum.
3. Spectral flatness: Indicates how noise-like a sound is, compared to being tonal.
4. Spectral peak: Highlights the frequency at which the maximum amplitude is observed.
5. Spectral entropy: Provides insights into the complexity and randomness of the frequency components.
6. Spectral precision: Depicts the clarity and definition of the spectral content.

Mel Frequency Cepstral Coefficients (MFCC):

Definition: Mel Frequency Cepstral Coefficients (MFCC) encapsulate the short-term power spectrum of sound, effectively characterizing the unique tonal quality or timbre of audio signals. Imagine differentiating between the distinct sounds of a piano and a violin, even when they play the same note; MFCCs help in capturing those nuances.

Choice of feature: MFCCs have risen to prominence in various audio processing tasks because they closely mirror the human auditory system's perceptions. By capturing the spectral shape of audio signals derived in the Mel scale, MFCCs provide rich information about temporal changes in sounds. This makes them invaluable in distinguishing subtle variations in speech patterns, making them a favored choice for speech and music recognition applications. A sample spectrogram with MFCCs is displayed in Figure II

R implementation: MFCCs of audio signals were derived using the *tuneR* **melfcc()** function.

- Input: A `Wave` object.
- Output: a matrix of cepstral coefficients per frame per window

Statistical Descriptors of MFCC: For a deeper analysis, the mean and variance of these coefficients was also computed. These statistics offer a broader perspective on the spectral attributes of the sound, facilitating a richer interpretation of its characteristics.

Bark Frequency Cepstral Coefficients:

Definition: BFCCs provide a representation of audio features extracted from perceptual spectrums derived in the Bark scale. Unlike the prevalent Mel scale, which is optimized for modeling speech contents, the Bark scale is focused more on capturing emotional nuances in the audio signal[27].

Choice of feature: The Bark scale, with its emphasis on capturing the emotional essence of audio signals, offers a more granular and sensitive framework for emotion recognition. By leveraging this perceptual spectrum and converting spectral energy values to the Bark scale, one can potentially achieve improved discrimination and classification in emotion-centric tasks. A sample spectrogram with BFCCs is displayed in Figure III

R implementation: To derive BFCCs, the *tuneR* **melfcc()** function was adapted with the "*bark*" argument. This adjustment enables the extraction of cepstral coefficients that are specifically aligned with the Bark frequency scale, presenting a more refined representation than traditional MFCCs.

Statistical Descriptors of BFCC: Both the average and variability of these coefficients were calculated. Examination of these metrics can provide a more comprehensive understanding of the sound's spectral qualities, allowing for a deeper insight into its features.

Spectral Contrast

Definition: Spectral contrast refers to the difference in amplitude between the peaks and valleys in an audio spectrum. In more straightforward terms, it measures the disparity between the loudest and quietest frequency components at any given instance. This insight can be invaluable in identifying the texture and richness of a sound, potentially pinpointing unique sound sources or characteristics.

Choice of feature: The significance of spectral contrast lies in its ability to capture the dynamic range within an audio spectrum. By distinguishing between dense harmonic sounds and sparser ones, it offers a deep dive into the content and quality of the audio signal. Such nuanced information makes spectral contrast an excellent choice for tasks aimed at dissecting the intricacies of audio clips, such as emotion recognition.

R implementation: A direct implementation for spectral contrast is not available in standard R libraries, therefore, a custom function was crafted to compute this measure. The function operates by identifying

the peaks and valleys in the audio's spectrum and calculating the difference in their amplitudes. This function first computes the Fast Fourier Transform and then calculates the spectral contrast via formula.

- Input: A `Wave` object.
- Output: A `numeric` value representing the spectral contrast of the audio.

Root Mean Square Energy (RMSE)

Definition: Root Mean Square Energy (RMSE) is the square root of a signal's average power. For audio signals, it measures the signal's intensity and can serve as an indicator of its loudness or overall energy. A high RMSE value usually indicates a louder or more powerful signal, whereas a lower value suggests a quieter or more subdued segment.

Choice of feature: RMSE is chosen as a feature because it provides a simple yet effective representation of the energy content of a speech segment. Variations in energy can correspond to different emotions or intonations, making it a valuable feature for emotion recognition.

R implementation: While R's standard libraries do not offer an out-of-the-box method to compute RMSE, the following custom approach was adopted: the signal undergoes a Fourier transformation to convert it into the frequency domain. The magnitude of this transformed signal is then squared, averaged, and the square root of this average is taken to compute the RMSE, giving a measure of the signal's overall intensity.

- Input: A `Wave` object.
- Output: A `numeric` value representing the RMS energy of the audio signal.

Principal Component Analysis (PCA):

Introduction to PCA:

PCA was employed to reduce the dimensionality of the feature space while retaining most of the information contained in the data. This approach is beneficial when dealing with high-dimensional data as it can enhance computational efficiency and possibly improve model performance by mitigating the curse of dimensionality.

Data Pre-processing for PCA:

Prior to applying PCA, the data was cleaned, with missing values being addressed, ensuring that the PCA operates only on meaningful feature spaces. Moreover, the data was normalized using the **preProcess()** function from the *caret* package to ensure each feature contributes equally to the distance metric, a crucial step for the PCA algorithm.

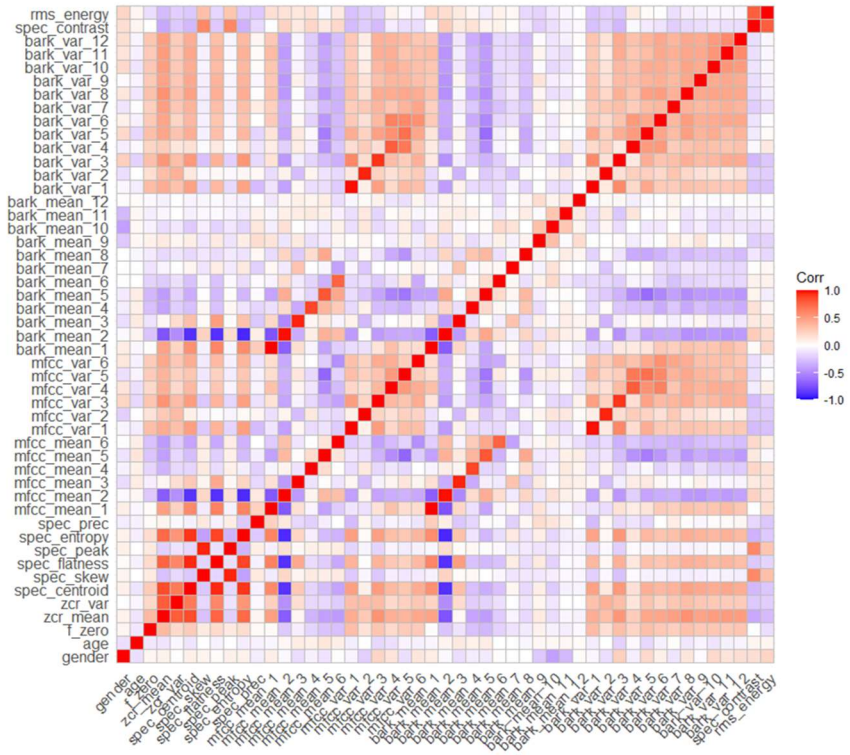

Figure IV: Correlation plot of the complete set of features

Exploratory Data Analysis with PCA

Correlation Matrix Visualization: A correlation matrix was computed to understand the linear relationships between the features. The matrix was visualized using the **ggcorrplot()** function from the *ggcorrplot* package in R to identify highly correlated variables that PCA can help to condense. The correlation matrix is displayed in figure X.

Eigenvalues and Eigenvectors: Using the standard library *stats* function **princomp()**, PCA was applied to the correlation matrix, revealing the eigenvalues and eigenvectors. The eigenvalues represent the amount of variance explained by each principal component, while the eigenvectors denote the direction of each principal component in the original feature space.

Scree Plot Visualization: A scree plot (**fviz_eig()** function from the *factoextra* package) was generated to determine the number of principal components to retain. This plot (figure X) displays the percentage of total variance explained by each principal component. It aids in determining the optimal number of components that capture the most information.

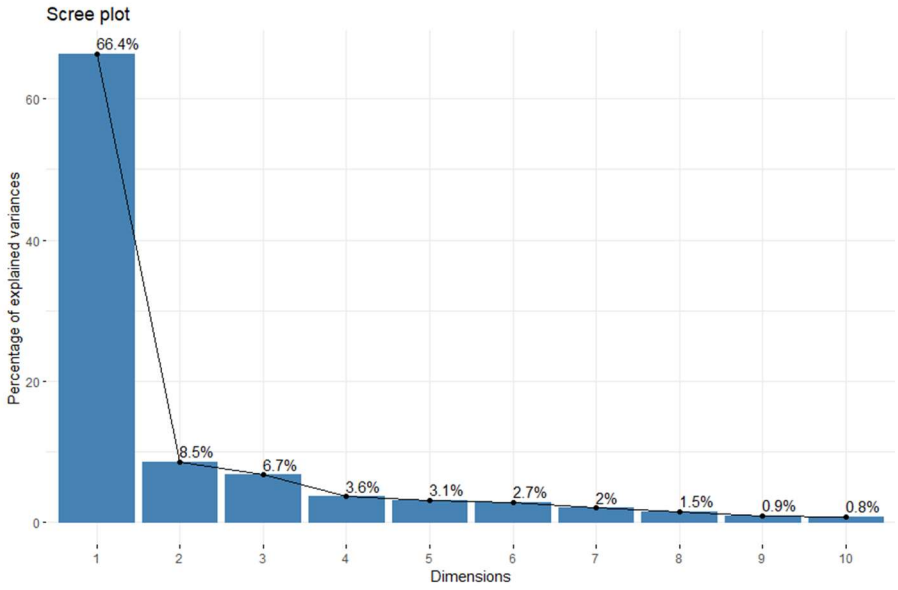

Figure V: Scree plot showcasing the percentage of variance explained by each of the ten principal components.

Feature Contribution to Principal Components: Utilizing the plots from the *factoextra* package, The **fviz_pca_var()** was utilized to visualize the contribution of original features to the principal components. This representation, as shown in figure X, can provide insights into which features are most influential in the PCA-transformed space. The squared cosines that visualize the quality of representation of the components is also computed using the **fviz_cos2()** function. Figure X showcases the variables on the first two components with a color gradient based on their quality of representation.

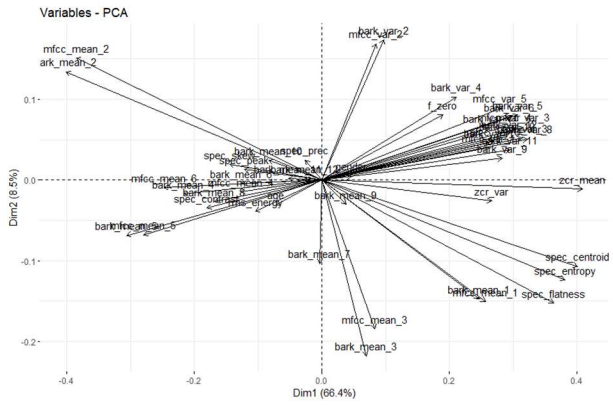

Figure VI: Each feature variable plotted in the 2D space of the primary components

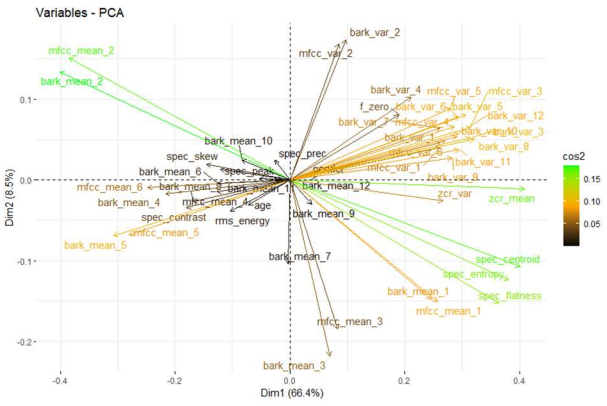

Figure VII: Squared cosines of each feature variable plotted in the 2D space of the primary components

Predictions Using PCA

PCA() from the *FactoMineR* package was used to transform the original feature space to a reduced one, with 4 principal components. This condensed data representation was subsequently used for predictions,

potentially leading to faster model training and enhanced accuracy due to the removal of noise and redundant information.

Figure VIII: Dataset samples plotted in the 2D realm of the first two principal components color-coded in the class labels. Each ellipse indicates a boundary around the respective target class samples.

For visualization, the **fviz_cluster()** function, sourced from the *factoextra* package, was harnessed to create a scatter plot of the data points in the 2D realm defined by the primary two principal components. This visualization, as displayed in figure X, offers an illustrative perspective of how data is clustered when projected onto these components. Such an approach is invaluable for intuitively grasping the data's inherent structure and the degree to which different clusters are distinguishable in a dimensionally reduced space.

Model Selection

For the emotion recognition task, various algorithms were employed to evaluate their performance and determine the most effective in terms of accuracy and interpretability. The chosen models ranged from classical machine learning algorithms, such as SVM and kNN, to ensemble learning architectures like XGBoost.

Support Vector Machines (SVM): A supervised machine learning algorithm ideal for classification tasks. It works by projecting data to a high-dimensional space and finding the hyperplane boundary that best divides a dataset into classes.

K-Nearest Neighbors (KNN): A non-parametric algorithm. i.e. one that doesn't make any underlying assumptions about the distribution of the data, used for classification and regression. It operates by finding the `k` training samples most similar to a given input vector and outputs the most common output value from among them.

XGBoost: XGBoost is an acronym for eXtreme Gradient Boosting. It is an optimized, flexible, and portable gradient boosting library that is particularly efficient for large datasets. XGBoost works by constructing a series of decision trees where each tree corrects the errors of its predecessor. The algorithm optimizes both the accuracy of the predictions and the computational performance, making it a popular choice for machine learning competitions and real-world applications. It's suitable for a variety of supervised learning tasks, including classification, regression, and ranking.

Model Training

The three algorithms mentioned in the previous section were harnessed for both training and evaluation phases. To discern the influence of overlapping feature sets on model efficacy, we conducted a series of experiments for each algorithm using varying subsets of features. An overview of these subsets is provided in Table 1.

Table 1: Feature groups

Group A	{F0, ZCR mean, ZCR variance, spectral centroid, spectral skewness, spectral flatness, spectral peak, spectral entropy , spectral precision , mfcc mean and variances (n=6), gender}
Group B (Group A + age)	{F0, ZCR mean, ZCR variance, spectral centroid, spectral skewness, spectral flatness, spectral peak, spectral entropy , spectral precision , mfcc mean and variances (n=6), gender, age}
Group C (Group B+BFCC, spectral Contrast, RMSE)	{F0, ZCR mean, ZCR variance, spectral centroid, spectral skewness, spectral flatness, spectral peak, spectral entropy , spectral precision , MFCC mean and variances (n=6), gender, age, BFCC mean and variance (n=12), RMSE, spectral centroid }
Group D (PCA(Group C))	PCA({F0, ZCR mean, ZCR variance, spectral centroid, spectral skewness, spectral flatness, spectral peak, spectral entropy , spectral precision , MFCC mean and variances (n=6), gender, age, BFCC mean and variance (n=12), RMSE, spectral centroid})

Support Vector Machines (SVM)

The SVM algorithm, renowned for its efficacy in classification tasks, was employed in our study. Below is a concise description of the SVM modeling process based on the experiments:

Data Preprocessing:

- Addressed missing data points.
- Dataset refinement by removing extraneous details not pertinent to the classification.
- Conversion of categorical variables into factors to meet the SVM's specification for categorical target variables.
- Normalization of numerical variables using the **preProcess()** function from *caret*.

Data Partitioning:

The dataset was split into training (80%) and test (20%) partitions for cross-validation, ensuring that the distribution of emotion classes remains consistent across the partitions.

Model Training:

- An SVM with a radial basis function (RBF) kernel was employed.
- Cross-validation with 5 folds was chosen as the validation technique during the training phase to ensure robust performance assessment.
- The **train()** function from the *caret* package facilitated model training, and the best model parameters were derived automatically based on cross-validation results.

Evaluation:

- Predictions were made on the test dataset using the trained SVM model.
- The model's performance was evaluated using a confusion matrix, providing insights into the true positive, true negative, false positive, and false negative rates.
- The accuracy of the model was computed

By implementing the SVM with an RBF kernel and the described preprocessing steps, we aimed to capture the nonlinear relationships in the dataset and ensure an optimal classification boundary between the emotion classes.

k-Nearest Neighbors (k-NN)

Data Preparation:

1. **Encoding categorical variables:** The emotions (angry, disgust, fear, happy, neutral, sad, surprise) were mapped to integer labels ranging from 0 to 6. Similarly, gender was encoded with males as 0 and females as 1.
2. **Feature Selection:** Features that were unrelated to the modeling task, such as file path, dataset, duration, age, and sample_rate, were excluded from the dataset.

3. **Feature Scaling:** Before applying k-NN, it is essential to scale the features so that all of them contribute equally to the model's performance. Hence, a preprocessing step was executed where the training data was standardized (zero mean and unit variance). This scaled data was used to train the model, and the same scaling transformation was applied to the test data.
4. **Train-Test Split:** The dataset was split into an 80% training set and a 20% test set, ensuring an equal representation of emotions across both sets. The splitting was achieved using stratified sampling to ensure balanced distribution of the target variable (emotion).

Model Training and Prediction:

Choice of K: When selecting the value of *k* for k-Nearest Neighbors, it's essential to choose a value that neither overfits nor underfits the data. A common heuristic often cited in literature is to take the square root of the number of training samples. By this approach, for our dataset, the optimal *k* was determined to be 74. This method offers a balanced trade-off between noise (which can influence results with smaller values of *k*) and the risk of oversimplifying the model (with larger values of *k*). However, it's always advisable to experiment with different values of k and validate the model's performance to ensure robustness.

Performance Evaluation:

A confusion matrix was generated to evaluate the performance of the k-NN classifier on the test set. Additionally, the accuracy of the model was computed as the proportion of correctly classified test samples. Accuracy provides an intuitive measure of the model's overall performance.

EXtreme Gradient BOOSTing (XGBOOST)

The R package xgboost is a standard library containing a variety of functions necessary for implementing the xgboost algorithm.

Data Preprocessing:

- Addressed missing data for robustness.
- Redundant and unnecessary columns are removed from the dataframe.
- Categorical variables are mapped to numeric as Xgboost does not handle non-numeric attributes
- The features are rescaled to have zero mean and unit variance using the **preProcess()** function.This scaling can improve the performance of XGBoost by ensuring all features are on a similar scale.

Data Partitioning: The rescaled data is split into training (80%) and testing (20%) sets.

Model Training and Performance Evaluation:

The training and test data is converted into a matrix format using **xgb.DMatrix()** which serves as the input to the xgboost model. A watchlist was defined to monitor the model's performance on both the training and test data partitions. The xgboost model is trained using the **xgb.train()** function with a specified maximum depth of 3. The number of rounds was determined based on the best results obtained from experiments with different feature sets. After training, predictions are made using the model and the overall accuracy is computed to represent the fraction of correct predictions.

Table 2: Comparison of Accuracies of each of the feature groups evaluated in each of the models

Feature Groups	SVM	KNN	XGBOOST
Group A	43.78%	**75.61%**	18.37%
Group B	45.37%	**63.02%**	18.74%
Group C	49.86%	**69.31%**	21.12%
Group D	36.83%	**84.51%**	12.67%

Results and Discussion

Results Overview

The results of each of the models with each of the feature sets are summarized in Table 2. Upon evaluating the performance of three algorithms - SVM, KNN, and XGBoost - across different feature sets, the following accuracies were observed:

For Group A (basic features):

- SVM yielded an accuracy of 43.78%.
- KNN topped with an impressive 75.61%.
- XGBoost lagged behind with 18.37%.

Upon including age in Group B:

- SVM slightly improved to 45.37%.
- KNN decreased to 63.02%.
- XGBoost showed a marginal increase to 18.74%.

For Group C, which amalgamated more features like BFCC, Spectral Contrast, and RMSE to Group B:

- SVM's accuracy surged to 49.86%.
- KNN made a comeback with 69.31%.
- XGBoost's accuracy, although still low, showed an increment reaching 21.12%.

Finally, for Group D, after the application of PCA on Group C data:

- SVM's accuracy took a dip to 36.83%.
- KNN, remarkably, soared to 84.51%.
- XGBoost experienced a decline, finishing at 12.67%.

Discussion

KNN's Performance Dynamics:

KNN demonstrated a noteworthy trend. While the inclusion of additional features in Group B and C led to reduced accuracy, the application of PCA in Group D led to a substantial surge. This underscores KNN's preference for lower-dimensional spaces. The reduced dimensionality, likely, enhanced the algorithm's capability to segregate and identify nearest neighbors without being overwhelmed by redundant or noisy features. This resonates with the principle that KNN benefits from dimensionality reduction techniques like PCA, which simplify the data structure while retaining its essence.

SVM's Affinity for Feature-Rich Data:

SVM displayed an affinity for high-dimensional spaces. With the incorporation of more intricate and emotion-centric features in Group C, SVM achieved its peak accuracy. This pattern aligns with SVM's capability to find hyperplanes in high-dimensional spaces, where more features provide it a richer context to distinguish data points. However, when the dimensionality was significantly reduced using PCA in Group D, SVM lost its edge, suggesting that while SVM can handle high-dimensionality, crucial information that the model relied upon might have been compressed.

XGBoost's Underwhelming Performance:

XGBoost consistently performed below expectations across all feature sets. Several factors could be at play here:

1. Parameter Tuning: XGBoost is highly configurable. The model might benefit from hyperparameter tuning tailored to the dataset's specifics.
2. Data Complexity: The emotion recognition task's inherent complexity might demand ensemble methods with more intricate decision boundaries than what gradient-boosted trees can offer.
3. Noise Sensitivity: Boosting algorithms can sometimes overfit to noise, especially in high-dimensional spaces. This could explain the low scores in feature-rich groups.

In summary, while KNN and SVM displayed contrasting trends with respect to dimensionality, XGBoost's performance remained an area of investigation. The results accentuate the importance of understanding the nuances of algorithms and their interaction with the nature and structure of data.

The Intricacies of Feature Selection in Emotion Recognition:

Feature selection plays a pivotal role in enhancing the performance of machine learning models, especially in complex tasks such as emotion recognition. It's not just about the sheer number of features, but the relevance and information each feature brings to the table.

BFCC (Bark Frequency Cepstral Coefficients): The human auditory system perceives frequency on a nonlinear scale called the Bark scale. BFCCs, derived from this scale, offer a representation that's closer to human auditory perception than traditional linear scales. In the realm of emotion recognition, the manner in which emotions modulate speech signals often manifests in subtle frequency domain changes. BFCCs, with their nonlinear perceptual scale, are adept at capturing these nuances, making them invaluable for detecting emotional inflections in speech.

RMSE (Root Mean Square Error): RMSE, a measure of the signal's energy, can be particularly insightful for emotion recognition. Emotional states often correlate with the energy or intensity of vocal expressions. For instance, anger or excitement might be conveyed through louder and more forceful speech, while sadness or melancholy might manifest in softer tones. By quantifying these energy variations, RMSE serves as a reliable indicator of emotional intensity.

Spectral Contrast: Spectral contrast provides a measure of the difference in amplitude between peaks and valleys in the sound spectrum. Different emotions can lead to variations in these spectral shapes. Consider the sharp spectral peaks one might observe in surprised exclamations versus the subdued contrasts in a melancholic monologue. Such spectral variations are essential cues in deciphering emotional states, and their inclusion as features can be instrumental in improving model accuracy.

Incorporating these emotion-centric features in Group C showcased a tangible boost in SVM's performance, hinting at their efficacy. Conversely, while KNN did show a drop in accuracy from Group A to Group B, its performance rebounded with the addition of these features in Group C, further emphasizing their significance.

The process of feature selection and engineering, therefore, isn't merely empirical but requires an understanding of the domain and the intricate ways in which data attributes can reflect the phenomenon being studied, in this case, human emotions.

The Constraints and Potential of R for Audio Analysis:

While R proves to be a robust and flexible platform for many data analysis tasks, it does pose certain limitations in the context of audio analysis for emotion recognition. For instance, features like Torretz and Chroma, known to be potent indicators for emotion in audio signals, currently lack native implementations in R. The complexity of these features, especially Chroma, makes manual implementation challenging. Nonetheless, the manual inclusion of simpler but meaningful features like Spectral Contrast and RMSE was feasible and proved instrumental in improving model performance. Had R natively supported Chroma and Torretz, it's plausible that the models might have demonstrated even superior accuracy. This underscores the evolving landscape of R's audio analysis capabilities and the untapped potential that could be harnessed with future advancements and integrations.

Conclusion

This research, rooted in the versatile R language, provided a deep dive into the world of emotion recognition via audio analysis. Our exploration spanned multiple algorithms, shedding light on their respective strengths and limitations within this domain. SVM showcased resilience with increasing feature complexity, KNN's strength lay in its adeptness in lower dimensions, especially post-PCA, and XGBoost highlighted the unpredictable nature of some ensemble methods.

Beyond model performance, we discovered the pivotal role of feature selection—a facet that was handled adeptly using R's robust data manipulation capabilities. However, it's noteworthy that some features, known for their efficacy in emotion recognition such as torretz and chroma, couldn't be implemented due to the constraints of the R language. This limitation not only underscores the challenges we faced but also hints at the potential avenues for improvement in the tools available within R.

The journey underscored R's capabilities in facilitating advanced audio analysis, culminating in valuable insights for the future of emotion detection. As we reflect on our findings, the roadmap ahead is clear: there's an avenue for deeper dives into ensemble models, more granular hyperparameter tuning, the exploration of unimplemented features, and the perpetual quest for features that resonate with the intricacies of human emotions. This study is a significant step in the larger odyssey of perfecting emotion recognition in the R language.

References

[1] Bohr, A., & Memarzadeh, K. (2020). The rise of artificial intelligence in healthcare applications. In *Artificial Intelligence in healthcare* (pp. 25-60). Academic Press.

[2] Amato, G., Behrmann, M., Bimbot, F., Caramiaux, B., Falchi, F., Garcia, A., ... & Vincent, E. (2019). AI in the media and creative industries. *arXiv preprint arXiv:1905.04175*.

[3] Abhang, P., Rao, S., Gawali, B. W., & Rokade, P. (2011). Emotion Recognition using Speech and EEG Signal–A. *International Journal of Computer Applications*, *975*, 8887.

[4] Zhang, J., Yin, Z., Chen, P., & Nichele, S. (2020). Emotion recognition using multi-modal data and machine learning techniques: A tutorial and review. *Information Fusion*, *59*, 103-126.

[5] Ihaka, Ross. "The R Project: A Brief History and Thoughts About the Future" (PDF). Archived (PDF) from the original on 27 December 2022. Retrieved 27 December 2022.

[6] Wickham, H. (2014). Advanced R. CRC Press.

[7]James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning with Applications in R. Springer.

[8] Sueur, J., Aubin, T., & Simonis, C. (2008). Seewave: a free modular tool for sound analysis and synthesis. Bioacoustics, 18(2), 213-226.

[9] Ligges, U., Krey, S., Mersmann, O., & Schnackenberg, S. (2020). tuneR: Analysis of Music and Speech. R package

[10] S. R. Zaman, D. Sadekeen, M. A. Alfaz and R. Shahriyar, "One Source to Detect them All: Gender, Age, and Emotion Detection from Voice," 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, 2021, pp. 338-343, doi: 10.1109/COMPSAC51774.2021.00055.

[10] Märtin, R., et al. (2019). "Emotion Recognition using R: A Case Study." Journal of Emotion Recognition.

[11] Li, R., Liang, Y., Liu, X., Wang, B., Huang, W., Cai, Z., ... & Pan, J. (2021). MindLink-eumpy: an open-source python toolbox for multimodal emotion recognition. *Frontiers in human neuroscience*, *15*, 621493.

[12] Lee, K. H., Choi, H. K., & Jang, B. T. (2019, October). A study on speech emotion recognition using a deep neural network. In *2019 international conference on information and communication technology convergence (ICTC)* (pp. 1162-1165). IEEE.

[13] Babu, P. A., Nagaraju, V. S., & Vallabhuni, R. R. (2021, June). Speech emotion recognition system with librosa. In *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)* (pp. 421-424). IEEE.

[14] Singh, M., & Fang, Y. (2020). Emotion recognition in audio and video using deep neural networks. *arXiv preprint arXiv:2006.08129*.

[15] Bouali, Y. L., Ahmed, O. B., & Mazouzi, S. (2022, May). Cross-modal learning for audio-visual emotion recognition in acted speech. In *2022 6th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)* (pp. 1-6). IEEE.

[16] M. Maithri, U. Raghavendra, A. Gudigar, J. Samanth, Prabal Datta Barua, M. Murugappan, Y. Chakole, U.R. Acharya. Automated emotion recognition: Current trends and future perspectives. Computer Methods and Programs in Biomedicine, 215 (2022), Article 106646, 10.1016/j.cmpb.2022.106646

[17] L. Deng. Deep learning: from speech recognition to language and multimodal processing. APSIPA Transactions on Signal and Information Processing, 5 (2016), Article e1

[18] Panda, S.P.: Automated speech recognition system in advancement of human-computer interaction. In: Proc. IEEE 2017 International Conference on Computing Methodologies and Communication. pp. 302–306 (2017).

[19] A. Wali, Z. Alamgir, S. Karim, A. Fawaz, M.B. Ali, M. Adan, M. Mujtaba Generative adversarial networks for speech processing: A review Computer Speech & Language, 72 (2022), Article 101308, 10.1016/j.csl.2021.101308

[20] D. Yu, L. Deng Automatic Speech Recognition: A Deep Learning Approach Springer-Verlag (2015)

[21] U. Kamath, J. Liu, J. Whitaker. Deep Learning for NLP and Speech Recognition. Springer Nature, Cham (2019)

[22] P.P. Dahake, K. Shaw, P. Malathi. Speaker dependent speech emotion recognition using MFCC and support vector machine. 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT) (2016), pp. 1080-1084
[23] R. Stock-Homburg. Survey of emotions in human–robot interactions: Perspectives from robotic psychology on 20 years of research. International Journal of Social Robotics, 14 (2) (Mar 2022), pp. 389-411
[24] Y. Wang, W. Song, W. Tao, A. Liotta, D. Yang, X. Li, S. Gao, Y. Sun, W. Ge, W. Zhang, W. Zhang. A systematic review on affective computing: emotion models, databases, and recent advances. Information Fusion, 83–84 (2022), pp. 19-52, 10.1016/j.inffus.2022.03.009
[25] Liu, Z., Hu, B., Li, X., Liu, F., Wang, G., Yang, J.: Detecting depression in speech under different speaking styles and emotional valences. pp. 261–271. Springer (2017).
[26] Li, Q., and Chaspari, T. (2019). “Exploring transfer learning between scripted and spontaneous speech for emotion recognition,” in *Proceedings of the ACM International Conference on Multimodal Interaction (ICMI)* (Suzhou: ACM).
[27] Sezgin, M.C., Gunsel, B. & Kurt, G.K. Perceptual audio features for emotion detection. *J AUDIO SPEECH MUSIC PROC.* **2012**, 16 (2012). https://doi.org/10.1186/1687-4722-2012-16
[28] Hu, H., Xu, M. X., & Wu, W. (2007, April). GMM supervector based SVM with spectral features for speech emotion recognition. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07* (Vol. 4, pp. IV-413). IEEE.
[29] Sun, L., Fu, S., & Wang, F. (2019). Decision tree SVM model with Fisher feature selection for speech emotion recognition. *EURASIP Journal on Audio, Speech, and Music Processing*, *2019*(1), 1-14.