Princess Sumaya جامعـــة
University الأميـــرة سميّـــة
for Technology للتكنولوجيا

Speech Emotion Recognition For Smart-Home Application

Authors

Ibrahim Sanna 20200491/ Ibrahim Ghula 20200679

Instructor

Dr. Rawan Ghnemat

King Hussein School of Computing Sciences, Artificial Intelligence

May 2023

1 Introduction

In recent years, the emergence of smart home technology has revolutionized the way we interact with our living spaces. With the integration of various sensors and intelligent systems, smart homes offer a wide range of features and functionalities to enhance comfort, convenience, and energy efficiency. However, despite their remarkable capabilities, smart home systems often lack the ability to perceive and respond to human emotions, which are integral to effective communication and personalized experiences.

Recognizing the significance of emotions in human-machine interaction, researchers have turned their attention to incorporating speech emotion recognition (SER) techniques into smart home applications. SER, a subfield of affective computing, aims to detect and interpret emotional cues from speech signals, allowing systems to discern and respond to the emotional states of users. By leveraging this technology, smart homes can create a more empathetic and intuitive environment, providing a tailored experience that meets the emotional needs of the residents.

In this project, we aim to utilize the advancements in speech emotion recognition to facilitate more effective human-smart home interactions, creating a more emotionally intelligent and responsive environment.

2 Background and Motivation

You can never stop talking about the significance of SER in smart home applications, but here are the points we found to be most important:

1. Mental Health Monitoring: Mental health is a significant and complex issue that affects individuals, families, and societies worldwide. It encompasses a broad range of conditions and disorders that impact a person's emotional, psychological, and social well-being. Mental health problems can manifest in various forms, including anxiety disorders, depression, bipolar disorder, schizophrenia, and others.
   One of the primary challenges related to mental health is the prevalence and scope of the problem. According to a study in 2017, it is estimated that 792 million people lived with a mental health disorder. This is slightly more than one in ten people globally (10.7%)[1].
   This is just the tip of the iceberg, imagine the millions of people who suffer from mental health issues and do not seek medical help, that could be for different reasons one of them is mental health being a taboo in the society.
   We think that incorporating speech emotion recognition in the different smart home technologies, like the personal assistant, could help in identifying these issues, and provide help for the individuals, and in serious cases, recommend visiting mental health centers.
2. Enhanced User Experience: Speech emotion recognition can significantly improve the user experience in smart home applications by enabling systems to understand and respond to the emotional states of the residents. Emotion-aware smart home systems can adapt their behavior and provide tailored responses based on the detected emotions, creating a more personalized and empathetic interaction. This can lead to increased user satisfaction and engagement with the smart home technology.
   One example can be music recommendation base on the mood of the user[2],the AI recognizes the mood of the user based on their voice and recommends appropriate music for that mood.
3. Family Dynamics and Communication: SER can facilitate better understanding

and communication within families residing in smart homes. By recognizing emotional states in speech, the system can gauge the overall emotional atmosphere and detect potential conflicts or stressors. This insight can prompt the system to suggest conflict resolution strategies, family bonding activities, or initiate communication channels to address concerns. SER can foster improved family dynamics and promote open and supportive communication, especially when lack of communication is one of the important reasons for conflict between children and parents[3].
The system could offer lessons or tips on effective communication, anger management, or other relevant skills when it detects potential conflict situations. This could help family members improve their conflict resolution skills over time.

3 Previous Related Work

In order to build upon existing knowledge and understand the progress made in the field of speech emotion recognition (SER) for smart home applications, it is essential to examine the previous related work. This section provides an overview of the research and advancements that have been made in SER within the context of smart homes. those are some of the researches and approaches we found:

1. In a study in 2019, deep learning techniques were utilized to improve speech emotion recognition (SER) accuracy. Two types of Convolutional Neural Network and Long Short-Term Memory (CNN LSTM) networks were constructed: a 1D CNN LSTM network and a 2D CNN LSTM network. The networks aimed to learn local and global emotion-related features from speech and log-mel spectrogram data, respectively. By combining the strengths of CNN and LSTM networks, the designed networks achieved excellent performance on benchmark databases. The 2D CNN LSTM network outperformed traditional approaches, achieving recognition accuracies of 95.33% and 95.89% on the Berlin EmoDB database and 89.16% and 52.14% on the IEMOCAP database for speaker-dependent and speaker-independent experiments, respectively. These results demonstrate the effectiveness of deep learning-based SER models in enhancing speech emotion recognition accuracy[4].
2. Speech Emotion Recognition (SER), which is based on artificial intelligence (AI), has become a common feature of smart home personal assistants. In order to deploy an effective method in Consumer Electronics home devices, this work,that was done in 2021, analyzes emotional states in speech. For the purpose of learning and categorizing the emotions connected to human speech, a 1-D convolutional neural network (CNN) is used. On common datasets (RAVDESS and TESS), the method achieves high classification accuracies of 90.48%, 95.79%, and 94.47%. The findings show that 1-D CNN models are effective at automatically predicting emotions, which qualifies them for inclusion in smart home assistants[5].
3. this work, that was done in 2020, proposes the approach of the two-layer fuzzy multiple random forest (TLFMRF). Traditional methods face challenges in personalized feature extraction and accounting for differences among individuals. To address these issues, the TLFMRF fuses personalized and non-personalized features and leverages fuzzy C-means clustering to divide high dimensional emotional features into subclasses. Multiple random forests are employed to recognize different emotional states,

and a TLFMRF model is established. The proposal also conducts separate classification for challenging emotions. Experimental results on CASIA corpus and Berlin EmoDB demonstrate the effectiveness of the approach, with recognition accuracies outperforming backpropagation neural network and random forest by 1.39%-7.64% and 4.06%-4.30% respectively. Application experiments with an emotional social robot system further indicate real-time tracking capabilities for six basic emotions (angry, fear, happy, neutral, sad, and surprise) by a mobile robot[6].

Our approach uses 1-D CNN model to make Speech Emotion Recognition AI that will be used in various smart home applications. While it is better to use CNN in combination with RNN like one of the methods we mentioned earlier[4], it is very time consuming and resource demanding. When it comes to the fuzzy approach, it is hard to implement, especially with the amount of parameters we have when dealing with audio. Our approach is very similar to the second approach we previously mentioned[5], but we will focus on processing the audio and extracting features more efficiently.

4 Implementation Description

Our approach uses 1-D CNN model to make Speech Emotion Recognition AI. We used the (CREMA) dataset to train our model. Appropriate feature extraction is very crucial to produce good accuracy and low loss, so we made sure to extract the (MFCC,ZCR and RMS) features for all audio files in the dataset using (librosa) library methods. Features were not only extracted for the original audio, we added noise to it, as well as pitched it before the extraction, this is necessary so the model doesn't memorize the data and cause over-fitting (you can read more on this here[7]). We used a simple CNN structure consisting of three 1-D conv layers with max pooling after each one, and a dense layer with dropout to reduce overfitting, and at the end a dense layer for the output.

The expected results before implementation were 70% or higher for val-accuracy, but the test results were unfortunately lower than expected at a 65.16% (you can look at the appendix for more details).

5 Discussion

In the end, We would say that the objectives of this project were successfully met, we highlighted the importance of using SER in smart home applications, we trained an AI model that recognizes speech emotion, and we gained a lot of experience in this field.

Using CNN as an approach in this project was a good choice, it is not very hard to implement or train, and it produced good results. Even if there are approaches that produced better results, we would say it is a problem in the execution on our side and not the approach itself.

In this project we used (Kaggle) as a code editor and dataset provider, it is needless to say that it was a good choice and we will definitely use it again.

If we want to redo this project, we would choose a larger dataset, and process the data more than just adding noise and pitching it, as well as try other different CNN structures.

At the end of this project, we gained a lot of experience when it comes to how to use machine learning with audio, and the importance of emotion recognition, whether it is facial or audible, in smart home applications. We also improved our skills in documentation and citation.

References

[1] S. Dattani, H. Ritchie, and M. Roser, "Mental health," *Our World in Data*, 2021. https://ourworldindata.org/mental-health.

[2] A. Kulkarni, S. Prajwal, H. Jayanthi, and K. S. Sowmya, "Music recommendation system using speech-based emotion recognition," *Gunjan, V.K., Zurada, J.M. (eds) Modern Approaches in Machine Learning Cognitive Science: A Walkthrough. Studies in Computational Intelligence*, vol. 1027, 2022.

[3] A. Ambhore, A. Ashtaputre, B. Aurangabad, P. Puri, and S. Bhutekar, "Communication problem and conflicts in parent child relationship," 03 2022.

[4] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1d 2d cnn lstm networks," *Biomedical Signal Processing and Control*, vol. 47, pp. 312–323, 2019.

[5] R. Chatterjee, S. Mazumdar, R. Sherratt, R. Halder, T. Maitra, and D. Giri, "Real time speech emotion analysis for smart home assistants," *IEEE Transactions on Consumer Electronics*, vol. PP, pp. 1–1, 02 2021.

[6] L. Chen, W. Su, Y. Feng, M. Wu, J. She, and K. Hirota, "Two-layer fuzzy multiple random forest for speech emotion recognition in human-robot interaction," *Information Sciences*, vol. 509, pp. 150–163, 2020.

[7] V.-V. Eklund, "Data augmentation techniques for robust audio analysis," 2019.

Appendix

.

Code Link:

https://www.kaggle.com/code/ibrahimsanna/speech-emotion-recognition-in-cnn

Code Snippets:

creating the dataframe.

In [61]:

```
files_Path='../input/speech-emotion-recognition-en/Crema/'

c_emotions = {
    'NEU':'neutral',
    'HAP':'happy',
    'SAD':'sad',
    'ANG':'angry',
    'FEA':'fear',
    'DIS':'disgust'}

c_file = []
for wav in os.listdir(files_Path):
    emo = wav.partition(".wav")[0].split('_')
    emotion = c_emotions[emo[2]]
    c_file.append((files_Path+'/'+wav,emotion))

data_df = pd.DataFrame(c_file, columns = ['File_path', 'Emotion'])
data_df.to_csv('data_df.csv')
data_df.shape
data_df.head(10)
```

Out[61]:

	File_path	Emotion
0	../input/speech-emotion-recognition-en/Crema//...	disgust
1	../input/speech-emotion-recognition-en/Crema//...	happy
2	../input/speech-emotion-recognition-en/Crema//...	happy
3	../input/speech-emotion-recognition-en/Crema//...	disgust
4	../input/speech-emotion-recognition-en/Crema//...	disgust
5	../input/speech-emotion-recognition-en/Crema//...	disgust
6	../input/speech-emotion-recognition-en/Crema//...	happy
7	../input/speech-emotion-recognition-en/Crema//...	sad
8	../input/speech-emotion-recognition-en/Crema//...	disgust
9	../input/speech-emotion-recognition-en/Crema//...	sad

extracting features.

In [64]:

```
def add_noise(data,random=False,rate=0.035,threshold=0.075):
    if random:
        rate=np.random.random()*threshold
    noise=rate*np.random.uniform()*np.amax(data)
    augmented_data=data+noise*np.random.normal(size=data.shape[0])
    return augmented_data

def pitching(data,sr,pitch_factor=0.7,random=False):
    if random:
        pitch_factor=np.random.random() * pitch_factor
    return librosa.effects.pitch_shift(y=data,sr=sr,n_steps=pitch_factor)
#-----------------------------------------------------------------------------------------
def zcr(data,frame_length,hop_length):
    zcr=librosa.feature.zero_crossing_rate(y=data,frame_length=frame_length,hop_length=hop_length)
    return np.squeeze(zcr)
def rmse(data,frame_length=2048,hop_length=512):
    rmse=librosa.feature.rms(y=data,frame_length=frame_length,hop_length=hop_length)
    return np.squeeze(rmse)
def mfcc(data,sr,frame_length=2048,hop_length=512,flatten:bool=True):
    mfcc=librosa.feature.mfcc(y=data,sr=sr)
    return np.squeeze(mfcc.T)if not flatten else np.ravel(mfcc.T)
#-----------------------------------------------------------------------------------------
def extract_features(data,sr,frame_length=2048,hop_length=512):
    result=np.array([])

    result=np.hstack((result,
                      zcr(data,frame_length,hop_length),
                      rmse(data,frame_length,hop_length),
                      mfcc(data,sr,frame_length,hop_length)
                     ))
    return result
```

```
def get_features(path,duration=2.5, offset=0.6):
    data,sr=librosa.load(path,duration=duration,offset=offset)
    aud=extract_features(data,sr)
    audio=np.array(aud)

    noised_audio=add_noise(data,random=True)
    aud2=extract_features(noised_audio,sr)
    audio=np.vstack((audio,aud2))

    pitched_audio=pitching(data,sr,random=True)
    aud3=extract_features(pitched_audio,sr)
    audio=np.vstack((audio,aud3))

    return audio
#-----------------------------------------------------------------------------------------
X, Y = [], []

for path, emotion, index in zip(data_df.File_path, data_df.Emotion, range(data_df.File_path.shape
[0])):
    features = get_features(path)
    if index % 500 == 0:
        print(f'{index} audio has been processed')
    for i in features:
        X.append(i)
        Y.append(emotion)

print('Done')
```

the model.

In [75]:

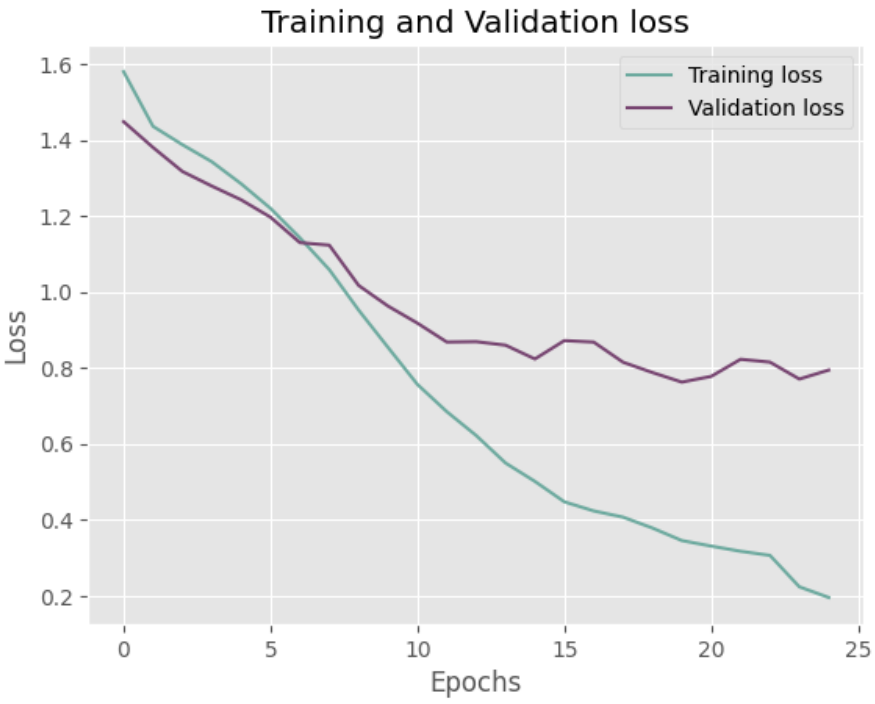
```
import tensorflow.keras.layers as L
import tensorflow as tf
from keras.callbacks import ReduceLROnPlateau
early_stop = callbacks.EarlyStopping(monitor='val_loss', mode='auto', patience=5, restore_best_weigh
ts=True)
lr_reduction = ReduceLROnPlateau(monitor='val_loss', patience=3, verbose=1, factor=0.1, min_lr=0.000
01)
EPOCH=50
BATCH_SIZE=30

model=tf.keras.Sequential([
    L.Conv1D(32,kernel_size=6, strides=1,padding='same', activation='relu',input_shape=(X_train.shap
e[1],1)),
    L.MaxPool1D(pool_size=5,strides=2,padding='same'),
    L.Conv1D(64,kernel_size=6,strides=1,padding='same',activation='relu'),
    L.MaxPool1D(pool_size=5,strides=2,padding='same'),
    L.Conv1D(128,kernel_size=6,strides=1,padding='same',activation='relu'),
    L.MaxPool1D(pool_size=5,strides=2,padding='same'),
    L.Flatten(),
    L.Dense(256,activation='relu'),
    L.Dropout(0.5),
    L.Dense(6,activation='softmax')
])
model.compile(optimizer='nadam',loss='categorical_crossentropy',metrics='accuracy')
history=model.fit(X_train, y_train, epochs=EPOCH, validation_data=(X_val,y_val), batch_size=BATCH_SI
ZE,callbacks=[early_stop,lr_reduction])
```

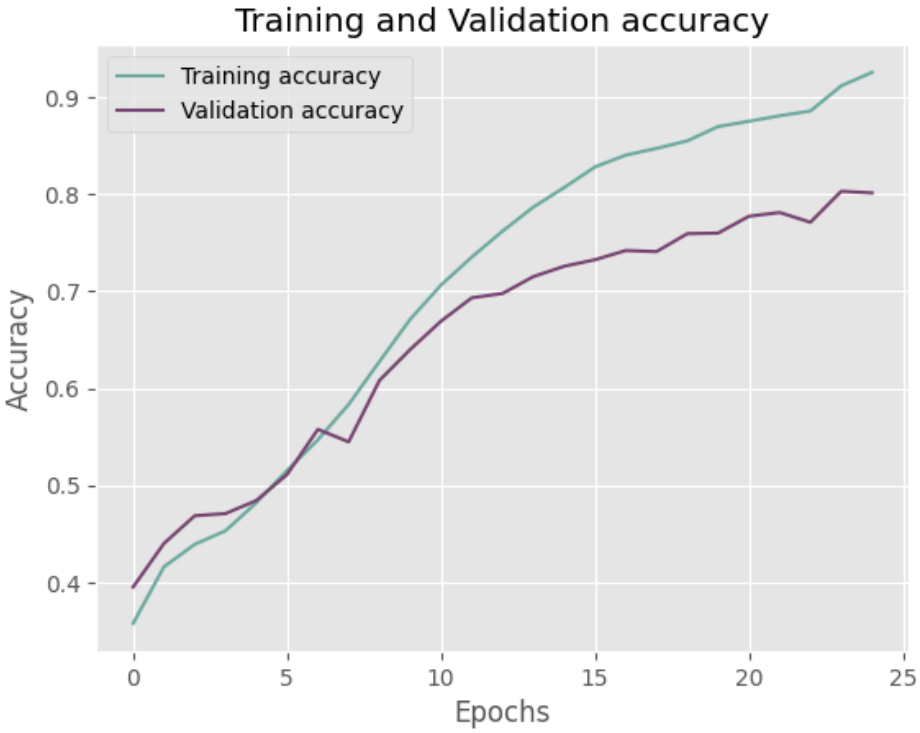
Test results:

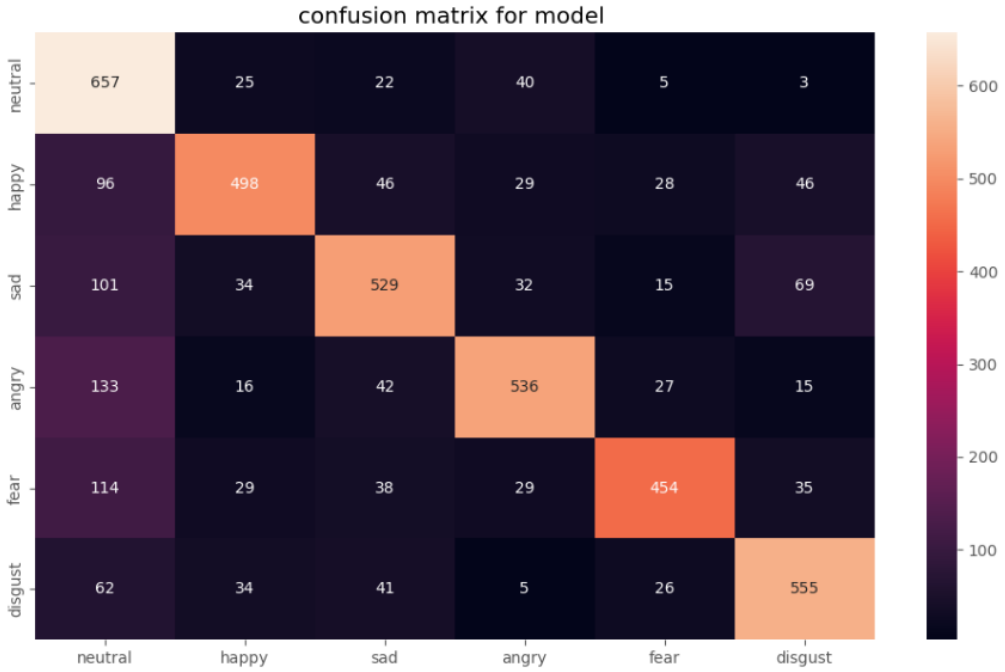
In [76]:

```
val_accuracy = np.mean(history.history['val_accuracy'])
print("\n%s: %.2f%%" % ('val_accuracy', val_accuracy*100))
```

```
val_accuracy: 65.16%
```




Training and Validation accuracy
Training accuracy
Validation accuracy
Accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0
5
10
15
20
25
Epochs


confusion matrix for model
neutral
happy
sad
angry
fear
disgust
657 25 22 40 5 3
96 498 46 29 28 46
101 34 529 32 15 69
133 16 42 536 27 15
114 29 38 29 454 35
62 34 41 5 26 555
600
500
400
300
200
100

Project plan:

-timing: the project was planned on the period from the 15th of April to 24th of May.

-Task division:

Task	Participation	Time
Choosing the dataset	Both team members	15/4 to 25/4
Material research	Both team members	25/4 to 2/5
Choosing references	Ibrahim Ghula	2/5 to 4/5
Code research	Ibrahim Sanna	4/5 to 17/5
Coding	Ibrahim Sanna 80%, Ibrahim Ghula 20%	6/5 to 17/5
Documentation	Both team members	25/4 to 24/5