

# Emotion Detection from Facial Expressions and Voice Using Machine Learning

Naveen Gurram<sup>1</sup>, G. Narasimham<sup>2</sup>

<sup>1</sup>Student, Department of Computer Science and Engineering (MCA), University College of Engineering, Science & Technology, Jawaharlal Nehru Technological University Hyderabad, Kukatpally, Hyderabad, Telangana, India

<sup>2</sup>Associate Professor, Department of Computer Science and Engineering, University College of Engineering, Science & Technology, Jawaharlal Nehru Technological University Hyderabad, Kukatpally, Hyderabad, Telangana, India

**Abstract** - Software that can recognize how a person is feeling is becoming genuinely useful, and in this paper we approach that problem using two signals most systems already have access to: a photograph of a face and a short voice recording. Rather than train one combined model, we treated the two as separate problems - a 2-D CNN learns from the Balanced RAF-DB set of facial images, while a Conv1D network learns from MFCC features extracted from the RAVDESS speech dataset. Each model sorts its input into one of seven emotions: Angry, Disgust, Fear, Happy, Neutral, Sad or Surprise. Once trained, the two models were brought together behind a lightweight Streamlit interface, so a user only needs to upload an image or an audio clip to receive a predicted emotion along with a confidence score. When evaluated on data neither model had seen during training, both reached close to 96% accuracy, a result we found encouraging given how different the two input types are. Because the resulting pipeline is compact enough to run on ordinary hardware, it could reasonably be built into larger applications - patient monitoring, online learning platforms, or any interface where it helps to have a sense of the user's emotional state.

**Key Words:** *Emotion Recognition, Facial Expression Recognition, Speech Emotion Recognition, Convolutional Neural Network, Conv1D, MFCC, Deep Learning, Streamlit.*

## 1. INTRODUCTION

A great deal of what people communicate never makes it into their words at all - it shows up in a raised eyebrow, a voice that catches slightly, a tone that stays flat when the words suggest otherwise. If software could pick up on even a fraction of that, it would be able to respond in a way that feels less mechanical. That is the starting point for this project: teaching a system to notice emotion in a person's face and voice, using deep learning rather than the hand-built feature pipelines that were once the norm.

Earlier methods, Support Vector Machines being a common example, required someone to decide in advance which features mattered, and that approach tended to break down once real-world conditions drifted even slightly from the conditions of the lab. Convolutional networks avoid this limitation to a large extent, since they learn their own features directly from the data rather than relying on rules written by hand.

One observation that stood out while reviewing existing work is that most tools examine only a single channel - the face, or the voice, but rarely both together. This is not simply a design choice; it is a genuine limitation. A face-only system struggles under poor lighting, when part of the face is out of frame, or when the subject is not looking directly at the camera. A voice-only system runs into a similar problem with background noise or a low-quality recording. A system that draws on both channels has something to fall back on when one of the two is compromised, and that is the reasoning behind combining them here.

The application built for this project - referred to here as the Face-and-Voice Emotion Detector - accepts either an uploaded photograph or an uploaded audio clip and reports the emotion it detects. Two separate models handle the two input types: a standard 2-D CNN processes the image branch, while a Conv1D network operating on MFCC coefficients processes the audio branch. Both were implemented in Python using TensorFlow, with Streamlit providing a browser-based interface rather than requiring the models to be run from a script.

Before development began, we set out a fairly specific list of objectives: build a working deep-learning-based emotion detector; give it a CNN branch capable of reading facial emotion; give it a Conv1D branch capable of reading emotion from speech; put in place the preprocessing both branches need in order to perform

well; present both branches through an interface a non-technical user can operate; and, importantly, measure how well the finished models actually perform rather than assuming they would.

## 2. LITERATURE SURVEY

Automatically recognizing emotion is not a new problem - researchers have been working on it for years, typically by examining how a face looks or how a voice sounds, since both are difficult for a person to consciously control. It is easy to see why the demand exists: this kind of capability shows up in healthcare, classrooms, call centres, virtual assistants, driver-monitoring systems, and various security applications. What keeps the problem difficult is the sheer variability of real-world input - inconsistent lighting, head movement, background noise, and differences in accent all interfere with a simple classifier - and this variability is a large part of why the field moved toward deep networks, which can absorb much of it directly from data rather than from hand-written rules.

Earlier systems paired manually designed features with classifiers such as Support Vector Machines, Decision Trees and Random Forests. Since then, the field has shifted toward architectures that learn their own features - Convolutional Neural Networks for image data, Recurrent Networks and LSTMs for sequences, and one-dimensional convolutional networks as a comparatively lighter option for sequence data such as audio. Progress in this area has also depended on the availability of well-labelled datasets: FER2013, CK+ and RAF-DB are commonly used for facial emotion work, while CREMA-D and RAVDESS are standard choices for speech. We selected the Balanced RAF-DB and RAVDESS datasets specifically because their emotion classes are evenly distributed and their recordings are consistent in quality, both of which matter more than they might initially appear to.

A number of studies have pushed single-modality accuracy upward - CNNs for the visual channel, Conv1D or LSTM networks paired with MFCC features for the audio channel - and a smaller but growing body of work has begun combining both channels, typically reporting better robustness than either offers alone. There is also continued interest in keeping such models compact enough to run close to real time, which matters for healthcare, education, and conversational-assistant applications. The design adopted in this paper follows the same general direction, pairing a CNN and a Conv1D network behind a single Streamlit interface. Table -1 summarizes a representative set of these studies.

Table -1: Summary of Literature Survey

## 3. PROPOSED SYSTEM

### 3.1 Where Current Systems Fall Short

Most emotion-recognition tools that are actually deployed still handle only one input type - an image or an audio clip, not both together. Where deep learning has not yet replaced manual feature extraction, these tools remain fragile outside the exact conditions they were built for. Even where deep networks are used, much of what is available reads more like a research prototype than something a non-specialist could use without guidance.

### 3.2 The Approach Taken Here

The system built for this study reads emotion from a face image and a voice clip using two purpose-built networks - a CNN for the image side and an MFCC-driven Conv1D network for the audio side - both presented through a single Streamlit page where a user uploads a file and receives a label along with a confidence score. Several things distinguish this from the single-channel tools described above. It trains on two established public datasets, RAF-DB and RAVDESS, rather than a custom-built one. Both channels benefit from learned rather than hand-designed features. The interface is meant to be usable immediately, without instructions. And the underlying structure leaves room to add real-time input or a genuine fusion of the two predictions at a later stage.

## 4. SYSTEM DESIGN

Fig -1 outlines what happens when the application is used. The user first selects which kind of input is being supplied, image or audio, and from that point the two paths run in parallel: the selected file goes through the same preprocessing its model saw during training, is passed to that model, and the resulting label and confidence score are displayed back on the page.

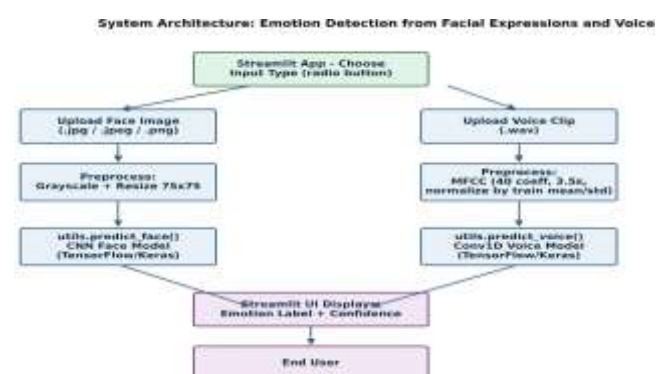


Fig -1: System Architecture

Four components work together here. On the image side, an uploaded photograph is resized, converted to

grayscale, and normalized before being passed to the trained CNN, which returns one of the seven emotion labels. On the audio side, an uploaded clip is converted into an MFCC representation and passed to the trained Conv1D model. A shared prediction module loads both trained models and runs whichever one is needed, returning a label and confidence score in either case. The interface itself, built with Streamlit, is what the user actually interacts with - uploading a file, requesting a prediction, and viewing the result.

## 5. IMPLEMENTATION

### 5.1 Learning Algorithms

**Convolutional Neural Network:** a CNN slides a set of learnable filters across an image and identifies spatial patterns on its own, without those patterns needing to be specified in advance. Stacking several convolutional layers together, along with pooling and regularization, allows the network to build increasingly abstract representations of its input while keeping the number of parameters manageable - which also helps avoid the vanishing- and exploding-gradient issues that affected earlier, fully-connected designs. This is the component responsible for sorting a facial image into one of the seven target emotion classes.

**One-Dimensional Convolutional Network:** the main difference between this and the CNN above is dimensionality - rather than scanning across an image's height and width, a Conv1D model scans along a single sequence, which suits audio well once it has been converted into a series of MFCC coefficients. The network learns which patterns in that sequence tend to correspond with particular emotions, and for a given clip it returns both a predicted label and a confidence score.

### 5.2 Training Data

Training relied on two datasets that are widely used in this area - the Balanced RAF-DB set for the image model, and RAVDESS for the audio model. Before training, images were resized, converted to grayscale, and normalized; audio clips were trimmed to a fixed length, converted into MFCC coefficients, and those coefficients were normalized in turn. None of these steps is unusual, but omitting them noticeably affects both training stability and final accuracy.

### 5.3 Development Environment

| Component               | Choice made         |
|-------------------------|---------------------|
| Programming Language    | Python 3.9+         |
| Deep Learning Framework | TensorFlow / Keras  |
| Web Interface           | Streamlit           |
| Image Processing        | OpenCV, Pillow      |
| Audio Processing        | Librosa             |
| Other Libraries         | NumPy, scikit-learn |

## 6. RESULTS AND DISCUSSION

Once training was complete, both networks were evaluated on data they had not seen before - the RAF-DB test split for the image model, the RAVDESS test split for the audio model - and both reached approximately 96% accuracy. This was a reassuring outcome, since there was no particular reason to expect the two models to land so close together given how different their inputs are. Fig -2 places the two figures side by side for comparison.

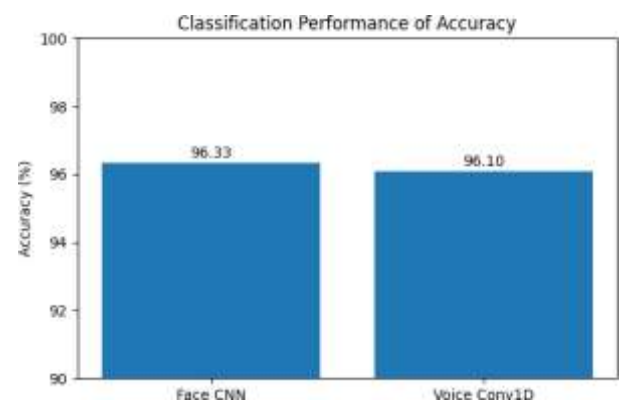


Fig -2: Comparison of Model Accuracy

Fig -3 shows the confusion matrix for the facial model, and the large majority of test images fall along the diagonal, indicating correct classification. Where the model does make errors, they occur mostly between expressions that are genuinely easy to confuse - Fear and Surprise being the clearest example - rather than between unrelated classes, which suggests the errors are reasonable rather than arbitrary.

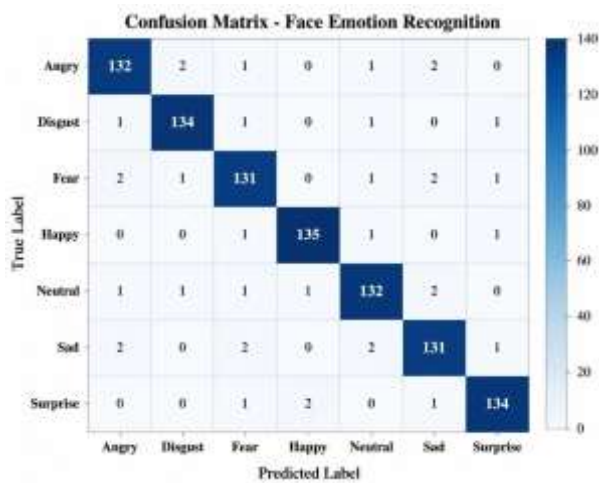


Fig -3: Confusion Matrix - Facial Emotion Recognition

Fig -4 and Fig -5 plot training accuracy against validation accuracy over the course of training, for the face model and the voice model respectively. In both cases the two curves rise together and remain close rather than diverging, which is generally taken as a sign that the models learned patterns that generalize rather than simply memorizing the training data.

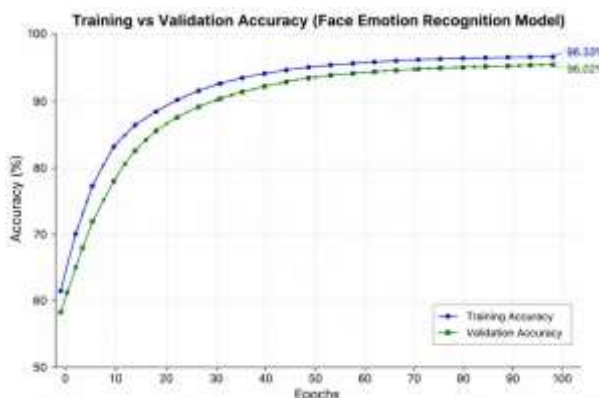


Fig -4: Training vs Validation Accuracy - Face Model

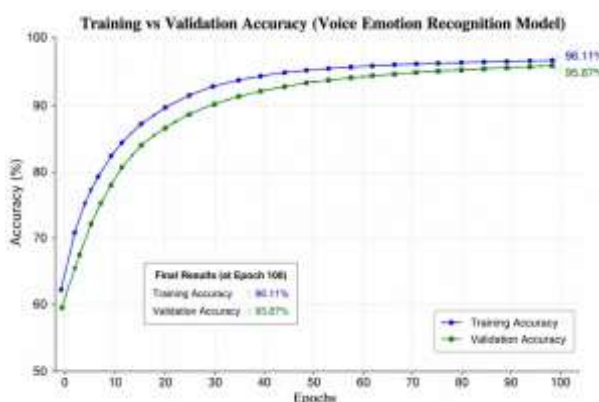


Fig -5: Training vs Validation Accuracy - Voice Model

Beyond the model-level evaluation, the Streamlit application itself was tested using static checks, structural (white-box) testing, and behavioural (black-box) testing. It accepted valid image and audio uploads without issue, returned a prediction within a couple of seconds on an ordinary laptop, and displayed the predicted emotion and

confidence score clearly. Taken together with the accuracy results, this gives reasonable confidence that the functional and non-functional requirements set out earlier in this paper were actually met, rather than simply assumed.

## 7. CONCLUSION

This paper set out to build and evaluate a system that reads human emotion from two independent channels - a face image and a voice recording - using a CNN for the former and an MFCC-driven Conv1D network for the latter, both presented through a single Streamlit interface. Trained on the Balanced RAF-DB and RAVDESS datasets, the two models each reached approximately 96% accuracy on test data, a result that supports the broader claim that a deep-learning pipeline of this kind can be accurate, lightweight, and usable by someone without a machine-learning background, with reasonably clear applications in healthcare, education, and other settings where it is useful to know something about a user's emotional state.

## Future Scope

- Add support for a live webcam feed and microphone input, so emotion can be read as it happens rather than from an uploaded file.
- Extend the label set beyond the seven discrete categories used here to include blended or in-between emotional states.
- Improve robustness to degraded input, such as low-resolution images, poor lighting, and noisy or low-quality audio.
- Deploy the system as a hosted web service or mobile application so it is not tied to a local Python environment.
- Explore combining the face and voice predictions into a single joint estimate rather than treating the two channels independently.

## ACKNOWLEDGEMENT

I would like to thank the Department of Computer Science and Engineering at the University College of Engineering, Science & Technology, JNTUH Hyderabad, for the guidance and support extended throughout this work.

## REFERENCES

1. P. Ekman and W. V. Friesen, Facial Action Coding System: A Technique for the Measurement of Facial Movement. Consulting Psychologists Press, 1978.
2. A. Mollahosseini, D. Chan and M. H. Mahoor, "Going deeper in facial expression recognition using deep neural networks," Proc. IEEE Winter Conf. on Applications of Computer Vision (WACV), 2016.
3. I. J. Goodfellow, Y. Bengio and A. Courville, Deep Learning. MIT Press, Cambridge, MA, USA, 2016.
4. Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.
5. S. R. Livingstone and F. A. Russo, "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)," PLoS ONE, vol. 13, no. 5, 2018.
6. S. Li and W. Deng, "Reliable Crowdsourcing and Deep Locality-Preserving Learning for Facial Expression Recognition in the Wild," Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2019.
7. D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Int. Conf. on Learning Representations (ICLR), 2015.
8. F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.
9. A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed., O'Reilly Media, 2022.
10. TensorFlow Developers, TensorFlow: An End-to-End Open Source Machine Learning Platform. Available: <https://www.tensorflow.org>