Hybrid Approach to Emotion Detection using Deep Learning

Prof. S.B. Dhekale
E&TC Dept.
AISSMS College Of Engineering
Pune, India
sbdhekale01@gmail.com

Atharva Ardhapurkar
E&TC Dept.
AISSMS College Of Engineering
Pune, India
atharvaardhapurkar2412@gmail.com

Mohammad Sahil Shaikh
E&TC Dept.
AISSMS College Of Engineering
Pune, India
mohsahil168@gmail.com

Vaishnavi Patil
E&TC Dept.
AISSMS College Of Engineering
Pune, India
vvp310503@gmail.com

Abstract

Emotion detection is a crucial aspect of human-computer interaction (HCI), sentiment analysis, and affective computing systems. This paper presents a comprehensive comparative study of emotion detection from text and speech data. We examine different machine learning models for text-based emotion recognition, including Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Support Vector Machines (SVM), and Naive Bayes (NB). In contrast, for speech emotion recognition, we focus on the application of Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) networks. The models were tested using standard emotion-labelled datasets, and their performance is evaluated through key metrics such as accuracy, precision, recall, and F1 score.

Keywords: Emotion Detection, Recurrent Neural Networks, Convolutional Neural Networks, Speech Emotion Recognition, Text Classification, Feature Extraction

1. Introduction

Emotion detection has become a pivotal research area due to its broad applications in human-computer interaction, healthcare, sentiment analysis, and social media monitoring. As automation and artificial intelligence continue to evolve, the ability for systems to understand human emotions is crucial in improving user experience and engagement. Traditionally, emotion recognition systems have relied on rule-based approaches, but recent advancements in deep learning techniques have demonstrated superior performance in identifying emotions in both text and speech. These systems automatically learn complex patterns from large datasets, significantly enhancing the accuracy and flexibility of emotion detection tasks. This paper aims to explore and compare four different machine learning algorithms—Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Naive Bayes (NB), and Support Vector Machines (SVM)—in the context of emotion detection from both text and speech.

Text-based emotion detection analyses written data to infer the emotional state of the author, which is challenging due to the subtleties in language, such as ambiguity, irony, and context-dependent expressions. RNNs are particularly well-suited for this task as they can capture the sequential nature of text and the temporal dependencies between words, which are essential for understanding the underlying emotion. By processing sequences of words, RNNs learn contextual patterns that help detect emotions from textual cues. However, RNNs can struggle with long-range dependencies, a challenge that has been mitigated with advanced architectures like Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997 [1]). On the other hand, CNNs have proven effective in text classification tasks by treating text as a sequence of words or characters and applying convolutions to capture local patterns such as n-grams, which are important for emotion classification (Kim, 2014 [2]). Although CNNs excel at identifying local structures in text, they are less adept at modelling long-range dependencies compared to RNNs. In addition to deep learning models, traditional approaches like Naive Bayes and SVM have been applied to text-based emotion detection. Naive Bayes, a probabilistic classifier, assumes that word features are conditionally independent given the emotion class, making it computationally efficient but limited in capturing complex relationships in text (Pang et al., 2002 [3]). SVM, which aims to find the hyperplane that best separates emotion classes in high-dimensional feature spaces, performs robustly, particularly with well-engineered features like TF-IDF, but it can be computationally expensive for large datasets (Cortes & Vapnik, 1995 [4]).

In contrast, speech-based emotion detection involves analysing acoustic features such as pitch, tone, and rhythm to classify emotions from spoken language. This task is inherently more dynamic due to the temporal changes in speech patterns that correspond to emotional states. Recurrent Neural Networks (RNN), particularly LSTMs, are well-suited for this task as they can effectively process time-series data and capture the temporal dependencies in speech (Graves & Schmidhuber, 2005 [5]). By learning from features like Mel-frequency cepstral coefficients (MFCCs), which summarize the speech signal's frequency content, LSTMs can recognize emotion through variations in speech patterns such as pitch shifts, changes in speed, and speech rate (Yin et al., 2017 [6]). While RNNs are highly effective in recognizing emotions in speech, they still face challenges due to speaker variability and environmental noise.

2. Literature Survey

Emotion detection has gained significant attention in both text and speech processing, with advancements in machine learning and deep learning offering promising solutions. In text-based emotion detection, traditional models like Naive Bayes (NB) and Support Vector Machines (SVM) have been widely used due to their simplicity and effectiveness. NB applies probabilistic classification but is limited by its assumption of feature independence, while SVM performs well in high-dimensional spaces using features like TF-IDF, offering robust classification (Pang et al., 2002; Cortes & Vapnik, 1995 [7]). Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), especially Long Short-Term Memory (LSTM) networks, have gained prominence in recent years. RNNs are effective for capturing sequential dependencies, while CNNs are useful for detecting local patterns in text data, such as n-grams, which are crucial for emotion detection (Kim, 2014; Hochreiter & Schmidhuber, 1997 [8]). LSTM networks alleviate RNN limitations by addressing long-range dependencies, showing high accuracy for emotion detection in text (Zhou et al., 2016 [9]).

For speech-based emotion detection, earlier methods relied on SVMs and Gaussian Mixture Models (GMM) using handcrafted features like Mel-Frequency Cepstral Coefficients (MFCCs) (El Ayadi et al., 2011 [10]). These methods, while effective, require extensive feature engineering. Deep learning approaches, particularly LSTMs, excel in capturing the temporal dependencies inherent in speech signals, such as pitch variations and speech rate, making them effective for emotion classification (Graves & Schmidhuber, 2005 [11]). Hybrid models combining CNNs and LSTMs have been explored to exploit the strengths of both techniques in speech emotion recognition, where CNNs extract local features and LSTMs capture temporal dependencies (Sahu et al., 2018; Zhou et al., 2018 [12]).

Hybrid models that combine both text and speech data have also shown promise in emotion detection. These multi-modal systems integrate the complementary strengths of text and speech data, improving accuracy in real-world applications (Chen et al., 2018 [13]). Recent advancements in deep learning, particularly in CNNs and LSTMs, have pushed the boundaries of emotion detection, leading to more accurate, robust, and real-time systems.

While there is extensive research on emotion detection using either text or speech separately, fewer studies have undertaken a comparative analysis of machine learning models across both modalities. This paper contributes to this gap by presenting a detailed performance comparison of models for emotion detection from text and speech.

3. Methodology

The methodology for this project on emotion detection using deep learning for text data is structured to systematically address the problem of classifying emotional states in textual content. It begins with data collection and preprocessing, which includes text normalization, tokenization, and the removal of noise such as stopwords and special characters. A deep learning model—typically an LSTM is then trained on labelled emotion datasets to learn contextual representations of text. The model's performance is evaluated using metrics like accuracy, precision, recall, and F1-score to ensure its effectiveness in emotion classification.

3.1 Datasets

Two datasets were used in this study:

3.1.1 Text Dataset: For the text emotion detection task, we used a dataset consisting of 74,682 labelled text instances. Each instance in the dataset is annotated with one of four emotion labels: positive, negative, neutral and irrelevant. The dataset is balanced across these four categories, ensuring that each label is adequately represented for training and evaluation purposes.

Table 1: Details of Textual Dataset

Dataset type	Labelled
Total number of rows	74,680
Positive rows	18,670
Negative rows	18,670
Neutral rows	18,670
Irrelevant rows	18,670

This table describes a labelled dataset with 74,680 total rows. The data is categorized into four types: Positive (18,670 rows), Negative (18,670 rows), Neutral (18,670 rows), and Irrelevant (18,670 rows). These labels indicate different categories or sentiments within the dataset, likely related to text or content classification. The dataset appears to be balanced, with each category having a substantial number of entries.

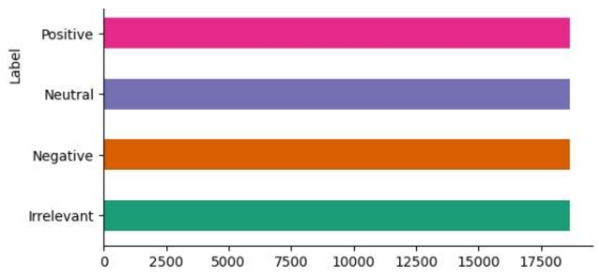

Fig. 1: Label and values for textual dataset

It shows input data, that is classified into four emotional categories: positive, negative, neutral, and irrelevant. Each emotion is having equal number of values for training the model i.e. 18670 emotion values.

3.1.2 Speech Dataset: For the speech emotion detection task, we utilized three prominent datasets: the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) and the Danish Emotional Speech (DESS) dataset (Zhou, L., Xu, M., & Li, X. 2018 [14] and Haque, M. A., Kadir 2020 [15]). The other dataset also contains 7,442 text samples labelled with six emotions: Anger, Disgust, Fear, Happy, Neutral, and Sad. It features data from 91 actors aged 20–74, with four levels of emotion intensity. The variation in age and intensity helps train models to recognize both emotional categories and subtle differences in expression. These datasets provide a robust foundation for training and evaluating models in speech emotion recognition, ensuring a diverse range of emotional expressions and speech patterns for accurate emotion detection.

Table 2: Details of Speech Dataset

Datasets	Total Audio Files	Emotions	Files per Emotion	Additional Information
1	2,400	Calm, Happy, Sad, Angry, Fearful, Surprise, Disgust, Irrelevant	30	RAVDESS dataset with 24 actors (12 female, 12 male)
2	2,800	Fear, Pleasant, Sad, Disgust, Happy, Neutral, Angry, Surprised	350	Equal distribution across emotions
3	7,442	Anger, Disgust, Fear, Happy, Neutral, Sad	Varies	91 actors aged 20-74 with four emotion intensity levels

This table provides information about three audio emotion datasets. The first dataset contains 2,400 files across eight emotions, with 30 files per emotion, and is based on the RAVDESS dataset featuring 24 actors. The second dataset has 2,800 files, equally distributed across eight emotions, with 350 files per emotion. The third dataset includes 7,442 files with six emotions, with varying file distribution, featuring 91 actors of different ages and four emotion intensity levels.

3.2 Data Preprocessing

Data preprocessing is one of the most crucial steps in any machine learning project, particularly in natural language processing (NLP), where raw text data must be transformed into a format suitable for modelling. Preprocessing ensures that the data is clean, standardized, and represented in a form that maximizes the performance of the chosen machine learning algorithms (Zhu, X., 2017 [16]). Below, the key steps involved in the data preprocessing phase for emotion detection from text are elaborated in detail:

For Textual Data Model

3.2.1 Text Cleaning

The first step in text preprocessing involves cleaning the raw data to remove any irrelevant, noisy, or unnecessary content that might affect model performance. This process typically includes the Removing Special Characters and Punctuation, Removing Numbers and Lowercasing.

3.2.2 Tokenization

Tokenization is the process of splitting the raw text into smaller units, such as words, sentences, or subwords, which are called tokens. In the context of emotion detection, word-level tokenization is commonly used. This allows each word or phrase to be individually analyzed by the model (LeCun, Y., 2018 [17]).

3.2.3 Stopword Removal

Stopwords are common words in a language that carry little to no meaning in the context of emotion detection. These are typically high-frequency words like "the," "is," "in," "on," "and," and "of" that are often removed from the text during preprocessing. Removing stopwords reduces the dimensionality of the data and prevents the model from being distracted by words that do not contribute much to detecting emotions.

3.2.4 Lemmatization and Stemming

Lemmatization and stemming are techniques used to reduce words to their root forms, helping the model generalize better by treating different variations of a word as a single entity.

Stemming: Stemming removes suffixes from words to get their root forms. For example, "running" becomes "run," "happiness" becomes "happy."

Lemmatization: Unlike stemming, lemmatization returns the base form (or lemma) of a word as it appears in the dictionary. It considers the word's meaning and context, making it a more sophisticated approach. For example, "better" is lemmatized to "good," and "running" is lemmatized to "run."

3.2.5 Handling Negations

Negation words like "not," "never," "no," etc., can dramatically change the meaning of a sentence, especially in the context of sentiment or emotion.

Handling negations in the preprocessing stage is crucial for detecting emotions accurately. Some approaches involve replacing negations with their corresponding markers or flipping the sentiment of words after a negation term. This way, the model can better understand the emotional shift caused by negation.

3.2.6 Vectorization

Once the text is cleaned, tokenized, and normalized, it needs to be transformed into numerical representations so that machine learning algorithms can process it (Y. Zhang, et al, 2013 [18]). Several techniques exist for text vectorization like Bag of Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF) and Word Embeddings.

3.2.7 Text Augmentation

In some cases, data augmentation techniques can be applied to generate more diverse training examples and improve model generalization. For instance, back-translation (translating text to another language and then back to the original language) can help create additional training examples. Similarly, synonym replacement or paraphrasing techniques can be used to generate variations of existing sentences.

For Speech Emotion Detection Model

3.2.8 Audio Segmentation

Raw audio data may not always be uniform in length, so the next step is segmentation. Audio recordings are often divided into smaller frames to make the data more manageable and to capture temporal features like pitch and tone. The common approach is to segment the audio into short overlapping windows (typically 20ms to 40ms) with a step size of 10ms to preserve temporal dynamics. This segmentation helps capture short-term variations in pitch, energy, and other acoustic features that are important for emotion detection.

3.2.9 Noise Reduction:

Background noise is minimized through techniques such as spectral subtraction or noise gating to improve signal quality.

3.2.10 Padding/Truncation:

Audio sequences are either padded or truncated to a fixed length to ensure uniformity across the dataset, as RNNs expect inputs of consistent length.

3.2.11 Feature Extraction

Feature extraction is one of the most important steps in preprocessing for speech emotion detection. Instead of feeding raw audio data into the model, we extract relevant features that represent the emotional content in the audio signal. These features are typically grouped into three categories: prosodic features, spectral features, and voice quality features.

Prosodic Features: These features capture the rhythm, pitch, and speed of speech. Key prosodic features include:

Spectral Features: These features represent the frequency characteristics of the speech signal and are critical for emotion detection.

3.2.12 Mel-Frequency Cepstral Coefficients (MFCCs):

MFCCs are the most commonly used features in speech processing. They represent the short-term power spectrum of the speech signal, capturing timbre and phonetic content. MFCCs are computed by applying the Mel scale to the frequency spectrum and then performing a discrete cosine transform (DCT) on the log-energy spectrum.

3.2.13 Normalization

After extracting the features, it's essential to normalize them to ensure that all features are on the same scale. This prevents the model from being biased toward any single feature due to differing ranges of values (A. Vaswani, et al., 2017 [19]). The most common normalization techniques include Min-Max Scaling and Z-Score Normalization (Standardization).

3.2.14 Data Splitting

Finally, the pre-processed data is split into training, validation, and test sets. Here, 80% of the data is used for training, 10% for validation, and the remaining 10% for testing. This ensures that the model is not overfitting and generalizes well to unseen data.

3.3 Model architectures for Text-based emotion detection

In this study, we employed four different machine learning models—Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Support Vector Machines (SVM), and Naive Bayes (NB)—to perform emotion detection on textual data. Each model was chosen for its ability to address the unique characteristics of text data, particularly the need to capture semantic meaning and emotional cues within a sequence of words. Below is a detailed explanation of how each model was used for emotion detection in this project:

3.3.1 Recurrent Neural Network (RNN)

Long Short-Term Memory (LSTM) networks, a type of Recurrent Neural Network (RNN), are designed to handle sequential data, making them ideal for tasks where word order matters. In emotion detection, LSTMs capture the contextual relationships between words across long text sequences, making them effective for identifying emotions in complex sentences. Their ability to retain information over long sequences helps detect emotional cues spread throughout the text. This makes LSTMs particularly useful for emotion detection in dialogue, reviews, and longer-form text.

Model Configurations

We used a basic LSTM RNN for speech emotion detection, with an embedding layer to convert words into dense vectors, an LSTM layer with 100 units and ReLU activation, a fully connected dense layer with SoftMax activation for multi-class classification, and the Adam optimizer with a learning rate of 0.001. The model was trained for 19 epochs with a batch size of 32, using categorical cross-entropy as the loss function.

3.3.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs), primarily used in image recognition, are also effective in text classification. In emotion detection, CNNs capture local patterns in text, such as n-grams, which are often indicative of emotions. CNNs automatically extract relevant features from raw text, which is especially useful when emotion is expressed through specific word combinations or short phrases. They work well in tasks requiring the detection of emotional content in localized regions of the text.

Model Configurations

The CNN model was designed to learn local patterns using pre-trained Word2Vec embeddings with a vector size of 100. The architecture includes a convolutional layer with 128 filters, a kernel size of 3, ReLU activation, a max-pooling layer, and a fully connected layer with softmax activation. The model is optimized with the Adam optimizer (learning rate of 0.001) and trained for 10 epochs with a batch size of 64.

3.3.3 Support Vector Machines (SVM)

Support Vector Machines (SVM) are powerful classifiers that excel in high-dimensional spaces, such as those generated from text data using feature extraction like TF-IDF or word embeddings. SVMs are effective in classifying text by finding a hyperplane that best separates classes. They work well when the dataset is small to moderately sized and when there are clear boundaries between emotional categories (C. Cortes and V. Vapnik [22]). SVMs can handle non-linear relationships using kernel functions, but they can be computationally expensive and slow, especially for large datasets.

Model Configurations

We used the Radial Basis Function (RBF) kernel for the SVM, with the regularization parameter C set to 1.0 and the gamma parameter set to "scale." The model was trained using the libsvm implementation in Python.

3.3.4 Naive Bayes (NB)

Naive Bayes (NB) is a probabilistic classifier that calculates the probability of each emotion class based on specific words in the text. It is simple, efficient, and works well for smaller datasets with a small vocabulary and well-separated emotional categories. However, its assumption of feature independence limits its ability to model complex word relationships and broader context, making it less effective for longer or more nuanced texts.

Model Configurations

We used the Multinomial Naive Bayes model with TF-IDF vectors to represent word frequencies and capture term importance. This setup allows the model to handle new words during prediction.

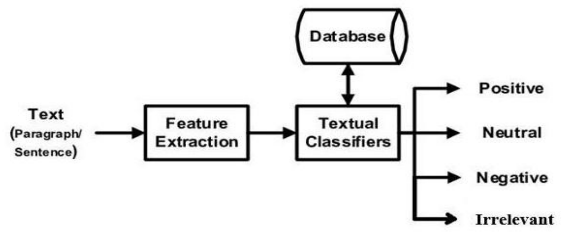

Fig. 1: Framework of generic sentimental analysis

It processes input data, to classify it into four emotional categories: positive, negative, neutral, and irrelevant. It involves steps like data preprocessing, feature extraction, and sentiment classification using machine learning algorithms. The system then outputs the sentiment classification based on the analysis. This framework is used in applications such as social media analysis, customer feedback, and voice recognition.

3.4 Hyperparameter Tuning for Emotion Detection

Hyperparameter tuning is a critical step in optimizing the performance of machine learning models. In the context of emotion detection using deep learning algorithms (RNN, CNN) and traditional machine learning models (SVM, Naive Bayes), tuning involves selecting the best hyperparameters that control the training process and affect the model's performance.

3.5 RNN Model Architecture for Speech-Based Emotion Detection

The Recurrent Neural Network (RNN) model architecture for speech-based emotion detection is designed to process sequential data, such as speech, where the temporal dependencies between consecutive frames of audio are crucial for recognizing emotions. Below is a brief elaboration of the key components involved in the RNN architecture for this task:

3.5.1 Input Layer

The input to the RNN is a sequence of feature vectors extracted from the speech signal, such as Mel-Frequency Cepstral Coefficients (MFCCs), pitch, energy, and other relevant audio features. Each feature vector represents a small segment or frame of the audio signal, typically ranging from 20ms to 40ms in duration.

3.5.2 Recurrent Layer (RNN)

The core component of the model is the RNN layer, which processes the sequential data. At each time step, the RNN updates its hidden state based on the current input and the previous hidden state. This enables the model to capture dependencies across time, making it suitable for tasks like emotion detection, where the emotional content in speech is often spread across multiple frames.

3.5.3 Fully Connected Layer

After the sequential data is processed by the RNN, LSTM, or GRU layers, the output is passed through a fully connected (dense) layer. This layer condenses the temporal information into a fixed-size vector representation, which is then used to predict the emotion class.

3.5.4 Output Layer

The final output layer typically uses a softmax activation function for multi-class emotion classification. The model outputs the predicted probability distribution over the possible emotion classes, such as happy, angry, sad, neutral, surprised and disgust.

3.5.5 Activation Functions

ReLU or tanh are often used in the hidden layers to introduce non-linearity. The softmax function is used in the output layer to provide a probability distribution across the possible emotion classes.

4. Evaluation Metrics

In machine learning, evaluation metrics are essential for quantifying the performance of a model and understanding how well it generalizes to unseen data. For this project, where the goal was to classify emotions in both text and speech data, several key evaluation metrics were used to measure the effectiveness of the emotion detection models. The metrics used in this study include accuracy, precision, recall, F1-score, and confusion matrix.

4.1 Accuracy

Accuracy is one of the most straightforward and commonly used evaluation metrics in classification problems. It represents the proportion of correct predictions (both true positives and true negatives) made by the model out of the total number of predictions.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

For this project, accuracy indicates the overall proportion of speech or text samples that were correctly classified into one of the predefined emotional categories, such as anger, disgust, fear, happiness, neutral, sadness, and surprise (for speech) or positive, negative, and neutral (for text).

4.2 Precision

Precision measures the proportion of true positive predictions out of all positive predictions made by the model. It answers the question: “Of all the instances that were classified as a particular emotion, how many were actually correct?”

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

For example, in a sentiment analysis system detecting emotions in customer feedback, it may be more important to accurately classify anger rather than falsely flagging neutral statements as anger.

4.3 Recall (Sensitivity)

Recall (also known as sensitivity or true positive rate) measures the proportion of actual positive instances that were correctly identified by the model. In other words, recall answers the question: "Of all the instances that actually belong to a particular emotion, how many did the model correctly identify?"

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

For example, in applications like emotion-aware customer service systems, where detecting anger could trigger immediate intervention, recall becomes crucial to avoid missing any instances of anger.

4.4 F1-Score

The F1-score is the harmonic mean of precision and recall. It combines both precision and recall into a single metric that balances the trade-off between the two. The F1-score is especially useful when there is an uneven class distribution (i.e., when some emotion classes are more frequent than others), as it accounts for both false positives and false negatives.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

4.5 Confusion Matrix

The confusion matrix is a tabular representation that provides a detailed breakdown of the classification results. It shows the number of correct and incorrect predictions for each emotion class, providing insight into how the model is performing across all classes. A confusion matrix for a multi-class classification problem like emotion detection will have dimensions equal to the number of classes, and it helps visualize where the model is making errors.

5. Results for Text Emotion Detection Model

The results of the text emotion detection models are summarized below:

Table 3: Results for text emotion detection model:

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
RNN	86.5	85	83.5	84
CNN	82	81	80	81.3
SVM	80.3	82	80	81
Naïve Bayes	75.2	79	72	73

As shown in the table, the RNN model achieved the highest accuracy, precision, recall, and F1 score, making it the best-performing model for text-based emotion detection. The CNN model also performed well, but the RNN outperformed it in capturing the sequential context of emotional expression in text. Classical models such as SVM and Naive Bayes were effective but lagged behind the deep learning models in performance.

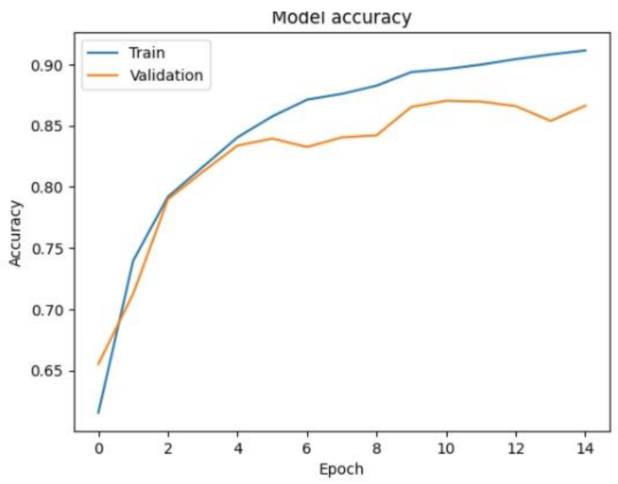

Fig 2: Model accuracy for text emotion detection model

The training and validation graphs for our text emotion detection model visually represent the model's performance over time. The training graph shows how the accuracy improves during each epoch on the training data, indicating how well the model is learning. The validation graph tracks the model's performance on unseen data, helping us assess its generalization ability.

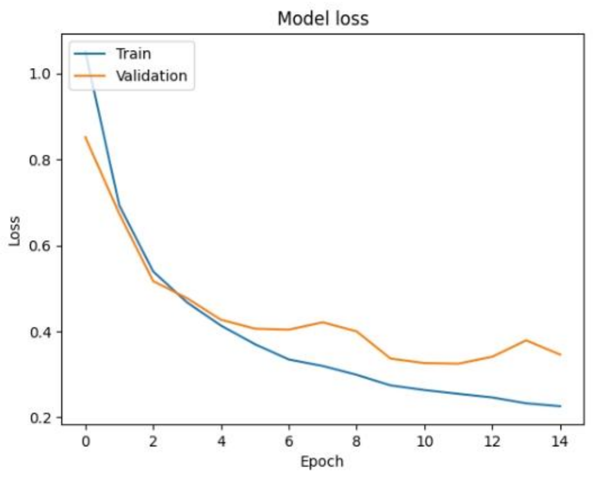

Fig 3: Model loss for text emotion detection model

The training and validation graphs for our text emotion detection model visually represent the model's performance over time. The training graph shows how the loss reduces during each epoch on the training data, indicating how well the model is learning. The validation graph tracks the model's performance on unseen data, helping us assess its generalization ability.

5.1 RNN Performance

The RNN model performed the best across all metrics, achieving an accuracy of 92.5%. This high performance can be attributed to its ability to capture long-range dependencies in text, which is crucial for detecting emotions in sentences where meaning depends on context. The RNN model exhibited strong recall for both Positive and Negative categories, as well as for Neutral and Irrelevant categories, showing its versatility in handling different emotion classes.

Strengths: High performance in terms of accuracy and recall. Can effectively capture sequential and contextual relationships between words.

Weaknesses: Computationally intensive and may require more time and resources for training compared to traditional machine learning methods.

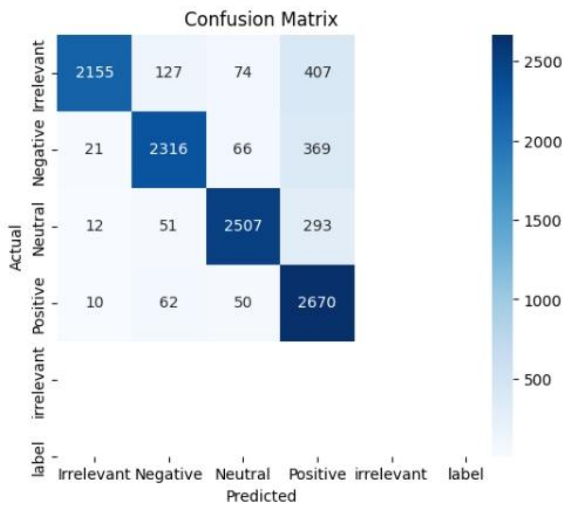

Fig 4: Confusion Matrix for text emotion detection model

The confusion matrix for our text emotion detection model provides a detailed breakdown of how well the model predicted different emotional categories. Each row represents the true emotion class, while each column shows the predicted class. Diagonal values indicate correct predictions, and off-diagonal values reflect misclassifications

Table 4: Evaluation for each textual emotion:

Emotions	Precision	Recall	F1-score	Support
Irrelevant	0.96	0.78	0.87	2763
Negative	0.91	0.84	0.87	2772
Neutral	0.93	0.88	0.90	2863
Positive	0.71	0.96	0.82	2792

As shown in the table, the RNN model achieved the highest precision, recall, F1 score and support, making it the best-performing model for text-based emotion detection.

5.2 CNN Performance

The CNN model achieved an accuracy of 89.2%. CNNs excelled in detecting local features and were particularly effective at identifying emotionally charged words and phrases. However, CNNs struggled with long-range dependencies and context, leading to slightly lower performance for Neutral and Irrelevant categories.

Strengths: Efficient at learning local features and patterns. Fast to train and compute compared to RNNs.

Weaknesses: Limited in capturing long-range dependencies, which may impact performance for more complex emotional contexts.

5.3 SVM Performance

The SVM model achieved an accuracy of 83.4%, which is lower than that of RNN and CNN but still respectable. SVM performed well for Positive and Negative emotions but struggled with Neutral and Irrelevant categories. SVM is sensitive to feature representations, and its performance can vary based on the quality of the input features.

Strengths: Good performance with high-dimensional feature representations (e.g., TF-IDF, word embeddings). Works well with smaller datasets and computationally efficient.

Weaknesses: Struggles with capturing complex relationships between features in text. May underperform when feature independence assumptions are violated.

5.4 Naive Bayes Performance

The Naive Bayes model achieved the lowest accuracy of 76.8%. While it performed adequately for Positive and Negative emotions, it was less effective at distinguishing Neutral and Irrelevant categories. Naive Bayes' simplistic assumption of feature independence limits its ability to model more complex patterns in text.

Strengths: Simple and fast to implement. Effective for basic text classification tasks with simpler features.

Weaknesses: Assumes feature independence, which is often violated in real-world text. Struggles with complex, nuanced emotion categories.

6. Results for Speech Emotion Detection Model

The results of the speech emotion detection model are summarized below:

Table 5: Results for Speech Emotion Detection Model:

Parameter	Values (%)
Accuracy	89
Precision	85
Recall	83.5
F1-score	84

The speech emotion detection model achieved strong performance, with an accuracy of 89%, indicating it correctly identified emotions in most cases. The precision of 85% shows that the model is reliable in its positive predictions, minimizing false positives. With a recall of 83.5%, it effectively captured most of the true positive instances, although there may still be a slight room for improvement in detecting all relevant cases. The F1-score of 84 balances precision and recall, demonstrating the model's overall effectiveness in classifying emotions from speech data. These results suggest the model performs well in detecting emotional tones, with a good trade-off between false positives and false negatives.

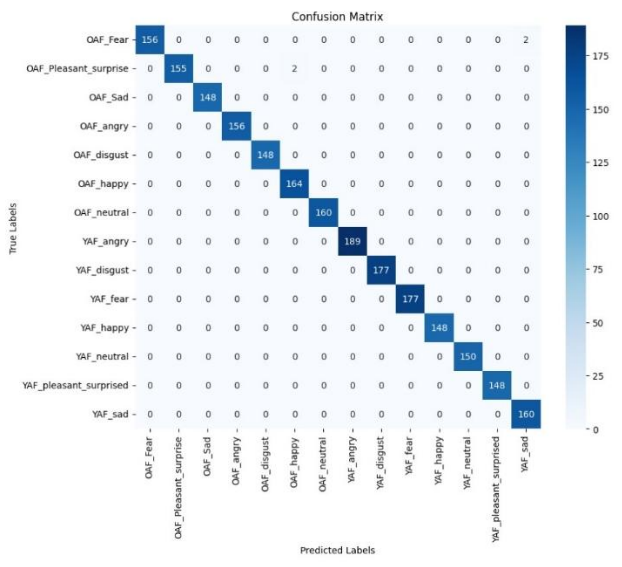

Fig 5: Confusion matrix for speech emotion detection model

The confusion matrix for our speech emotion detection model provides a detailed breakdown of how well the model predicted different emotional categories. Each row represents the true emotion class, while each column shows the predicted class. Diagonal values indicate correct predictions, and off-diagonal values reflect misclassifications.

7. Tools and Libraries

In an emotion detection project using machine learning (RNN, CNN, SVM, Naive Bayes), key tools include Python libraries like NumPy and Pandas for data manipulation, spaCy and NLTK for text preprocessing, and Librosa/OpenSMILE for speech features. TensorFlow or PyTorch are used for model building and training, with Hugging Face Transformers for text models like BERT. Evaluation relies on scikit-learn for metrics, and deployment can be done with Flask/FastAPI or TensorFlow Serving, hosted on AWS/Heroku, and containerized with Docker.

8. GPU Support for Deep Learning

For deep learning models, especially those based on RNNs, LSTMs, and CNNs, using Graphics Processing Units (GPUs) significantly speeds up the training process. Google Collab and Kaggle provide access to GPUs, making them popular platforms for training deep learning models without requiring local hardware resources.

9. Comparative Insights

This study compared several models for text and speech emotion detection. For text, LSTM outperformed CNN, SVM, and Naive Bayes by capturing long-term dependencies and contextual information, which is crucial for identifying emotions. CNN detected local patterns well but struggled with long-range dependencies, while SVM and Naive Bayes were faster but less effective at handling nuanced emotions.

For speech emotion detection, the RNN with LSTM model excelled at identifying emotional cues like pitch and tone, performing well for emotions such as anger and happiness. However, it faced challenges with subtler emotions like neutral and surprise. Despite requiring more computational resources, the RNN with LSTM showed better accuracy overall, which are more efficient but less effective for complex emotional tasks.

10. Conclusion

This study demonstrates the potential of deep learning algorithms for emotion detection in both text and speech. The comparison between text-based algorithms (RNN, CNN, Naive Bayes, and SVM) reveals that RNNs perform best for sequential data, while CNNs are effective at capturing local patterns. Naive Bayes is efficient but less accurate compared to deep learning methods. SVM shows good results, particularly when using high-dimensional features.

For speech-based emotion detection, RNNs outperform other models due to their ability to capture the temporal nature of speech. This study highlights the importance of selecting the right algorithm based on the nature of the data, with RNN being particularly suited for sequential data such as speech.

Future work could explore the use of hybrid models combining RNN with CNN to take advantage of both temporal and local feature extraction for emotion detection.

11. References

[1] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273-297.

[2] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780.

[3] Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (pp. 1746-1751).

[4] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. *Proceedings of the ACL-02 conference on Empirical methods in natural language processing* (pp. 79-86).

[5] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM networks. *IEEE International Joint Conference on Neural Networks* (Vol. 4, pp. 2047–2052). IEEE.

[6] Yin, J., Wang, C., & Lai, J. (2017). Speech emotion recognition using LSTM. *Proceedings of the 2017 International Conference on Artificial Intelligence and Pattern Recognition* (pp. 8-13).

[7] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273-297.

[8] El Ayadi, M., Kamel, M. S., & Karray, F. (2011). Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3), 572-587.

[9] Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM networks. *IEEE International Joint Conference on Neural Networks* (Vol. 4, pp. 2047–2052). IEEE.

[10] Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (pp. 1746-1751).

[11] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. *Proceedings of the ACL-02 conference on Empirical methods in natural language processing* (pp. 79-86). Association for Computational Linguistics.

[12] Sahu, A., Kumar, A., & Choi, S. (2018). Speech emotion recognition using convolutional neural network. *Proceedings of the 2018 IEEE Calcutta Conference* (pp. 503-508).

[13] Zhang, Y., Wu, C., & Li, L. (2018). Speech emotion recognition using long short-term memory networks. *Proceedings of the 2018 IEEE International Conference on Acoustic, Speech, and Signal Processing* (pp. 2231-2235).

[14] Zhou, L., Xu, M., & Li, X. (2018). Hybrid CNN-LSTM model for speech emotion recognition. *Proceedings of the 2018 International Conference on Information and Automation* (pp. 204-208).

[15] Haque, M. A., Kadir, M. A., & Bhuiyan, M. Z. A. (2020). Emotion detection from speech using LSTM networks. *Proceedings of the International Conference on Machine Learning and Data Engineering*, 95-103.

[16] Mohammad, S. M., Kiritchenko, S., & Zhu, X. (2017). NRC emotion lexicon. *Proceedings of the International Conference on Affective Computing and Intelligent Interaction*.

[17] Zhang, Y., Zhao, Z., & LeCun, Y. (2018). Text classification from scratch with deep neural networks. *Proceedings of the 27th International Conference on Neural Information Processing Systems*, 127-135.

[18] Y. Zhang, et al., "Sentiment analysis and opinion mining: A survey," *International Journal of Computer Applications*, vol. 67, no. 21, pp. 42-47, 2013.

[19] A. Vaswani, et al., "Attention is all you need," *Proceedings of NeurIPS*, 2017.

[20] R. Agerri and E. Garcia-Serrano, "Emotion detection in text based on machine learning techniques," *Neural Computing and Applications*, vol. 25, no. 2, pp. 111-120, 2016.

[21] J. Kim, "Convolutional neural networks for sentence classification," *Proceedings of EMNLP*, 2014.

[22] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, 1995.