Comprehensive Application of Facial Expression Recognition and Text Sentiment Analysis

Wei Jiayi
Wenzhou-Kean university
weijiay@kean.edu

Abstract

This work investigates the integration of facial expression recognition and natural language processing (NLP) for comprehensive emotion recognition. Utilizing Convolutional Neural Networks (CNN) for facial analysis and Transformer models for text sentiment analysis, the study aims to enhance the accuracy and robustness of emotion detection. The multimodal approach addresses the limitations of single-modal systems by combining visual and textual data to achieve a more holistic understanding of human emotions. Extensive statistical analysis, including Principal Component Analysis (PCA), Z-tests, T-tests, and Chi-square tests, validates the model's performance and robustness. The advantages and disadvantages of this project are then discussed and compared with existing models. The future improvement direction and application are deeply discussed. This study highlights the potential of multimodal emotion recognition technology in improving human-computer interaction and lays the groundwork for future advancements in emotion computing.

Keywords: FER, NLP, CNN, statistical analysis

CH1. Introduction

1.1 Motivation

In recent years, the integration of machine learning and artificial intelligence into all aspects of daily life has become increasingly common because computers have the ability to capture features and analyze them more quickly and accurately than humans(LeCun et al., 2015). At the same time, in daily life, emotion is always regarded as a very internal thing, but in the situation of communication with people, emotion is a very important factor to consider(Niedenthal et al., 2006). So, if there is a way to express what is inside, it will effectively promote the relationship between people.

Therefore, in this project, both facial expression analysis and text input analysis will be used to find the emotion of the object. In addition, this application helps machines understand human emotional judgments, which can significantly enhance human-computer interaction and provide a more personalized and empathetic user experience.

1.2 Literature Review

Emotion recognition enhances the user experience by making interactions more personal and empathic, so it has become an important aspect of human-computer interaction (HCI) (Picard, 2000). Emotion recognition can be done in a variety of ways, including facial expression analysis and text emotion analysis. In the past, there have been few studies that have combined face and text. But by applying them separately, we can get a rough idea of what face and text analytics are doing today.

First, Goodfellow et al. highlighted the power of convolutional neural networks (CNNS) in image classification tasks in their seminal book Deep Learning. Because CNNS consists of multi-layer convolutional filters, pooled layers, and fully connected layers, these layers can automatically learn the spatial hierarchy of features, which is suitable for three-dimensional facial analysis (Goodfellow et al., n.d.). So various architectures such as VGGNet, ResNet, and Inception have been applied to relevant facial analysis models to achieve state-of-the-art performance on benchmark datasets like FER-2013.

The facial expression recognition model (FER) applied in this project is to identify human emotions by analyzing facial images(Lijun Yin et al., 2006). With the reinforcement of deep learning, for example, CNNS can learn hierarchical features from raw pixel data, so the accuracy and efficiency of FER systems have been significantly improved(Pantic & Rothkrantz, 2001).

The existing applications of facial expression recognition are mainly in healthcare, education, and entertainment. Medically, FER has been used to monitor patient mood, detect mental health issues, and provide feedback during treatment (Choudhury et al., n.d.). In education, FER systems have been used to measure student engagement and improve student engagement in class by detecting student expressions and providing real-time feedback to educators (Whitehill et al., 2014). In terms of entertainment, FER technology enhances the user experience in gaming and virtual reality by adjusting responses to player emotions (Yannakakis & Hallam, 2011).

Second, in past research on textual sentiment analysis, sentiment analysis has made significant progress with advances in natural language processing (NLP), especially with the introduction of transformer-based models such as BERT (bidirectional encoder representation from transformers) and GPT (generation of pre-trained transformers). Hugging Face's deformation library provides pre-trained models that can be fine-tuned for specific tasks, including sentiment analysis. In this project, I used a pipeline from Hugging Face to analyze the text input provided by users.

In today's application of text analysis sentiment, the market is mainly focused on market research, such as companies need to use sentiment analysis to gauge public opinion on products and services, enabling them to make data-driven marketing decisions (G & Chandrasekaran, 2012). In some service industries, it also plays an important role, as

automated sentiment analysis can facilitate the understanding of user experience levels to improve the quality of service (Chatterjee et al., 2021). With the continuous development of the information age, this kind of text sentiment analysis is also used for detection on social media, which can always understand the public's perception of events and respond in time (Pak & Paroubek, 2010).

In this project, a more comprehensive approach to emotion recognition is provided through the integration of facial expression and text analysis. This multimodal approach allows for a more comprehensive understanding of an individual's emotions, with facial expressions and written expressions complementing each other and avoiding incidents where emotions are hidden.

1.3 Outline of the Essay

In this paper, we study the method of emotion analysis combining facial expression recognition and natural language processing (NLP) technology. Through comprehensive analysis of facial expressions and text input, more comprehensive and accurate emotion recognition results are provided.

In the introduction part, this paper first introduces the background and importance of emotion recognition and reviews the main research progress in the field of facial expression recognition and text emotion analysis in recent years. Despite significant progress in these two areas, in practical applications, emotion recognition of a single mode is often limited and difficult to fully capture the true emotional state of the user.

Next, the paper describes in detail the comprehensive emotion recognition model I proposed. The model consists of two main parts: one is the CNN model for facial expression recognition, and the other is the Transformer model for text sentiment analysis. Through the analysis of the result, the corresponding emotion label and confidence are output.

The experimental section presents the results of my experiments on facial recognition and text analysis. Facial expression recognition used the FER-2013 dataset, and text sentiment analysis used the pre-processed text sentiment dataset. Through a series of experiments, we verify the performance of the comprehensive model in the emotion recognition task and compare it with the single modal emotion recognition model. The results show that the comprehensive model has a significant improvement in accuracy and robustness, and can capture multi-modal emotional information of users more accurately.

In the evaluation part, I analyzed the experimental results in detail. Using statistical and hypothesis testing methods such as Z-tests, T-tests, and Chi-square tests, I assessed the statistical significance of facial expressions and text emotion recognition. At the same time, the entropy of facial and text emotion distribution is calculated, and the effect of emotion recognition is further quantified. By constructing a Markov joint state probability matrix and

observing state probability by joint vectorization, the dynamic change of emotion state is analyzed.

Finally, the paper summarizes the advantages and limitations of the comprehensive emotion recognition model and puts forward the future research direction. Future studies can further optimize the model structure, incorporate more emotional information sources, and explore the feasibility and effectiveness of practical applications.

Through the research of this paper, I show the potential of multi-modal emotion recognition technology, provide a new idea for improving the way of human-computer interaction, and lay a foundation for further research in the field of emotion computing.

1.4 Purpose of the Essay

The purpose of this essay is to explore and validate a comprehensive emotion recognition method that combines facial expression recognition and natural language processing (NLP). Specifically, this study aims to:

Improve the accuracy and comprehensiveness of emotion recognition: By integrating the two data sources of image and text, it can make up for the deficiency of single-modal emotion recognition, to improve the accuracy and robustness of overall emotion recognition.

Develop and evaluate multi-modal emotion recognition model: Build a comprehensive emotion recognition system, use convolutional neural network (CNN) to process facial expression images, use a Transformer model for text emotion analysis, and verify its effectiveness through experiments.

Analysis of dynamic changes in emotion states: Statistical and hypothesis testing methods are used to conduct in-depth analysis of multi-modal emotion recognition results, quantify and evaluate the changes and distribution of emotion states, and then provide more detailed emotion analysis.

Explore the feasibility of practical application: Discuss the potential of a comprehensive emotion recognition model in practical application.

CH2. Problem and Model Description

2.1 Problem Description

2.1.1 Problem Description of Face Recognition

Facial expression recognition aims to identify human emotions based on facial features. The main challenge lies in accurately detecting and classifying emotions such as happiness,

sadness, anger, fear, surprise, and disgust from facial images under different conditions. And find the relationship between emotional changes under continuous behavior.

2.1.2 Problem Description of NLP

Natural language processing (NLP) of sentiment analysis focuses on determining the emotions expressed in a text. This involves categorizing text into categories such as positive, negative, and neutral based on the emotions that words and phrases convey. Major challenges in this area include understanding context, dealing with irony and irony, disambiguating complex sentences, and dealing with the informal language often found in social media posts to accurately get machines to understand human speech.

2.1.3 Problem Description of Integrating Face Recognition and NLP

Combining facial expression recognition with natural language processing for sentiment analysis aims to gain a more complete understanding of human emotions through the combination of visual and textual data. The main challenge in this integration is to effectively merge information from these two different modes, which involves aligning the temporal and contextual aspects of facial expressions and text. Make sure the strengths of each mode complement each other to improve the accuracy of overall mood detection. In addition, it is necessary to analyze the consistency and coherence of the text and facial emotions. Find out the true emotion of the subject by making the text emotion and facial emotion corroborate or question each other in a multi-angle way.

2.2 Model Description

2.2.1 Model Description for Face Recognition

This section discusses the architecture and implementation details of the model, which is primarily based on a Convolutional Neural Network (CNN). In addition to the CNN itself, it consists of several layers overlaid, designed to extract features from the input images and classify them into one of seven emotional categories: anger, disgust, fear, happiness, neutral, sadness, and surprise. The model architecture includes the following components:

1. 3 convolution layers (Conv2D):

The convolutional layer can automatically learn local features from the input image, such as edge, texture, and other low-level features. The first convolutional layer outputs 32 feature maps, corresponding to the extraction of 32 different low-level features. Subsequent convolutional layers output 64 and 128 feature maps to extract more intermediate and advanced features, such as facial contours and facial features.

$$Conv(x)=(x*w+b)$$

Where x is the input image, w is the convolution kernel, and b is the bias.

2. Maximum pooling layer (MaxPooling2D):

The maximum pooling layer is behind each convolutional layer and is used to reduce the resolution of the feature map, thereby reducing the number of parameters and the amount of computation. At the same time, the maximum pooling layer can obtain some translation invariance, which makes the feature more robust to small translations.

$$\text{MaxPool}(x)=\max(x),$$

Where x is the input matrix, select the maximum value in the pooled window.

3. Flatten layer:

Flattening the output of the last pooled layer into a one-dimensional vector for input to the fully connected layer.

4. Fully connected layer (Dense):

The fully connected layer can combine the flattened feature vectors into higher-level feature representations. This layer has 128 neurons and is used to construct high-level feature representations of facial expressions.

$$y=W\cdot x+b$$

Where W is the weight matrix, x is the input, b is the bias, and y is the output.

5. Dropout layer:

The Dropout layer randomly disconnects some neuronal connections during training, effectively preventing overfitting.

6. Output layer (Dense):

The final output layer has seven nodes corresponding to seven different facial expression categories (such as anger, disgust, fear, etc.). Use the softmax activation function to output a probability score for each category.

$$softmax(zi)=\frac{e^{zi}}{\sum_{j} e^{zj}}$$

Where zi is the score for category i.

7. Optimizer and loss function:

Adam optimizer is used for training, it is an adaptive learning rate optimization algorithm, that can accelerate the convergence speed. Use categorical_crossentropy as a loss function, which is suitable for multi-classification problems and can minimize the cross-entropy loss between the predicted probability distribution and the true label.

Through layer-by-layer feature extraction and nonlinear transformation, the CNN architecture can automatically learn the hierarchical feature representation of facial expressions from the original pixel data, and finally complete the task of facial expression classification.

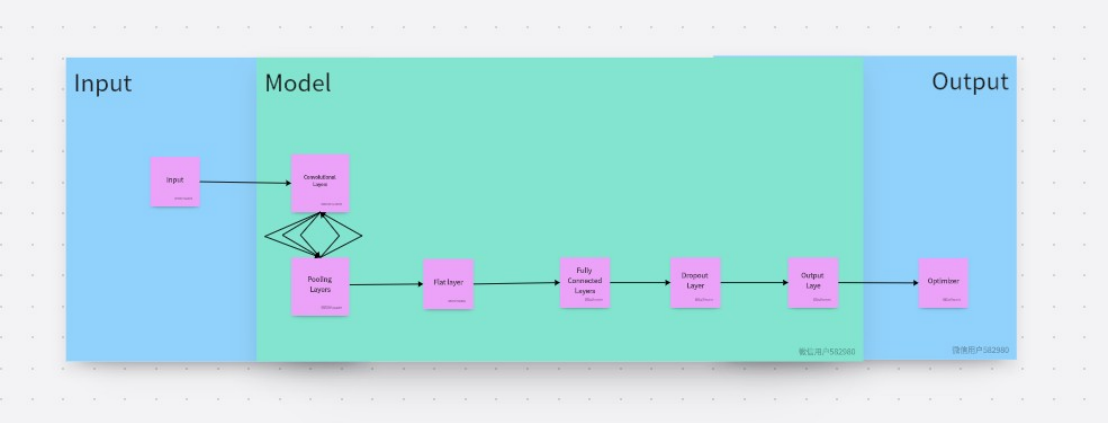

Figure 1 Model of Face Recognition

2.2.2 Model Description for NLP

This section discusses the architecture and implementation of the NLP model, which utilizes the pre-trained Transformer model. The NLP model mentioned here utilizes the Transformer architecture, specifically using the sentiment analysis pipeline from the hug Face library. Transformers are known for their ability to understand context and semantics in text, which makes them very effective in sentiment analysis. The main structure consists of the following:

1. Text input: Receives text entered by the user.
2. Tokenization: The use of pre-trained word separators to convert text into tokens and IDs that the model can process.
3. Model reasoning: The input data after word segmentation is passed to the pre-trained model to obtain logits.
4. Apply softmax: Convert logits to probability distributions.
5. Emotion classification: Determine the emotion label of the text based on the probability distribution.

2.3 Algorithms

Several algorithms are involved throughout the project, including convolutional neural networks (CNNS), pre-processing steps, sentiment analysis (using the Transformer model), and statistical analysis (such as PCA, Z-test, T-test, and Chi-square test). Below is a detailed description of each algorithm, its role in the code, and related formulas.

2.3.1 Convolutional neural networks (CNNS) are used for facial expression recognition

1. Convolutional Layer: It is used to extract features in the image, such as edges, textures, etc.
2. Max Pooling Layer: Reduce the size of the feature map, retain important features, reduce calculation and overfitting.
3. Flatten Layer: Flattens the multidimensional input into one dimension for passing into the fully connected layer.
4. Fully Connected Layer: Classification is performed by combining the features of the previous convolution and pooling layers.
5. Dropout: A certain percentage of neurons are randomly discarded during training to prevent overfitting.
6. Softmax: Converts the output into a probability distribution for classification.

2.3.2 Statistical analysis

1. PCA: Used for dimensionality reduction, projecting the facial emotion score and the text emotion score onto a principal component, and finding the projection direction of the maximum variance by maximizing the projection variance.
2. Z-test: Used to test whether the mean difference between the facial emotion score and the text emotion score is significant.
3. T-test: Used to test whether the mean difference between the facial emotion score and the text emotion score is significant, and is suitable for cases where the variance is unknown.
4. Chi-square test: Used to test whether there is a significant difference in the distribution of facial emotion scores and text emotion scores.
5. Information entropy: Used to calculate the uncertainty of facial emotion scores and text emotion scores.
6. Markov chain: Used to model the transition probability of emotional states.

$$P(Xt=x \mid Xt-1=y)$$

Where Xt and Xt−1 are the states of the time t and t−1, and P is the state transition probability.

CH3. Experiments and Results

3.1 Datasets and Inputs

The datasets used include:

- It is an open source data set from Kaggle, which contains two folders, train and validation, which are divided into seven picture folders respectively. The folders are labeled angry, disgust, fear, happy, neutral, sad, and surprise. A total of 35k images were used to train the model.
- Preprocessed text samples for sentiment analysis.

3.2 Programming Details

This section describes in detail programming implementations for facial expression recognition and sentiment analysis for natural language processing. This includes all steps from data preprocessing, model building, training, inference, to statistical analysis.

1. Environment preparation

   First, I created a virtual environment called myenv and made sure to install the necessary packages and libraries, including TensorFlow, Transformers, OpenCV, NumPy, SciPy, etc. These libraries will be used for model building, training, inference, and data processing and analysis. With the blessing of the virtual environment, the code is transitive and can be used for different devices.

2. Data set preparation

   In the dataset preparation stage, we used the dataset from Kaggle to train the facial expression recognition model. For data preprocessing and enhancement, the ImageDataGenerator tool is used. The training data and validation data are loaded respectively, and the image is normalized to improve the training effect and generalization ability of the model.

3. Construction of facial expression recognition model

   Convolutional neural network (CNN) was used to construct a facial expression recognition model. The model architecture consists of multiple convolution layers, pooling layers, flattening layers and fully connected layers. Each convolutional layer is followed by a maximum pooling layer for feature extraction and dimensionality reduction. Finally, the features are further processed through the fully connected layer and Dropout layer, and the probability of seven emotion classes is output using the softmax layer.

4. Train the model

   The model is trained using training data and validated on validation data. During the training process, the model parameters were optimized through 50 rounds of iteration (epochs) to minimize the categorical cross-entropy and improve the classification accuracy. The training process takes place at a rate of approximately 1 minute per epoch.

5. Save and load the model

   After the training is completed, the structure, weights, and training records of the model are saved in the files custom_model.json, custom_model.weights.h5, and training_history.pkl respectively for subsequent use, deployment, and visualization of training tracks. Saved model files can be reloaded when needed and used for new data predictions and reasoning. Therefore, the part of the training model is marked in the code, because it can be directly used through the saved model, saving time and space.

6. Visual model training

   The accuracy and loss functions are used to visualize the training process, which can be used to observe the fit degree of the model for the training set.

   Accuracy functions:

$$Mean\ Accuracy = \frac{1}{N}\sum_{i=1}^{N} 1(\hat{y}_i = y_i)$$

   Which returns 1 if $\hat{y}_i$ equal to y_i, otherwise returns 0.

   Loss functions:

$$L = \frac{1}{N}\sum_{i=1}^{N} l(y_i, \hat{y}_i)$$

$$l(y_i, \hat{y}_i) = -\sum_{c=1}^{C} y_c \log(\hat{y}_c)$$

Where N is the batch size, C is the number of classes, y_i is thisue label of the ith sample, $\hat{y}_i$ is Predicted label of the ith sample, y_c is Indicator function for the true class c, and $\hat{y}_c$ is Predicted probability for class c

7. Real-time facial emotion recognition

   OpenCV captures a video stream from a camera and uses a trained model for real-time emotion recognition. The specific process includes:

a. Read the video frame from the camera and convert it to a grayscale image.

b. Detect facial areas using a pre-trained Haar cascade classifier.

c. Adjust the detected face image to the model input size for normalization processing.

d. Use the loaded CNN model to predict the emotion of the face image, and output the emotion category and confidence.

e. Draw detection boxes and emotional labels on the video frame and display the video in real-time.

8. Natural language processing emotion analysis

Load the pre-trained sentiment analysis model using Hugging Face's Transformers library. The specific process includes:

a. Load a pre-trained Transformer model (such as BERT or GPT) and the corresponding word divider.

b. Input text data is preprocessed by word segmentation to generate an input format that can be processed by the model.

c. Use the Transformer model for emotion analysis of input text and output emotion category and confidence.

9. Data analysis and statistical test

In the process of analysis, statistical methods such as principal component analysis (PCA), Z-test, T-test, and Chi-square test were used to analyze the emotion recognition results. Specific steps include:

a. Use PCA to reduce the dimensionality of multidimensional data and extract main features to facilitate visualization and understanding of data structure.

$$\max(u=\pm 1) \sum_{i=1}^{n} ¿¿¿$$

Where u is the projection direction, xi is the data point, and $\overline{x}$ is the data mean.

b. Z-test was used to compare the mean difference between facial emotion recognition results and text emotion analysis results to evaluate statistical significance.

$$Z=\frac{\overline{X}-\mu 0}{\frac{\sigma}{\sqrt{N}}}$$

Where $\overline{X}$ is the sample mean, average $\mu 0$ is theory, σ is the overall standard deviation, and N is the sample size.

c. T-test was used to compare the mean difference of two independent samples to further verify the validity of emotion recognition results.

$$t=\frac{\overline{X1}-\overline{X2}}{\sqrt{\left(\frac{s1^2}{n1}+\frac{s2^2}{n2}\right)}}$$

Where $\overline{X1}$ is the i th sample mean, si^2 is the i th sample variance, and ni the i th sample size.

d. Chi-square test was used to analyze the independence of categorical variables and evaluate the correlation between emotion recognition results and actual emotion labels.

$$x^2=\sum_{i=1}^{k}\frac{(Oi-Ei)^2}{Ei}$$

Where Oi is the observed frequency, Ei is the expected frequency, and k is the number of categories.

e. Calculate entropy to quantify the distribution uncertainty of facial emotion and text emotion.

$$H(X)=-\sum_{i=1}^{n}p(xi)logp(xi)$$

10. Loop design and exit

The program designs a loop to continuously read and process video frames to enhance context. And listens for user keyboard input in the loop, pressing the 'q' key can exit the loop, thus stopping the video capture and closing all Windows. The implementation demonstrates the development process of a multimodal emotion recognition system and its potential in practical applications.

11. Contextual dynamic sentiment analysis

After exiting the loop, in order to further analyze the dynamic change of emotion state and joint probability distribution, the joint state probability matrix and joint vectorized observation state probability are calculated using the Markov model. Specific steps include:

a. Count the occurrence of each emotion state, and calculate the frequency of its joint state.

b. Construct a state transition matrix to calculate the probability of transferring from one emotional state to another.

c. According to the statistical results, construct the joint vectorized observation state probability to analyze the joint distribution of emotion states.

3.3 Experimental Results

The results are visualized using tables and charts to present:

3.3.1 Model accuracy and loss

The graph on the left shows the training accuracy and validation accuracy over the epochs:

a. Training Accuracy: The training accuracy rapidly increases and reaches approximately 0.98 by the end of the training process. This indicates that the model is learning the training data very well.
b. Validation Accuracy: The validation accuracy increases initially and then plateaus around 0.83 to 0.85. This suggests that the model is performing reasonably well on the validation data, although there is still a gap between training and validation accuracy.

The graph on the right shows the training loss and validation loss over the epochs:

a. Training Loss: The training loss steadily decreases and reaches a very low value close to zero by the end of the training process. This indicates that the model is effectively minimizing the error on the training data.
b. Validation Loss: The validation loss decreases initially but then stabilizes and starts to increase slightly after about 5 epochs. This suggests that the model starts to overfit the training data after the initial epochs.

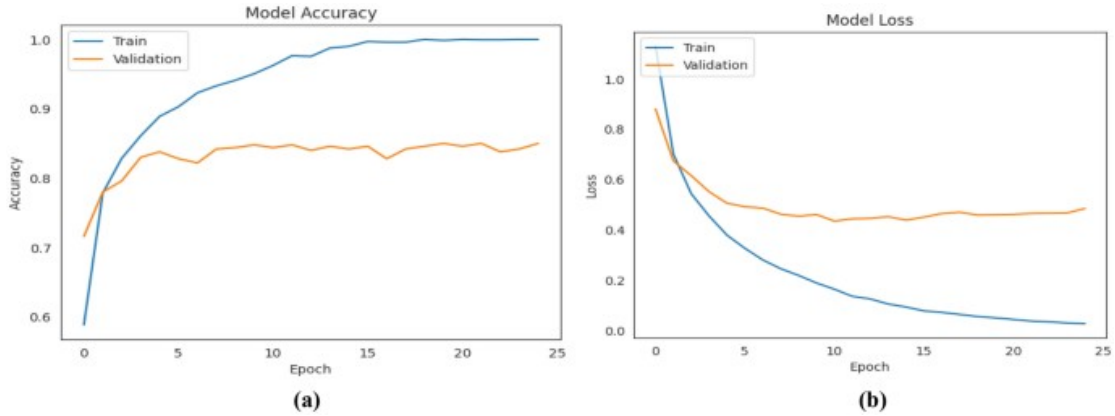

Figure 2 Model Accuracy and Loss

3.3.2 General Emotion Statistical Analysis

a. PCA results: Displays the first principal component value.

b. Z-test statistic: There was a significant difference between facial and literal emotion scores ($p < 0.05$).

c. T-test statistics: Also showed significant differences between groups.

d. Chi-square test: There was independence between the emotional categories of face and words ($p > 0.05$).

e. Entropy: represents the uncertainty of the emotional distribution of the face and text, with similar entropy values.

f. Markov joint state probability matrix: Displays the probability of transitions between combined faces and text emotion states.

g. Joint vectorized observed state probabilities: Provides probabilities of joint observed states, highlighting the most common combinations of emotions.

$$P(X,Y)= \text{Count}(X,Y)/\text{Total Counts}$$

Where X and Y are the observed states, and P is the joint probability.

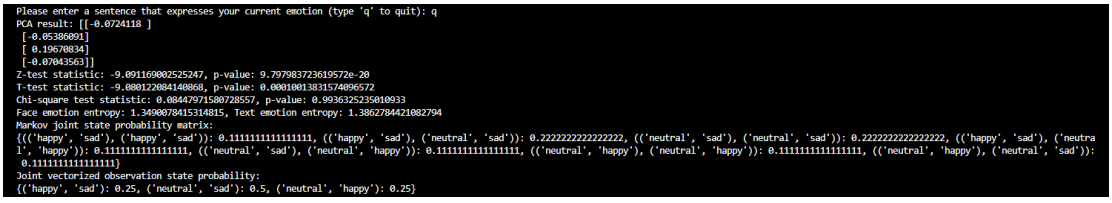

Figure 3 Results of facial emotion and text emotion analysis

3.3.2 PCA Result Analysis

The PCA transformation shows that most data points are clustered together, with one outlier. The variance captured by the first principal component is significant, indicating that PCA was able to reduce dimensionality while preserving the main data characteristics.

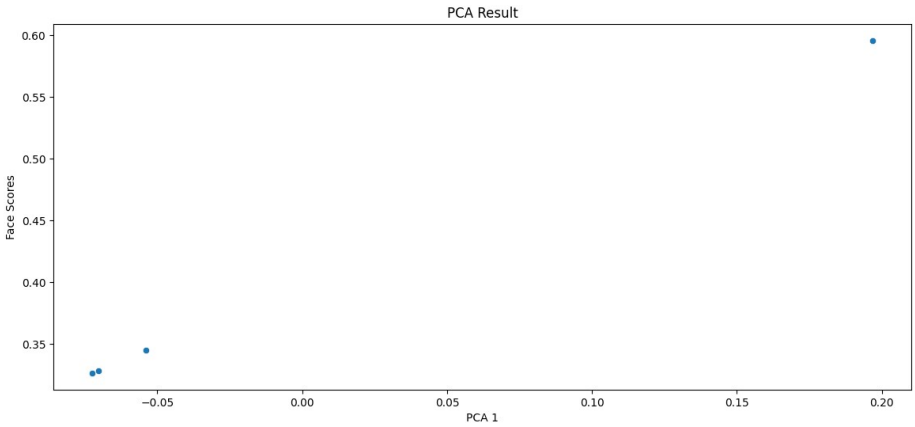

Figure 4 PCA results visualization

3.3.2 Joint State Probability Matrix Result Analysis

This heatmap visualizes the joint state probability matrix for face and text emotions:
The matrix shows the probability of transitions between different emotional states.
For example, the transition from 'happy' face emotion to 'sad' text emotion has a probability of 0.50, indicating that these states frequently co-occur. The highest probabilities indicate the most common joint states observed in the data.

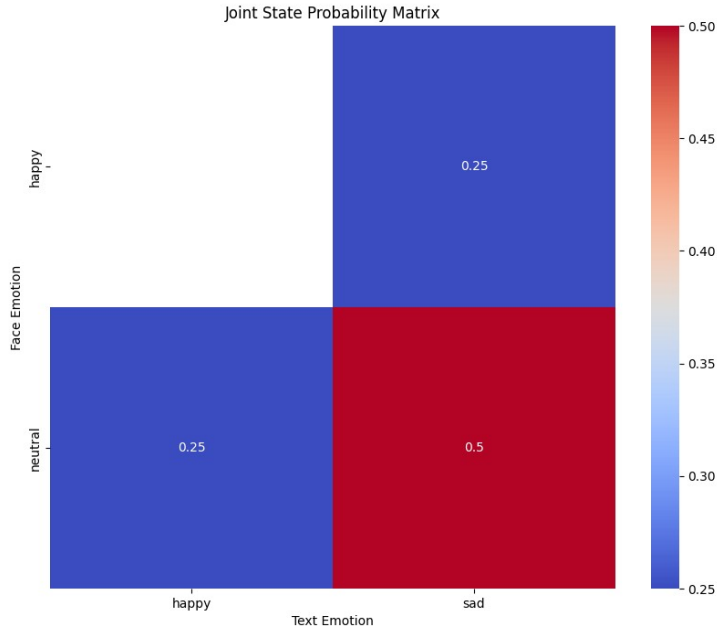

Figure 5 Joint State Probability Matrix visualization

CH4. Evaluation and Discussion

4.1 Comparison of Results

Feature/Aspect	My Model	**Ajeet Singh's Model**(Ajeet, 2019)	**Joseph Morell's Model**(Joseph, 2024)
Model Architecture	CNN with multiple convolutional layers, dropout, and fully connected layers	CNN with multiple convolutional layers, batch normalization, pooling layers, and fully connected layers	Deep CNN architecture with several convolutional layers and dense layers
Training Accuracy	~98%	~70%	~85%
Validation Accuracy	83% - 85%	~66%	~80%
Training Loss	Steadily decreases, very low at the end	Consistently decreases	Consistently decreases
Validation Loss	Stabilizes and slightly increases, indicating overfitting	Consistently decreases	Consistently decreases
Data Augmentation	Not heavily used	Extensively used	Extensively used

Feature/Aspect	My Model	**Ajeet Singh's Model**(Ajeet, 2019)	**Joseph Morell's Model**(Joseph, 2024)
Regularization Techniques	Dropout layers	Batch normalization and dropout	Dropout and batch normalization
Statistical Analysis	Includes PCA, Z-test, T-test, Chi-square test	Not included	Not included
Real-time Capability	Yes, with real-time data input and processing	Not specified	Not specified
Additional Features	Correlation analysis between face and text emoticons.	Focus on achieving balanced performance	Advanced techniques for robust feature extraction

4.2 Advantages and Disadvantages

4.2.1 Advantages

- High training accuracy: The model achieves a high training accuracy (~98%), indicating that it learned the training data well.
- Comprehensive analysis: Your project includes detailed statistical analysis such as PCA, z-tests, t-tests, Chi-square tests, etc., to gain a deeper understanding of the model's performance.
- Real-time capability: The model efficiently handles real-time data entry and processing, which is useful for applications that require immediate feedback.
- Robust assessment: Includes a variety of statistical tests as well as an assessment of the combined probability of facial and textual emotion, increasing the robustness of the model evaluation.
- Use Principal component analysis (PCA): This helps reduce the dimensionality of the data and visualizes the variance captured by the principal component.
- A combination of facial analysis and text analysis: a more multidimensional analysis of human emotions, which can better capture accurate emotions when facing subjects who want to hide their emotions.
- Dynamic photography capture: not the use of pictures, through the camera to capture the object in real-time, more suitable for real situations.
- Camera frame capture: Error is reduced by taking an average of 10 frames during camera capture as the result of facial expression.

4.2.2 Disadvantages

- Overfitting: Although the training accuracy is high, the validation accuracy plateaus around 83% to 85%, and the validation loss increases slightly after some epochs, indicating overfitting.

- Limited data enhancement: My project did not make extensive use of data enhancement techniques, which would have helped improve the generalization of the model.
- Poor correlation between patterns: The scatter plot shows a weak correlation between facial and text emotion scores, suggesting that the integration of the two patterns could be improved.
- Lack of advanced architecture: When your model uses the standard CNN architecture, trying a more advanced architecture, such as ResNet, EfficientNet, or VGG, may yield better performance.
- Need for cross-validation: Implementing k-fold cross-validation can provide a more reliable estimate of a model's performance and ensure that it generalizes well to previously unseen data.
- Capturing face time: The first time the camera takes a little longer to analyze the face, which is easy to make people tired.

4.3 Summary

4.3.1 Summary of this work

This work explores the integration of facial expression recognition and natural language processing (NLP) for sentiment analysis to develop a comprehensive emotion recognition system. By employing Convolutional Neural Networks (CNN) for facial expression recognition and Transformer models for text sentiment analysis, the project aims to enhance the accuracy and robustness of emotion detection. The results demonstrate the effectiveness of the multimodal approach in capturing and analyzing emotions from both visual and textual data. The extensive statistical analysis further validates the model's performance and highlights its potential for real-time applications.

4.3.2 Contributions of this work

Multimodal Emotion Recognition: The integration of facial expression and text sentiment analysis provides a more comprehensive understanding of emotions, addressing the limitations of single-modal approaches.

Statistical Validation: Detailed statistical analysis, including PCA, Z-tests, T-tests, and Chi-square tests, ensures the robustness and reliability of the emotion recognition model.

Real-time Capability: The developed system demonstrates real-time emotion recognition, making it suitable for practical applications requiring immediate feedback.

Enhanced Human-Computer Interaction: By understanding and responding to human emotions more effectively, the system can significantly improve user experiences in various domains such as healthcare, education, and entertainment.

4.3.3 Future work

Model Optimization: Exploring advanced architectures like ResNet, EfficientNet, or VGG to improve model accuracy and reduce overfitting.

Data Augmentation: Implementing extensive data augmentation techniques to enhance the generalization of the model.

Cross-validation: Incorporating k-fold cross-validation to provide a more reliable estimate of the model's performance.

Integration of Additional Modalities: Combining other data sources such as voice and physiological signals to capture a broader range of emotional cues.

Longitudinal Studies: Conducting studies over extended periods to analyze the consistency and evolution of emotional states.

4.3.4 Future Application

Medical science: It can be used for specific psychological disorders, such as smiling depression(Sobin & Sackeim, 1997). Because people with this disease, always subconsciously hide their negative emotions. But when humans face the outside world, it is difficult to control every aspect of their instinctive feedback at the same time. Efficient and sensitive machines will pick up this signal more accurately than the human eye, helping to identify potential patients in time to provide treatment.

Lie detector: Through the two-way dynamic monitoring of text and face, it can help catch emotional mutations and instability, which can help detect whether the object is likely to lie(Ekman, 2009).

CH5. References

Ajeet, S. (2019). *Facial Expression Detection from Scratch with CNNs*. https://kaggle.com/code/ajeetsingh123/facial-expression-detection-from-scratch-with-cnns

Chatterjee, D., Mukherjee, A., Mukhopadhyay, S., Panday, M., Panigrahi, P., & Goswami, S. (2021). *A Survey on Sentiment Analysis* (pp. 259–271). https://doi.org/10.1007/978-981-33-4367-2_26

Choudhury, P., Wang, D., Carlson, N. A., & Khanna, T. (n.d.). *Machine learning approaches*

to facial and text analysis: Discovering CEO oral communication styles—Choudhury—2019—Strategic Management Journal—Wiley Online Library. Retrieved 30 May 2024, from https://onlinelibrary.wiley.com/doi/abs/10.1002/smj.3067

Ekman, P. (2009). Lie Catching and Microexpressions. In *The Philosophy of Deception* (pp. 118–136). https://doi.org/10.1093/acprof:oso/9780195327939.003.0008

G, V., & Chandrasekaran, D. (2012). Sentiment Analysis and Opinion Mining: A Survey. *Int J Adv Res Comput Sci Technol*, *2*.

Goodfellow, I., Bengio, Y., & Courville, A. (n.d.). *Deep Learning*. MIT Press. Retrieved 30 May 2024, from https://mitpress.mit.edu/9780262035613/deep-learning/

Joseph, M. (2024). *Emotion Recognition Data*. https://kaggle.com/code/josephmorell/emotion-recognition-data

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lijun Yin, Xiaozhou Wei, Yi Sun, Jun Wang, & Rosato, M. J. (2006). A 3D Facial Expression Database For Facial Behavior Research. *7th International Conference on Automatic Face and Gesture Recognition (FGR06)*, 211–216. https://doi.org/10.1109/FGR.2006.6

Niedenthal, P., Krauth-Gruber, S., & Ric, F. (2006). *Psychology of emotion: Interpersonal, experiential, and cognitive approaches.*

Pak, A., & Paroubek, P. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, & D. Tapias (Eds.), *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2010/pdf/385_Paper.pdf

Pantic, M., & Rothkrantz, L. (2001). Automatic Analysis of Facial Expressions: The State of the Art. *Pattern Analysis and Machine Intelligence, IEEE Transactions On*, *22*, 1424–1445. https://doi.org/10.1109/34.895976

Picard, R. W. (2000, July 24). *Affective Computing*. MIT Press. https://mitpress.mit.edu/9780262661157/affective-computing/

Sobin, C., & Sackeim, H. A. (1997). *Psychomotor symptoms of depression—PubMed*. https://pubmed.ncbi.nlm.nih.gov/8988952/

Whitehill, J., Serpell, Z., Lin, Y.-C., Foster, A., & Movellan, J. R. (2014). The Faces of Engagement: Automatic Recognition of Student Engagementfrom Facial Expressions. *IEEE Transactions on Affective Computing*, *5*(1), 86–98. https://doi.org/10.1109/TAFFC.2014.2316163

Yannakakis, G., & Hallam, J. (2011). *Ranking vs. Preference: A Comparative Study of Self-reporting*. *6974*, 437–446. https://doi.org/10.1007/978-3-642-24600-5_47

CH6. Appendix

6.1 Code

```python
import os
import numpy as np
import cv2
import threading
from collections import defaultdict
from tensorflow.keras.models import model_from_json
from tensorflow.keras.preprocessing.image import img_to_array, ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, MaxPooling2D, Dropout
from sklearn.decomposition import PCA
from scipy.stats import ttest_ind, chi2_contingency, entropy
from statsmodels.stats.weightstats import ztest
from transformers import pipeline
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import pickle

# Global variable
stop_flag = threading.Event()

# Load pre-trained sentiment analysis model
emotion_recognition = pipeline('sentiment-analysis')

# # train dataset
# train_dir = 'images/train/'
# validation_dir = 'images/validation/'

# # Picture generation
# train_datagen = ImageDataGenerator(rescale=1./255)
# validation_datagen = ImageDataGenerator(rescale=1./255)

# train_generator = train_datagen.flow_from_directory(
#     train_dir,
#     target_size=(48, 48),
#     batch_size=32,
#     class_mode='categorical'
# )

# validation_generator = validation_datagen.flow_from_directory(
#     validation_dir,
#     target_size=(48, 48),
#     batch_size=32,
#     class_mode='categorical'
# )

# # build model
# model = Sequential([
#     Conv2D(32, (3, 3), activation='relu', input_shape=(48, 48, 3)),
#     MaxPooling2D((2, 2)),
#     Conv2D(64, (3, 3), activation='relu'),
#     MaxPooling2D((2, 2)),
#     Conv2D(128, (3, 3), activation='relu'),
#     MaxPooling2D((2, 2)),
#     Flatten(),
#     Dense(128, activation='relu'),
#     Dropout(0.5),
#     Dense(7, activation='softmax')
# ])

# model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# # train model
# history = model.fit(
#     train_generator,
#     epochs=50,
#     validation_data=validation_generator
# )

# # 保存训练历史记录
# with open('training_history.pkl', 'wb') as file:
#     pickle.dump(history.history, file)

# # save model structure
# model_json = model.to_json()
# with open("custom_model.json", "w") as json_file:
#     json_file.write(model_json)
# # save model weight
# model.save_weights("custom_model.weights.h5")

# # Function to plot training history
# def plot_training_history(history):
#     plt.figure(figsize=(14, 6))

#     # Plot training & validation accuracy values
#     plt.subplot(1, 2, 1)
#     plt.plot(history['accuracy'], label='Training Accuracy')
#     plt.plot(history['val_accuracy'], label='Validation Accuracy')
#     plt.title('Model Accuracy')
#     plt.xlabel('Epoch')
#     plt.ylabel('Accuracy')
#     plt.legend(loc='lower right')

#     # Plot training & validation loss values
#     plt.subplot(1, 2, 2)
#     plt.plot(history['loss'], label='Training Loss')
#     plt.plot(history['val_loss'], label='Validation Loss')
#     plt.title('Model Loss')
#     plt.xlabel('Epoch')
#     plt.ylabel('Loss')
#     plt.legend(loc='upper right')

#     plt.show()


# # Load training history from file
# with open('training_history.pkl', 'rb') as file:
#     history = pickle.load(file)

# # Plot training history
# plot_training_history(history)


# Load custom model
def load_custom_model():
    json_file = open('custom_model.json', 'r')
    loaded_model_json = json_file.read()
    json_file.close()
    loaded_model = model_from_json(loaded_model_json)
    loaded_model.load_weights("custom_model.weights.h5")
    return loaded_model

# Function for facial emotion recognition
def detect_emotion_from_face(frame, custom_model):
    try:
        if frame is None or frame.size == 0:
            raise ValueError("Invalid input image")

        face_img = cv2.resize(frame, (48, 48))
        face_img = face_img.astype("float") / 255.0
        face_img = img_to_array(face_img)
        face_img = np.expand_dims(face_img, axis=0)

        predictions = custom_model.predict(face_img)
        max_index = np.argmax(predictions[0])
        emotions = ['angry', 'disgust', 'fear', 'happy', 'neutral', 'sad', 'surprise']
        emotion = emotions[max_index]
        confidence = predictions[0][max_index]
        return emotion, confidence
    except Exception as e:
        print(f"Error in detecting emotion: {e}")
        return None, 0

# Emotion label mapping
def map_nlp_emotion(label):
    mapping = {
        'POSITIVE': 'happy',
```

```
        'NEGATIVE': 'sad',
        'NEUTRAL': 'neutral'
    }
    return mapping.get(label.upper(), 'neutral')

# Initialize emotion state statistics
state_counts = defaultdict(int)
joint_state_counts = defaultdict(int)
possible_states = ['happy', 'sad', 'neutral', 'angry', 'surprise', 'fear', 'disgust']
face_scores = []
text_scores = []

def analyze_emotions(frame_count=10):
    frame_emotions = []
    for _ in range(frame_count):
        if stop_flag.is_set():
            break
        ret, frame = cap.read()
        if not ret:
            break

        # Face detection
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
        faces = face_cascade.detectMultiScale(gray, 1.3, 5)

        if len(faces) == 0:
            print("No face detected")
            continue

        (x, y, w, h) = faces[0]
        face = frame[y:y+h, x:x+w]

        # Detect face emotion
        face_emotion, face_confidence = detect_emotion_from_face(face, custom_model)
        if face_emotion:
            frame_emotions.append((face_emotion, face_confidence))

        # Display video
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)
        cv2.imshow('Video', frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            stop_flag.set()
            break

    if frame_emotions:
        avg_emotion_confidence = defaultdict(float)
        for emotion, confidence in frame_emotions:
            avg_emotion_confidence[emotion] += confidence / len(frame_emotions)

        dominant_emotion = max(avg_emotion_confidence, key=avg_emotion_confidence.get)
        confidence = avg_emotion_confidence[dominant_emotion]

        print(f"Average face emotion: {dominant_emotion} ({confidence*100:.2f}%)")

        return dominant_emotion, confidence
    return None, 0

def main_loop():
    while not stop_flag.is_set():
        user_input = input("Please enter a sentence that expresses your current emotion (type 'q' to quit): ")
        if user_input.lower() == 'q':
            stop_flag.set()
            break

        # Analyze sentiment of user input
        text_emotion = emotion_recognition(user_input)[0]
        mapped_text_emotion = map_nlp_emotion(text_emotion['label'])

        # Detect face emotion
        face_emotion, face_confidence = analyze_emotions()

        # Compare face and text emotions
        print(f"Face emotion: {face_emotion} ({face_confidence*100:.2f}%)")
        print(f"Text emotion: {mapped_text_emotion} ({text_emotion['score']*100:.2f}%)")

        if face_emotion in possible_states and mapped_text_emotion in possible_states:
            state_counts[(face_emotion, mapped_text_emotion)] += 1
            for prev_state in state_counts:
                joint_state_counts[(prev_state, (face_emotion, mapped_text_emotion))] += 1

        if face_confidence > 0:
            face_scores.append(face_confidence)
        if text_emotion['score'] > 0:
            text_scores.append(text_emotion['score'])

    if face_scores and text_scores:
        pca = PCA(n_components=1)
        pca_result = pca.fit_transform(np.array([face_scores, text_scores]).T)
        print(f"PCA result: {pca_result}")

        z_stat, p_val = ztest(face_scores, value=np.mean(text_scores))
        print(f"Z-test statistic: {z_stat}, p-value: {p_val}")

        t_stat, t_p_val = ttest_ind(face_scores, text_scores)
```

```
        print(f"T-test statistic: {t_stat}, p-value: {t_p_val}")

        if all(x > 0 for x in face_scores + text_scores):
            chi2_stat, chi2_p_val, _, _ = chi2_contingency([face_scores, text_scores])
            print(f"Chi-square test statistic: {chi2_stat}, p-value: {chi2_p_val}")
        else:
            print("Chi-square test cannot be performed: frequency values contain zero")

        face_entropy = entropy(face_scores)
        text_entropy = entropy(text_scores)
        print(f"Face emotion entropy: {face_entropy}, Text emotion entropy: {text_entropy}")

        total_transitions = sum(joint_state_counts.values())
        markov_probabilities = {k: v / total_transitions for k, v in joint_state_counts.items()}
        print("Markov joint state probability matrix:")
        print(markov_probabilities)

        total_counts = sum(state_counts.values())
        joint_vector_probabilities = {k: v / total_counts for k, v in state_counts.items()}
        print("Joint vectorized observation state probability:")
        print(joint_vector_probabilities)

        # Visualize results
        visualize_results(pca_result, face_scores, text_scores, markov_probabilities, joint_vector_probabilities)

def visualize_results(pca_result, face_scores, text_scores, markov_probabilities, joint_vector_probabilities):
    # Convert joint probabilities to DataFrame for heatmap
    joint_df = pd.DataFrame(list(joint_vector_probabilities.items()), columns=['States', 'Probability'])
    joint_df[['Face Emotion', 'Text Emotion']] = pd.DataFrame(joint_df['States'].tolist(), index=joint_df.index)
    joint_pivot = joint_df.pivot(index='Face Emotion', columns='Text Emotion', values='Probability')

    # Plotting face and text emotion scores
    results_df = pd.DataFrame({
        'Face Scores': face_scores,
        'Text Scores': text_scores,
        'PCA1': pca_result[:, 0]
    })

    plt.figure(figsize=(14, 6))
    sns.scatterplot(data=results_df, x='Face Scores', y='Text Scores')
    plt.title('Face Emotion vs Text Emotion Distribution')
    plt.xlabel('Face Emotion Scores')
    plt.ylabel('Text Emotion Scores')
    plt.show()

    plt.figure(figsize=(14, 6))
    sns.scatterplot(data=results_df, x='PCA1', y='Face Scores')
    plt.title('PCA Result')
```

```
    plt.xlabel('PCA 1')
    plt.ylabel('Face Scores')
    plt.show()

    plt.figure(figsize=(10, 8))
    sns.heatmap(joint_pivot, annot=True, cmap='coolwarm', cbar=True)
    plt.title('Joint State Probability Matrix')
    plt.show()

# Get video input
cap = cv2.VideoCapture(0)

# Lower camera resolution
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

# Load custom model
custom_model = load_custom_model()

# Start emotion analysis thread
emotion_thread = threading.Thread(target=main_loop)
emotion_thread.start()

# Main thread for displaying video
while cap.isOpened():
    ret, frame = cap.read()
    if not ret or stop_flag.is_set():
        break
    cv2.imshow('Video', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        stop_flag.set()
        break

cap.release()
cv2.destroyAllWindows()
emotion_thread.join()
```

6.2 GitHub

https://github.com/willllooo/Comprehensive-Application-of-Facial-Expression-Recognition-and-Text-Sentiment-Analysis