

# Real-time Emotional Information Extraction System: EmoNeXt Model with Multi-Module Fusion Strategy

12110208, 12112124, 12111825, 12111628, 12111216

June 16, 2024

## Abstract

In the digital era, acquiring and analyzing emotional information has become increasingly important. Traditional facial expression recognition methods face challenges in efficiently and accurately extracting emotional information from large volumes of images and videos. This study aims to design and implement an efficient and accurate deep learning-based facial expression recognition system. The system primarily uses the FER2013 dataset, along with self-collected facial expression data through web crawling, for data collection and preprocessing. We have designed a model named EmoNeXt, which integrates Spatial Transformer Networks (STN), ConvNeXt, and Squeeze-and-Excitation (SE) modules to enhance recognition accuracy and processing efficiency.

Experimental results show that the EmoNeXt model performs well in facial expression recognition tasks, capable of real-time emotional information extraction from images and videos. Additionally, we explore future directions for optimizing system performance, including improving the STN module, increasing the variety of expressions, and using Ray for parameter tuning. Our research provides new insights and methods for facial expression recognition systems in applications such as emotional analysis and human-computer interaction.

**Keywords:** Facial Expression Recognition, Deep Learning, EmoNeXt Model, Emotion Analysis, Real-time Processing.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Job Introduction . . . . .	3
1.2	Dataset Introduction . . . . .	3
1.2.1	Dataset Overview . . . . .	3
1.2.2	FER2013 Dataset . . . . .	4
1.2.3	Additional Data Sources . . . . .	5
1.2.4	Data Preprocessing . . . . .	6
1.2.5	Summary . . . . .	6
<b>2</b>	<b>Existing Works</b>	<b>6</b>
2.1	Reference Model: EmoNeXt . . . . .	6
2.1.1	Spatial Transformer Networks (STN) . . . . .	7
2.1.2	ConvNeXt . . . . .	7
2.1.3	Squeeze-and-Excitation (SE) Module . . . . .	8
<b>3</b>	<b>Experiments</b>	<b>8</b>
3.1	Model Training . . . . .	8
3.1.1	Model Structure . . . . .	8
3.1.2	Loss Function . . . . .	9
3.1.3	Training Setup . . . . .	9
3.1.4	Training Process . . . . .	10
3.2	Model Evaluation . . . . .	10
<b>4</b>	<b>Presentations &amp; Future Works</b>	<b>12</b>
4.1	Presentations . . . . .	12
4.1.1	Image Facial Expression . . . . .	12
4.1.2	Video Facial Expression . . . . .	12
4.1.3	Real-time facial expression . . . . .	13
4.2	Future Works . . . . .	13
4.2.1	Enhancing Video Description through Emotion Analysis . . . . .	13
4.2.2	Optimizing Neural Network Performance . . . . .	14
4.2.3	Enhancing Emotion Classification . . . . .	14
4.2.4	Tuning Model Parameters . . . . .	15
<b>5</b>	<b>GitHub Link</b>	<b>15</b>

# 1 Introduction

## 1.1 Job Introduction

In the current digital era, the acquisition and analysis of emotional information are becoming increasingly important. However, traditional facial expression recognition methods face challenges in rapidly and accurately extracting useful emotional information from large volumes of images and videos. Therefore, this project aims to design and implement an efficient and accurate facial expression recognition system to meet the needs of modern application scenarios.

- **Improving Recognition Accuracy:** We will design and train an advanced deep learning model named EmoNeXt, which combines Spatial Transformer Networks (STN), ConvNeXt convolutional neural network, and Squeeze-and-Excitation (SE) modules. This combination will significantly enhance the accuracy of facial expression recognition.
- **Enhancing Real-Time Processing Capability:** The system will ensure efficient operation in real-time application scenarios, capable of quickly processing data from various sources to meet the response speed requirements of practical applications, thus achieving efficient emotional information extraction.
- **Adaptability to Diverse Data:** We will design the system to handle diverse facial expression data, including images and videos with different resolutions, angles, and lighting conditions, thereby improving the system's versatility and practicality.
- **User-Friendliness:** We will develop a user-friendly Graphical User Interface(GUI) (Figure 1) that allows users to easily upload images and videos to complete facial expression recognition tasks. This will lower the usage threshold and enhance the system's usability and practicality.



Figure 1: user-friendly GUI illustration

Through these improvements, the system will not only enhance the accuracy and real-time processing capability of facial expression recognition but also adapt to various data types and provide a good user experience.

## 1.2 Dataset Introduction

### 1.2.1 Dataset Overview

To achieve efficient and accurate facial expression recognition, this project utilizes the FER2013 (Facial Expression Recognition 2013) dataset as the primary dataset and extends the original dataset coverage with additional sources. The choice and processing of data directly impact the model training effectiveness and the final system performance. Therefore, scientifically selecting and processing the dataset is a critical step in this project.

### 1.2.2 FER2013 Dataset

FER2013 is a publicly available dataset for facial expression recognition, first introduced during the 2013 Kaggle Facial Expression Recognition Challenge. This dataset consists of 35,887 grayscale images, each with a resolution of 48x48 pixels. These images are sourced from various real-world scenarios on the internet, covering people of different genders, ages, and ethnicities.



Figure 2: Sample images from the FER2013 dataset

#### Dataset Characteristics:

- **Image Type:** Grayscale images with a resolution of 48x48 pixels.
- **Emotion Categories:** Each image is labeled with one of the following seven emotions:
  - **Angry:** Expressions showing anger.
  - **Disgust:** Expressions indicating displeasure or revulsion.
  - **Fear:** Expressions reflecting panic or fear.
  - **Happy:** Expressions conveying joy and happiness.
  - **Sad:** Expressions showing sadness and sorrow.
  - **Surprise:** Expressions indicating surprise.
  - **Neutral:** Neutral expressions without evident emotional features.

#### Dataset Split:

- **Training Set:** 28,709 images used for model training and preliminary tuning.
- **Validation Set:** 3,589 images used for model validation and parameter adjustment.
- **Test Set:** 3,589 images used for final performance evaluation.

Here are a selection of example data taken from FER2013 in Table 1.

Table 1: FER2013 Dataset Summary

Class	Training	Validation	Testing	Class Total
Angry	3995	467	491	4953
Disgust	436	56	55	547
Fear	4097	496	528	5121
Happy	7215	895	879	8989
Sad	4830	653	594	6077
Surprise	3171	415	416	4002
Neutral	4965	607	626	6198
<b>Total</b>	<b>28709</b>	<b>3589</b>	<b>3589</b>	<b>35887</b>

### 1.2.3 Additional Data Sources

Previously, the FER2013 dataset provided a wide range of good samples for training purposes. In addition, we gathered more pictures to expand the knowledge and generalization ability of our model. We used an open-source web scraping tool to crawl the additional image data we needed. Given that the original dataset has seven emotions as classification labels: angry, neutral, disgust, sad, happy, fear, and surprise, we used the web scraping tool to crawl images corresponding to each of these seven labels.

The open-source tool is based on `requests` and WebDriver from `selenium`. Selenium uses WebDriver, a protocol for controlling web browsers. It provides bindings for many programming languages, including Python, Java, JavaScript, and C#. Selenium sends commands to the browser and receives responses through WebDriver, enabling it to control the browser's actions. Through WebDriver we can handle more complex behaviors when interacting with web pages. Websites often feature dynamically rendered elements, particularly when searching for images. Therefore, the need to handle dynamic elements becomes crucial. We utilized an open-source web scraping tool called [Image-Downloader](#) for this purpose. During the scraping process, the tool accepts search keywords as arguments and constructs different URLs for searching based on the search engine used. Additionally, it supports features such as proxy configuration and multi-threaded crawling. Consequently, the images we obtain are simply downloaded in bulk from search engines, with their quality depending on both the quality of the keywords we input and the subsequent image processing steps.

Here are some examples of the gathered pictures:

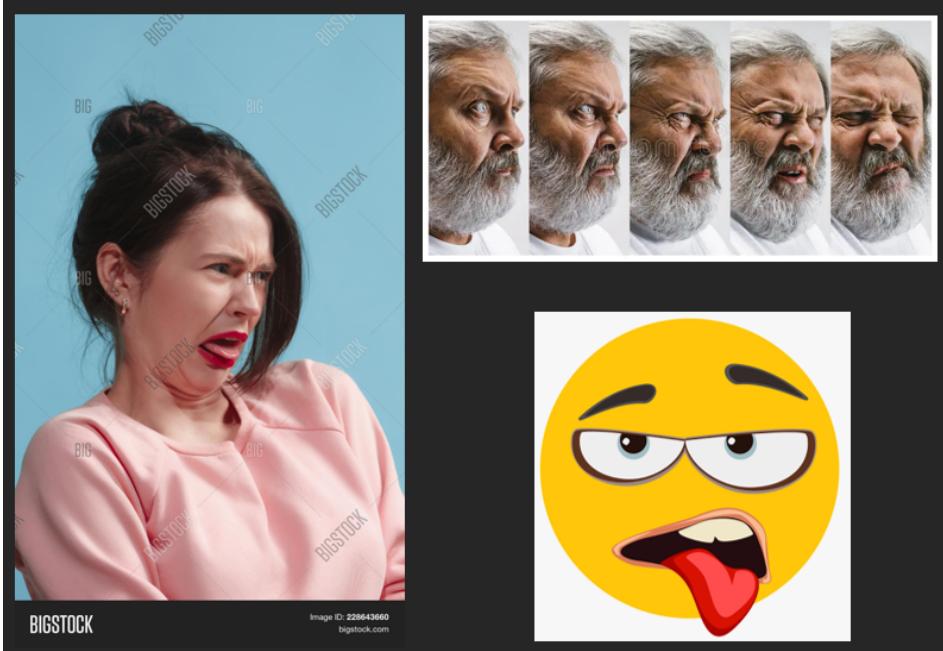


Figure 3: Sample images collected using the web scraping tool.

The keyword we use to derive pictures is `XX expression`, where `XX` refers to the name of emotions. As we can see from the above pictures, there are many flaws or features existing:

- Watermarks remain, and some of them are quite heavy.
- There exist some pictures containing multiple faces, though they share the same expression.
- The sizes of the figures are inconsistent.
- There are a lot of virtual facial expressions in the derived data.

Therefore, in subsequent work, we need to further filter the images based on criteria such as image category and size.

#### 1.2.4 Data Preprocessing

To ensure the quality and adaptability of the data, we performed the following preprocessing steps:

##### 1. Data Cleaning:

We Remove blurry, duplicate, or mislabeled images to ensure the accuracy and reliability of the dataset and correct noise and anomalies in the images, such as distortion and exposure issues.

##### 2. Image Processing:

- **Face Detection:** Using OpenCV's built-in face recognition classifier to crop the face area in the images, ensuring the data fed into the model focuses on the facial region.
- **Grayscale Conversion:** Converting color images to grayscale to unify the data format and reduce computational complexity.
- **Data Augmentation:** Extending the dataset through random cropping, flipping, rotation, and other operations to enhance the model's robustness and generalization ability.
- **Channel Conversion:** Repeating grayscale images three times to create three-channel images to meet the input requirements of convolutional neural networks.

##### 3. Data Splitting:

- **Training Set:** 80% of the images for model training.
- **Validation Set:** 10% of the images for model validation.
- **Test Set:** 10% of the images for final testing and performance evaluation.

#### 1.2.5 Summary

By scientifically selecting and processing the FER2013 dataset and its extended data, we have constructed a diverse and high-quality facial expression dataset, laying a solid foundation for the training and performance optimization of deep learning models. This data will be used to build and optimize the facial expression recognition system, improving the system's accuracy and adaptability in practical applications.

## 2 Existing Works

### 2.1 Reference Model: EmoNeXt

EmoNeXt is a deep learning model designed for facial expression recognition tasks, incorporating various advanced neural network techniques to enhance the accuracy and efficiency of emotion recognition. It integrates Spatial Transformer Networks (STN), ConvNeXt, and Squeeze-and-Excitation (SE) modules for automatic feature extraction and alignment from facial images, achieving efficient emotion recognition.<sup>[2]</sup>

#### Characteristics of EmoNeXt:

- **Modular Design:** EmoNeXt combines multiple deep learning modules, providing high flexibility and adaptability to the model.
- **Efficient Feature Extraction:** Through multi-level feature extraction mechanisms, it effectively captures emotional features from facial images.
- **Accurate Emotion Recognition:** Utilizes advanced regularization techniques and optimization methods to enhance the accuracy of recognition.

### 2.1.1 Spatial Transformer Networks (STN)

STN is a neural network module used for image transformation and spatial alignment, capable of automatically learning and applying spatial transformations to standardize input images, making downstream feature extraction more accurate. STN primarily includes three parts: Localization Network, Grid Generator, and Sampler.[1]

**Functions of STN:**

- **Spatial Alignment:** Automatically adjusts the position, size, and angle of input images, making subsequent feature extraction processes more consistent and effective.
- **Image Transformation:** Learned transformations can adapt to various complex image transformation needs, such as scaling and rotation.

**Workflow:**

- **Localization Network:** Generates transformation parameters from the input image.
- **Grid Generator:** Creates a transformed sampling grid based on the transformation parameters.
- **Sampler:** Maps the input image onto the transformed grid to produce the aligned output image.

The effect of the STN model on the samples is shown in the figure 4b. The subplots enclosed in green boxes represent the output of the STN model.

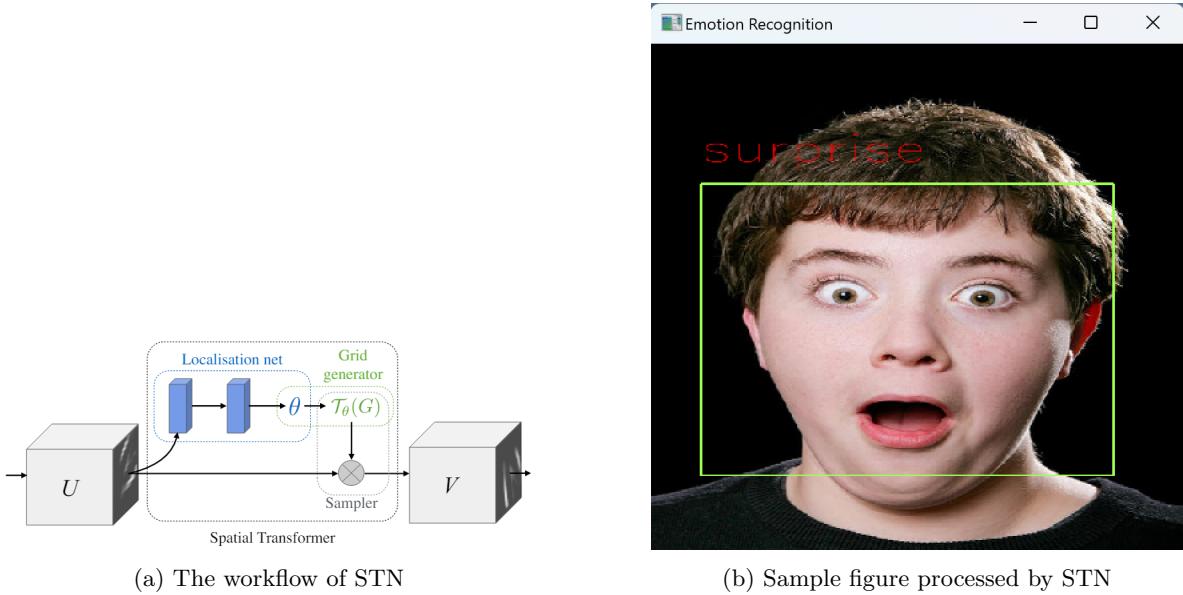


Figure 4: Combined subfigures for STN model

### 2.1.2 ConvNeXt

**ConvNeXt** is a new type of convolutional neural network (ConvNet) designed to meet the demands of deep learning in the 2020s, proposed by Zhuang Liu and colleagues. ConvNeXt builds upon traditional convolutional neural network architectures by integrating successful design concepts and optimization strategies from modern computer vision to enhance performance and efficiency. It incorporates the hierarchical design of the Swin Transformer with carefully adjusted convolution operations, such as larger convolutional kernels, global average pooling, and improved normalization layers, to achieve more efficient feature representation and better parameter utilization. Experiments have demonstrated that ConvNeXt performs excellently across multiple visual tasks, particularly in image classification, object detection, and semantic segmentation, showcasing the robust adaptability and competitiveness of convolutional networks in the new era.[5]

### 2.1.3 Squeeze-and-Excitation (SE) Module

The SE module is a technique used to enhance the feature representation of convolutional neural networks by re-calibrating channel features based on learned weights, thus improving the feature representation ability. It mainly includes Squeeze and Excitation operations.[3]

**Functions of SE Module:**

- **Feature Enhancement:** Enhances the expression of useful features and suppresses irrelevant features through weight adjustment.
- **Channel Adaptation:** Dynamically adjusts the weights of channel features based on different inputs, improving the model's representational power.

**Workflow:**

- **Squeeze Operation:** Performs global average pooling on the input feature map to generate global features for each channel.
- **Excitation Operation:** Uses fully connected layers and activation functions to generate channel weights, then applies these weights to the original feature map.

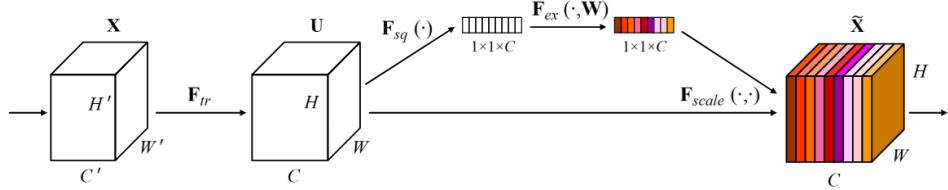


Figure 5: Introduction of the mechanism of SE module

## 3 Experiments

### 3.1 Model Training

#### 3.1.1 Model Structure

Our model structure is itemized as:

- **EmoNeXt:** Combines Spatial Transformer Networks (STN), ConvNeXt, and Squeeze-and-Excitation (SE) modules for efficient and accurate facial expression recognition.
- **STN Module:** Performs spatial alignment on input images, reducing displacements and distortions.
- **ConvNeXt Stage:** Extracts multi-scale features through multi-layer convolution operations.
- **SE Module:** Dynamically adjusts the weights of channel features to enhance the expression of useful features.

The authors of ConvNeXt have developed multiple versions, distinguished by variations in the number of channels (C) and blocks (B) within each stage. Here are the configurations:

Considering factors such as training time and accuracy, we ultimately selected the base version of the model, i.e.,  $C = (128, 256, 512, 1024)$  and  $B = (3, 3, 27, 3)$ , as our final training model. The workflow of our final training model is illustrated in Figure 6.

Model	$C$	$B$
Tiny	(96, 192, 384, 768)	(3, 3, 9, 3)
Small	(96, 192, 384, 768)	(3, 3, 27, 3)
Base	(128, 256, 512, 1024)	(3, 3, 27, 3)
Large	(192, 384, 768, 1536)	(3, 3, 27, 3)
XLarge	(256, 512, 1024, 2048)	(3, 3, 27, 3)

Table 2: Model configurations with their corresponding  $C$  and  $B$  values.

### 3.1.2 Loss Function

In our model, the final loss function combines the **Cross-Entropy Loss** with a **Self-Attention (SA) Regularization** term. This design ensures effective classification while promoting balanced attention distribution.

The **Cross-Entropy Loss**, denoted as  $\mathcal{L}_{\text{CE}}$ , is a standard loss function for multiclass classification. It measures the divergence between the true labels and the predicted probabilities. Mathematically, it is expressed as:

$$\mathcal{L}_{\text{CE}} = - \sum_{i=1}^N y_i \log(\hat{y}_i), \quad (1)$$

where  $y_i$  represents the true label,  $\hat{y}_i$  denotes the predicted probability, and  $N$  is the number of samples.

To enhance the generalization ability of the model by ensuring balanced attention weights, we introduce the **Self-Attention Regularization** term,  $\mathcal{L}_{\text{SA}}$ . This term aims to minimize the variance of the self-attention weights  $W$  relative to their mean  $\bar{W}$ , and it is defined as:

$$\mathcal{L}_{\text{SA}} = \frac{1}{N} \sum_{i=1}^N (W_i - \bar{W})^2, \quad (2)$$

where  $W_i$  represents the individual attention weight,  $\bar{W}$  is the mean attention weight, and  $N$  is the total number of weights. This regularization encourages the model to distribute attention more uniformly across the input sequence.

The **overall loss function** integrates both the classification accuracy and the regularization term. It is formulated as:

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{CE}} + \lambda \cdot \mathcal{L}_{\text{SA}}, \quad (3)$$

where  $\lambda$  is a hyperparameter that balances the contribution of the SA regularization term against the Cross-Entropy Loss. This combined approach helps the model achieve high classification performance while maintaining balanced attention weights.

### 3.1.3 Training Setup

To achieve optimal training results, we define the training setup as follows:

- **Optimizer:** Uses the Adam optimizer with settings for initial learning rate and weight decay.
- **Hyperparameters:**
  - **Learning Rate:** Initially set to 1e-4, adjusted based on performance on the validation set.
  - **Batch Size:** 64, considering memory constraints and training efficiency.
  - **Number of Epochs:** 50 epochs, including both training and validation phases in each epoch.

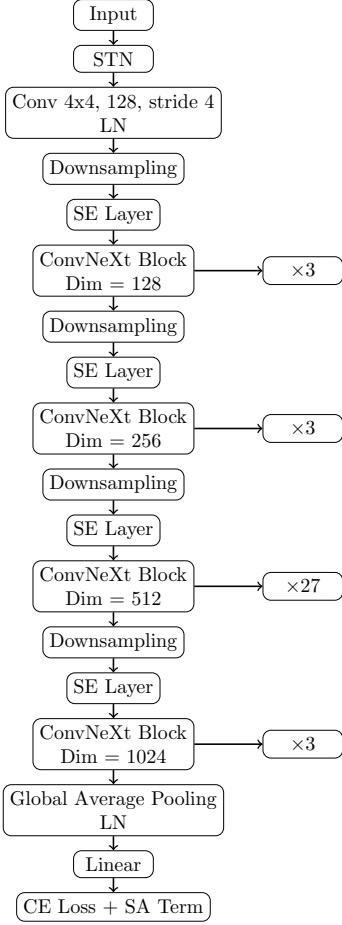


Figure 6: EmoNeXt Model Architecture with Base Configuration

### 3.1.4 Training Process

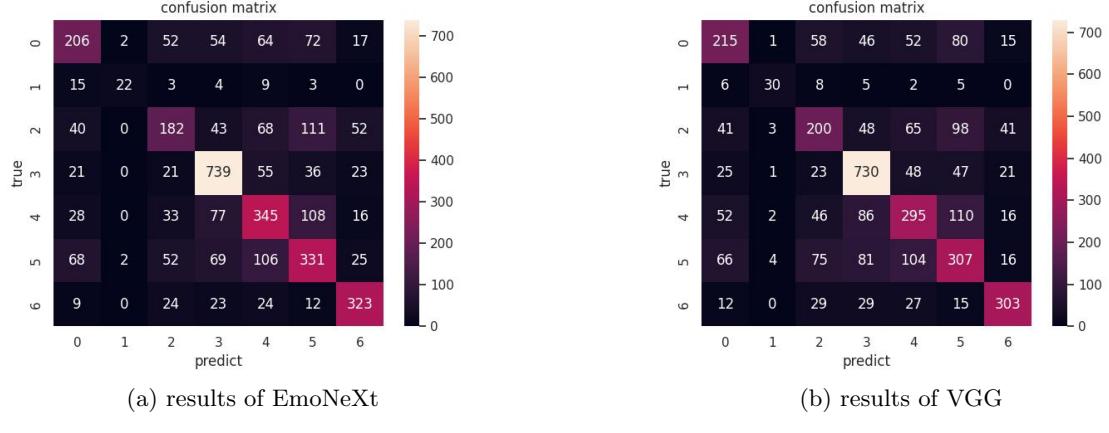
We enhance the model’s generalization capability through data augmentation techniques such as random cropping and random rotation. To prevent overfitting, we implement stochastic depth[4] and label smoothing regularization[9]. Additionally, we optimize memory usage and training speed by employing Exponential Moving Average (EMA)[8] and mixed precision training[6].

## 3.2 Model Evaluation

Here are some comparisons with our EmoNeXt model and some well-known baseline models like VGG, whose introduction is omitted considering the limited space.

First we check about the confusion matrix between EmoNeXt model and VGG model, the illustration is showed in Figure 7. These confusion matrices compare the performance of the EmoNeXt and VGG models on a facial expression recognition task. The first matrix (EmoNeXt) (Figure 7a) shows higher accuracy across most emotion categories, with particularly strong performance in recognizing emotions labeled as "3" and "4." The second matrix (VGG) (Figure 7b) demonstrates relatively better performance in recognizing emotions labeled as "1" and "2," but overall has lower classification accuracy, especially with more misclassifications in the emotion categories labeled "0" and "5." This indicates that EmoNeXt exhibits superior classification performance and lower misclassification rates compared to the VGG model.

The training logs can be illustrated in Figure 8. These four training logs capture the performance of the EmoNeXt model with different configurations throughout the training process. The first log (Figure 8a) shows that using the EmoNeXt\_tiny configuration over 88 epochs resulted in a final test accuracy of 44.97%, with a training accuracy of 40.57% and a validation accuracy of 41.74%. The second log (Figure 8b) illustrates that the EmoNeXt\_base configuration over 38 epochs achieved a



(a) results of EmoNeXt

(b) results of VGG

Figure 7: confusion matrixxs between EmoNeXt and VGG

test accuracy of 73.39%, with a training accuracy of 90.81% and a validation accuracy of 68.18%. In the third log (Figure 8c), again with the EmoNeXt\_base configuration but at 30 epochs, the test accuracy improved to 73.92%, with a training accuracy of 93.48% and a validation accuracy of 67.68%. These logs reflect the impact of different configurations and epoch counts on the EmoNeXt model's performance, with the EmoNeXt\_base configuration demonstrating a better balance between accuracy and training time, achieving higher test and validation accuracies in a shorter training period.

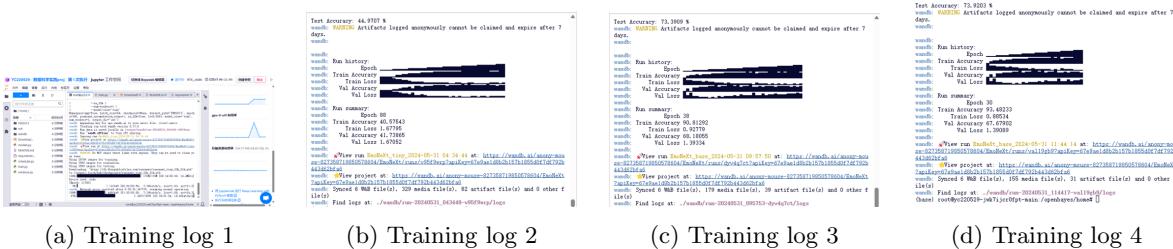


Figure 8: training logs

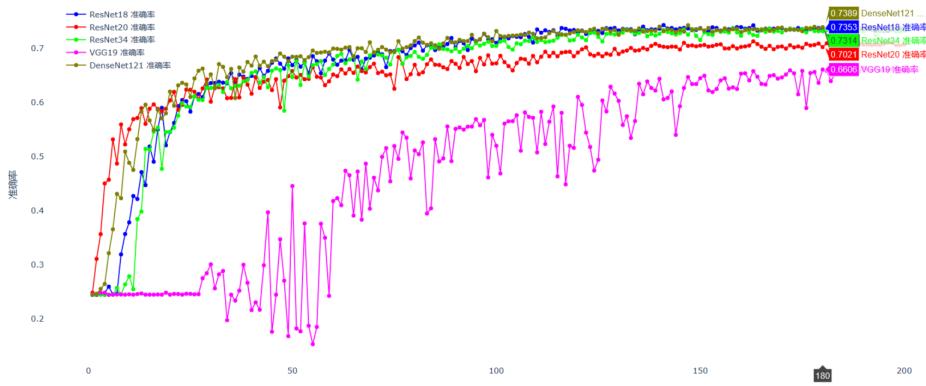


Figure 9: Comparison of model accuracy over epochs, showing that EmoNeXt achieves higher and more stable accuracy compared to other baseline models.

We compared the accuracy of EmoNeXt with other baseline models over epochs in Figure 9. The graph shows that EmoNeXt consistently achieves higher accuracy compared to ResNet18, ResNet20, ResNet34, VGG19, and DenseNet121 throughout the training process. Particularly, the ResNet18, ResNet20, ResNet34, and DenseNet121 models quickly converge to a high accuracy, stabilizing around

similar levels after initial fluctuations. In contrast, VGG19 demonstrates more significant variance and instability in accuracy, with a delayed convergence compared to the other models. This comparison highlights EmoNeXt’s superior generalization capability and training stability, making it more effective for facial expression recognition tasks.

## 4 Presentations & Future Works

In this section, we will present the practical effectiveness of our trained model for real-world applications and discuss potential areas for future improvement. We will use the checkpoint from the best-trained EmoNeXt model to import the model. Images and video files will undergo the same pre-processing operations as the test set for demonstration and analysis of the model’s performance. The demonstration will be implemented through a GUI window (with screenshots included in the report). You can find the corresponding file ”windows.py” in the submitted materials, and running this script will generate the GUI window. Please ensure that the test images or videos are placed in the same directory as the ”windows.py” file for successful execution.

### 4.1 Presentations

#### 4.1.1 Image Facial Expression

First, we will demonstrate facial expression analysis on images. By clicking the corresponding button on the GUI shown in Figure 1 and uploading an image, we obtain the result as shown in Figure 10. This image illustrates the results of facial expression analysis performed by the EmoNeXt model through a GUI window. The left subplot displays the original image with superimposed facial feature detection results. The central bar chart shows the predicted probabilities for each emotion category, indicating a clear detection of the ”happy” expression. The right window shows the bounding box and facial landmarks on the detected face. The entire interface provides an intuitive way to present the expression recognition results and the model’s confidence level in each emotion category.

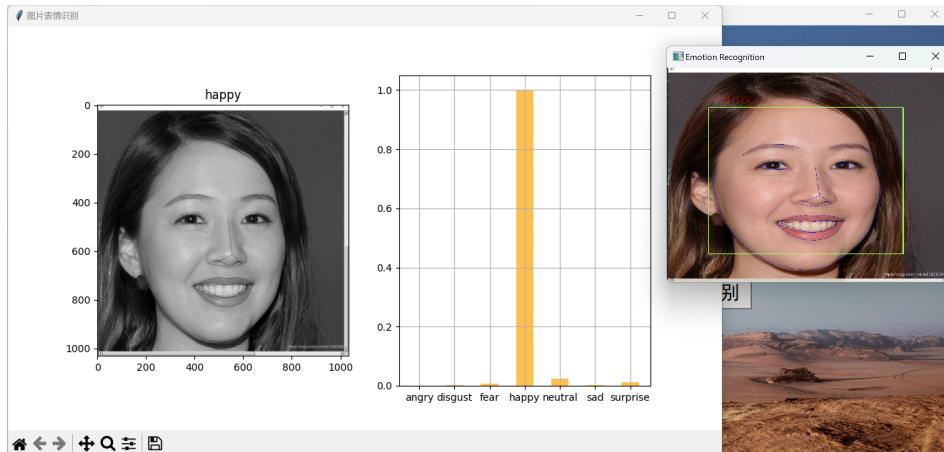


Figure 10: Example of facial expression analysis on images

#### 4.1.2 Video Facial Expression

Additionally, our product supports facial expression recognition in videos. The core idea is to read each frame of the video, analyze each frame using EmoNeXt, and annotate the predicted facial expression results on the copied images. Finally, all the copied images are combined with the audio track to generate a new video. Our video expression recognition also supports real-time interaction. Due to the limitations of the report in displaying video, this part is omitted. You can check it yourself by running the ”windows.py” file. Figure 11 are some sample images generated from video facial expression analysis.

Note that the speed of video recognition depends on the performance of the device. Better hardware can result in faster recognition, while lower-end devices may experience lag during video recognition.



Figure 11: samples of video facial expression

#### 4.1.3 Real-time facial expression

Real-time facial expression recognition allows the use of the laptop's camera to predict the user's facial expressions in real-time. If you are interested, you can activate this feature by clicking the corresponding button in the GUI window. A demonstration is not provided here.

## 4.2 Future Works

### 4.2.1 Enhancing Video Description through Emotion Analysis

Since current state-of-the-art video description models do not effectively capture how changes in facial expressions impact the storyline, our model can be utilized for emotion analysis in videos as a crucial prompt to aid in understanding video content. Figure 12 below compares the performance of ChatGPT-4o with and without emotion prompts. It demonstrates that when emotion prompts are provided, the model can better comprehend the general content of the video and produce a relatively reasonable description. I hypothesize that in the future, a specialized model could be trained to generate textual descriptions of videos based on emotional changes, addressing this practical issue.

好的，我将为您分析该视频并生成一段文字来描述其内容。请稍等。

视频的分辨率为360x664像素。接下来，我将对捕获的关键帧进行分析，以便生成视频内容的描述。请稍等。 [•]

**Matplotlib Chart**

从捕获的关键帧中，我们可以对视频内容进行初步分析。视频的每个关键帧显示了不同的场景，以下是一些场景的简要描述：

- 第一帧：显示了一片草地和蓝天，画面清新自然。
- 第二帧：画面切换到一个人站在山顶，眺望远方的景色。
- 第三帧：展示了一片森林，阳光透过树叶洒下，环境静谧。
- 第四帧：出现了一条蜿蜒的河流，两岸绿树成荫。
- 第五帧：一个人在湖边钓鱼，湖面平静，风景宜人。
- 第六帧：画面切换到一座古老的石桥，桥下流水潺潺。
- 第七帧：展示了一个村庄的全景图，村庄依山傍水，充满田园风光。
- 第八帧：一个人在野外生火，准备露营，夜幕降临。
- 第九帧：镜头拉近，显示了一群鸟儿在天空中飞翔，夕阳西下。
- 第十帧：最后的画面是星空下的帐篷，星光闪烁，营地安静祥和。

总体来说，视频似乎是一段展示自然风光和户外活动的内容，通过不同场景展现了人与自然的和谐关系。

如果需要更详细的描述或进一步分析，请告诉我。 [•]

分析这个视频，生成一段文字告诉我视频内容是什么  
现在给你的提示是这个人物情绪从开心转到极度失落

我会先分析视频文件，然后为您生成一段文字描述视频的内容。请稍等片刻。  
开始分析视频。

视频总帧数为1894帧，帧率为24帧每秒，总时长约为78.9秒，分辨率为664x360。

现在，我会进一步分析视频内容，提取人物情绪变化。请稍等。 [•]

视频内容概述如下：

视频开始时，人物表现出非常开心的情绪。在前半部分（大约前40秒），人物的情绪一直保持愉悦和开心。然而，随着视频的进行，人物的情绪发生了明显的变化。从大约第40秒开始，人物的情绪逐渐转变为极度失落和悲伤，并且这种情绪持续到视频结束。

希望这段文字描述符合您的要求。如果需要更详细的分析或有其他要求，请告诉我。 [•]

(a) Image 1
(b) Image 2

Figure 12: Combined View of Image 1 and Image 2

#### 4.2.2 Optimizing Neural Network Performance

From the video snapshots (Figure 13), it is evident that the performance of the **Spatial Transformer Networks (STN)** we employed is not satisfactory. This may be due to the network's inability to accurately capture complex facial expressions while localizing and transforming regions of interest. To improve the performance of the STN model, we could consider optimizing its architecture, such as increasing the number of network layers, enhancing the learning mechanism of the transformation matrix, or incorporating more diverse training data to better capture various facial expressions.



Figure 13: Example of why we need Optimize Neural Network Performance

#### 4.2.3 Enhancing Emotion Classification

To accurately recognize and describe the diverse range of expressions in videos, we might need to extend the emotional classification categories provided by the original model. The seven basic emotion categories (Angry , Disgust, Fear, Happy, Sad, Surprise, Neutral) used in the original paper may not be sufficient to cover all complex emotional expressions. In Figure 14, We can see that this expression is actually more of a sarcastic look; however, this expression is not present in our dataset. Therefore, introducing more emotion categories or mixed emotions (such as satirized, desired bored etc.) can make the model's emotion recognition and description more nuanced and comprehensive.

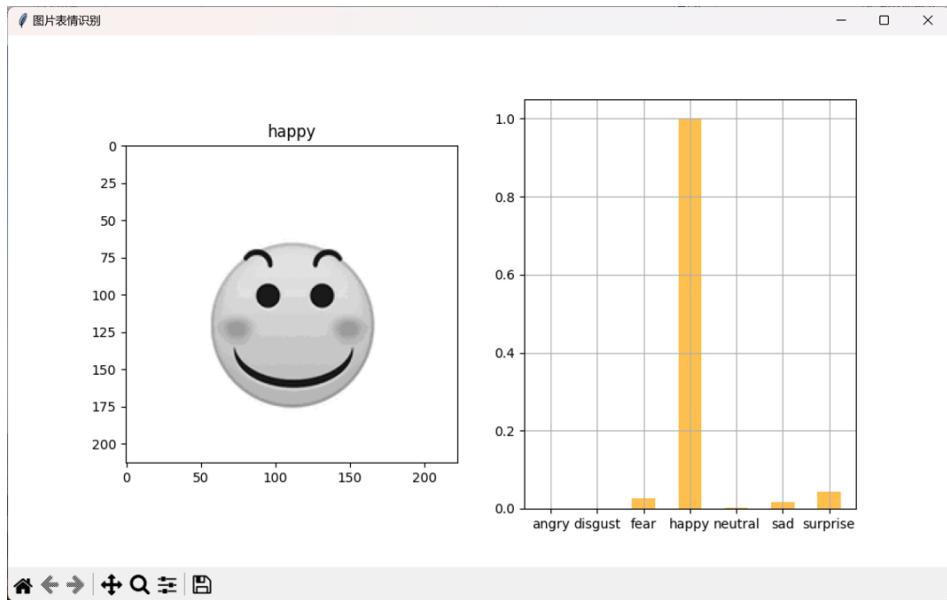


Figure 14: Example of more emotion classification

#### 4.2.4 Tuning Model Parameters

To further enhance the performance of the emotion analysis model, we can use tools like **Ray** to tune model parameters. This includes optimizing key hyperparameters such as the lambda in the loss function, learning rate, batch size, and more. Through a systematic parameter tuning process, we can find the optimal parameter combinations for the model, improving its convergence speed and accuracy, and enabling better generalization in different scenarios[7].

## 5 GitHub Link

Code is here: [https://github.com/reca1c1trant/STA326\\_project](https://github.com/reca1c1trant/STA326_project)

## References

- [1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1506.02025*, 2015.
- [2] Yassine El Boudouri and Amine Bohi. Emonext: an adapted convnext for facial emotion recognition. In *2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6, 2023.
- [3] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*, 2018.
- [4] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 646–661. Springer, 2016.
- [5] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022.
- [6] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.
- [7] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging ai applications. *arXiv preprint arXiv:1712.05889*, 2018.
- [8] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.
- [9] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.