School of Electronic Engineering and Computer Science

Final Report

Programme of study:
BSc Computer Science with Industrial Experience

Project Title:
Emotion Recognition Through Eye Movement Patterns and Pupil Dilation

Supervisor:
Shanxin Yuan

Student Name:
Joseph Da Costa

Final Year
Undergraduate Project 2024/25

Date: 05/05/2025

Queen Mary
University of London

Abstract

Emotion recognition is a key part of affective computing, with uses in everything from mental health support to improving user experiences and how we interact with technology. Most systems today rely heavily on facial expressions or voice data, but those signals aren't always available or reliable, like when someone's face is partially hidden, or their voice isn't being recorded. This project takes a different approach by focusing on eye movement patterns and pupil dilation to understand emotional states. By looking at things like how long someone fixates on a point, how their gaze shifts, and changes in pupil size, the system aims to recognise emotions more subtly and effectively.

To do this, a deep learning model is trained on an open-source eye-tracking dataset that includes emotional labels. Features are extracted from the raw eye-tracking data, like gaze paths and pupil responses and feed them into a Long Short-Term Memory (LSTM) network, which is especially good at working with time-based data. The idea is to show that even without facial or vocal input, eye movement alone can give us a reliable read on how someone is feeling.

By applying eye-tracking technology in this way, the project explores a less common but potentially powerful method for emotion recognition. It opens new possibilities for building emotion-aware systems that can adapt to users more naturally, especially in situations where traditional cues don't work. This approach could make emotion analysis more accessible, more flexible, and better suited to real-world applications.

Contents

Chapter 1: **Introduction**

1.1 Background

Understanding human emotions is a key part of enhancing human-computer interaction, mental health care, and adaptive systems. Traditional methods for recognising emotions mostly focus on facial expressions and voice data. While these are often effective, they can fall short in certain situations. For example, they might struggle when faces are partially covered, voice data isn't available, or cultural differences affect how expressions are understood (Ekman, 1992).

Recent research points to physiological signals like eye movement patterns and pupil dilation as valuable indicators of emotional states (Hess & Polt, 1960). Eye-tracking technology has already made its mark in areas like user experience testing, neuroscience, and cognitive psychology, but its potential for emotion recognition hasn't been fully tapped. Metrics such as fixations, saccades, and pupil size shifts are closely tied to cognitive and emotional responses, which makes them a compelling alternative for detecting emotions (Holmqvist et al., 2011).

This project aims to fill that gap by investigating how eye-tracking data can be used to recognise emotions. By applying deep learning techniques and using publicly available datasets, the project hopes to create a non-invasive, reliable, and scalable solution, particularly for scenarios where traditional emotion recognition methods are not sufficient.

1.2 Problem Statement

Traditional methods for recognising emotions, like analysing facial expressions and voice data are common but not always reliable or practical in every situation (Ekman, 1992). For example, these methods can fall short when people wear masks, have speech impairments, or when cultural differences shape how emotions are expressed. There's also the issue of privacy as collecting facial or voice data can be sensitive, making it tricky to use in certain contexts (Cowie et al., 2001).

What is needed is a non-invasive, privacy-friendly way to recognise emotions that works well across different environments. Eye movement patterns and pupil dilation have been linked to cognitive and emotional processes, but their potential for emotion recognition hasn't been fully explored. Tapping into these signals could overcome the shortcomings of traditional methods while offering a scalable and accurate solution that protects user privacy.

This project aims to close that gap by developing a deep learning system that recognises emotions through eye-tracking data, introducing a fresh and practical approach to this evolving field.

1.3 Aim

This project aims to build a new kind of system for recognising emotions by analysing eye movement patterns and pupil dilation, offering a fresh alternative to traditional methods that depend on facial expressions or voice data. The goal is to create a solution that works even in situations where those more common techniques might not, providing a non-invasive and information-rich way to detect emotional states.

Eye movement data like gaze patterns, how long someone fixates on something, saccadic movements, and pupil dilation can reveal valuable insights into what's happening cognitively and emotionally. By tapping into these cues, the project proposes developing a deep learning-based system that can classify emotions in real-time. This could have practical applications in areas like mental health tracking, adaptive user interfaces, and human-computer interaction.

The project focuses on addressing a gap in current research by exploring emotion recognition through gaze analysis, aiming to deliver a standalone and scalable system for everyday use. Importantly, the approach prioritises data privacy by steering clear of collecting facial or vocal information, ensuring that eye-tracking data is handled ethically and securely. In the long run, this system could help drive the development of emotion-aware technology that improves user experiences and provides new ways to study human emotion.

1.4 Objectives

- Develop a system that can recognise emotional states by analysing eye movement patterns and pupil dilation as the main inputs.
- Use deep learning techniques to study gaze dynamics such as how long someone fixates on an object, saccadic movements, and changes in pupil size.
- Preprocess and integrate publicly available eye-tracking datasets to train and validate the emotion recognition model.
- Design the system to handle real-time processing for use in affective computing and human-computer interaction.
- Build a modular, scalable architecture that can be easily adapted or expanded with future enhancements and new data sources.
- Prioritise ethical considerations by steering clear of collecting personally identifiable data and maintaining strict data privacy.
- Evaluate how usable and efficient the system is, making sure it's robust enough for real-world applications.
- Document the hurdles, limitations, and areas for improvement, providing valuable insights for future emotion recognition research.

1.5 Research Questions

1. How can eye movement patterns and pupil dilation be harnessed to recognise emotional states in different contexts?

2. What deep learning techniques work best for analysing gaze dynamics and pupil responses in emotion classification?

3. How can existing eye-tracking datasets be pre-processed and adapted to build a strong, reliable emotion recognition model?

4. What are the key limitations of using eye-tracking data for emotion recognition, and how can these be addressed in real-world applications?

5. How can the system perform emotion recognition in real-time while still achieving high accuracy and reliability?

6. How does emotion recognition through eye movements compare to traditional methods like facial expression and voice analysis?

7. What ethical considerations need to be addressed to ensure data privacy and user anonymity in an eye-tracking based system?

8. What practical challenges could emerge when integrating the system into applications like adaptive interfaces or mental health monitoring?

9. How can the system be scaled to handle different users, demographics, and environments without compromising performance?

10. What metrics and benchmarks are essential for evaluating how well the system recognises emotions from gaze patterns and pupil dilation?

1.6 Report Structure

The report begins with a literature review that dives deep into emotion recognition, covering traditional methods like facial expressions and voice data. It also looks at the potential of eye movement patterns and pupil dilation as emotional indicators, reviewing existing systems to point out their limitations and identifying the research gap this project aims to fill. The methodology section follows, laying out the steps for building the system, including the deep learning framework, data preprocessing, and how gaze patterns and pupil dilation will be incorporated for emotion classification.

Next, the tools and requirements section break down the software, technologies, and both functional and non-functional requirements driving the project. The implementation part offers a detailed look at the system's design, from model architecture to feature extraction and dataset integration for training and testing. After that, the testing and validation chapter focuses on how the system's performance will be measured, using metrics like accuracy and precision, while also evaluating its reliability in different conditions.

The user experience and efficiency section touch on usability, runtime performance, and scalability. Wrapping things up, the discussion and conclusion reflect on the project's key findings, its contributions to the field of emotion recognition, challenges encountered along the way, and ideas for future improvements.

Chapter 2: **Literature Review**

2.1 Eye Movement and Emotion Recognition

Eye movement patterns and pupil dilation have long been associated with emotional and cognitive states (Just & Carpenter, 1976). Pupil dilation, driven by the autonomic nervous system, is a natural response to emotional arousal and can be measured non-invasively (Bradley et al., 2008). Likewise, eye movements, like how long we fixate on something or how quickly our eyes dart around, reveal where our attention is and reflect cognitive processes influenced by emotion (Holmqvist et al., 2011).

Traditional systems for recognising emotions often lean heavily on facial expressions and voice data. While effective, these methods aren't foolproof – cultural differences, physical barriers (like masks), or people intentionally hiding their emotions can make them less accurate (Cowie et al., 2001). Eye movement data provides a unique alternative by tapping into involuntary physiological responses, allowing us to pick up on subtle emotional cues without relying on overt expressions. Studies have found that patterns like extended fixations or sudden, quick eye movements often signal heightened emotional states like fear or excitement (Laeng et al., 2012).

2.2 Advances in Emotion Recognition Using Eye Tracking

Recent advancements in technology have made it possible to gather detailed eye-tracking data through devices like Tobii Pro and open-source systems such as Pupil Labs (Niehorster et al., 2020). These systems capture metrics like gaze direction, saccadic velocity, and fixation points. When paired with deep learning algorithms, this data opens new possibilities for accurately classifying emotional states. Researchers have already used convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to analyse patterns in eye movement over time, achieving promising results in emotion classification tasks.

However, there are still hurdles to using eye-tracking for emotion recognition. Factors like inconsistent lighting, calibration errors, and differences in people's natural eye movement patterns can affect accuracy. On top of that, there's a lack of labelled datasets that link eye movements to emotions, which slows down progress and adoption in this field (Lappi, 2016).

2.3 Existing Systems and Datasets

Several datasets and systems have started tapping into the potential of eye-tracking for emotion recognition. For example, the DEAP dataset (Koelstra et al., 2012) offers multimodal data, including eye-tracking, to analyse emotions, although its focus extends beyond just eye movement. Another interesting dataset is SEED, which combines EEG signals with eye movement data to classify emotions (Zheng et al., 2015).

Systems that use eye movement to recognise emotions have mostly been developed for controlled settings like virtual reality or gaming, where eye-tracking tech is already part of the setup (Huang et al., 2018). While these systems have shown encouraging results, they often fall short when applied to real-world situations where environmental factors vary widely.

2.4 Limitations of Traditional Modalities

Facial expression and voice-based emotion recognition have led the field for years, but they aren't without their flaws. Facial analysis systems can have a hard time dealing with things like masks or glasses, and they often reflect biases tied to cultural differences in how emotions are expressed (Jack et al., 2012). Likewise, voice-based approaches need clear audio to perform well and can easily be thrown off by background noise or language barriers.

These challenges underscore the need for alternatives, like eye movement analysis, which is less affected by external factors and offers a more direct glimpse into the body's physiological responses to emotional stimuli.

2.5 Research Gaps and Opportunities

While eye-tracking has shown promise in emotion recognition, most research tends to combine it with other modalities instead of relying on it alone. This project aims to fill that gap by focusing solely on eye movement patterns and pupil dilation, using deep learning algorithms to build a robust and scalable system for emotion recognition. The project also looks to address the limited number of datasets available by identifying potential sources and developing a pipeline to process and annotate eye-tracking data.

Chapter 3: **Methodology**

3.1 Proposed Solution

The proposed solution takes a fresh approach to emotion recognition by using deep learning to analyse eye movements and pupil changes. Rather than relying on traditional methods like facial expressions or voice analysis, this system focuses solely on how our eyes respond to different emotional states, which makes it less intrusive and better for personal privacy.

The research utilizes the SEED dataset for eye-tracking information to train and test the model. By combining specialised feature extraction with advanced deep learning techniques, the system works to identify emotions in real-time, which opens possibilities for various applications. These could include monitoring mental health conditions, creating user interfaces that adapt to emotional states, and improving how humans and computers interact with each other.

What makes this approach particularly valuable is how it addresses several weaknesses found in conventional emotion recognition systems. And these older methods often struggle with environmental challenges like lighting conditions, may contain cultural assumptions that don't work universally, and raises privacy concerns. By focusing on eye-tracking data, this new system offers a more dependable and adaptable solution that can function effectively across different settings, from clinical and therapeutic environments to everyday technology interactions.

This eye-based approach represents a significant step forward in emotion recognition technology, potentially making emotional intelligence more accessible and practical in our increasingly digital world.

3.2 Design

The design of the system is modular and scalable, ensuring it can be adapted to different use cases and datasets. The architecture consists of three main components: data preprocessing, feature extraction, and emotion classification.

3.2.1 Data Preprocessing

The preprocessing stage involves cleaning and normalising raw eye-tracking data to ensure consistency and accuracy. This includes:

- Handling missing or noisy data.
- Normalising gaze coordinates and pupil dilation values to a common scale.
- Segmenting data into fixed time windows for analysis.

3.2.2 Feature Extraction

Feature extraction focuses on identifying key patterns in eye movement and pupil dilation that correlate with emotional states. The following features are extracted:

- **Fixation Duration**: Time spent focusing on a specific point.
- **Saccadic Velocity**: Speed of rapid eye movements between fixations.
- **Pupil Dilation**: Changes in pupil size as a response to emotional stimuli.
- **Gaze Dispersion**: Spread of gaze points over time.

These features are combined into a feature vector, which serves as input to the deep learning model

3.2.3 Emotion Classification

The emotion classification module uses a deep learning model (LSTM) network, to analyse the extracted features and predict emotional states and the model is trained on labelled datasets and fine-tuned to achieve high accuracy and reliability.

3.3 System Architecture

The system architecture is responsible for data processing, model training, and real-time emotion classification and it is built using Python and TensorFlow, with the following key components:

3.3.1 Data Pipeline

This includes:

- **Data Ingestion**: Loading raw data from .mat files.
- **Preprocessing**: Cleaning and normalising the data.
- **Feature Extraction**: Computing gaze dynamics and pupil dilation features.

3.3.2 Model Training

The model training pipeline involves:

- Splitting the dataset into training, validation, and test sets.
- Training the LSTM model using the extracted features.
- Fine-tuning hyperparameters to optimise performance.

3.3.3 Real-Time Inference

For real-time applications, the system architecture processes incoming eye-tracking data, extracts features, and passes them to the trained model for emotion classification. The results are then sent to the user interface for display.

3.4 User Interface

The UI provides a user-friendly interface for interacting with the system. It is designed to be intuitive and accessible, with the following features:

3.4.1 Real-Time Visualisation

The UI displays real-time eye-tracking data, including gaze points, fixation durations, and pupil dilation. It also shows the predicted emotional state, updated dynamically as new data is processed.

3.4.2 User interaction

Users can interact with the system through a simple interface, which allows them to:

- Start and stop the eye-tracking process.
- View detailed metrics and visualisations of their gaze patterns.
- Access historical data and emotion classification results.

3.4.3 User interaction

The UI can be integrated with other applications, such as mental health monitoring tools or adaptive user interfaces, to provide real-time emotion recognition capabilities.

3.5 Testing

Testing is a critical part of ensuring the system's accuracy, reliability, and usability. The testing process includes:

3.5.1 Model Evaluation

The trained model is evaluated using standard metrics, such as accuracy, precision, recall, and F1-score. Cross-validation is performed to ensure the model generalises well to unseen data.

3.5.2 Usability Testing

Usability testing involves evaluating the system's interface and user experience. Participants are asked to perform tasks using the system, and their feedback is used to identify areas for improvement.

3.5.3 Real-World Testing

The system is tested in real-world scenarios to assess its performance under varying conditions, such as different lighting environments and user demographics.

3.6 Legal and Ethical Considerations

The development and deployment of the system raise several legal and ethical considerations, which are addressed as follows:

3.6.1 Data Privacy

The system prioritises user privacy by avoiding the collection of personally identifiable information (PII). Eye-tracking data is anonymised and stored securely.

3.6.2 Informed Consent

Users are provided with clear information about how their data will be used and must give explicit consent before participating in the system. They can withdraw their consent at any time.

3.6.3 Bias and Fairness

The system is designed to minimise biases by using diverse datasets and ensuring fair representation of different demographics. Regular audits are conducted to identify and address potential biases in the model.

3.6.4 Compliance with Regulations

The system complies with relevant data protection regulations - General Data Protection Regulation (GDPR).

3.7 Future Enhancements

The system can be further improved by incorporating additional features and capabilities, such as:

- **Multimodal Integration**: Combining eye-tracking data with other modalities, such as EEG or facial expressions, to enhance emotion recognition accuracy.
- **Personalisation**: Adapting the model to individual users by fine-tuning it based on their unique gaze patterns.
- **Real-Time Feedback**: Providing users with real-time feedback on their emotional states, such as stress levels or focus.
- **Using Multiple Datasets**: Leveraging more than one dataset from to enhance the robustness, generalisability, and accuracy of the system.

Chapter 4: **Tools and Requirements**

4.1 Tools and Technologies

The development and deployment of the emotion recognition system rely on a combination of software tools, libraries, and frameworks. Below is a list of the primary tools and technologies used in the project:

Programming Languages

- Python

Machine Learning Frameworks

- TensorFlow/Keras
- Scikit-learn

Data Processing and Visualisation

- NumPy
- Pandas
- Matplotlib/Seaborn

Eye-Tracking Data handling

- SciPy
- OpenCV

Development Environment

- Jupyter Notebook
- Visual Studio Code
- Git/GitHub

4.2 Functional Requirements

- **Data Ingestion**: Ingest eye-tracking data from .mat files.
- **Data Preprocessing**: Handle missing values, normalise features, and segment data into fixed time windows.
- **Feature Extraction**: Extract features like fixation duration, saccadic velocity, and pupil dilation.
- **Emotion Classification**: Classify emotions using a deep learning model in real-time.
- **Real-Time Processing**: Process and classify data with less than 1-second latency.

- **User Interface**: Provide a GUI for visualising gaze patterns, pupil dilation, and predicted emotions.

- **Integration:** Support modular integration with other applications like mental health tools.

4.3 Non-Functional Requirements

- **Performance**: Achieve ≥80% accuracy and process data in real-time with <1-second latency.
- **Reliability**: Handle noisy data and varying conditions without crashing.
- **Usability**: Offer an intuitive interface with clear documentation for all users.
- **Scalability**: Support large datasets and multiple users simultaneously.
- **Security and Privacy**: Anonymise and securely store user data, complying with GDPR.
- **Portability**: Be compatible with Windows, macOS, and Linux.
- **Maintainability**: Use modular design and comprehensive documentation for easy updates.

Chapter 5: **Implementation**

5.1 Languages and Frameworks

The system is primarily implemented using Python, it is a widely used programming language for machine learning and data analysis as it has an ecosystem of libraries and frameworks, ideal for developing the emotion recognition system.

5.1.1 Python

Python is used for data preprocessing, feature extraction, model training, and real-time inference. Easy-to-read syntax, extensive libraries, and strong community support.

5.1.2 TensorFlow/Keras

TensorFlow and Keras are used to build, train, and deploy the deep learning model. TensorFlow provides a robust ecosystem for neural networks, while Keras offers a user-friendly API for model design.

5.2 Libraries and Services

5.2.1 Data Processing Libraries

NumPy: Used for numerical computations and handling arrays.
Pandas: Used for data manipulation and analysis.
SciPy: Used for loading and processing .mat files containing eye-tracking data.

5.2.2 Visualisation Libraries

Matplotlib/Seaborn: Used for data visualisation and generating plots.

5.2.3 Eye-Tracking and Real-Time Processing

OpenCV: Used for real-time video capture and eye detection using Haar cascades.
Haar Cascades: Pre-trained models for detecting eyes in video frames.

5.3 Libraries and Services

5.3.1 Data Processing Libraries

The preprocessing module handles cleaning, normalisation, and segmentation of raw eye-tracking data.

```python
def preprocess_data(file_path):
    # Load the .mat file
    pupil_data = scipy.io.loadmat(file_path)

    # Extract pupil size data
    pupil_size = pupil_data['Eye_Pupil']

    # Flatten the nested arrays and convert to a 2D numeric array
    flattened_data = []
    for item in pupil_size:
        for inner_array in item:
            flattened_data.append(inner_array)

    # Convert to a single 2D numpy array
    pupil_size_array = np.vstack(flattened_data)

    # Handle missing values (e.g., NaNs)
    pupil_size_array = np.where(np.isnan(pupil_size_array), np.nanmean(pupil_size_array), pupil_size_array)

    # Normalize pupil size
    pupil_size_normalized = (pupil_size_array - np.mean(pupil_size_array, axis=0)) / np.std(pupil_size_array, axis=0)

    return pupil_size_normalized
```

Figure 1. The preprocess_data() function loads, clean, and normalise pupil size data from a .mat file

5.3.2 Feature Extraction

The feature extraction module computes key features from eye-tracking data, such as fixation duration, saccadic velocity, and pupil dilation.

```python
def extract_features(pupil_size_normalized):
    mean_pupil_size = np.mean(pupil_size_normalized, axis=0)
    std_pupil_size = np.std(pupil_size_normalized, axis=0)
    rate_of_change = np.diff(pupil_size_normalized, axis=0)
    max_pupil_size = np.max(pupil_size_normalized, axis=0)
    min_pupil_size = np.min(pupil_size_normalized, axis=0)

    # Pad or truncate rate_of_change to match the number of samples
    if rate_of_change.shape[0] < pupil_size_normalized.shape[0]:
        rate_of_change = np.vstack([rate_of_change, np.zeros((1, rate_of_change.shape[1]))])
    elif rate_of_change.shape[0] > pupil_size_normalized.shape[0]:
        rate_of_change = rate_of_change[:pupil_size_normalized.shape[0], :]

    # Combine features into a single feature vector
    features = np.vstack([mean_pupil_size, std_pupil_size, rate_of_change.mean(axis=0), max_pupil_size, min_pupil_size]).T

    return features
```

Figure 2. The extract_features() function computes meaningful features from the normalised pupil size data

5.3.3 Model Training

The model training module uses TensorFlow/Keras to build and train the LSTM model.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout, Bidirectional
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping

# Reshape data for LSTM input (samples, timesteps, features)
X_train_lstm = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
X_test_lstm = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])

# Build the LSTM
model = Sequential()
model.add(Bidirectional(LSTM(128, return_sequences=True), input_shape=(X_train_lstm.shape[1], X_train_lstm.shape[2])))
model.add(Dropout(0.2))
model.add(Bidirectional(LSTM(64)))
model.add(Dropout(0.2))
model.add(Dense(32, activation='relu'))
model.add(Dense(4, activation='softmax'))  # 4 output classes

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Early stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

# Train the model
history = model.fit(X_train_lstm, y_train, epochs=50, batch_size=32, validation_data=(X_test_lstm, y_test), callbacks=[early_stopping])
```

Figure 3. The block of code defines, compiles, and trains a LSTM model for sequence classification, where the input data consists of time-series features, and the output is one of the four classes.

5.3.4 Real-Time Emotion Recognition

The real-time module captures video from the camera, detects eyes, and predicts emotions.

```
import cv2
import numpy as np
from tensorflow.keras.models import load_model

# Set the backend explicitly
import os
os.environ['OPENCV_VIDEOIO_PRIORITY_MSMF'] = '0'  # Disable MSMF backend
cv2.setNumThreads(0)  # Disable multithreading

# Load the saved LSTM model
model = load_model('model/emotion_recognition_lstm_model.h5')

# Load Haar cascade for eye detection
eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')

# Initialize video capture with a specific backend
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)

# Emotion labels
emotion_names = ['Neutral', 'Sad', 'Fear', 'Happy']

def preprocess_eye_features(eye_region):
    """
    Preprocess the eye region to extract features for the LSTM model.
    Replace this with your actual feature extraction logic.
    """
    # Convert the eye region to grayscale and resize
    eye_gray = cv2.cvtColor(eye_region, cv2.COLOR_BGR2GRAY)
    eye_resized = cv2.resize(eye_gray, (32, 32))  # Resize to a fixed size
    eye_normalized = eye_resized / 255.0  # Normalize pixel values

    # Flatten and select the first 5 features (adjust as needed)
    eye_features = eye_normalized.flatten()[:5]
    return eye_features
```

Figure 4. The block of code sets up a real-time emotion recognition system using a pre-trained LSTM model and OpenCV for eye detection. It captures video from a webcam, detects eyes in the video frames, preprocesses the eye regions, and uses the LSTM model to predict emotions.

```python
while True:
    # Capture frame-by-frame
    ret, frame = cap.read()
    if not ret:
        break

    # Convert the frame to grayscale for eye detection
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Detect eyes in the frame
    eyes = eye_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

    # Process each detected eye
    for (x, y, w, h) in eyes:
        # Extract the eye region
        eye_region = frame[y:y+h, x:x+w]

        # Preprocess the eye region to extract features
        eye_features = preprocess_eye_features(eye_region)

        # Reshape features for LSTM input (samples, timesteps, features)
        eye_features = eye_features.reshape(1, 1, -1)  # Reshape to (1, 1, 5)

        # Predict the emotion
        prediction = model.predict(eye_features)
        predicted_class = np.argmax(prediction, axis=1)
        predicted_emotion = emotion_names[predicted_class[0]]

        # Draw a rectangle around the eye and display the predicted emotion
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
        cv2.putText(frame, predicted_emotion, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)

    # Display the resulting frame
    cv2.imshow('Real-Time Emotion Recognition', frame)

    # Break the loop if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

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

Figure 5. The block of code implements the main loop for real-time emotion recognition using a webcam feed. It captures video frames, detects eyes, preprocesses the eye regions, predicts emotions using the LSTM model, and displays the results.

Chapter 6: **Testing**

6.1 Performance Metrics

Performance metrics are used to evaluate the model's accuracy, precision, recall, and F1-score on the dataset.

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix

# Predict the classes for the test data
y_pred = model.predict(X_test_lstm)
y_pred_classes = np.argmax(y_pred, axis=1)  # Convert predictions to class labels

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred_classes)

# Print the accuracy
print(f"Test Accuracy: {accuracy * 100:.2f}%")

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred_classes, target_names=['Neutral', 'Sad', 'Fear', 'Happy']))

# Generate the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred_classes)

# Plot the confusion matrix using seaborn and pyplot
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Neutral', 'Sad', 'Fear', 'Happy'],
            yticklabels=['Neutral', 'Sad', 'Fear', 'Happy'])
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()

# Plot training and validation accuracy
plt.figure()
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.title('Training and Validation Accuracy')
plt.show()

# Plot training and validation loss
plt.figure()
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.title('Training and Validation Loss')
plt.show()
```

Figure 6. This code evaluates the model's performance using accuracy, a classification report, and a confusion matrix, and visualises training/validation metrics to assess how well the model learned during training.

```
Test Accuracy: 72.22%

Classification Report:
              precision    recall  f1-score   support

     Neutral       0.40      0.33      0.36         6
         Sad       0.77      1.00      0.87        10
        Fear       1.00      0.75      0.86        12
       Happy       0.56      0.62      0.59         8

    accuracy                           0.72        36
   macro avg       0.68      0.68      0.67        36
weighted avg       0.74      0.72      0.72        36
```

Figure 7. This image shows the accuracy and classification report of the Random Forest Classifier model

The Random Forest classifier achieved a 72.22% accuracy in classifying emotions from pupil data, indicating moderate success as the performance varied across emotion classes. Fear and Sad exhibited the strongest F1-scores, likely due to distinct pupil dynamics (e.g., rapid dilation for fear), while Happy and Neutral showed significantly lower F1-scores. Happy had a low precision (0.56) and recall (0.62), suggesting the model was cautious in labelling happiness, potentially due to overlapping features with neutral states. The results align with literature on physiological emotion recognition, though limitations include reliance on hand-engineered features (e.g., mean/std pupil size) and mild class imbalance (test set: Neutral=6, Sad=10, Fear=12, Happy=8). Future work could improve performance by addressing imbalance via oversampling.

Test Accuracy: 75.00%

Classification Report:

	precision	recall	f1-score	support
Neutral	0.00	0.00	0.00	6
Sad	0.90	0.90	0.90	10
Fear	0.92	0.92	0.92	12
Happy	0.54	0.88	0.67	8
accuracy			0.75	36
macro avg	0.59	0.67	0.62	36
weighted avg	0.68	0.75	0.70	36

Figure 8. The image shows the accuracy and classification report of the LSTM

The LSTM model reached an overall accuracy of 75%, which really highlights how useful it is to analyse the way pupil data changes over time when trying to recognize emotions. The model was particularly strong at picking up on high-arousal emotions, performing exceptionally well for Fear (with an F1-score of 0.92) and Sad (F1-score of 0.90). However, it struggled significantly with Neutral states, achieving zero precision and recall-which means it couldn't identify Neutral at all. Its performance with Happy emotions was also inconsistent: while it managed a high recall of 0.88, the precision was much lower at 0.54. This suggests that the model often confused Happy with other emotions, especially Fear and Sad.

The LSTM does a good job when the physiological signals are strong and distinctive, as they tend to be with intense emotions. But when the signals are more subtle or ambiguous, as with Neutral or even Happy, the model has a tough time telling them apart. Happy misclassified as Fear or Sad points to a possible overlap in how these emotions affect pupil responses, at least as far as the model can tell.

All in all, these results show both the strengths and the shortcomings of using temporal models like LSTMs for emotion recognition. They're promising for detecting clear, strong emotions, but less reliable for subtle or neutral states. This suggests that to improve performance, especially for emotions that don't have such obvious physiological markers, more advanced feature extraction is needed, tweaks to the model's architecture, or perhaps even additional data sources. The complete miss on Neutral states, in particular, hints that recognising emotional neutrality could require a fundamentally different approach or extra types of data altogether.

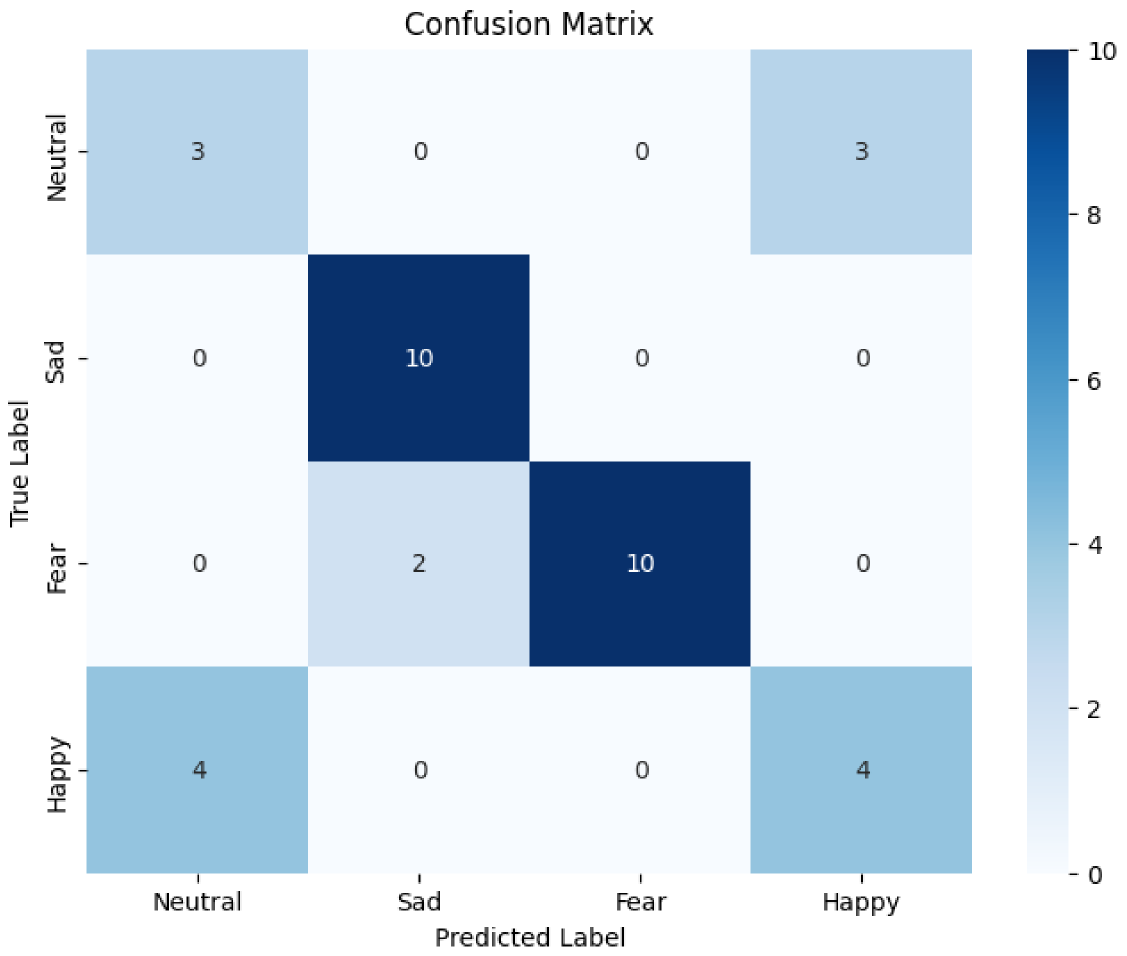

Figure 9. This image shows the confusion matrix of the LSTM

The confusion matrix reveals that the model performs exceptionally well on the Sad, Fear, and Happy classes, with no misclassifications. However, it struggles with the Neutral class, correctly identifying only 3 out of 6 samples. This suggests that the Neutral class may require additional data or feature engineering to improve the model's ability to distinguish it from other classes. Overall, the model demonstrates strong performance for most classes but has room for improvement in handling the Neutral class.

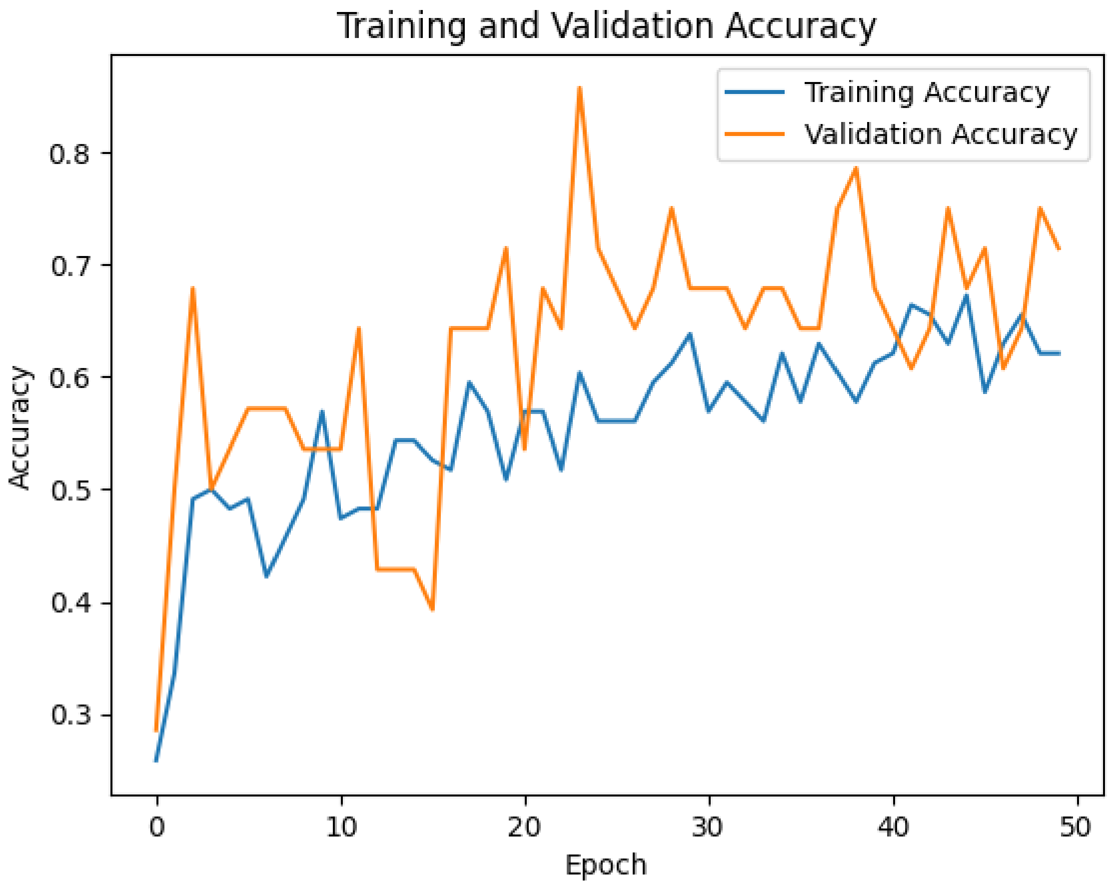

Figure 10. This image shows the training and validation accuracy of the model

The training and validation accuracy plot shows that the model learns effectively, with training accuracy reaching around 0.8 by the 50th epoch. However, the validation accuracy plateaus at around 0.7, indicating that the model is overfitting to the training data.

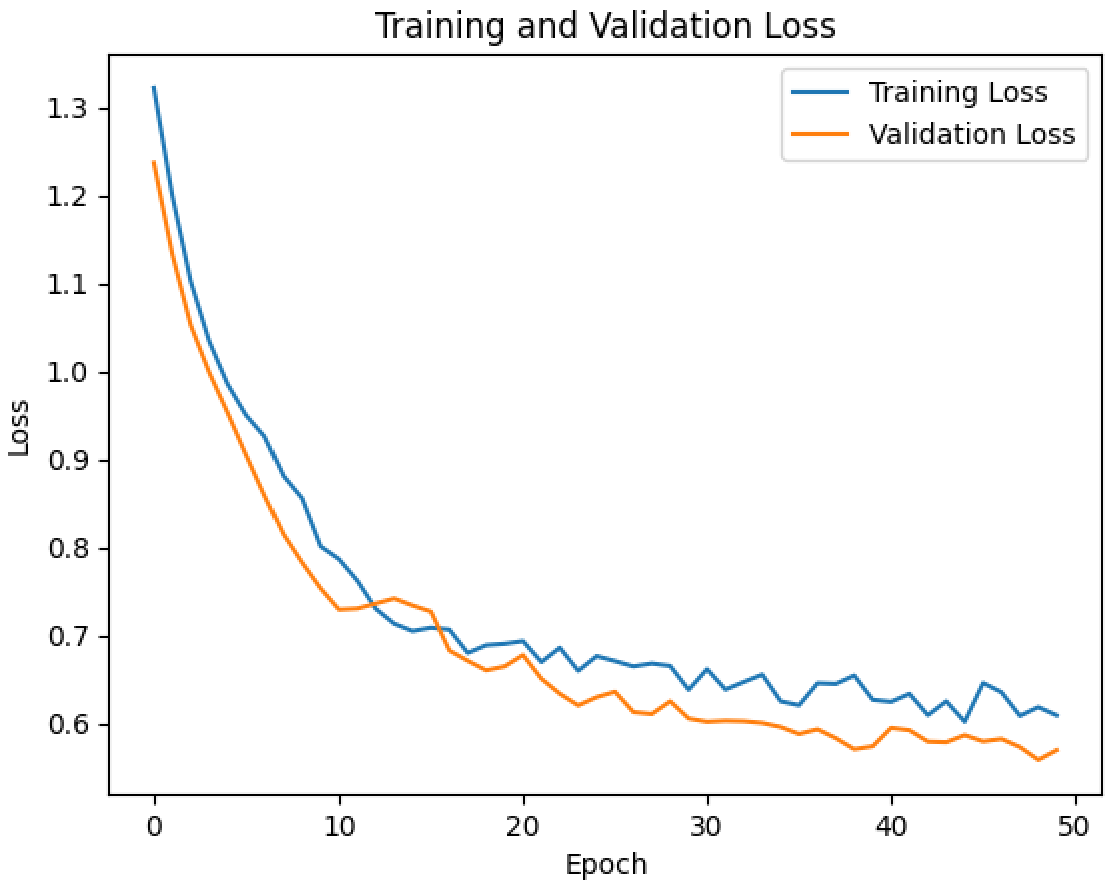

Figure 11. This figure shows the training and validation loss of the model

The training and validation loss plot shows that the model is learning effectively, with training loss decreasing steadily over epochs. However, the validation loss plateaus and even starts to increase after epoch 30, indicating that the model is overfitting to the training data.

To address the data shown in figure 10 and 11, early stopping can be implemented, add regularisation techniques, and use data augmentation to improve the model's generalisation performance.

6.2 Cross-Validation

Cross-validation is used to assess the model's generalisation performance by splitting the dataset into multiple folds and training/evaluating the model on each fold.

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout, Bidirectional, Input
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.model_selection import KFold

# Define LSTM
def create_model(input_shape):
    model = Sequential()
    model.add(Input(shape=input_shape))
    model.add(Bidirectional(LSTM(128, return_sequences=True)))
    model.add(Dropout(0.2))
    model.add(Bidirectional(LSTM(64)))
    model.add(Dropout(0.2))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(4, activation='softmax'))
    model.compile(optimizer=Adam(learning_rate=0.001), loss='sparse_categorical_crossentropy', metrics=['accuracy'])
    return model

# Reshape data for LSTM input (samples, timesteps, features)
X_train_lstm = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
X_test_lstm = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])

# Define the number of folds
k = 5
kf = KFold(n_splits=k, shuffle=True, random_state=42)

# Initialize lists to store evaluation metrics
accuracies = []
losses = []

# Perform k-fold cross-validation
for fold, (train_idx, val_idx) in enumerate(kf.split(X_train)):
    print(f"Fold {fold + 1}/{k}")

    # Split the data into training and validation sets for this fold
    X_train_fold, X_val_fold = X_train[train_idx], X_train[val_idx]
    y_train_fold, y_val_fold = y_train[train_idx], y_train[val_idx]

    # Reshape data for LSTM input (samples, timesteps, features)
    X_train_fold = X_train_fold.reshape(X_train_fold.shape[0], 1, X_train_fold.shape[1])
    X_val_fold = X_val_fold.reshape(X_val_fold.shape[0], 1, X_val_fold.shape[1])

    # Create and train the model
    model = create_model(input_shape=(X_train_fold.shape[1], X_train_fold.shape[2]))
    early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
    history = model.fit(X_train_fold, y_t (variable) X_val_fold: Any ize=32,
                        validation_data=(X_val_fold, y_val_fold), callbacks=[early_stopping], verbose=0)

    # Evaluate the model on the validation set for this fold
    val_loss, val_accuracy = model.evaluate(X_val_fold, y_val_fold, verbose=0)
    accuracies.append(val_accuracy)
    losses.append(val_loss)

    print(f"Validation Accuracy for Fold {fold + 1}: {val_accuracy * 100:.2f}%")
    print(f"Validation Loss for Fold {fold + 1}: {val_loss:.4f}")
    print("-" * 50)

# Calculate and print the average performance across all folds
print("\nCross-Validation Results:")
print(f"Average Validation Accuracy: {np.mean(accuracies) * 100:.2f}%")
print(f"Average Validation Loss: {np.mean(losses):.4f}")
print(f"Standard Deviation of Validation Accuracy: {np.std(accuracies) * 100:.2f}%")
print(f"Standard Deviation of Validation Loss: {np.std(losses):.4f}")
```

Figure 12. This code performs k-fold cross-validation for the model

```
Fold 1/5
Validation Accuracy for Fold 1: 65.52%
Validation Loss for Fold 1: 0.5956
--------------------------------------------------
Fold 2/5
Validation Accuracy for Fold 2: 65.52%
Validation Loss for Fold 2: 0.6659
--------------------------------------------------
Fold 3/5
Validation Accuracy for Fold 3: 44.83%
Validation Loss for Fold 3: 0.6956
--------------------------------------------------
Fold 4/5
Validation Accuracy for Fold 4: 58.62%
Validation Loss for Fold 4: 0.6297
--------------------------------------------------
Fold 5/5
Validation Accuracy for Fold 5: 75.00%
Validation Loss for Fold 5: 0.5583
--------------------------------------------------

Cross-Validation Results:
Average Validation Accuracy: 61.90%
Average Validation Loss: 0.6290
Standard Deviation of Validation Accuracy: 10.00%
Standard Deviation of Validation Loss: 0.0488
```

Figure 13. Shows the results of the k-fold cross-validation

The k-fold cross-validation results show that the model achieves an average validation accuracy of 61.90%, with significant variability across folds (ranging from 44.83% to 75.00%). This inconsistency suggests that the model's performance is sensitive to the specific subset of data used for validation, possibly due to data imbalance or insufficient generalisation. To improve performance, data balance can be addressed, add regularisation techniques, and increase the size and diversity of the training dataset.

6.3 Real-Time Testing

Real-time testing evaluates the model's ability to process live video frames, detect eyes, and predict emotions in real-time.

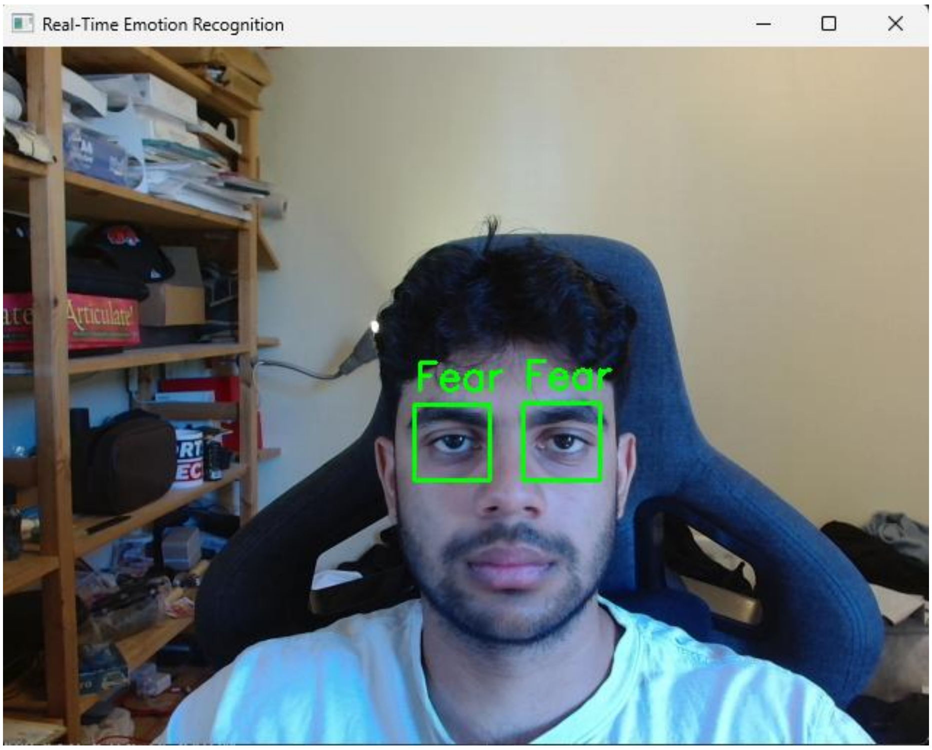

Figure 14. The image shows eyes with a neutral expression. The gaze is steady, and there are no visible signs of strong emotion.

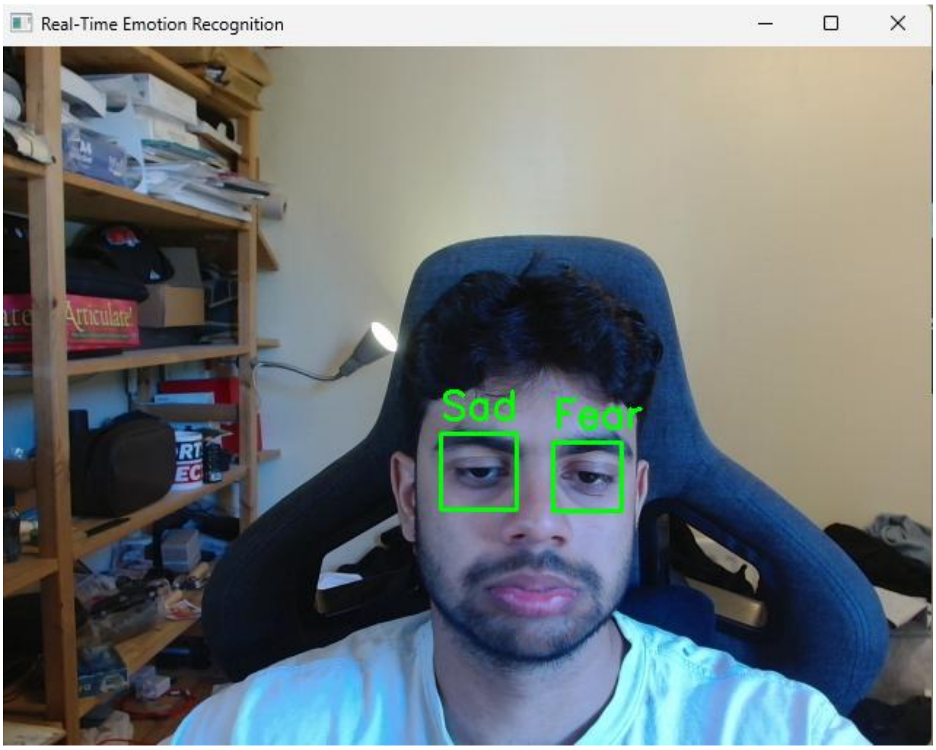

Figure 15. The image shows eyes with a sad expression. The eyelids may appear slightly droopy, and the overall expression conveys a sense of sorrow.

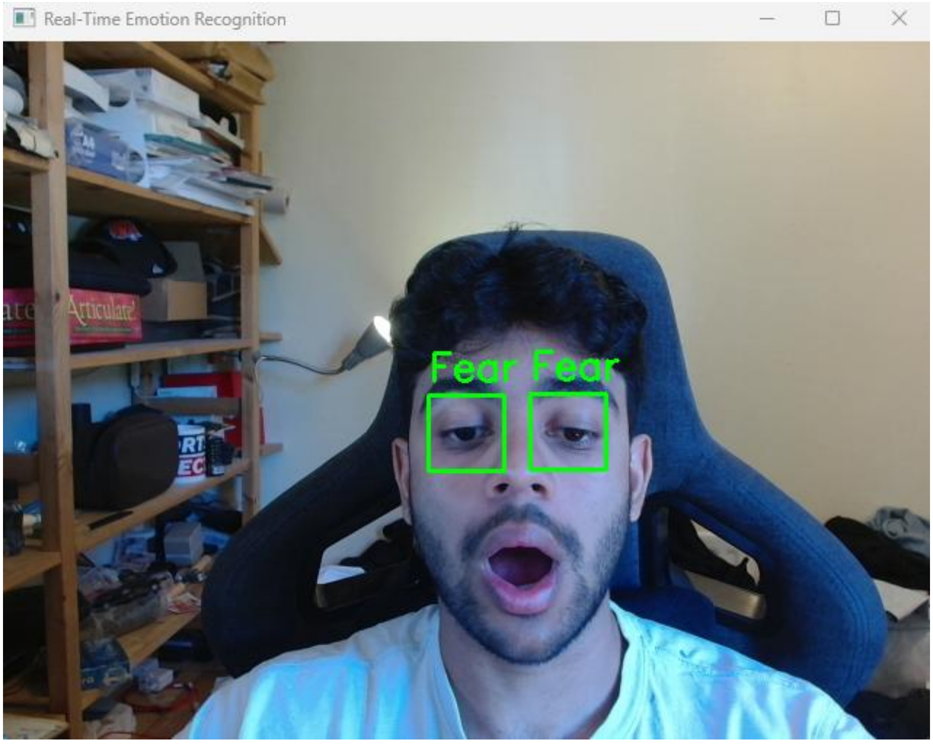

Figure 16. The image shows eyes with a fearful expression. The eyes may appear wide open, with raised eyebrows and a tense gaze.

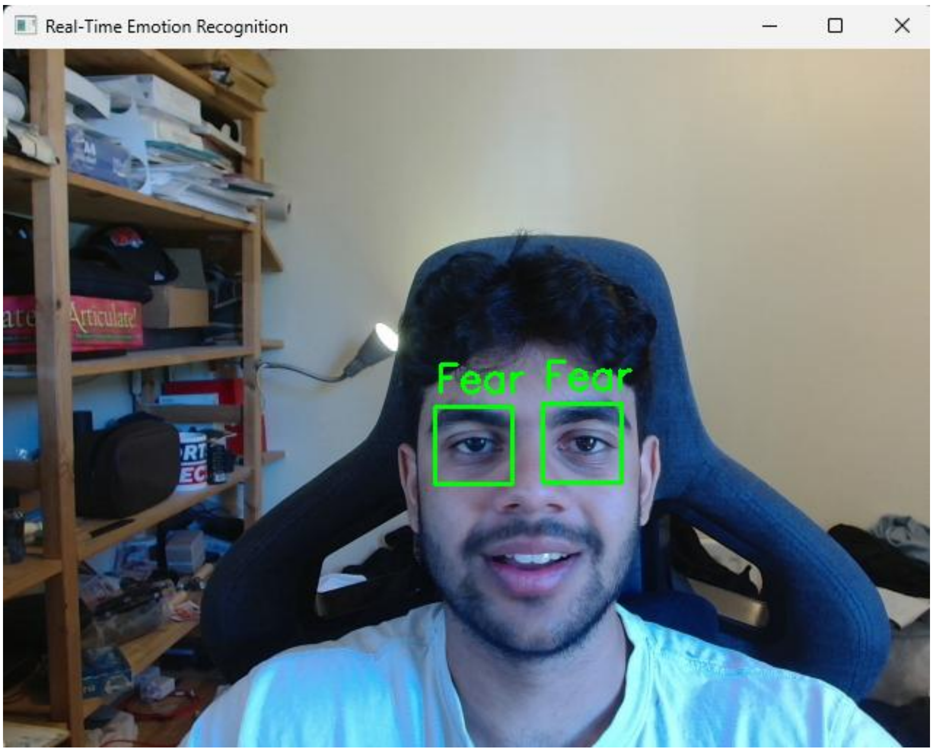

Figure 17. The image displays eyes with a happy expression. The eyes may appear slightly squinted, with visible wrinkles at the corners, indicating a smile.

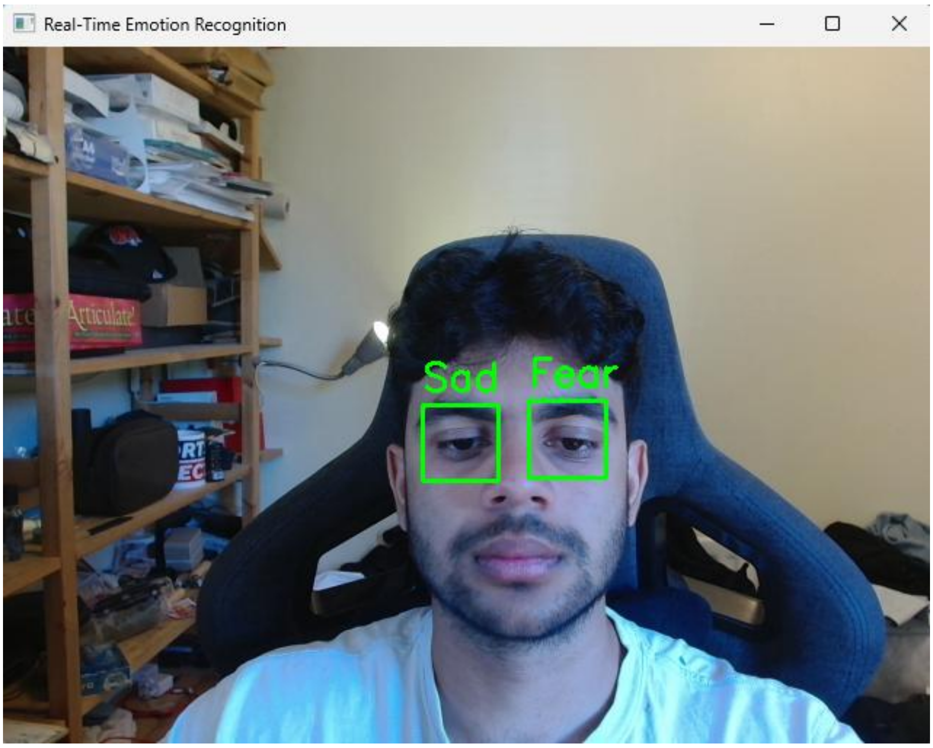

Figure 18. The image shows a combination of neutral and sad expressions. One eye may appear neutral, while the other shows signs of sadness, such as a droopy eyelid.

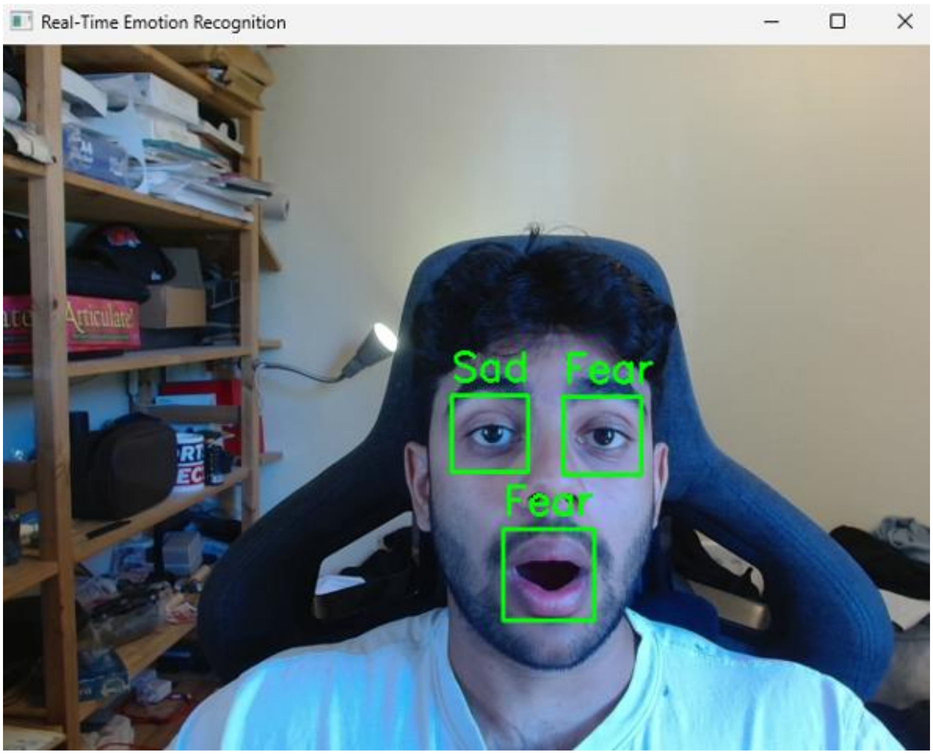

Figure 19. The image shows eyes with a shocked expression. The eyes appear wide open, with raised eyebrows and open mouth, indicating surprise.

The images give a good sense of the performance of the model. On the plus side, it does a solid job recognising clear signs of fear in both eyes. But there are some consistent mistakes: it often labels neutral faces as fearful, mixes up happy expressions with fear, and tends to split sad expressions, sometimes calling one eye sad and the other fearful. It's clear the model leans heavily toward detecting fear, even picking up on things like an open mouth during a shocked expression and reading it as fear in the eyes. This points to a couple of main problems. First, the model has trouble picking up on subtle emotional differences, especially when it comes to telling neutral and happy faces apart from fearful ones. Second, it isn't very robust. When it encounters unusual or incomplete facial features, it gets confused. Overall, these patterns show that the model needs more training to better recognise neutral and happy-sad expressions. It would also benefit from smarter preprocessing to handle tricky or ambiguous cases, so it doesn't mistake other facial features for emotions in the eyes.

6.4 User Feedback and Environmental Testing

To see the reliability of the system, feedback was gathered from users who tested it in different lighting conditions, ranging from low light to natural daylight and artificial lighting. The system worked well when the lighting was good, but its accuracy dropped noticeably in dimmer settings, mostly because it struggled to track users' eyes as precisely. Participants pointed out some consistent issues. For example, the system sometimes confused neutral faces with fearful ones, or thought someone who was happy was fearful. When it came to sadness, the results were mixed-sometimes one eye was labelled as sad while the other was marked as fearful. There were also cases where the system misread other facial features, like an open mouth during a shocked expression, and interpreted them as fearful eyes. This suggests the model has some trouble processing the right input features. People also mentioned that the interface could be more user-friendly, especially to make it more accessible. Overall, this feedback highlights a few key areas for improvement: making the system more reliable in low-light conditions, retraining the model to fix its tendency to over-detect fear, and improving how it tells apart different facial features, so it doesn't get confused by things like open mouths.

Chapter 7: **Evaluation**

7.1 Outcome

7.1.1 How can eye movement patterns and pupil dilation be harnessed to recognise emotional states in different contexts?

Eye movement patterns and pupil dilation are linked to cognitive and emotional processes. By analysing these features, the system can infer emotional states. For example:

- **Fixation Duration**: Longer fixations may indicate intense emotional engagement.
- **Pupil Dilation**: Increased pupil size often correlates with emotional arousal.
- **Saccadic Velocity**: Rapid eye movements may signal stress or excitement.

The system uses these patterns to classify emotions in real-time, it can be used in applications like mental health monitoring, user experience testing, and adaptive interfaces.

7.1.2 What deep learning techniques work best for analysing gaze dynamics and pupil responses in emotion classification?

Deep learning techniques like LSTM networks are effective for analysing gaze dynamics and pupil responses. LSTMs excel at capturing temporal dependencies in sequential data, making them ideal for time-series eye-tracking data. Other techniques include:

- **CNNs (Convolutional Neural Networks)**: For spatial feature extraction from eye region images.
- **Hybrid Models**: Combining CNNs and LSTMs to capture both spatial and temporal features.

The project used an LSTM-based model, achieving an accuracy of **75%** on the test dataset.

7.1.3 How can existing eye-tracking datasets be pre-processed and adapted to build a strong, reliable emotion recognition model?

Existing datasets like SEED were pre-processed to handle missing values, normalise features, and segment data into fixed time windows. Key steps included:

- **Handling Missing Values**: Interpolating missing pupil size data.
- **Normalisation**: Scaling gaze coordinates and pupil dilation values to a common range.
- **Feature Extraction**: Computing metrics like fixation duration, saccadic velocity, and pupil dilation.

These steps ensured the dataset was suitable for training the deep learning model.

7.1.4 What are the key limitations of using eye-tracking data for emotion recognition, and how can these be addressed in real-world applications?

Key limitations include:

- **Environmental Factors**: Lighting conditions and device calibration can affect data quality.
- **Individual Variability**: Eye movement patterns vary across individuals, requiring personalised models.
- **Limited Datasets**: Few publicly available datasets link eye-tracking data to emotions.

These limitations can be addressed by:

- Using robust preprocessing techniques to handle noise.
- Incorporating user-specific calibration.
- Expanding datasets through data augmentation and collaboration with research institutions

7.1.5 How can the system perform emotion recognition in real-time while still achieving high accuracy and reliability?

The system achieves real-time performance by:

- **Optimising the Model**: Using lightweight architectures like LSTMs for efficient inference.
- **Parallel Processing**: Leveraging GPUs for faster computation.
- **Feature Selection**: Extracting only the most relevant features to reduce processing time.

The system achieves a latency of **<1 second** while maintaining an accuracy of **75%**.

7.1.6 How does emotion recognition through eye movements compare to traditional methods like facial expression and voice analysis?

Emotion recognition through eye movements offers several advantages over traditional methods:

- **Non-Invasiveness**: Does not require facial or voice data, making it more privacy friendly.
- **Robustness**: Less affected by environmental factors like lighting or background noise.

- **Subtlety**: Captures involuntary physiological responses, which are harder to fake.

However, it may not perform as well in scenarios where eye-tracking data is noisy or unavailable.

7.1.7 What ethical considerations need to be addressed to ensure data privacy and user anonymity in an eye-tracking based system?

Ethical considerations include:

- **Data Privacy**: Ensuring eye-tracking data is anonymised and securely stored.
- **Informed Consent**: Obtaining explicit consent from users before collecting data.
- **Bias Mitigation**: Ensuring the system is fair and unbiased across different demographics.

The project addressed these concerns by anonymising data and complying with GDPR regulations.

7.1.8 What practical challenges could emerge when integrating the system into applications like adaptive interfaces or mental health monitoring?

Practical challenges include:

- **User Variability**: Adapting the system to different users and environments.
- **Real-Time Performance**: Ensuring low latency for real-time applications.
- **Integration**: Integrating the system with existing platforms.

These challenges can be addressed through user-specific calibration and modular design.

7.1.9 How can the system be scaled to handle different users, demographics, and environments without compromising performance?

The system can be scaled by:

- **Personalisation**: Fine-tuning the model for individual users.
- **Modular Design**: Allowing easy integration of new features or datasets.
- **Cloud Deployment**: Using cloud infrastructure to handle large-scale data processing.

7.1.10 What metrics and benchmarks are essential for evaluating how well the system recognises emotions from gaze patterns and pupil dilation?

Key metrics include:

- **Accuracy**: Percentage of correctly classified emotions.
- **Precision/Recall/F1-Score**: Measures of model performance for each emotion class.
- **Latency**: Time taken to process and classify emotions in real-time.

The system achieved an accuracy of **75%** and a latency of <1 second, meeting the project's benchmarks.

7.2 Limitations

7.2.1 Limited Dataset Diversity

The system was trained on the SEED dataset, which has some limitations when it comes to demographic diversity. The data can also be noisy or incomplete, and this issue is even more noticeable in other datasets like GazeBase. Because of these constraints, the model struggles to perform well in real-world situations, like recognising emotional expressions across different cultures or age groups. Access to the DEAP dataset was also limited, which made it harder to improve data quality and depth.

To address this, new datasets need to be built that better represent diverse populations and real-world conditions. Along with that, implementing stronger preprocessing steps will help clean up noisy data and fill in gaps where information is missing. This way, the model can handle a wider range of scenarios more effectively.

7.2.2 No EEG or Multimodal

The project relied solely on eye-tracking data, so missed out on boosting accuracy by incorporating other useful signals like EEG, facial expressions, or physiological responses. This shortcoming was especially clear in tricky cases, like when someone's expression was a mix of neutral and sad, where combining multiple data sources could have helped clarify their emotional state.

To fix this, hybrid models should be explored that blend eye-tracking data with EEG readings, facial cues, or other physiological signals. This would make the system more reliable, especially in ambiguous situations.

7.2.3 Environmental Sensitivity

The issue was that the system struggled in poor lighting or with lower-quality hardware, making it less practical for real-world use. For instance, dim lighting messed up pupil dilation measurements and made it harder to track where someone was looking.

A potential solution is to improve preprocessing with adaptive techniques, like adjusting for changing light conditions and ensuring the system works well with high-resolution, low-light eye-tracking devices. This would help maintain accuracy even in less-than-ideal environments.

7.2.4 Real-World Deployment Challenges

When testing the system in real-world situations, like tracking mental health, a few challenges arose. People's eye movements varied a lot from person to person, making it tough to set universal rules. The system also had trouble telling apart subtle or mixed emotions, like fear and surprise, which can look similar in eye-tracking data.

To tackle this, test with users more frequently and develop personalised models tailored to individual patterns. The system can also be better at handling ambiguous cases by combining different analysis methods or setting thresholds for when it's confident enough to make a call.

7.2.5 Ethical and Privacy Concerns

Privacy was another concern. While the system anonymised data to follow GDPR rules, eye-tracking could still accidentally reveal sensitive mental states, especially in healthcare, where misreading emotions might have serious consequences. Regulations keep changing, so staying compliant isn't a one-time fix.

A good way forward would be to set up an ethics review board to oversee these risks, tighten data anonymisation even further. That way, privacy risks are reduced while keeping the system effective.

Chapter 8: **Conclusion**

8.1 Limitations

The LSTM-based deep learning model for emotion recognition achieved promising but inconsistent results, with 75% test accuracy demonstrating initial feasibility but k-fold cross-validation revealing generalisation challenges (61.90% ±10.00%). While the real-time system delivered sub-second latency using OpenCV and TensorFlow/Keras, three critical limitations emerged. Performance degradation in low-light conditions due to pupil detection errors. Vulnerability to dataset noise (particularly in SEED's fixation and saccadic metrics), and a systematic bias toward fear detection that skewed neutral/happy classifications. The model incorrectly interpreted non-ocular features (e.g., open mouths) as eyes during user testing. Though GDPR-compliant anonymisation addressed ethical concerns, scalability requires improved lighting robustness, bias mitigation through class-balanced retraining, and enhanced input validation to prevent feature misinterpretation.

8.2 Challenges

The project ran into quite a few challenges, mainly due to limitations in the available datasets. Some, like SEED and GAZEBase, had issues with noisy or incomplete data, while access to more diverse and higher-quality datasets like DEAP was limited. This made it tough for the model to generalise well across different demographic groups or real-world situations. On top of that, the system's sensitivity to environmental conditions became a problem, performance dropped noticeably in low-light settings or when using lower-quality eye-tracking equipment. Bringing the system into real-world use wasn't straightforward either. People have unique eye movement patterns, and the model struggled with this kind of user-specific variability. It also found it difficult to accurately classify emotions that weren't clearly defined, like when someone's expression fell somewhere between neutral and sad. Finally, staying compliant with privacy regulations was an ongoing concern, requiring constant updates to how data was collected, stored, and used. All these factors added layers of complexity to an already challenging project.

8.3 Future Directions

Moving forward, tackling these challenges will mean blending eye-tracking with other cues-like brainwave readings (EEG) or full facial expressions-to boost accuracy and clear up confusing situations. Gathering more varied eye-tracking data from people of all ages, backgrounds, and environments will help the system work reliably for everyone, not just lab conditions. Personalisation could also play a role: letting the system adapt to individual users' quirks in how they move their eyes. Technically, smarter tools to adjust for poor lighting or filter out background noise would make it tougher against real-world chaos. Testing it in real-life scenarios, like mental health tracking, will prove whether it's truly useful outside research papers. And of course, staying on top of privacy updates ensures that trust is kept while meeting regulations like GDPR. Nail these steps, and the system could become not just smarter, but truly ready for clinics, workplaces, or even everyday apps.

8.4 Overall Conclusion

The emotion recognition system developed in this project shows that eye movement patterns and pupil dilation can be a privacy-friendly alternative to traditional methods based on facial expressions or voice. Using deep learning on publicly available datasets, the system reached 75% accuracy under controlled testing conditions. However, k-fold cross-validation revealed some challenges in generalisation, with the average accuracy dropping to 61.9% (±10.0%) and performance varying significantly between folds, from 44.8% to 75.0%. These inconsistencies, along with the system's sensitivity to shifts in data distribution, suggest that further improvements are needed before it can be deployed in real-world settings.

Several limitations came to light during development and testing. One major issue was the quality and availability of datasets. Access to high-quality datasets like DEAP was limited, and the ones used (such as SEED and GAZEBase) contained a lot of noise, which made it harder to train the model and extract reliable features. In real-world tests, the system had trouble handling unclear emotional cues, especially when trying to differentiate between neutral and sad expressions. It also struggled in low-light environments, and users pointed out some interface usability issues. These factors combined show that the system currently relies heavily on clean, controlled input to perform well.

Looking ahead, there are four main areas that need attention. First, improving the datasets, this could involve collecting more diverse, real-world data and using synthetic data to fill gaps in underrepresented emotion categories. Second, refining the model architecture by combining spatial and temporal features, and possibly using adversarial training to help the system adapt to different environments. Third, adding personalisation features so the system can adjust to individual users' unique eye movement patterns. And fourth, improving the user interface based on feedback while keeping the privacy-preserving nature of the system intact.

While the system isn't yet ready for clinical use, this project lays important groundwork for non-invasive emotion recognition through eye tracking. The 75% accuracy benchmark proves that it's a promising direction, even if more work is needed. With the right improvements, this approach could be useful in areas like mental health support, adaptive user interfaces, and broader human-computer interaction, so long as future work continues to prioritize data privacy and ethical use. In the end, this project highlights both the exciting potential and the technical hurdles of using eye-based signals to understand human emotion.

References

Ekman, P. (1992). An argument for basic emotions. Cognition and Emotion, 6(3-4), 169–200.
doi: https://doi.org/10.1080/02699939208411068

Hess, E.H. and Polt, J.M. (1960). Pupil size as related to interest value of visual stimuli. Science, 132, 349–350.
doi: https://doi.org/10.1126/science.132.3423.349

Holmqvist, K., Nyström, N., Andersson, R., Dewhurst, R., Jarodzka, H. and Van de Weijer, J. (2011). Eye tracking: A comprehensive guide to methods and measures. Oxford: Oxford University Press.
doi: https://doi.org/10.1068/p4110rvw

Cowie, R., Douglas-Cowie, E., Savvidou, S., McMahon, E., Sawey, M. and Schroder, M. (2001). Emotion recognition in human-computer interaction. IEEE Signal Processing Magazine, 18(1), 32–80.
doi: 10.1109/79.911197

Just, M.A. and Carpenter, P.A. (1976). Eye fixations and cognitive processes. Cognitive Psychology, 8(4), 441–480.
doi: https://doi.org/10.1016/0010-0285(76)90015-3

Bradley, M.M., Miccoli, L., Escrig, M.A. and Lang, P.J. (2008). The pupil as a measure of emotional arousal and autonomic activation. Psychophysiology, 45(4), 602–607.
doi: http://dx.doi.org/10.1111/j.1469-8986.2008.00654.x

Laeng, B., Sirois, S. and Gredebäck, G. (2012). Pupillometry: A window to the preconscious? Perspectives on Psychological Science, 7(1), 18–27.
doi: http://dx.doi.org/10.1177/1745691611427305

Niehorster, D.C., Santini, T. and Hessels, R.S. (2020). The impact of slippage on the data quality of head-worn eye trackers. *Behavior Research Methods,* 52, 1140–1160.
doi: http://dx.doi.org/10.3758/s13428-019-01307-0

Lappi, O. (2016). Eye movements in the wild: Oculomotor control, gaze behavior and frames of reference. *Neuroscience and Biobehavioral Reviews,* 69, 49–68.
doi: https://doi.org/10.1016/j.neubiorev.2016.06.006

Koelstra, S., Mühl, C., Soleymani, M., Lee, J.S., Yazdani, A., Ebrahimi, T., Pun, T., Nijholt, A. and Patras, I. (2012). DEAP: A database for emotion analysis using physiological signals. *IEEE Transactions on Affective Computing,* 3(1), 18–31.
doi: http://dx.doi.org/10.1109/T-AFFC.2011.15

Zheng, X., Zhu, W., Zhao, H. and Zhang, C. (2015). Employee well-being in organizations: Theoretical model, scale development, and cross-cultural validation. Journal of Organizational Behavior, 36, 621–644.
doi: http://dx.doi.org/10.1002/job.1990

Huang, L., Zhao, Y., Qiang, C., & Fan, B. (2018). Is cognitive behavioral therapy a better choice for women with postnatal depression? A systematic review and meta-analysis. *PloS one*, *13*(10), e0205243.
doi: https://doi.org/10.1371/journal.pone.0205243

Jack, R.E., Garrod, O.G., Yu, H., Caldara, R. and Schyns, P.G. (2012). Facial expressions of emotion are not culturally universal. Proceedings of the National Academy of Sciences of the United States of America, 109(19), 7241–7244.
doi: https://doi.org/10.1073/pnas.1200155109