Homework III Classification Report

1. Dataset and Prediction Target Description

1.1 Problem Statement and Objective

I want to create a model to predict whether a user is watching the screen. I'm going to use MediaPipe face landmarker and blendshape features. MediaPipe face landmarker provides a transformation matrix that can be used to describe the user's head pose.

Blendshapes provide 52 high-level semantic features like browDownLeft, browDownRight, jawForward, etc. I will use 8 of them to train this model:

- eyeLookDownLeft
- eyeLookDownRight
- eyeLookUpLeft
- eyeLookUpRight
- eyeLookInLeft
- eyeLookInRight
- eyeLookOutLeft
- eyeLookOutRight

I will also use eyeBlinkLeft and eyeBlinkRight as filters. If the user's eyes are closed, I don't want to include those frames in the dataset.

1.2 Dataset Structure

I recorded several videos and labeled them as yes/no. Here's the structure of the dataset:

```
dataset
├── no
│   ├── 2025-08-31 18-50-30_mirrored.mp4
│   ├── 2025-08-31 18-50-30.mp4
│   ├── 2025-08-31 19-42-47_mirrored.mp4
│   ├── 2025-08-31 19-42-47.mp4
│   ├── 2025-10-18 21-13-27_mirrored.mp4
│   ├── 2025-10-18 21-13-27.mp4
│   ├── 2025-10-18 21-13-59_mirrored.mp4
│   ├── 2025-10-18 21-13-59.mp4
│   ├── 2025-10-18 21-14-27_mirrored.mp4
│   ├── 2025-10-18 21-14-27.mp4
│   ├── 2025-10-18 21-14-39_mirrored.mp4
│   ├── 2025-10-18 21-14-39.mp4
│   ├── 2025-10-18 21-24-30_mirrored.mp4
│   ├── 2025-10-18 21-24-30.mp4
│   ├── 2025-10-18 21-46-18_mirrored.mp4
│   ├── 2025-10-18 21-46-18.mp4
│   ├── 2025-10-18 22-19-04_mirrored.mp4
│   ├── 2025-10-18 22-19-04.mp4
│   ├── 2025-10-18 22-20-54_mirrored.mp4
│   ├── 2025-10-18 22-20-54.mp4
│   ├── 2025-10-18 22-25-49_mirrored.mp4
│   ├── 2025-10-18 22-25-49.mp4
│   ├── 2025-10-18 22-28-46_mirrored.mp4
│   └── 2025-10-18 22-28-46.mp4
└── yes
    ├── 2025-08-31 18-52-02_mirrored.mp4
    ├── 2025-08-31 18-52-02.mp4
    ├── 2025-08-31 19-41-47_mirrored.mp4
    ├── 2025-08-31 19-41-47.mp4
    ├── 2025-10-18 21-15-40_mirrored.mp4
    ├── 2025-10-18 21-15-40.mp4
    ├── 2025-10-18 21-24-57_mirrored.mp4
    ├── 2025-10-18 21-24-57.mp4
    ├── 2025-10-18 21-33-55_mirrored.mp4
    ├── 2025-10-18 21-33-55.mp4
    ├── 2025-10-18 21-34-03_mirrored.mp4
    ├── 2025-10-18 21-34-03.mp4
    ├── 2025-10-18 21-40-51_mirrored.mp4
    ├── 2025-10-18 21-40-51.mp4
    ├── 2025-10-18 21-41-15_mirrored.mp4
    ├── 2025-10-18 21-41-15.mp4
    ├── 2025-10-18 21-46-12_mirrored.mp4
    ├── 2025-10-18 21-46-12.mp4
    ├── 2025-10-18 21-47-13_mirrored.mp4
    ├── 2025-10-18 21-47-13.mp4
    ├── 2025-10-18 22-32-12_mirrored.mp4
    └── 2025-10-18 22-32-12.mp4

3 directories, 46 files
```

1.3 Data Augmentation

To augment the dataset and improve model generalization, I applied horizontal mirroring to all videos using ffmpeg. This effectively doubles the dataset size and helps the model learn from different orientations.

I will process these videos by extracting every 5th frame, and extract transformation matrix and blendshape features from them. After filtering out frames with closed eyes and applying data augmentation:

- Total samples: 4,654
- Yes samples: 2,500
- No samples: 2,154

2. Building the Classifier

2.1 Choosing the Classification Model

Now I'm going to train an SVM, but there are several types of SVM kernels:

1. 'linear' - Linear kernel
    - Formula: K(x, x') = x^T x'
    - Best for linearly separable data
    - Fast training and prediction
    - Good for high-dimensional sparse data
2. 'rbf' (Radial Basis Function) - DEFAULT
    - Formula: $K(x, x') = \exp(-\gamma ||x - x'||^2)$
    - Most popular kernel
    - Good for non-linear data
    - Parameters: gamma (kernel coefficient)
3. 'poly' - Polynomial kernel
    - Formula: K(x, x') = (γ x^T x' + coef0)^degree
    - Good for polynomial relationships
    - Parameters: degree, gamma, coef0
4. 'sigmoid' - Sigmoid kernel
    - Formula: K(x, x') = tanh(γ x^T x' + coef0)
    - Similar to neural network activation
    - Parameters: gamma, coef0
5. 'precomputed' - Pre-computed kernel matrix
    - You provide your own kernel matrix
    - Advanced use case

I know the features have non-linear relationships because of the compensation effect between head pose and eye gaze. For example, when the head turns right but the user is still watching the screen, the left eye looks outward while the right eye looks inward to compensate for the head rotation. This means the same eye gaze values can indicate different states depending on the head pose, which is a typical feature interaction that cannot be captured by linear models.

Therefore, I will use the RBF (Radial Basis Function) kernel for SVM training, which is well-suited for capturing non-linear relationships. I will perform hyperparameter tuning to find the optimal combination of `C` (regularization parameter) and `gamma` (kernel coefficient) that achieves the maximum accuracy.

2.2 Training Plan

Here's the training plan:

- Load all_videos_combined.csv dataset
    - Filter out closed-eye frames (eyeBlinkLeft/Right > threshold)
    - Use features: 8 eye gaze + 3 head rotation (11 features total)
    - Split data: training/testing sets (80% train, 20% test)
    - Focus on RBF kernel only
    - Hyperparameter tuning using GridSearchCV:
        - C: Regularization parameter (e.g., [0.1, 1, 10, 100, 1000])
        - gamma: Kernel coefficient (e.g., [0.001, 0.01, 0.1, 1, 'scale', 'auto'])
    - Find best combination with maximum accuracy
    - Evaluate best model with 7 metrics:
        - Confusion Matrix
        - Accuracy
        - Precision
        - Recall
        - F1 Score
        - ROC Curve
        - AUC
    - Save best model and results
    - Generate visualizations

3. Model Evaluation with Seven Metrics

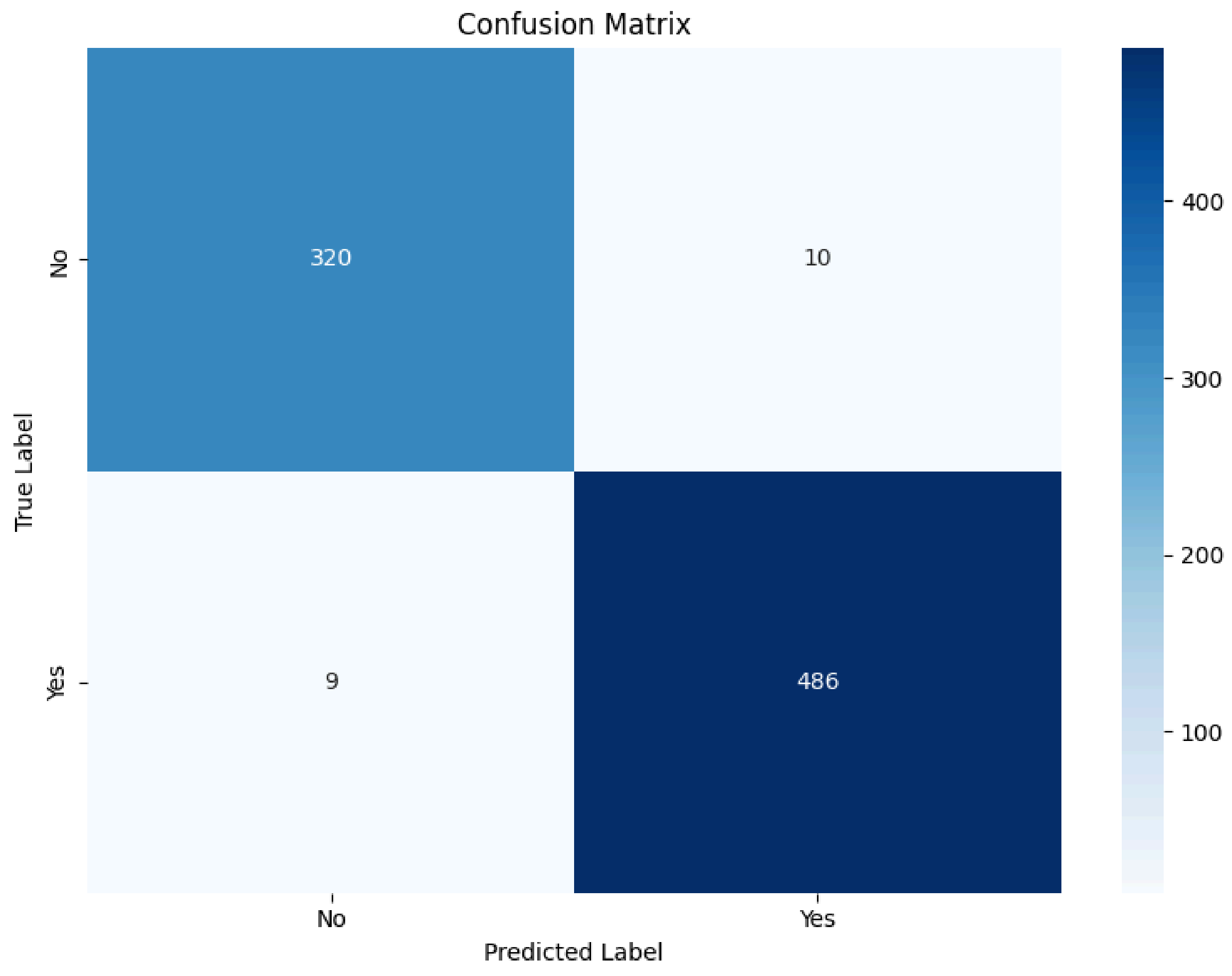
Here's the training results:

```
============================================================
2025-10-19 17:09:18,038 - INFO - Test Set Evaluation Metrics:
2025-10-19 17:09:18,038 - INFO -
============================================================
2025-10-19 17:09:18,039 - INFO -
1. Confusion Matrix:
[[320  10]
 [  9 486]]
2025-10-19 17:09:18,161 - INFO - Confusion matrix saved to svm_results/confusion_matrix.png
2025-10-19 17:09:18,161 - INFO -
2. Accuracy: 0.9770
2025-10-19 17:09:18,162 - INFO - 3. Precision: 0.9798
2025-10-19 17:09:18,163 - INFO - 4. Recall: 0.9818
2025-10-19 17:09:18,164 - INFO - 5. F1 Score: 0.9808
2025-10-19 17:09:18,165 - INFO - 6. ROC Curve: Saved to file
2025-10-19 17:09:18,165 - INFO - 7. AUC: 0.9964
2025-10-19 17:09:18,256 - INFO - ROC curve saved to svm_results/roc_curve.png
```

```
2025-10-19 17:09:18,256 - INFO -
Detailed Classification Report:
2025-10-19 17:09:18,259 - INFO -
              precision    recall  f1-score   support

          No       0.97      0.97      0.97       330
         Yes       0.98      0.98      0.98       495

    accuracy                           0.98       825
   macro avg       0.98      0.98      0.98       825
weighted avg       0.98      0.98      0.98       825

2025-10-19 17:09:18,260 - INFO -
Metrics saved to svm_results/metrics.json
2025-10-19 17:09:18,261 - INFO - Model saved to svm_results/best_svm_model.pkl
2025-10-19 17:09:18,261 - INFO - Scaler saved to svm_results/scaler.pkl
2025-10-19 17:09:18,261 - INFO -
============================================================
2025-10-19 17:09:18,261 - INFO -
============================================================
2025-10-19 17:09:18,261 - INFO - Training completed successfully!
2025-10-19 17:09:18,261 - INFO -
============================================================
```

The model achieved excellent results with 97.7% accuracy on the test set.

3.1 Confusion Matrix



The confusion matrix shows:

- True Negatives (No predicted as No): 320
- False Positives (No predicted as Yes): 10
- False Negatives (Yes predicted as No): 9
- True Positives (Yes predicted as Yes): 486

3.2 Accuracy

Accuracy: 0.9770 (97.7%)

This means the model correctly classified 97.7% of all test samples.

3.3 Precision

Precision: 0.9798 (97.98%)

This means when the model predicts "Yes" (watching screen), it's correct 97.98% of the time.

3.4 Recall

Recall: 0.9818 (98.18%)

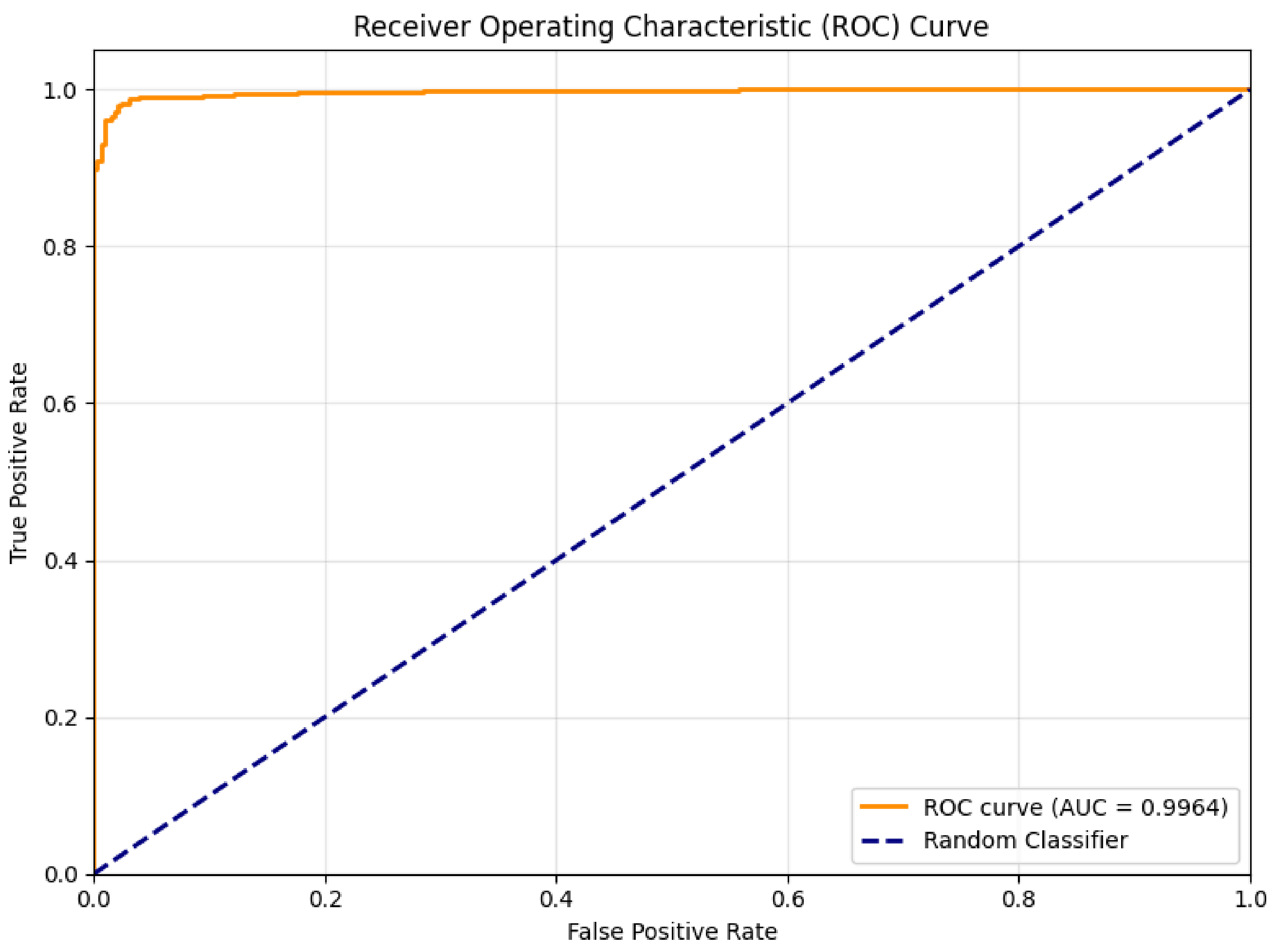
This means the model can identify 98.18% of all actual "Yes" cases.

3.5 F1 Score

F1 Score: 0.9808 (98.08%)

F1 Score is the harmonic mean of precision and recall, showing balanced performance.

3.6 ROC Curve



The ROC curve shows the trade-off between true positive rate and false positive rate.

3.7 AUC (Area Under the ROC Curve)

AUC: 0.9964 (99.64%)

An AUC of 99.64% indicates excellent discriminative ability of the classifier.

4. Real-World Testing

I was curious whether it was overfitting, so I asked AI to write a real-time frame prediction program.

I recorded a video to demo the SVM, and it works pretty well:

Demo Video

The real-time testing shows that the model generalizes well and is not overfitting to the training data.

5. Conclusion

I successfully built a gaze detection classifier using SVM with RBF kernel. The model uses 11 features (8 eye gaze features and 3 head rotation features) extracted from MediaPipe face landmarker. After data augmentation with horizontal mirroring, the dataset contains 4,654 samples.

The model achieved excellent performance on all seven evaluation metrics:

- Accuracy: 97.7%
- Precision: 97.98%
- Recall: 98.18%
- F1 Score: 98.08%
- AUC: 99.64%

Real-world testing with webcam confirmed that the model works well in practice and is not overfitting.