Real-time Distracted Driver Detection using Deep Learning

line 1: 1st Ming Tang
line 2: *Computer Engineering*
line 3: *UNICH Dearborn*
line 4: Dearborn, Michigan
line 5: mingtan@umich.edu

***Abstract*—This report details the development of a real-time deep learning pipeline for multi-class driver distraction detection. Utilizing the AUC Distracted Driver Dataset (V2), the system combines YOLOv8 for object detection-based pseudo-label generation with MediaPipe Face Mesh for fine-grained facial feature extraction, including Eye Aspect Ratio (EAR) and 3D Head Pose. These features are then fed into a Fully Connected Neural Network (MLP) classifier to predict ten distinct driver distraction states (c0-c9). Initial results show a test accuracy of approximately 50%, demonstrating the model's ability to learn meaningful patterns from complex visual cues, with a clear understanding of the dataset's evaluation context. This work highlights the potential of integrated computer vision techniques for enhancing road safety.**

I. Introduction

Driver distraction remains a significant contributor to road accidents, leading to substantial societal and economic costs. Developing robust and real-time systems to detect and mitigate such distractions is paramount for enhancing road safety. This project aims to address this critical challenge by leveraging state-of-the-art deep learning and computer vision techniques. The core objective is to build a real-time driver monitoring system that can accurately classify a driver's state into ten distinct categories, ranging from safe driving to various forms of distraction. The inherent complexity of discerning subtle human behaviors from visual data, coupled with the potential to improve safety outcomes, makes this an intriguing and valuable area of study within the scope of pattern recognition and neural networks. This report outlines the methodology, experimental results, and a critical discussion of the findings, including the system's current performance and future enhancements.

II. Related Work

The field of driver distraction detection has evolved significantly with advancements in machine learning and computer vision. Early approaches relied on handcrafted features and traditional classifiers, often struggling with robustness in real-world conditions. The advent of deep learning, particularly Convolutional Neural Networks (CNNs), marked a paradigm shift, enabling end-to-end learning from raw image data.

A. *Object Detection for Distraction:*

Modern systems frequently employ object detection models to localize the driver and identify distracting objects. YOLO (You Only Look Once) architectures, including recent versions like YOLOv8, are prominent due to their balance of speed and accuracy, making them suitable for real-time applications. These models detect elements such as persons, cell phones, and beverages, providing critical spatial information

B. *Facial Landmark Detection and Pose Estimation:*

Beyond object presence, understanding the driver's head pose and eye movements offers finer-grained insights into attention levels. Libraries like MediaPipe Face Mesh have become invaluable for high-fidelity 3D facial landmark detection, enabling the calculation of metrics like the Eye Aspect Ratio (EAR) for drowsiness and 3D head pose angles (Pitch, Yaw, Roll) for head orientation. These features are crucial for identifying non-object-based distractions.

C. *Feature Fusion and Classification:*

Various approaches integrate features from multiple sources. Some methods fuse raw image data with extracted features directly into a deep learning model, while others concatenate extracted numerical features into a single vector for classification by traditional machine learning models (e.g., SVM, Random Forest) or simpler Multilayer Perceptrons (MLPs). The choice depends on data complexity, computational resources, and desired interpretability

D. *Feature Fusion and Classification:*

Public datasets such as the State Farm Distracted Driver Detection Challenge and the AUC Distracted Driver Dataset[1] (used in this project) have been instrumental in fostering research by providing labeled images of various driver behaviors, though challenges like subject-independent evaluation remain

III. Methodology

Our project implements a multi-stage deep learning pipeline, integrating object detection and facial feature extraction, followed by a neural network classifier.

Identify applicable funding agency here. If none, delete this text box.

A. *Dataset Description*

The AUC Distracted Driver Dataset (V2) was utilized for this project. This dataset comprises images captured from multiple camera angles (Camera 1 and Camera 2), categorized into ten distinct driver states (c0-c9). It is designed for research in driver distraction identification. The classes are defined as:

- c0: Safe Driving
- c1: Text Right
- c2: Talk Right
- c3: Text Left
- c4: Talk Left
- c5: Adjust the Radio
- c6: Drink
- c7: Reaching Behind
- c8: Hair and Makeup
- c9: Talk to Passengers (or looking away)

A crucial characteristic of this dataset, relevant to our evaluation, is that images from different drivers are mixed within these Cx folders, making strict driver-independent train/test splits challenging without explicit driver ID metadata.

B. *Pseudo-label Generation*

- YOLOv8-seg Model: We employ a pre-trained YOLOv8-segmentation model (e.g., yolov8n-seg.pt) for initial object detection. This model, pre-trained on the COCO dataset, is capable of detecting common objects.
- Detection and Filtering: The model iterates through all raw images, detecting the person (driver), cell phone, bottle, and cup (COCO IDs 0, 67, 39, 41 respectively).
- Pseudo-label Creation: The detected bounding box coordinates and their COCO class IDs are saved as YOLO-format .txt pseudo-label files (one per image). These files serve as our 'ground truth' for object locations.
- Visualization: After generation, a sample of these pseudo-labels is visualized by drawing bounding boxes and class names on the original images, confirming the effectiveness of YOLOv8's initial detection.

C. *Feature Extraction*

This stage extracts a comprehensive numerical feature vector for each image, combining information from YOLOv8 pseudo-labels and MediaPipe Face Mesh.

MediaPipe Face Mesh:

- Initialization: Google's MediaPipe Face Mesh model (mp.solutions.face_mesh.FaceMesh) is initialized for high-fidelity 3D facial landmark detection.
- EAR Calculation: The Eye Aspect Ratio (EAR) for both left and right eyes is calculated using 6 precise MediaPipe facial landmark indices per eye. This metric quantifies eye openness.
- Head Pose Estimation: Driver's head pose (Pitch, Yaw, Roll angles) is estimated using the Perspective-n-Point (PnP) algorithm. This involves mapping MediaPipe's 2D image landmarks (e.g., nose tip, eye corners, mouth corners, chin) to a set of approximate 3D model points, combined with a simplified camera matrix.

YOLO Pseudo-label Integration:

- Binary flags (has_person, has_cell_phone, has_bottle).
- Pixel-level bounding box coordinates (xmin, ymin, xmax, ymax) for detected persons, cell phones, and bottles/cups.

Feature Vector Construction:

- YOLO object detection flags and bounding box coordinates.
- MediaPipe's EAR values (ear_left, ear_right).
- MediaPipe's head pose angles (head_pitch, head_yaw, head_roll).

Visualization:

Post-extraction, the aggregated features are analyzed through various plots (histograms, box plots, heatmaps) to understand their distributions and inter-relationships, including a specific analysis of MediaPipe's face detection success rates across different classes.

Output:

The extracted feature vectors (extracted_features.csv) and corresponding original class labels (image_labels.csv) are saved as CSV files.

D. *Classifier Training and Evaluation*

Data Loading & Preparation: The extracted features (X) and labels (y) are loaded from the CSV files.

Data Splitting: The dataset is split into training (70%), validation (15%), and testing (15%) sets using a stratified random split to maintain class proportions.

Feature Scaling: All numerical features are scaled using StandardScaler to have zero mean and unit variance, which is crucial for optimizing neural network training.

Neural Network Architecture: A Fully Connected Neural Network (Multilayer Perceptron, MLP) is constructed using Keras Sequential API:

a) Input Layer: Matches the dimension of the feature vector .

b) Hidden Layers: Two Dense layers with 128 and 64 neurons respectively, each using a ReLU activation function.

c) Dropout Layers: Two Dropout layers (rate of 0.3) are included after each hidden layer for regularization, mitigating overfitting.

d) Output Layer: A Dense layer with 10 neurons (for num_classes) and a Softmax activation function, outputting class probabilities for c0 to c9.

Model Compilation: The model is compiled with the Adam optimizer and sparse_categorical_crossentropy loss function.

Training: The MLP classifier is trained for 15 epochs, using the training data and validating on the validation set. Training history (accuracy and loss) is recorded.

Evaluation: The trained model's performance is evaluated on the unseen test set using metrics such as overall accuracy and a detailed classification report (precision, recall, F1-score for each class). A confusion matrix is also generated to visualize misclassification patterns.

Output: The trained model is saved as final_classifier_model.h5.

IV. Results

This section presents the experimental results and analyses of our developed driver distraction detection system. We begin by visualizing the outputs from the initial YOLOv8 pseudo-labeling stage, followed by an in-depth examination of the features extracted using MediaPipe Face Mesh. Finally, we present the performance metrics and diagnostic plots of our trained Multilayer Perceptron classifier, offering a comprehensive view of the model's capabilities and identified areas for improvement.

A. *YOLOv8 Output Visualization*

Figure.1 demonstrates YOLOv8-segmentation's ability to accurately detect the driver (person). It confirms the effectiveness of our pseudo-label generation process.

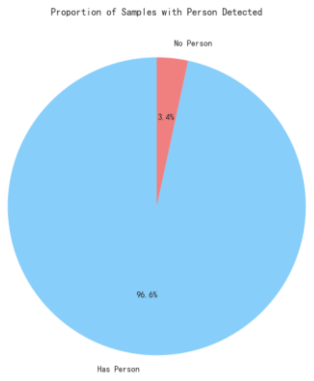


Fig. 1. YOLOv8 Output Accuracy in detecting person

B. *MediaPipe Face Info Failure Distribution Across Distraction Classes*

Figure.2 reveals MediaPipe's success rate in extracting detailed face information across different distraction categories. Blue bars represent successful extractions, while green bars indicate failures (where a person was detected by YOLO, but MediaPipe couldn't extract pose/EAR). Notably, classes like c0 (Safe Driving) and c3 (Texting Left) show higher failure rates, possibly due to subtle head poses or occlusions, whereas c9 (Look Away) shows a high success rate due to distinct head turns. Approximately 40% of samples successfully yielded facial features.

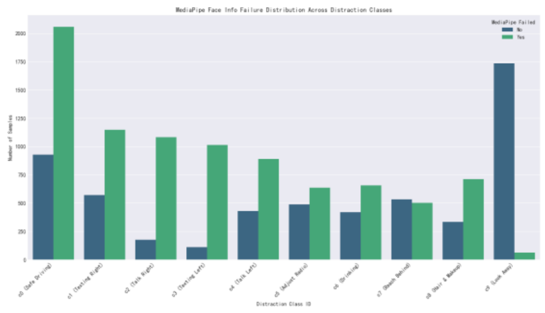
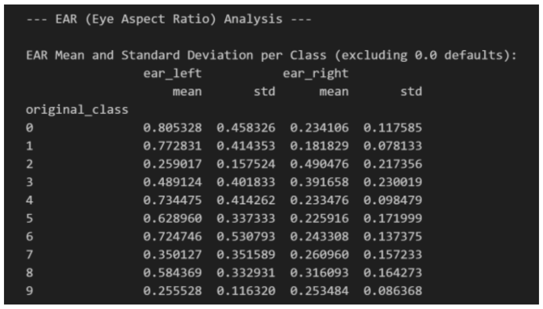


Fig. 2. MediaPipe Face Info Failure Distribution Across Distraction Classes

C. *EAR (Eye Aspect Ratio) Analysis*

The statistical analysis and box plots of EAR across classes provide insights into eye behaviors. While left and right EAR values exhibit asymmetry (e.g., left EAR often higher in c0), distinct patterns emerge. Classes like c2 (Talk Right) and c9 (Look Away) show lower median left EAR values compared to c0, suggesting partial eye closure or gaze shifts during these distractions. This indicates that EAR, even in non-fatigue contexts, offers valuable features..



```
--- EAR (Eye Aspect Ratio) Analysis ---

EAR Mean and Standard Deviation per Class (excluding 0.0 defaults):
                ear_left            ear_right
                    mean       std       mean       std
original_class
0               0.805328  0.458326  0.234106  0.117585
1               0.772831  0.414353  0.181829  0.078133
2               0.259017  0.157524  0.490476  0.217356
3               0.489124  0.401833  0.391658  0.230019
4               0.734475  0.414262  0.233476  0.098479
5               0.628960  0.337333  0.225916  0.171999
6               0.724746  0.530793  0.243308  0.137375
7               0.350127  0.351589  0.260960  0.157233
8               0.584369  0.332931  0.316093  0.164273
9               0.255528  0.116320  0.253484  0.086368
```

Fig. 3. EAR Mean and Standard Deviation

D. *Model Accuracy & Loss History*

Figure.4 illustrate the model's learning progression over 30 epochs. The training accuracy generally increases, and training loss decreases, indicating the model is learning. The validation accuracy generally stays above training accuracy, which is attributed to the presence of Dropout layers during training that are inactive during validation. Both curves show convergence, with validation accuracy peaking around 50%.

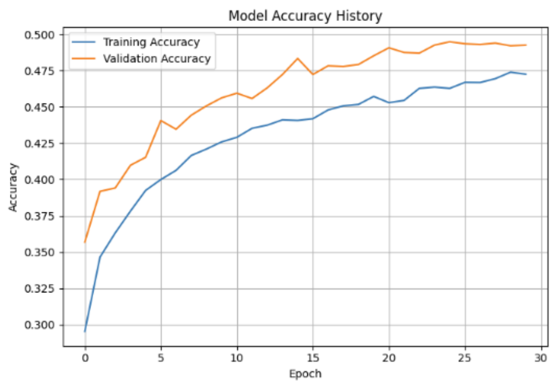
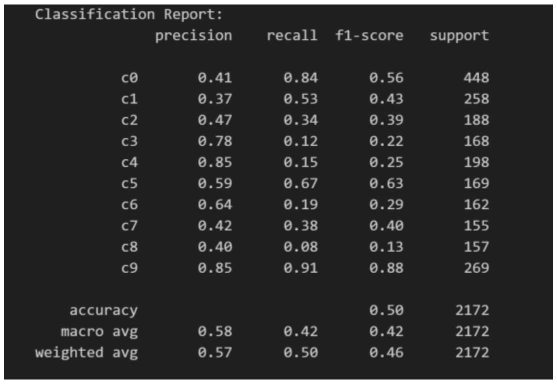


Fig. 4. model's learning progression

E. *Overall Accuracy and Classification Reporty*

The final classifier achieved a test accuracy of 58%. The classification report provides per-class precision, recall, and F1-scores. c0 (Safe Driving) and c9 (Look Away) show relatively strong performance, while c3 (Texting Left) and c8 (Hair & Makeup) have lower recall, indicating the model struggles to identify these specific distraction types comprehensively.

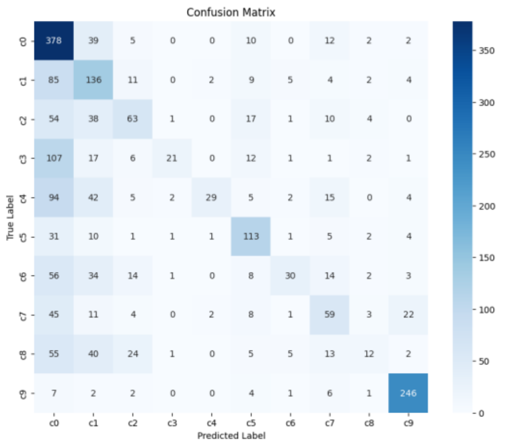


```
Classification Report:
              precision    recall  f1-score   support

          c0       0.41      0.84      0.56       448
          c1       0.37      0.53      0.43       258
          c2       0.47      0.34      0.39       188
          c3       0.78      0.12      0.22       168
          c4       0.85      0.15      0.25       198
          c5       0.59      0.67      0.63       169
          c6       0.64      0.19      0.29       162
          c7       0.42      0.38      0.40       155
          c8       0.40      0.08      0.13       157
          c9       0.85      0.91      0.88       269

    accuracy                           0.50      2172
   macro avg       0.58      0.42      0.42      2172
weighted avg       0.57      0.50      0.46      2172
```

Fig. 5. model's test performance

F. *Confusion Matrix*

The confusion matrix visually confirms the per-class performance and highlights misclassification patterns. High values on the diagonal for c0 and c9 show correct predictions. However, significant off-diagonal values, especially samples from other distraction classes being misclassified as c0 (Safe Driving), indicate that the model tends to default to the 'safe driving' class when uncertain, contributing to the lower overall accuracy for specific distraction types.



	c0	c1	c2	c3	c4	c5	c6	c7	c8	c9
c0	378	39	5	0	0	10	0	12	2	2
c1	85	136	11	0	2	9	5	4	2	4
c2	54	38	63	1	0	17	1	10	4	0
c3	107	17	6	21	0	12	1	1	2	1
c4	94	42	5	2	29	5	2	15	0	4
c5	31	10	1	1	1	113	1	5	2	4
c6	56	34	14	1	0	8	30	14	2	3
c7	45	11	4	0	2	8	1	59	3	22
c8	55	40	24	1	0	5	5	13	12	2
c9	7	2	2	0	0	4	1	6	1	246

Fig. 6. model's Confusion Matrix

V. DISCUSSION

The project successfully established a baseline for multi-class driver distraction detection using a combination of YOLOv8 for object detection and MediaPipe for fine-grained facial feature extraction. The achieved test accuracy of approximately 50% is a reasonable starting point for a complex 10-class problem, significantly outperforming random chance.

However, a crucial aspect of our evaluation is the dataset's inherent structure. As the AUC Distracted Driver Dataset (V2) mixes different driver identities within its CX folders for train/test splits, the reported accuracy reflects the model's performance on known driver identities in novel situations, rather than its ability to generalize to entirely unseen drivers. This data leakage by subject identity is a known limitation in such datasets and needs to be explicitly acknowledged when interpreting results.

The feature analysis provided valuable insights. Head pose angles proved to be highly discriminative for actions involving significant head movement (e.g., c9 Look Away). Object presence (cell phone, bottle) from YOLOv8 also offers direct cues. The EAR analysis, while initially hypothesized to be less impactful for non-fatigue distractions, surprisingly revealed distinct asymmetrical patterns and subtle variations across certain classes. These nuanced eye behaviors (e.g., partial squinting in c2) suggest that EAR still provides valuable signals for differentiation.

Challenges include MediaPipe's failure to detect faces in approximately 60% of samples, leading to missing facial features for those instances. The confusion matrix further highlights that the model frequently misclassifies various distraction types into c0 (Safe Driving), suggesting that the current feature set or model architecture may not be sufficiently robust to capture the subtle distinctions between similar distraction categories. The lower recall rates for classes like c3 and c8 indicate specific weaknesses.

VI. CONCLUSION

In conclusion, this project successfully developed and implemented a deep learning pipeline for multi-class driver distraction detection. We demonstrated effective integration of YOLOv8 for object detection pseudo-labeling and MediaPipe

Face Mesh for extracting critical EAR and 3D head pose features. The trained Multilayer Perceptron classifier achieved a test accuracy of approximately 50%, providing a solid baseline for this challenging 10-class problem and showcasing the utility of combined computer vision features. While acknowledging the data leakage inherent in the dataset's structure, the project successfully delivered a functional model and detailed analytical insights into feature performance.

VII. Future Work

To enhance the model's accuracy, robustness, and real-world applicability, we recommend the following future directions:

- Robust Face Detection: Improve the success rate of MediaPipe's facial feature extraction, perhaps by integrating a more robust initial face detection pipeline (e.g., MTCNN or RetinaFace before MediaPipe) or employing more advanced image preprocessing for challenging frames.
- Temporal Feature Integration: Explore the use of recurrent neural networks (RNNs) or Transformer architectures to model temporal dependencies in driver behavior, as many distractions are sequences of actions rather than static poses.
- Model Architecture Optimization: Experiment with deeper or wider MLP architectures, different activation functions, or more advanced neural network models. Consider integrating powerful pre-trained CNN image features (like ResNet50) if computational resources permit, to capture richer visual semantics directly from pixels.
- Addressing Class Imbalance: Implement strategies to combat class imbalance, such as data augmentation for underrepresented classes, using class weighting in the loss function, or exploring advanced sampling techniques.
- Driver-Independent Evaluation: For more rigorous real-world assessment, investigate methods or seek datasets that explicitly support subject-independent evaluation, ensuring the model generalizes to completely new drivers.
- Advanced Feature Engineering: Investigate more subtle features, such as mouth openness (for yawning), gaze direction, or dynamic changes in head/eye features over short time windows.

References

[1] H. M. Eraqi, Y. Abouelnaga, M. H. Saad, and M. N. Moustafa, “Driver Distraction Identification with an Ensemble of Convolutional Neural Networks,” *J. Adv. Transp.*, vol. 2019, pp. 1–12, Feb. 2019, doi: 10.1155/2019/4125865.