YAWN DETECTION USING SUPPORT VECTOR MACHINE CLASSIFICATION

An Undergraduate Research Scholars Thesis

by

SARVESH MAYILVAHANAN

Submitted to the Undergraduate Research Scholars Program at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by Research Advisor: Dr. Anxiao Jiang

May 2020

Major: Mechanical Engineering

TABLE OF CONTENTS

ABSTRACT

Yawn Detection Using Support Vector Machine Classification

Sarvesh Mayilvahanan
Department of Mechanical Engineering
Texas A&M University

Research Advisor: Dr. Anxiao Jiang
Department of Computer Science
Texas A&M University

A model to detect yawning in a video sequence from a regular camera is proposed. Landmark detectors are able to quickly and accurately estimate the pose of a face with various amounts of light and facial expressions. We can show that the information from these landmark detectors is useful in being able to detect when a person is yawning. The proposed model is therefore able to take in information from the landmark detector and process it by creating new features to characterize the yawning in each frame. A support vector machine classifier then detects yawning as a pattern of the constructed feature values in a short temporal window. We made a dataset and annotated it to train and test the model, which is composed of frames from several videos that include a person yawning and not yawning at different angles to the camera.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Jiang, for his guidance and support throughout the course of my research.

Thanks also go to the graduate students in Dr. Jiang's lab for their help with various issues that I encountered along the process of my research.

Finally, I would like to thank my family and friends for their support and encouragement.

CHAPTER I

INTRODUCTION

Problem

Being able to detect when a person is yawning is important and relevant to systems that monitor workers and ensure that drivers do not fall asleep at the wheel, among other useful applications. Driver fatigue is one of the leading causes of automobile crashes. According to the National Highway Traffic Safety Administration (NHTSA) [1], close to 80% of crashes are due to inattentive drivers, as shown in **Figure 1**.

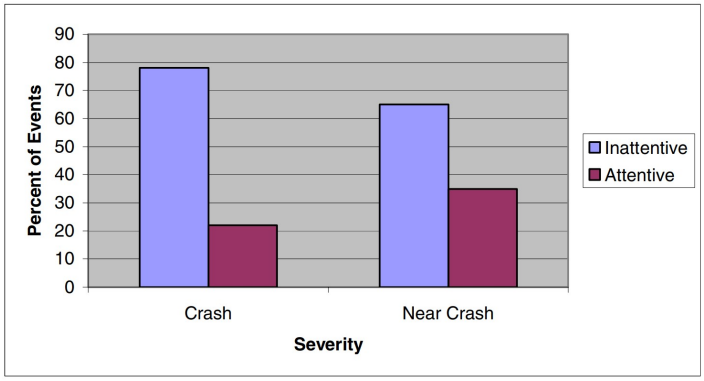

Figure 1: Data from the NHTSA showing the percentage of crashes due to inattentive drivers.

The objective of this research is to create a model capable of quickly and accurately identifying yawning from a video. To accomplish this, it is clear that the focus should be placed on the face and how it reacts and changes while yawning. In order to capture these changes in a consistent and accurate manner, facial landmark detection is used. Facial landmark detection is an application of deep learning in which a model trained on a large dataset of varying images is able to detect facial features of a subject from an image. To utilize this information, we employ the use

of machine learning and support vector machine (SVM) classification and developed a model to take in the data from the landmark detector and predict whether the frame contains yawning or not.

Current Methods

Current methods for detecting yawning mainly focus on driver fatigue, and are not generalized to a wider range of environments where it could be applicable. This includes Sun et al. who focused solely on driver fatigue and used a relatively small dataset with which to train the model [2]. Previous methods also mainly only utilize the current frame to predict whether the person is yawning or not. However, the context of the previous and following frames could be helpful in the model's classification. These previous methods are also relatively outdated due to the ongoing improvement and implementation of facial landmark detection models.

For example, in [3], an SVM model was trained to locate and track the subject's mouth in order to predict yawning. However, since this research was published, facial landmark recognition technology has greatly improved and has allowed for much more accurate mouth tracking as is used in the model presented in this paper.

Other methods utilize other facial features in order to predict yawning, including Abtahi et al. in [4], who utilize mouth, eye, and overall face tracking. Their approach utilizes the single frame that is being evaluated without considering previous or following frames. This could potentially cause the model to recognize false positives due to an unnatural movement.

Facial Landmark Detectors

Facial landmark detectors utilize deep learning technology, a subset of machine learning, to accurately model a human face through facial landmarks. This includes the eyes, nose, mouth, eyebrows, and jaw. By performing a regression, facial landmark detection models can plot points in a 2-dimensional or 3-dimensional space to represent these facial features. The coordinates of these plotted points can then be fed into other models, such as the one created in this research, for other purposes such as detecting yawning.

The development of facial landmark detectors that are very capable and can detect landmarks in a wide variety of head angles and poses is extremely useful in the area of facial feature

recognition. This is due to the facial landmark detectors being trained on a vast amount of data in a wide area of situations, making it very robust in tackling images involving various facial expressions, backgrounds, and lighting conditions. The landmark detector will take in an image containing a face and is able to provide the locations of different points on the face that help capture important characteristics such as the mouth, eyes, and nose. The location data from the landmark detector can then be used to create new features that represent more physical aspects of the human face.

Therefore, by using these facial landmark detectors, a relatively simple but accurate model can be constructed to detect yawning.

CHAPTER II

DATASET

Dataset Formation

The dataset was created by recording six videos of a subject in different situations. This includes the subject yawning and not yawning and showing different angles of the face as well as situations that would be more likely to prompt false positive and negatives. This was done to ensure the model had a set level of robustness so that it could be applied in more situations while still maintaining a high level of accuracy. To further increase the robustness, the dataset could be expanded to include more subjects in a wider variety of situations such as the dataset detailed in [5]. All the videos were recorded at 30 frames per second, which could be altered to make adjustments based on what device is being used to record the input video. The videos were then formed into the dataset by taking all the frames from each video and putting them together, totalling 1014 images. The dataset was hand-annotated by assigning a value of 1 to a frame where the subject was in the act of yawning, and a 0 otherwise. An additional test dataset was constructed in the same manner, consisting of 754 images to test different methods as well as the accuracy of the model.

Example Images

The figures below (**Figures 2-5**) are examples of some images contained in the dataset.

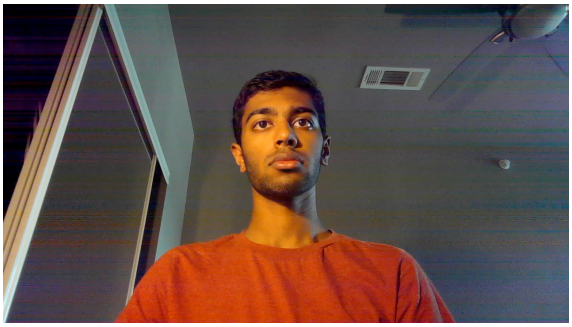

Figure 2: An example of an image with the full face clearly visible, labeled 0.

Figure 3: An example of an image with the full face clearly visible, labeled 1.

Figure 4: An example of an image with the face to the side, labeled 0.

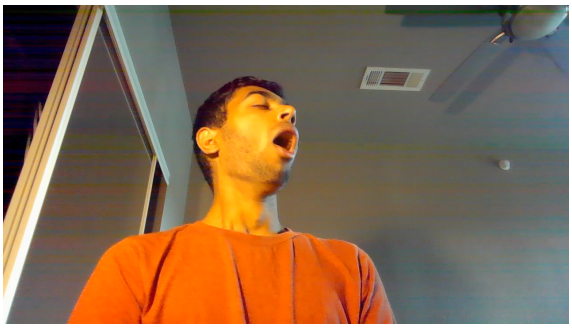

Figure 5: An example of an image with the face to the side, labeled 1.

Facial Landmarks

Facial landmarks are critical to how the dataset is used and implemented into the classification model. The landmark detector plots points on a coordinate system to represent the eyes, nose, mouth, eyebrows, and jaw. From this information, only those representing the eyes and mouth were used. This is because in the motion of yawning, the mouth and eyes react the most, providing the classification model with the most relevant and applicable information to make a prediction.

The facial landmark detector that is used is Face Alignment, created by Adrian Bulat [6]. This is an extremely robust and state-of-the-art facial landmark detector capable of detecting points in both 2-dimensional and 3-dimensional spaces.

Figure 6 and **Figure 7** show how the facial landmark recognition model plots points around the eye and mouth to represent those features numerically, which can be used to calculated other values.

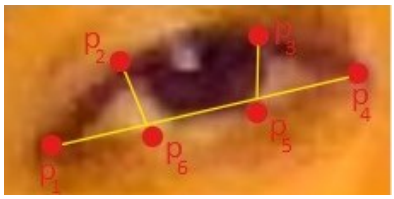

Figure 6: A close up of how the facial landmark recognition plots represents the eye.

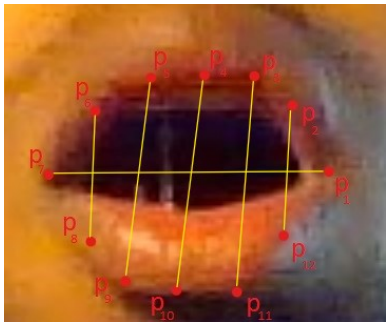

Figure 7: A close up of how the facial landmark recognition plots represents the mouth.

The facial landmarks were extracted from each of the images in the dataset. Several features were engineered using these facial landmarks, including Eye Aspect Ratio (EAR) and a Mouth Aspect Ratio (MAR), which are based on the engineered features from [7]. Using the feature for eye aspect ratio (EAR) from [7], which is calculated using the points in **Figure 6**, we can create the equation below.

These functions calculate the distance between the pairs of points. By summing the vertical distances and taking their average and then dividing by the horizontal distance, these ratio can accurately represent how open the eyes and mouth are.

$$EAR = \frac{||p_2 - p_6|| + ||p_3 - p_5||}{2||p_1 - p_4||} \quad \text{(Eq. 1)}$$

This was also extended to the mouth to create a new feature, mouth aspect ratio (MAR) using the points in **Figure 7**, which is shown below.

$$MAR = \frac{||p_2 - p_{12}|| + ||p_3 - p_{11}|| + ||p_4 - p_{10}|| + ||p_5 - p_9|| + ||p_4 - p_8||}{5||p_1 - p_7||} \quad \text{(Eq. 2)}$$

These engineered features represent how open the eyes and mouth are, which is critical information for the SVM classifier in deciding whether the frame contains yawning or not. The classifier uses Sci-Kit Learn SVC packages and employs the grid search method to make a final prediction.

CHAPTER III

METHODS

A number of different methods were used and tested when creating the model.

Model Flow

As shown in **Figure 8**, the video frames are passed into a facial landmark detector, which will plot points on the subjects face, representing certain facial features. The current frame is then made into a temporal window by including the data of previous and following frames. This provides the model with more context to make a more meaningful and accurate prediction. Then, values for EAR and MAR will be calculated based on Eq. 1 and Eq. 2, which are passed into the SVM model to ultimately make a prediction.

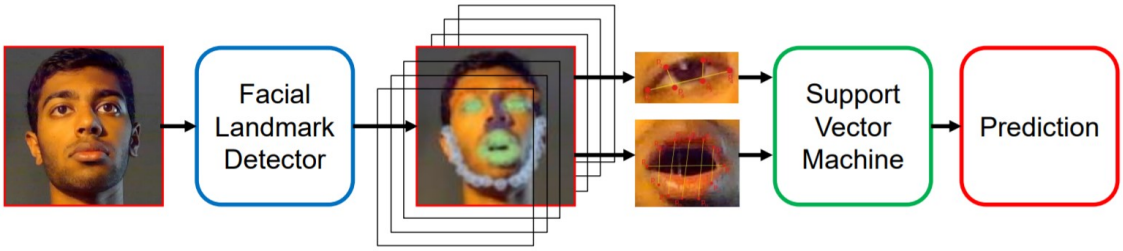

Figure 8: Flow of how model makes a prediction.

Temporal Window

The temporal window is essential to how the model makes a yawning prediction as the context of the previous and following frames help determine what stage in the yawning process the subject is in. It also helps correct any small underlying errors in the facial landmark detector that may be caused by poor lighting or the subject being recorded at a bad angle. This is due to the model taking in data from many frames, which makes errors in a few frames not as significant to the final prediction.

The size of the temporal window can be changed to alter the model's accuracy and process time. A larger temporal window provides the model with more context, however, a temporal window that is too large may provide too much information to the model, clouding the important information that is closer to the current frame with less relevant frames.

Frame Representation

Frame representation is the process of representing a specified number of frames by the information in a single frame. This is done to reduce the processing time for the facial landmark detector, which takes the most time out of all the elements in the model. By representing n frames by 1 frame, we can reduce the processing time of the facial landmark detector by a factor of n. The downside of frame representation is the loss of information contained in those frames, which ultimately gives the model less data to work with. Therefore, a value of n that is too high may lose too much information and cause a steep decrease in the model's accuracy. This can occur if the value of n is greater than or close to the number of frames needed to represent the motion of yawning.

A variety of different values of n are tested and its effects on the model's accuracy are shown in **Table 1**.

CHAPTER IV

CODE

The code that was used to train and test the support vector machine model is shown below. To initially extract the frames from the video, OpenCV and Haarcascade were used. The frames were then passed into the facial landmark detector [6] and values for EAR, represented as (Eq. 1), and MAR, represented as (Eq. 2), were calculated and stored.

Code Segments

In **Figure 9**, the size of the temporal window is defined and the data is appropriately read from the data file, adding the MAR and EAR values to a features array and the true yawning values to a labels array.

```
half_window = 6
window = 13

features = []
labels = []

for i in range(101):
    line = data.readline()
    linedata = line.split(",")


    labels.append(float(linedata[window*2+2]))
    temp = []

    for j in range(window*2):
        temp.append(float(linedata[j+1]))
    features.append(temp)

data.close()
```

Figure 9: Reading in data from training data file.

The parameters of the support vector machine model are defined based on the Gridsearch method as shown in **Figure 10**. As detailed in [8], this is a method which can help over-fitting of the model. Over-fitting is a common issue with machine learning models that is caused by the model modelling itself to the training data too well, and learning the specific detail and noise

contained in the training dataset. This could negatively impact the model's performance on new data that may hold slight differences from the training data.

```
model = SVC(C=1000, cache_size=200, class_weight='balanced', coef0=0.1, degree=3, gamma='auto',
kernel='linear', max_iter=-1, probability=True, random_state=12,shrinking=True, tol=0.001, verbose=False)
model.fit(features, labels)
```

Figure 10: The SVM model is defined and trained.

Figure 11 shows the data from the facial landmark detector and calculated EAR and MAR values being put into an array to be used later.

```
x_test = []

for i in range(represented_frame_count):
    line = testdata.readline()
    linedata = line.split(",")

    temp = []
    temp.append(int(linedata[0]))
    temp.append(float(linedata[1]))
    temp.append(float(linedata[2]))

    x_test.append(temp)

testdata.close()
```

Figure 11: Reading in data from test data file.

In **Figure 12**, the model applies the temporal window to the test data and creates new features composed of the EAR and MAR values from the previous and following frames as defined by the window size in **Figure 9**. This new feature is passed into the trained model and the predictions are stored.

```
predictions = []

for i in range(len(x_test)):
    if (i >= half_window and i <= len(x_test)-half_window-1):
        testvector = []

        for j in range(2):
            for k in range(window):
                testvector.append(x_test[i-half_window+k][j+1])

        pred = (model.predict_proba(np.array(testvector).reshape(1,-1)))[0]
        predictions.append(pred)
```

Figure 12: The model makes predictions on the test data.

Figures 13 and 14 Show the model's predictions for the test data being taken and processed to provide a final prediction of yawning or not yawning. A threshold of 70% is used to make this final evaluation as the model must be at least 70% confident that the frame contains yawning to classify is as yawning. Otherwise, it is classified as not yawning. Changing this threshold value will effect how the model perceives the input data. By increasing the threshold closer to 100%, the model must be extremely confident in its prediction, and by lowering it closer to 0%, the model is extremely loose in its prediction, allowing frames that may contain false positives to slip through. This threshold value can be tailored to specific applications of the model.

```
preds = []
for i in range(len(predictions)):
    preds.append(predictions[i][1])
```

Figure 13: The prediction values are extracted.

```
for i in range(len(preds)):
    if (preds[i] <= 0.7):
        preds[i] = 0
    else:
        preds[i] = 1
```

Figure 14: A threshold is applied to the predictions.

In **Figure 15**, the model is evaluated in terms of accuracy, sensitivity, and specificity. These values are calculated by comparing the predicted values from the model with the true labeled values.

```
accuracy = format(accuracy_score(y_test, preds),'.4f')
sensitivity = format(recall_score(y_test, preds,pos_label=1,average='binary'),'.4f')
specificity = format(recall_score(y_test, preds,pos_label=0,average='binary'),'.4f')

print('Accuracy : ', accuracy)
print('Sensitivity : ', sensitivity)
print('Specificity : ', specificity)
```

Figure 15: The model's performance is evaluated.

CHAPTER V

RESULTS AND CONCLUSION

A number of different methods were used and tested when creating the model. The different methods that were used and their effects on how the model makes its predictions can be seen below.

The results below show that through the use of a landmark detector, meaningful facial features can be gathered and calculated to make an accurate prediction in classifying yawning. By expanding the dataset and increasing the number of situations the model may learn from, it will be more robust in being able to classify yawning from a number of different situations.

Data Representation

In **Figure 16**, the gray areas represent the true not yawning labels and the white areas represent the true yawning labels for the test dataset. In **Figure 17**, the model's predicted labels for the test dataset are shown, with gray representing the model's prediction for not yawning and white represents the model's predictions for yawning. This is for a model with no frame representation and a temporal window of size 13.

For this particular variation of the model, it is more likely to ensure that the frame is truly yawning and consider the frames before and after the yawning period as not yawning, which leads to a low amount of false positives.

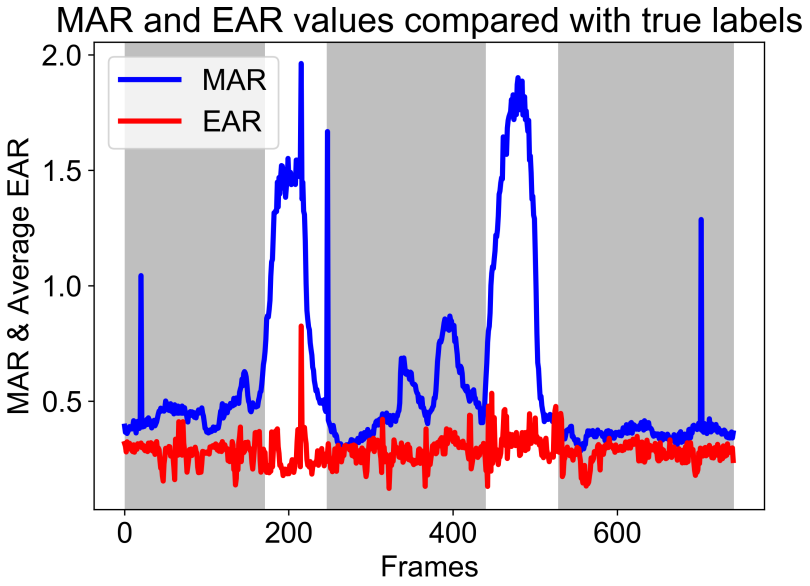

Figure 16: True labels of test video.

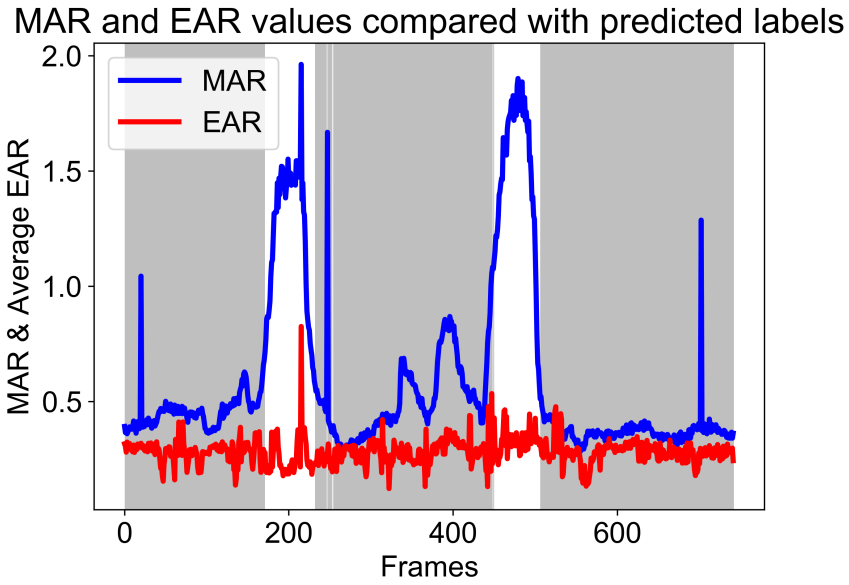

Figure 17: Predictions of test video for no frame representation, 13-Window.

Figure 18 illustrates how the model classifies frames into yawning or not yawning based on the input EAR and MAR values. This data is based on a testing dataset consisting of 754 images produced in the same method as the training dataset. From the visualization of this data, we can see how the SVM model separates the data based on the EAR and MAR values by creating a plane to separate the values and classify them.

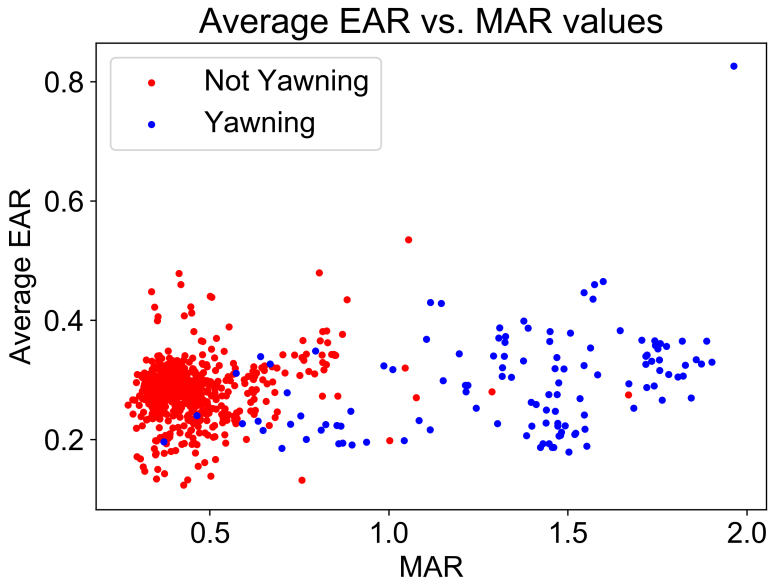

Figure 18: MAR and EAR of the test dataset.

Table 1: Accuracy results of model variations.

Method	Accuracy
Train-Test Split Dataset, 13-Window	87.89%
Average EAR, Train-Test Split, 13-Window	88.95%
Full Test Dataset, 13-Window	93.67%
Subset of Full Test Dataset (angled)	90.94%
Subset of Full Test Dataset (straight)	96.00%
5-Frame Representation, 9-Window	97.84%
7-Frame Representation, 9-Window	96.00%
10-Frame Representation, 9-Window	89.55%
9-Frame Representation, 13-Window	97.22%
11-Frame Representation, 13-Window	80.70%

Model Accuracy

Table 1 shows the various techniques and methods tested as well as their accuracy on the test dataset.

The results in **Table 1** show that using the full training data for training and using a separate test dataset is beneficial to the model's accuracy. It is also apparent that averaging the EAR values for the two eyes and using the average as a feature instead of two separate features is beneficial to the model. This can be attributed to the average removing some of the variation in how the landmark detector recognizes each eye. Based on the angle of the face in the video, the accuracy will vary. The subsets of the full test dataset having varying accuracy shows that yawning in a straight-on video, where the subject's face is fully visible, is more likely to be recognized completely and accurately than yawning at an angle. This may be attributed to one eye not being visible to the facial landmark detector when calculating the EAR value.

Figures 19-23 show the predictions from each of the frame representation model variations that were tested. From testing various values of n for frame representation in an effort to make the model quicker and more efficient, we can see from **Table 1**, with a temporal window of size 9, the limit is representing 7 frames by 1 frame, as anything above that will significantly drop the model's accuracy. By increasing the size of the temporal window to 13, we can see that this limit can be increased to a frame representation of 9 without any significant drop in accuracy.

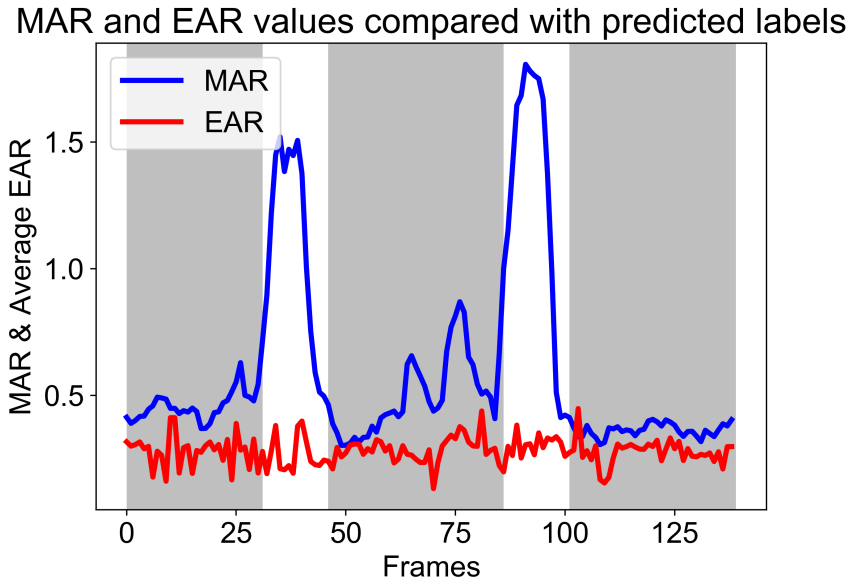

Figure 19: MAR and EAR of the test dataset for 5-Frame Representation with 9-Window.

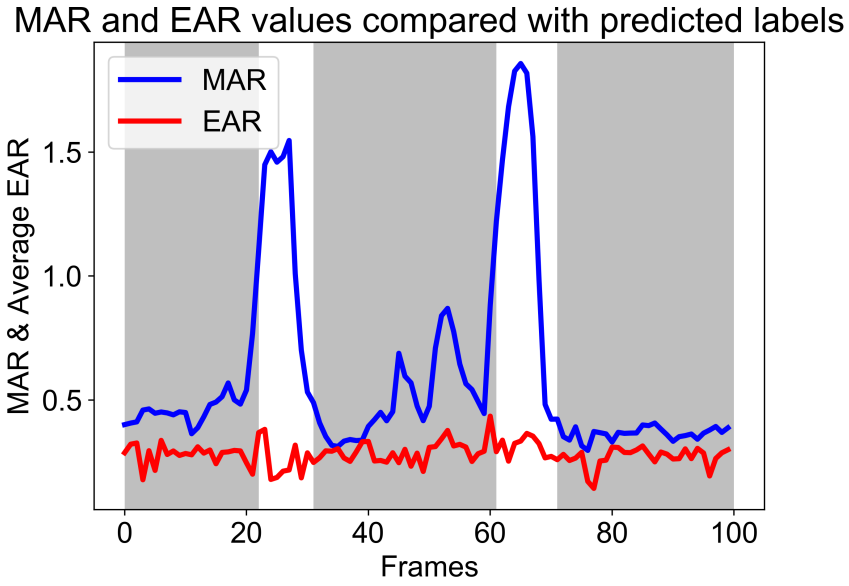

Figure 20: MAR and EAR of the test dataset for 7-Frame Representation with 9-Window.

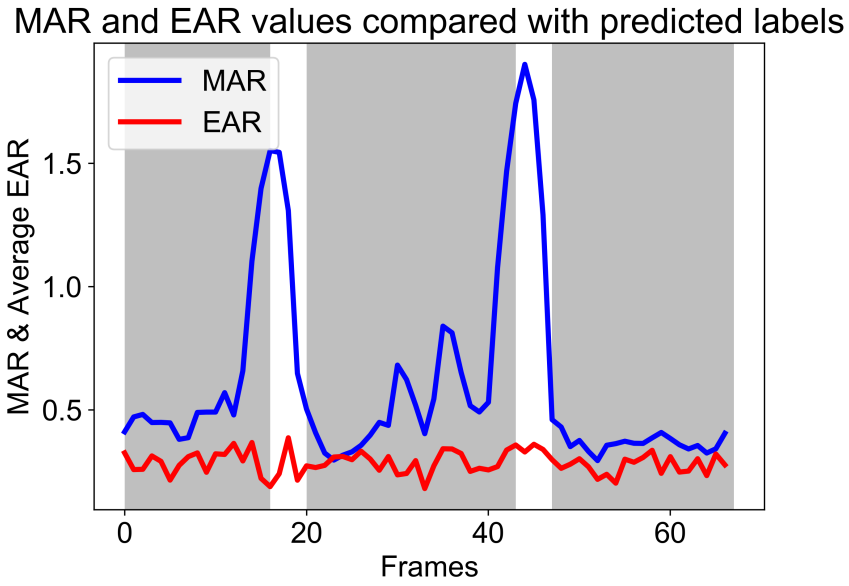

Figure 21: MAR and EAR of the test dataset for 10-Frame Representation with 9-Window.

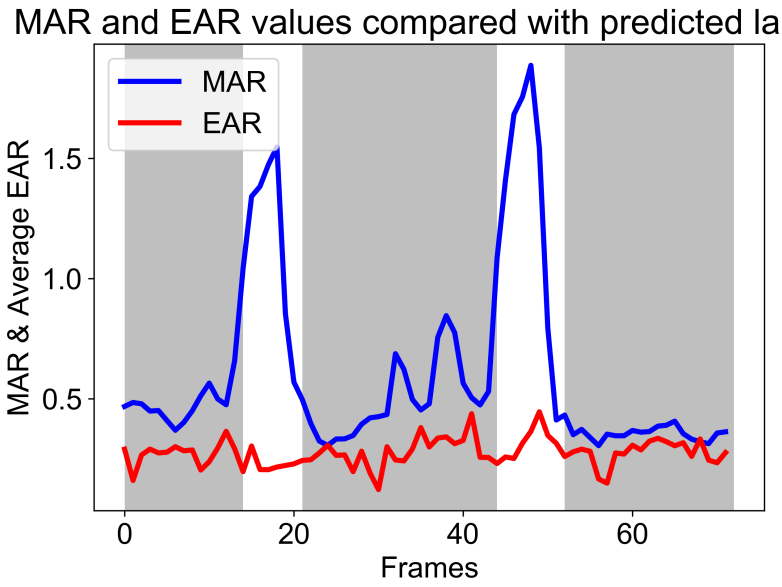

Figure 22: MAR and EAR of the test dataset for 9-Frame Representation with 13-Window.

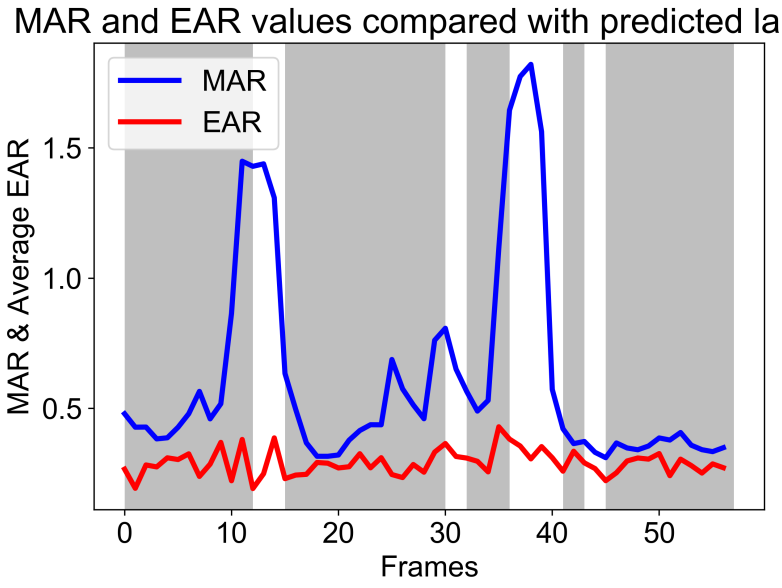

Figure 23: MAR and EAR of the test dataset for 11-Frame Representation with 13-Window.

REFERENCES

[1] P. S. Rau, "Drowsy driver detection and warning system for commercial vehicle drivers: field operational test design, data analyses, and progress," *19th International Conference on Enhanced Safety of Vehicles*, 2005.

[2] X. Fan, B. Yin, and Y. Sun, "Yawning detection for monitoring driver fatigue," *International Conference on Machine Learning and Cybernetics*, vol. 2, 2007.

[3] M. Saradadevi and P. Bajaj, "Driver fatigue detection using mouth and yawning analysis," *International journal of Computer science and network security*, vol. 8.6, 2008.

[4] S. Abtahi, B. Hariri, and S. Shirmohammadi, "Driver drowsiness monitoring based on yawning detection," pp. 1 – 4, 05 2011.

[5] S. Abtahi, O. Mona, S. Shervin, and H. Behnoosh, "Yawdd: A yawning detection dataset," *Proceedings of the 5th ACM Multimedia Systems Conference*, 2014.

[6] A. Bulat and G. Tzimiropoulos, "How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks)," in *International Conference on Computer Vision*, 2017.

[7] T. Soukupova and J. Cech, "Eye blink detection using facial landmarks," *21st Computer Vision Winter Workshop*, 2016.

[8] P. Lameski, E. Zdravevski, R. Mingov, and A. Kulakov, "Svm parameter tuning with grid search and its impact on reduction of model over-fitting," in *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, pp. 464–474, Springer, 2015.