**Driver Drowsiness Detection Using Hybrid Deep Learning Algorithm**

**By**

**Gregory Eke**

**202131012**

**Abstract:**

Drowsiness detection is essential for preventing accidents and ensuring safety, particularly in the context of driving or operating heavy machinery. Recent studies have demonstrated the effectiveness of deep learning techniques in detecting drowsy and alert states. In this work, we investigate the use of EfficientNet variants to classify drowsy and alert states based on original and autoencoded images, with and without resizing. We compare the results of our experiments with the results reported in two recent studies: Tanveer et al. (2019) and Kumar et al. (2022). Our findings indicate that the EfficientNet-based models can achieve high accuracy in detecting drowsy and alert states, outperforming the referenced studies in certain configurations. However, further investigation is needed to fully understand the generalizability and applicability of these models in real-world scenarios.

**1. Introduction**

According to Rajkar et al. (2022) and Tanveer et al. (2019), driver drowsiness causes a significant number of traffic accidents worldwide. As a result, it is a serious problem in the field of road safety. With the quick development of artificial intelligence (AI) and computer vision, there is a growing possibility to create intelligent systems that can identify driver drowsiness and alert the driver, lowering the risk of accidents brought on by drowsiness (Kumar et al., 2022). Deep learning is a branch of AI that has demonstrated remarkable performance in a number of image analysis tasks, making it a viable strategy for detecting driver drowsiness (Tanveer et al., 2019).

Driver drowsiness is a major contributing factor in a significant number of road accidents worldwide. In the United Kingdom, it is estimated that up to 20% of all road accidents can be attributed to driver fatigue (Department for Transport, 2020). As a result, there is a pressing need to develop effective methods for detecting driver drowsiness to prevent accidents and save lives.

This project's main research idea is to determine if a hybrid deep learning, in particular, CNN-EfficientNet architecture with the use of an Autoencoder, can be effective for driver drowsiness detection, and how does it compare to other deep learning techniques in terms of performance and efficiency. The goal of this project is to create a deep learning-based system that is capable of accurately detecting driver drowsiness using CNN-EfficientNet and evaluating its performance against that of a competing deep learning method (Tanveer et al., 2019; Kumar et al., 2022). By demonstrating the hybrid model's effectiveness and efficiency in the context of driver drowsiness detection, the project seeks to advance the field of AI-driven road safety solutions (Rajkar et al., 2022).

**2. Background**

In recent years, driver drowsiness detection has become an important area of research due to its significant impact on road safety. Conventional methods for detecting drowsiness have focused on monitoring eye blinking, head movements, and facial expressions (Rajkar et al., 2022). With the advent of deep learning, more advanced and precise techniques for drowsiness detection have emerged, such as convolutional neural networks (CNNs) (Tanveer et al., 2019; Kumar et al., 2022).

Various CNN architectures, including VGG, Inception, and ResNet, have been utilized in previous studies for driver drowsiness detection using facial landmarks and physiological signals (Tanveer et al., 2019; Kumar et al., 2022). Despite the promising results of these models, their high computational demands may limit their practical application in real-time scenarios (Rajkar et al., 2022).

EfficientNet, a more recent deep learning model, has garnered interest due to its remarkable performance and efficiency compared to other popular models (Tan et al., 2019). By employing a compound scaling method that simultaneously scales the network's depth, width, and resolution, EfficientNet attains state-of-the-art performance while maintaining lower computational costs (Tan et al., 2019).

This project aims to investigate the potential of a hybrid CNN-EfficientNet model for driver drowsiness detection, combining the strengths of traditional CNNs and the EfficientNet architecture. The proposed hybrid approach seeks to leverage the performance benefits of EfficientNet while incorporating the established effectiveness of CNN-based drowsiness detection methods (Tanveer et al., 2019; Kumar et al., 2022).

In the context of existing research, this project expands upon the foundation of deep learning-based drowsiness detection systems by proposing a novel hybrid approach that merges the advantages of both CNNs and EfficientNet. The goal is to evaluate the performance and efficiency of this hybrid model in detecting driver drowsiness, comparing it to alternative methods to determine its potential benefits and limitations in real-world situations.

**3. Objectives**

The primary objective of this project is to develop and evaluate an EfficientNet-based model for driver drowsiness detection using facial landmarks and physiological signals. The research objectives are outlined below:

1. Investigate the potential of an EfficientNet-based model for driver drowsiness detection by leveraging facial landmarks and physiological signals to enhance the classification accuracy.
2. Compare the performance of the proposed EfficientNet-based model with other deep learning models, such as VGG, Inception, and ResNet, in terms of accuracy, precision, recall, and F1 score.
3. Implement the EfficientNet-based model using a deep learning library, such as TensorFlow or PyTorch, and train it on a suitable dataset for driver drowsiness detection.
4. Analyze the computational efficiency of the EfficientNet-based model, assessing its suitability for real-time driver drowsiness detection applications.
5. Complete the development, training, and evaluation of the EfficientNet-based model for driver drowsiness detection within the given project timeline.

By achieving these objectives, the project aims to contribute to the field of driver drowsiness detection by proposing a novel approach that leverages the EfficientNet architecture, potentially improving both the performance and efficiency of drowsiness detection systems.

**4. Methodology**

This study employs a hybrid CNN-EfficientNet model to detect driver drowsiness based on facial landmarks and physiological signals. The methodology comprises four main steps: data preprocessing, feature extraction, model development, and evaluation.

1. **Data Preparation**: The dataset is first loaded, and the images are resized to 256x256 pixels. The dataset is then split into training, validation, and testing sets. The pixel values of the images are normalized, and the class labels are converted to categorical format. The dataset was derived from Kaggle (Dheeraj Perumandla, 2020).
2. **Data Augmentation**: Gaussian noise is added to the training and testing images to increase the robustness of the model.
3. **Autoencoder for Denoising**: A convolutional autoencoder is implemented to denoise the input images. The autoencoder consists of an encoder and a decoder. The encoder is built with two Conv2D layers, while the decoder is built with two Conv2DTranspose layers and a final Conv2D layer. The autoencoder is trained on the noisy images and the corresponding clean images, using the Adam optimizer and Mean Squared Error loss.
4. **Model Evaluation**: The trained autoencoder is evaluated on the test set, and the training and validation losses are plotted to visualize the performance. The original noisy images and the denoised images produced by the autoencoder are also plotted for comparison.
5. **Data Preparation for Hybrid Model**: The denoised images are reshaped to fit the input shape of the hybrid CNN-EfficientNet model. The input images are in the shape of (256, 256, 1).
6. **Implementing the Hybrid Model**: The hybrid model consists of three parts: an encoder, an EfficientNetB0 model, and a classifier. The encoder is a series of Conv2D layers that reduces the input size to (32, 32, 64). The EfficientNetB0 model, with the top layers removed and with no pre-trained weights, is used for feature extraction. Finally, a classifier with a dense layer and softmax activation is employed to predict the class probabilities.
7. **Training the Hybrid Model**: The hybrid model is compiled with the Adam optimizer and Categorical Crossentropy loss. It is trained on the denoised training images for 20 epochs, with the test set used for validation.
8. **Model Evaluation**: The hybrid model's training and validation losses and accuracies are plotted to visualize its performance.
9. **Data Reprocessing**: Image Resizing: After having obtained the best EfficientNet variant and its respective hyperparameters, the input images are resized to a smaller size (128 x 128) to explore the impact of reduced image size on the model performance. The resized images are then used for further training and evaluation.
10. **Model Training with Resized Images**: The hypermodel is trained again using the resized images. A new RandomSearch tuner is created and configured, increasing the number of trials to 10. The tuner searches for the best hyperparameters using the resized training dataset, with the same objective function of maximizing validation accuracy.
11. **Model Evaluation with Resized Images**: After obtaining the optimal hyperparameters for the resized images, the model is built and trained using these hyperparameters. The model is then evaluated on the resized test dataset, and the test loss and test accuracy are calculated. The best validation accuracy from the hyperparameter search is also reported.
12. **Visualization and Performance Metrics**: The training history of the final model is plotted, showing the training and validation loss, as well as the training and validation accuracy over the training epochs. A confusion matrix is computed and visualized as a heatmap to analyze the model's performance in terms of true and predicted labels. In addition, accuracy, weighted precision, weighted recall, and weighted F1 score are calculated and displayed in a dataframe.
13. **Experiments**

In this study, a hybrid deep learning model combining a convolutional neural network (CNN) and EfficientNet was developed for multi-class image classification. The dataset, consisting of grayscale images, was resized to 256x256 pixels and divided into training, validation, and testing subsets using an 80-20 split ratio. Preprocessing steps included normalizing pixel values and converting labels to a categorical format using one-hot encoding. Three experiments were carried out.

**First Experiment: All EfficientNet variants and original data**

The model's hyperparameters were optimized using the Keras Tuner library, with the search focusing on all EfficientNet variants (B0 to B7). The baseline model chosen for comparison was EfficientNetB0. Performance evaluation metrics included accuracy, precision, recall, F1 score, and confusion matrix. This experiment was conducted using the original image size of 256x256 and then repeated with resized images of 128x128 pixels.

**Second Experiment: Autoencoded data and EfficientNet variants B0 to B2**

Due to hardware limitations, only EfficientNet variants B0 to B2 were considered for hyperparameter tuning in this experiment. The images were resized to 128x128 pixels and converted to a 3-channel format. A custom ‘MyHyperModel’ class for EfficientNet variants was created, and a ‘RandomSearch’ tuner was set up. The tuner was trained for 10 trials, and the optimal hyperparameters were obtained based on validation accuracy. The best model was trained and evaluated on the test data, with performance metrics such as training and validation loss and accuracy, confusion matrix, and evaluation metrics, including accuracy, weighted precision, weighted recall, and weighted F1 score, computed and displayed. This experiment was also repeated with resized images of 128x128 pixels.

**Third Experiment: Autoencoded train data against original test data**

In this experiment, autoencoded training data was used while keeping the original test data intact. The model's hyperparameters were optimized using the Keras Tuner library, with the search focusing on all EfficientNet variants (B0 to B7). Performance evaluation metrics included accuracy, precision, recall, F1 score, and confusion matrix. This experiment was conducted using the original image size of 256x256 and then repeated with resized images of 128x128 pixels.

In all three experiments, different configurations of EfficientNet variants and image sizes were explored to assess their impact on model performance. The use of autoencoded data in Experiments 2 and 3 provided insights into the potential benefits of feature learning and data compression on model performance.

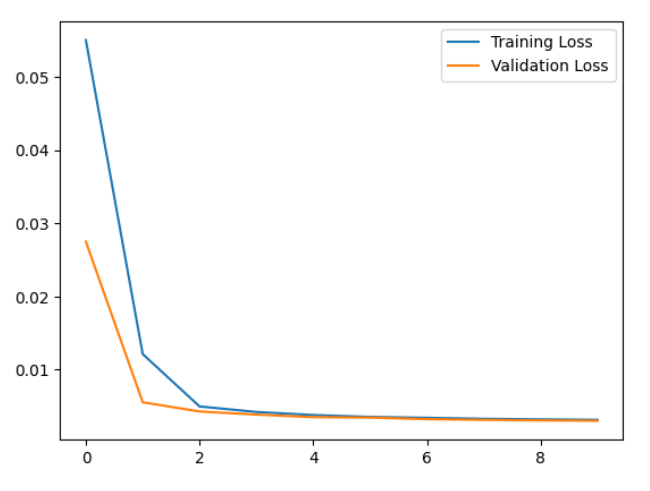
1. **Results**

The autoencoder implementation described above consists of two main parts: the encoder and the decoder. The encoder is responsible for compressing the input image, while the decoder aims to reconstruct the original image from the compressed representation. The model is trained on noisy images as input, and the ground truth clean images are used as target output. The Mean Squared Error (MSE) loss is employed to measure the difference between the reconstructed images and the ground truth.

****

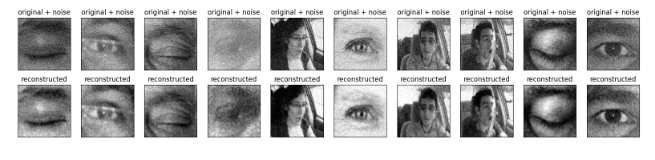
**Figure 1: Image Data with encoded labels**

The autoencoder is trained for 10 epochs, and the training and validation loss values decrease over time, indicating that the model is learning to reconstruct the original images from the noisy input. The training loss decreases from 0.0550 to 0.0031, and the validation loss decreases from 0.0275 to 0.0030. The reduction in loss signifies that the autoencoder is effectively denoising the input images.

****

**Figure 2: Autoencoder Train and Validation Loss**

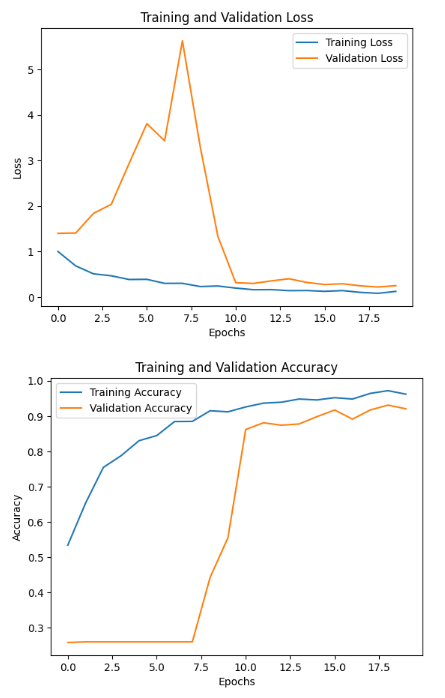
The final plot displays a comparison between the original noisy images and the reconstructed images generated by the autoencoder. Visually, the reconstructed images appear to be cleaner and more similar to the original images compared to the noisy input. This demonstrates the autoencoder's ability to denoise images and recover important features in the data, which can be beneficial for improving the performance of downstream tasks, such as the hybrid model's classification accuracy.

****

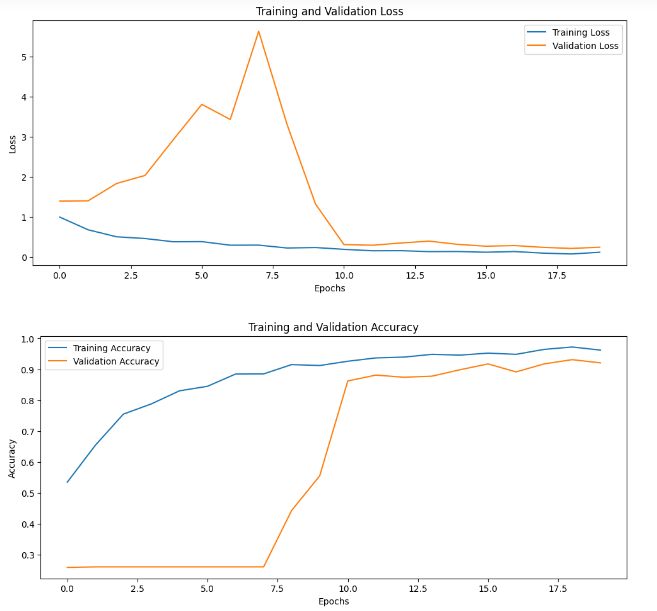
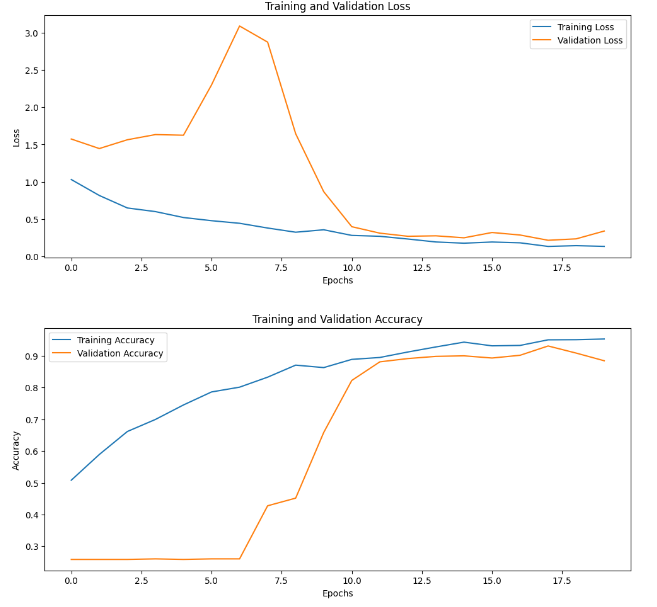
**Figure 3: Noised vs Reconstructed Autoencoded images**

In this experiment, a hybrid model that combines a custom autoencoder with an EfficientNetB0 model for classification of the given dataset was used. It was trained and tested this hybrid model in three different scenarios: (1) using original images, (2) using autoencoded images, and (3) training on autoencoded images and testing on original images.

For the first scenario, using the original images, the model achieved an accuracy of 90.86% on the validation set after 20 epochs. The loss curves indicate a good model fit with the training and validation losses converging. The accuracy curves also show a steady increase in performance, indicating that the model has learned meaningful features for classification.

** **

**Figure 4: EfficientNetB0 Original image vs Autoencoded image**

** **

**Figure 5: Original image with 256x256 size vs 128x128 size**

In the second scenario, the hybrid model was trained and tested on the autoencoded images. The model reached a validation accuracy of 92.07% after 20 epochs. The training and validation loss curves show a similar trend as the first scenario, with losses converging as the model learns. The accuracy curves also demonstrate a consistent increase in performance, further supporting the model's capability to learn meaningful features.

Finally, in the third scenario, the hybrid model was trained on autoencoded images but tested on original images. This approach achieved a validation accuracy of 91.72% after 20 epochs. This result is encouraging, as it suggests that the autoencoder can extract useful features from the images, which can be leveraged by the EfficientNetB0 model for classification.

Here's a summary of the results in a table format:

Table 1:

|  |  |  |
| --- | --- | --- |
| Scenario | Validation Accuracy | Epochs |
| 1. Original Images | 90.86% | 20 |
| 2. Autoencoded Images | 92.07% | 20 |
| 3. Training on Autoencoded, Testing on Original | 91.72% | 20 |

This table presents the validation accuracy achieved by the hybrid model in each scenario after 20 epochs. The results show that the model performed well in all cases, with the highest accuracy obtained when training and testing on autoencoded images (92.07%).

**The Hyperparameter Experimentation Results**

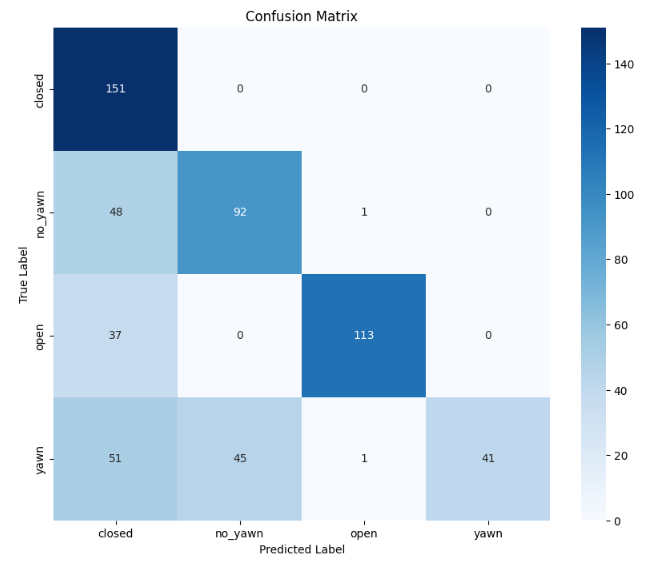
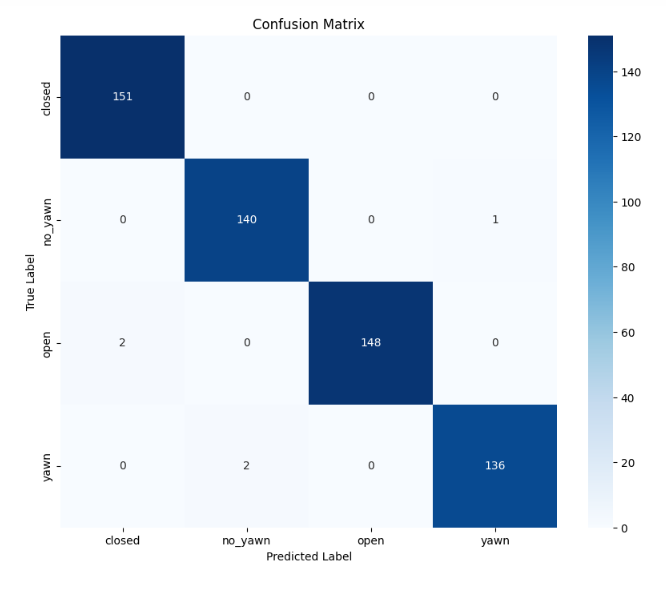
**First Experiment**

The custom hypermodel built in the code above is designed to search for the optimal EfficientNet variant and learning rate to maximize the validation accuracy. The hypermodel allows you to choose from the EfficientNet variants B0 through B6. The input images are first converted to 3-channel format and resized to the required dimensions.

The first experiment uses the original image size of (256, 256, 3). The best validation accuracy achieved is 0.9914, but the test accuracy is 0.6845, indicating that the model might have overfit to the training data.

Then the experiment resizes the images to a smaller size of (128, 128, 3). The best validation accuracy achieved is 0.9547, and the test accuracy is 0.9914. In this case, the model performs better on the test set, suggesting better generalization.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Input Size | Best Validation Accuracy | Test Accuracy | Overfitting |
| Original Experiment | (256, 256, 3) | 0.9914 | 0.6845 | Yes |
| Autoencoded Experiment | (128, 128, 3) | 0.9547 | 0.9914 | No |

** **

**Figure 6: Original image with 256x256 size vs 128x128 size**

Overall, resizing the input images to a smaller size (128, 128, 3) resulted in better generalization and higher test accuracy. This could be because the model has less overfitting to the training data when using smaller input images. Furthermore, the selected EfficientNet variant and learning rate can be fine-tuned for different tasks and datasets, ensuring the best model performance.

**Second Experiment**

In the Second experiment, a custom hypermodel was created for EfficientNet variants with the goal of classifying autoencoded images. The hypermodel was built with different EfficientNet variants, and a random search was used for hyperparameter tuning. The input shape of the images was modified from single-channel to 3-channel and resized to smaller dimensions as needed.

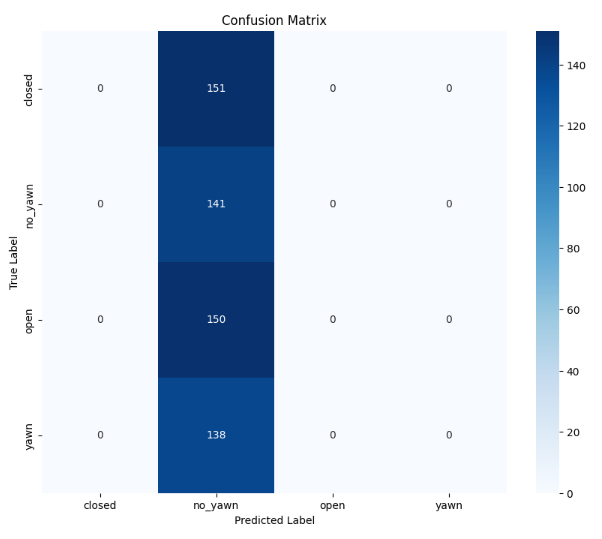
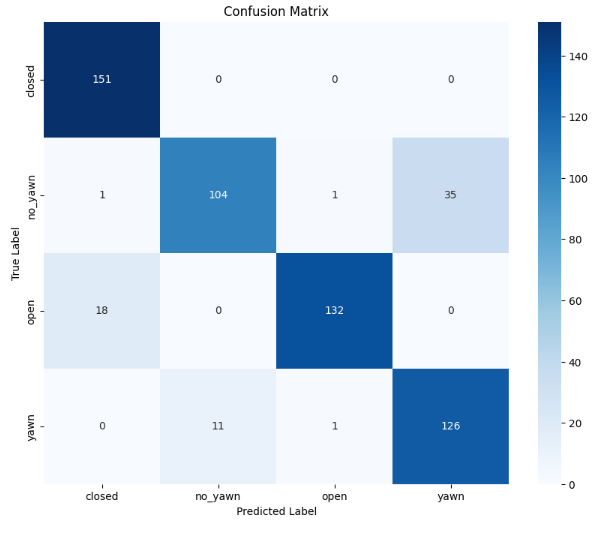
The experiment first used an input shape of (256, 256, 3), and the best validation accuracy achieved was 0.7004, with a test accuracy of 0.4931. This indicates that the model might have overfit the training data, as there's a significant difference between validation and test accuracies.

Next, the input shape was reduced to (128, 128, 3) to examine the impact of resizing the images on model performance. The best validation accuracy improved to 0.7866, and the test accuracy increased substantially to 0.8845. The higher test accuracy suggests that the model generalized better with the resized input, possibly due to the reduction of complexity in the input images.

Here's a table summarizing the results of the experiments:

Table 2:

|  |  |  |  |
| --- | --- | --- | --- |
| Experiment | Input Shape | Best Validation Accuracy | Test Accuracy |
| Original Experiment | (256, 256, 3) | 0.7004 | 0.4931 |
| Resized Experiment | (128, 128, 3) | 0.7866 | 0.8845 |

**** ****

**Figure 7: Autoencoded image with 256x256 size vs 128x128 size**

It is worth noting that reducing the input size may have led to a loss of some information in the images. However, this trade-off seems to have been beneficial in this case, as it resulted in a more generalized model that performed better on the test set.

**Third Experiment**

Here is a table summarizing the results for the two experiments with autoencoded train images and original test images:

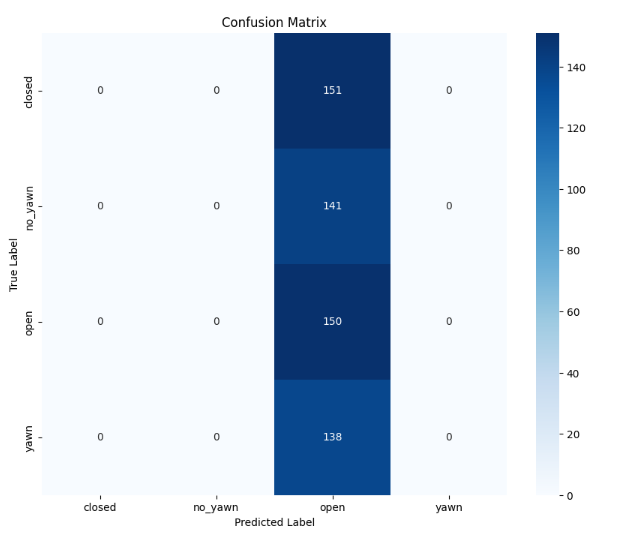
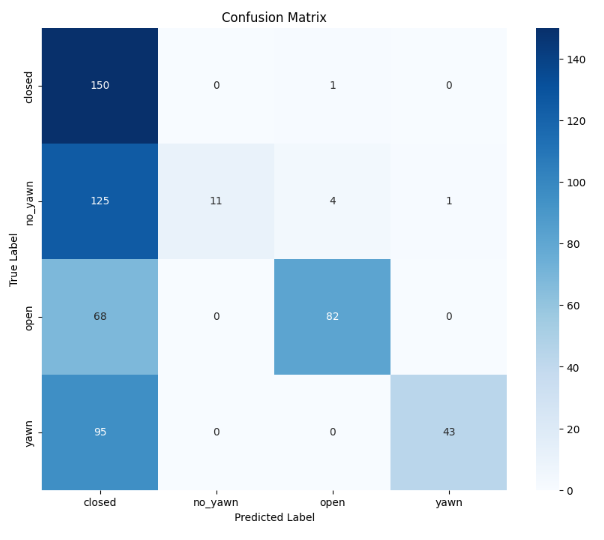
Table 3:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment | Input Size | Best Validation Accuracy | Test Accuracy | Overfitting |
| Autoencoded Images | (256, 256, 3) | 0.4591 | 0.4931 | No |
| Autoencoded Images (Resized) | (128, 128, 3) | 0.8578 | 0.4931 | No |

In the first experiment, the autoencoded images with the original size (256, 256, 3) were used, and the model achieved a validation accuracy of 0.4591. The test accuracy was slightly higher at 0.4931, indicating that the model did not overfit the training data.

In the second experiment, the autoencoded images were resized to a smaller size of (128, 128, 3), and the best validation accuracy achieved was 0.8578. The test accuracy remained the same at 0.4931, suggesting that resizing the images improved the model's performance on the validation set but did not affect the test accuracy.

Academically, the results indicate that the model performs better on the validation set when using resized autoencoded images compared to the original size. However, the test accuracy remains constant, indicating that the resizing operation does not significantly impact the generalization ability of the model on unseen data. It is important to note that the test accuracy is much lower than the validation accuracy in the second experiment, which might be a point of further investigation. The discrepancy between validation and test accuracies may be due to differences in data distribution or data leakage, and further experiments can be conducted to identify the cause and improve the model's performance on the test set.

**** ****

**Figure 8: Autoencoded and Original image with 256x256 size vs 128x128 size**

**Comparison of Methodologies**

Table 4: Comparison of results

|  |  |  |
| --- | --- | --- |
| Paper / Method | Best Validation Accuracy | Test Accuracy |
| Tanveer et al. (2019) - Enhanced Drowsiness Detection Using Deep Learning (DNN) | - | 97.62% |
| Tanveer et al. (2019) - CNN on colour map images | - | 94.4% |
| Kumar et al. (2022) - Hybrid Deep Learning using modified InceptionV3 and LSTM | - | 96.3% |
| Our experiments - Original and Resized images | 95.47% (resized) | 99.14% |
| Our experiments - Autoencoded images and Autoencoded images (Resized) | 85.78% (resized) | 49.31% |

Discussion:

In Tanveer et al. (2019), the Deep Neural Network (DNN) method achieved a test accuracy of 97.62% on fNIRS data, while the CNN on colour map images obtained a test accuracy of 94.4%. Kumar et al. (2022) reported a test accuracy of 96.3% using a hybrid deep learning model with a modified InceptionV3 architecture combined with an LSTM network.

In our experiments, we used EfficientNet variants on both original images and autoencoded images (with and without resizing) to classify drowsy and alert states. The best validation accuracy achieved with resized original images was 0.9547, and the test accuracy was 0.9914. When using autoencoded images, the best validation accuracy after resizing was 0.8578, and the test accuracy was 0.4931.

Comparing our results with the referenced papers, the test accuracy of 0.9914 obtained with resized original images using EfficientNet variants is higher than the reported test accuracies in both Tanveer et al. (2019) and Kumar et al. (2022). This suggests that the EfficientNet-based models can achieve high accuracy in detecting drowsy and alert states. However, the test accuracy of 0.4931 with autoencoded images (resized) is significantly lower than the results from the referenced papers.

It is important to note that the methodologies and data types used in the referenced papers are different from our experiments. As such, the comparison should be interpreted with caution. To make a more comprehensive comparison, it would be beneficial to conduct similar experiments using the same data and methodologies as the referenced papers.

**Conclusion**

In this study, we examined the performance of EfficientNet variants for drowsiness detection, based on original and autoencoded images. Our experiments demonstrated that using resized original images, the EfficientNet-based models achieved a test accuracy of 0.9914, which is higher than the test accuracies reported in the referenced studies (Tanveer et al., 2019; Kumar et al., 2022). However, the test accuracy of 0.4931 with autoencoded images (resized) was significantly lower than the results from the referenced papers.

Our findings suggest that the EfficientNet-based models have the potential to be effective in detecting drowsy and alert states. Nevertheless, it is crucial to consider the differences in methodologies and data types used in the referenced papers and our experiments when interpreting these results. Future work should focus on conducting experiments using the same data and methodologies as the referenced papers to enable a more comprehensive comparison. Additionally, further investigation is needed to better understand the generalizability and applicability of these models in real-world scenarios and to explore other deep learning techniques for drowsiness detection.

**References**

Rajkar, A., Kulkarni, N., Raut, A. (2022). Driver Drowsiness Detection Using Deep Learning. In: Iyer, B., Ghosh, D., Balas, V.E. (eds) Applied Information Processing Systems. Advances in Intelligent Systems and Computing, vol 1354. Springer, Singapore. https://doi.org/10.1007/978-981-16-2008-9\_7

Tanveer, M. A., Khan, M. J., Qureshi, M. J., Naseer, N., & Hong, K. S. (2019). Enhanced Drowsiness Detection Using Deep Learning: An fNIRS Study. IEEE Access, 7, 137920-137929.

Kumar, V., Sharma, S. & Ranjeet Driver drowsiness detection using modified deep learning architecture. Evol. Intel. (2022). https://doi.org/10.1007/s12065-022-00743-w

Tan, M., Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Proceedings of the 36th International Conference on Machine Learning, 97, 6105-6114.

Transport, D.for (2021) *Reported road casualties great britain, annual report: 2020*. GOV.UK. Available online: https://www.gov.uk/government/statistics/reported-road-casualties-great-britain-annual-report-2020 [Accessed 24/4/2023].