Fatigue Detection from Facial Images Using Deep Learning Models

Brief overview of problem, dataset, models used, and key results.

Samir Shrestha
Artificial Intelligence and Machine Learning Program
Fanshawe College
London, Canada
s_shrestha255053@fanshaweonline.ca

Abstract— *The work explores an image-based approach for fatigue detection using deep learning models using PyTorch. Images labeled as "fatigue", and "non-fatigue" are used to train CNN models. The experiments include pre-trained models such as ResNet50, VIGG16, MobileNetV2, and a custom-built CNN model. The results show ResNet50 having the best performance with recall of 90% and f1-score of 81% on fatigue class.*

Keywords—Fatigue Detection, Image Classification, Convolutional Neural Network, Deep Learning, PyTorch

I. Introduction

Fatigue Detection is a critical task in fields such as healthcare, transportation, and general workplace safety and productivity where timely detection of fatigue can prevent accidents and improve productivity. Traditional methods rely on self-reporting, which can be highly inaccurate, and physiological sensors are often impractical and intrusive.

Detecting fatigue from facial images can be challenging due to variations in illumination, head pose, and subtle changes in facial expressions. Deep learning models, particularly Convolutional Neural Networks (CNNs), have shown promise in capturing these patterns automatically.

This work implements and evaluates several deep learning models ResNet50, VGG16, MobileNetV2, and a custom-built CNN, using PyTorch. The models are then trained on a dataset of facial images labeled as "fatigue" and "non-fatigue" and performance is evaluated using metrics such as accuracy, recall, precision and f1-score on both classes.

II. Methodology

This section explains the dataset, preprocessing steps, model architectures, and training setup.

A. Dataset

The dataset, sourced from Kaggle, contains 2200 facial images, evenly distributed across two labeled categories: fatigue and non-fatigue. The dataset contains a balanced set of images captured under diverse real-world conditions for the purpose of training and evaluating deep learning models in image-based fatigue classification.

B. Data Preprocessing and Prepration

The images stored were separated in two folders respectively to each class, fatigue and non-fatigue. The dataset was loaded using *ImageFolder* utility in Pytorch. A preprocessing pipeline was applied to the folders which consisted of resizing to 224x224 pixels, randomly horizontally flipped, converted to tensors, and normalized using mean and standard deviation values to match pretrained model expectations.

C. Model Architecture

Four convolutional neural network (CNN) models were evaluated, ResNet50, VGG16, MobileNetV2 and a custom-made CNN model. For the pretrained models, all feature extraction layers were frozen, and only the final classification layers were fine-tuned to predict two classes (Fatigue and Non-Fatigue).

The custom CNN consisted of two convolutional layers (32 and 64 filters), each followed by ReLU activation and max pooling. The extracted features were flattened and passed through a fully connected layer with 128 units and a final two-class output layer.

D. Training Methodology

A standard supervised training procedure was used following common practices in deep learning. The models were trained for 5 epochs and a learning rate of 1×10^{-4} was used along with Adam optimizer. Cross-entropy loss was used as a criterion for objective loss measurement.

E. Evaluation Methodology

Model performance was measured on the test set using precision, recall, F1 and accuracy as the metrics. A classification report was generated for each of the models using predicted and true labels, and a confusion matrix was generated to visualize class-wise performance. These metrics were used to evaluate and compare the performance of each of the models.

All models were implemented in PyTorch using Jupyter Notebook on Google Colab. The notebook, training code, evaluation scripts, and Streamlit demo are available at: **https://github.com/Thefakedeal/fatigue-classification** .The notebook contains additional intermediate results; confusion matrix, training logs, and evaluation metrics that may be of interest can be made available upon request. For demonstration purposes, a streamlit app was developed to run on the trained models on CPU.

III. Experiments and Results

Models were trained and tested on Google Colab using a T4 GPU to leverage faster computation and efficient model convergence. For the Streamlit demonstration, inference was performed on a CPU to avoid exceeding Colab's GPU time limits and ease of use.

All models were trained for 5 epochs using the same dataset with split, 70% training, 15% validation, and 15% testing. Model performance was evaluated using accuracy, precision, recall, and F1-score, with most of the emphasis on recall value for the fatigue class.

A. Model Performance

The performance of all models on the test set is summarized in Table I. Since the main goal is to detect fatigue accurately, recall and F1-score for the Fatigue class are highlighted alongside overall accuracy.

Table 1: Important performance metrics for fatigue detection models:

Model	Accuracy	Fatigue Recall	Fatigue F1	Macro F1
ResNet50	0.84	0.86	0.84	0.84
VGG16	0.79	0.76	0.78	0.79
MobileNetV2	0.82	0.83	0.82	0.82
Custom CNN	0.84	0.86	0.84	0.84

B. Discussion

Out of all models, ResNet50 achieved the most balanced performance with high recall and f1 score on fatigue class while maintaining good accuracy. Custom CNN also had high recall and F1 score. MobileNetV2 and VGG16 showed moderate performance, with ResNet50 making the fewest fatigue misclassifications.

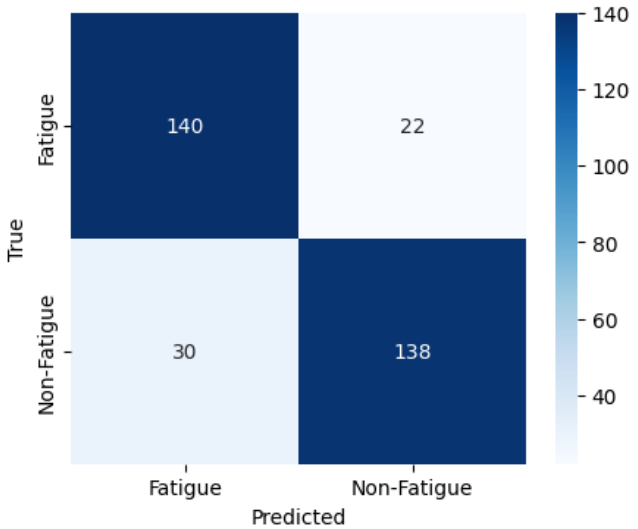


Fig 1: Confusion Matrix for ResNet50 comparing true vs predicted labels for both classes

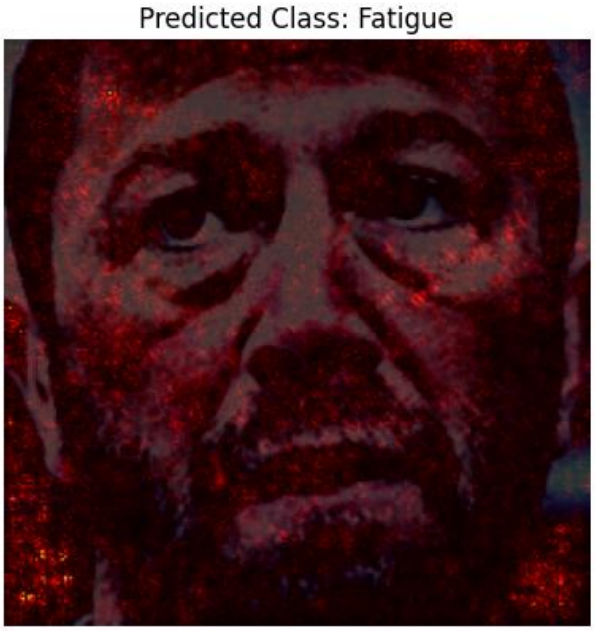
As shown in Figure 1, ResNet50 made fewer misclassifications of fatigue cases compared to other models

Fig 2: Sample images and their true labels along with predicted labels from ResNet50

C. Model Interpretability

Saliency maps were generated for ResNet50 to visualize the pixels that were most influential in its predictions. The maps highlight key facial regions contributing to fatigue detection, providing insight into what the model considers important (Figure 3).

Fig 3: Saliency map for one test sample with ResNet50



IV. Conclusion

This report compared the performance of different pre-trained models with a custom-build CNN model as a baseline for fatigue detection. ResNet50 achieved the best overall performance, demonstrating the effectiveness of transfer learning for this task. Future work could include more pre-trained models, expand the dataset, and explore advanced interpretability.

References

[1] R. Kaci, Fatigue Dataset, Kaggle, 2025. [Online]. Available: https://www.kaggle.com/datasets/rihabkaci99/fatigue-dataset