Final Report: Drowsiness Detection Using Deep Learning on Jetson Nano

Donggen Li (eaz7wk)
Guang Wu (uzd5zk)
Linze Fan (fxp3fw)

Objective

The goal of this project is to build a drowsiness detection system that identifies whether a person's eyes are open or closed using real-time video feeds. The system utilizes deep learning models (MobileNetV2, ResNet18, and EfficientNet-B0) and is implemented on the Jetson Nano platform. The project also demonstrates model training, evaluation, and real-time deployment with a webcam.

Dataset

The dataset consists of four folders:

closedLeftEyes: Images of closed left eyes
closedRightEyes: Images of closed right eyes
openLeftEyes: Images of open left eyes
openRightEyes: Images of open right eyes

Each folder contains .jpg images, which are used for training, validation, and testing.

Project Components

1. Data Preprocessing

A custom PyTorch Dataset class (DrowsinessDataset) is implemented to load images and labels. The preprocessing steps include:

- Conversion of images to grayscale.
- Resizing images to 128×128128 \times 128128×128 pixels.
- Normalization of pixel values to the range [-1, 1].
- The dataset is split into training (80%) and validation (20%) sets.

2. Model Architectures

Three models are used:

MobileNetV2: A lightweight model with a modified input layer for grayscale images and a custom output layer for binary classification.

ResNet18: A deeper architecture trained similarly to MobileNetV2.

EfficientNet-B0: A more efficient model for real-time applications.

3. Model Training

The training pipeline includes:

(1) Loading the dataset using DataLoader.
(2) Training the model in batches with gradient updates.
(3) Evaluating the model on the validation set at the end of each epoch.
(4) Saving the trained model as .pth files for deployment.

4. Real-Time Drowsiness Detection

The trained models are deployed for real-time detection using a webcam. The steps are:

Face and Eye Detection:

- Uses OpenCV's Haar cascades to detect faces and eyes.

Eye Classification:

- Extracts each detected eye and predicts whether it is open or closed using the trained model.

Visualization:

- Draws rectangles around detected eyes and labels them as "Open" or "Closed" with color-coded annotations.

Key Files:

- mobilenet_model.py: Loads and deploys the MobileNetV2 model.
- resnet18_model.py: Loads and deploys the ResNet18 model.
- EfficientNet_B0model.py: Loads and deploys the EfficientNet-B0 model.

5.Platform Setup

Environment Installation:

- Use the official Nvidia JetPack to install required drivers, libraries, and tools.
- Install Python, PyTorch, and OpenCV compatible with Jetson Nano's system.

IDE:

- Recommended: VSCode. Install it from the official repository to ensure compatibility.

Dependencies:

- Install necessary Python libraries (torch, torchvision, opencv-python, etc.).

6.Known Issues

- **Frame Rate:** On Jetson Nano, real-time performance may drop if the model is too complex. Adjust batch sizes or optimize the model for deployment.
- **Lighting Conditions:** Low-light environments may affect eye detection accuracy.
- **False Positives:** Haar cascades occasionally misidentify objects as faces or eyes.

How to Run the Project

1) Training:

Train models using the provided scripts (mobilenet.py, resnet18.py, EfficientNet-B0.py).

Adjust parameters (e.g., epochs, learning rate) in the scripts as needed.

2) Deployment:

Use mobilenet_model.py, resnet18_model.py, or efficientnet_model.py to run real-time detection.

Connect a webcam and ensure it is recognized by the system.

Future Work

- **Integration with YOLO:** Extend the system to detect drowsiness within a multi-person setting.
- **Deployment Optimization:** Use TensorRT to optimize the model for Jetson Nano.
- **Enhanced Features:** Incorporate additional metrics such as blink rate or head movement for more robust detection.

Conclusion

This project successfully implements a real-time drowsiness detection system on Jetson Nano. The approach is lightweight, scalable, and can be further optimized for deployment in automotive or healthcare settings.