Report: Integration of YOLOv8 into Ambianic Fall Detection Model

Introduction:

This report outlines the process of integrating the YOLOv8 pose estimation model into the existing Ambianic fall detection system. The goal was to improve the accuracy and efficiency of fall detection by leveraging YOLOv8's advanced capabilities.

Methodology:

The integration process involved modifying several key files in the Ambianic repository. Each file required specific changes to accommodate the new YOLOv8 model.

1. **demo-fall-detection.py:**

- Updated the script to load and use the YOLOv8 model instead of the previous PoseNet-based model.
- Implemented a new fall detection logic based on YOLOv8's person detection and bounding box analysis.
- Key changes included:
    - Loading the YOLOv8 model using the YOLO class from ultralytics
    - Modifying the preprocessing steps to match YOLOv8 input requirements
    - Adjusting the postprocessing to interpret YOLOv8 outputs for fall detection

2. **demo-fall-detection-cmd.py:**

- Modified the command-line interface to work with YOLOv8.
- Updated the fall detection logic similar to demo-fall-detection.py.
- Ensured compatibility with both two-image and three-image inputs.

3. **fall_detection.py:**

- Completely overhauled the core fall detection module to use YOLOv8.
- Implemented new functions for preprocessing frames and postprocessing results.
- Developed a novel approach to detect falls by analyzing the aspect ratio of detected person bounding boxes.

4. **requirements.txt:**

- Updated the dependencies to include YOLOv8 prerequisites.
- Added ultralytics, torch, and opencv-python-headless.
- Removed TensorFlow-related dependencies.

5. **install_requirements.sh:**

- Modified the installation script to set up the environment for YOLOv8.
- Added commands to install PyTorch and other YOLOv8 dependencies.
- Removed TensorFlow Lite installation steps.

6. **FallDetect-inference.ipynb:**

- Rewrote the notebook to demonstrate fall detection using YOLOv8.
- Implemented functions for single image prediction, directory-based prediction, and URL-based prediction.
- Updated visualization code to show YOLOv8 predictions.

7. **README.md:**

- Updated the project description to reflect the use of YOLOv8.
- Modified installation and usage instructions.
- Added information about YOLOv8 and its benefits for fall detection.

Error Handling and Solutions for Fall Detection Model

In my journey of developing a fall detection model using YOLOv8, I encountered numerous errors. Here, I detail each error and the solutions I implemented to overcome them.

Error 1: File Not Found

Error Message

C:\Users\User\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\python.exe: can't open file 'C:\\Users\\User\\Desktop\\fall-detection\\test_model.py': [Errno 2] No such file or directory

Solution:

The error indicated that the `test_model.py` file did not exist in the specified directory. To resolve this:

1. File Creation:

- I created the `test_model.py` file using a text editor and saved it in the `C:\Users\User\Desktop\fall-detection` directory.

2. Directory Check:

- I ensured that my command prompt was in the correct directory by using the command:

  cd C:\Users\User\Desktop\fall-detection

3. Virtual Environment Activation:

- I activated the virtual environment with:

  .venv\Scripts\activate

4. Running the Script:

- I reran the script using:

  python test_model.py

Error 2: Missing DLL File

Error Message

OSError: [WinError 126] The specified module could not be found. Error loading "c:\Users\User\Desktop\fall-detection\.venv\Lib\site-packages\torch\lib\fbgemm.dll" or one of its dependencies.

Solution:

This error indicated an issue with the PyTorch installation, particularly with the `fbgemm.dll` file.

1. Reinstalling PyTorch:

- I uninstalled the current PyTorch installation:

  pip uninstall torch torchvision torchaudio

- I reinstalled the CPU-only version of PyTorch

  pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

2. Reinstalling `ultralytics`:

- I reinstalled the `ultralytics` package:

  pip install ultralytics

3. Checking Python Version Compatibility:

- I verified that my Python version was compatible with the installed PyTorch version. It turned out I was using Python 3.12, which was not yet supported by the PyTorch version I had.

4. Creating a New Virtual Environment with Compatible Python Version:

- I created a new virtual environment with Python 3.10:

  python -m venv .venv_py310 --python=python3.10

- I activated the new environment and installed the necessary packages again.

Error 3: Installing Specific Versions of PyTorch

Error Message:

ERROR: No matching distribution found for torch==1.9.0+cpu

Solution:

The error indicated that the specified version of PyTorch was not available for my Python version.

1. Installing Latest Compatible Version of PyTorch:

- I installed the latest compatible version of PyTorch for CPU:

  pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

2. Reinstalling `ultralytics`:

- I reinstalled the `ultralytics` package:

  pip install ultralytics

3. Ensuring All Dependencies:

- I installed other required packages from `requirements.txt`:

  pip install -r requirements.txt

Error 4: Compatibility Issues with Python 3.12

Solution:

After numerous attempts, it became evident that Python 3.12 was not fully compatible with the necessary packages. To resolve this, I decided to downgrade my Python version

1. Downloading and Installing Python 3.10:

- I downloaded Python 3.10 from the [official Python website](https://www.python.org/downloads/release/python-31012/).

- I ran the installer and ensured to add Python 3.10 to PATH

2. Creating a New Virtual Environment:

- I created a new virtual environment with Python 3.10:

  python -m venv .venv_py310 --python=python3.10

- I activated the virtual environment:

  .venv_py310\Scripts\activate

3. Reinstalling Required Packages:

- I installed PyTorch, `ultralytics`, and other required packages in the new environment.

4. Running the Script Successfully:

- Finally, I ran the script successfully with:

  python test_model.py