Tracing Car Evolution with AI

Frontend User Guide Report

Submitted by:

Syed Ammad Sohail
261162304

BUSA-613-075

Date: 06-12-2024

Overview

The “Tracing Car Design Evolution with AI” tool is a powerful AI-driven application designed to detect key design changes in car models using advanced computer vision techniques. It uses two deep learning models: one for detecting bounding boxes and keypoints (Model 1) and the other exclusively for detecting keypoints (Model 2). The system provides an interactive interface where users can upload images of cars for comparison and visualize the design differences.

Application Architecture

1. **Models:**
    - **Model 1 (Bounding Box + Keypoints):**
        - Uses a combination of ResNet-34 and Vision Transformer (ViT) for robust feature extraction.
        - Predicts bounding boxes and keypoints for uploaded car images.
        - Enforces keypoints to stay within the predicted bounding box for enhanced reliability.
    - **Model 2 (Keypoints Only):**
        - Employs ResNet-50 with a Feature Pyramid Network (FPN) and attention mechanism to detect keypoints.
        - Focuses solely on predicting normalized keypoints without bounding box constraints.
2. **Backend Training (Backend.ipynb):**
    - The two models were trained using a dataset of 17 annotated images.
    - Preprocessing included resizing images to 1280x855 and normalizing bounding box and keypoint coordinates.
    - Training was implemented using PyTorch and involved multiple loss functions tailored for bounding box and keypoint predictions.
3. **Frontend Application (Front_end.py):**
    - Built with Streamlit for a user-friendly, interactive interface.
    - Allows users to upload two car images, view keypoints and bounding box visualizations, and compare design differences.

User Guide

Step 1: Setup and Launch

1. **Install the Required Dependencies:**
    - Ensure you have Python installed on your system.
    - Install the required Python packages by running:

```
pip install streamlit torch torchvision numpy opencv-python-headless pillow
```

2. **Directory Structure:**
    - Ensure the Front_end.py and models.py files are in the same folder. This is essential for the application to correctly import the model architectures and functions.

3. **Run the Application:**
    - Open a terminal or command prompt.
    - Navigate to the folder containing the Front_end.py script.
    - Start the application by running:

```
streamlit run Front_end.py
```

The application will launch in your default web browser. If it doesn't open automatically, access it at http://localhost:8501.

Step 2: Application Interface Overview

1. **Welcome Screen:**
    - A GIF animation of a car is displayed, along with the title "Tracing Car Design Evolution with AI."
    - A brief description of the application is provided, highlighting its capability to detect car design changes.
2. **Sidebar: Upload Section**
    - Upload Image 1**:** Select the first car image by clicking the "Browse files" button.
    - Upload Image 2**:** Select the second car image by clicking the "Browse files" button.
    - Supported file formats include .jpg, .jpeg, and .png.
3. **Inference Results Section:**

- After uploading the images, the application processes them and displays the results for both models:
  - Model 1: Bounding box and keypoints.
  - Model 2: Keypoints only.
- The **Design Difference** score is displayed to quantify the differences between the two images.

Step 3: Uploading Images

1. **Image Requirements:**
   - Images should preferably contain a front-facing view of cars for optimal keypoint and bounding box predictions.
   - Ensure images are clear, well-lit, and show distinguishable car features.
2. **Uploading Process:**
   - In the sidebar, click "Browse files" to upload the first car image.
   - Repeat the process for the second car image.
   - Once both images are uploaded, the system automatically processes the images.

Step 4: Viewing Inference Results

1. **Model 1 (Bounding Box + Keypoints):**
   - Displays bounding boxes around the car and overlays predicted keypoints for both images.
   - The bounding box indicates the detected car region, while the green dots represent keypoints corresponding to specific car parts (e.g., headlights, grille, wheels).
   - The Design Difference score is calculated based on normalized Euclidean distances between corresponding keypoints in the two images.
2. **Model 2 (Keypoints Only):**
   - Displays only keypoints for both images without bounding boxes.
   - The Design Difference score is calculated in a similar manner as in Model 1.
3. **Key Visualization Features:**
   - Bounding boxes (in blue) clearly outline the car's detected region.
   - Keypoints (in green) are overlaid on specific car features.

- Captions are provided for each image to distinguish between Model 1 and Model 2 results.

Step 5: Interpreting Results

1. **Bounding Box and Keypoints (Model 1):**
    - Bounding boxes help verify the area detected by the model.
    - Keypoints within the bounding box represent critical design features.
    - The Design Difference score quantifies how much the keypoint arrangement differs between the two images.
2. **Keypoints Only (Model 2):**
    - Keypoints focus solely on critical design features without a bounding box constraint.
    - The Design Difference score highlights changes in key design elements.
3. **Design Difference Score:**
    - A higher score indicates more significant design differences between the two cars.
    - A lower score indicates greater similarity in design.

Recommendations for Future Improvements

1. **Enhanced Training Data:**
    - Increase the size and diversity of the training dataset to improve the generalizability of both models.
    - Annotate images with various car angles, lighting conditions, and environmental settings.
2. **Additional Features:**
    - Add a feature to compare more than two images simultaneously for in-depth analysis.
    - Provide detailed statistics on specific design changes (e.g., grille width, headlight positioning).
3. **Improved User Experience:**
    - Enable the option to download annotated images with bounding boxes and keypoints.

- Add sliders or dropdowns to adjust keypoint visualization parameters (e.g., radius, color).

4. **Integration with Cloud Services:**
    - Deploy the application on a cloud platform to enable large-scale processing and accommodate high-resolution images.

Known Limitations

1. **Training Data Size:**
    - The models were trained on a small dataset of 17 annotated images, which might limit their performance on unseen car models.
2. **Static Scaling:**
    - The current scaling assumes all images are resized to 1280x855, which may lead to inaccuracies if input images are processed with different dimensions.