REPORT -VEHANT LAB CHALLENGE

Title: Person Attribute Recognition (PAR) in Indian Context

Team Details:

- Name: TEAM INVISIBLE
- Members-Tejas Gupta , Ayush Goel and Hardik Goel
- Degree Program: Bachelors in Technology
- College -Indian Institute of Technology, Jodhpur

Abstract: Person Attribute Recognition (PAR) is a key element of modern surveillance systems, providing the capability to identify and track individuals based on various attributes. This project addresses the unique challenges of recognizing person attributes in the Indian context, where traditional attire and diverse skin tones are prevalent. Utilizing a deep learning approach, we designed a model tailored to detect attributes specific to Indian attire, enhancing the relevance and accuracy of surveillance systems in smart city environments.

Introduction:

1.1 Overview: Person Attribute Recognition (PAR) is a critical component of city surveillance systems, enabling the identification and tracking of individuals across multiple cameras. This capability enhances the system's ability to retrieve instances with specific attributes, a crucial requirement in surveillance applications. Typical attributes include gender, clothing style, carrying items, etc., which provide high-level semantic information.

1.2 Problem Statement: Existing standard datasets like PETA, Market 1501, and PA100K lack attribute classes relevant to Indian attire, such as kurta, salwar, dupatta, and saree, which are essential for the Indian context. Many solutions in the literature yield good results on these datasets but are not directly applicable to the Indian scenario where skin color, dressing style, and class information differ significantly. This challenge addresses this gap by focusing on detecting person attributes explicitly tailored for the Indian scenario, enhancing the accuracy and relevance of attribute recognition in smart city environments. A sample dataset is provided to sensitize participants to the Indian scenario, and participants are encouraged to enhance the dataset to meet their training requirements suitably.

---

2. Project Objectives:

- To preprocess image data for training and validation.
- To build and train a deep learning model for recognizing multiple attributes in images.
- To perform inference using the trained model to predict attributes in new images.
- To address the unique challenges posed by the Indian context in terms of attire.

---

3. Project Structure:

The project is organized into several directories and scripts as follows:

- **src/:** Contains the source code and data files.
  - **extract.py:** Script to extract the dataset from ZIP files.
  - **preprocess.py:** Script for preprocessing the dataset and defining custom PyTorch datasets.
  - **model.py:** Defines the neural network model architecture.
  - **train.py:** Script to train the model using PyTorch.
  - **inference.py:** Script to perform inference using the trained model.
- **models/:** Directory to store trained model weights.
- **env.yml:** YAML file specifying the environment setup.
- **README.md:** Project documentation.

---

4. Environment Setup:

The environment setup is crucial to ensure that all dependencies required for the project are installed correctly. The `env.yml` file contains a list of dependencies, including specific versions of Python packages and libraries. Using this file, the environment can be created with a command that reads the file and sets up the environment accordingly. This ensures that the development and execution environments are consistent, avoiding potential issues related to package incompatibilities.

---

5. Data Preparation:

5.1 Dataset Extraction: The dataset is extracted from ZIP files using a script that handles the unzipping process and organizes the images into the required directory structure.

5.2 Data Preprocessing: Image preprocessing involves resizing and normalizing the images. Custom PyTorch datasets are defined to load and preprocess the data efficiently. This step is crucial for preparing the data in a format suitable for training the deep learning model. The images are resized to 224x224 pixels, normalized to have a mean and standard deviation matching the pretrained ResNet-50 model, and augmented with techniques such as random cropping, horizontal flipping, and color jittering to enhance robustness and generalization.

---

6. Model Architecture:

6.1 Neural Network Design: The neural network model utilizes a pretrained ResNet-50 backbone, known for its powerful feature extraction capabilities. This backbone is combined with a custom fully connected layer designed to predict the specific attributes relevant to the Indian context. This architecture balances the generalization capability of ResNet-50 with the specificity required for recognizing Indian attire and other attributes.

6.2 Loss Function and Optimizer: The model is trained using the Adam optimizer, known for its efficiency and adaptive learning rate. The loss function used is BCEWithLogitsLoss, suitable for multi-label classification tasks where each attribute is treated as an independent binary label.

---

7. Training and Evaluation:

7.1 Dataset Splitting: The dataset is split into training and validation sets to ensure that the model is evaluated on unseen data during training. This split is typically 80% for training and 20% for validation. The training set is used to update the model weights, while the validation set provides a measure of the model's generalization capability.

7.2 Loss Function and Optimizer: Binary Cross Entropy with Logits Loss (BCEWithLogitsLoss) is used as the loss function. This loss function is suitable for multi-label classification tasks where each label is independent. The Adam optimizer is chosen for its efficiency and ability to handle sparse gradients, making it a good fit for deep learning tasks.

7.3 Training Loop: The model is trained over several epochs. In each epoch, images are fed into the model, predictions are made, and the loss is calculated. The optimizer then adjusts the model weights to minimize the loss. After each epoch, the model is evaluated on the validation set to monitor its performance. The validation loss is calculated, and mean label accuracy is measured to assess the model's ability to generalize to different attributes.

7.4 Mean Label Accuracy: Mean label accuracy is calculated as the average accuracy of all predicted attributes. This metric provides a comprehensive measure of the model's performance across all attributes, highlighting its ability to recognize each attribute accurately.

Validation Label-Based Mean Accuracy: 0.9506

Example

```
Predicted Labels: ['UB_tshirt', 'SLEEVES_short', 'Carry_handbag', 'POSE_standing', 'VIEW_back']
Accurate Labels: ['UBCOLOR_blue', 'UBCOLOR_green', 'UBCOLOR_orange', 'UBCOLOR_red', 'UBCOLOR_white', 'UBCOLOR_yellow', 'UBCOLOR_mix', 'UBCOLOR_other'
Inaccurate Labels: ['UBCOLOR_black', 'LBCOLOR_mix', 'LB_Shorts']
46
```

---

8. Inference:

8.1 Performing Inference: Inference on unseen images is conducted using the trained model. The model weights are loaded, and predictions are made on the test dataset. These predictions provide attribute labels for each image, which can be used for surveillance and monitoring purposes.

8.2 Visualization: Results are visualized to provide insights into the model's performance and areas for improvement. Visualization helps in understanding which attributes are correctly recognized and which need further refinement. Techniques such as heatmaps and attribute-specific highlighting are used to show the model's attention and predictions.

---

9. Results and Evaluation:

9.1 Submission File:
The predictions are formatted into a text file suitable for submission to the challenge. The file contains binary values indicating the presence or absence of each attribute for each image. This format ensures that the results are easily interpretable and can be evaluated against the ground truth.

---

10. Challenges and Solutions:

10.1 Addressing the Indian Context: The unique challenges posed by the Indian context include variety of traditional attires. The dataset was enhanced to include relevant classes, and the model was fine-tuned to handle these variations effectively. This enhancement ensures the model's relevance and accuracy in real-world Indian surveillance scenarios.

10.2 Data Augmentation: Data augmentation techniques were employed to improve the model's robustness and generalization ability. These techniques help the model handle variations in the input images and improve performance on unseen data. Augmentation methods included random cropping, horizontal flipping, rotation, scaling, and color adjustments to simulate real-world variations and improve model robustness.

---

11. Uniqueness and Novelty:

11.1 Pretrained ResNet-50 Backbone: Utilization of a pretrained ResNet-50 backbone combined with a custom fully connected layer for attribute prediction. This approach leverages the powerful feature extraction capabilities of ResNet-50 while allowing customization for specific attributes.

11.2 Custom PyTorch Datasets: Efficient preprocessing techniques and custom PyTorch datasets handle large-scale image datasets, ensuring smooth and efficient training. This customization allows for optimized data loading, augmentation, and preprocessing pipelines, enhancing training efficiency and model performance.

11.3 BCEWithLogitsLoss: Use of BCEWithLogitsLoss for multi-label classification, allowing the model to predict multiple binary attributes simultaneously. This loss function is well-suited for tasks where each attribute is an independent binary classification, enabling the model to handle multiple attributes effectively.

---

12. Flow Diagram: The following flow diagram represents the major steps in the project:

```
+-------------------+
 |                   |
 |    Extract Zip    |
 |                   |
 +-------------------+
          |
          V
 +-------------------+
 |                   |
 |   Preprocess Data |
 |                   |
```

```
+--------------------+
         |
         V
+--------------------+
|                    |
|    Train Model     |
|                    |
+--------------------+
         |
         V
+--------------------+
|                    |
|   Make Predictions|
|                    |
+--------------------+
```

---

13. Conclusion: The Person Attribute Recognition project effectively addresses the challenges posed by the Indian context in surveillance systems. By leveraging a pretrained ResNet-50 backbone and enhancing the dataset to include relevant Indian attire, the model achieves high accuracy in attribute recognition. The structured approach to data preparation, model training, and evaluation ensures robust performance, making this solution suitable for real-world applications in smart city environments.

---

14. Credits: This project relies on several key libraries and tools, including:

- **PyTorch:** For building and training the deep learning model.
- **scikit-image:** For image preprocessing and augmentation.
- **Pandas:** For dataset management and manipulation.
- **NumPy:** For numerical operations and data manipulation.
- **Matplotlib:** For visualizing results and performance metrics.
- **TQDM:** For providing progress bars during training and data loading.
- **OpenCV:** For additional image processing and visualization tasks.

These resources are instrumental in building and training the deep learning model, preprocessing the data, and visualizing the results.