Digital Image Processing

Project

Name:	**Sehrish khan**
Section:	A
Submitted on:	12/3/2023
Submitted to:	Sir Akhtar Jamil

1

article graphicx

Cloth Classification System

This project aimed to develop a comprehensive cloth classification system using a combination of digital image processing techniques and a deep learning approach. The primary goal was to create an efficient and accurate system capable of classifying clothing items into specific categories based on images. The project's main features include:

Techniques used for developing the project

Various techniques, such as resizing, color correction, noise reduction, and contrast enhancement, were implemented to ensure consistent and high-quality input data.

Two separate approaches were employed to classify clothing items – first utilizing Digital Image Processing (DIP) techniques independently and subsequently employing a Deep Learning Approach. Image processing methods were employed to extract relevant features, such as shape attributes. These extracted features served as key components in enhancing the system's ability to discern distinct clothing categories.In the latter, a machine learning model, specifically a Convolutional Neural Network (CNN), was trained to recognize and classify clothing items based on the features extracted using DIP techniques.

Features of the project

A user-friendly interface was developed to facilitate user interaction. This interface allowed users to seamlessly upload images of clothing items and receive detailed classification results. The design prioritized ease of use and accessibility for a wide range of users.

Upon image classification, the system provided users with comprehensive results, including the predicted category and other relevant information. This output format ensured that users could interpret and utilize the system's classification results effectively for their specific needs.

Accuracy Assessment

Methods for evaluating the accuracy and performance of the classification model were implemented, incorporating metrics such as precision, recall, and F1 score.

Software Used

- Python
- OpenCV
- TensorFlow or PyTorch
- Scikit-learn
- Tkinter, PyQt, or wxPython

Developing the project using Digital Image Processing Techniques

article graphicx enumitem

Cloth Classification System

The Python code implements a cloth classification system using a combination of digital image processing techniques. The primary objective is to categorize clothing items into specific categories, such as shirts, trousers, and more.

In general, the cloth's shape is extracted through image processing techniques, including background removal and contour extraction. Grayscale conversion, Gaussian blur, and Canny edge detection emphasize the clothing item's distinctive features. Likewise, Keypoint detection, facilitated by the ORB (Oriented FAST and Rotated BRIEF) algorithm, identifies unique points of interest in the image, capturing essential characteristics of the cloth's shape. Feature matching compares these keypoints with reference images for various clothing categories, enhancing the accuracy of cloth classification by focusing on distinctive features. This combined approach provides a robust and efficient system for categorizing clothing based on their shapes.

Functionalities:

[left=0pt]**Image Preprocessing:**

- – Techniques such as resizing, color correction, noise reduction, and contrast enhancement are applied to ensure consistent and high-quality input data.

- **Background Removal:**
  - The code includes a function (`remove_background`) that removes the background from the uploaded image. This step is crucial for focusing on the clothing item itself.

- **Grayscale Conversion:**
  - The uploaded image is converted to grayscale using the `convert_to_grayscale` function, simplifying further image processing.

- **Gaussian Blur:**
  - A Gaussian blur is applied to the grayscale image using the `apply_gaussian_blur` function. This helps in reducing noise and enhancing feature extraction.

- **Canny Edge Detection:**
  - The Canny edge detection algorithm is utilized (`apply_canny_edge_detection`) to identify edges in the image, emphasizing key features.

- **Contours Extraction:**
  - The code extracts valid outermost contours from the Canny edge-detected image using the `find_valid_contours` function. This step helps in identifying the shape of the clothing item.

- **Contours Drawing:**

- The code includes a function (`draw_and_display_contours`) to draw and display the identified contours on a black background. This visually represents the detected features.

- **Cloth Identification:**
  - The code employs keypoint detection and feature matching using the ORB (Oriented FAST and Rotated BRIEF) algorithm. It compares the features of the uploaded image with reference images of various clothing categories.

- **Result Display:**
  - The result of the cloth identification process is displayed to the user through the GUI. The identified category, such as shirt, trouser, etc., is shown.

- **Graphical User Interface (GUI):**
  - The code utilizes the Tkinter library to create a user-friendly interface. Users can upload images, trigger the processing, and view the classification results.

- **Algorithm Used:**
  - The ORB algorithm is used for keypoint detection and feature matching. ORB is efficient for computer vision tasks, especially in the context of identifying distinctive features in images.

- **Reference Images:**
  - The code includes a set of reference images for various clothing categories, enabling the system to match and identify the category of the uploaded image.

Output:

The output of the code is a description of the category of the cloth to which it belongs. The image of the user-interface and the output is shared below.

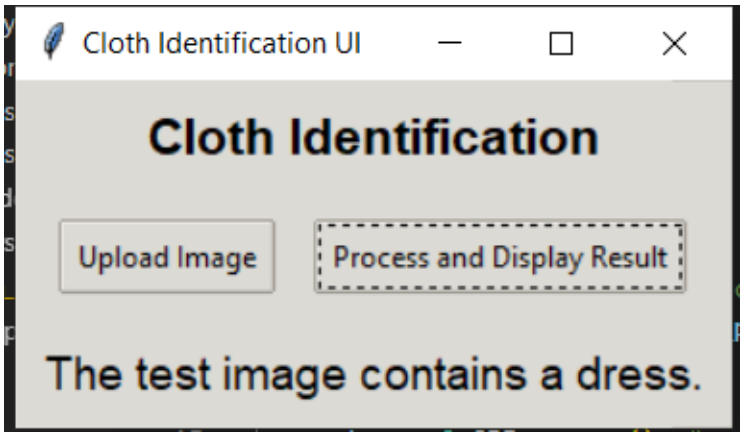

Figure 1: User Interface

Using Convolutional Neural Network (CNN) for Cloth Classification

The code implements a Convolutional Neural Network (CNN) for the classification of clothing items using the Fashion MNIST dataset. CNN is expected to learn features such as edges and textures etc that enable it to distinguish between different types of clothing based on patterns, textures, and shapes present in the Fashion MNIST images.2D numpy array of 28(28 pixels is used as features here. The exact features learned depend on the dataset and the specific characteristics of the clothing items within it. During training, the model adjusts its parameters to optimize its ability to extract features that are informative for the classification task. The key functionalities are as follows:

1. **Data Loading and Preprocessing:**
   - The Fashion MNIST dataset is loaded, containing grayscale images of 10 different clothing categories.
   - Random samples from the dataset are visualized.
2. **Data Preprocessing:**
   - Image pixel values are normalized to the range [0, 1].
   - The dataset is split into training and validation sets.
3. **CNN Model Definition:**
   - A sequential model is defined using Keras with convolutional layers, max-pooling layers, flattening layers, and dense layers.
   - The model is compiled with the Adam optimizer and sparse categorical crossentropy loss for multi-class classification.
4. **Model Training:**
   - The model is trained on the training set and validated on the validation set.
   - Training progress is visualized with accuracy and loss plots.
5. **Model Evaluation:**
   - The trained model is evaluated on the test set.
   - Random samples from the test set are visualized along with their actual and predicted labels.
6. **Confusion Matrix and Classification Report:**
   - Confusion matrix and classification report are generated to assess the model's performance on the test set.
   - The confusion matrix is visualized using Seaborn's heatmap.
7. **Model Saving:**
   - The trained model is saved to a file ('fashion_mnist_cnn_model.h5') for future use.
8. **Image Prediction:**
   - An external image ('PDAY.jpg') is loaded, preprocessed, and used to demonstrate how the trained model predicts the class of the clothing item in the image.

Additional Information

- The code includes necessary installations and imports.
- The model's architecture and training parameters are specified.
- The code outputs a classification report, confusion matrix visualization, and a prediction for an external image.

Comparing DIP with CNN used for cloth classification

DIP Approach:

- **Simplicity and Transparency:** The logic behind each image processing step is clear and transparent, making it easy to debug and modify.

CNN Approach:

- Utilizes a deep learning model with convolutional layers, max-pooling layers, and dense layers.
- Learns hierarchical features through backpropagation during training.

Strengths:

[label=–, left=0pt, itemsep=0pt]**Representation Learning:** CNNs automatically learn representations at different levels, capturing intricate features.

Model Training and Learning

DIP Approach:

- Rule-based feature matching using ORB descriptors.
- Relies on predefined rules and reference images.

Strengths:

[label=–, left=0pt, itemsep=0pt]**Applicability to Limited Data:** DIP techniques may perform well with limited data, especially when explicit rules can capture relevant features.

CNN Approach:

- Learns features from data during training, adjusting weights through backpropagation.
- Generalizes well to unseen data with proper training.

Strengths:

[label=–, left=0pt, itemsep=0pt]**Generalization:** CNNs can generalize well to diverse datasets, provided an adequate amount of training data.

Flexibility

DIP Approach:

- May require fine-tuning for different datasets and environments.
- Relies on human-defined rules.

Strengths:

[label=–, left=0pt, itemsep=0pt]**Explicit Control over Processing Steps:** Provides explicit control over each processing step, allowing fine-tuning of parameters.

CNN Approach:

- Generalizes well to various datasets but requires substantial training data.
- Learns features adaptively from data.

Strengths:

[label=–, left=0pt, itemsep=0pt]**Adaptability:** CNNs adapt to different datasets without extensive manual adjustments.

Conclusion

DIP Approach:

- Rule-based, interpretable, and suitable for scenarios with well-defined features.
- May require manual adjustments for different datasets.

CNN Approach:

- Automatically learns features, adaptable to various datasets.
- Effective for complex tasks and excels in scenarios with intricate and abstract features.

Complexity

- The complexity is determined by the image processing functions, and the dominant factor is usually the number of pixels (p) in the image. Therefore, the overall complexity of the code can be expressed as O(p) for image processing operations.

Visualizations

article enumitem

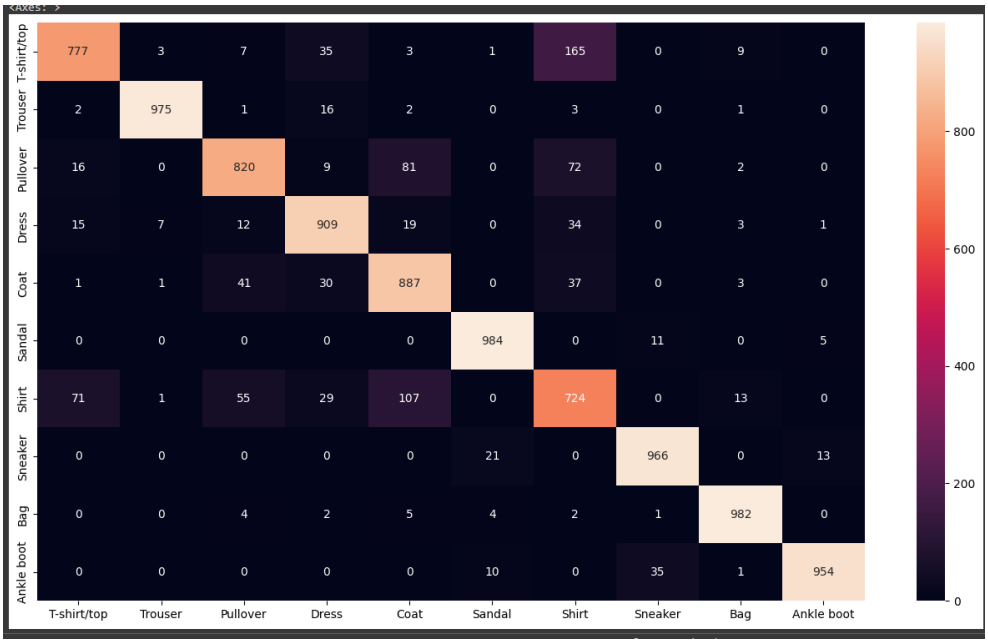

Figure 2: Prediction Graph

	precision	recall	f1-score	support
T-shirt/top	0.81	0.89	0.85	1000
Trouser	0.99	0.98	0.98	1000
Pullover	0.83	0.88	0.85	1000
Dress	0.90	0.92	0.91	1000
Coat	0.85	0.82	0.83	1000
Sandal	0.98	0.98	0.98	1000
Shirt	0.76	0.65	0.70	1000
Sneaker	0.93	0.98	0.96	1000
Bag	0.97	0.98	0.98	1000
Ankle boot	0.99	0.94	0.96	1000
accuracy			0.90	10000
macro avg	0.90	0.90	0.90	10000
weighted avg	0.90	0.90	0.90	10000

Figure 3: Table