IMAGE CLASSIFICATION

Topic Name and Group Details

IMAGE CLASSIFICATION

Group Name: Group A

Group Members:

First name	Last Name	Student number
Sneh	Patel	C0869367
Malav	Shah	C0870066
Dhruvraj	Chavda	C0867457
Nandiniben	Patel	C0869773

Table of Contents

Contents

Abstract

- Our Project aim is ideal for **developing and training image classification models** in the field of **sports celebrity recognition.**
- The dataset can be used to build machine learning models that can accurately classify images of sports celebrities based on their respective disciplines.
- This code is for building an image classifier using **Python** and OpenCV.
- The goal of this code is to train a model to classify images of sports celebrities into their respective categories.
- The celebrities in this code include **Kane Williamson, Cristiano Ronaldo, Maria Sharapova, and Kobe Bryant.**
- The images that we have taken from Getty Images website.
- It includes images of varying resolutions and sizes, which have been uniformly **resized to 200x200 pixels.**
- The dataset also provides labels for each image, indicating the name of the celebrity and their respective sport discipline.

Introduction

- Image classification is a type of machine learning technique that involves training a computer program to recognize and classify images based on their visual characteristics.

- This is typically done by feeding the program a large dataset of labeled images and using algorithms to identify patterns and features within the images that can be used to classify them.

- The program then applies these learned patterns and features to new images to determine their content or label.

- Image classification has a wide range of practical applications, including object recognition, facial recognition, medical imaging, and many others.

- The **first image** shows a **cricket player** mid-swing, with the ball in the background. This image represents the cricket celebrity **Kane Williamson**, who is **one** of the four sports celebrities included in this dataset.

- The **second image** shows a **football player** at mid-kick, with the ball in the foreground. This image represents the football celebrity **Cristiano Ronaldo**, who is another sports celebrity included in the dataset.

- The **third image** shows a female **tennis player** in action, with a blurred background and a focused look on her face. This image represents the tennis celebrity **Maria Sharapova**, who is also included in the dataset.

- The **fourth image** shows a close-up of a basketball player's face, with sweat dripping down his forehead. This image represents the **basketball** celebrity **Kobe Bryant**, who is the fourth and final sports celebrity included in this dataset.

- Overall, these **four images represent a diverse set of sports celebrities from different sports and backgrounds** and provide a glimpse into the type of images included in this dataset.

- The images in the dataset are labeled with the name of the sports celebrity that they depict, which makes it easy to train a model to recognize each celebrity.

- The dataset is suitable for use in a variety of machine learning frameworks, including **Keras, TensorFlow**, and more.

Overall, **the Sport Celebrity Image Classification dataset** is a useful resource for **researchers, developers, and data scientists** who are working on image classification tasks related to sports celebrities.

Understanding Problem Statement

- Building a model that can reliably categorize photos of four distinct sports celebrities—**Kobe Bryant**, **Kane Williamson**, **Maria Sharapova**, and **Cristiano Ronaldo**—**is the problem statement for the Sport Celebrity Image Classification dataset.**

- The objective is to use a set of labelled photographs of these celebrities to train a machine learning model so that, when it encounters a fresh, unlabeled image of one of these celebrities, it can properly categorize the image and determine which celebrity it represents.

- The importance of this kind of picture categorization assignment may be seen in a variety of domains, such as **sports analytics and computer vision.**

- Automated object detection, face recognition, and other critical applications may all be improved with accurate picture category.

Importing Necessary Libraries

- Python's **arrays and numerical** operations are supported by the **NumPy** library.
- Python **data analysis and manipulation** are supported by the **Pandas** library.
- Python **image and video processing** is supported by the **OpenCV** library.
- The Python programming language has a **plotting library** called **Matplotlib**.
- Running on top of **TensorFlow** is the high-level **neural networks API** known as **Keras.**
- For **data mining and data analysis**, including machine learning algorithms, the **scikit-learn** library offers straightforward and effective tools**.**

Loading the Dataset

- This dataset reads every image in the folder with the **extension.jpg** and loads a specific celebrity, **Kane Williamson, from the specified path.**
- The images are **resized** to the desired shape of **(200,200)** and added to **two distinct lists, kane_images and kane_labels, respectively**, along with their corresponding labels.
- Similar to this, the code loads and processes the pictures of the remaining celebrities, including **Kobe Bryant, Maria Sharapova, and Ronaldo**.
- The lists **rona_images, rona_labels, maria_images, maria_labels, kobe_images, and kobe_labels** contain the processed **images and labels.**

Data Preprocessing

- For the four celebrities **(Kane Williamson, Ronaldo, Maria Sharapova, and Kobe Bryant),** we combined their **labels and images into two NumPy arrays** called **all_labels and all_images**.

- The labels and images for each of the four celebrities are **concatenated** into a single array using the **np. concatenate ()** function.

- **All_images** contain the actual images, while **all_labels** contain the labels for each image **(i.e., the name of the celebrity).** This is a step in the process of getting the data ready for machine learning model's training.

- For Example, the label for the **300th image in the dataset**, which corresponds to the well-known **tennis player Maria Sharapova**, is printed when we use the **(all_labels [300])** output function.

- **Plt.imshow(all_images [300])** in the following line displays the **actual image** linked to that label. **Maria Sharapova** can be seen in the picture performing, possibly during a tennis match.

- This image is being shown to the reader to give them an idea of the type of data that is present in the dataset and to show them what these sports celebrities look like in their images.

- The number of rows in the **2D array all_images** is changed to be **441** to represent the total number of celebrity images, and the number of columns is determined by flattening each image to a **1D array of pixels**.

- The length of the flattened array is used to determine the number of columns automatically using the **-1** argument in the **reshape function**.

- The data must be **reshaped to be fed into a machine learning algorithm.**

Details about Target Column

- To **all_images dataframe**, we included a target column.
- The **names of the celebrities** associated with each image can be found in **all_labels list.**
- All the celebrities' images and labels were combined, and the **combined images were then reshaped into a 2D array**.
- The **target** column was then added to the dataframe by using **all_labels** list to create a pandas dataframe from the **reshaped images**.
- The **label** for the image classification model will be taken from this **target column**.
- The dataframe for all_images are divided into **input features and target variables**.
- The 'target' column from all_images dataframe is removed before input features are saved in the variable X.
- Using the **get_dummies ()** method from the pandas library, the target variable is **first stored in the variable y and then one-hot encoded**.
- Using a technique called **one-hot encoding**, **categorical variables** can be transformed into a **format that machine learning models can use to make better predictions**.

Training and Testing Data Set

- After preparing the dataset, the next step is to **split** the data into **training and testing sets**.

- This is necessary to **evaluate the performance of the model** on unseen data. We can use the **train_test_split function** from the **scikit-learn library** to split the data into training and testing sets.

- The train_test_split function randomly splits the dataset into a specified ratio of training and testing data.

- In this case, we can split the dataset into **70% training** data and **30% testing** data.

- Once we have the training and testing datasets, we can start building our **CNN model to classify the sports celebrity images.**

Convolutional Neural Network (CNN)

- A kind of **artificial neural network** called a **convolutional neural network (CNN)** is most frequently used to **analyze visual images**.

- For these reasons, the Convolutional Neural Network adopts a **different strategy and imitates how our eyes help us comprehend our surroundings**.

- When we perceive a picture, we instantly break it down into several smaller **sub-images and examine** each one separately. We analyze and interpret the image by putting these smaller images together.

- They have uses in the **recognition of images and videos**, **recommender systems**, **classification and segmentation of images**, **analysis of images used in medicine**, **natural language processing**, **brain-computer interfaces**, and **time series analysis of financial data**.

Conclusion

- Based on the provided code, a **convolutional neural network (CNN)** model was developed for **image classification** using images of **four different sports celebrities.**

- The model consisted of **several layers of convolutional**, **pooling**, and **fully connected layers**, and was trained using the **Adam optimizer** and categorical **cross entropy loss function**.

- The **accuracy and loss metrics** of the model were evaluated on a test set, and a plot of the model's loss was generated to visualize the training process.

- It suggests that the trained CNN model has a **loss of 1.723** and an accuracy of **57.14%** on the test dataset.

- The model has been trained to **classify images** of four different celebrities, and it **correctly identified** the celebrity in **57.14%** of the **test images.**

- However, there is scope for improvement, as the accuracy is not very high. This could be due to factors such as a **small dataset**, **insufficient training time**, or **suboptimal hyperparameters.**

- However, it is important to note that this is a simplified example and there are many ways to improve the performance of a CNN model, such as using **transfer learning**, **data augmentation**, and **hyperparameter tuning**.

- Additionally, **further testing** and **evaluation** of the model would be necessary to determine its **real-world performance** and **potential limitations**.

References

Gautam, T. (2021). Create your Own Image Classification Model using Python and Keras. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/10/create-image-classification-model-python-keras/

GeeksforGeeks. (2023). Image Classifier using CNN. GeeksforGeeks. https://www.geeksforgeeks.org/image-classifier-using-cnn/

Sport Celebrity Image Classification. (2021, November 26). Kaggle. https://www.kaggle.com/datasets/yaswanthgali/sport-celebrity-image-classification/code