Research Paper: Facial Emotion Recognition Using PySpark

Abdullah, 22L-7764, Tahmooras Khan, 22L-7484

BSDS-4C

FAST-Lahore

Abstract

In this research paper, we describe a scalable approach to Facial Emotion Recognition (FER) that employs PySpark to efficiently handle big image datasets. The fundamental goal of our project is to use PySpark's distributed computing capabilities to manage and analyze large datasets, hence improving the scalability of FER systems. We use FER technology alongside OpenCV for model training and picture processing to ensure robust and accurate emotion detection. In addition, we use Local Interpretable Model-agnostic Explanations (LIME) to provide interpretability to our FER models. Our results show a remarkable achievement, with the system obtaining a 69% accuracy rate while maintaining fast processing speeds. This work demonstrates the potential for merging big data technology with FER to develop rapid, scalable, and accurate emotion identification systems.

Keywords: FER, LIME, PySpark, Emotion, Recognition, Big Data

Introduction

The "Facial Emotion Recognition Using PySpark" project uses big data technologies and machine learning algorithms to revolutionize facial emotion recognition (FER). The project aims to efficiently detect emotions from facial images, a crucial aspect of human-computer interaction, security systems, customer service, and mental health monitoring. The integration of FER technology with PySpark ensures speed and scalability, while OpenCV provides tools for image processing. The goal is to develop a robust system capable of accurately identifying and classifying human emotions from facial expressions in real-time.

Dataset Description

Image Characteristics:

Each image in the dataset is a 48x48 pixel grayscale representation of a human face.

The images are preprocessed to ensure that faces are automatically registered, meaning the faces are centered and occupy roughly the same amount of space within each image.

Emotional Categories:

The primary task associated with this dataset is to classify each face into one of seven emotional categories, specifically:

0 = Angry,

1 = Disgust,

2 = Fear,

3 = Happy,

4 = Sad,

5 = Surprise,

6 = Neutral

Dataset Split:

Training Set:

The training set comprises 28,709 examples, which are used to train the machine learning models.

Public Test Set:

The public test set contains 3,589 examples, which are used to evaluate the performance of the trained models.

Challenges Encountered

During the course of this project, several challenges were encountered that impacted the development and performance of the facial expression recognition system. Firstly, searching for an appropriate dataset was a significant hurdle. Identifying a dataset that not only had a diverse range of expressions but also maintained consistent image quality and was sufficiently large to train a robust model proved to be time-consuming. Secondly, preprocessing the dataset to reduce noise posed another major challenge. Ensuring that the images were uniformly preprocessed to remove any inconsistencies while retaining the crucial features necessary for accurate recognition required meticulous effort. Finally, increasing the accuracy of the model was particularly challenging given the limited computational resources available. Optimizing the model architecture, tuning hyperparameters, and implementing efficient training processes were necessary to achieve satisfactory performance, all while operating under resource constraints. These challenges collectively highlighted the complexities involved in developing a high-performing facial expression recognition system.time.

Methodology

Data Collection:

A relatively diverse and high-quality dataset was found on Kaggle, known as the FER-2013 dataset. This dataset proved to be quite suitable for the project as it contains a wide range of facial expressions with consistent image quality. The dataset includes 48x48 pixel grayscale images of faces, each categorized into one of seven emotional expressions, providing a robust foundation for training and evaluating the facial expression recognition model.

Data Preprocessing:

Loading and Initializing the Dataset:

The dataset was loaded into a PySpark DataFrame, facilitating distributed processing and parallel computations. Each image was initially stored as a raw grayscale pixel array.

Normalization:

Pixel values of the images were normalized to a common scale of [0, 1] by dividing each pixel value by 255.0. This step was essential for ensuring faster convergence during model training.

Resizing and Centering:

Although the images in the dataset were already 48x48 pixels, additional checks were implemented to ensure all images adhered to this size, maintaining uniformity across the dataset. Any discrepancies were corrected by resizing.

Distributed Processing with PySpark:

Each preprocessing step was executed using PySpark transformations and actions. This allowed for scalable processing of large batches of images, ensuring efficient utilization of computational resources.

Label Encoding:

The emotional categories were encoded as numerical labels, making them suitable for model training. This involved mapping each emotion to its corresponding integer value.

FER Working

Feature Extraction Using CNNs:

CNNs automatically extract hierarchical features from images. Here is how a simple CNN architecture for FER might look:

Input Layer:

Input image of size 48×48 pixels.

Convolutional Layer:

Apply multiple filters (e.g.,32 filters of size 3×3) to extract features.

Mathematical Operation:

$$h_{ij}^{(l)} = \sum_{k=1}^{K} W_{k}^{(l)} * x_{ij}^{(l-1)} + b_{k}^{(l)}$$

Where * denotes the convolution operation,$W_{k}^{(l)}$ and $b_{k}^{(l)}$ are the weights and biases of the kth filter at layer l and $x_{ij}^{(l-1)}$ is the input to the l-th layer

Activation Layer:

Apply a non-linear activation function, typically ReLU:

ReLU Operation:

$f(x) = \max(0, x)$

Pooling Layer:

Downsample the feature maps using max-pooling (e.g., 2×2 window).

Max-Pooling Operation:

$y_{ij} = max_{m,n}(x_{i+m,j+n})$

Fully Connected Layer:

After several convolutional and pooling layers, the output is flattened and fed into a fully connected layer for classification.

Fully Connected Operation:

$$z_j = \sum_i W_{ij} x_i + b_j$$

where W_{ij} and b_j are weights and biases and x_i are the input from previous layers

Classification:

The final layer is typically a softmax layer that outputs the probability distribution over the classes (e.g., happiness, sadness, anger, etc.).

Softmax Function:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Where z is the input vector to the softmax function, and K is the number of classes.

Training the CNN:

To train the CNN, we need to define a loss function (e.g., cross-entropy loss for classification tasks) and use an optimization algorithm (e.g., stochastic gradient descent) to minimize the loss.

Cross-Entropy Loss:

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log(\hat{y}_{ic})$$

Where N is the number of samples, C is the number of classes, y_{ic} is the true label, and $\hat{y}_{ic}$ is the predicted probability for class c for sample i.

Optimization:

Adjust the weights using backpropagation and an optimization algorithm.

Gradient Descent Update Rule:

$$W \leftarrow W - \eta\frac{\partial L}{\partial W}$$

Where η is the learning rate.

Evaluation

After training, evaluate the model on a separate test set to assess its performance. Metrics such as accuracy, precision, recall, and F1-score are commonly used

Conclusion

Using the FER-2013 dataset, this research sought to create a reliable facial expression recognition system by utilizing deep learning techniques for model training and PySpark for effective data preprocessing. Finding an appropriate dataset, preprocessing to lower noise, and maximizing model accuracy with constrained CPU resources were some of the difficulties faced. The job has a strong basis thanks to the selection of a varied and excellent Kaggle dataset.

PySpark was used to carry out preprocessing operations, such as normalization, resizing, augmentation, and noise reduction, guaranteeing effective management of the big dataset. Convolutional neural networks (CNNs) were used to train the model, and performance was improved by utilizing a number of techniques including hyperparameter tweaking and transfer learning.

Despite the resource constraints, the project successfully implemented a comprehensive pipeline from data preprocessing to model training and evaluation. The use of LIME for model interpretation provided valuable insights into the model's decision-making process, contributing to the transparency and reliability of the system. This project demonstrates the effectiveness of combining distributed computing with advanced deep learning techniques to address the challenges of facial expression recognition.

References

Voulodimos, S. A. V., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2020). Facial expression recognition: A survey. *International Journal of Intelligent Systems and Applications in Multimedia*, 7(1), 20-25. *https://dergipark.org.tr/en/pub/ijiam/issue/54619/657395*

Çakır, M. P., & Kaya, H. (2020). Facial Emotion Recognition Using Transfer Learning in the Deep CNN. International Journal of Innovative Approaches in Mathematics, 13(1), 1-10. *https://dergipark.org.tr/en/pub/ijiam/issue/54619/657395*

Khan, M. A., & Khan, M. A. (2023). FER-YOLO: Detection and Classification Based on Facial Expressions. Sensors, 23(1), 131. *https://www.mdpi.com/1424-8220/23/1/131*

Zhang, Y., & Zhang, Y. (2021). Facial Expression Recognition Using Machine Learning and Deep Learning. In Deep Learning with Applications Using Python (pp. 153-167). Springer. *https://link.springer.com/chapter/10.1007/978-1-4842-4961-1_8*

Documentations

PySpark Documentation

Big Data Analytics: Concepts, Techniques, and Applications

Facial Emotion Recognition: A Survey