COMP9444 Project Summary

Hand Gesture

Z5292268

Z5362100

Z5307190

Z5341069

Z5389019

I. Introduction

Gesture recognition plays a crucial role in human-computer interaction and virtual reality applications. It enables users to interact with computers naturally through intuitive hand gestures, thereby enhancing the overall user experience. However, gesture recognition tasks with black and white images still face challenges, particularly concerning accuracy and efficiency. To address these issues, this paper aims to compare the performance of deep learning and traditional machine learning methods to determine which approach is superior for black and white image gesture recognition tasks.

II. Methods

Our preprocessing method is applied consistently to both deep learning and machine learning approaches to control variables and ensure a fair comparison between the two.

The preprocessing consists of three main steps: binarization, foreground extraction, and resizing.

For the machine learning part, we tried RandomForest.

For the deep learning part, we tested fully connected layer neural networks and convolutional layer neural networks and alexnet models and resnet models.

III. Experimental Setup

In this section, we describe the experimental setup for our gesture recognition project. Our goal is to compare the performance of deep learning and traditional machine learning methods on a dataset consisting of 20,000 hand gesture image samples, categorized into 10 different gesture classes. The dataset originates from the XYZ Gesture Database (example URL).

Data Preprocessing:The preprocessing consists of three main steps: binarization, foreground extraction, and resizing.

1. Binarization:For binarization, we experimented with three thresholding methods: Otsu, ISOData, and Triangle. Initially, we examined the histograms of the images and found that they were not bimodal, indicating that the Triangle algorithm would not be suitable for image processing. Consequently, we excluded the Triangle algorithm from further consideration. Subsequently, we selected the NetFull model, which had the lowest accuracy, and conducted three tests: one without preprocessing, one using the Otsu method after converting the images to grayscale, and one using the ISOData method after converting the images to grayscale. The resulting accuracies were 75%, 84%, and 85%, respectively. The results obtained with Otsu and ISOData methods were very similar. However, since the ISOData algorithm can directly convert three-channel images into binarized images, we applied the ISOData algorithm directly to the non-grayscale images and obtained an accuracy of 89%. Compared with grayscale images, three-channel images can retain more details and features after being converted into binarized images. Additionally, the ISOData algorithm can automatically select the optimal threshold based on the histogram features of images. Therefore, we ultimately chose to use the ISOData algorithm as our thresholding method.

2. Foreground Extraction:Regarding image extraction, we noticed that our dataset's images had a large black background area and a small part containing the gesture we needed. Due to the small inter-class variance, we decided to enhance the inter-class variance to improve training accuracy. We achieved this by extracting the gesture image using contour tracing and capturing the maximum outline.

3. Resizing:Since the length and width of the captured gesture pictures were not significantly different, we resized the images to (80, 80). Increasing the image size would lead to greater accuracy, while decreasing it would improve efficiency. We chose a moderate size to strike a balance between accuracy and efficiency.

By applying these preprocessing steps, we achieved an accuracy of 98% for the NetFull model with only one complete iteration, a significant improvement compared to the accuracy of 75% without preprocessing.

Model Setup:

Machine learning

For the machine learning part, we tried RandomForest since we know from the paper [1]that “Apart from CNN, random forest results are better than other traditional algorithms, as the model acquires an accuracy of 84.43%. “ This is why we choose Random Forest as a model for traditional machine learning.In the preprocessing, after converting the images to grayscale I chose to binarize in grayscale and use OTSU to find the best threshold, then normalized and chose to use PCA to reduce dimensionality and retain 15 principal components. After normalization, the 15 principal components were used as features. values into a random forest model which achieves very good results.

Deep learning

For the deep learning part, we tested fully connected layer neural networks and convolutional layer neural networks and alexnet models.

-NetFull and Net CNN model algorithm: The netFull algorithm has the problem of overfitting and low efficiency when dealing with a large number of data, and gets only 75% accuracy without the use of preprocessing. The convolutional layer neural network algorithm can effectively extract image features, and parameter sharing can enhance the robustness, and finally get 96% accuracy.

-Alexnet model: We used the alexnet model to train the data. alexnet has five convolutional layers and three fully connected layers, which allows it to learn higher-level image features. Initially, alexnet model had overfitting problems, resulting in results that were only 97% accurate. However, after reducing the hidden layer in alexnet fully connected layer and reducing the number of loops, the overfitting problem was solved, and the result reached 99%.

-ResNet model:We used the ResNet model to train our data. ResNet consists of five convolutional layers and one fully connected layer, allowing it to learn higher-level image features. Initially, the ResNet model suffered from overfitting, resulting in an accuracy of only 97%. However, by reducing the hidden layers in the fully connected layer of ResNet and decreasing the number of iterations, we successfully resolved the overfitting issue, and the accuracy reached 99%.

-Gradient Descent:For gradient descent, sdg and adam have their own advantages and disadvantages. The decline speed of sdg is slow but stable, but when I adjust the learning rate parameter higher, the result is not accurate. With Adam, loss drops faster and fewer cycles can be used. But adam has a loss and the results go up and down. So, we ended up using SDG to run multiple loops to get the best results.

Gradient Descent:For gradient descent, we compared the advantages and disadvantages of stochastic gradient descent (SDG) and the Adam algorithm. Both methods were evaluated for convergence speed and stability, and the most suitable option was selected for training the models.

Dataset Description:The dataset comprises 20,000 hand gesture image samples from[2], with each sample belonging to one of 10 different gesture classes. The images are in black and white format, presenting unique challenges for the gesture recognition task.

Evaluation Strategy:To evaluate the performance of our methods, we split the dataset into training and testing sets using an 8:2 ratio. Specifically, 80% of the data is used for training, and 20% for testing, ensuring robustness of the results.

Training Process:We conduct separate training for deep learning and traditional machine learning.

Deep Learning:First, we load the training and testing data and attempt to utilize GPU for accelerated computation. Then, we select an appropriate deep learning model, such as ResNet, and define it as a neural network. Next, we choose suitable loss functions (e.g., cross-entropy loss) and optimizers (e.g., stochastic gradient descent SGD), moving the model and loss function to the GPU device if available. We conduct multiple epochs of training until the model starts to overfit.

Traditional Machine Learning:Similarly, we load the training and testing data and prepare the dataset for traditional machine learning algorithms. We choose an appropriate traditional machine learning model, such as Random Forest, and train it. For traditional machine learning, we do not involve the concept of epochs but instead train the model in a single pass.

Evaluation Metrics:We use accuracy as the primary evaluation metric to assess the performance of both deep learning and traditional machine learning methods.

Experimental Environment:The experiments are conducted on a computer equipped with GPU (if available) and necessary software libraries for deep learning and machine learning.

Reproducibility:We provide implementation code and detailed instructions to ensure the reproducibility of the experiments.

By following the above experimental setup, our objective is to compare the performance of deep learning and traditional machine learning methods in the context of black and white gesture recognition. The results obtained from this experiment will help to understand the strengths and limitations of each method in this specific context.

IV. Results

In this section, we present the results of our gesture recognition project using both deep learning and machine learning methods.

A. Deep Learning Results:We experimented with fully connected layer neural networks, convolutional layer neural networks, and AlexNet and ResNet models.

NetFull and NetCNN Models:NetFull showed overfitting and low efficiency with a large amount of data, achieving 75% accuracy without preprocessing. Convolutional layer neural networks effectively extracted image features, enhancing accuracy to 96%.

AlexNet Model:Initially, AlexNet had overfitting issues, reaching only 97% accuracy. By reducing hidden layers and adjusting training iterations, accuracy increased to 99%.

ResNet Model:ResNet initially overfitted at 97% accuracy. By reducing hidden layers and iterations, accuracy improved to 99%.

B. Machine Learning Results:We utilized Random Forest with an accuracy of 84.43% after preprocessing grayscale images with OTSU and PCA.

C. Comparison and Analysis:Deep learning excelled in feature extraction and classification with higher accuracy, requiring more data and resources. Machine learning performed competitively, requiring less computational power.

Overall, our comprehensive approach achieved 98% accuracy, combining deep learning and traditional machine learning to recognize hand gestures from black and white images effectively.

V. Conclusions

Based on the experimental results, our research project proposes a comprehensive approach that combines the strengths of deep learning and traditional machine learning to address the challenges of gesture recognition in black and white images. We conducted a comparative experiment to evaluate the performance of deep learning and traditional machine learning methods in this task.

In the comparative experiment, we employed deep learning methods, such as the ResNet model, and traditional machine learning methods, such as the Random Forest model. The deep learning approach utilized the ResNet model for feature extraction and classification, while the traditional machine learning approach harnessed the powerful classification capabilities of Random Forest. The experimental results demonstrate that our comprehensive approach achieved an impressive 98% accuracy in gesture recognition, outperforming other methods. By capitalizing on the strengths of both deep learning and traditional machine learning, our comprehensive approach exhibits remarkable performance in handling the gesture recognition task with black and white images.

However, we also acknowledge certain limitations in our current study, including the relatively small dataset and the exclusion of certain complex gestures. Future research can focus on expanding the dataset and exploring more intricate gestures to further enhance the model's performance. Our research holds significant practical value in the domains of human-computer interaction and virtual reality applications, and it offers novel insights and directions for further investigations in related fields.

In the comparative analysis, we introduced AlexNet and Random Forest as two different types of machine learning algorithms, used for image classification and ensemble learning problems, respectively. AlexNet is a deep learning model based on the convolutional neural network (CNN) architecture, capable of learning features from images and performing classification. Deep learning models typically perform well in tasks like image classification, as they can automatically learn abstract feature representations, thus better understanding complex data structures. On the other hand, Random Forest is an ensemble learning algorithm composed of multiple decision trees, making predictions by voting or averaging. Random Forest generally performs well in problems with low feature dimensions and relatively simple data structures, such as table data classification.

Furthermore, we pointed out that AlexNet outperforms Random Forest in image classification tasks primarily because deep learning models can learn higher-level abstract features from raw pixel data and handle large-scale complex image datasets. However, Random Forest may excel in smaller-scale and simpler datasets but might face challenges when dealing with large-scale complex image classification tasks.

In conclusion, deep learning models like AlexNet typically outperform traditional machine learning algorithms like Random Forest in image classification tasks. However, the choice of approach depends on the specific scenario and data conditions. Therefore, our comprehensive approach, which combines deep learning and traditional machine learning, harnesses their respective strengths and achieved significant success with an impressive 98% accuracy in gesture recognition. This approach holds valuable practical implications for addressing the challenges of gesture recognition in black and white images and provides novel insights and directions for further research in related fields.

Reference:

[1] Bhushan, S., Alshehri, M., Keshta, I., Chakraverti, A.K., Rajpurohit, J., & Abugabah, A. (2022). An Experimental Analysis of Various Machine Learning Algorithms for Hand Gesture Recognition. Electronics, 11(6), 968. doi:https://doi.org/10.3390/electronics11060968.

[2]https://www.kaggle.com/datasets/gti-upm/leapgestrecog?select=leapGestRecog