UNIVERSITY OF TECHNOLOGY, SYDNEY

2024

Deep Learning and Convolutional Neural Network

Handwritten Mathematical Symbol Recognition Using Support Vector Machines and Neural Networks with Different Feature Extraction Methods

Monali Patil

Introduction:

The project involves handwritten mathematical symbol recognition and classification aimed at developing machine learning models capable of accurately recognizing and categorizing symbols commonly used in mathematical notation.

It involves building models with 6 experiments utilizing two specific classifiers with 3 feature extraction techniques from the handwritten symbol images as mentioned below.

Classifier Models:

- Support Vector Machines (SVM)
- Artificial Neural Networks (ANN)

Feature Extraction Techniques:

- Histogram-Of-Oriented-Gradients (HoG)
- Local Binary Pattern (LBP)
- Raw image/pixels

Thus, each model is constructed by extracting relevant features from the handwritten symbol images through the above three methods, resulting in a total of 6 experiments.

Dataset:

The given dataset comprises handwritten images of 10 various mathematical symbols. These symbols are stored in separate folders, each corresponding to a specific symbol category.

Image resolution: (45, 45) and Image Datatype: uint8

The data type of the image is uint8, which stands for unsigned 8-bit integers representing the intensity value of a pixel in the image. This data type is commonly used to represent pixel values in images.

The images have a pixel size of (45, 45) which is essentially a collection of 2D arrays, where each array represents a grey image of a math symbol.

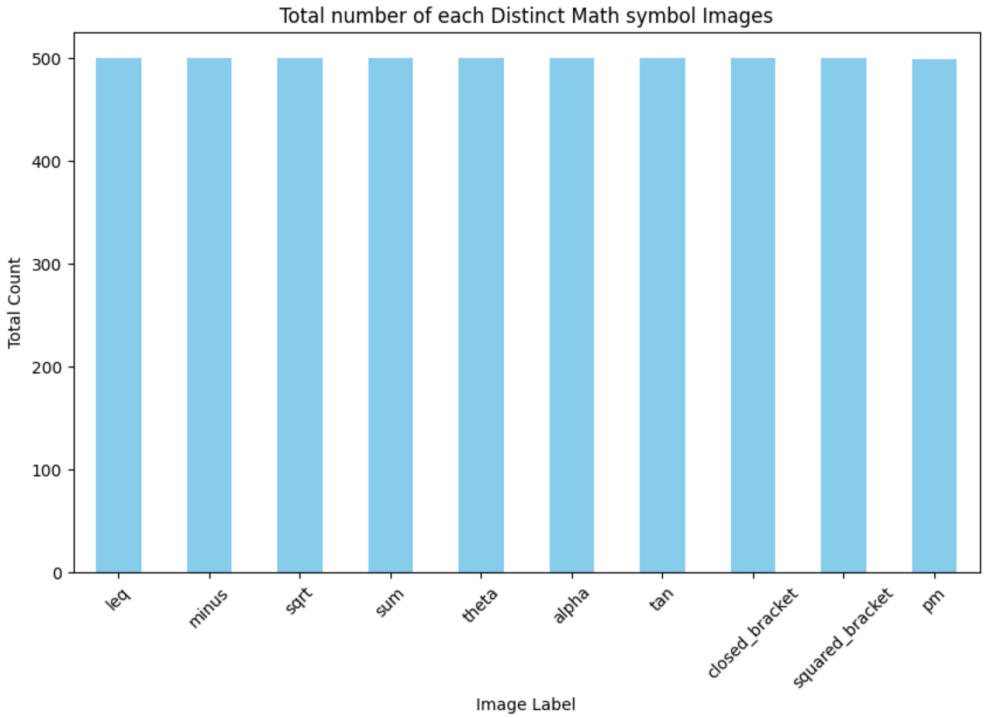

Figure 1. Bar graph informing a total number of unique images.

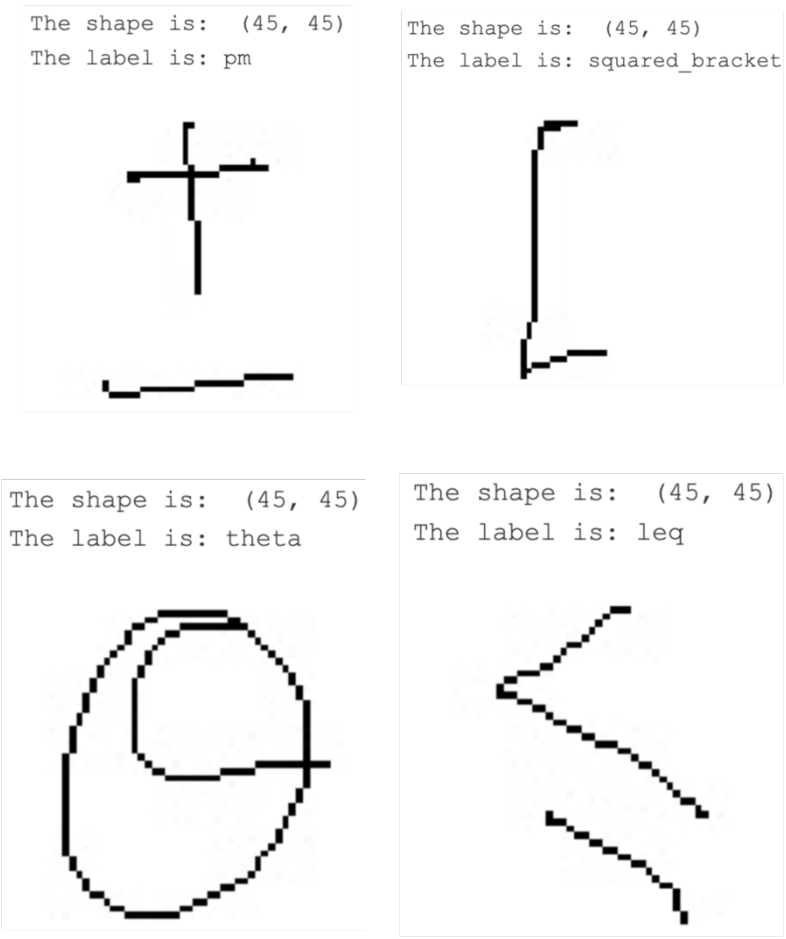

Figure 2. Few Images of some unique symbols.

As shown in figure 1 the given dataset consists of a total of 4999 handwritten math symbol images and a few of them are illustrated in figure 2.

Note: For detailed dataset information please refer to section 2: Visualising and Exploring images from the Assignment1_Image_Preprocessing.ipynb notebook.

Experimental Settings:

The table below provides details regarding the configuration of classifiers used in various experiments conducted for this image classification task.

Python NoteBook	Experiments	Feature Extraction	Classifier	Hyper Parameter/Framework
Assignment1_LBP_Feature.ipynb	Experiment 1 (Exercise 1.1)	LBP	SVM	kernel='rbf', C=15.0
	Experiment 1 (Exercise 1.2)			kernel='rbf', C=65.0
	Experiment 2 (Exercise 2.1)	LBP	ANN	Input Neurons: 26, Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: Adam
	Experiment 2 (Exercise 2.2)			Input Neurons: 26, Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: SDG
Assignment1_HOG_Features.ipynb	Experiment 1 (Exercise 1.1)	HOG	SVM	kernel='rbf', C=65.0 and extracted HOG features with parameters orientations=9, pixels_per_cell=(8, 8), cells_per_block=(3, 3)
	Experiment 1 (Exercise 1.2)			kernel='rbf', C=100.0 and extracted HOG features with parameters orientations=9, pixels_per_cell=(10, 10), cells_per_block=(2, 2)
	Experiment 2 (Exercise 2.1)	HOG	ANN	Input Neurons: 324, Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: Adam
	Experiment 2 (Exercise 2.2)			Input Neurons: 324, Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: SDG
Assignment1_RawPixels_Feature.ipynb	Experiment 1.1	Raw Pixels	SVM	kernel='rbf', C=65.0
	Experiment 2 (Exercise 2.1)	Raw Pixels	ANN	Input Neurons: 2025 flatten layer (45, 45), Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: Adam
	Experiment 2 (Exercise 2.2)			Input Neurons: 2025 flatten layer (45, 45), Activation function: ReLU (Rectified Linear Unit) for the hidden layers and Softmax in the output layer, Loss function: Sparse categorical cross-entropy, Output layer: 10 neurons and Optimiser: SDG

Experimental Results:

The below table presents the highest accuracy score for each feature extraction technique and employed classifier model.

Classifier/Feature	LBP	HOG	Raw Input
SVM	Train: 74.36 Test: 74.00	Train: 100 Test: 99.47	Train: 100 Test: 97.73
ANN	Train: 51.21 Test: 53.27	Train: 99.89 Test: 99.13	Train: 93.86 Test: 89.13

Below are the confusion matrixes for these highest accuracy scores classifiers and features informing how well the classifier model performed for each class.

Note: For detailed information on the confusion matrix please refer to LBP, HOG and RawPixels notebook's Modelling (3rd) section.

1] LBP features and SVM classifier.

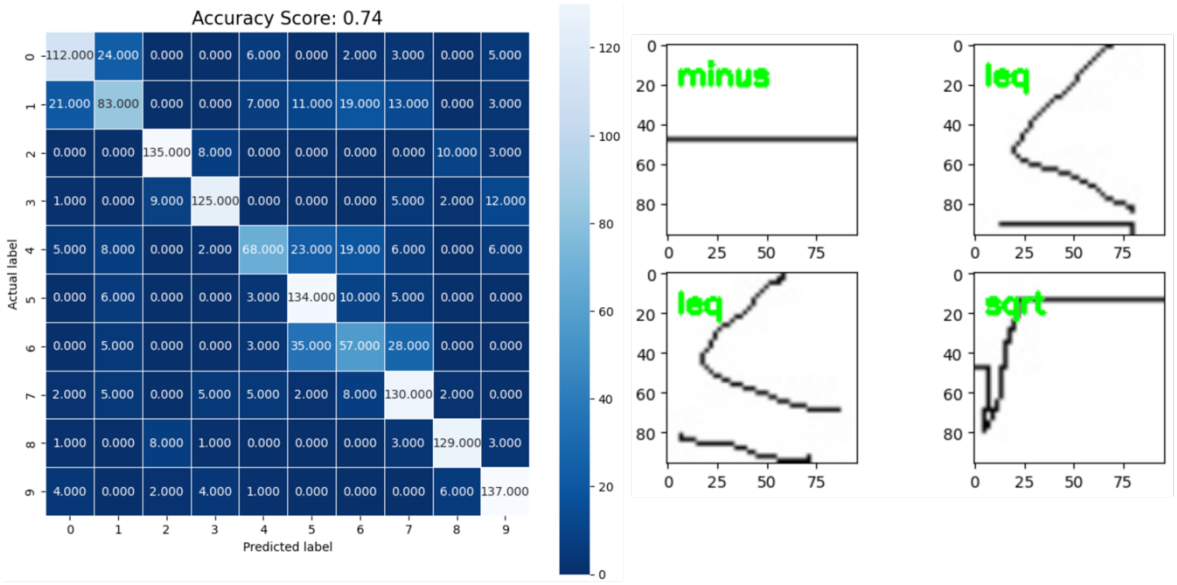

Figure 3: Confusion matrix and Classification results for LBP features and SVM classifier

In Figure 3's confusion matrix, class 6 ("alpha") is often misclassified as other symbols: 35 times as class 5 ("theta"), 28 times as class 7 ("tan"), 5 times as class 1 ("leq"), and 3 times as class 4 ("sum"). This suggests visual similarities between

"alpha" and these symbols, posing challenges for accurate differentiation, likely due to shared visual features or complexities in their representations.

Additionally, the classification result informs a selection of images chosen at random, all of which have been accurately classified by the SVM classifier utilizing LBP-extracted features.

2] HOG features and ANN classifier.

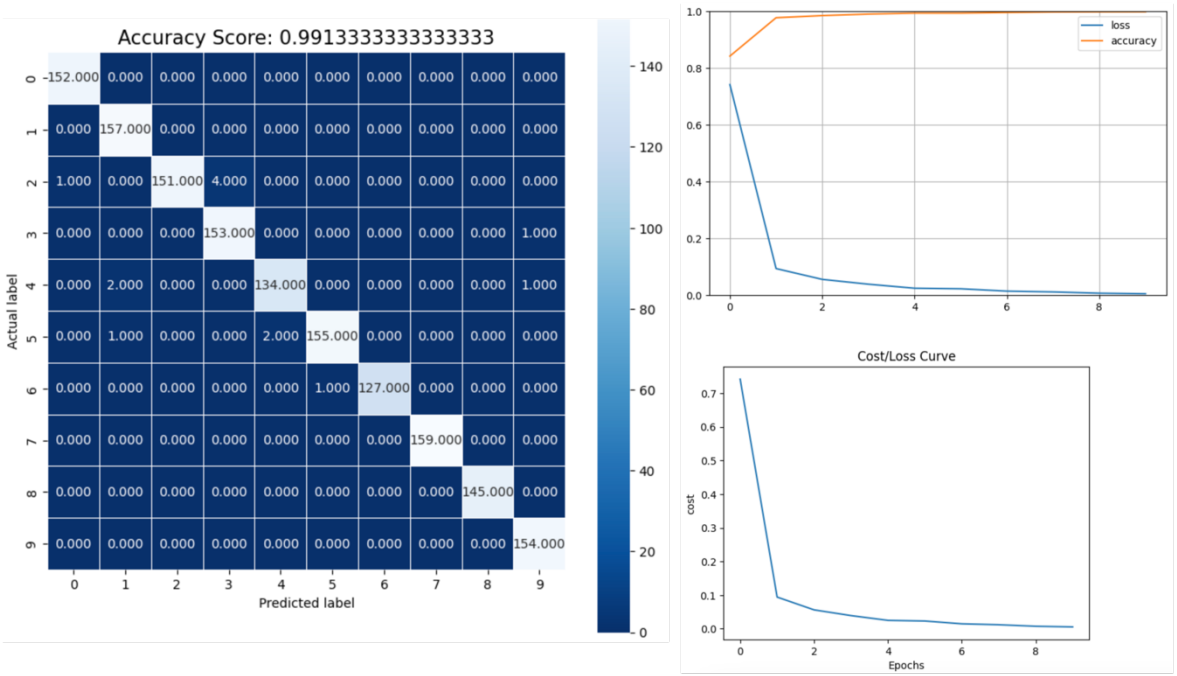

Figure 4: Confusion matrix and Cost/Loss curve for HOG features and ANN classifier

In Figure 4, class 5 ("theta") is misclassified twice as class 4 ("sum") and once as class 1 ("leq"). These misclassifications may stem from visual similarities, such as circular shapes or specific orientations, shared by "theta" and "sum" symbols.

Overall, the model demonstrates strong performance, where no misidentifications occur for classes 0 (pm), 1 (leq), 7 (tan), 8 (closed_bracket), and 9 (squared_bracket). Most predictions closely match the actual labels, indicated by high values along the diagonal, signifying accurate classifications.

Additionally, the loss curve depicts a decreasing trend with increasing epochs, while accuracy rises and eventually stabilizes.

Note: For detailed information on the confusion matrix please refer to LBP, HOG and RawPixels notebook's Modelling (3rd) section.

Models Summary:

HOG features outperform LBP features and Raw pixels in all combinations due to their ability to capture shape and structure information more effectively.

HOG features outperform LBP features in all combinations due to their ability to capture shape and structure information more effectively. It calculates the distribution of gradient orientations in an image, which helps in capturing local shape and edge information.

While LBP is a texture descriptor used for texture analysis in images. It quantifies the local structure of an image by comparing each pixel with its neighbouring pixels. The Raw pixel values refer to the intensity values of individual pixels in an image.

Thus, HOG features are well-suited for capturing shape and edge information, and SVM classifiers perform excellently in high-dimensional feature spaces. The combination of HOG features and SVM classifier results in near-perfect accuracy.

Similar to the HOG and SVM combination, HOG features provide rich information about shape and structure, allowing the ANN model to effectively learn and generalize, leading to high accuracy scores.

Note: Please refer to LBP, HOG and RawPixels notebook's Modelling (3rd) section for a detailed description of the errors in classification and accuracy scores.

Conclusion

Based on the conducted experiments, it can be stated that HOG features provide a more distinctive representation of the shape and edge information in the handwritten symbols. By focusing on gradients and edge orientations, HOG features can effectively capture the unique characteristics of different symbols, making them more efficient for this classification task involving handwritten symbols compared to LBP and raw pixel values.