Handwritten Mathematical Symbols Classification

Jagan Dwarampudi
Department of CISE
University of Florida
Gainesville, Florida
jdwarampudi@ufl.edu

Vamsi Panchamatla
Department of CISE
University of Florida
Gainesville, Florida
vamsi.pachamatla@ufl.edu

Yashas Kuchimanchi
Department of CISE
University of Florida
Gainesville, Florida
kuchimanchi.s@ufl.edu

Veera Nitish Mattaparthi
Department of CISE
University of Florida
Gainesville, Florida
veeramattaparthii@ufl.edu

Abstract

Automatic handwritten mathematical symbols classification is essential to developing computer-aided systems in many ways. It serves as the cornerstone for mastering one of the trickiest jobs ever, like reading handwritten mathematical calculations. HMC encounters several challenges while attempting to categorize photos accurately, much like other comparable automated handwritten character classification jobs. We classified ten different handwritten symbols with the help of convolution neural networks. We implemented tensor flow and Keras and tested them on massive datasets. We managed to classify the symbols with an accuracy of 90.55%.

Keywords—Convolution neural networks, KNN, machine learning algorithms, Keras, support vector machine

INTRODUCTION

The project's objective is to build a machine-learning model that can correctly recognize a collection of mathematical symbols. It was accomplished by the application of three fundamental concepts.

1. Support vector machines:

SVM's are one of the powerful classification tools when it comes to applications in machine learning. Being an unsupervised approach, it implements a boundary between data in a data set facilitating faster label prediction using feature vectors.

2. Convolution neural networks:

CNN (convolutional neural networks) is a visual image processing and classification algorithm. By this approach we can achieve highly accurate classification as seen in the paper 'Recognition of Basic Handwritten Math Symbols Using Convolutional Neural Network with Data Augmentation' [1]. They have worked with three available datasets using the CNN model and with different augmentation techniques over 11 layers. After implementing this approach on three different datasets, they achieved an accuracy of 98.71%, 99.01%, 99.85% on respective datasets.

We have implemented multiclass logistic regression, support vector machines, and convolution neural networks with Keras, among all these, multiclass logistic regression performed worst with 60% accuracy while convolution neural networks performed best with the highest accuracy of 90.55%.

I. IMPLEMENTATION

We are implementing our project approach using a large data set with various symbols to be categorized by the code. The dataset is created by the team and includes 10 symbol patterns to match and classify. Each team member contributed to about 10 images of varied forms of each symbol, I.e., 100 symbols by a single team member. We have used about 400 images to be classified by the algorithm. Each image in the data set can be characterized into any one of the symbols given below. Each symbol has three characteristic parameters. These parameters are Math Symbols, Label and Integer Encoding. Math symbol is the symbol pattern that needs to be matched by the algorithm with the data to classify. The label is the character equivalent of each symbol, and Integer Encoding is an integer digit assigned to each symbol, and the values range from 0 to 9, as shown in fig1.

Math Symbol	Label	Integer Encoding
x	x	0
$\surd$	square root	1
$+$	plus sign	2
-	negative sign	3
$=$	equal	4
$\%$	percent	5
∂	partial	6
$\prod$	product	7
π	pi	8
$\sum$	summation	9

Figure 1 Dataset labels.

The data was pre-processed by slitting it into various labels, as the data is already assigned with a label the pre-processing work is done and the data is shuffled and will split between training, testing, and validation dataset using the module of test_train_split in Sklearn. Now the data must be running through a model to recognize what type of class the image belongs to. For this, the accuracy of the model must be more than 90.55%, so CNN is chosen to execute this task. For this, the model was implemented using CNN in which we are using transfer learning so that the weights of the model will converge more quickly and efficiently as we are using a well-established transfer learning model. ResNet50 is used as the transfer learning module here in our case, as we can now train the model with a better convergence rate.

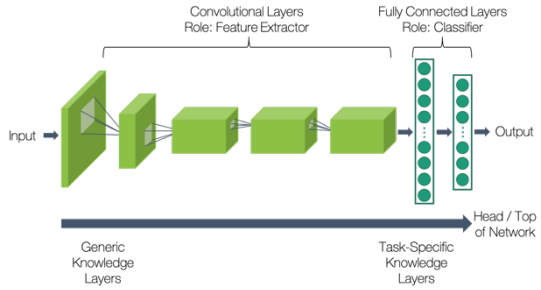

Figure 2 Resnet50 transfer learning model.

After the application of the ResNet50 transfer learning model then optimized values from that network o ResNet50 are passed through a global average polling layer of 2 dimensions, after that the resulting values will be passed through a dense layer of 500 nodes in which the classes can communicate between each other to look at what weight is better suited for the following data point and given label, then we are applying batch normalization on our data batch which is of size 16. We are using the Relu activation function to prevent non-negative values that are smaller than -2 so that we can normalize the weight without taking the outlier's negative values into our model. To prevent overfitting, we are dropping out some of the weight so that the model doesn't overfit and does not show bias towards the training data. We are repeating this process after ResNet50 with reduced sizes of the dense layer of 500,250,100 and moving towards 50 nodes in the dense layer. After that, we are applying the SoftMax function to the resulting values produced by our perceptron so that the probabilities of the predicted classes for a value would be equal to one, and then we can take the class which has the highest probability which is predicted by our model.

II. Experiments

A. Support Vector Machines:

Using the SVM in the model to predict the labels is extremely inefficient because we are using the principal components and dividing them based on the label of trained data using (n-1) dimensional vector. Dividing the data points with an n-1 vector will leave a lot of points untended in the dataset as they need to be separated correctly using the SVM. Hence, the model only yielded 68% accuracy in predicting the correct labels on the validation dataset.

B. CNN:

1. Hyperparameter tuning

The hyperparameters are chosen by experimenting with different hyperparameters like batch size and varied sizes of networks in hidden layers with padding, striding, and averaging pooling to reduce the network size. The network of 500 nodes of the dense layer with batch normalization and activation layer of Relu is working the best after manually experimenting with those hyperparameters. The batch size of 16 is used to maintain optimal processing and train the model in a time-efficient mode.

2. Transfer Learning (ResNet50):

In order to detect and classify the various patterns of each symbol, we use various layers to train the machine learning model. Adding layers helps in classifying easily as each layer can store various aspects of the symbol, like one layer to detect edges, one for texture, one to detect objects, and such. Even though an increase in these layers may slow down the training process and there is a possibility of vanishing gradient problems, we are implementing ResNet50 to increase the efficiency and accuracy of the algorithm. ResNet50 includes a block system called residues made of skip connections. As gradient flows across through alternate paths provided by the ResNet50 model, the vanishing gradient issue can be mitigated. The residual blocks make sure that the performance of dense networks is on par with the shallow network despite adding new layers. This allows the ResNet50 to generate faster results compared to all other training models.

C. Results:

The results from the algorithm are plotted as below graph with the number of epochs on X-axis and the loss or penalty from bad prediction on Y-axis. We can observe in the graph that initially the loss is at its maximum at the start, but as the training progresses, there is a steady decline in losses reaching a minimum.

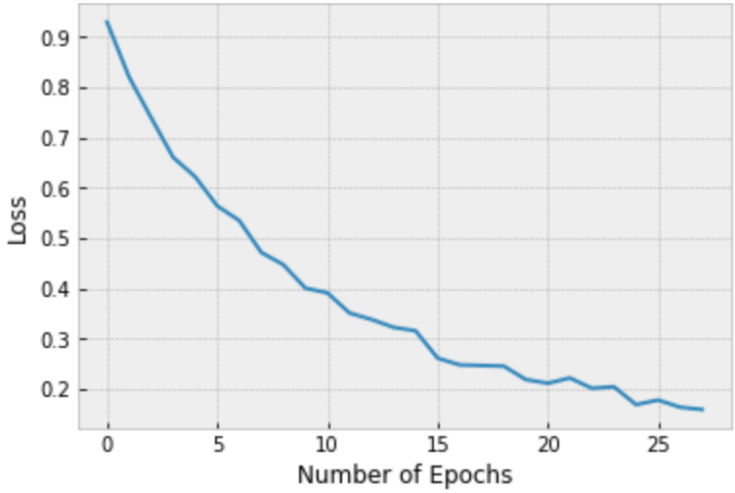

Figure 3 Plot between the number of epochs and loss.

The results regarding the prediction accuracy for validations that are randomly denoted by val_accuracy is plotted on Y-axis with each point difference of 0.01 and the number of epochs are plotted on X-axis with a point difference of 5 epochs. We can see from the graph that the val_accuracy does not follow a set pattern, and even though its value shows an overall increase, there are various intervals of decrease.

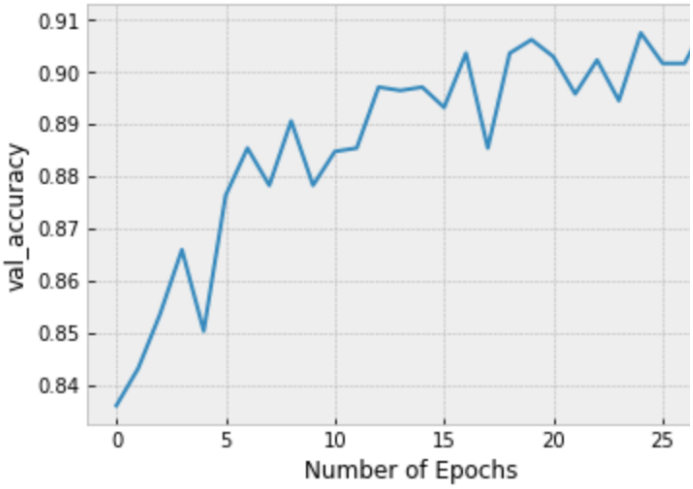

Figure 4 Plot between the number of epochs and val_accuracy.

The graph below shows the results of our experiments in terms of the number of Epochs on the X-axis plotted with a difference of 5 and the accuracy in predictions on Y-axis. At the start, the algorithm shows low prediction accuracy but as the Epochs increase, the accuracy rate improves steadily to make predictions more accurate with an increase in Epochs. However, after about 15 Epochs, the increase slows down as the accuracy converges1.

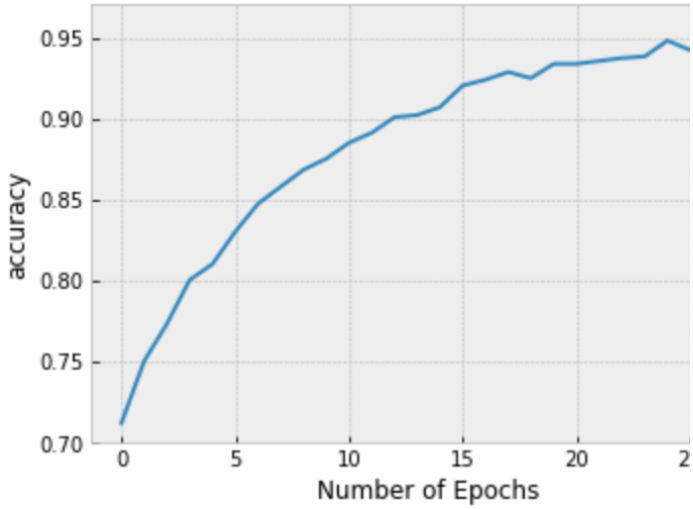

Figure 5 Plot between the number of epochs and accuracy.

III. Conclusions

In this project, we have used Resnet50 to classify different handwritten symbols and achieved a 90.55% over 9032 test samples. The performance of the classification can be shown using the confusion matrix between the predicted and the actual values over the test set shown in the figure below.

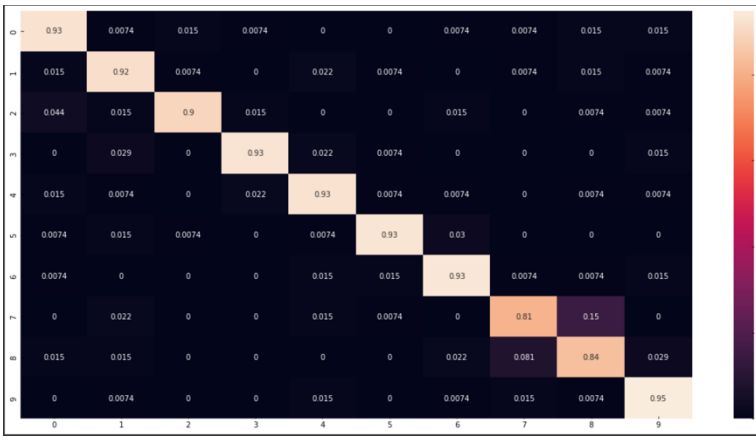

Figure 6 Confusion matrix.

References

[1] R. I. Sultan, M. N. Hasan and M. Kasedullah, "Recognition of Basic Handwritten Math Symbols Using Convolutional Neural Network with Data Augmentation," 2021 5th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), 2021, pp. 1-6, doi: 10.1109/ICEEICT53905.2021.9667794.

[2] Sakshi, Gautam, S., Sharma, C., Kukreja, V. (2021). Handwritten Mathematical Symbols Classification Using WEKA. In: Choudhary, A., Agrawal, A.P (Academic Personnel)., Logeswaran, R., Unhelkar, B. (eds) Applications of Artificial Intelligence and Machine Learning. Lecture Notes in Electrical Engineering, vol 778. Springer, Singapore.

[3] Luo ZX, Shi Y, Soong FK (2008) Symbol graph based discriminative training and rescoring for improved math symbol recognition. In: 2008 IEEE international conference on acoustics, speech and signal processing, ICASSP 2008. IEEE, pp 1953–1956.

[4] Kasthuri M, Shanthi V (2014) Noise reduction and pre-processing techniques in handwritten character recognition using neural networks. Technia 6(2):940.

[5] Hu L, Zanibbi R (2011) HMM-based recognition of online handwritten mathematical symbols using segmental k-means initialization and a modified pen-up/down feature. In: 2011 international conference on document analysis and recognition (ICDAR). IEEE, pp 457–462.

[6] Rahiman MA, Rajasree MS, Masha N, Rema M, Meenakshi R, Kumar GM (2011) Recognition of handwritten Malayalam characters using vertical & horizontal line positional analyzer algorithm. In: 2011 3rd international conference on electronics computer technology (ICECT). IEEE, vol 2, pp 268–274.

[7] Aradhya VM, Kumar GH, Noushath S (2007) Robust unconstrained handwritten digit recognition using radon transform. In: 2007 signal processing, communications and networking, ICSCN 2007. IEEE, pp 626–629.

[8] Ratnamala SP (2016) Shilpa proposed a novel method for handwritten mathematical document based on equation symbols recognition using K-NN and A-NN classifiers. International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE) 23(6). (SPECIAL ISSUE), ISSN 0976-1353.