Handwritten digit recognition using Keras And Tensorflow

Greeshma
Machine Learning Engineer Intern
www.ai-techsystems.com.
Hyderabad,India
greeshma.mareddy@gmail.com

***Abstract*— Handwritten digit recognition is one of the practically important issues in recognition applications. The applications of digit recognition includes in postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize hand written digits and which is submitted by users by the way of a scanner, tablet, and other digital devices. This paper presents an approach to off-line handwritten digit recognition based on different machine learning technique. The main objective of this paper is to ensure effective and reliable approaches for recognition of handwritten digits. Machine learning algorithm using keras and backend tensorflow with different optimizers has been used for the recognition of digits using Python. The result of this paper shows that highest 98.11% accuracy has been obtained for CNN optimizer called RMSprop.**

Keywords— handwritten recognition, digit recognition, machine learning, Python, CNN(Convolutional neural network), machine learning algorithm, Keras , Tensorflow.

I. INTRODUCTION

Image recognition analysis is the most appealing research area in Artificial Intelligence/Deep Learning and also important for a variety of present open research difficulties. Handwritten digits recognition is a well-researched subarea within the field that is concerned with learning models to distinguish pre-segmented handwritten digits. It is the most important issues in data mining, machine learning, Image recognition along with many other disciplines of Artificial Intelligence [1].The main application of CNN machine learning algorithm over the decade has determined effective in conforming determining systems which are not only competing in performance but also accomplish far improved than manually written classical artificial intelligence systems used in the beginnings of character recognition technology [2]. However, not all features of those specific models have been previously inspected.

A great attempt of research work in machine learning and data mining has been contrived to achieve efficient approaches for approximation of digit recognition from data [3]. In 21st Century handwritten digit communication has its own standard and most of the times in daily life are being used as means of the information to be shared. One of the challenges in handwritten characters recognition wholly lies in the variation and distortion of handwritten character set because distinct community may use diverse style of handwriting, and control to draw the similar pattern of the characters of their recognized script. [4].The challenge in handwritten character recognition is mainly caused by the large variation of individual writing styles [5]. Hence, robust feature extraction is very important to improve the performance of a handwritten character recognition system. Nowadays handwritten digit recognition has obtained lot of concentration in the area of pattern recognition system sowing to its application in diverse fields.

In next days, character recognition system might serve as a cornerstone to initiate paperless surroundings by digitizing and processing existing paper documents. [6]. In addition the curves are not necessarily smooth like the printed characters. Furthermore, characters dataset can be drawn in different sizes and the orientation which are always supposed to be written on a guideline in an upright or downright point. Accordingly, an efficient handwritten recognition system can be developed by considering these limitations. It is quiet exhausting that sometimes to identify hand written characters as it can be seen that most of the human beings can't even recognize their own written scripts. Identification of digit from where best discriminating features can be extracted is one of the major tasks in the area of digit recognition system, Hence, there exists constraint for a writer to write apparently for recognition of handwritten documents.

II. DATA PREPARATION

A. Load Data

Handwritten Digit recognition, we have used MNIST ("Modified National Institute of Standards and Technology") dataset. This is a large database of handwritten digits that is commonly used for training various image processing systems. The database is also widely used for training and testing in the field of machine learning. It was created by "re-mixing" the samples from NIST's original datasets.

B. Data Preprocessing

The MNIST dataset contains 60,000 training cases and 10,000 test cases of handwritten digits (0 to 9). Each digit is normalized and centered in a gray-scale (0 - 255) image with size 28 × 28. Each image consists of 784 pixels that represent the features of the digits. Train and test images (28px x 28px) have been stock into pandas. Dataframe as 1D vectors of 784 values. We reshape all data to 28x28x1 3D matrices.

Keras requires an extra dimension in the end which correspond to channels. MNIST images are gray scaled so it use only one channel. For RGB images, there is 3 channels, we would have reshaped 784px vectors to 28x28x3 3D matrices.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

As labels are 10 digits numbers from 0 to 9. We need to encode these lables to one hot vectors (ex : 2 -> [0,0,1,0,0,0,0,0,0,0]), then split the data into training and validation.

III. CNN (Convolutional Neural Network)

When you start learning deep learning with neural network, you realize that one of the most powerful supervised deep learning techniques is the Convolution Neural Networks (abbreviated as "CNN"). The final structure of a CNN is actually very similar to Regular Neural Networks (Regular Nets) where there are neurons with weights and biases. In addition, just like in Regular Nets, we use a loss function (e.g. crossentropy or softmax) and an optimizer (e.g. adam optimizer) in CNNs. Additionally though, in CNNs, there are also Convolutional Layers, Pooling Layers, and Flatten Layers. CNNs are mainly used for image classification although you may find other application areas such as natural language processing etc.

A. Convolutional Layers

Convolutional layer is the very first layer where we extract features from the images in our datasets. Due to the fact that pixels are only related with the adjacent and close pixels, convolution allows us to preserve the relationship between different parts of an image. Convolution is basically filtering the image with a smaller pixel filter to decrease the size of the image without loosing the relationship between pixels. When we apply convolution to 5x5 images by using a 3x3 filter with 1x1 strides (1 pixel shift at each step). We will end up having a 3x3 output (64% decrease in complexity).

B. Build CNN model

- We will build our model by using high level Keras API which uses TensorFlow on the backend. There are several high level TensorFlow APIs such as Layers, Keras, and Estimators which helps us create neural networks with high level knowledge
- However, this may lead to confusion since they all varies in their implementation structure. Therefore, if you see completely different codes for the same neural network although they all use tensorflow, this is why. Use the most straightforward API which is Keras.
- Therefore, import the Sequential Model from Keras and add Conv2D, MaxPooling, Flatten, Dropout, and Dense layers.
- In addition, Dropout layers fight with the overfitting by disregarding some of the neurons while training while Flatten layers flatten 2D arrays to 1D array before building the fully connected layers.
- We may experiment with any number for the first Dense layer; however, the final Dense layer must have 10 neurons since we have 10 number classes (0, 1, 2, …, 9). You may always experiment with kernel size, pool size, activation functions, dropout rate, and number of neurons in the first dense layer to get a better result.

C. Set The Optimizer

An optimizer is one of the two arguments required for compiling a Keras model. We can use either instantiate an optimizer before passing it to model.compile() , or you can call it by its name, in which the default parameters for the optimizer will be used.

These optimizers are : [1]"sgd (Stochastic gradient descent)",This optimiser includes support for momentum, learning rate decay, and Nesterov momentum.[2] RMSProp optimizer, It is recommended to leave the parameters of this optimizer at their default values (except the learning rate, which can be freely tuned).This optimizer is usually a good choice for recurrent neural networks.[3] Adagrad, This optimizer with parameter-specific learning rates, which are adapted relative to how frequently a parameter gets updated during training. The more updates a parameter receives, the smaller the learning rate. It is recommended to leave the parameters of this optimizer at their default values. [4] Adadelta, Adadelta is a more robust extension of Adagrad that adapts learning rates based on a moving window of gradient updates, instead of accumulating all past gradients. This way, Adadelta continues learning even when many updates have been done. Compared to Adagrad, in the original version of Adadelta you don't have to set an initial learning rate. In this version, initial learning rate and decay factor can be set, as in most other Keras optimizers, It is recommended to leave the parameters of this optimizer at their default values.[5]Adam, This is first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters.

D. Data Augumentation

- For image data it is possible to create plausible transformations of existing samples that preserves label information, with the validation of label integrity being performed by a human observer (can a human still recognize the object). One of the significant improvements in performance of classifiers on the MNIST database was through the introduction of elastic deformations , in addition to the existing affine transformations, for data augmentation.
- The elastic deformation was performed by defining a normalized random displacement field u(x, y) that for each pixel location (x, y) in an image specifies a unit displacement vector, such that $Rw = Ro + \alpha u$, where Rw and Ro describe the location of the pixels in the original and warped images respectively. The strength of the displacement in pixels is given by α. The smoothness of the displacement field is controlled by the parameter σ, which is the standard deviation of the Gaussian that is convolved with matrices of uniformly distributed random values that form the x and y dimensions of the displacement field u.
- For the MNIST data set we found that large deformations, with the displacement $\alpha \geq 8$ pixels, could occasionally result in characters that were difficult to recognize by a human observer, that is label information was not preserved.

Identify applicable funding agency here. If none, delete this text box.

- This loss of label integrity was caused by shifting a critical part of the character outside of image boundary, or by introducing a “kink” that rendered the character illegible, as illustrated in Figure 2. We empirically set $\alpha = 1.2$ pixels with $\sigma = 20$ based on performance of the (quick to train) CELM algorithm. The samples needed for elastic warping were generated offline and the same data applied to each of the experiments.

IV. Evaluate Model

After the model is defined, we need to evaluate it.

The training dataset is shuffled prior to being split, and the sample shuffling is performed each time, so that any model we evaluate will have the same train and test datasets in each fold, providing an apples-to-apples comparison between models.

We will train the baseline model for a modest 10 training epochs with a default batch size of 32 examples. The test set for each fold will be used to evaluate the model both during each epoch of the training run, so that we can later create learning curves, and at the end of the run, so that we can estimate the performance of the model. As such, we will keep track of the resulting history from each run, as well as the classification accuracy of the fold.

A. Evaluation Curves or Validation Curves

Model validation curves is used to determine how effective an estimator is on data that it has been trained on as well as how generalizable it is to new input. To measure a model’s performance we first split the dataset into training and test splits, fitting the model on the training data and scoring it on the reserved test data.

In order to maximize the score, the hyperparameters of the model must be selected which best allow the model to operate in the specified feature space. Most models have multiple hyperparameters and the best way to choose a combination of those parameters is with a grid search. However, it is sometimes useful to plot the influence of a single hyperparameter on the training and test data to determine if the estimator is underfitting or overfitting for some hyperparameter values.

B. Curves for Different Optimizers

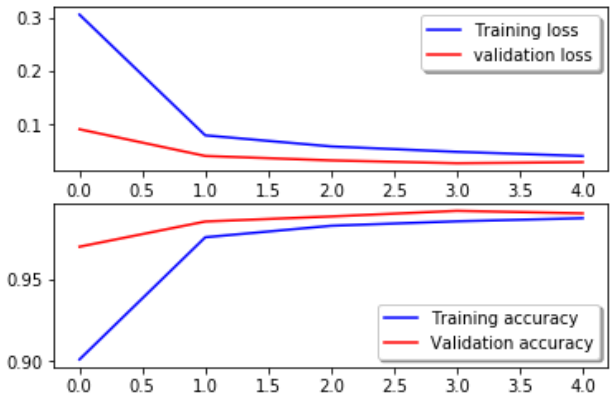

Fig. 1. RMSProp Optimizer

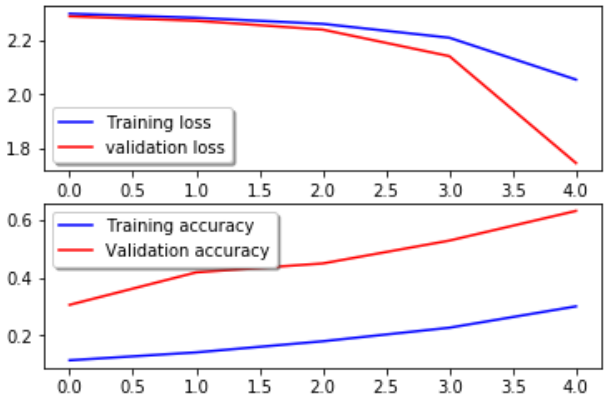

Fig. 2. SGD Optimizer

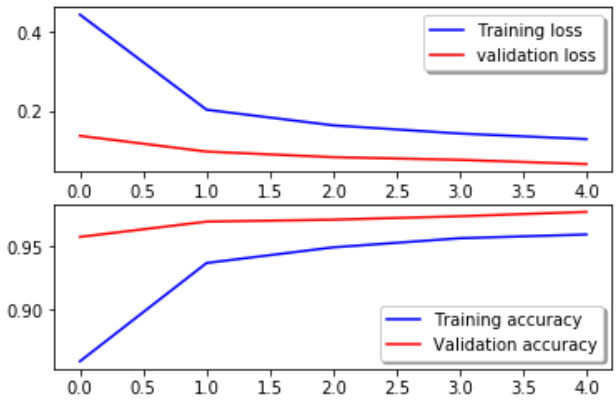

Fig. 3. Adagrad Optimizer

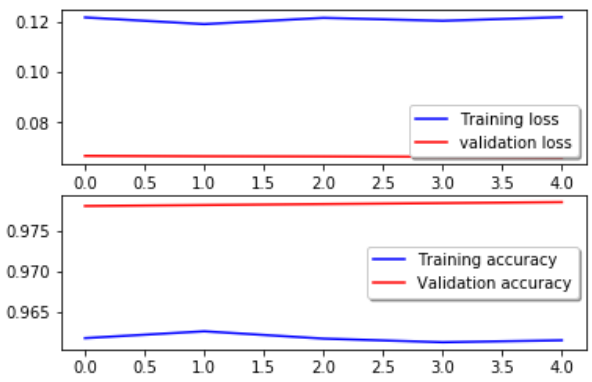

Fig. 4. Adadelta Optimizer

C. *Loss Function & Accuracy scores*

TABLE I. LOSS & ACCURACY SCORES

No. of Optmi-zer	Train		
	Type of Optimizer	***Loss function***	***Accuracy***
1	SGD	0.2676	0.9179
2	RMSProp	0.0421	0.9873
3	Adagrad	0.0462	0.9852
4	Adadelta	0.0338	0.9900

No. of Optmi-zer	Validation		
	Type of Optimizer	***Loss function***	***Accuracy***
1	SGD	0.1325	0.9600
2	RMSProp	0.0261	0.9923
3	Adagrad	0.0300	0.9890
4	Adadelta	0.0301	0.9901

CONCLUSION

The best objective of this experiment is to find a representation of isolated handwritten digits that allow their effective recognition. In this paper used different optimizers for CNN Machine Learning algorithmn for recognition of handwritten numerals. In any recognition process, the important problem is to address the feature extraction and correct classification approaches. The proposed optimizer tries to address both the factors and well in terms of accuracy and Loss function. The overall highest accuracy 99.23% is achieved in the recognition process by Adadelta optimizer. This work is carried out as an initial attempt, and the aim of the paper is to facilitate for recognition of handwritten numeral without using any standard classification techniques.

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an MSW document, this method is somewhat more stable than directly inserting a picture.

To have non-visible rules on your frame, use the MSWord "Format" pull-down menu, select Text Box > Colors and Lines to choose No Fill and No Line.

REFERENCES

[1] ChristianSzegedy Google Inc., WeiLiu University of North Carolina, Chapel Hill, YangqingJia Google Inc., PierreSermanet Google Inc.,ScottReed University of Michigan,DragomirAnguelov Google Inc.,DumitruErhan Google Inc.,VincentVanhoucke Google Inc., AndrewRabinovich Google Inc.,2015.

[2] Kloesgen, W., & Zytkow, J. Handbook of Knowledge Discovery and Data Mining.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105

[4] Handwritten Digit Recognition using Machine Learning Algorithms By S M Shamim, Mohammad Badrul Alam Miah, Angona Sarker, Masud Rana & Abdullah Al Jobair

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[6] . https://www.kaggle.com/yassineghouzam/introduction-to-cnn-keras

[7] Understanding data augmentation for classification: when to warp? By Sebastien C. Wong Defence Science and Technology Edinburgh SA, Australia Email: sebastien.wong@dsto.defence.gov.au,Adam Gatt Australian Defence Force Edinburgh SA, Australia,Victor Stamatescu and Mark D. McDonnell Computational Learning Systems Laboratory Information Technology and Mathematical Sciences University of South Australia Mawson Lakes SA, Australi 2016.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.