International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 09 | Sep 2021 www.irjet.net p-ISSN: 2395-0072

A Deep Learning Application that Implements Handwritten Calculator for Wearable devices

Mohammed Sameer Pasha[1]

[1]B.Tech, Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India

---

Abstract - *Wearable devices are the future of modern technology. Most of these devices allow us to do a lot of operations right at our fingertips. These devices are hands free and portable, eliminating the need to take them out of our pockets. The invention of touchscreen technology has laid ground for a lot of wearable devices that can be interacted through the screen. This simple fact sparks the idea to do a research on handwritten calculator so that people can directly write the formula and get the result. To implement the idea deep learning operations are performed on images that are captured from the user through the screen. Convolution Neural Network(CNN) algorithm is chosen for this application because this is one of the most used algorithms in pattern recognition, such as voice recognition, handwriting recognition, gesture. Every input from handwriting will be processed in several phases, from preprocessing to feature extraction. These features will then be transformed into a form of codeword based on codebook which is built by using training data. These set of codewords are then compared with CNN models previously built with training data. Eventually we will be able to recognize the handwritten mathematical expressions and calculate the outcome of the equations and implement a better way of using a calculator on smaller gadgets.*

Keywords: Deep Learning, Image Processing, Convolutional Neural Networks, Matrix, Correlation.

1.INTRODUCTION

In general, the application must be used to solve complex equations. According to the support package, calculations must be entered in a different format. Our aim is to build an application that intuitively bridges the gap between technology and traditional paper style approach. In the computer-generated font and hand-written text, the user records and measures and shows a solution on the screen. These applications are, however, primarily for predefined equations where users place numbers in static text boxes. Since the restricted interface ensures a consistent recording of the data, the user is slow to enter an equation by clicking multiple times. It would be more intuitive to grab a written picture to the user.

The general approach to solving an image containing an equation is as follows:

1) Screen capture.
2) Picture binarization.
3) Feature extraction.
4) Predicting the numbers.
5) Prompt confirmation to Customer.
6) Equation Solve.

Due to limitations in the accuracy of text detection, a prompt for user approval was inserted between steps 5 and 6 to ensure the right equation is passed to the solver.

1.1 Deep Learning

Deep learning is a class of algorithms used by multi-layered machines to gain more level insights from the information provided. For example, the lower levels of image processing can detect and recognize the edges, while the higher layers can detect human data that includes letters, digits, faces etc. Convolution neural networks (CNNs) are used for most popular deep learning models. Through level extracts and transforms the input into a brighter and more fitting representation in deep learning. The input can be pixel matrix and the representation layer can provide pixel abstraction, identification and encoding of borders for the image recognition application. Edges can be written and coded in a second layer. The strokes can be encrypted in the next third layer. Also, in the fourth layer the images with numbers can be identified. A method of deep learning may learn to work on which features to place themselves optimally at what point.

1.2 Convolution Neural Network

The Convolutional Neural-Network consists of the network which can use a mathematical operation known as Convolution. The Convolution is the special form for linear operation. Convolutional networks can be termed as simply neural-networks that can use convolution in place of generally used matrix-multiplication in minimum of one of the layers. In deep-learning, a Convolutional Neural-Network (CNN) is one among the classes of deep neural-network. They are mostly used for analysis for the visual imagery. They can also be called as space-invariant artificial neural-networks which works with the shared weights architecture. They are also used in language processing, recommender systems, medical image analysis, image classification, image and video recognition and image classification. The Convolutional Neural-Network contains the input layer, output layer, and other multi-hidden layers. The hidden-layers of the CNN usually include a set of convolutional-layers that are developed with the

multiplication/other dot product. It can be represented as either cross-correlation or sliding dot product . This has much importance for the value of indices in matrix, and has influence on how weight can be specified at specific index points.

1.3 Python GUI – Tkinter

For GUI (Graphical User Interface), Python provides several choices. Tkinter is the most used tool of all GUI systems. It is a trendy Python interface to the Python toolkit Tk GUI. The fastest and easiest way to create GUI applications is by using Python tkinter. The development of a GUI is a smooth process to use tkinter.

To create a tkinter app:

1.The module – tkinter is imported.

2.Create the key (container) window.

3. Attach any widget number to the main page.

4. Apply the Trigger event on the controls.

1.4 MNIST Database

MNIST is a de facto "Hello World" computer vision dataset ('Modified National Institute of Standards and Technology'). This classic hand-written image data set has been used as the basis for benchmarking algorithms since its release in 1999. When new technologies appear, MNIST remains a valuable tool for researchers. As methods are known. This method helps us to use Keras as the key package deal for building an simple neural network to predict digits out of hand-written snapshots in a way that we can (with TensorFlow as our backend). We will specifically call the Keras Functional Model API and build a neural network with 4 layers and 5 layers. It can be used as the basis to learn and to practice how neural neural networks for photographic categories can be created, evaluated and used deeply convolving. It is a dataset containing 60,000 tiny 28 by 28-pixel unmarried digits handwritten images between zero and 9. It is a dataset. The project aims to identify one of 10 workouts with integer values from zero to 9 as one of a particular pictures of a manual digit. It is a broadly used and deeply understood data set and top performing models are deeply researching convolution neural networks, which achieve a type precision of over 99% with an error load between null and zero.2%. Here is the picture of the database. It has all the possible ways of writing each number.

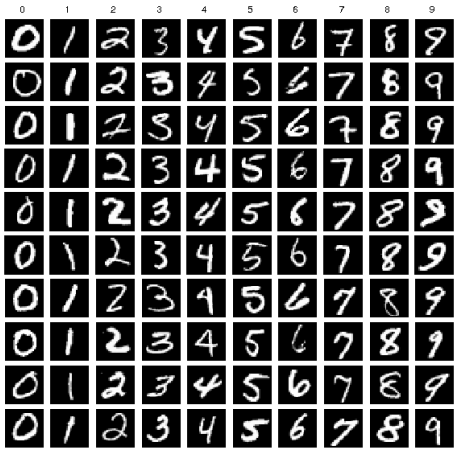

Fig. 1.1. Sample Dataset

2. Building a CNN model on MNIST Dataset

The below python code snippet does not contain the whole code to implement the application. It is just to build a CNN model for the dataset.

```
from keras.datasets import mnist
import keras.backend as K
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Conv2D, MaxPool2D, Flatten, Activation
from keras.utils.np_utils import to_categorical
from keras.optimizers import Adam

import pandas as pd
import numpy as np

from matplotlib import pyplot as plt
%matplotlib inline
%config InlineBackend.figure_format='retina'

(X_train, y_train), (X_test, y_test) = mnist.load_data()
X_train = X_train.reshape(-1, 28, 28, 1)
X_test = X_test.reshape(-1, 28, 28, 1)

y_train_cat = to_categorical(y_train)
y_test_cat = to_categorical(y_test)
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= 255.0
X_test /= 255.0

model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(28, 28, 1)))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Activation('relu'))

model.add(Flatten())
```

```
model.add(Dense(128, activation='relu'))

model.add(Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy',
optimizer=Adam(), metrics=['accuracy'])


model.fit(X_train, y_train_cat, batch_size=32, epochs=2, verbose=1, validation_split=0.3)

model.fit(X_train, y_train_cat, batch_size=32, epochs=2, verbose=1, validation_split=0.3)
```

3. Experiment Analysis

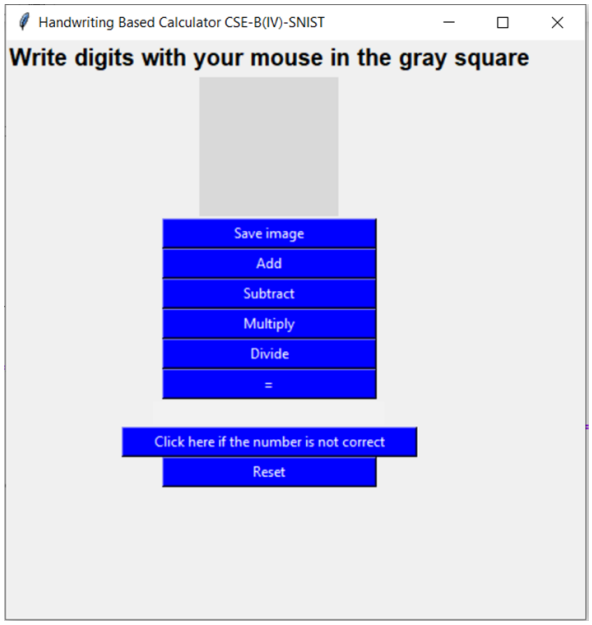

Fig. 3.1. User interface

The GUI for the application is very simple to understand.

Steps to use the GUI:

1. Draw the operand in the empty canvas field on the top.
2.On clicking the save image button the number gets recognized, and the number will be displayed in the below empty field.
3. Choose the operator by clicking the required button.
4.Now draw the second operator with hand and click the save image and the number gets recognized and displayed in the below field.
5.Enter more operands and operators and perform operations
6.To get the answer click the equal to button to get the answer of the equation displayed in the empty field.
7.If any mistake is made in entering the equation then restart the process again by clicking on the "Reset" button.
8.Whenever the operand that is drawn is not recognized correctly then draw it again by clicking on the "Click here if the number is not correct button".

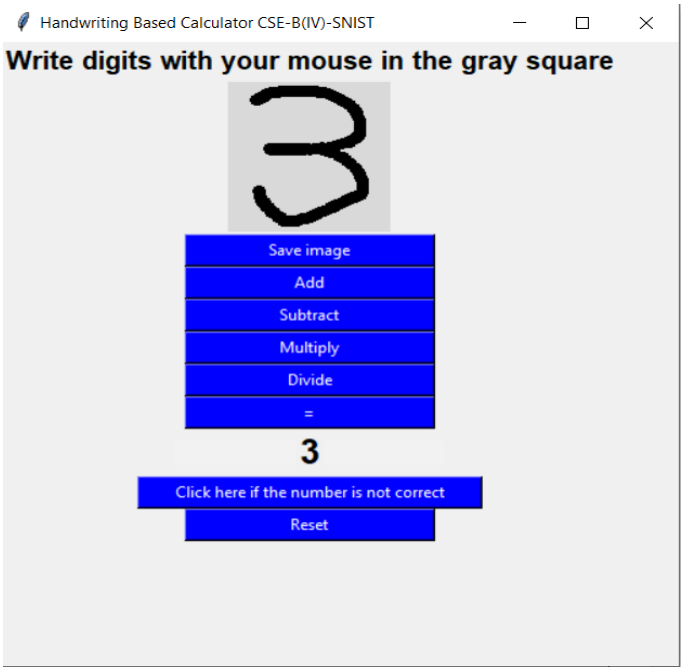

Fig. 3.2. Valid data

On drawing the number "3" in the given drawing pad the number gets recognized as 3 and and displayed below in the empty canvas field.
If the number which is recognized is wrong due to a drawing mistake then we can click the "Click here if the number is not correct" button and remove the number and start writing the operand again.If the equation which you entered is wrong then we can restart the process again by clicking the"Reset"button.

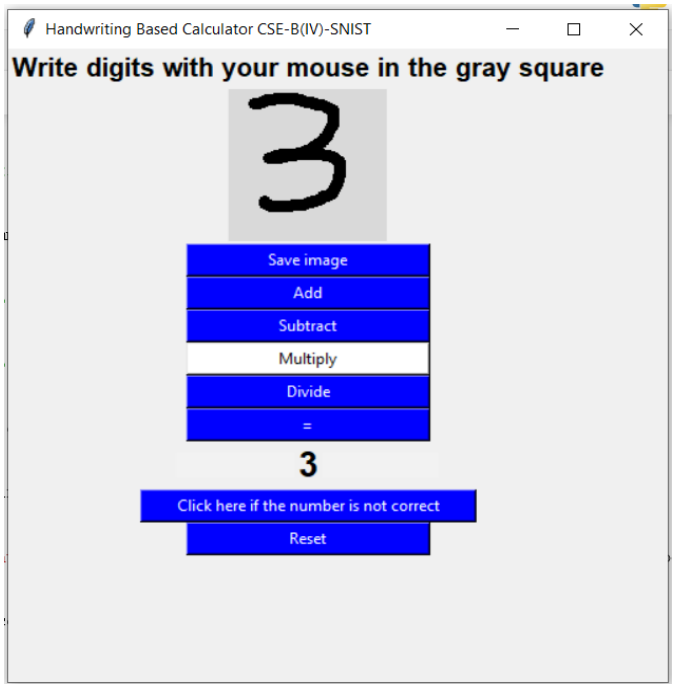

Fig. 3.3. Selecting operators

After the number is drawn and recognized correctly then choose the operator to perform the operation. Above we have clicked the multiply button to multiply number 3. If we make a mistake in our equation by pressing the wrong button we can restart the process again by clicking the "Reset" button.

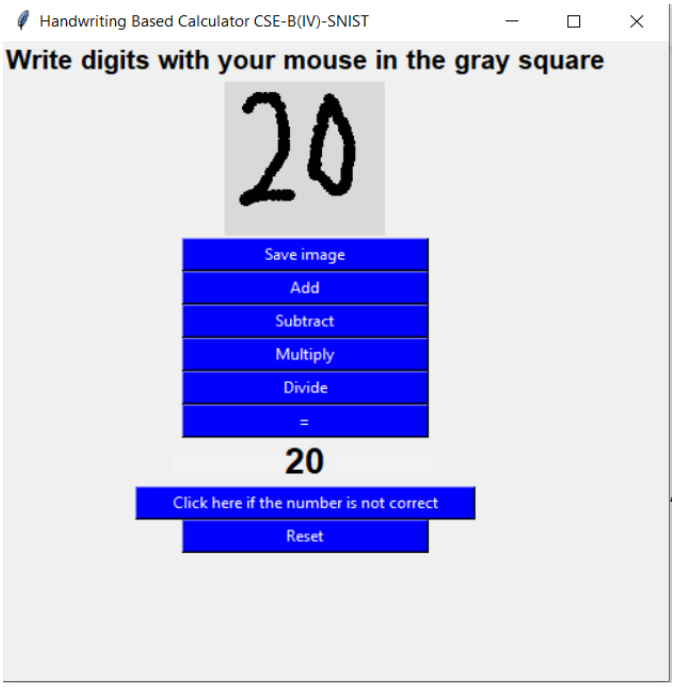

Fig. 3.4. Multiple digit number detection.

Now draw the second operand to perform the operation on the first number after drawing second number, If the number recognized is wrong then we can press the "Click here if the number is not correct" button to clear the operand and reenter the number again to perform the operation. If the equation which is pressed is wrong, then restart the process again by clicking the "Reset" button. In the above output we drew the number "20" and the number got recognized as 20 and got printed in the canvas field. If we need to perform calculations for large numbers such as 234 + 456, draw 2 and click save image and then 3 and save image and then 4 and save image. The numbers 2,3,4 will be concatenated, and the number 234 is formed. Select operator and do the same for 456 and click "=" to get the result.

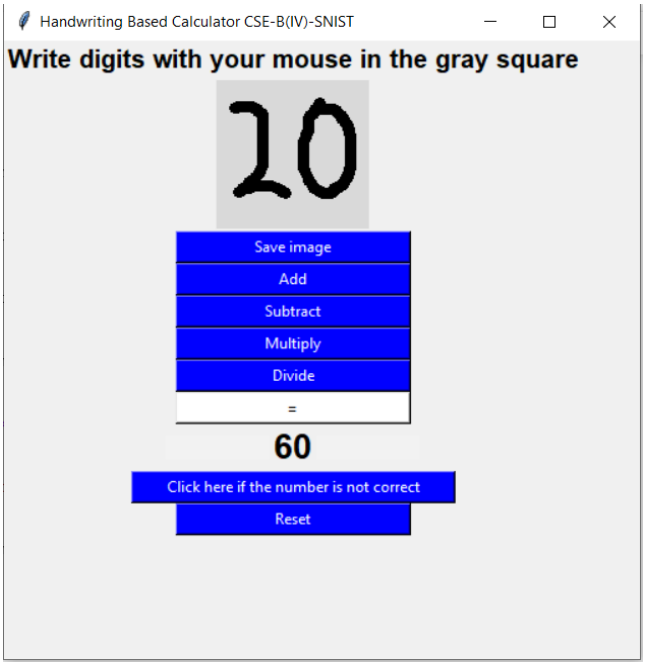

Fig. 3.5. Calculated Result.

After the second number is correctly recognized then we can add extra operators and operands if we need, or we can click the "=" button to get the answer for the equation we needed. In the above output after clicking the "=" button we got the answer as "60" for the equation "3X20=60". If the equation we pressed if wrong, then we can press the "Reset" button to start the process again. To perform furthermore operations, press the "Reset" button again and start other calculations.

3.1 Acceptance Testing

Sno	Test case ID	Input	Expected output	Actual Output	Test Case Pass /fail
1	T1	Draw a Valid num	Print Recognized number from the image as 3	Image recognized as 3	Pass
2	T2	Click the save image	Image is saved	Image is saved	Pass
3	T3	Click on multi-ply	The multiplication operator should be stored	Multiplication operator stored	Pass

4	T4	Draw a valid number in the given empty canvas field	Print recognized number from the image as 20	Displays the number as 20 recognized from draw pad	Pass
5	T5	Click on the equal to button.	The answer is displayed as 60.	Displays 60 as result.	Pass
6	T6	Draw an invalid number in the field	The number is not detected	Invalid number.	Pass
7	T7	Draw an invalid number in the field and change it	The draw pad field must be changed and made empty.	The draw pad is made empty	Pass
8	T8	Click on the Reset	The draw pad must be empty	The draw pad is empty.	Pass

4. CONCLUSION

We can conclude that this method is easy to operate than the traditional way of calculating and it also uses a Neural network with best tried layers giving us an added feature of having a higher tolerance to noise, thus giving accurate results. In neural network the model called as feed forward is mainly trained using the back-propagation algorithm as to classify and recognize the characters and get trained more and more. Apart from these, use of normalization along with feature extraction yielded the better and higher accuracy results in character recognition. It is also observed that bigger our training data set and better neural network design, the better accurate is the result because it can learn a lot of patterns and detect the numbers.

5. FUTURE ENHANCEMENTS

The system can perform various basic mathematical equations for a given wearable device, it is the first step for future calculator. The system can also be implemented in handwriting recognition for simplifying the conversion of writings into documents. The convolutional neural networks can be implemented in an effective way in the future to improve the accuracy and time for the output. This application can be embedded into devices such as watches, touchscreen calculators and other wearable devices. Overall, it can be the first step in futurization of mathematical calculations.

6. REFERENCES

[1] https://www.tensorflow.org/datasets/catalog/mnist

[2] https://www.kaggle.com/kanncaa1/convolutional-neural-network-cnn-tutorial

[3] https://docs.python.org/3/library/tkinter.html

[4] https://colab.research.google.com

[5] https://www.anaconda.com/products/individual-d

[6] https://www.tutorialspoint.com