Abstract - Using dedicated hardware to do machine learning typically ends up in disaster because of cost, obsolescence, and poor software. The popularization of Graphic Processing Units (GPUs), which are now available on every PC, provides an attractive alternative. We propose a fully connected neural network GPU implementation with implementation of CNN algorithm to recognize text in an image. There are many scripts in the world, several of which are used by hundreds of millions of people. Handwritten character recognition studies of several of these scripts are found in the literature. However, convolutional neural network (CNN) has recently been used as an efficient unsupervised feature vector extractor. It is more efficient as a feature extractor than as a classifier. We performed certain amount of training of as-layer CNN for a moderately large class character recognition problem. We used this CNN trained model for a recognizing text of handwritten form in English and output it in a document.

Key Words: CNN, Handwritten, Text, GPU, OCR

1. INTRODUCTION

OCR and on-line handwritten recognition are computationally expensive. Training time is a major bottleneck for improving handwriting recognition. Using dedicated hardware to do machine learning most often ends up in disaster. The hardware is typically expensive, unreliable, without libraries, poorly documented.

The situation has changed recently with the popularization of Graphic Processing Units (GPUs). The GPU is a single-chip processor that is designed to accelerate the real-time three-dimensional (3D) graphics that are displayed to a user. Initially a feature of high-end graphics workstations, the GPU has found its way onto the personal computer as an enhancer of graphics functions for which a conventional central processing unit (CPU) was too slow.

Since recent advancement in development of GPUs and how powerful high-end GPUs tend to be, we used our GPUs to train our model. After researching on several machine learning algorithms such as SVM, decision trees, HMM. We have found that CNN works the best when combined with GPU to output most optimum result.

1.1 FUNDAMENTALS

The main purpose of the system is to scan the images of handwritten text and return an output with correct prediction of correct text after recognizing it. PROBLEM STATEMENT

Designing a system for recognizing and correctly predicting the given handwritten text in English. While predicting the output there are various parameters to be considered such as inaccuracy of handwritten text, curves and fonts of written text, the system will predict the outcome based on the percentage of how close the letter resembles the official letters in English Script.

2. ARCHITECTURE

In deep learning, there is a class called convolutional neural network, most commonly applied to analyzing visual imagery. We will be using 7-layer CNN algorithm's architecture as our base reference system for recognition of text.
**Fig(1): Proposed System Architecture**

**The stages of the CNN method for image recognition in writing as below:**

2.1 **Pre-Processing:** The image is resized, if too large then the calculation will be high or too small will be difficult to adjust to large networks. Larger images are cut and padding will be applied to smaller images to get the standard size. Noise filtering, smoothing and standardization are to be done in this stage.

2.2 **Creation of datasets:** an open source dataset is used from the following website: https://www.kaggle.com/crawford/emnist

2.3 **Final Data Determination:** A large dataset is required to train CNN. To achieve this, the images that have been obtained are modified and changed to get a large number of variations

2.4 **Classification:** The CNN end layer is the SoftMax layer and the SoftMax layer is used to classify the given input image

2.5 **Testing:** The test module is related to the test image. The test images were obtained by splitting the randomly enlarged dataset.

3. **IMPLEMENTATION**

**SOFTWARE PLATFORM**

3.1 **TensorFlow**

TensorFlow is an amazing information stream in machine learning library made by the Brain Team of Google and made open source in 2015. It is intended to ease the use and broadly relevant to both numeric and neural system issues just as different spaces. Fundamentally, TensorFlow is a low-level tool for doing entangled math and it targets specialists who recognize what they're doing to construct exploratory learning structures, to play around with them and to transform them into running programs. For the most, it can be considered as a programming framework in which one can entitle to calculations as graphs. Nodes in the graph speak the math activities, and the edges contain the multi-dimensional information clusters (tensors) related between them.

3.2 **Python 3.7**

Python is broadly utilized universally and is a high-level programming language. It was primarily introduced for prominence on code, and its language structure enables software engineers to express ideas in fewer lines of code. Python is a programming language that gives you a chance to work rapidly and coordinate frameworks more effectively.

3.3 **Anaconda3**

Anaconda is a free and open-source appropriation of the Python and R programming for logical figuring like information science, AI applications, large-scale information preparing, prescient investigation, and so forth. Anaconda accompanies in excess of 1000s of packages just as the Conda package and virtual environment director, called Anaconda Navigator, so it takes out the need to figure out how to introduce every library freely. Anaconda Navigator is a graphical UI (GUI) incorporated into Anaconda appropriation that enables clients to dispatch applications and oversee conda packages, conditions and channels without utilizing command line directions.

**HARDWARE PLATFORM**

3.4 **NVIDIA GeForce Graphic Card:**

Nvidia Corporation who specializes in manufacturing high quality Graphical Processing Units (GPUs) for the gaming, graphic work and simulation. We have used a NVIDIA RTX 2060 GPU for training our model.

**ALLOCATI0NG GPU MEMORY TO TENSORFLOW**

*Run this piece of code before starting the program*

**PYTHON CODE:**

```python
import tensorflow as tf
gpus = tf.config.experimental.list_physical_devices('GPU')
try:
```

tf.config.experimental.set_memory_growth(gpus[0], True)
except RuntimeError as e:
    # Memory growth must be set before GPUs have been initialized
    print(e)

4. RESULT

We have implemented a generic 7-layer fully connected neural network using the Pixel Shader and languages for the GPU. These typically represent a very small fraction of the computation for the end-to-end system. These results must be interpreted with caution. Memory speeds play a significant role on the performance. Handwritten Character data are very much essential for this work for training and testing the proposed Handwritten Character Recognition System.

In English alphabets 26 characters and as well as in numerical 10 characters are to be developed in different forms. For example take the character "U", this character can be written in different various forms. For Training the character data we used Convolutional Neural Network algorithm. For this reason, the character data is to be taken and recognize what character it is.

The following are the results achieved after implementation:

4.1

<table>
<thead>
<tr>
<th>Letter prediction : U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage : 94.12%</td>
</tr>
<tr>
<td>Second best</td>
</tr>
<tr>
<td>Prediction : u</td>
</tr>
<tr>
<td>Percentage : 3.57%</td>
</tr>
</tbody>
</table>

4.2

<table>
<thead>
<tr>
<th>Letter prediction : K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage : 61.16%</td>
</tr>
<tr>
<td>Second best</td>
</tr>
<tr>
<td>Prediction : k</td>
</tr>
<tr>
<td>Percentage : 37.84%</td>
</tr>
</tbody>
</table>

Above output explains how the character is to be drawn and learns the character database and calculate percentage. Based on percentage we can recognize the character. The above screenshots explains how the character set is to be learned based on that we can display the learning graph.

MODEL EVALUATION RESULTS:

MODEL LOSS = 0.3740
ACCURACY = 0.8598
ACCURACY PERCENTAGE = 85.98%
training. However, in the present approach we did not allow the pain to train the CNN architecture through fine-tuning of its parameter. On the other hand, we trained this architecture until we achieved only moderate recognition performance.

ACKNOWLEDGEMENT

We would like to express our sincere thanks to Prof. Teena Varma, our project in charge, for her guidance and supervision regarding project outlook and mindset and information which helped us to complete this project, her support is greatly appreciated.

REFERENCES

1. DATABASE USED

2. K. Radha Revathi, A.N.L. Kumar, Andey Krishnaji Neural Network Based Hand-Written Character Recognition IJCSE E-ISSN: 2347-2693.


