OCR By CS-92, CS-94, CS-100 and CS-79

PROJECT REPORT

Tasks given:

1. Without using pytesseract library
2. Extract texts from images
3. GpGPU implementation
4. One of reconstruction and translation work

OCR:

Optical Character Recognition (OCR) technology is used for distinguishing the printed/handwritten text inside the digital images of a physical document. For example, a scanned paper document. The process of OCR involves an analysis of the document text and so translating the text into code that can further be used for data processing. The technology, OCR is also referred to as text recognition as it recognizes the text from an image file or document.

Deep Learning and OCR:

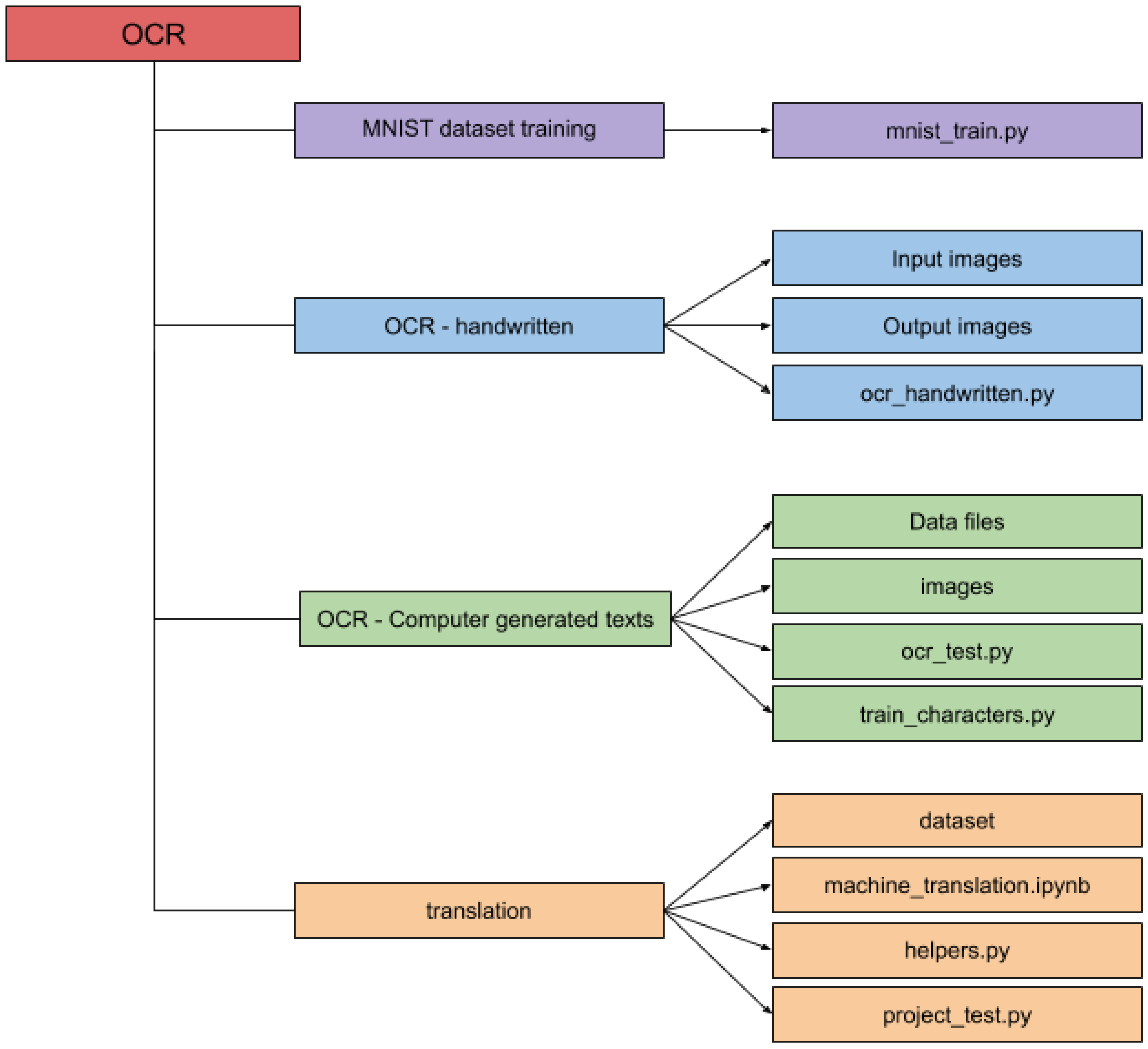
To function the OCR technology without errors, deep learning models are trained with an excess of data. The more the data, the better will be results of the OCR deep learning model. The testing data is trained in the model again to make it work with a variety of different inputs.

KNN Algorithm:

KNN simply K nearest neighbourhood In this k means it is a hyperparameter(parameter whose value is used to control the learning process) it works based on the nearest neighbourhood. KNN is one of the simplest machine learning models. It can be used in classification as well as regression problems.

OCR By CS-92, CS-94, CS-100 and CS-79

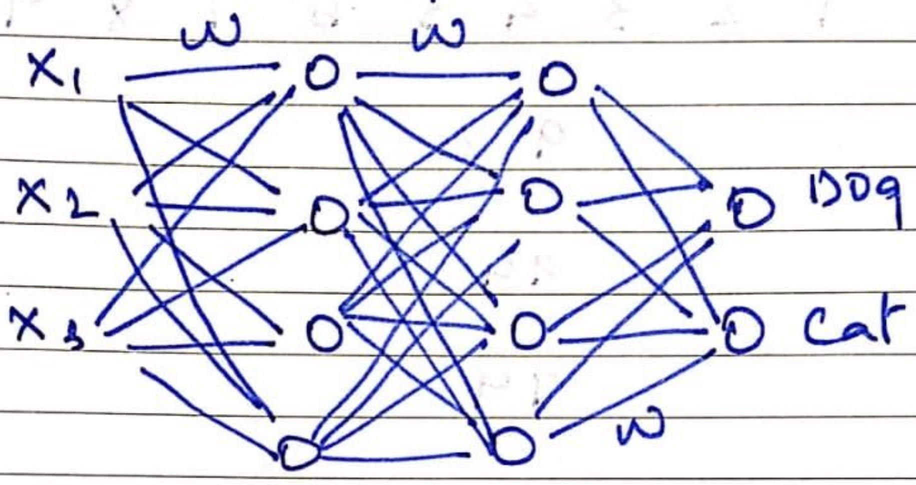
PROJECT TREE:



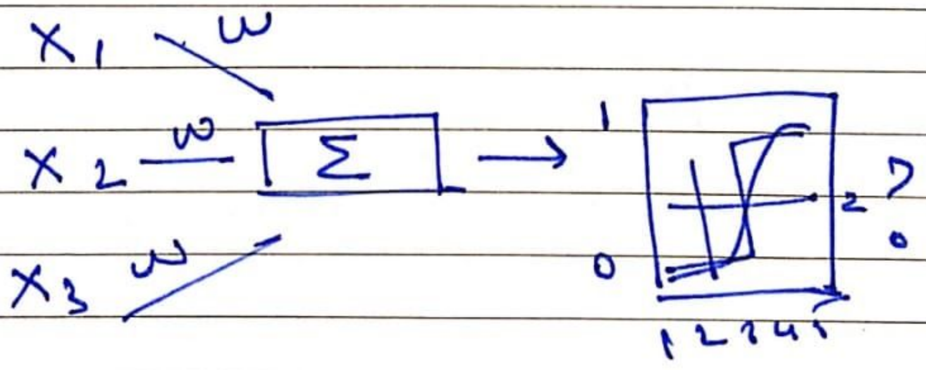
Neural Networks:

Date:

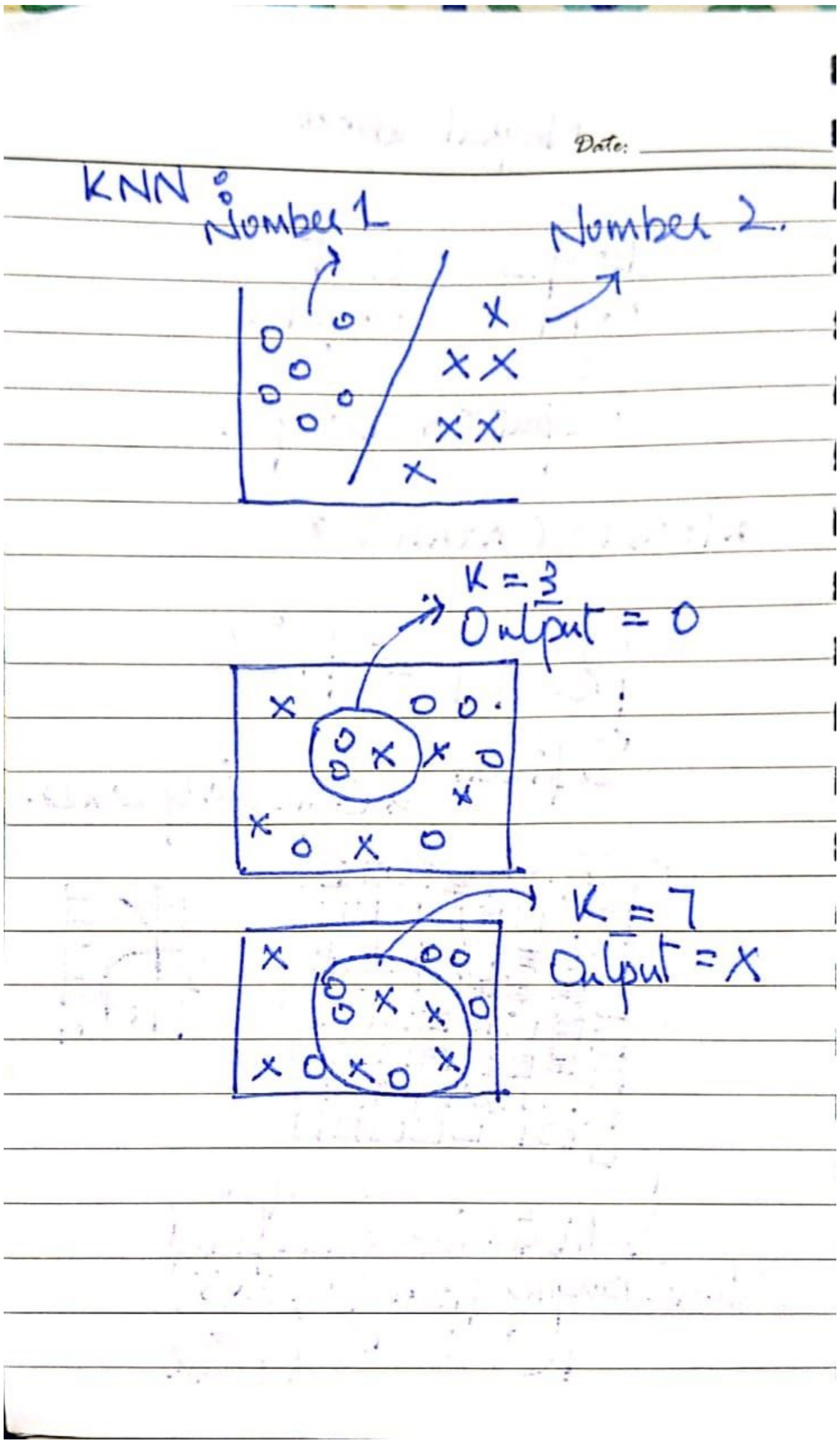
Layering



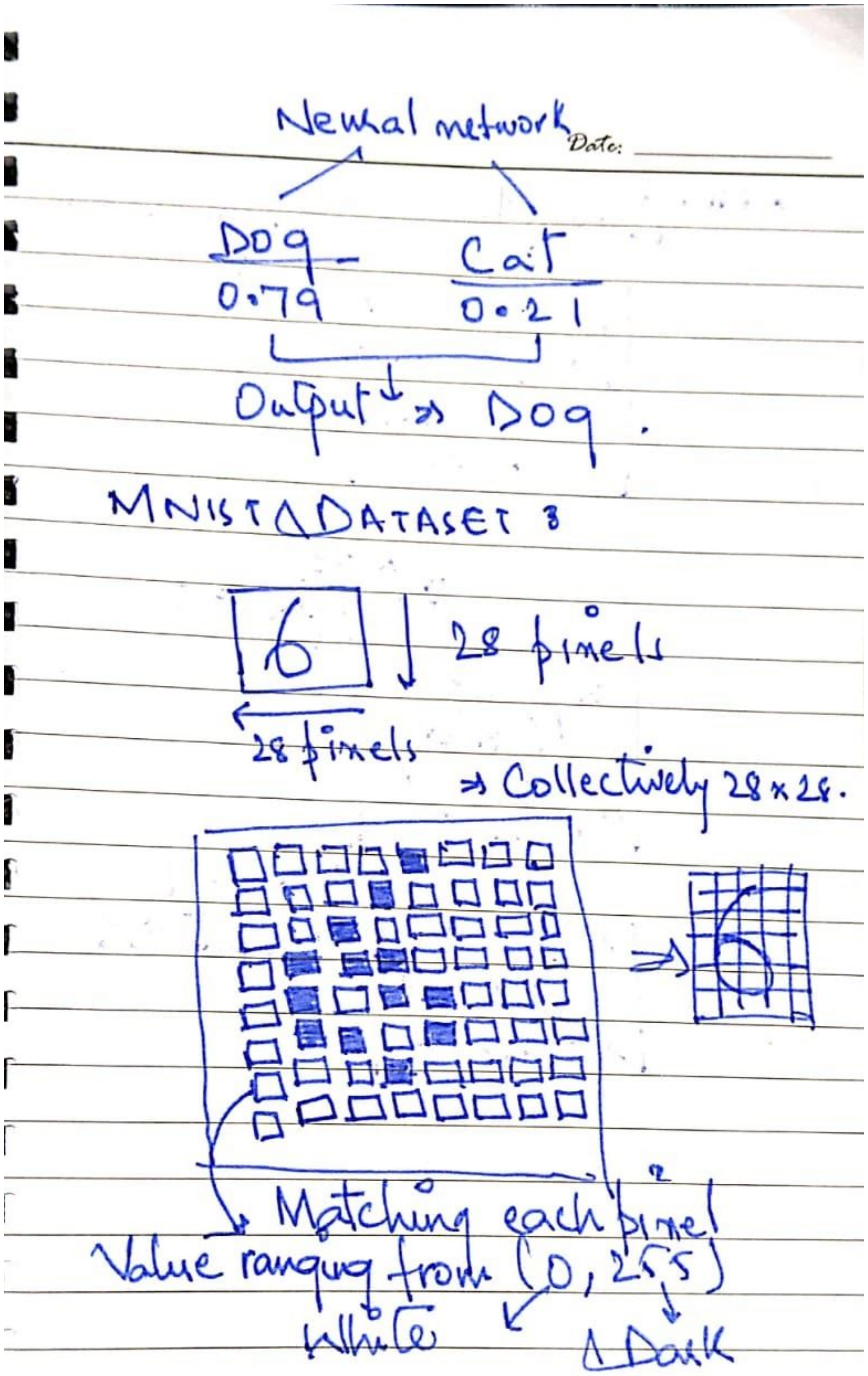
Similar case with numbers



KNN ALGORITHM:



MNIST DATASET AND WORKING:



Neural network

Date:

Dog	Cat
0.79	0.21

Output ⇒ Dog.

MNIST DATASET :

6

28 pixels

28 pixels

⇒ Collectively 28 x 28.

Matching each pixel

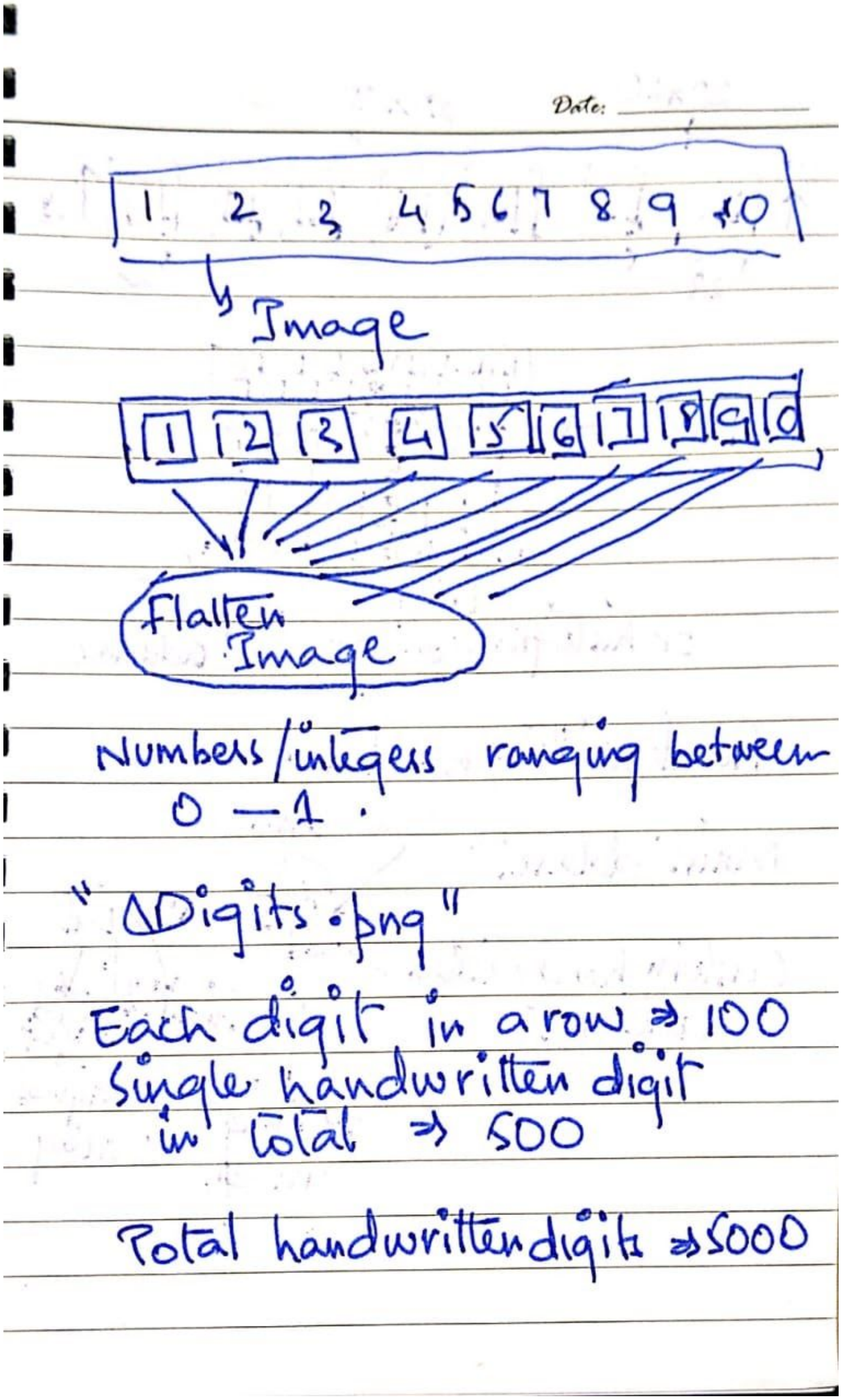
Value ranging from (0, 255)

White

Dark

OCR By CS-92, CS-94, CS-100 and CS-79

IMAGE DETECTION AND PROCESSING USING OPENCV:



Date:

1 2 3 4 5 6 7 8 9 10

Image

1 2 3 4 5 6 7 8 9 0

Flatten Image

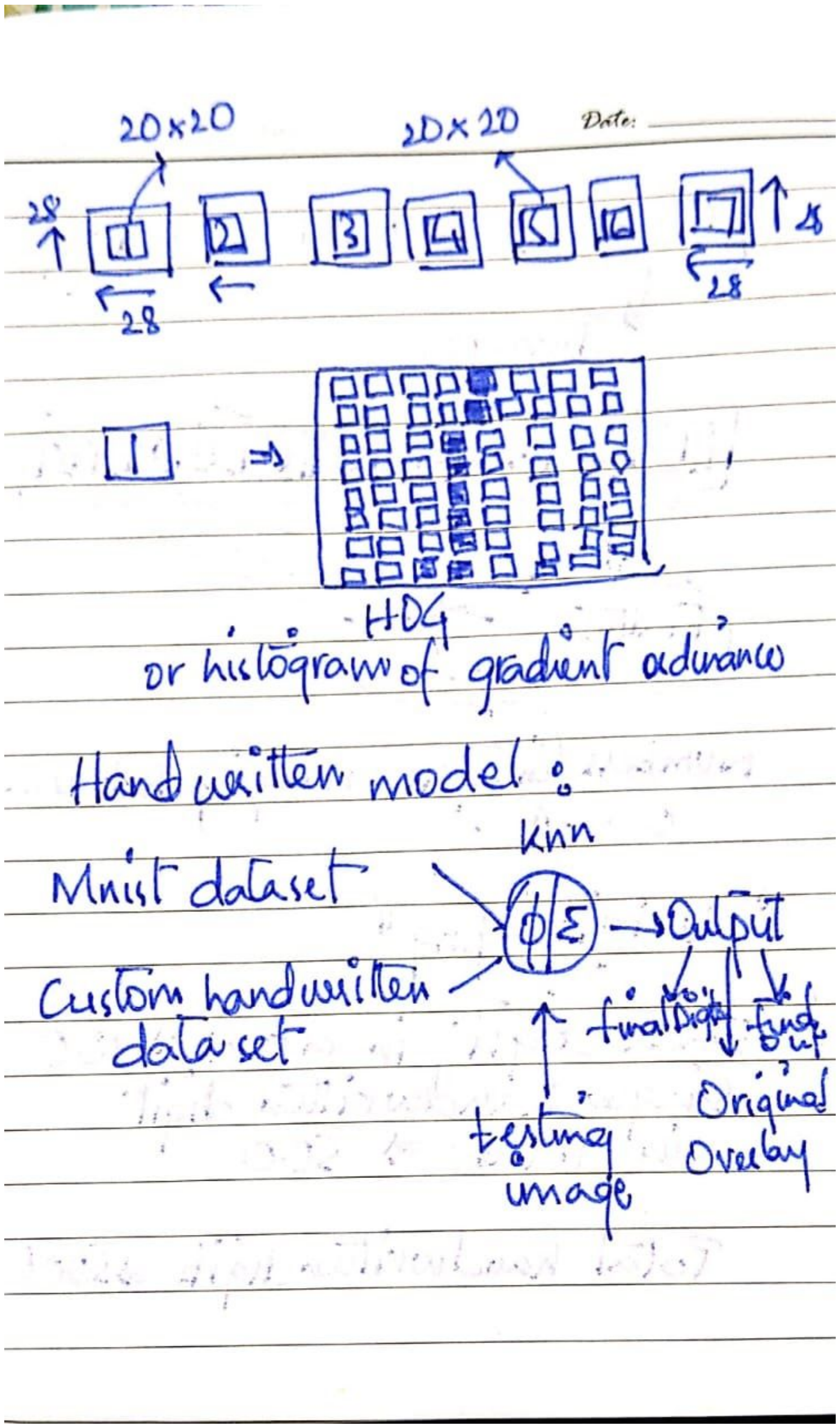
Numbers/integers ranging between 0 – 1.

"Digits.png"

Each digit in a row ⇒ 100 single handwritten digit in total ⇒ 500

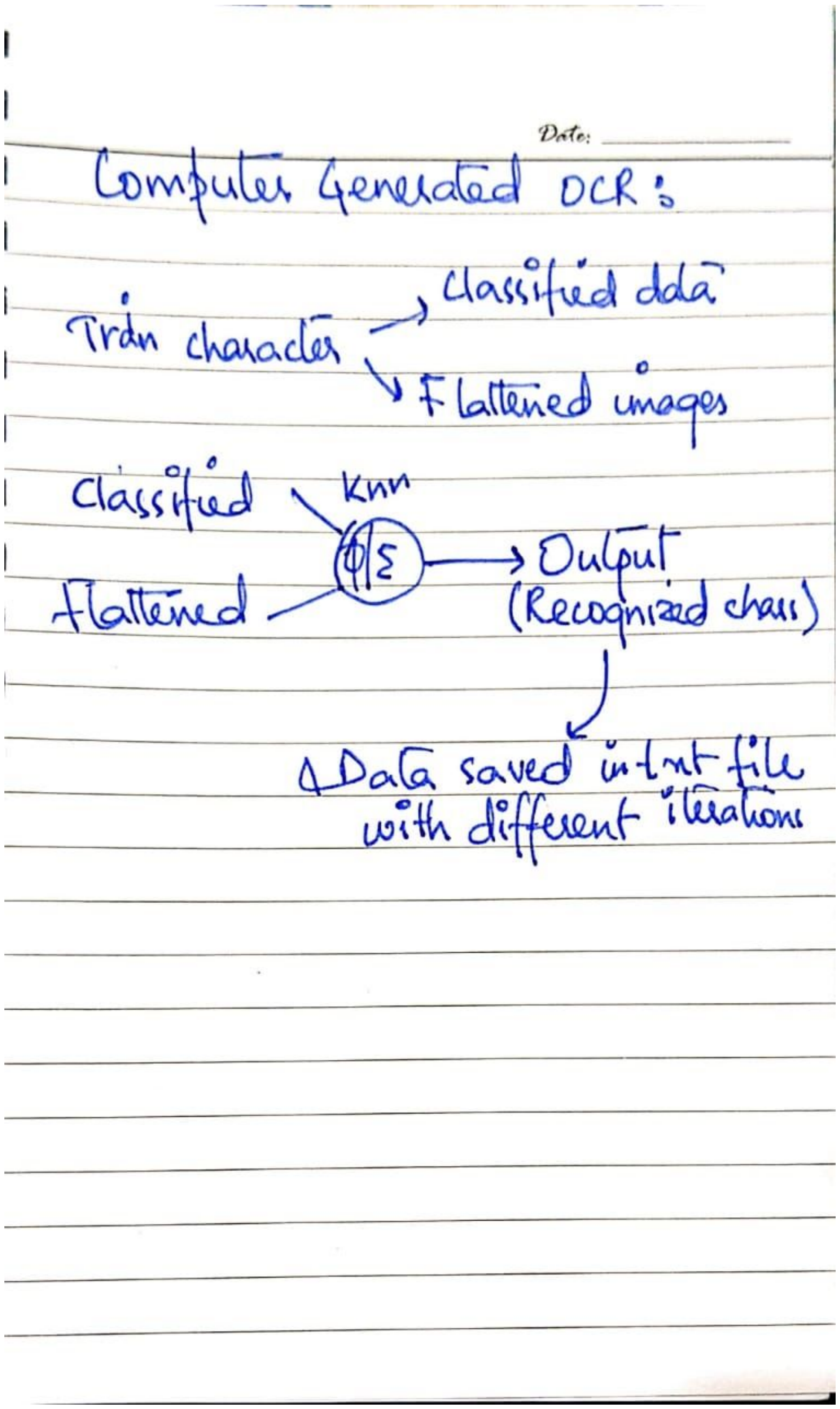
Total handwritten digits ⇒ 5000

HANDWRITTEN MODEL:



OCR By CS-92, CS-94, CS-100 and CS-79

COMPUTER GENERATED TEXTS - OCR:


Date:

Computer Generated OCR:

Train character
Classified data
Flattened images
Classified
Flattened
knn
Φ|Σ
Output (Recognized chars)
Data saved in txt file with different iterations

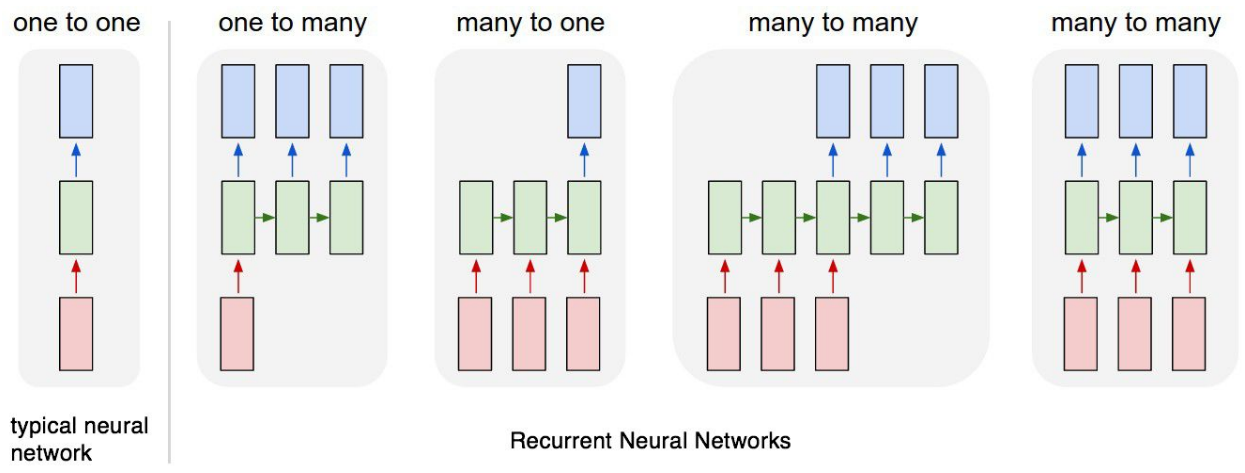
Machine Translation

Neural Machine Translation

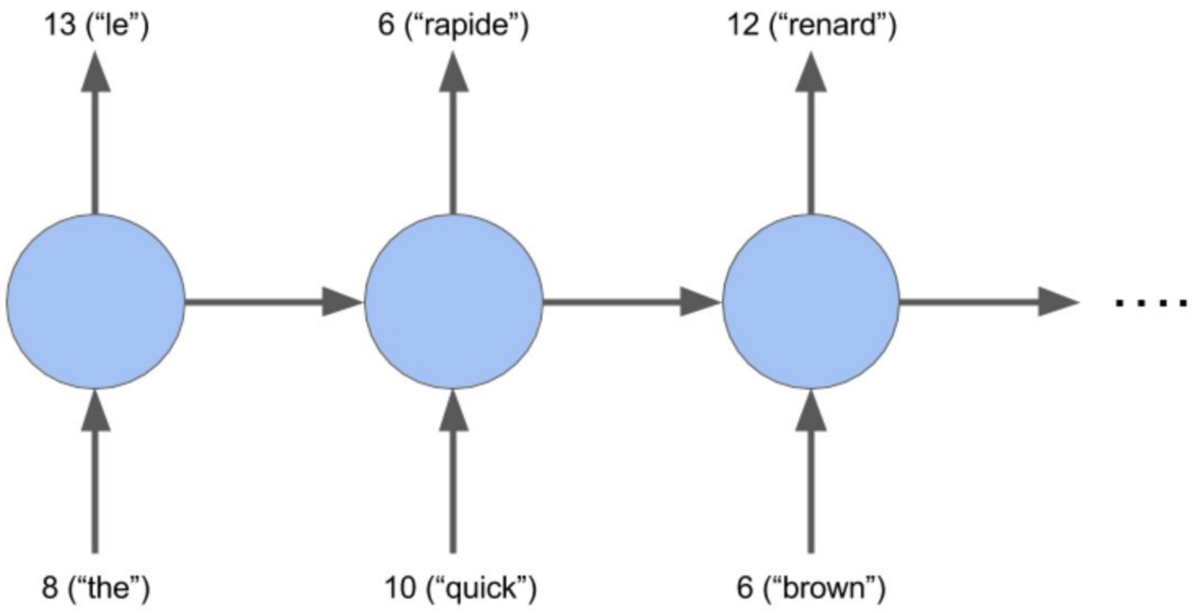
Neural machine translation (NMT) is typically software used to translate words from one language to another.NMTs encompass all types of machine translation where an artificial neural network is used to predict a sequence of numbers when provided with a sequence of numbers.In the case of translation, each word in the input sentence (e.g English) is encoded as a number to be translated by the neural network into a resulting sequence of numbers representing the translated target sentence (e.g french).

RNN Model

RNNs are designed to take sequences of text as inputs or return sequences of text as outputs, or both. They're called recurrent because the network's hidden layers have a loop in which the output and cell state from each time step become inputs at the next time step. This recurrence serves as a form of memory. It allows contextual information to flow through the network so that relevant outputs from previous time steps can be applied to network operations at the current time step.



In Our case we are using many to many relation



This Image shows the working of recurrent neural networks

Building the Pipeline:

1. **Preprocessing :** load and examine data, cleaning, tokenization, padding
2. **Modeling:** build, train, and test the model
3. **Prediction:** generate specific translations of English to French, and compare the output translations to the ground truth translations

Tokenization

We need to tokenize the data to convert the text to numerical values. This allows the neural network to perform operations on the input data. For this project, each word and punctuation mark will be given a unique ID

Padding

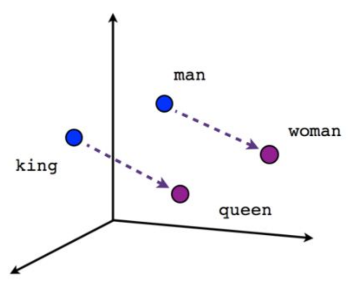
When we feed our sequences of word IDs into the model, each sequence needs to be the same length. To achieve this, padding is added to any sequence that is shorter than the max length

Modeling

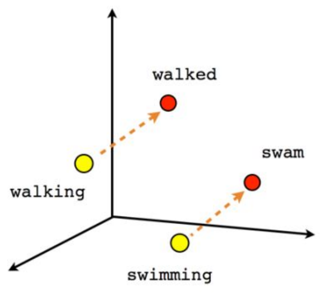
1. **Input :** Input sequences are fed into the model with one word for every time step. Each word is encoded as a unique integer or one-hot encoded vector that maps to the English dataset vocabulary.
2. **Embedding Layer:** Embeddings are used to convert each word to a vector. The size of the vector depends on the complexity of the vocabulary.
3. **Recurrent Layer:** This is where the context from word vectors in previous time steps is applied to the current word vector.
4. **Dense Layer :** These are typical fully connected layers used to decode the encoded input into the correct translation sequence.
5. **Output Layer:** The outputs are returned as a sequence of integers or one-hot encoded vectors which can then be mapped to the French dataset vocabulary.

Embedding

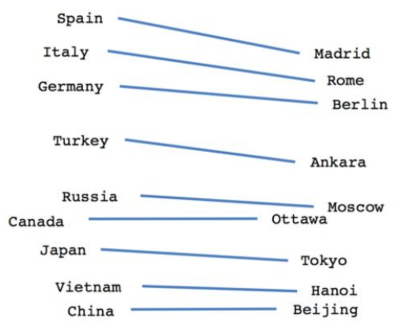
Embeddings allow us to capture more precise syntactic and semantic word relationships. This is achieved by projecting each word into n-dimensional space. Words with similar meanings occupy similar regions of this space; the closer two words are, the more similar they are.



Male-Female



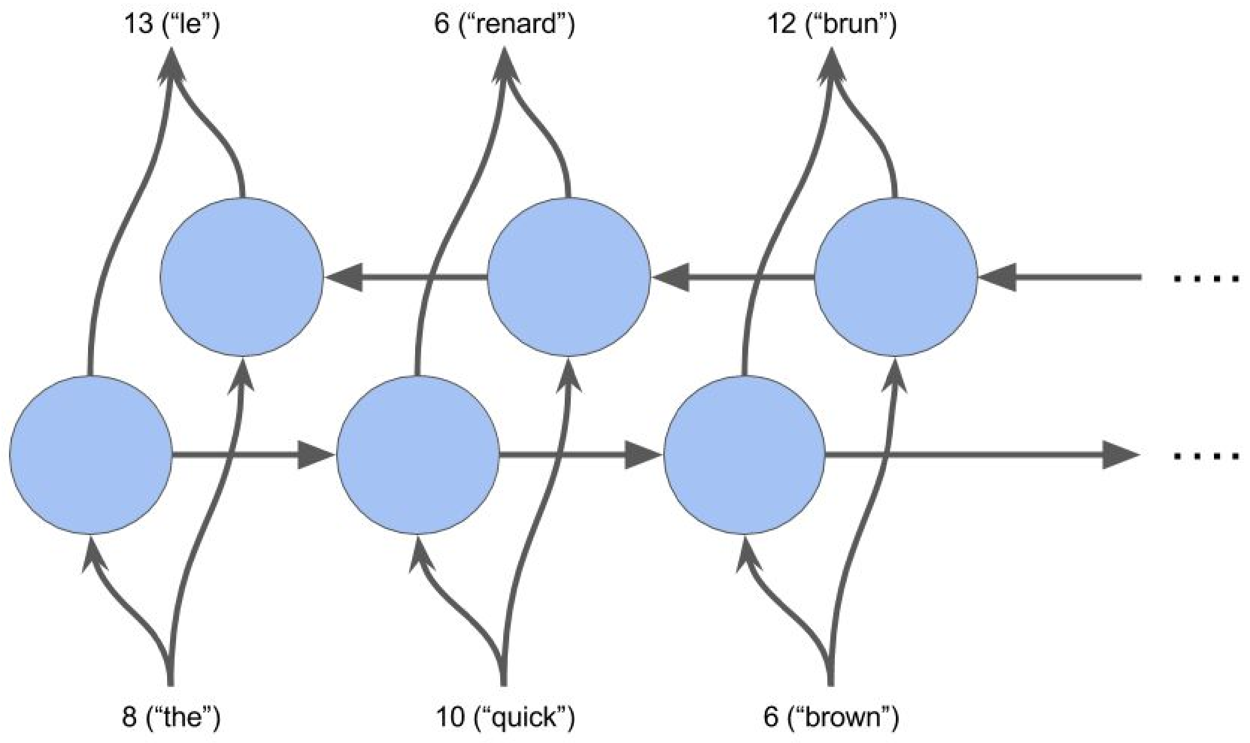
Verb tense



Country-Capital

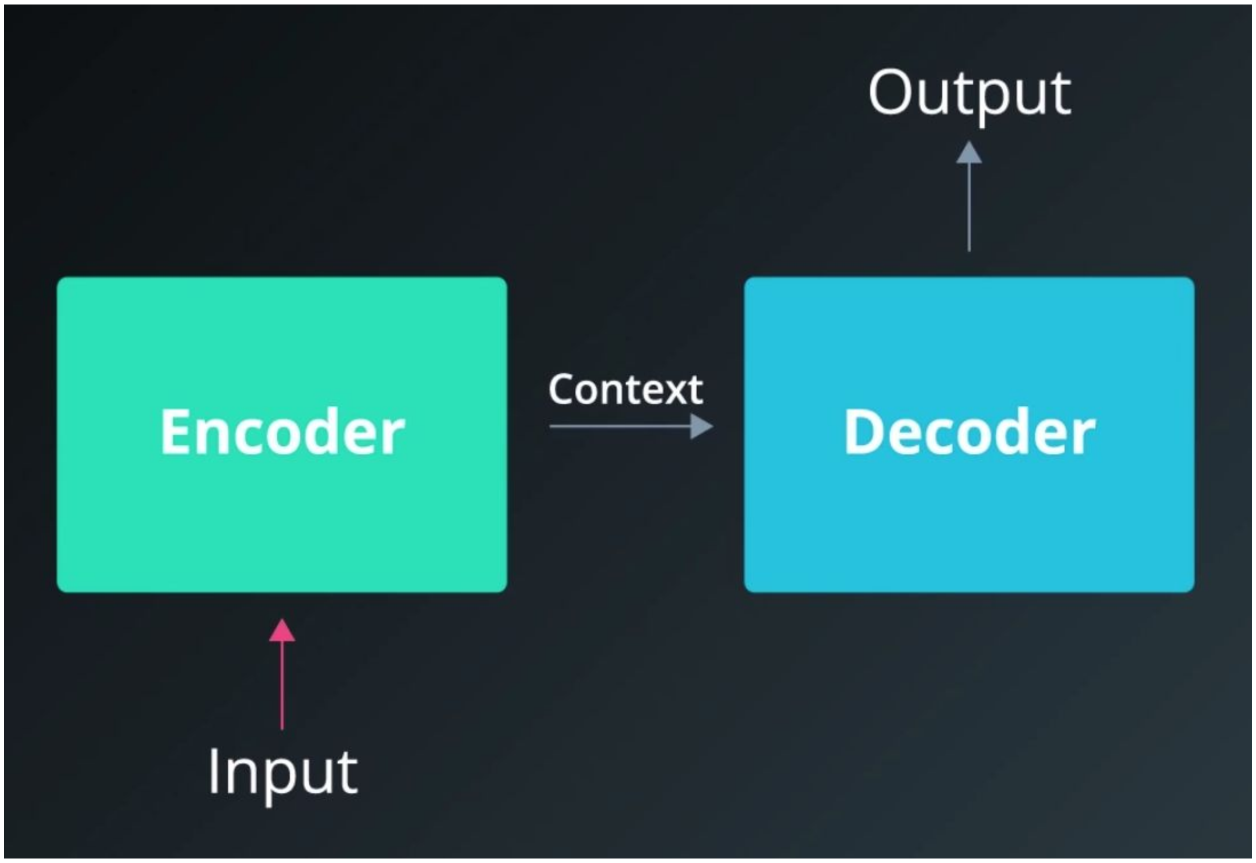
Bidirectional Layer

Bidirectional Recurrent Neural Networks connect two hidden layers of opposite directions to the same output.



Encoder and Decoder

Our sequence-to-sequence model links two recurrent networks: an encoder and decoder. The encoder summarizes the input into a context variable, also called the state. This context is then decoded and the output sequence is generated.



Graphical Processing Unit

WHAT ARE GPUs?

The graphics processing unit, or GPU, has become one of the most important types of computing technology, both for personal and business computing. Designed for parallel processing, the GPU is used in a wide range of applications, including graphics and video rendering. Although they're best known for their capabilities in gaming, GPUs are becoming more popular for use in creative production and artificial intelligence (AI).

USING OUR SYSTEMS LOCAL GPU:

If you have NVIDIA's graphics card installed along with CUDA and cuDNN one can use it to train a tensorflow model and it now supports GPU.
However, AMD GPUs do not support CUDA and require openCL to convert 80% of CUDA.

USING CLOUD BASED GPUs:

OCR By CS-92, CS-94, CS-100 and CS-79

With the emergence in scope of AI, many platform's now provide now provide online GPUs. Google Colab provides a single 12GB **NVIDIA Tesla K80 GPU** that can be used up to 12 hours continuously. To use the google colab in a GPU mode you have to make sure the hardware accelerator is configured to GPU.
Jupyter Notebook also provides a GPU but we need to set up an environment to run that.

Research Paper:

The research paper **"OPTICAL CHARACTER RECOGNITION — A Complete Guide"** is written in latex, following IEEE's format.

Links:

Github repository: https://github.com/MSaaad/ocr-app-using-algorithm

Google Colab for translation:
https://drive.google.com/file/u/0/d/1kC8Hmm-Z5dpiJQlWY4XqnihxFZjOO_YW/edit