Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Handwritten Prescription Recognition

June 2019

Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Handwritten Prescription Recognition

By:

Abdelrahman Ibrahim Abdelrahman [CS]
Aya Abd El-Basset Abd El-Samad [CS]
Donia Adel Mohamed [CS]
Ranya Roshdy Hassan [CS]
Youssef Sherif Eid [CS]

Under Supervision of:

Prof. Mohamed Ismail Roshdy
Professor,
CS Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Dr. Ahmed Salah
Lecturer,
CS Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

AL. Ghada Hammed
Assistant Lecturer,
SC Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Acknowledgement

All praise and thanks to ALLAH, who provided us the ability to complete this work.

We are grateful of our parents and our families who are always providing help and support throughout the whole years of study. We hope we can give that back to them.

We also offer our sincerest gratitude to our supervisors, Dr. Mohamed Ismail Roshdy, Dr. Ahmed Salah and T.A. Ghada Hammed who have supported team throughout our thesis with their patience, knowledge and experience.

Finally, we would thank our friends and all people who gave us support and encouragement.

Abstract

Despite the abundance of technological writing tools, many people still choose to take their notes traditionally with pen and paper. Handwritten Character Recognition (HCR) has been a fascinating and challenging research area in the field of image processing and pattern recognition. Handwriting recognition means to scribe the written words from the paper. HCR is applied in many domains, like reading your own notes, reading old papers and messages, etc. We are trying to apply the handwriting recognition on medical forms of doctors and trying to get some pattern which will be used to provide what the most probable prescribed medication is. During the first phases the prescription will not be exact due to the difficulty of doctors' handwriting for non-medicinal persons, but in the further phases this will be improved.
HCR by same/different person(s) vary in both size and shapes. Numerous variations in writing styles of individual character make the recognition task difficult. In this project we aim to recognize the doctors' handwriting in medical forms to make it easier for pharmacist, nurses and patients to read them.

Table of Contents

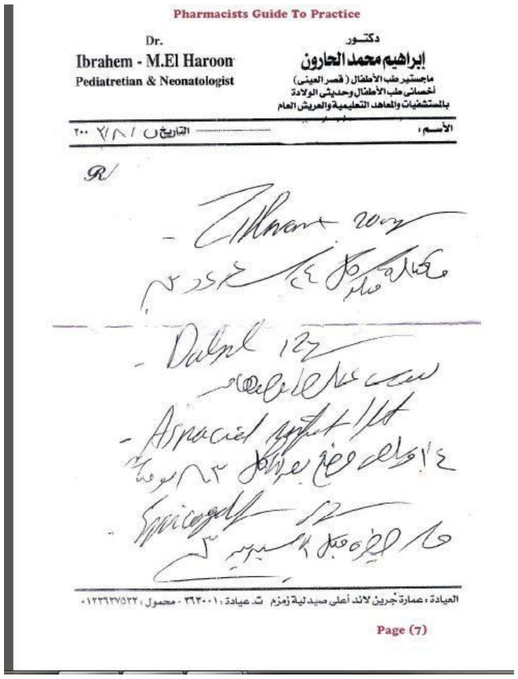
List of Figures

List of Abbreviations

API	Application Program Interface
BLSTM	Bidirectional Long Short-Term Memory
BP	Back Propagation
CNN	Convolution Neural Network
CRNN	Convolutional Recurrent Neural Network
CTC	Connectionist Temporal Classification
HCR	Handwritten Character Recognition
HMM	Hidden Markov Model
LSTM	Long Short-Term Memory
RCA	Radio Corporation of America

1- Introduction

1.1 Motivation


Pharmacists Guide To Practice

Dr. Ibrahem - M.El Haroon
Pediatretian & Neonatologist

دكتور
إبراهيم محمد الحارون
ماجستير طب الأطفال (قصر العينى)
أخصائى طب الأطفال وحديثى الولادة
بالمستشفيات والمعاهد التعليمية والعريش العام

الأسم: التاريخ / ٨ / ٧ ٢٠٠

R/

- Tilhvant 200 [illegible]

- Dalsol 12 [illegible]

- Aspocid [illegible]

- [illegible] 12 [illegible]

العيادة: عمارة جرين لاند أعلى صيدلية زمزم ت. عيادة: ٣٦٣٢٠٠١ - محمول: ٠١٢٣٦٣٧٥٢٢

Page (7)

Figure 1: Prescription

We all know that it is difficult to read the handwriting of doctors for non-medicinal personal doctors' cursive handwriting causes many problems and making patients in risky situations. Lots of pharmacists couldn't read the accurate name of treatment and the dose because of many doctors don't writing the complete name of the treatment and the letters individually extremely hard reading which lead to wrong reading of treatment's name which can make serious disasters for the patient according to his health and often lead to death.

Studies and statistics have proven that In India and all over the world there is a big number of death cases each year because of this issue that motivate us to do our best to find a solution to this problem and limit of number of death cases that's result from this issue specially in Egypt.

1.2 Problem Definition

As patients, we have all been puzzled by what doctors have scribbled on prescriptions. Hardly a word can be recognized, often not even a single letter. We have a doctor prescription which extremely hard reading, uncompleted names of treatments and
non-observed doses doctors' handwriting is crossed and slanted which is difficult to segment which can be read.

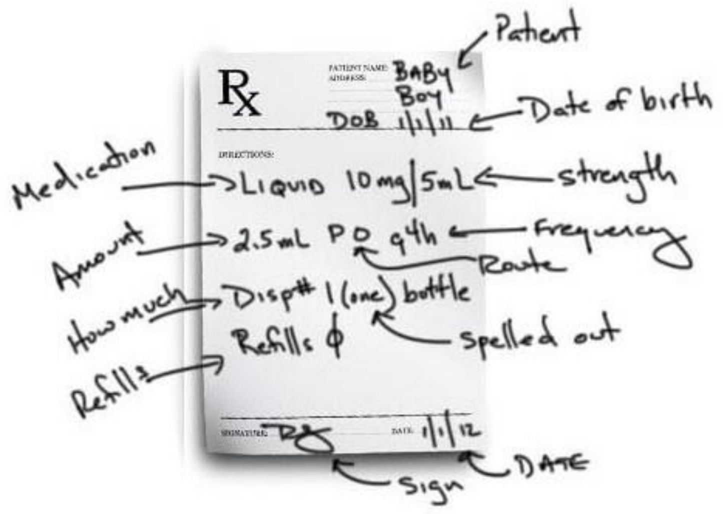


Figure 2: prescription

Input: prescription images.
Output: Treatment’s exact name

1.3 Objective

Implementing state-of-the-art of the offline handwriting text recognition (HTR) systems specifically doctor medical forms by working at the line level by transforming the text-line image into a sequence of feature vectors.

We are trying to recognize the handwritten medical forms of doctors and trying to get some pattern which will be used to guess what the most probable prescribed medication is We are providing prescribed medication’s name.

Researching on improving the current results by working on large dataset and robust network.
It serves people in critical cases and reduce number of death cases.
Implementing the final improved system as a mobile or web application to be ready to use and reach to large number of users to achieve our target and save humans.

1.4 Time Plan

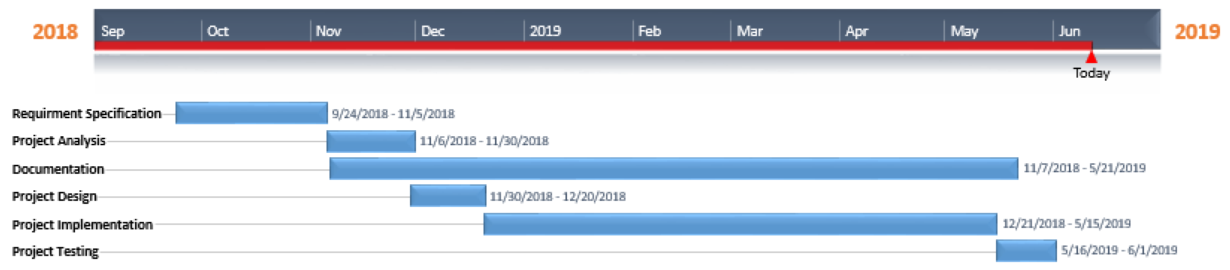


Figure 3: time plan

1.4 Document Organization

Chapter 2: A detailed description of the field of the project, all the scientific background related to the project, a survey of the work done in the field, description of existing similar systems and description of any technology used.

Chapter 3: the description of all modules and a functional and non-functional requirement and to whom the system is build, and how each group of users will use the system.
The Use case diagram, Class diagram and Sequence diagram.

Chapter 4: A detailed description of all the functions in the system and of all the techniques and algorithms implemented.
Description of any new technologies used in implementation.

Chapter 5: This chapter should describe in detail how to operate the project along with screen shots of the project representing all steps.
This chapter should also include an "Installation Guide" that would describe how to install the program, and all required third party tools that needs to be available for the project to run.
The installation guide will also be included as a readme file in the CDs delivered at the end of the year.

Chapter 6: A complete summary of the whole project along with results obtained and what can be done in the future to improve the performance of the project and what additional functions should be added.

References: The list of references used during the project or in writing the document. All references about the project filed which helped in implementation.

2- Background

2.1 Description of handwriting recognition filed:

Handwriting recognition is the ability of a machine to receive and interpret handwritten input from multiple sources like paper documents, photographs, touch screen devices etc. Recognition of handwritten and machine characters is an emerging area of research and finds extensive applications in banks, offices and industries.

The recognition of handwriting still is considered an open research problem due to its substantial variation in appearance.

Handwritten character recognition (HCR) is the process of conversion of handwritten text into machine readable form the major problem in handwritten character recognition (HCR) system is the variation of the handwriting styles, which can be completely different for different writers. The objective of handwritten character recognition system is to implement user friendly computer assisted character representation that will allow successful extraction of characters from handwritten documents and to digitalize and translate the handwritten text into machine readable text.

Handwritten character Recognition system is divided into two categories

- On-line character recognition.

It is system in which recognition is performed when characters are under creation.

- Off-line character recognition.

It is system in which first handwritten documents are generated, scanned, stored in computer and then they are recognized.

Most organizations use documents to acquire information from customers. These documents are generally handwritten. Such documents can be forms, checks, etc.

For their easier retrieval or information collection documents are transformed and stored in digital formats Common practice to handle that information is manually filling same data into computer.

It would be tiresome and time consuming to handle such documents manually hence the requirement of a special Handwritten Character Recognition Software arises which will automatically recognize texts from image of documents.

The process of extracting data from the handwritten documents and storing it in electronic formats has made easy by Handwritten Character Recognition (HCR) Software Banking sectors, Health care industries and many such organizations where handwritten documents are used regularly. HCR systems also find applications in newly emerging areas where handwriting data entry is required, such as development of electronic libraries, multimedia database etc.

2.2 All the scientific background related to the project:

Optical character recognition system has been studied in last many decades.
In 1914 Emanuel Goldberg developed a system that read handwritten character and digits and converted then into a telegraph code. At the same time Edmund Fournier dale developed the Otophone, a handheld scanner that scan the printed page and produced the output.
Goldberg continued to develop Handwriting recognition system for data entry. After some time, he proposed matching the images with the templates containing the desire identification. This technique is known as template matching method.
After that Paul Handel also proposed a US patent on template matching handwriting technology in USA in 1933.

In 1994 RCA engineers proposed first primitive computer type optical character recognition to help blind people. It designed to convert the handwritten report into punched cards for input in the computer for help in processing the shipment of 20-25
million books in a year.

In 1965 Reader’s Digest and RCA collaborated to build an optical character recognition system.

In 1985 structural approaches were proposed with the statistical methods. In these systems broke the characters into set of patterns like horizontal and vertical lines and different curves. In this method system focused on shape of the characters.

After 1990 the real progress is achieved using new techniques and methodologies in image processing and pattern recognition.

In today’s world more powerful computers and more accurate equipment like electronics pen, scanner and tablets are used. Many approaches like HMM, neural network, back propagation algorithm, fuzzy neural network is using to recognize handwritten documents.

2.2.1 Image Features

In computer vision and image processing, a feature is a piece of information which is relevant for solving the computational task related to a certain application. This is the same sense as feature in machine learning and pattern recognition generally, though image processing has a very sophisticated collection of features. Features may be specific structures in the image such as points, edges or objects. Features may also be the result of a general neighborhood operation or feature detection applied to the image.

Other examples of features are related to motion in image sequences, to shapes defined in terms of curves or boundaries between different image regions, or to properties of such a region.
The feature concept is very general and the choice of features in a particular computer vision system may be highly dependent on the specific problem at hand.

Introduction When features are defined in terms of local neighborhood operations applied to an image, a procedure commonly referred to as feature extraction, one can distinguish between feature detection approaches that produce local decisions whether there is a feature of a given type at a given image point or not, and those who produce non-binary data as result. The distinction becomes relevant when the resulting detected features are relatively sparse.

Although local decisions are made, the output from a feature detection step does not need to be a binary image. The result is often represented in terms sets of (connected or unconnected) coordinates of the image points where features have been detected, sometimes with sub-pixel accuracy.

When feature extraction is done without local decision making, the result is often referred to as a feature image. Consequently, a feature image can be seen as an image in the sense that it is a function of the same spatial (or temporal) variables as the original image, but where the pixel values hold information about image features instead of intensity or color. This means that a feature image can be processed in a similar way as an ordinary image generated by an image sensor. Feature images are also often computed as integrated step in algorithms for feature detection.

2.2.2 Artificial Intelligence (AI)

Artificial Intelligence (AI) is usually defined as the science of making computers do things that require intelligence when done by humans. AI has had some success in limited, or simplified, domains. However, the five decades since the inception of AI have brought only very slow progress, and early optimism concerning the attainment of human-level intelligence has given way to an appreciation of the profound difficulty of the problem.

AI Goal:
The overall research goal of artificial intelligence is to create technology that allows computers and machines to function in an intelligent manner. The general problem of simulating (or creating) intelligence has been broken down into sub-problems. These consist of particular traits or capabilities that researchers expect an intelligent system to display. There are many traits described below have received the most attention. Reasoning, problem solving, Knowledge representation, Planning, Machine Learning, Natural language processing, Perception, Social intelligence.

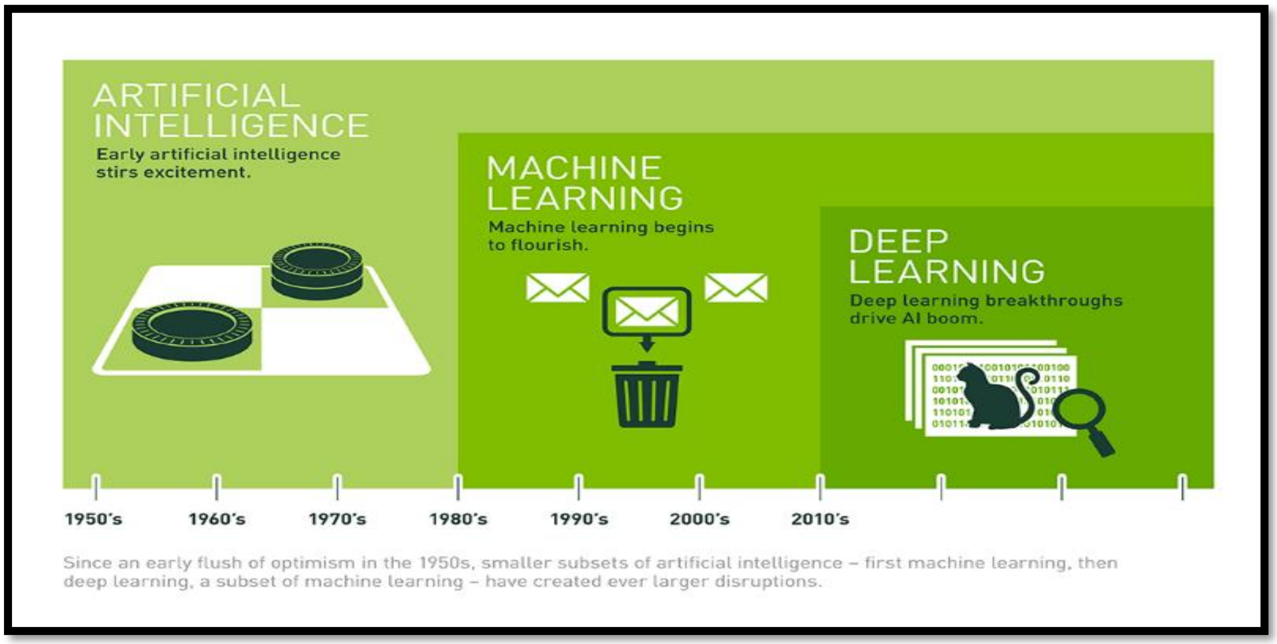

ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.
MACHINE LEARNING
Machine learning begins to flourish.
DEEP LEARNING
Deep learning breakthroughs drive AI boom.
1950's
1960's
1970's
1980's
1990's
2000's
2010's
Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Figure 4: describe AI and ML and DL

2.2.3 Machine learning (ML)

Machine learning is the study of computer algorithms that improve automatically through experience and has been central to AI research since the field's inception.
Machine learning explores the study and construction of algorithms that can learn from and make predictions on data such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions through building a model from sample inputs.

Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or unfeasible; example applications include email filtering, detection of network intruders or malicious insiders working towards a data breach, optical character recognition (OCR), learning to rank and computer vision. Learning process is divided into two categories: **supervised**, **unsupervised** learning and **reinforcement learning.**

2.2.4 Supervised Learning

The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.

Supervised learning includes both classification and numerical regression. Classification is used to determine what category something belongs in, after seeing a number of examples of things from several categories. Regression is the attempt to produce a function that describes the relationship between inputs and outputs and predicts how the outputs should change as the inputs change.

2.2.5 Unsupervised Learning

Unsupervised Learning No labels are given to the learning algorithm, leaving it on its own to find structure in its input.

Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning). The machine task of inferring a function to describe hidden structure from "unlabeled" data (a classification or categorization is not included in the observations).

Since the examples given to the learner are unlabeled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm which is one way of distinguishing unsupervised learning from supervised learning and reinforcement learning.

2.2.6 Neural Network (NN)

A neural network is a machine learning algorithm based on the model of a human neuron. The human brain consists of millions of neurons.
A beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data based on the structure and functions of biological neural networks.
It intended to simulate the behavior of biological systems composed of "**neurons**". ANNs are computational models inspired by an animal's central nervous systems. It is capable of **machine learning** as well as pattern recognition. These presented as systems of interconnected "**neurons**" which can compute values from inputs.

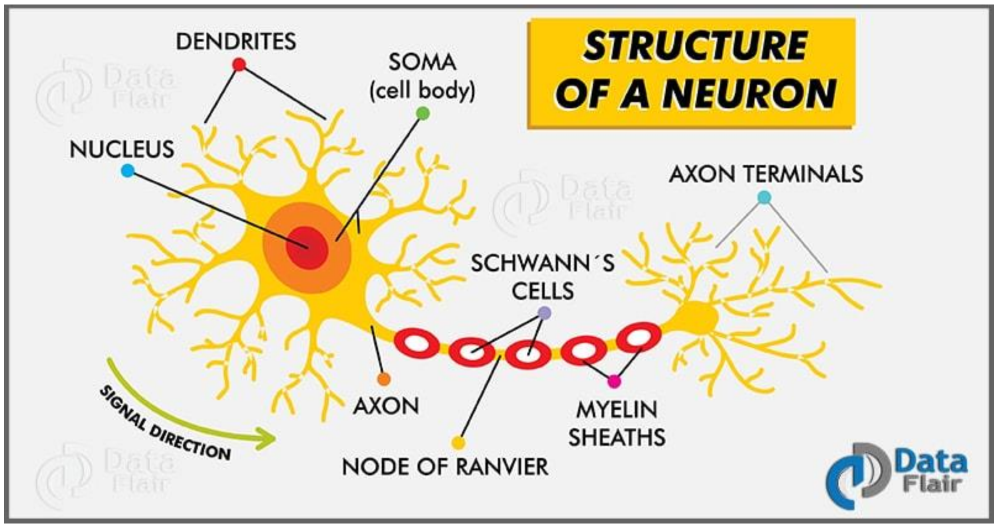


Figure 5: Structure of a human neuron.

A neural network is an oriented graph. It consists of nodes which in the biological analogy represent neurons, connected by arcs. It corresponds to dendrites and synapses. Each arc associated with a weight while at each node. Apply the values received as input by the node and define Activation function along the incoming arcs, adjusted by the weights of the arcs.

2.2.7 Artificial Neural Network (ANN)

An Artificial Neural Network is an information processing technique. It works like the way human brain processes information.
ANN includes a large number of connected processing units that work together to process information. They also generate meaningful results from it. We can apply neural network not only for classification. It can also apply for regression of continuous target attributes.
Neural networks find great application in data mining used in sectors. For example, economics, forensics and for pattern recognition. It can be also used for data classification in a large amount of data after careful training.

A neural network may contain the following 3 layers:

- **Input layer** – The activity of the input units represents the raw information that can feed into the network.

- **Hidden layer** – To determine the activity of each hidden unit. The activities of the input units and the weights on the connections between the input and the hidden units. There may be one or more hidden layers.

- **Output layer** – The behavior of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

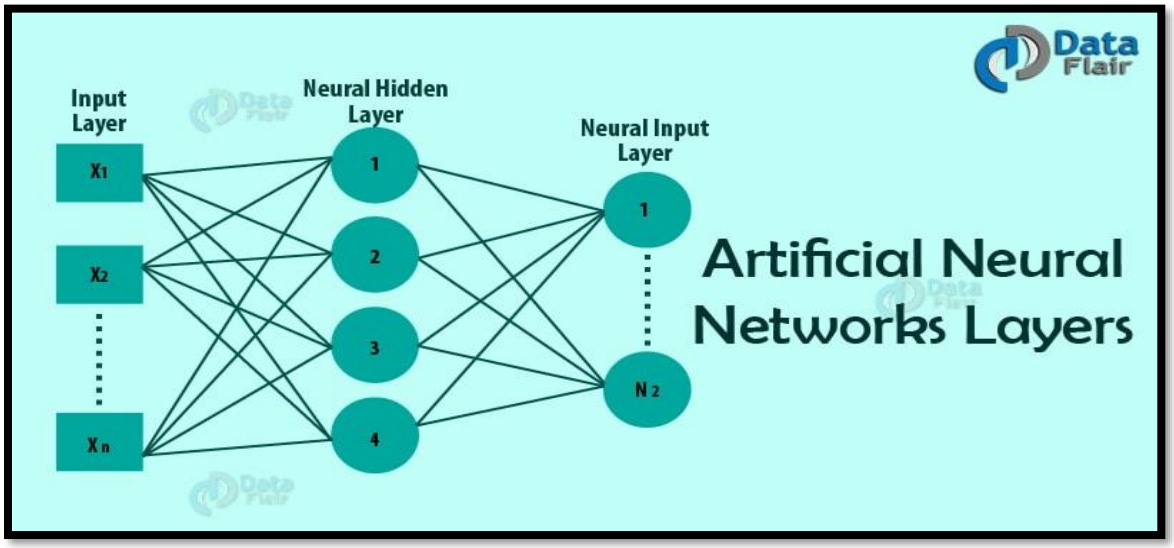


Figure 6: ANN layers

a. Input layer

The purpose of the input layer is to receive as input the values of the explanatory attributes for each observation. Usually, the number of input nodes in an input layer is equal to the number of explanatory variables. ‘Input layer’ presents the patterns to the network, which communicates to one or more ‘hidden layers’.

The nodes of the input layer are passive, meaning they do not change the data. They receive a single value on their input and duplicate the value to their many outputs. From the input layer, it duplicates each value and sent to all the hidden nodes.

b. Hidden layer

The Hidden layers apply given transformations to the input values inside the network. In this, incoming arcs that go from other hidden nodes or from input nodes connected to each node. It connects with outgoing arcs to output nodes or to other hidden nodes. In hidden layer, the actual processing is done via a system of weighted connections. There may be one or more hidden layers. The values entering a hidden node multiplied by weights, a set of predetermined numbers stored in the program. The weighted inputs are then added to produce a single number.

c. Output layer

The hidden layers then link to an ‘output layer ‘. Output layer receives connections from hidden layers or from input layer. It returns an output value that corresponds to the prediction of the response variable. In classification problems, there is usually only one output node. The active nodes of the output layer combine and change the data to produce the output values.

The ability of the neural network to provide useful data manipulation lies in the proper selection of the weights. This is different from conventional information processing

2.2.8 Convolutional Neural Network (CNN)

Convolutional neural networks were inspired by biological processes. The connectivity pattern between neurons is inspired by the animal visual cortex. Although, Individual cortical neurons are used to respond to stimuli that are only in a restricted region of the visual field known as the receptive field. Further, this field of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms and they have applications in image and video recognition, recommender systems and natural language processing.

Architecture of Convolutional Neural Networks:

- Convolutional layer
- Pooling layer
- Flatten layer
- Fully connected layer

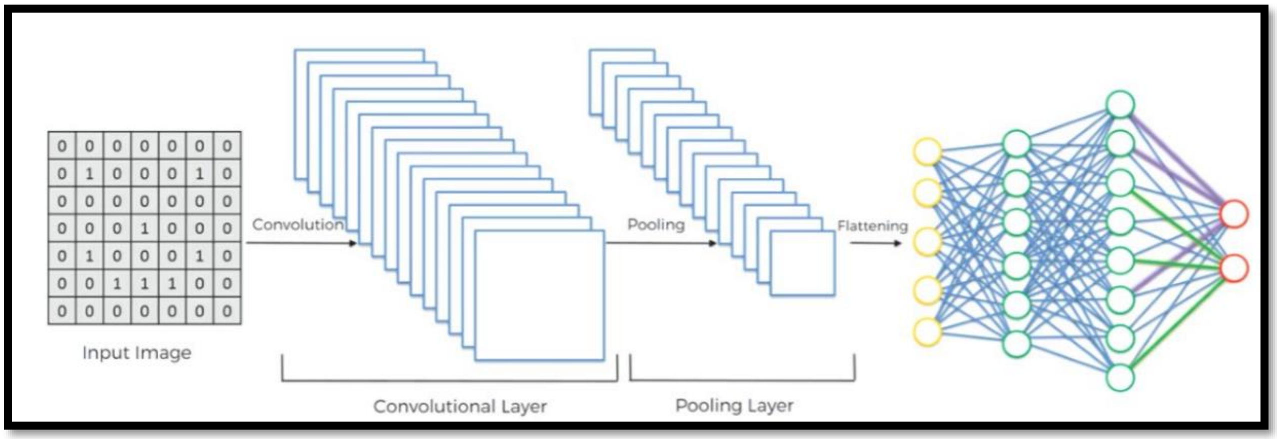


Figure 7: Architecture of CNN.

❖ Convolutional layer (First Layer)

The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable **filters** (or kernels), each of these filters can be thought of as **feature identifiers** which have a small receptive field but extend through the full depth of the input volume.

During the forward pass, each filter is **convolved** across the width and height of the input volume, computing the **dot product** between the entries of the filter and the input and producing a 2-dimensional activation map of that filter. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input

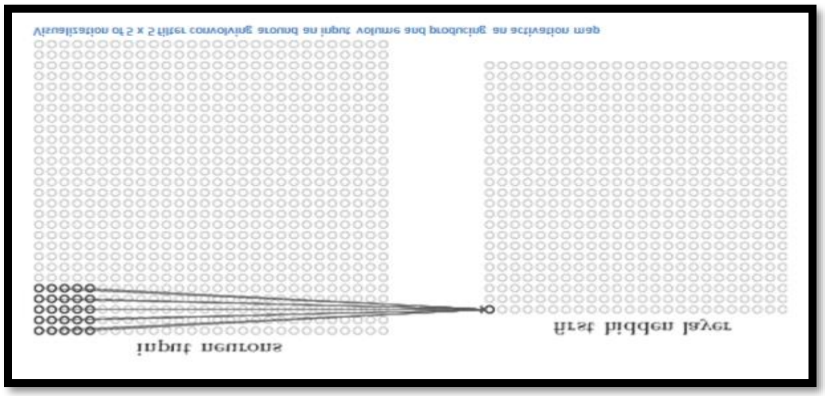

Visualization of 5 x 5 filter convolving around an input volume and producing an activation map
input neurons
first hidden layer

Figure 8: apply 5x5 filter convolving input to produce Activation map.

To visualizing this **mathematically**. When we have this filter at the top left corner of the input volume, it is computing multiplications between the filter and pixel values at that region. Now let's take an example of an image that we want to classify, and let's put our filter at the top left corner.

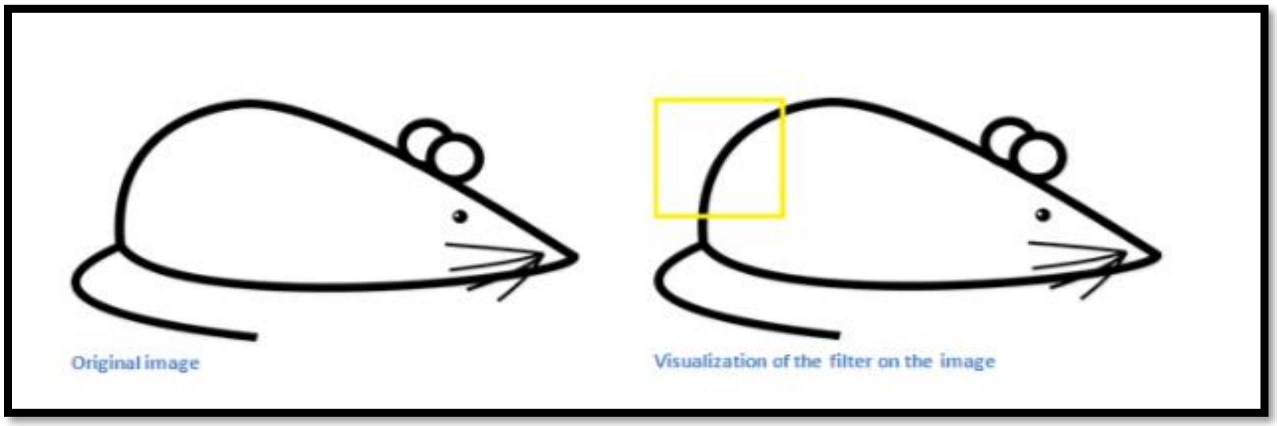
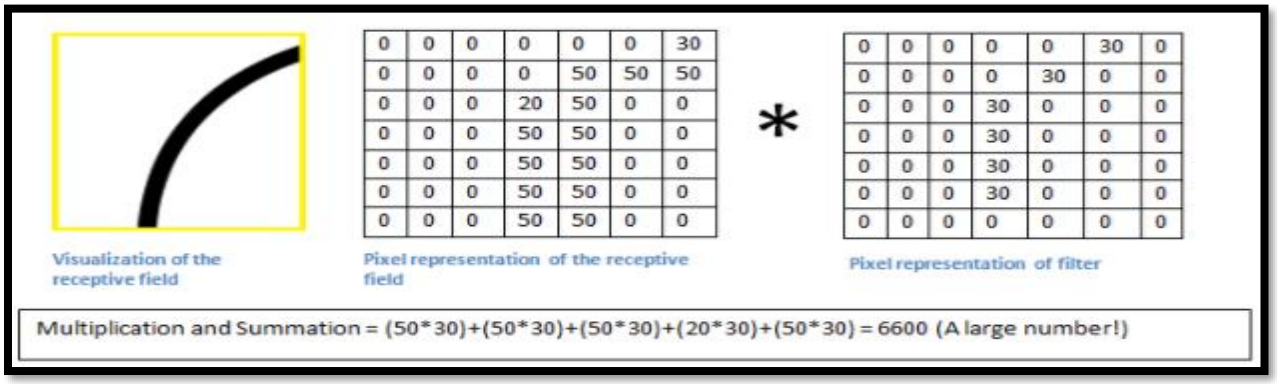


Figure 9: *apply the filter convolving input.*

Remember, what we have to do is multiply the values in the filter with the original pixel values of the image.


Visualization of the receptive field
Pixel representation of the receptive field
*
Pixel representation of filter

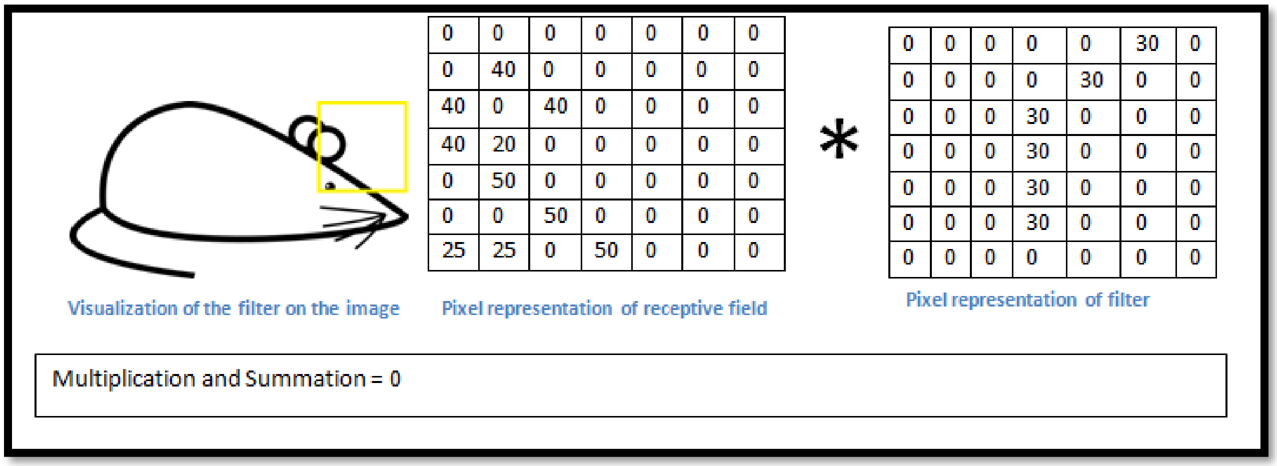
0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Figure 10: *apply filter with the original pixel values of the image.*

Basically, in the input image, if there is a shape that generally resembles the curve that this filter is representing, then all of the multiplications summed together will result in a large value! Now let's see what happens when we move our filter.


Visualization of the filter on the image
Pixel representation of receptive field
*
Pixel representation of filter
Multiplication and Summation = 0

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Figure 11: we move our filter to another original pixel values of the image.

❖ Pooling layer

Another important concept of CNNs is pooling, which programmers choose to apply a **pooling layer**. It is also referred to **as a non-linear down-sampling layer**. In this category, there are also several layer options, with "max-pooling" being the most popular. This basically takes a filter (normally of size 2x2) and a stride of the same length. It then applies it to the input volume and outputs the maximum number in every sub-region that the filter convolves around.

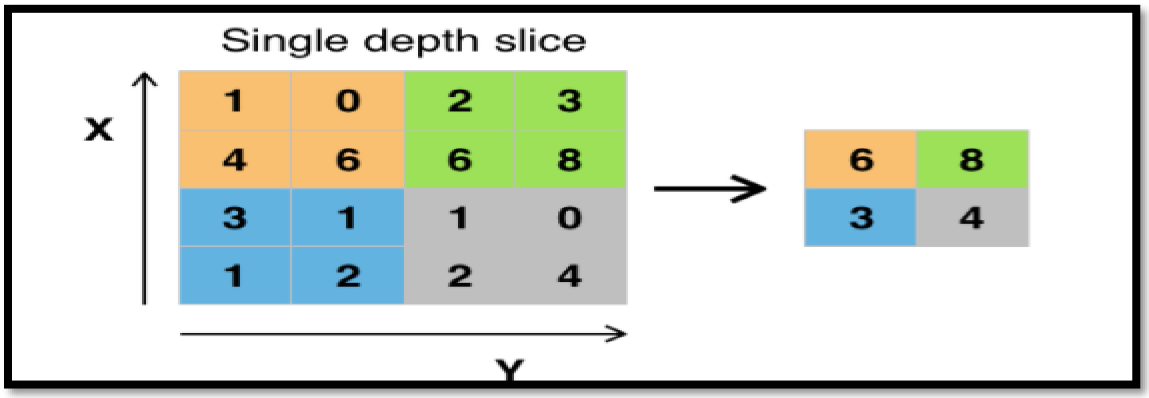


Figure 12: Max-pool with a 2x2 filter and a stride 2.

❖ Flatten layer

Using the pooled feature map, we flatten it into one column. Starting from the top row, left to right.

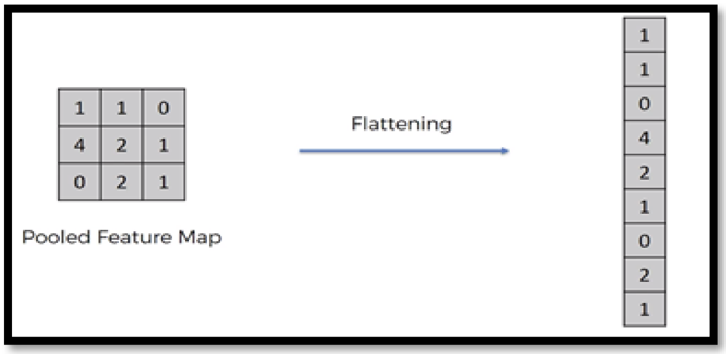


Figure 13: flatten Features map to one vector.

This is done so that we can fit this into an Artificial Neural Network for further processing.

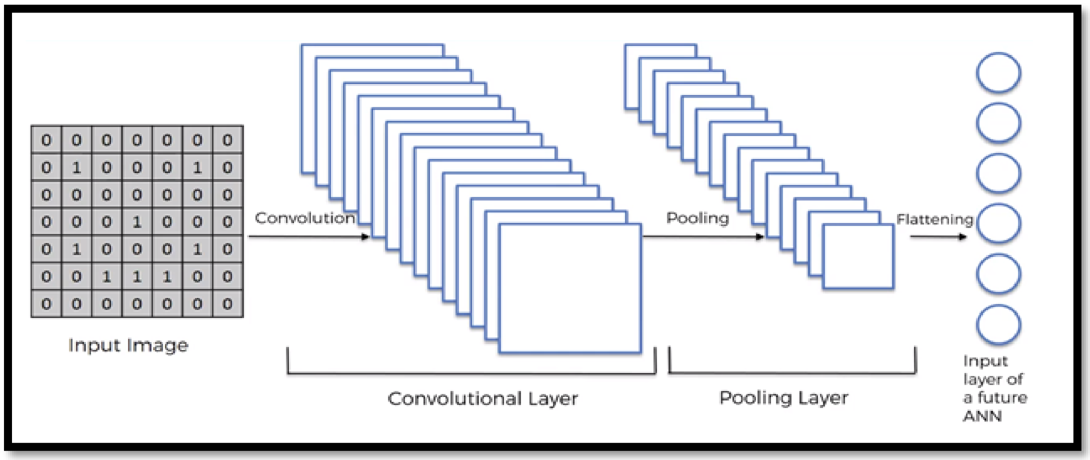


Figure 14: All Layers of CNN.

❖ Fully Connected layer

This adds an Artificial Neural Network onto the flattened input image. The hidden layers inside a Convolutional Neural Network are called Fully Connected Layers. These are a specific type of hidden layer which must be used within the CNN.

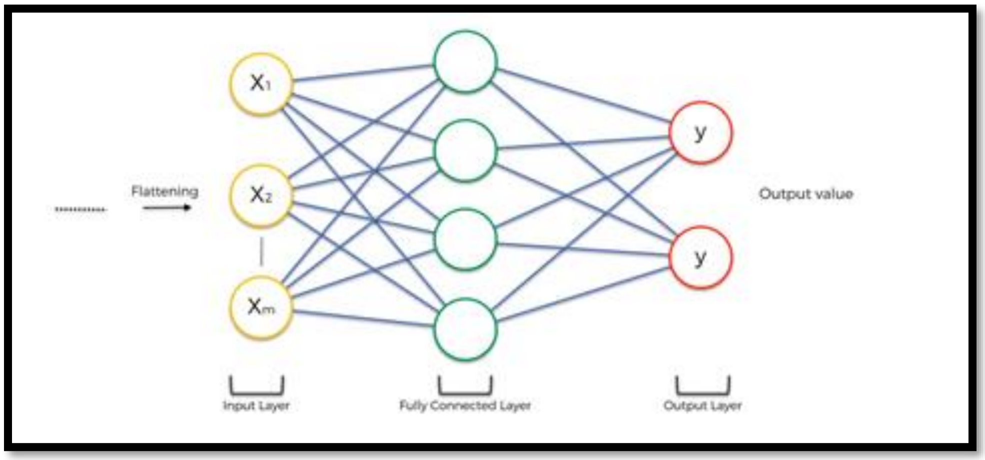


Figure 15: Fully Connected Layers.

Using Fully Connected Layers the output layer is passed into the input layer. This is used to combine the features into more attributes that predict the outputs (classes) more accurately. Once flattened, some features are already encoded from the vector, but the ANN builds on that and improves it.

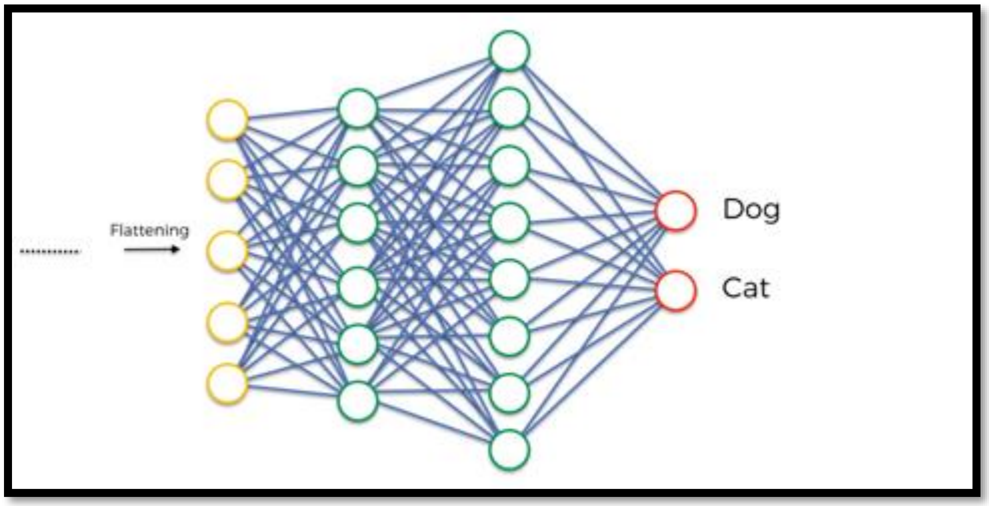


Figure 16: fully connected layer representation.

2.2.9 Loss Functions

Errors in CNNs are called a Loss Function which uses Cross-Entropy and mean-squared.
When the CNN is Back-propagated we adjust the weights and adjust the feature map to determine if we are using the correct one. Having multiple output layers, we need to understand what weights we assign to the synapses to each output

2.2.10 SoftMax Function

The SoftMax function is used to provide output values with a probability between each of them. For example: there is an image of a dog that has been passed through a CNN, the machine decides that the image is 0.95 likely to be a dog and 0.05 likely to be a

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

Figure 17: SoftMax Function

The original value is squashed into much smaller values that both add up to 1 to allow the machine to provide a suitable probability of each image.

2.2.11 Cross-Entropy Function

Cross-Entropy is a function that comes hand-in-hand with SoftMax. The original formula is:

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

Figure 18: Cross-Entropy Function

The function we are going to use (as it's easier to calculate) is:

$$H(p,q) = -\sum_x p(x)\,\log q(x)$$

Figure 19: Cross-Entropy function

A Cross-Entropy function is used to calculate the Loss Function and helps to get a neural network to an optimal state. This method is the preferred method for classification. If using regression, you would use Mean Squared Error.

2.2.12 Algorithms

The main goal of training the deep convolution neural network is to find the suitable weights to map the input to desired output. Training process accomplished by pass the input to the input layer in CNN and compare the output with desired output and calculate the error. Then updating the weights using the pre-calculated error by propagate the error from output node back to lower layers.

- A famous Algorithm used to train and optimize CNN is Back-propagation algorithm that divide into two phases: -

- Forward-propagation.
- Back-propagation and update weights.

2.2.13 Forward propagation

▶ **Convolution Operation (Convolution Layer)**

To perform a convolution operation, the kernel slid across the input feature map in equal and finite strides. At each location, the product between each element of the kernel and the input feature map element it overlaps is computed and the results summed up to obtain the output at that current location.

This procedure is repeated using different kernels to form as many output feature maps as desired. The concept of weight sharing is used as demonstrated in the diagram below:

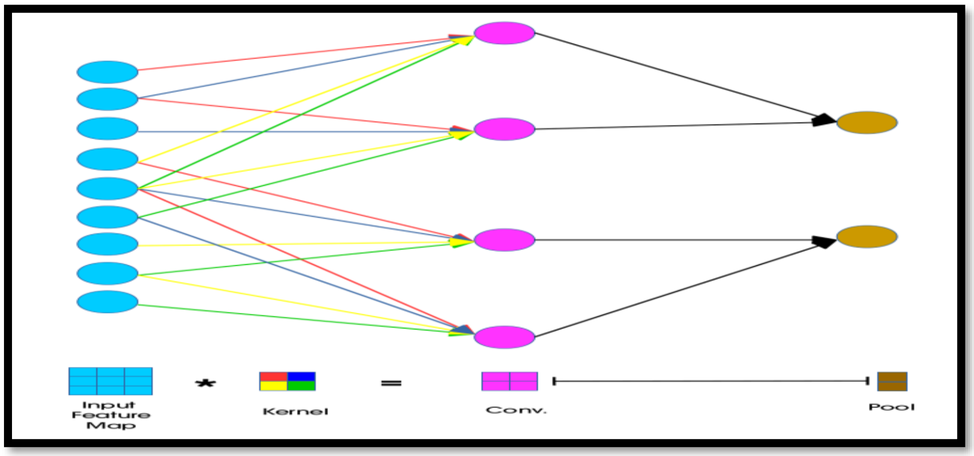


Figure 20: Convolution Layer operations.

Units in convolutional layer illustrated above have receptive fields of size 4 in the input feature map and are thus only connected to 4 adjacent neurons in the input layer. This is the idea of sparse connectivity in CNNs where there exists local connectivity pattern between neurons in adjacent layers.

The color codes of the weights joining the input layer to the convolutional layer show how the kernel weights are distributed (shared) amongst neurons in the adjacent layers. Weights of the same color are constrained to be identical.

The convolution equation of the input at layer all is given by:

$$x^l_{i,j} = \sum_m \sum_n w^l_{m,n} o^{l-1}_{i+m,j+n} + b^l_{i,j}$$

Figure 21: convolution equation

Then apply activation function:

$$o^l_{i,j} = f(x^l_{i,j})$$

Figure 22: activation function

Like: RELU = MAX (0, X).

This is illustrated below:

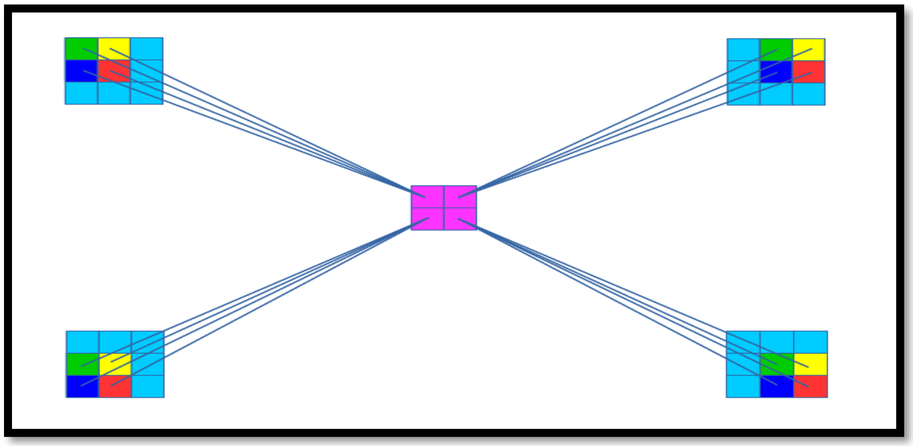


Figure 23: Convolution Layer operations details.

2. Pooling operation (Pooling Layer)

The function of the pooling layer is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control over fitting. No learning takes place on the pooling layers.

Pooling units are obtained using functions like max-pooling, average pooling and even L2-norm pooling but max pooling being the most popular.

This basically takes a filter (normally of size 2x2) and a stride of the same length. It then applies it to the input volume and outputs the maximum number in every sub region that the filter convolves around as shown below.

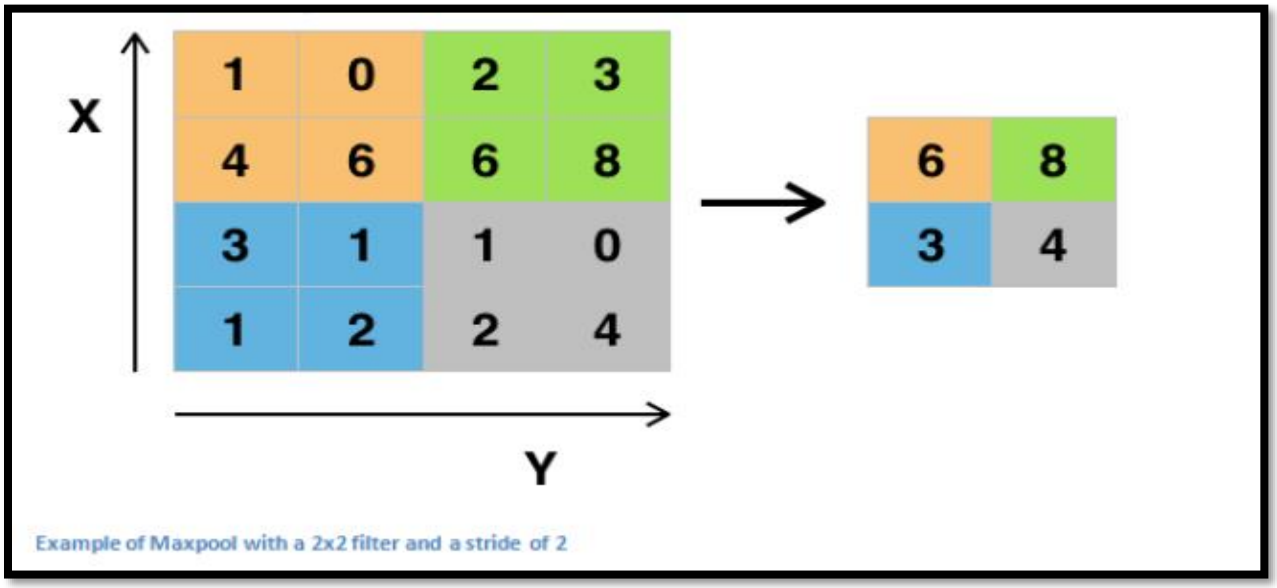


Figure 24: max-pooling operation.

3. Fully Connected Layer

Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks.

Soft-max regression applies soft-max nonlinearity to the output of the network.

2.2.14 Back propagation and update weights

▶ Error

For a total of P predictions, the predicted network outputs yp and their corresponding targeted values tp the mean squared error is given by:

$$E = \frac{1}{2} \sum_{p} (t_p - y_p)^2$$

Figure 25: Error Function.

Learning will be achieved by adjusting the weights such that yp is as close as possible or equals to corresponding tp. In the classical backpropagation algorithm, the weights are changed according to the gradient descent direction of an error surface E.

▶ Back-propagation

We are looking to compute $\frac{\partial E}{\partial w^{l}_{m',n'}}$ which can be interpreted as the measurement of how the change in a single pixel ′wm′, n′ in the weight kernel affects the loss function E.

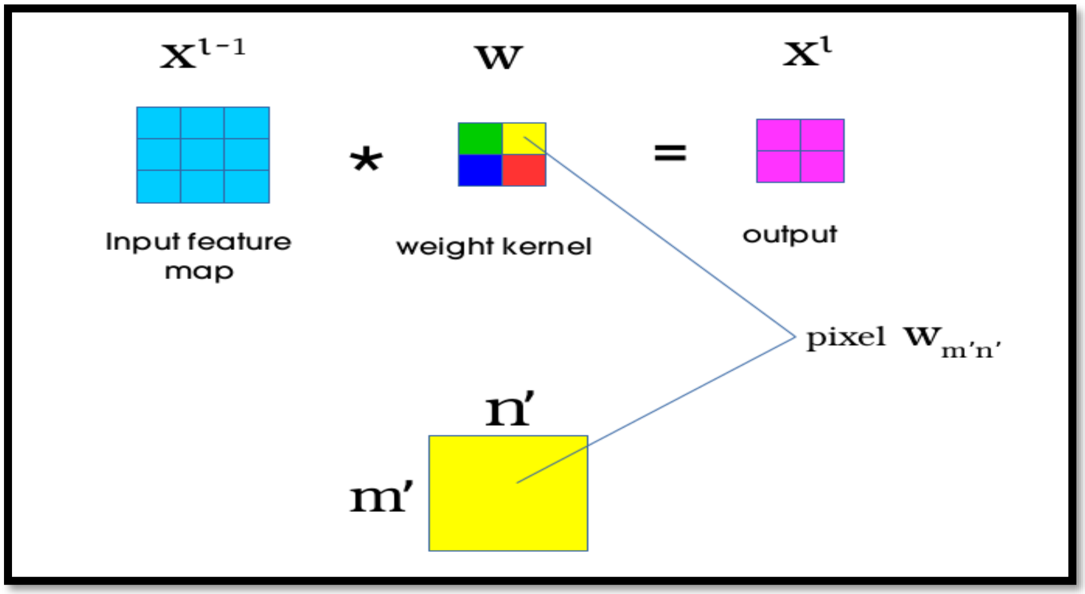


Figure 26: Back-propagation details.

Assume that get ∂O as input (from backward pass f the next layer) and our aim to calculate ∂W, by product matrix of gradient decent of output and input feature map.

Update weights of kernel by this formula:

$$\frac{\partial E}{\partial w^{l}_{m',n'}} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \delta^{l}_{i,j} o^{l-1}_{i+m',j+n'}$$

Figure 27: Update weight Function.

The diagram below shows gradients (δ11, δ12, δ21, δ22) generated during back-propagation:

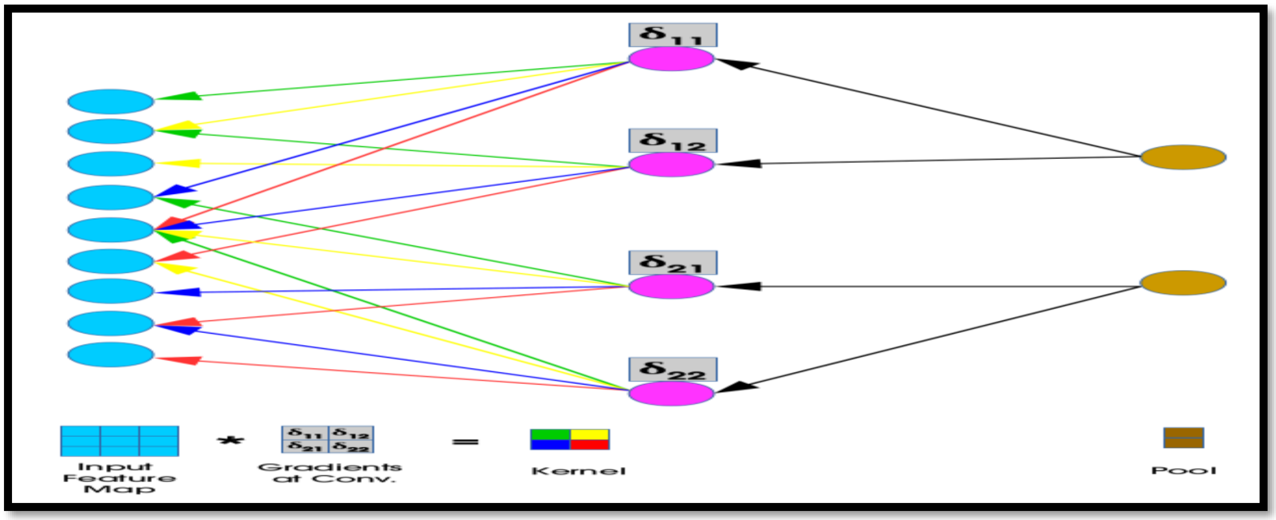


Figure 28: *Gradients generated during Back-propagation.*

The convolution operation used to obtain the new set of weights as is shown below:

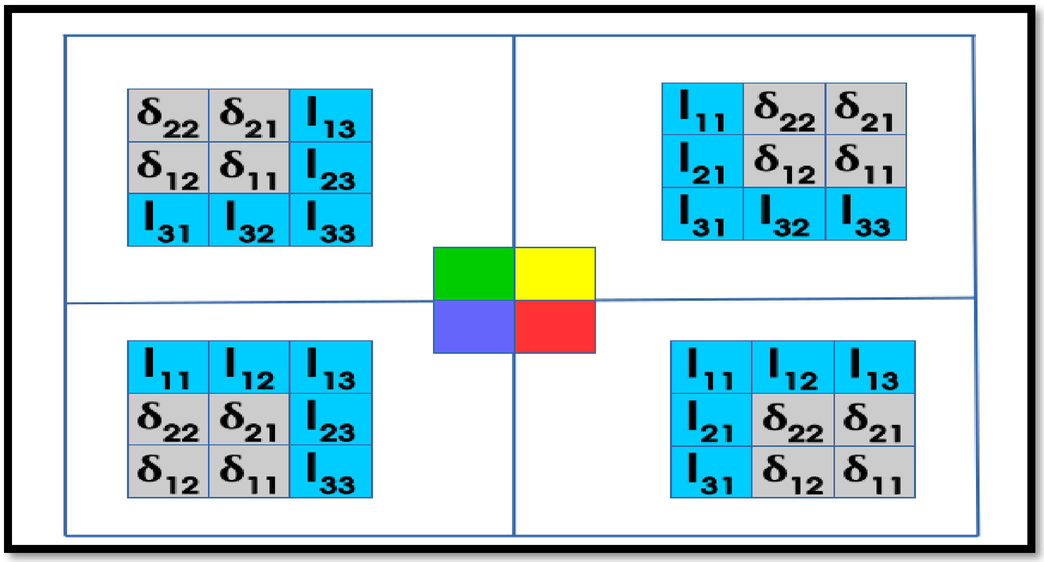


Figure 29: *Obtaining the new set of weights.*

2.3 Description of existing similar systems:

2.3.1 Pen to Print - Convert handwriting to text:

Figure30: Pen to print app

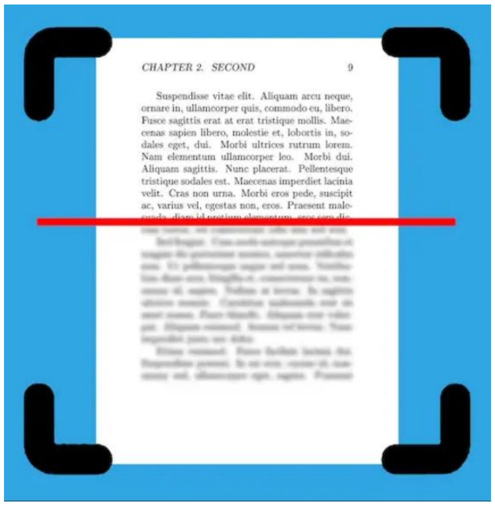
Pen to Print is the first handwriting to text OCR app converting scanned handwritten notes into digital text available for edits, search and storage in any digital platform.
Use this unique OCR scanner to scan, recognize and convert handwritten documents into digital text that can be edited, searched and stored on any device or cloud service.

2.3.2 OCR Text Scanner:

Figure31: OCR Text Scanner app

Convert an image to text.
OCR-Text Scanner is app to recognize the characters from an image with high (95% to 100%) accuracy.
It turns your mobile phone to text scanner and translator

2.3.3 Image To Text Converter (OFFLINE):



CHAPTER 2. SECOND 9

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dic- [illegible]

[illegible]

Figure32: Image To Text Converter app

This is free Image to text converter (OFFLINE). And it is suitable for everyone, such as students, businessmen, journalists for their assignment and project.

You can convert any text which is on newspaper, book, screens, boards, etc.

2.3.4 My Script Nebo:

Figure33: My Script Nebo app

This app is specially designed for iPad and Android users for making notes, writing and converting it into digital text. And not just that, you can also organize all of your documents by assigning titles to them. Once you are finished with finalizing the document then you can save it as a Word file, HTML file or even a PDF.

2.3.5 Mazec:

Figure34: Mazec app

Mazec is a keyboard app that provides handwriting conversion to text in a variety of apps like email, notes and social posts**.**

3-Analysis and Design

3.1 System Overview

3.1.1 System Architecture

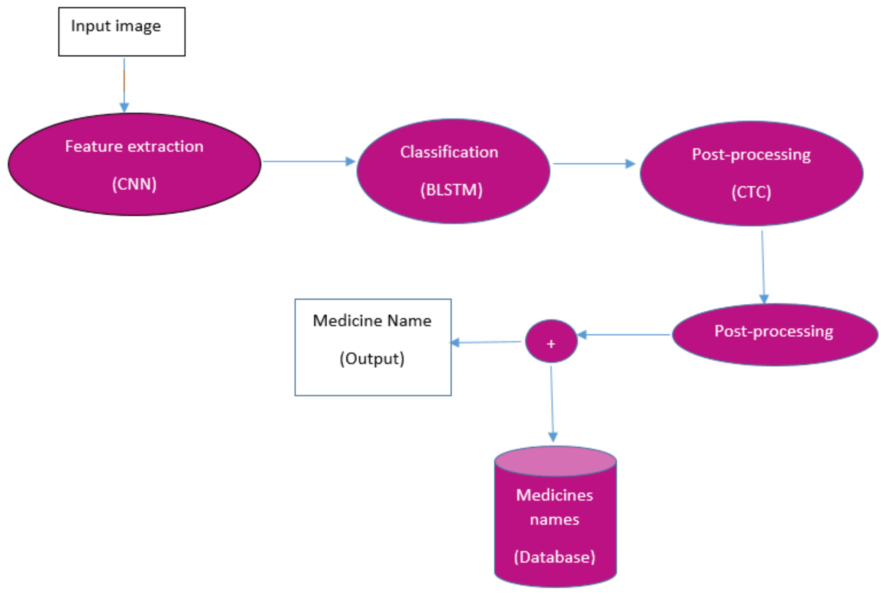


Figure35: System Architecture

- Preprocessing made manually by cropping name of medicine from prescription
  Height of Cropped image may be not equal 64 then we have two conditions if image height smaller than 64 we use padding to make image height 64 or if image height greater than 64 we use a factor to resize image to be 64.
- Postprocessing: We make edit distance between database and model distance.

3.1.2 System Users

Intended users:

- Patients use our system to read and know medicines from prescriptions.
- Pharmacist use our system to ensure that his read is correct

Figure36: prescription

3.2 System Analysis & Design

3.2.1 Use Case Diagram

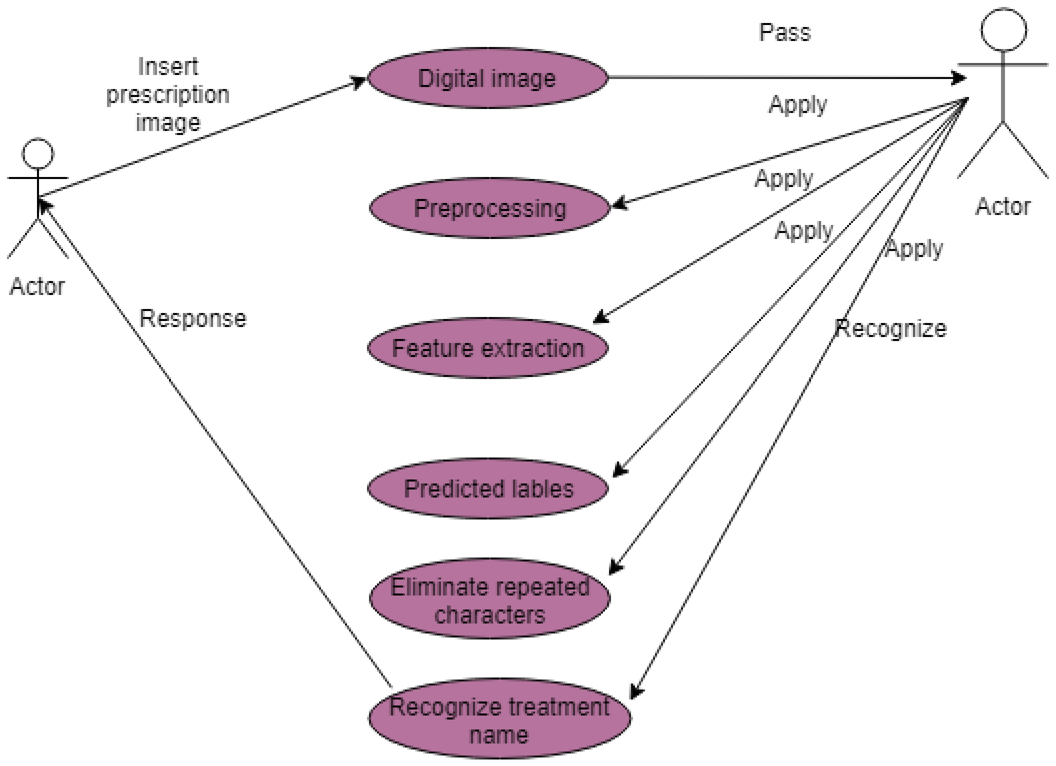


Figure37: Use Case Diagram

3.2.2 Sequence Diagram

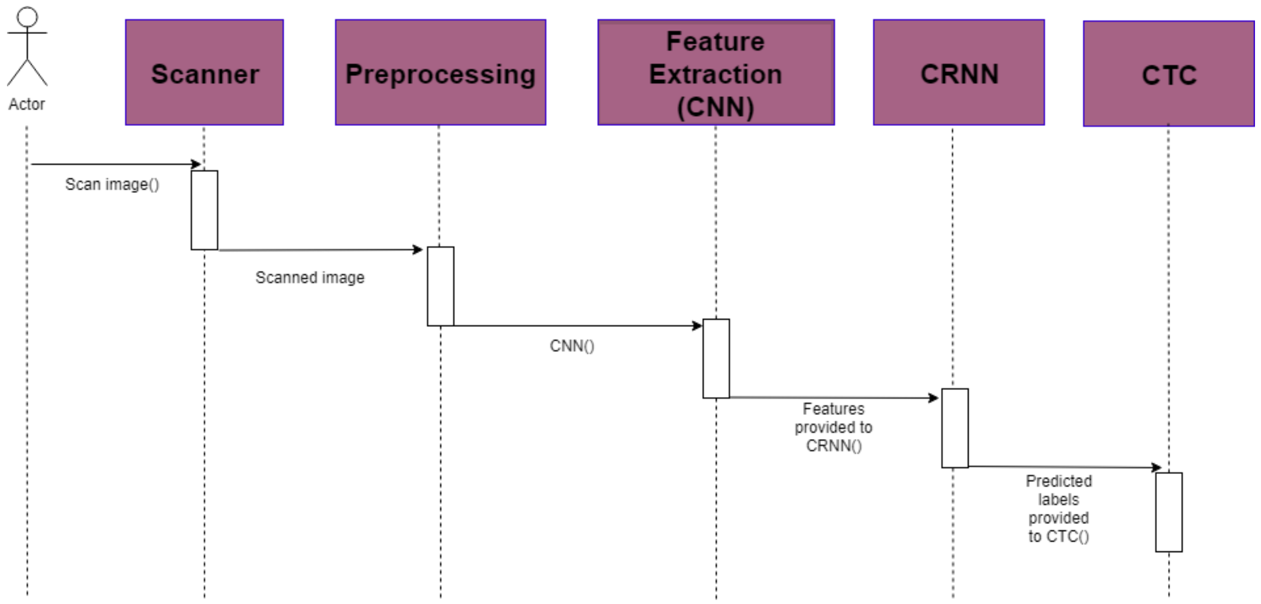
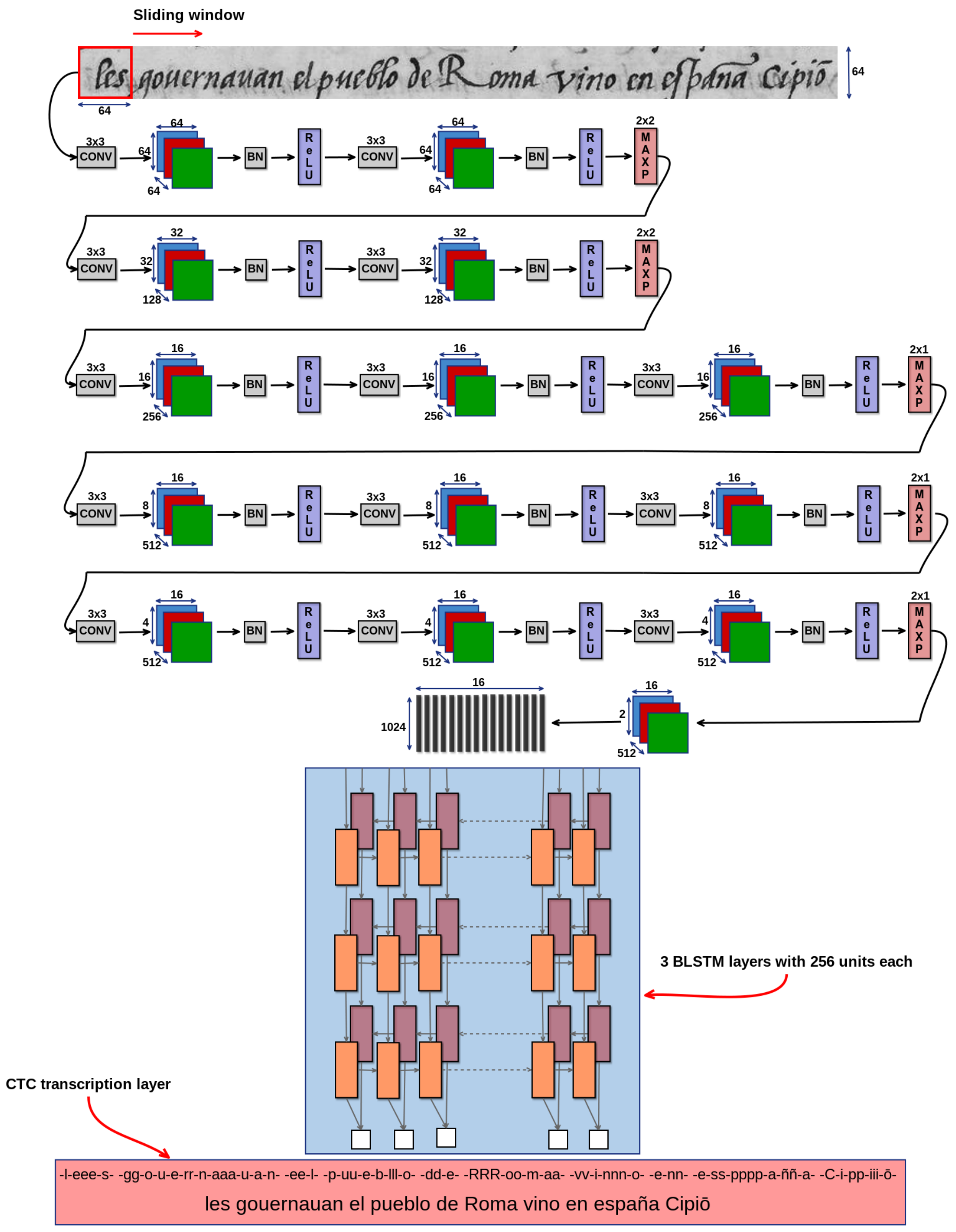


Figure38: Sequence Diagram

4- Implementation and Testing

Our system is a deep Convolutional Recurrent Neural Network (CRNN) inspired from the VGG16 architecture used for image recognition. We use a stack 13 convolutional (3 × 3 filters, 1 × 1 stride) layers followed by three Bidirectional LSTM layers with 256 units per layer. Each LSTM unit has one cell with enabled peephole connections. Special pooling (max) is employed after some convolutional layers. To introduce non-linearity, the Rectified Linear Unit (ReLU) activation function was used after each convolution. It has the advantage of being resistant to the vanishing gradient problem while being simple in terms of computation and was shown to work better than sigmoid and tanh activation functions. A square shaped sliding window is used to scan the text-line image in the direction of the writing. The height of the window is equal to the height of the text-line image, which has been normalized to 64 pixels. The window overlap is equal to 2 pixels to allow continuous transition of the convolution filters. For each analysis window of 64 × 64 pixels in size, 16 feature vectors are extracted from the feature maps produced by the last convolutional layer and fed into the observation sequence. It is worth noting that the amount of feature vectors extracted from each sliding windows is important. The number must be reasonable as to provide a good sampling for the image. Based on previous experiments, we found out that oversampling (32 feature vectors per window)
and under-sampling (8 feature vectors per window) will decrease the performance. Sixteen feature vectors were found to work best for our architecture. Since for each of the 16 columns of the last 512 feature maps, the columns of height 2 pixels are concatenated into a feature vector of size 1024 (512 × 2).

Thanks to the CTC objective function, the system is end-to-end trainable. The convolutional filters and the LSTM units' weights are thus jointly learned within the back-propagation procedure. We chose to keep the network simple with a relatively small number of parameters. We thus combine the forward and backward outputs at the end of the BLSTM stack rather than at each BLSTM layer. We also chose not to add additional fully-connected layers. The LSTM unit weights were initialized as per method, which proved to work well and helps the network convergence faster. This allows the network to maintain a constant variance across the network layers which keeps the signal from exploding to a high value or vanishing to zero.



Sliding window
64
3x3 CONV
BN
ReLU
2x2 MAXP
2x1 MAXP
1024
16
512
3 BLSTM layers with 256 units each
CTC transcription layer
-l-eee-s- -gg-o-u-e-rr-n-aaa-u-a-n- -ee-l- -p-uu-e-b-lll-o- -dd-e- -RRR-oo-m-aa- -vv-i-nnn-o- -e-nn- -e-ss-pppp-a-ññ-a- -C-i-pp-iii-ō-
les gouernauan el pueblo de Roma vino en españa Cipiō

Figure 39: Recognition System

4.1 Network:

4.1.1 Feature extractor:

VGG16 architecture here works as feature extractor, which consists of 13 layers of CNN and 5 Max Pooling layers. Input of our system is an image of variable size. A square shaped sliding window is used to scan the text-line image in the direction of the writing. The height of the window is equal to the height of the text-line image, which has been normalized to 64 pixels. The window overlap is equal to 2 pixels to allow continuous transition of the convolution filters. For each analysis window of 64 × 64 pixels in size, 16 feature vectors are extracted from the feature maps produced by the last convolutional layer and fed into the observation sequence. Since for each of the 16 columns of the last 512 feature maps, the columns of height 2 pixels are concatenated into a feature vector of size 1024 × 16

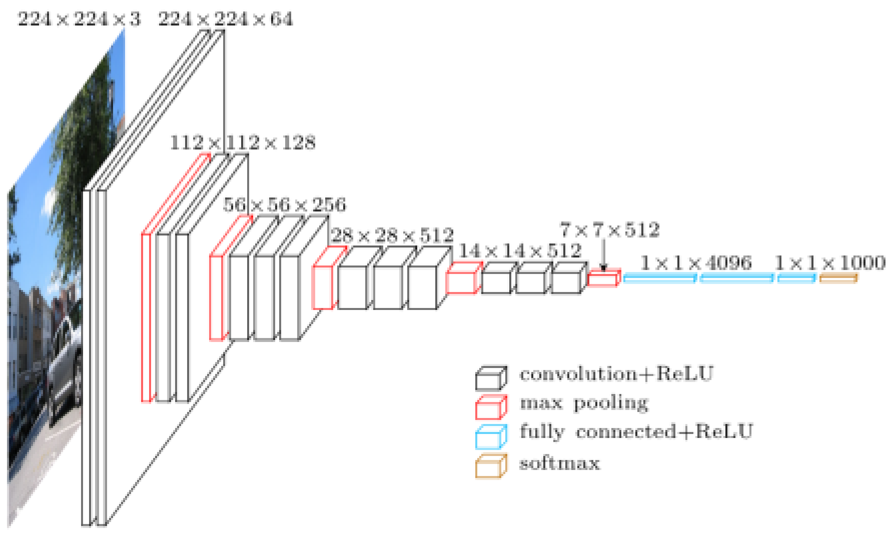


Figure 40: Visualization of the VGG architecture.

Convolutional neural networks are deep artificial neural networks that are used primarily to classify images (e.g. name what they see), cluster them by similarity (photo search), and perform object recognition within scenes. They are algorithms that can identify faces, individuals, street signs, tumors, platypuses and many other aspects of visual data.

Convolutional networks perform optical character recognition (OCR) to digitize text and make natural-language processing possible on analog and hand-written documents, where the images are symbols to be transcribed. CNNs can also be applied to sound when it is represented visually as a spectrogram. More recently, convolutional networks have been applied directly to text analytics as well as graph data with graph convolutional networks.

The efficacy of convolutional nets (ConvNets or CNNs) in image recognition is one of the main reasons why the world has woken up to the efficacy of deep learning. They are powering major advances in computer vision (CV), which has obvious applications for self-driving cars, robotics, drones, security, medical diagnoses, and treatments for the visually impaired.

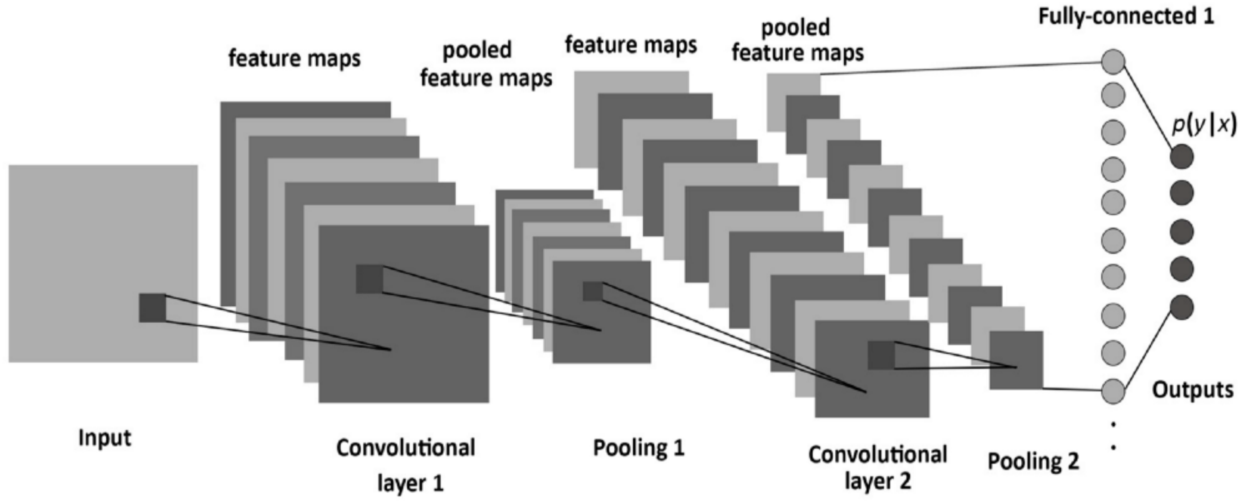


Figure 41: A diagram of Convolutional Neural Networks.

For a CNN that is used to describe different images like animals or faces. The features that the first convolutional layer looks for could be the different edges of the objects. It’s like making a list of the different edges in the picture. This list is then passed on to another convolutional layer which does a similar thing, except it looks for bigger shapes in the image. This could be a leg on an animal or an eye on a face. Ultimately, the features are taken in by a fully-connected layer which classifies the image.

4.1.2 RNN

Recurrent nets are a type of artificial neural network designed to recognize patterns in sequences of data, such as text, genomes, handwriting, the spoken word, or numerical times series data emanating from sensors, stock markets and government agencies. These algorithms take time and sequence into account, they have a temporal dimension. Research shows them to be one of the most powerful and useful type of neural network, alongside the attention mechanism and memory networks. RNNs are applicable even to images, which can be decomposed into a series of patches and treated as a sequence.

Since recurrent networks possess a certain type of memory, and memory is also part of the human condition, we'll make repeated analogies to memory in the brain.

Long Short-Term Memory Units (LSTMs)

In the mid-90s, a variation of recurrent net with so-called Long Short-Term Memory units, or LSTMs, was proposed by the German researchers Sepp Hoch Reiter and Juergen Schmid Huber as a solution to the vanishing gradient problem.

LSTMs help preserve the error that can be backpropagated through time and layers. By maintaining a more constant error, they allow recurrent nets to continue to learn over many time steps (over 1000), thereby opening a channel to link causes and effects remotely. This is one of the central challenges to machine learning and AI, since algorithms are frequently confronted by environments where reward signals are sparse and delayed, such as life itself. (Religious thinkers have tackled this same problem with ideas of karma or divine reward, theorizing invisible and distant consequences to our actions.)

LSTMs contain information outside the normal flow of the recurrent network in a gated cell. Information can be stored in, written to, or read from a cell, much like data in a computer's memory. The cell makes decisions about what to store, and when to allow reads, writes and erasures, via gates that open and close. Unlike the digital storage on computers, however, these gates are analog, implemented with element-wise multiplication by sigmoid, which are all in the range of 0-1. Analog has the advantage over digital of being differentiable, and therefore suitable for backpropagation.

Input of RNN is the feature map which was the output of the previous phase, Output is the recognized characters.

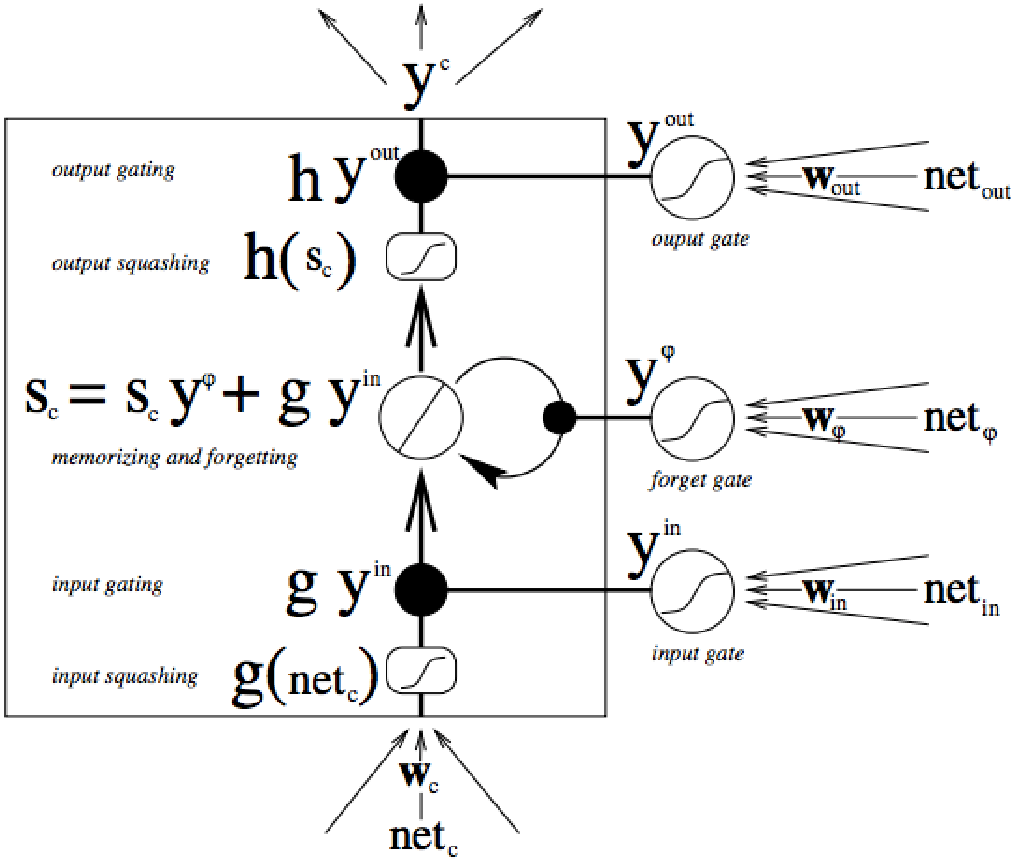


Figure 42: LSTM architecture.

It’s important to note that LSTMs’ memory cells give different roles to addition and multiplication in the transformation of input. The central plus sign in both diagrams is essentially the secret of LSTMs. Stupidly simple as it may seem, this basic change helps them preserve a constant error when it must be backpropagated at depth. Instead of determining the subsequent cell state by multiplying its current state with new input, they add the two, and that quite literally makes the difference.

4.1.3 CTC

If you want a computer to recognize text, neural networks (NN) are a good choice as they outperform all other approaches at the moment. The NN for such use-cases usually consists of convolutional layers (CNN) to extract a sequence of features and recurrent layers (RNN) to propagate information through this sequence. It outputs character-scores for each sequence-element, which simply is represented by a matrix. Now, there are two things we want to do with this matrix:

train: calculate the loss value to train the NN
infer: decode the matrix to get the text contained in the input image
Both tasks are achieved by the CTC operation. An overview of a handwriting recognition system is shown in Fig. 50.
Let’s have a closer look at the CTC operation and discuss how it works without hiding the clever ideas it is based on behind complicated formulas. At the end, I will point you to references where you can find Python code and the (not too complicated) formulas, if you are interested.

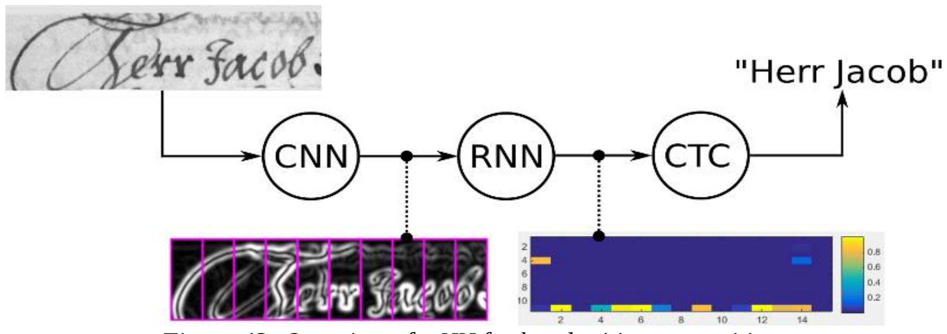


Figure 43: Overview of a NN for handwriting recognition.

4.1.3.1 Why we want to use CTC

We could, of course, create a data-set with images of text-lines, and then specify for each horizontal position of the image the corresponding character as shown in Fig. 51. Then, we could train a NN to output a character-score for each horizontal position. However, there are two problems with this naive solution:

- it is very time-consuming (and boring) to annotate a data-set on character-level.
- we only get character-scores and therefore need some further processing to get the final text from it. A single character can span multiple horizontal positions, e.g. we could get "ttooo" because the "o" is a wide character as shown in Fig. 51. We must remove all duplicate "t"s and "o"s. But what if
  the recognized text would have been "too"? Then removing all duplicate "o"s gets us the wrong result. How to handle this?

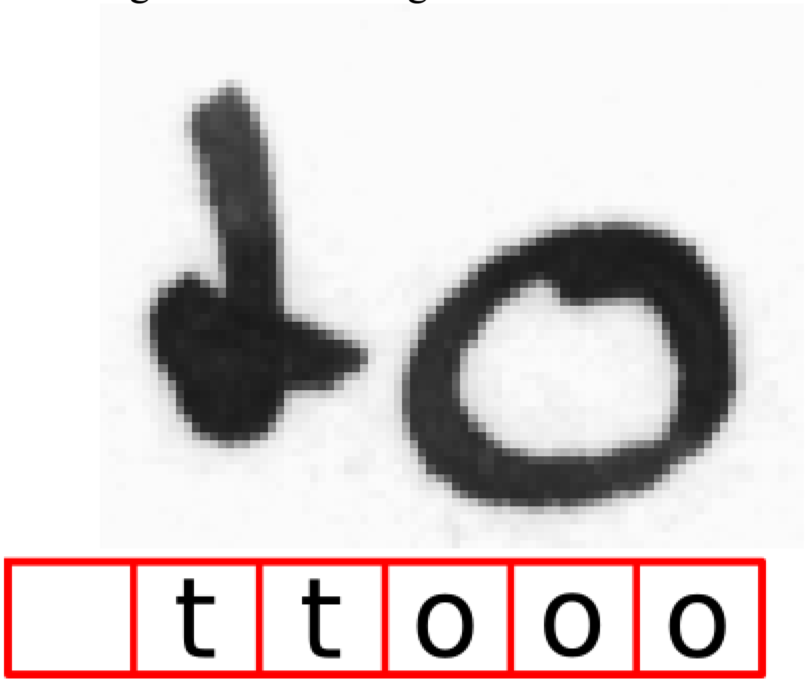


Figure 44: Annotation for each horizontal position of the image.

CTC solves both problems for us:

- we only have to tell the CTC loss function the text that occurs in the image. Therefore, we ignore both the position and width of the characters in the image.
- no further processing of the recognized text is needed.

4.1.3.2 How CTC works

As already discussed, we don't want to annotate the images at each horizontal position (which we call time-step from now on). The NN-training will be guided by the CTC loss function. We only feed the output matrix of the NN and the corresponding ground-truth (GT) text to the CTC loss function. But how does it know where each character occurs? Well, it does not know. Instead, it tries all possible alignments of the GT text in the image and takes the sum of all scores. This way, the score of a GT text is high if the sum over the alignment-scores has a high value.

4.1.3.3 Encoding the text

There was the issue of how to encode duplicate characters (you remember what we said about the word "too"?). It is solved by introducing a pseudo-character (called blank, but don't confuse it with a "real" blank, i.e. a white-space character). This special character will be denoted as "-" in the following text. We use a clever coding schema to solve the duplicate-character problem: when encoding a text, we can insert arbitrary many blanks at any position, which will be removed when decoding it. However, we must insert a blank between duplicate characters like in "hello". Further, we can repeat each character as often as we like.

Let's look at some examples:
- "to" → "---ttttttooo", or "-t-o-", or "to"
- "too" → "---ttttto-o", or "-t-o-o-", or "to-o", but not "too"

As you see, this schema also allows us to easily create different alignments of the same text, e.g. “t-o” and “too” and “-to” all represent the same text (“to”), but with different alignments to the image. The NN is trained to output an encoded text (encoded in the NN output matrix).

4.2 Training phase:

Convolutional Recurrent Neural Network (CRNN) inspired from the VGG16 architecture used for image recognition. We use a stack 13 convolutional (3 * 3 filters, 1 * 1 stride) layers followed by three Bidirectional LSTM layers with 256 units per layer. Each LSTM unit has one cell with enabled peephole connections. Spacial pooling (max) is employed after some convolutional layers. To introduce non-linearity, the Rectified Linear Unit (ReLU) activation function was used after each convolution.
Adam optimizer was used to train the network with
initial learning rate of 0.0005 With 30 epoch and patch size 10.

4.3 Testing Phase:

Our system is tested over around 3,000 of IAM database, and some of the cropped medications’ name of the collected medical forms. The images at first is resized to 64 × width of image. Then CNN is used to extract features and these features is fed to RNN to get scores of each window. After that these scores act like input to CTC to remove duplicated letters to get the final result of prediction.

Output of our system is a text file which contains the medications’ names of each cropped image and this text file is save on disk. We suppose to take these names and perform some post-processing on it to get the exact/desired medication name using edit distance.

4.3 Technologies used:

Here training model using Google Colab "Google Colaboratory"

- Google Colab is a free cloud service and now it supports free GPU.
    - Colab comes with libraries for accessing various Google services conveniently.
    - It saves to Google Drive, and you can share the notebooks easily without having to mess around hosting them.
    - Colaboratory allows you to use and share Jupyter notebooks with others without having to download, install, or run anything on your own computer other than a browser.
    - Improve your Python programming language coding skills.
    - Develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV.
    - The most important feature that distinguishes Colab from other free cloud services is; Colab provides GPU and is totally free
    - Free Tesla K80 GPU- using Keras, Tensorflow and PyTorch"

Figure 45: Google Collaboratory.

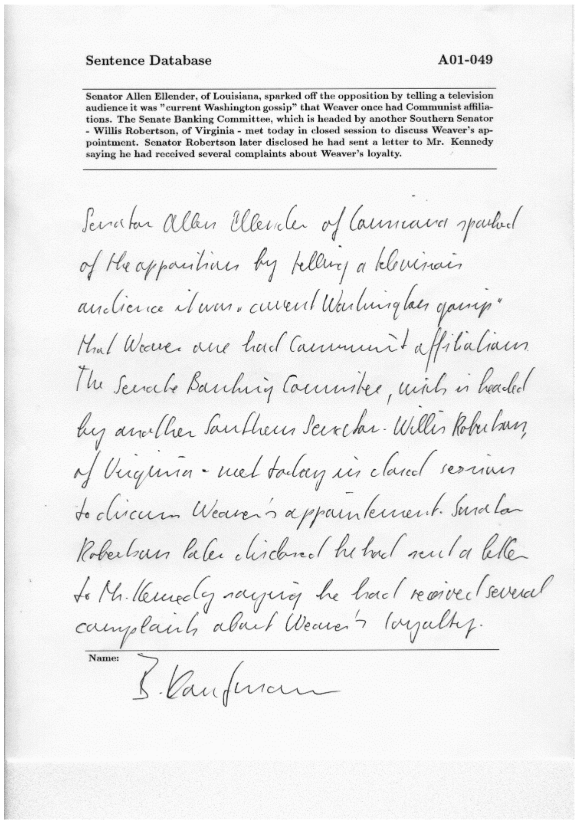
4.0 Dataset:

4.0.1 IAM dataset:

The IAM Handwriting Database contains forms of handwritten English text which can be used to train and test handwritten text recognizers and to perform writer identification and verification experiments.

The database was first published at the ICDAR 1999. Using this database an HMM based recognition system for handwritten sentences was developed and published at the ICPR 2000. The segmentation scheme used in the second version of the database is documented and has been published in the ICPR 2002. The IAM-database as of October 2002. We use the database extensively in our own research, see publications for further details.

The database contains forms of unconstrained handwritten text, which were scanned at a resolution of 300dpi and saved as PNG images with 256 gray levels. The figure below provides samples of a complete form.



Sentence Database **A01-049**

Senator Allen Ellender, of Louisiana, sparked off the opposition by telling a television audience it was "current Washington gossip" that Weaver once had Communist affiliations. The Senate Banking Committee, which is headed by another Southern Senator - Willis Robertson, of Virginia - met today in closed session to discuss Weaver's appointment. Senator Robertson later disclosed he had sent a letter to Mr. Kennedy saying he had received several complaints about Weaver's loyalty.

Senator Allen Ellender of Louisiana sparked
of the opposition by telling a television
audience it was "current Washington gossip"
that Weaver once had Communist affiliations
The Senate Banking Committee, which is headed
by another Southern Senator - Willis Robertson,
of Virginia - met today in closed session
to discuss Weaver's appointment. Senator
Robertson later disclosed he had sent a letter
to Mr. Kennedy saying he had received several
complaints about Weaver's loyalty.

Name: [illegible]

Figure 46: Sample of IAM database.

The IAM Handwriting Database 3.0 is structured as follows:

- 657 writers contributed samples of their handwriting
- 1'539 pages of scanned text
- 5'685 isolated and labeled sentences
- 13'353 isolated and labeled text lines
- 115'320 isolated and labeled words

The words have been extracted from pages of scanned text using an automatic segmentation scheme and were verified manually.

All form, line and word images are provided as PNG files and the corresponding form label files, including segmentation information and variety of estimated parameters, are included in the image files as meta-information in XML format which is described in XML file.

Here in our system we use IAM text lines to train our model using around 9,000 images for training and 3,000 images for testing. The figure below provides samples of a text line.

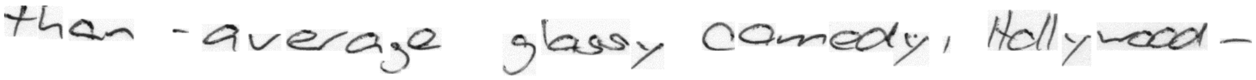

than-average glassy comedy, Hollywood-

Figure 47: Sample 1 of IAM text line.

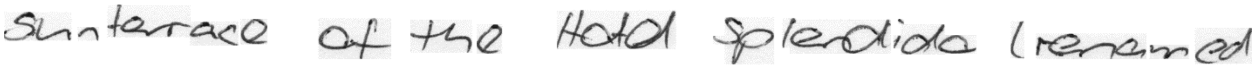
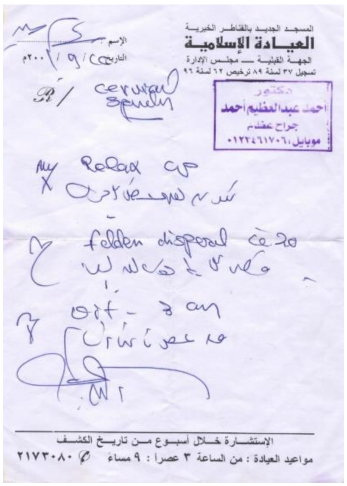

sunterrace of the Hotel Splendido (renamed

Figure 48: Sample 2 of IAM text line.

4.0.2 Other datasets:

We couldn't find medical forms as dataset due to the variations of medical forms all over the world. We collected a small sample of medical forms to fine-tune our model using it. Since our model is trained over text lines, we had to segment lines which consist of medication with hand.
Our system used around 500 prescription and cropped medicines from prescriptions we obtained around 800 medicine.
We combine 14000 images from IAM dataset with 700 medicine for training and 100 for testing.

The figure below provides samples of the prescriptions/medical forms we collected.


المسجد الجديد بالقناطر الخيرية
العيادة الإسلامية
الجهة القبلية ــــ مجلس الإدارة
تسجيل ٣٧ لسنة ٨٩ ترخيص ٦٢ لسنة ٩٦

الإسم
التاريخ ۲۲ / ۹ / ۲۰۰۱م

دكتور
أحمد عبدالعظيم أحمد
جراح عظام
موبايل: ٠١٢٢٤٦١٧٠٦

R/ cervical spondy

Relax cp

felden dispersal 20

oit-3 am

الإستشارة خلال أسبوع من تاريخ الكشف
مواعيد العيادة : من الساعة ٣ عصراً : ٩ مساء ٢١٧٣٠٨٠

Figure 49: Sample medical forms used in training.

Segmented lines are used to fine-tune our model, so we had to crop the medications' names out of the medical form. The figure below provides samples of cropped medications.

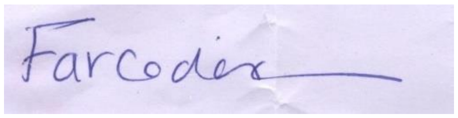


Figure 50: Sample 1 of medication.

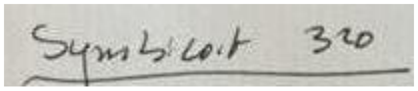

Symbicort 320

Figure 51: Sample 2 of medication.

4.0.3 Medications database:

We took database from pharmacy then we used it after filtering this database from cosmetics and medicine concentration to obtain medicine name only.
This database contains about 8500 medicine.

5- User Manual

In this chapter should describe in detail how to operate the project along with representing all steps in System Testing section and Installation Guide in System Deployment section.

4.1 System testing

Firstly, you should have a prescription saved on your hard disk. If you don't have one, you can capture it with mobile phone or scanner - preferred to be scanned- or a high-resolution image taken by phone.

You should run the project and select the prescription you want to know its medications' names by pressing on "Browse" button.

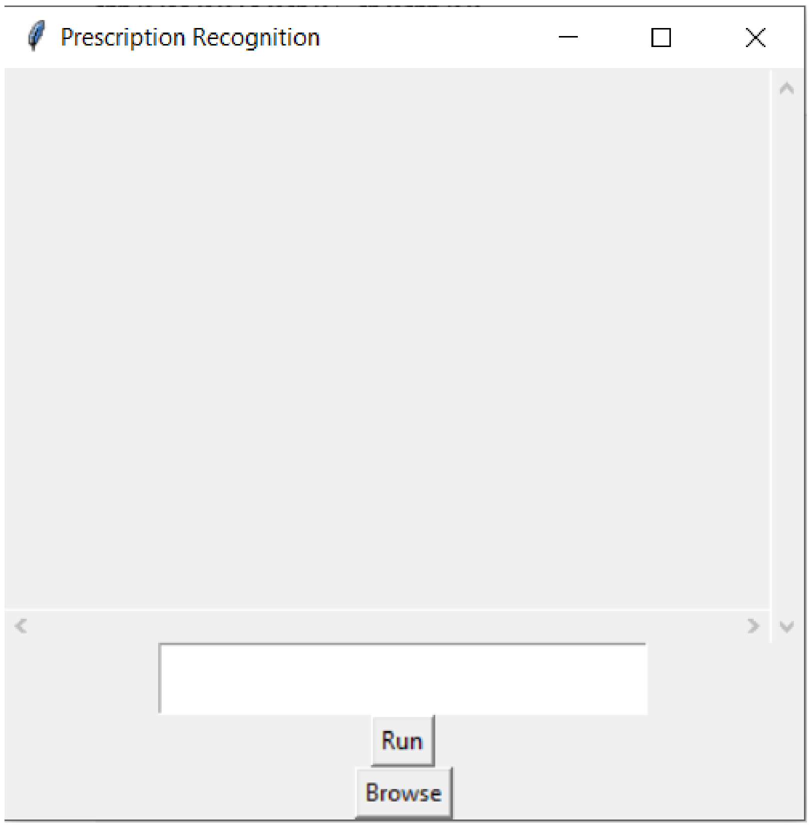


Figure 52: Prescription recognition form

A window named “Select file” will appear, so you will select the medical form you want to perform the project on it.

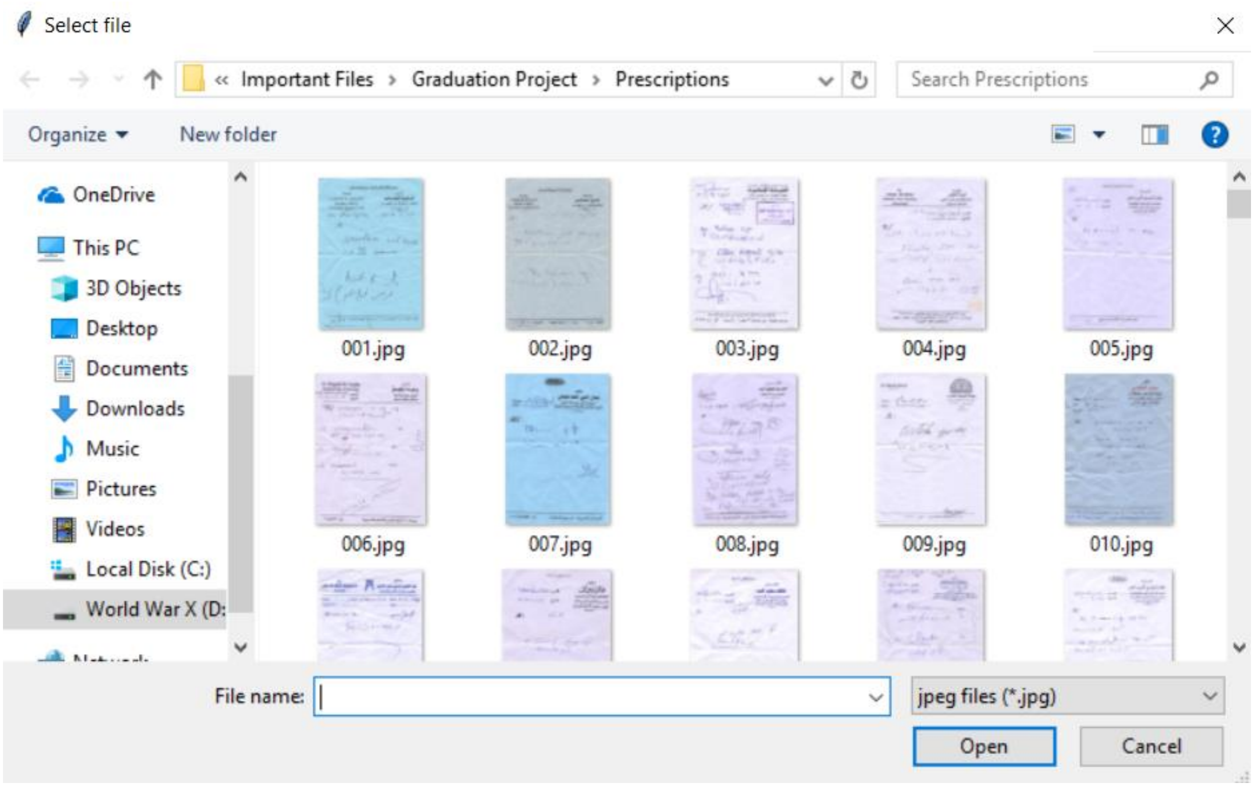
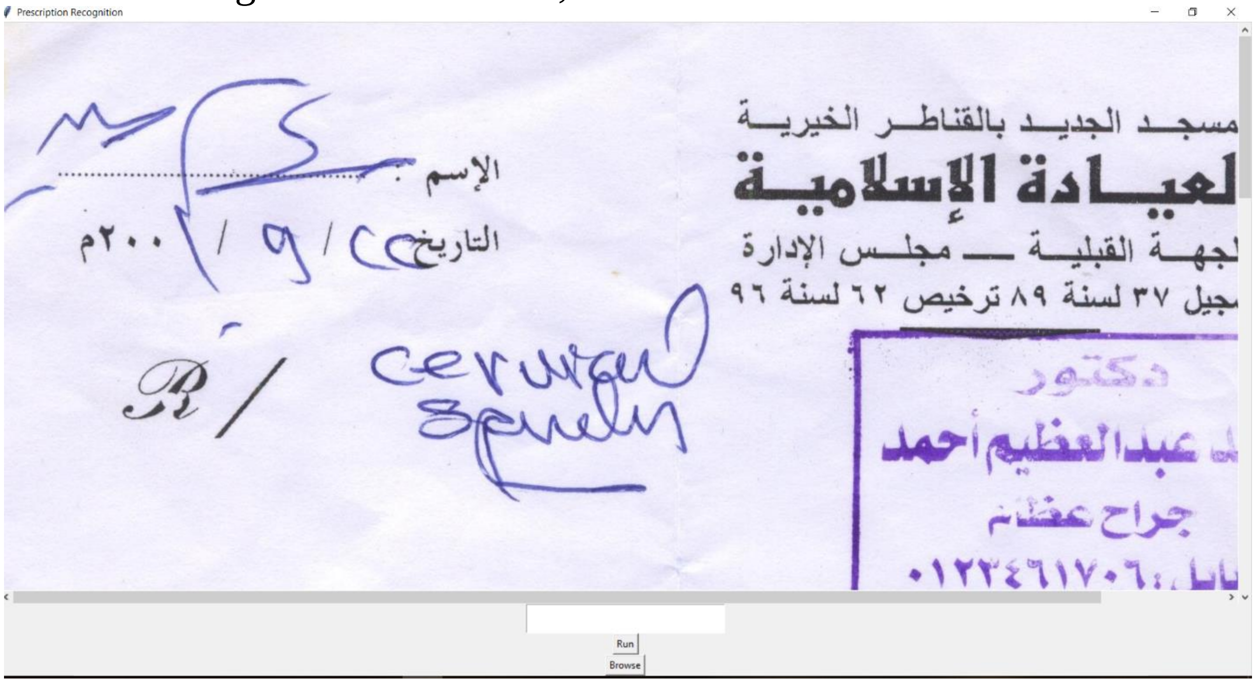


Figure 53: Browse for prescription

After selecting the medical form, this is how it will look like.



مسجد الجديد بالقناطر الخيرية
لعيادة الإسلامية
لجهة القبلية — مجلس الإدارة
جيل ٣٧ لسنة ٨٩ ترخيص ٦٢ لسنة ٩٦

الإسم :

التاريخ CC / 9 / ٢٠٠م

R/

دكتور
عبدالعظيم أحمد
جراح عظام
٠١٢٢٣٤٦١١٧٠٦٠

Figure 54: Selected prescription

Use the scrollbar to move inside the medical form. Find you desired medication name.

You have to specify which medication name you want to know by selecting it. Draw a rectangle over the desired name by giving top-left and bottom-right points. Pressing on 'Q' and 'W' draws these points. Red point represents the top-left point and the green one represents the bottom-right point.

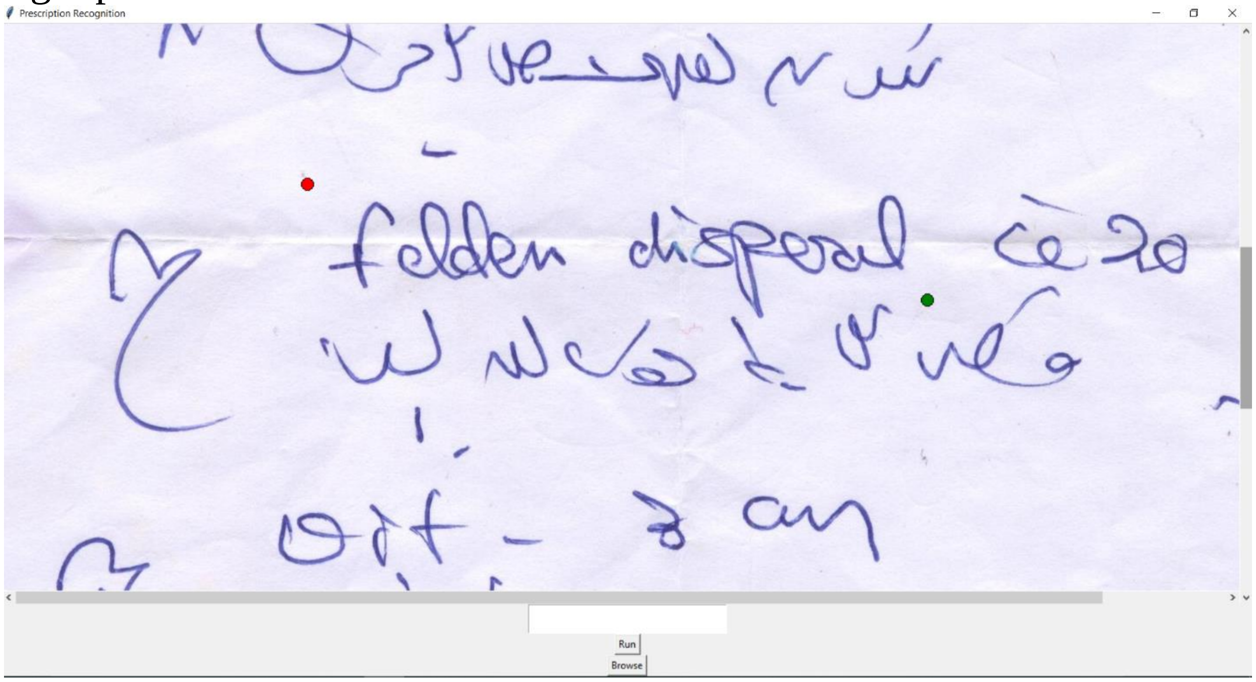


Figure 55: Select region of medication

After specifying these points, press 'F' key to crop the desired medication out of image.

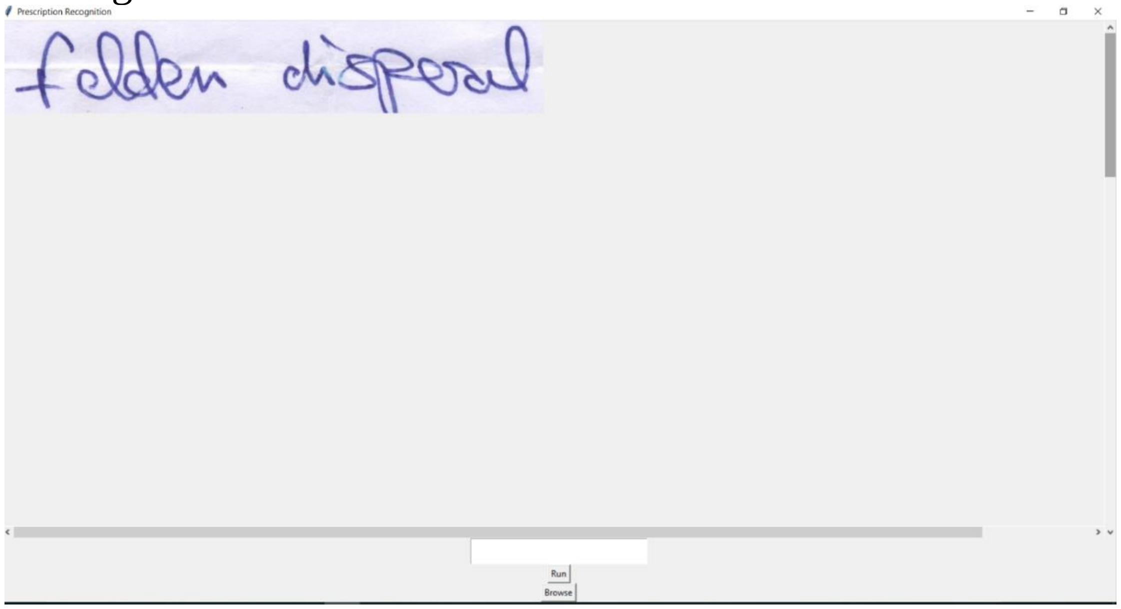
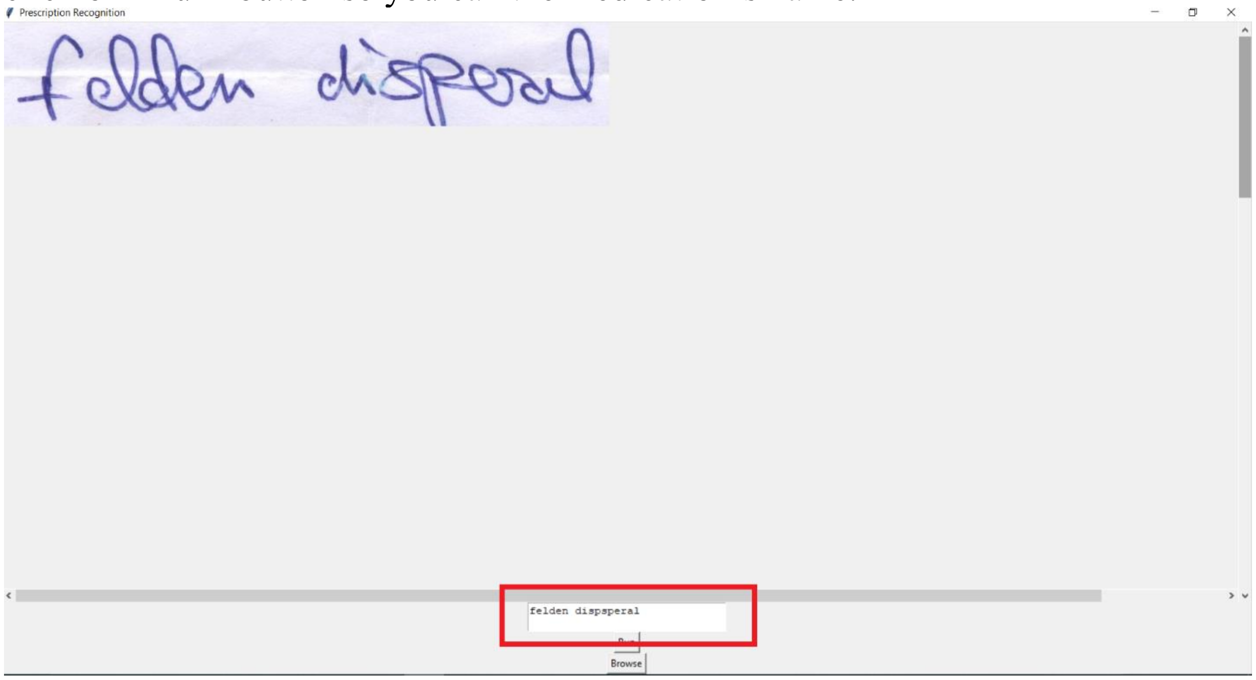


Figure 56: selected medication

You have the option to repeat this operation more than once if you want. Otherwise, click on “Run” button so you can the medication’s name.



```
felden dispsperal
```

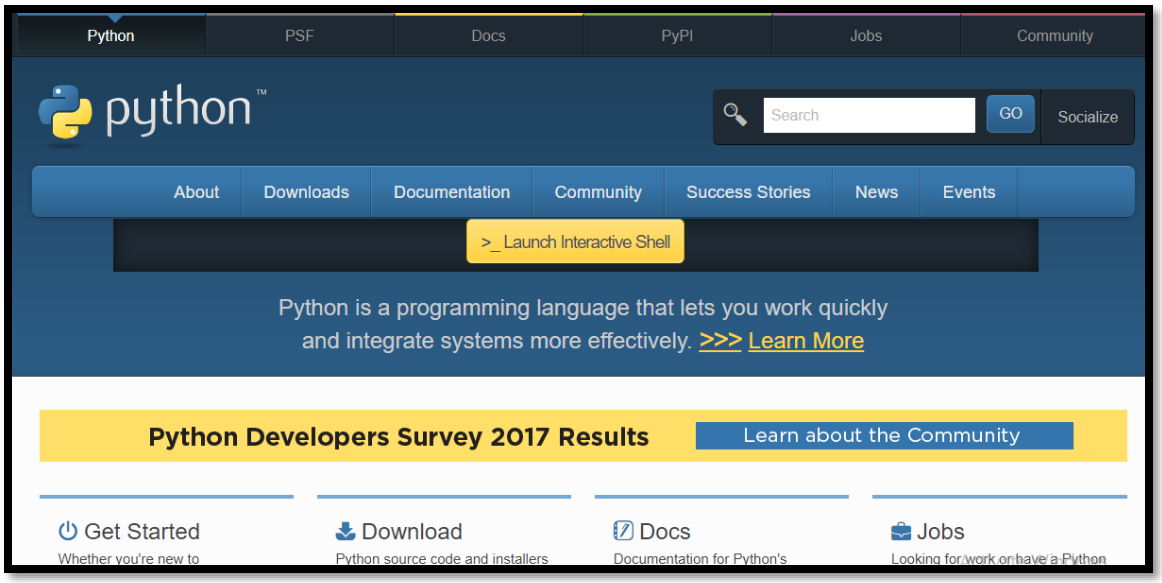
Figure 57: Output of the program

4.2 System Deployment

This chapter include an "Installation Guide" that would describe how to install the program, and all required third party tools that needs to be available for the project to run.

Installation steps:

1. setup the **Python environment**
   From the python site from here https://www.python.org.



Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Whether you're new to

Python source code and installers

Documentation for Python's

Looking for work or have a Python

Figure 58: Python website

2. Setup **Anaconda environment** and you can download from here *https://www.anaconda.com/download*

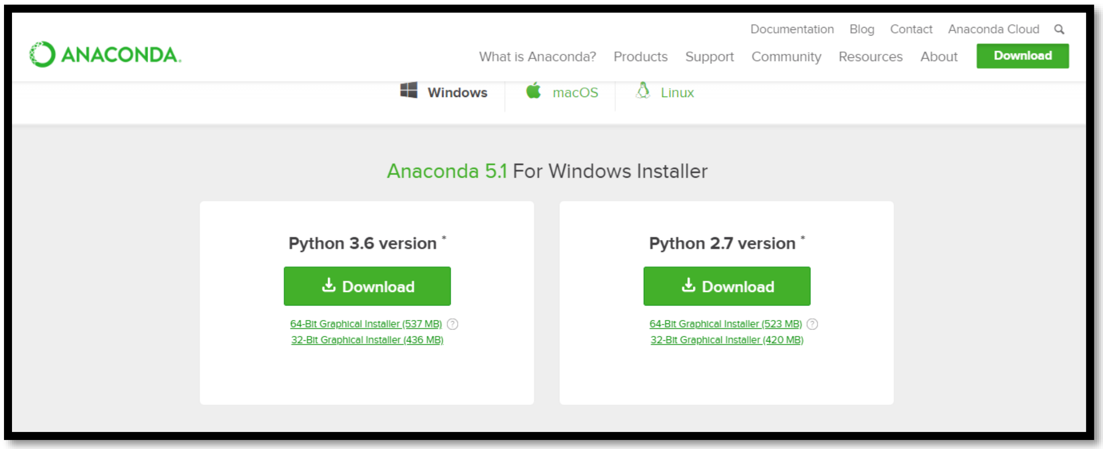

ANACONDA.
Documentation Blog Contact Anaconda Cloud
What is Anaconda? Products Support Community Resources About Download
Windows macOS Linux
Anaconda 5.1 For Windows Installer
Python 3.6 version
Download
64-Bit Graphical Installer (537 MB)
32-Bit Graphical Installer (436 MB)
Python 2.7 version
Download
64-Bit Graphical Installer (523 MB)
32-Bit Graphical Installer (420 MB)

Figure 59: Anaconda website

3. Install the libraries that you will use. You can install **TensorFlow** from here *https://www.tensorflow.org/*

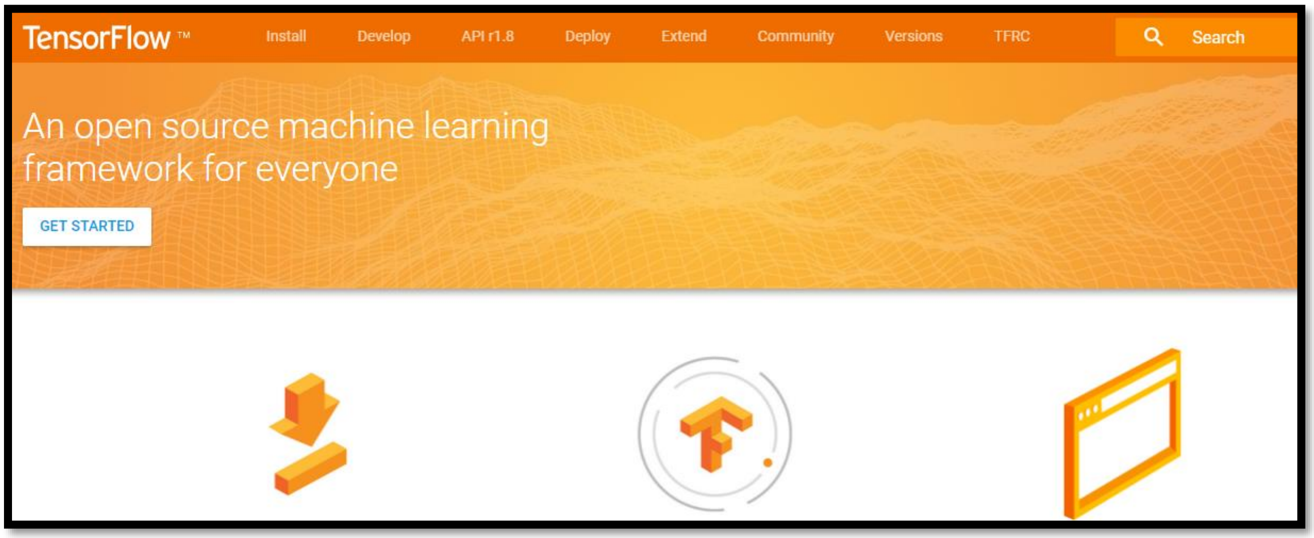

TensorFlow ™
Install Develop API r1.8 Deploy Extend Community Versions TFRC Search
An open source machine learning framework for everyone
GET STARTED

Figure 60: TensorFlow website

Results

- We train our model with 9000 images from IAM dataset and test it with 3000 image we obtain 60% accuracy.

- We train our model with 9000 images from IAM dataset and test it with 3000 image we obtain 65% accuracy after applying edit in data labels to perform CTC function to enhance the result.

- We train our model with 9000 images from IAM dataset and test it with 100 prescriptions we obtain 55%

7 - Conclusion and Future Work

7.1 Conclusion

Many regional languages throughout world have different writing styles which can be recognized with HCR systems using proper algorithm and strategies.

We have learning for recognition of English characters. It has been found that recognition of handwritten character becomes difficult due to presence of odd characters or similarity in shapes for multiple characters.
Scanned image is pre-processed manually by cropping name of medicine from prescription.
Height of Cropped image may be not equal 64 pixels then we have two conditions if image height smaller than 64 pixels we use padding to make image height 64 pixels or if image height greater than 64 pixels we use a factor to resize image to be 64 pixels.

In this work, we presented a state-of-the-art CRNN system for text-line recognition of Prescriptions. we tackled the problem of general, unconstrained text recognition. We presented a simple data and computation efficient neural network architecture that can be trained end-to-end on variable-sized images using variable-sized line level transcriptions.

The experiments demonstrated the simplicity of adopting architecture to any new text recognition task. We also conducted an extensive ablation study on our model that highlighted the importance of each of its sub-modules.

7.2 Future Work

- It is intended to expand the project so it can cover many platforms such like: Android, windows, etc.
- We will improve the performance of the project by using larger prescription dataset.
- It is intended to add more function such that we can use cloud applications to support our project with the nearest pharmacy to the patient where the medicine is available

References

[1] RECENT ACHIEVEMENTS IN OFF-LINE HANDWRITING RECOGNITION SYSTEMS B. VERMA, M. BLUMENSTEIN & S. KULKARNI School of Information Technology Griffith University–Gold Cost Computer
https://pdfs.semanticscholar.org/7a44/a78040bba0dc13dd3d860bb6faad275425c4.pdf

[2] HCR USING NEURAL NETWORK EPARTMENT OF COMPUTER SCIENCE AND ENGINEERING COLLEGE OF ENGINEERING & TECHNOLOGY BHUBANESWAR-751003
A Constituent College of Biju Patnaik University of Technology, Odisha October 2009
https://www.researchgate.net/publication/323547763_Handwritten_Character_Recognition_HCR_USING_NEURAL_NETWORK

[3] Handwritten Character Recognition in English: A Survey Monica Patel, Shital P. Thakkar Department of Electronics and Communication, Dharmsinh Desai University, Nadiad, Gujarat, India Associate Professor, Dept of Electronics and Communication Dharmsinh Desai University, Nadiad, Gujarat, India.
https://www.researchgate.net/publication/315053242_Handwritten_Character_Recognition_in_English_A_Survey

[4] Handwriting Recognition of Historical Documents with few labeled data Edgard Chammas and Chafic Mokbel University of Balamand El-Koura, Lebanon, Laurence Likforman-Sulem Institut Mines Telecom, Telecom ParisTech and Universit´e Paris-Saclay Paris.
https://perso.telecom-paristech.fr/lauli/Articles/published_das2018.pdf

[5] Offline Handwriting Character Recognition (for use of medical purpose) Using Neural Network Sanjay Kumar1, Narendra Sahu 2, Aakash Deep 3, Khushboo Gavel4, Miss Rumi Ghosh5

https://www.ijecs.in/index.php/ijecs/article/download/2691/2488/

[6] Handwritten Character Recognition in English: A Survey
Monica Patel1, Shital P. Thakkar2
https://www.researchgate.net/profile/Shital_Thakkar/publication/315053242_Handwritten_Character_Recognition_in_English_A_Survey/links/5b444f830f7e9bb59b1b28d3/Handwritten-Character-Recognition-in-English-A-Survey.pdf

[7] Handwritten Character Recognition-An Analysis Usha Tiwari, Monika Jain and Shabana Mahfouz
https://drive.google.com/file/d/1hb7QQzOWM7ImkStW8xRuTYhgNfGduNQf/view

[8] Offline Handwritten Character Recognition Techniques using Neural Network: A Review Vijay Sahu, Babita Kubde
https://www.semanticscholar.org/paper/Offline-Handwritten-Character-Recognition-using-%3A-A-Sahu-Kubde/8ccd4add95a7d7364973bfbfdeadb5bc3ef7968d

[9] Handwritten Character Recognition Using Deep-Learning
http://scihub.tw/10.1109/ICICCT.2018.8473291?fbclid=IwAR2Wiat47TXOLXV-_M2p-K__TvoX66f3c8ghtwQd01M3BfjcKBxr2BFlMgo

[10] Handwritten Text Recognition using Deep Learning Batuhan Balci bbalci, Dan Saadati, Dan Shiferaw
http://cs231n.stanford.edu/reports/2017/pdfs/810.pdf

[11] Handwriting Recognition – "Offline" Approach P. Shankar Rao, J. Aditya {Dept of CSE, Andhra University}
https://cs.stanford.edu/people/adityaj/HandwritingRecognition.pdf

[12] Character Recognition for Cursive English Handwriting to Recognize Medicine Name from Doctor's Prescription
Pritam S. Dhande,Reena Kharat

https://www.researchgate.net/publication/327805181_Character_Recognition_for_Cursive_English_Handwriting_to_Recognize_Medicine_Name_from_Doctor's_Prescription

[13] HANDWRITING RECOGNITION METHODS USING ARTIFICIAL NEURAL NETWORKS Wojciech Kacalak
https://www.researchgate.net/publication/275517344_HANDWRITING_RECOGNITION_METHODS_USING_ARTIFICIAL_NEURAL_NETWORKS

[14] Offline continuous handwriting recognition using sequence to sequence neural networks
Jorge Sueiras,Victoria Ruiz,Angel Sanchez,Jose F.Velez
https://www.sciencedirect.com/science/article/pii/S0925231218301371

[15] Handwriting Recognition using Tensorflow ,lon Moşnoi
https://www.sciencedirect.com/science/article/pii/S0925231218301371