VISVESVARAYA TECHNOLOGICALUNIVERSITY

"Jnana Sangama", Belgaum-590018.

A seminar Report On

"Handwritten Text Pattern Recognition"

Submitted in partial fulfillment for the requirements of the VIII Semester degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

For the Academic Year
2019-20
By

Anurag G	**1DB16CS023**
C Sagar Patil	**1DB16CS042**
Deepak S D	**1DB17CS403**
Harsha J K	**1DB16CS056**

Under the Guidance Of

Mrs. Hemalatha M

Asst. Professor,

Dept. of CSE

Department of Computer Science and Engineering

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Mysore Road, Bengaluru - 560074.

ACKNOWLEDGEMENT

The satisfaction and euphoria that successful completion of any project is incomplete without the mention of people who made it possible, whose constant support and encouragement made our effort fruitful.

In this connection, I would like to express my deep sense of gratitude to my beloved institution Don Bosco Institute of Engineering and also, I like to express my sincere gratitude and indebtedness to **Dr. Hemadri Naidu T, Principal, DBIT, Bangalore.**

I would like to express my sincere gratitude to **Prof. B.S. UMASHANKAR** Professor and Head of Dept. of Computer Science and Engineering, for providing a congenial environment to work in and carryout my seminar.

I would like to express the deepest sense of gratitude to thank my Project Guide **"Mrs. Hemalatha M"**, Asst. Professor, Department of Computer Science and Engineering, DBIT, Bangalore for his constant help and support extended towards me during the course of the project.

Finally, I am very much thankful to all the teaching and non-teaching members of the Department of Computer Science and Engineering, my seniors, friends and my parents for their constant encouragement, support and help throughout completion of report.

Anurag G **(1DB16CS023)**
C Sagar Patil **(1DB16CS042)**
Deepak S D **(1DB17CS403)**
Harsha J K **(1DB16CS056)**

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Bengaluru – 560074.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the mini project report entitled **"Handwritten Text Pattern Recognition"** is a Bonafide work carried out by **Anurag G(1DB16CS023), C Sagar Patil(1DB16CS042), Deepak S D(1DB17CS403), Harsha J K(1DB16CS056)** in partial fulfillment of award of Degree of **Bachelor of Engineering in Computer Science and Engineering** of Visvesvaraya Technological University, Belagavi, during the academic year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated. The mini project has been approved as it satisfies the academic requirements associated with the degree mentioned.

Mrs. Hemalatha M	**Prof. B S Umashankar**	**Dr. Hemadri Naidu T**
Asst. Prof. & Guide Dept. of CSE, DBIT	Prof. & Head Dept. of CSE, DBIT	Principal DBIT

External Viva

Name of the Examiners	Signature with date
1. _____	1._____
2. _____	2._____

DON BOSCO INSTITUTE OF TECHNOLOGY

Kumbalagodu, Bangalore – 560074.

DECLARATION

We, **Anurag G, C Sagar Patil, Deepak S D, Harsha J K** students of Eighth semester B.E, Computer Science and Engineering, Don Bosco Institute of Technology, Bengaluru declare that the project entitled "**Handwritten Text Pattern Recognition**" has been carried out and submitted in partial fulfilment of the course requirements for the Eight semester examination of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi during the academic year 2019-20. The matter embodied in this report has not been submitted to any other university or institution for the award of any other degree or diploma.

Place: Bangalore

Date:

Anurag G	**(1DB16CS023)**
C Sagar Patil	**(1DB16CS042)**
Deepak S D	**(1DB17CS403)**
Harsha J K	**(1DB16CS056)**

ABSTRACT

Character recognition from handwritten images is of great interest in the pattern recognition research community for their good application in many areas. To implement the system, it requires two steps, viz., feature extraction followed by character recognition based on any classification algorithm. Convolutional neural network (CNN) is an excellent feature extractor and classifier. It is having multiple applications fields such as robotics, medicine, and security and surveillance. In this article, a CNN is implemented for the NIST dataset with appropriate parameters for training and testing the system. The system provides accuracy up to 94%, which is better with respect to others. It also takes very low amount of time for training the system.

Contents

CHAPTER-1

INTRODUCTION

Hand-written text pattern recognition is considered as one of the methods to communicate between man and machine. In handwriting recognition (HWR) the device interprets the user's handwritten characters or words into a format that the computer understands. Many methods or models are provided to recognize text with various accuracy. Performance of the system is determined depending on attributes such as the size of the text, the writing style, and the rate of recognition. Web based application is used in our project that uses the text written image to display the results identified.

We used the Convolutional Neural Network Model in our system. We used the publicly available NIST Dataset which contains samples of handwritten characters from thousands of writers. The neural network model which we have used is Convolutional Neural Network. CNN's are State-of-Art neural networks which have huge applications is field of Computer Vision. The neural network model was trained using TensorFlow which is an open source library used for Machine learning applications. OpenCV was used to perform various image processing operations like segmentation, thresholding and Morphological Operations. OpenCV is an open source library which is used for Image processing.

Handwritten character recognition is an area of pattern recognition which defines an ability of a machine to analyze patterns and identify the character. Pattern recognition is the science of making inferences from perceptual data based on either a priori knowledge or on statistical information . The subject of pattern recognition spans a number of scientific disciplines uniting them in the search for the solution to the common problem of recognizing the members of a class in a set containing elements from many patterns in classes. A pattern class is a category determined by some given common attributes. A pattern is the description of any member of a category representing a pattern class.

The basic function of pattern recognition system is to detect and extract common features from the patterns describing the objects that belong to the same pattern classes and to recognize the pattern and classify it as a member of the pattern class under consideration. Since last few decades and advancement in technology, computers interact more effectively with humans and with the natural world e.g. speech recognition, handwritten recognition, gesture recognition etc.

Handwritten character recognition is an area of pattern recognition which defines an ability of a machine to analyze patterns and identify the character. Pattern recognition is the science of making inferences from perceptual data based on either a priori knowledge or on statistical information. The subject of pattern recognition spans a number of scientific disciplines uniting them in the search for the solution to the common problem of recognizing the members of a class in a set containing elements from many patterns in classes. A pattern class is a category determined by some given common attributes. A pattern is the description of any member of a category representing a pattern class.

Since last few decades and advancement in technology, computers interact more effectively with humans and with the natural world e.g. speech recognition, handwritten recognition, gesture recognition etc. However, Humans are outperforming far better than machines in recognizing patterns. Some of the tasks are generally easy for humans, such as identifying the human voice based on frequency and pitch, recognizing and differentiating aroma of flower and a food, identifying characters etc. These kinds of perceptual problems are difficult for the computer because of voluminous data with composite and hidden information each pattern usually contains.

In a curiosity to understand and uncover secret of how humans can recognize patterns many efforts are being attempted in this area to mimic human behavior. Handwritten character recognition is one such area where "Handwriting" is used to preserve information so as to retrieve it at a later stage and as a facilitating mode of communication using some artificial graphical symbols on a surface. Since many decades paper have been a popular surface to write. Due to the advancement of technology this surface is gradually being taken by a pressure sensitive touch screen. On the basis of surface handwritten character recognition technology is divided into two categories i.e. Online and Offline.

CHAPTER-2

Aims and Objectives

Aims:

- The main aim of this project is to build a hand-written text pattern recognition system.
- The system accepts an image, recognizes the text written in the image and converts it into a digital formatted text and gives it as an output.

Objectives:

- Implementing Convolutional neural network (CNN) model to have a system that can recognize the hand-written text patterns with a high recognition rate.
- Reduce the number of classification errors in transcribing these handwritten digits and reduce the time taken to complete this task.
- Handwritten text is converted into digital format to increase the readability of any written text.

Chapter 3

LITERATURE SURVEY

An early notable attempt in the area of character recognition research is by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as analysis-by-synthesis method suggested by Eden in 1968.

Gaurav and Bhatia P. K's "Analytical Review of Preprocessing Techniques for Offline Handwritten Character Recognition", this paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities. In this, different preprocessing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.

Salvador España-Boquera, Maria J. C. B., Jorge G. M. and Francisco Z. M., "Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models", in this paper hybrid Hidden Markov Model (HMM) model is proposed for recognizing unconstrained offline handwritten texts yielding a 88.8% of accuracy for lexicon size 40,000. In this, the structural part of the optical model has been modelled with Markov chains, and a Multilayer Perceptron is used to estimate the emission probabilities. In this paper, different techniques are applied to remove slope and slant from handwritten text and to normalize the size of text images with supervised learning methods. The key features of this recognition system were to develop a system having high accuracy in preprocessing and recognition, which are both based on ANNs.

Zafar M.F. et.al [1], proposes a online handwritten character recognition algorithm, suitable online handwritten character recognition algorithm, suitable for the home use computer. A character, written on a digitizing tablet, is expressed as a directional angle sequence. In order to recognize a quickly written multistroke character, the character pattern is converted into a single interconnected stroke pattern. The recognition is carried out using dynamic programming based pattern matching technique. Chiang, C.C. et.al [3], proposes a handwritten character recognition system implemented by a stochastic neural net (SNN) is presented. The learning process in this neural model incorporates the stochastic information extracted from the trained characters. The merit of SNN lies in the fast learning speed obtained through an online learning algorithm, in contrast to offline learning algorithms. This SNN has been applied to design an experimental adaptive character recognition system. This system can recognize handwritten/printed English and Chinese characters. According to preliminary experimental results, the recognition rates are about 90~94% and 80~85% for English and Chinese characters, respectively. The system is capable of learning while recognizing new patterns.

Liangwen Zhang et.al [4], this paper proposes online handwritten English characters segmentation method based on rules. Firstly, all the local minima of the character coordinate sequences are treated as potential segmentation points; Secondly, we calculate 5 features of every potential segmentation point; finally, the potential segmentation points are modified according to the rules to get the final segmentation points. The modification contains shift, deletion and maintenance. Thus, the coordinate sequences of characters are segmented to sub-sequences of characters by the final segmentation points. The experiment conducted on UNIPEN database shows that our method is feasible and effective.

CHAPTER- 4

EXISTING SYSTEM

- The present system doesn't use modern technology & tools. and has low accuracy which makes the model not usable in real time.

- In existing system preprocessing of data is time consuming.

- The existing system has low accuracy which makes the model not usable in real time.

CHAPTER- 5

PROPOSED SYSTEM

The purpose of this study is the development of system that takes handwritten English characters as input, process the input, extract the optimal features, train the neural network using either Resilient Back-propagation or Scaled conjugate gradient, recognize the class of input text, and finally generate the computerized form of input text. The complete system is divided into two major sections: Training of ANN with image database and testing of ANN with test images. Figure 1 is showing the block diagram of training part of ANN and Figure 2 is showing the block diagram of testing part of ANN.

The training part of proposed work involves: creation of dataset, preprocessing of that dataset, feature extraction from pre-processed dataset, generation of a feature vector and test vector, training of ANN and saving of trained ANN for testing purpose. The testing part involves some extra pre-processing steps as here we need to figure out the number of characters in the input image but it does not includes any training of ANN. On the contrary, it uses trained ANN directly after the feature vector generation. The segmentation is an important step of test procedure as it helps to figure out number of characters.The proposed system identifies the text by the image which is selected and uploaded from the local storage.

The system recognizes the image and gives the output in a digitalized text format which can be further converted to human speech.The time consumed for training the model has been reduced and accuracy has been increased.Progressive Web App has been implemented for user interaction which reduces the scope of mobile first approach.

SYSTEM REQUIREMENTS

I. Technical Requirement

i. Hardware Requirements

- A standalone computer
- Quad core Intel Core i7 Skylake or higher (Dual core is not the best for this kind of work, but manageable)
- 4GB of RAM
- Premium graphics cards, so things with GTX 980 or 980Ms would be the best for a laptop, and 1080s or 1070s would be the best for the desktop setup.

ii. Software requirements

- MATLAB Version 8.5.0(R2015a) or higher
- Windows 8 or higher

II. Functional Requirements

- System functional requirement describes activities and services that must provide.
- User must be able to enter UR: or upload necessary image of the product.
- Only authorized user must be able to make changes to the system.
- The information must be entered and managed properly.

III. Non-Functional Requirements

Non-functional Requirements are characteristics or attributes of the system that can judge its operation. The following points clarify them:

a. **Accuracy and Precision:** the system should perform its process in accuracy and precision to avoid problems.
b. **Flexibility:** the system should be easy to modify, any wrong should be correct.
c. **Security:** the system should be secure and saving user's privacy.
d. **Usability:** the system should be easy to deal with and simple to understand.

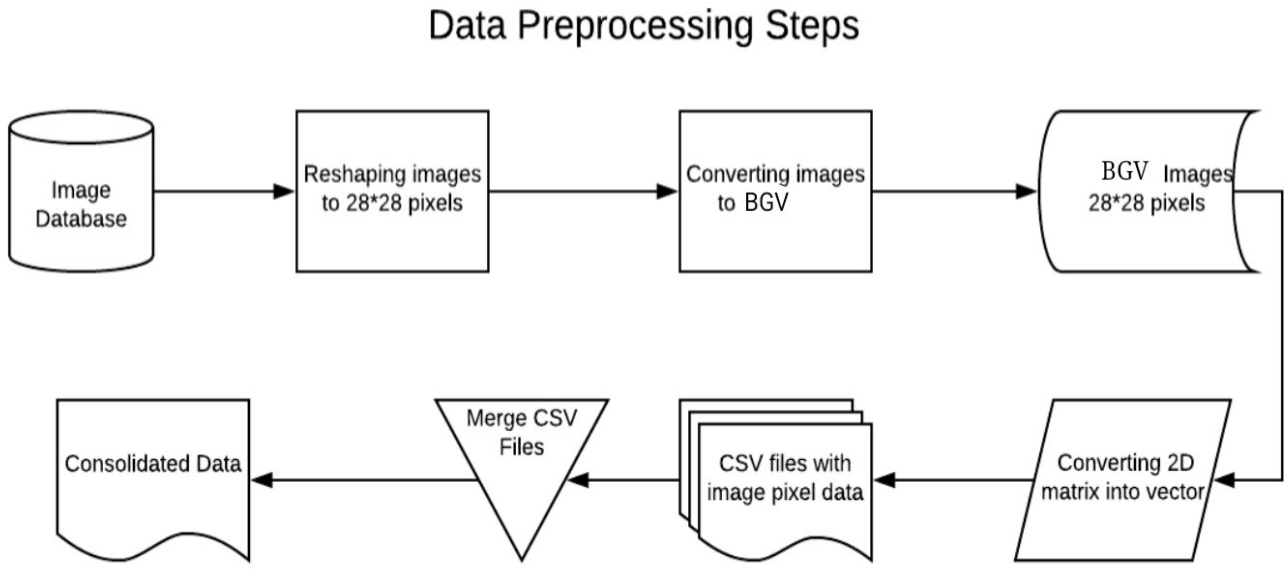
d. **Maintainability:** the maintenance group should be able to cope up with any problem when occurs suddenly.

f. **Speed and Responsiveness**: Execution of operations should be fast.

- **Non–Functional Requirements are as follow:**
- The GUI of the system will be user friendly.
- The data that will be showed to the users will be made sure that it is correct and is available for the time being.
- The system will be flexible to changes.
- The system will be extensible for changes and to the latest technologies.
- Efficiency and effectiveness of the system will be made sure.

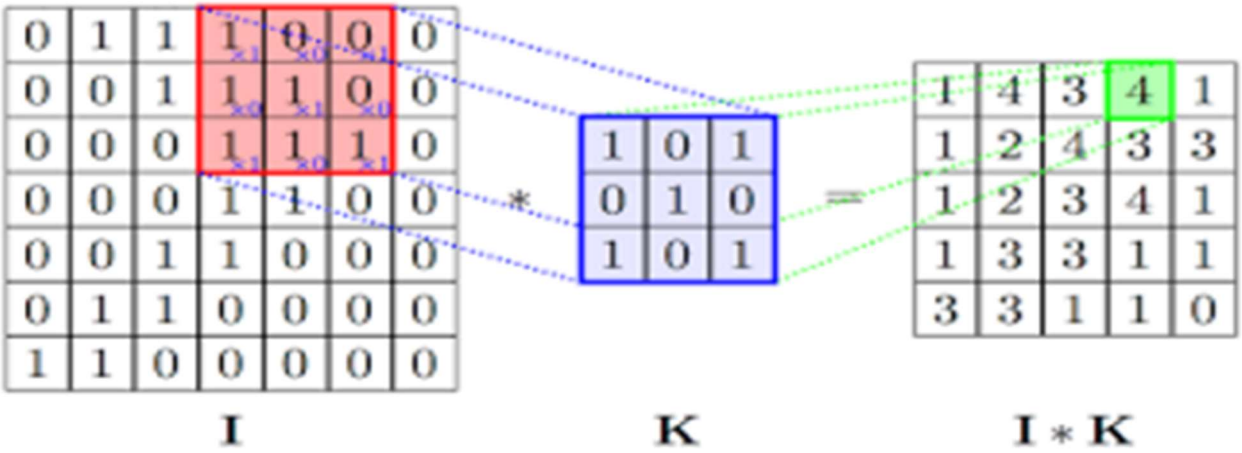
CHAPTER-6

SYSTEM DESIGN

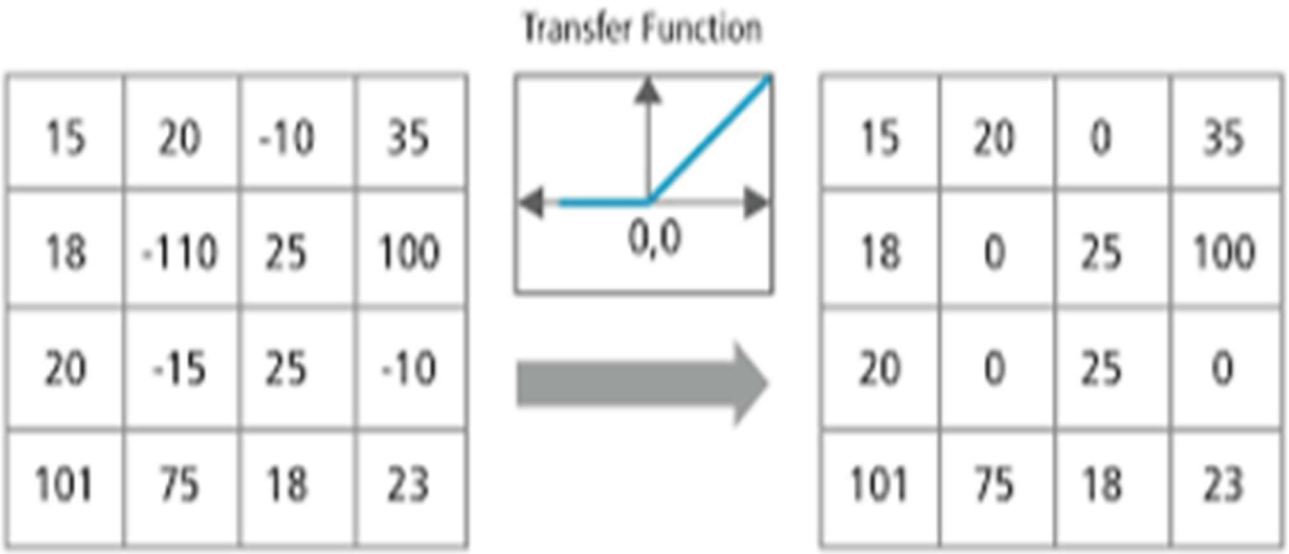


A series of operations are performed on the input image (In testing as well as training stage) during the pre-processing. It helps in enhancing the image rendering and makes the image suitable for segmentation. The main objective of pre-processing is to remove the background noise, enhance the region of interest in image and make a clear difference between foreground and background. In order to achieve these goals: noise filtering, conversion to binary and smoothing operations are performed on the input image. Figure 3 is showing an example of image normalization. The pre-processing also involves a compressed representation of the input image. The edge detection is also performed in order to select the region of interest. The conversion to binary insures a very good difference between foreground and background. Dilation of edges is also performed in the pre-processing step itself.

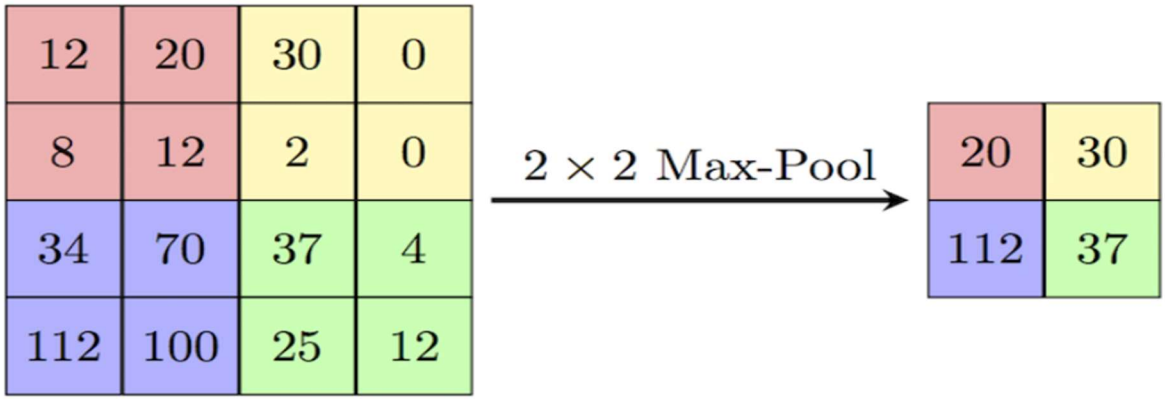
STEP-1: Convolution step



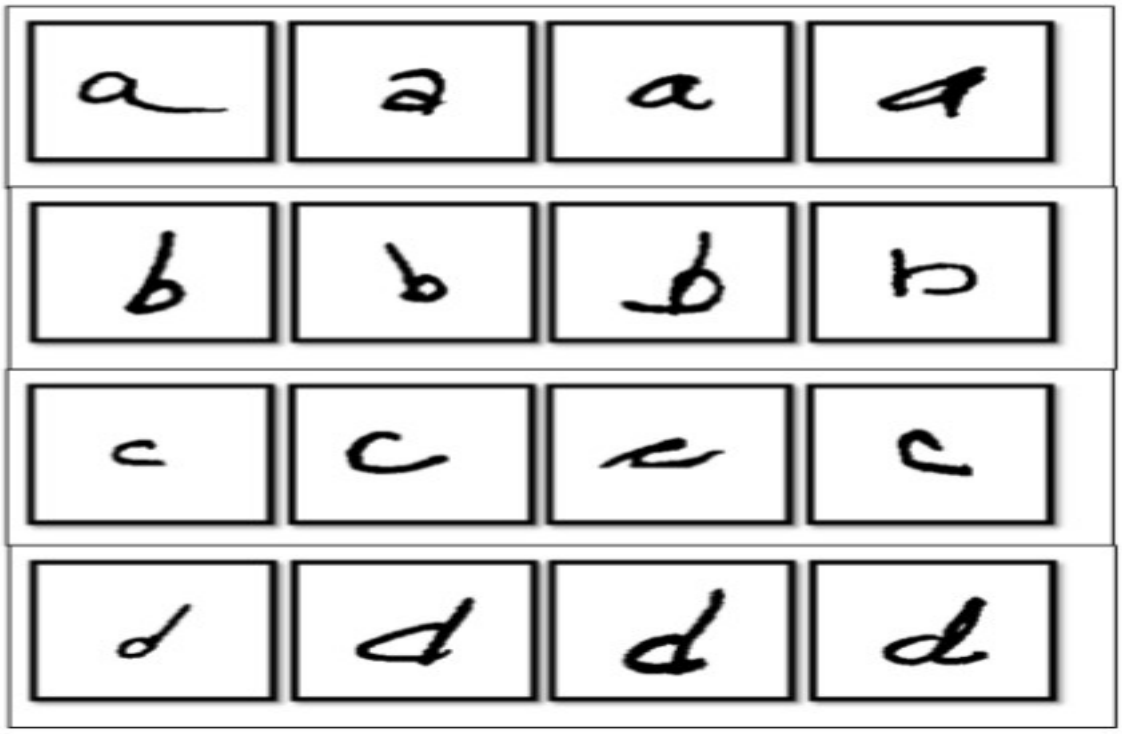
STEP-2: ReLu(Rectifier Linear Units)



STEP-3: Max pooling



STEP-4: SOME OF THE IMAGES USED FOR TRAINING NEURAL NETWORK



Following steps are involved in processing of images:

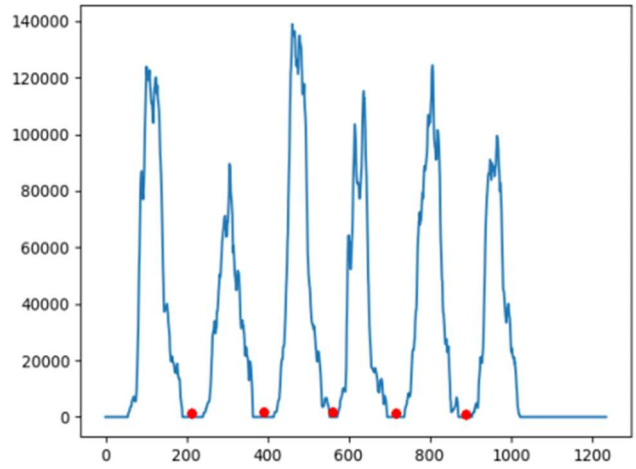
1) **Pre-processing**: This is the first step performed in image processing. In this step the noise from the image is removed by using median filtering. Median filtering is one of the most widely used noise reduction technique. This is because in median filtering the edges in image are preserved while the noise is still removed.

2) **Conversion to Gray-Scale**: After the pre-processing step, the image is converted into grayscale. Conversion into grayscale is necessary because different writers use pens of different colours with varying intensities. Also working on grayscale images reduces the overall complexity of the system.

3) **Thresholding**: When an image is converted into grayscale, the handwritten text is darker as compared to its background. With the help of thresholding we can separate the darker regions of the image from the lighter regions. Thus because of thresholding we can separate the handwritten text from its background.

4) **Image Segmentation**: A user can write text in the form of lines. Thus, the thresholder image is first segmented into individual lines. Then each individual line is segmented into individual words. Finally, each word is segmented into individual characters. Segmentation of image into lines is carried out using Horizontal projection method. First the thresholder image is inverted so that background becomes foreground and vice-versa. Now the image is scanned from top to bottom. While scanning, the sum of pixels in each row of image is calculated.

The sum of pixels will be zero if all the pixels in one particular row are black. The sum will be non-zero if some white pixels are present in a row. After this a horizontal histogram is plotted in which the X-axis represents the Y-coordinate of image(Starting from Top to Bottom) and the Y-axis represents the sum of pixels in the row corresponding to the Y-coordinate. The horizontal histogram is plotted using MatPlotLib and is as shown in Fig.

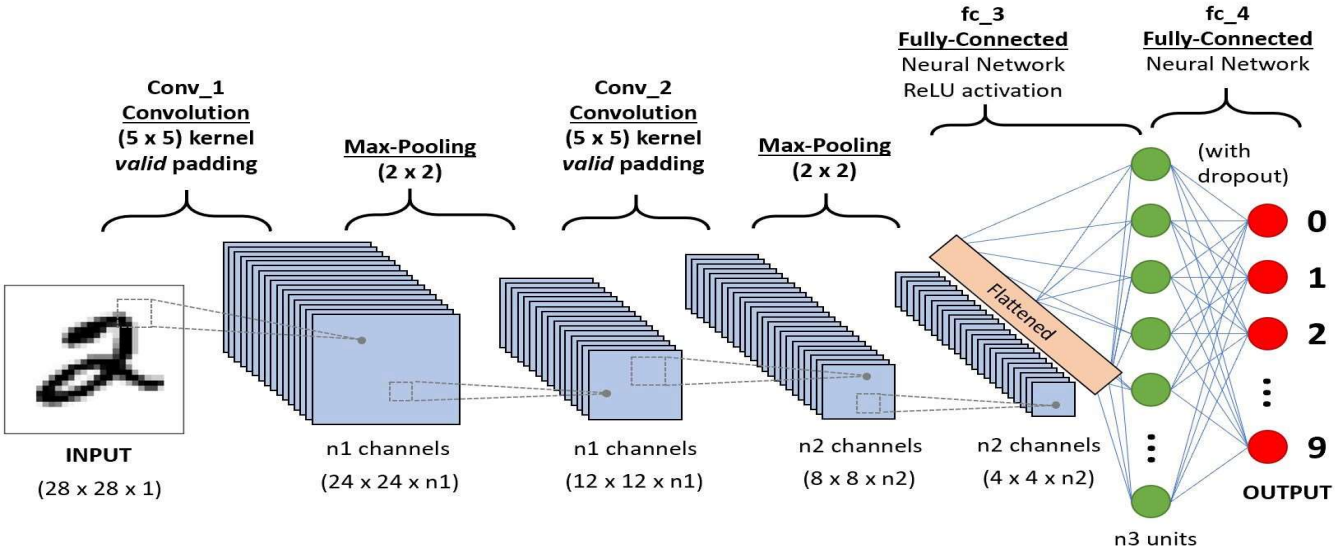


The points marked in red are the points corrosponding to the rows where sum of pixels are zero. After identifying all such rows we can easily segment handwritten text into lines at these points. Now once the image is segmented into lines , each line must be further segmented into individual words. Segmentation of a line into words can be performed using the Vertical projection method. For segmenting line into words , we can make use of the fact that the spacing between two words is larger than the spacing between two characters. To segment a single line into individual words , the image is scanned from left to right and sum of pixels in each column is calculated. A vertical histogram is plotted in which the X-axis represents the Xcoordinates of image and Y-axis represents the sum of pixels in each column. The vertical histogram is as shown below: As we can see the points which are marked as red in Fig.5(a) are the points corrosponding to the columns where sum of pixels is zero. The region where the sum of pixels is zero is wider when it is a region seperating two words as compared to the region which is seperating two characters. After segmenting a line into words , each word can be seperated into individual character using similar technique as explained earlier. Now these individual characters are given to the pre-trained neural network model and predictions are obtained.

CHAPTER-7

IMPLEMENTATION

In this section we discuss how our system has been implemented. Let us fifirst discuss the Android application that we developed for our system. Using this application the user clicks a photo of the handwritten document to be digitalized using the camera of the android phone. Shown next are the screen-shots of the developed application.



An Android application was developed using which the user can click a photo of handwritten text using their camera. The clicked picture is sent to our server for processing at the backend. A neural network runs on this server which can recognize the handwritten text from image. The recognized text is sent back as a response to the android application which is displayed to the user as shown in Fig. Now let us discuss how the backend of our system works. The backend of our system performs two important things. The first thing is hosting the pre-trained neural network model to serve predictions. The second thing is performing image processing operations on the image of handwritten text which is to be recognized. At the backend we have Neural network model trained using Tensorflflow and a python script which is equipped with OpenCV library. We have used the Convolutional Neural network model.

Convolutional neural network(CNN) is the current state-ofart neural network which has wide applications in feilds like Image and Video RecognitioN, Natural Language Processing, Recommender systems. CNN's are biologically inspired neuralnetworks. CNN's are very good at image recognition. In caseof CNN the input is a multi-channeled image(Often an imagehaving Red,Green and Blue channels). A CNN comprises of astack of Convolutional layer and a Max-pooling layer followedby a fully connected layer. The convolutional layer is the most important layer of network. It performs the convolution operation. The pooling layer comes after the convolutional layer. This layer is needed because in case of larger images, the number of trainable parameters can be very large. This increases the time taken to train a neural network and is not practical. The pooling layer is used to reduce the size of image. We used the NIST database which contains thousands of images of handwritten characters. Some of them are shown below. However these images were originally of size 128x128 pixels. The images in the training set were cropped to a size of 28x28. Reducing the size of images decreases the overall time taken to train the neural network model. After the training the Neural network model, an accuracy of upto 94% was obtained.

Implementation tools used:

- Flask Server on PORT 5000
- Jinja2 Template engine
- Keras is used to load Machine Learning Model into the server.

CSS Frameworks

A **CSS framework** is a library allowing for easier, more standards-compliant web design using the Cascading Style Sheets language. Most of these frameworks contain at least a grid. More functional frameworks also come with more features and additional JavaScript based functions, but are mostly design oriented and focused around interactive UI patterns. This detail differentiates CSS frameworks from other JavaScript frameworks.

- reset style sheet
- grid especially for responsive web design
- web typography
- set of icons in sprites or icon fonts
- styling for tooltips, buttons, elements of forms
- parts of graphical user interfaces like accordion, tabs, slideshow or modal windows (Lightbox)
- equalizer to create equal height content
- often used CSS helper classes (*left*, *hide*)
-

Representational State Transfer

Representational state transfer (**REST**) is a software architectural style that defines a set of constraints to be used for creating Web services. Web services that conform to the REST architectural style, called RESTful Web services, provide interoperability between computer systems on the Internet. RESTful Web services allow the requesting systems to access and manipulate textual representations of Web resources by using a uniform and predefined set of stateless operations. Other kinds of Web services, such as SOAP Web services, expose their own arbitrary sets of operations.

Architectural Properties

The constraints of the REST architectural style affect the following architectural properties:

- performance in component interactions, which can be the dominant factor in user- perceived performance and network efficiency;
- scalability allowing the support of large numbers of components and interactions among components. Roy Fielding describes REST's effect on scalability as follows:

  REST's client-server separation of concerns simplifies component implementation, reduces the complexity of connector semantics, improves the effectiveness of performance tuning, and increases the scalability of pure server components. Layered system constraints allow intermediaries—proxies, gateways, and firewalls—to be introduced at various points in the communication without changing the interfaces between components, thus allowing them to assist in communication translation or improve performance via large-scale, shared caching. REST enables intermediate processing by constraining messages to be self-descriptive: interaction is stateless between requests, standard methods and media types are used to indicate semantics and exchange information, and responses explicitly indicate cache ability.

- simplicity of a uniform interface
- modifiability of components to meet changing needs (even while the application is running)
- visibility of communication between components by service agents

- portability of components by moving program code with the data
- reliability in the resistance to failure at the system level in the presence of failures within components, connectors, or data.

Fetch API and Axios

A REST API is an API that follows what is structured in accordance with the REST Structure for APIs. REST stands for "Representational State Transfer". It consists of various rules that developers follow when creating APIs.

The benefits of REST APIs

1. Very easy to learn and understand.
2. It provides developers with the ability to organize complicated applications into simple resources.
3. It easy for external clients to build on your REST API without any complications.
4. It is very easy to scale.
5. A REST API is not language or platform-specific, but can be consumed with any language or run on any platform.

Consuming APIs Using the Fetch API

The fetch() API is an inbuilt JavaScript method for getting resources from a server or an API endpoint. It's similar to XMLHttpRequest, but the fetch API provides a more powerful and flexible feature set.It defines concepts such as CORS and the HTTP Origin header semantics, supplanting their separate definitions elsewhere.

The fetch() API method always takes in a compulsory argument, which is the path or URL to the resource you want to fetch. It returns a promise that points to the response from the request, whether the request is successful or not. You can also optionally pass in an init options object as the second argument.Once a response has been fetched, there are several inbuilt methods available to define what the body content is and how it should be handled.

Consuming APIs With Axios

Axios is an easy to use promise-based HTTP client for the browser and node.js. Since Axios is promise-based, we can take advantage of async and await for more readable and asynchronous code. With Axios, we get the ability to intercept and cancel request, it also has a built-in feature that provides client-side protection against cross-site request forgery.

Features of Axios

- Request and response interception
- Streamlined error handling
- Protection against **XSRF**
- Support for upload progress
- Response timeout
- The ability to cancel requests
- Support for older browsers
- Automatic JSON data transformation

Axios also provides a set of shorthand method for performing different HTTP requests. The Methods are as follows:

- axios.request(config)
- axios.get(url[, config])
- axios.delete(url[, config])
- axios.head(url[, config])
- axios.options(url[, config])
- axios.post(url[, data[, config]])
- axios.put(url[, data[, config]])
- axios.patch(url[, data[, config]])

Model Architecture- Xception

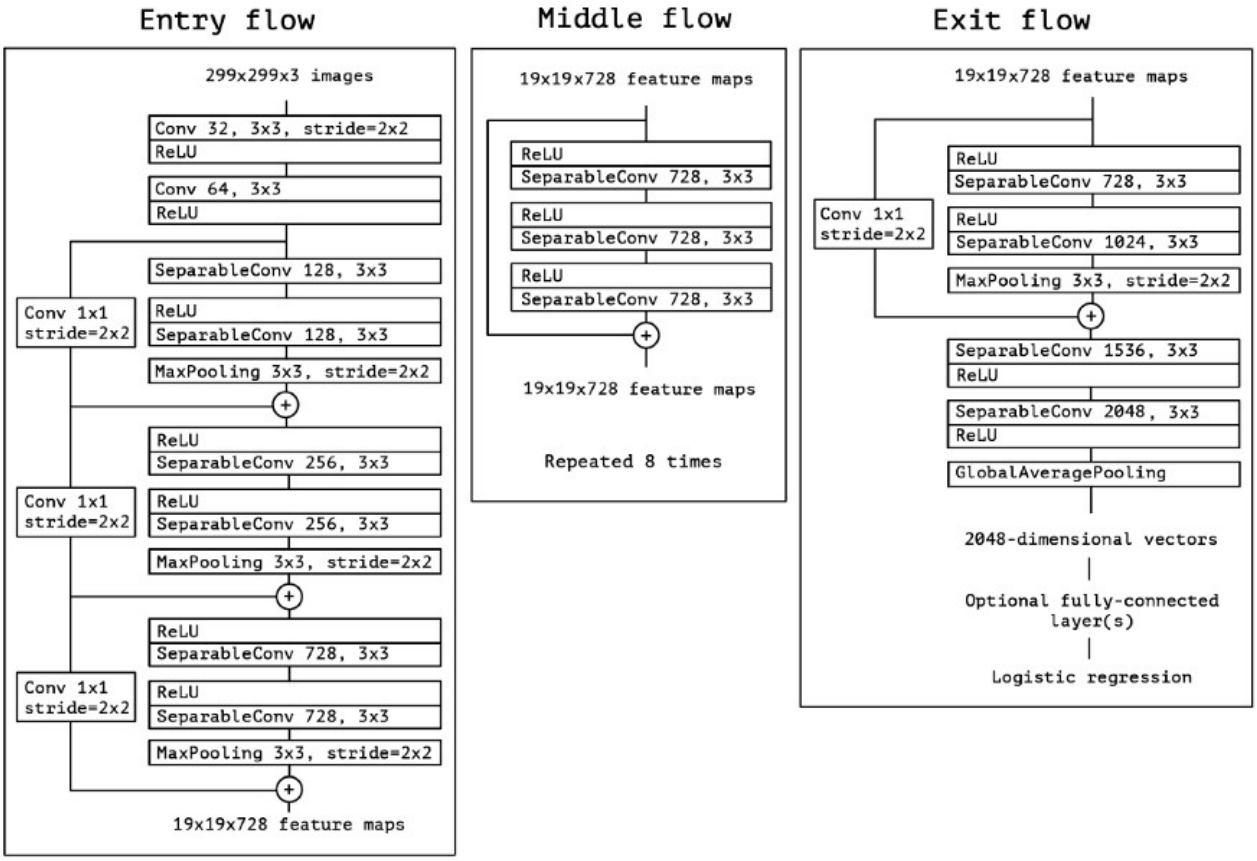


Fig. : Overall Architecture of Xception (Entry Flow > Middle Flow > Exit Flow)

As in the figure above, SeparableConv is the modified depthwise separable convolution. We can see that SeparableConvs are treated as Inception Modules and placed throughout the whole deep learning architecture.And there are residual (or shortcut/skip) connections, originally proposed by ResNet, placed for all flows.

Features

Components

Vue components extend basic HTML elements to encapsulate reusable code. At a high level, components are custom elements to which the Vue's compiler attaches behavior. In Vue, a component is essentially a Vue instance with pre-defined options.

Templates

Vue uses an HTML-based template syntax that allows binding the rendered DOM to the underlying Vue instance's data. All Vue templates are valid HTML that can be parsed by specification-compliant browsers and HTML parsers. Vue compiles the templates into virtual DOM render functions. A virtual Document Object Model (or "DOM") allows Vue to render components in its memory before updating the browser. Combined with the reactivity system, Vue is able to calculate the minimal number of components to re-render and apply the minimal amount of DOM manipulations when the app state changes.

Vue users can use template syntax or choose to directly write render functions using JSX. Render functions allow application to be built from software components.

Reactivity

Vue features a reactivity system that uses plain JavaScript objects and optimized re-rendering. Each component keeps track of its reactive dependencies during its render, so the system knows precisely when to re-render, and which components to re-render.

Transitions

Vue provides a variety of ways to apply transition effects when items are inserted, updated, or removed from the DOM. This includes tools to:

- Automatically apply classes for CSS transitions and animations
- Integrate third-party CSS animation libraries, such as Animate.css
- Use JavaScript to directly manipulate the DOM during transition hooks
- Integrate third-party JavaScript animation libraries, such as Velocity.js

When an element wrapped in a transition component is inserted or removed, this is what happens:

1. Vue will automatically sniff whether the target element has CSS transitions or animations applied. If it does, CSS transition classes will be added/removed at appropriate timings.
2. If the transition component provided JavaScript hooks, these hooks will be called at appropriate timings.
3. If no CSS transitions/animations are detected and no JavaScript hooks are provided, the DOM operations for insertion and/or removal will be executed immediately on next frame.

Routing

A traditional disadvantage of single-page applications (SPAs) is the inability to share links to the exact "sub" page within a specific web page. Because SPAs serve their users only one URL- based response from the server (it typically serves index.html or index.vue), bookmarking certain screens or sharing links to specific sections is normally difficult if not impossible. To solve this problem, many client-side routers delimit their dynamic URLs with a "hashbang" (#!),

e.g. *page.com/#!/*. However, with HTML5 most modern browsers support routing without hash bangs.

Vue provides an interface to change what is displayed on the page based on the current URL path -- regardless of how it was changed (whether by emailed link, refresh, or in-page links). Additionally, using a front-end router allows for the intentional transition of the browser path when certain browser events (i.e. clicks) occur on buttons or links. Vue itself doesn't come with front-end hashed routing. But the open source "vue-router" package provides an API to update the application's URL, supports the back button (navigating history), and email password resets or email verification links with authentication URL parameters. It supports mapping nested routes to nested components and offers fine-grained transition control. With Vue, developers are already composing applications with small building blocks building larger components. With vue-router added to the mix, components must merely be mapped to the routes they belong to, and parent/root routes must indicate where children should render.

Ecosystem

The core library comes with tools and libraries both developed by the core team and contributors.

Official Tooling

- **Devtools** - Browser devtools extension for debugging Vue.js applications
- **Vue CLI** - Standard Tooling for rapid Vue.js development
- **Vue Loader** - a webpack loader that allows the writing of Vue components in a format called Single-File Components (SFCs)

Official Libraries

- **Vue Router** - The official router for Vue.js
- **Vuex** - Flux-inspired Centralized State Management for Vue.js
- **Vue Server Renderer** - Server-Side Rendering for Vue.js

Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

Applications that use the Flask framework include Pinterest and LinkedIn.

Components

The microframework Flask is based on the Pocoo projects Werkzeug and Jinja2.

Werkzeug

Werkzeug is a utility library for the Python programming language, in other words a toolkit for Web Server Gateway Interface (WSGI) applications, and is licensed under a BSD License. Werkzeug can realize software objects for request, response, and utility functions. It can be used to build a custom software framework on top of it and supports Python 2.7 and 3.5 and later.

Jinja

Jinja, also by Ronacher, is a template engine for the Python programming language and is licensed under a BSD License. Similar to the Django web framework, it handles templates in a sandbox.

Features

- Development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja templating
- Support for secure cookies (client-side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

Flask RESTful API

Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries. Flask-RESTful encourages best practices with minimal setup.

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache License 2.0 on November 9, 2015.

TensorFlow is Google Brain's second generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.

Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.

In December 2017, developers from Google, Cisco, RedHat, CoreOS, and CA iCloud introduced Kubeflow at a conference. Kubeflow allows operation and deployment of TensorFlow on Kubernetes.

In March 2018, Google announced TensorFlow.js version 1.0 for machine learning in JavaScript In Jan 2019, Google announced TensorFlow 2.0. It became officially available in Sep 2019.

In May 2019, Google announced TensorFlow Graphics for deep learning in computer graphics.

Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model.

In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library.Chollet explained that Keras was conceived to be an interface rather than a standalone machine learning framework. It offers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational backend used. Microsoftadded a CNTK backend to Keras as well, available as of CNTK v2.0.

Features

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also allows use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU) principally in conjunction with CUDA.

TESTING

Testing Plan

A test plan is a great reminder and to-do list. It’s a way to keep track of everything you need to do to ensure you have the right participants at the right time in the right location with the right setup and the right set of tasks to perform. Having a test plan is also important when you communicate with the rest of the team. It’s one document that tells everyone involved what's going on, when, and why. The term "test plan" sounds quite formal, but really all that your plan is doing is listing out the decisions you've made and the things that need to happen for the usability test to take place.

This helps you keep track of what's going on and what still needs to be done. Most of the information in the test plan is stuff that comes from other documents you use during the planning and execution of the study. For instance, your participant profile, the study schedule, and your task list. You’ll also pull in information that might not be written down anywhere, such as the research questions that led to your task list.

Testing Methods

Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Beta Testing

Beta testing is the testing which is done by end users, a team outside development, or publicly releasing full pre-version of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.

Black box testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

Software testing

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also, to assess the feature of A software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

Verification

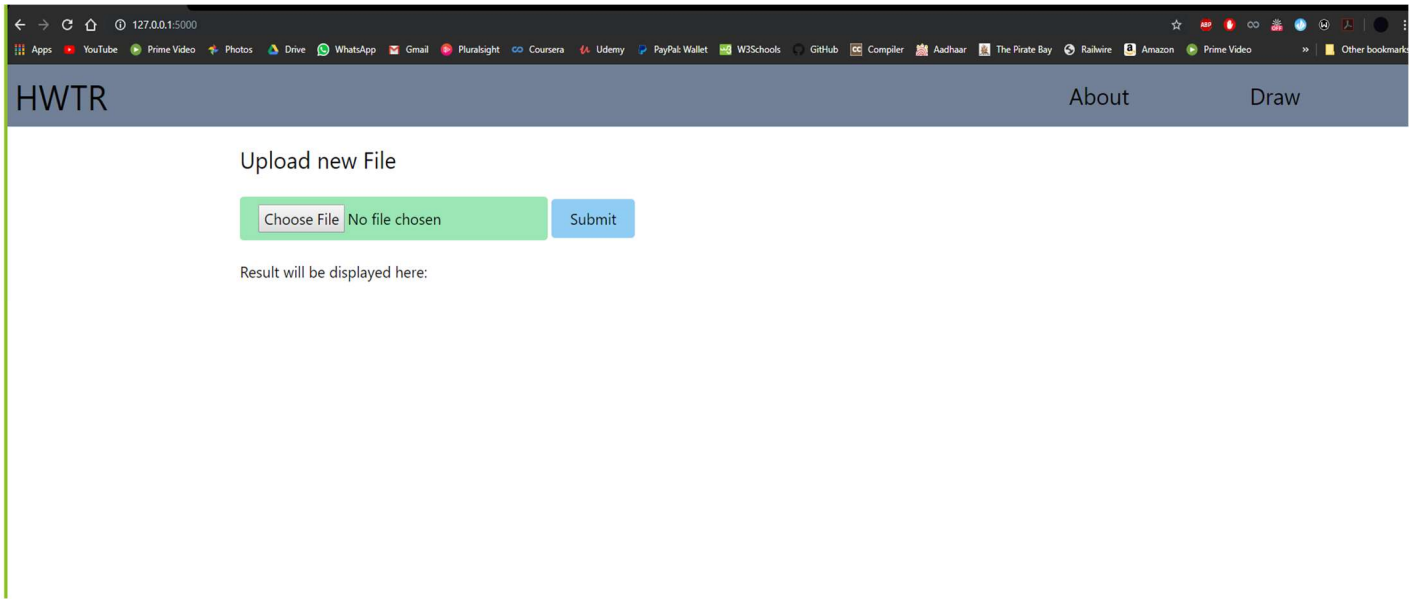
Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Validation

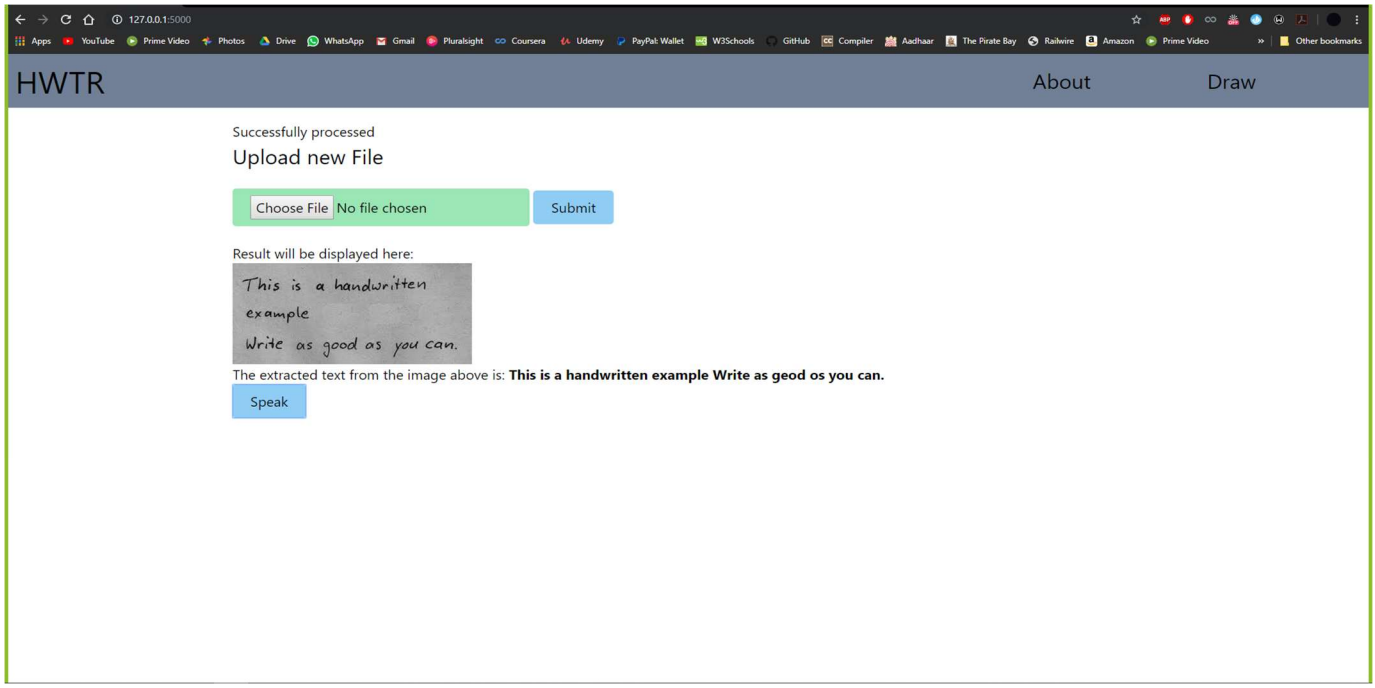
Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

CHAPTER-8

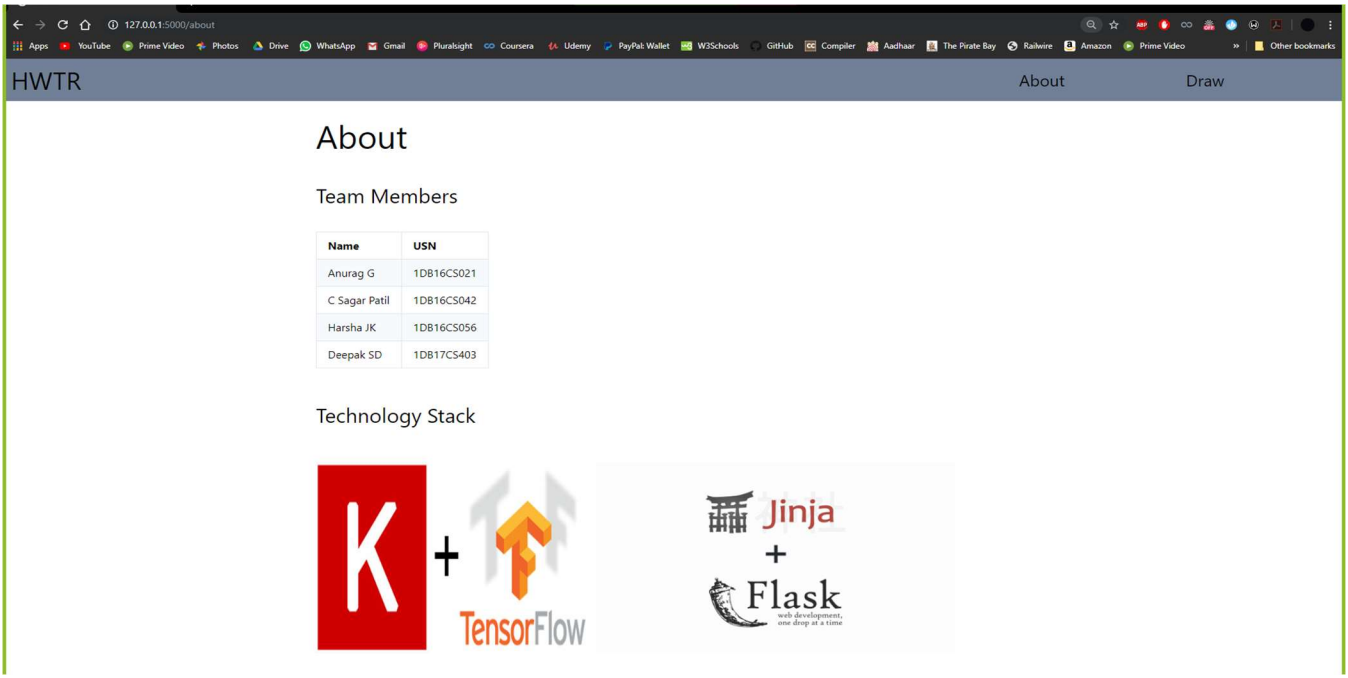
RESULTS



Snapshot-1


127.0.0.1:5000
HWTR
About
Draw
Successfully processed
Upload new File
Choose File
No file chosen
Submit
Result will be displayed here:
This is a handwritten example Write as good as you can.
The extracted text from the image above is: This is a handwritten example Write as geod os you can.
Speak

Snapshot-2


127.0.0.1:5000/about
HWTR
About
Draw
About
Team Members
Name
USN
Anurag G
1DB16CS021
C Sagar Patil
1DB16CS042
Harsha JK
1DB16CS056
Deepak SD
1DB17CS403
Technology Stack
TensorFlow
Jinja
+
Flask

Snapshot-3

CHAPTER-9

CONCLUSION AND FUTURE SCOPE

Conclusion:

- An online handwriting recognition system for English based characters has been developed. The system is writer-independent text recognizer developed based on convolutional neural network approach with text-to-speech conversion system.

Future Scope:

- The proposed work can be extended to work on degraded text or broken characters.
- The system could be designed to recognize the writer and thus can lead to achieve signature recognition which can be used in bank check processing.
- Language translation system can be implemented to covert the language into another desired language.

CHAPTER-10

REFERENCE WORK

- Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018), IEEE Xplore - Handwritten Character Recognition Using Deep-Learning.

- International Journal of Management, Technology And Engineering Volume 8, Issue VII, JULY/2018 ISSN NO : 2249-7455.

- NIST Handprinted Forms and Characters Database obtained by, https://www.kaggle.com/sachinpatel21/az-handwritten-alphabets-in-csv-format.

- https://medium.com/@himanshubeniwal/handwritten-digit-recognition-using-machine-learning-ad30562a9b64

CERTIFICATES

INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

CERTIFICATE OF PUBLICATION

This is to certify that

Harsha J. K.

published a paper entitled

Handwritten text pattern recognition

Volume-6, Issue-3 - May-June, 2020

ISSN:2454-132X
IMPACT FACTOR 6.018

Ref No. OAP/II/203/522

editor@ijariit.com, ijariit@gmail.com

Editor-In-Chief

INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

CERTIFICATE OF PUBLICATION

This is to certify that

Deepak S. D.

published a paper entitled

Handwritten text pattern recognition

Volume-6, Issue-3 - May-June, 2020

ISSN:2454-132X
IMPACT FACTOR 6.018

Ref No. OAP/II/203/521

editor@ijariit.com, ijariit@gmail.com

Editor-In-Chief