|  |  |
| --- | --- |
| Name of the Student | Aayush Verma |
| Internship Project Title | Automate Extraction of Handwritten Text from an Image. |
| Name of the Company | TCS iON |
| Name of the Industry Mentor | Anamika Chatterjee |
| Name of the Institute | Maharashtra Institute of Technology, Pune |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Start Date | End Date | Total Effort (hrs.) | Project Environment | Tools used |
| 18-May-2020 | 01-Aug-2020 | 210 | Google Colab – Jupiter Notebook | Opencv, Numpy, Python imaging library,  Pytesseract. |
| **Project Synopsis:** The industry project is all about extraction of handwritten text from an images, which is optical recognition of characters is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image.  Widely used as a form of data entry from printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation – it is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining. OCR is a field of research in pattern recognition, artificial intelligence and computer vision.  The main objective of this project is to develop machine learning algorithm in order to enable entity and knowledge extraction from documents with handwritten annotations, with an aim to identify handwritten words on an image. | | | | |
| **Solution Approach:** I’m using Pytesseract using machine learning approach although there are various approaches to implement this project, but at last main aim of this internship is to fulfill project aim and objectives.  To meet project objective I have studied about convolution neural networks. The agenda for choosing this field because it enable machines to view the world as humans do, perceive it in a similar manner and even use the knowledge for a multitude of tasks such as Image & Video recognition, Image Analysis & Classification, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning has been constructed and perfected with time, primarily over one particular algorithm is Convolutional Neural Network.  To implement my project I have used tesseract python as suggested in the Webinars as well as Self Learning module. Used python tools such as Opencv, Numpy, Python imaging library and Pytesseract. The project design is divided into parts. | | | | |

|  |
| --- |
| 1. The first part designed to take the input of the image whose text is to be extracted. 2. The second part of the project is the main part of the project which designed to reduce the noise and implements the Pytesseract with convolutional neural network algorithm step by step in order to identify the text present in the image. |
| **Assumptions:** The assumptions considered are as follows:   1. The handwritten text across must be in English. 2. The text across the input image must be clearly handwritten in order to achieve good results. 3. All machine dependencies must be installed properly. |
| **Project Diagrams:** Logic Flow of Convolution Neural Networks  Project Diagram.png  **Logic Flow of Pytesseract OCR Engine**  tesseract flowchart.jpg |

|  |
| --- |
| **Algorithms:** The algorithm used to implement the project is the Pytesseract ocr engine which uses convolutional neural network algorithm which is used by the tesseract optical character recognition engine in python. There are four layered concepts we should understand in convolutional neural networks:   1. Convolution 2. Rectified Linear Unit 3. Pooling Layers 4. Full Connectedness (Fully Connected Layer)   **Convolution Of An Image:** Convolution has the nice property of being translational invariant. Intuitively, this means that each convolution filter represents a feature of interest (e.g pixels in letters) and the Convolutional Neural Network algorithm learns which features comprise the resulting reference (i.e. alphabet).  We have 4 steps for convolution:   * Line up the feature and the image * Multiply each image pixel by corresponding feature pixel * Add the values and find the sum * Divide the sum by the total number of pixels in the feature   convolution.png  The output signal strength is not dependent on where the features are located, but simply whether  the features are present. Hence, an alphabet could be sitting in different positions and the convolutional neural network algorithm would still be able to recognize it.  **Rectified Linear Unit:** Transform function only activates a node if the input is above a certain quantity, while the input is below zero, the output is zero, but when the input rises above a certain threshold, it has a linear relationship with the dependent variable. |

|  |
| --- |
| relu.png  The main aim is to remove all the negative values from the convolution. All the positive values remain the same but all the negative values get changed to zero as shown below:  reluu.png  Inputs from the convolution layer can be smoothened to reduce the sensitivity of  the filters to noise and variations. This smoothing process is called sub sampling and can be achieved by taking averages or taking the maximum over a sample of the signal.  **Pooling Layer**: In this layer the shrink the image stack into a smaller size. Pooling is done after passing through the activation layer. We do this by implementing the following 4 steps:   * Pick a window size (usually 2 or 3) * Pick a stride (usually 2) * Walk your window across your filtered images * From each window, take the maximum value |

|  |
| --- |
| pooling.png  We took window size to be 2 and we got 4 values to choose from. From those 4 values, the maximum value there is 1 so we pick 1. Also, note that we started out with a 7×7 matrix but now the same matrix after pooling came down to 4×4.  But we need to move the window across the entire image. The procedure is exactly as same as above and we need to repeat that for the entire image. Do note that this is for one filter. We need to do it for 2 other filters as well. This is done and we arrive at the following result:  pool.png  Well the easy part of this process is over. Next up, we need to stack up all these layers!  **Stacking Up the Layers:** So to get the time-frame in one picture we’re here with a 4×4 matrix from  a 7×7 matrix after passing the input through 3 layers – Convolution, Rectified Linear Unit and Pooling as shown below: |

|  |
| --- |
| sul.png  We further reduce the image from 4×4 to 2x2 to achieve this we have to perform the 3 operations in iteration after the first pass. So after the second pass we arrive at a 2×2 matrix as shown below:  sull.png  The last layers in the network are fully connected, meaning that neurons of preceding layers are connected to every neuron in subsequent layers.  This mimics high level reasoning where all possible pathways from the input to output are considered.  Also, fully connected layer is the final layer where the classification actually happens. Here we take our filtered and shrieked images and put them into one single list as shown below:  sull1.png  So next, when we feed in, ‘X’ and ‘O’ there will be some element in the vector that will be high. Consider the |

|  |
| --- |
| image below, as you can see for ‘X’ there are different elements that are high and similarly, for ‘O’ we have different elements that are high:  sull2.png  Well, what did we understand from the above image is when the 1st, 4th, 5th, 10th and 11th values are high; we can classify the image as ‘x’. The concept is similar for the other alphabets as well – when  certain values are arranged the way they are, they can be mapped to an actual letter or a number which we require.  **Prediction of Image Using Convolutional Neural Networks – Fully Connected Layer**  At this point in time, we’re done training the network and we can begin to predict and check the working of the classifier. Let’s check out a simple example:  1-25.png  We have a 12 element vector obtained after passing the input of a random letter through all the layers of |

|  |
| --- |
| our network. We make predictions based on the output data by comparing the obtained values with list of ‘x’and ‘o’.  1-26-768x387.png  We just added the values we which found out as high (1st, 4th, 5th, 10th and 11th) from the vector table of X and we got the sum to be 5. We did the exact same thing with the input image and got a value of 4.56**.**  When we divide the value we have a probability match to be 0.91! Let’s do the same with the vector table of ‘o’ now:  2-15-768x380.png  We have the output as 0.51 with this table. Well, probability being 0.51 is less than 0.91, isn’t it? So we can conclude that the resulting input image is an ‘x’. And this is how prediction work is done. |

|  |
| --- |
| **Outcome:** The algorithm is able to detect and segment handwritten text from an image. The model successfully able to detect maximum words in a given line of sentence or words, which makes it about 90% accurate while implementation and testing.  For example the input image having the handwritten text is given as following:  test img2.jpg  The model processes the image removes the noise from the image and the Pytesseract performs the convolution neural networks and predicts the text.  Extracted Text : **TCS jON RIO - 125**  As we can see the model is quite accurate and successfully able to extract the handwritten text. The model predicts and extracts the text from the image as follows. |

|  |
| --- |
| Whereas another image having the handwritten text is given as following:  test img.jpeg  The model processes the image removes the noise from the image and the Pytesseract performs the convolution neural networks and predicts the text.  Extracted Text : **Jim Morrison**  As we can see the model is quite accurate and successfully able to extract the handwritten text. The model predicts and extracts the text from the image as follows: |

|  |
| --- |
| Whereas another image having the handwritten text is given as following:  screenshot1.jpeg  The model processes the image removes the noise from the image and the Pytesseract performs the convolution neural networks and predicts the text.  Extracted Text : **This is the handwritten**  **example**  **Write as qooal as you can**  As we can see the model is quite accurate and successfully able to extract the handwritten text. The model predicts and extracts the text from the image as follows: |

|  |
| --- |
| **Exceptions considered:** The exceptions considered are as follows:   1. The text across the input image must be of the same color not multicolor handwritten text. 2. The image doesn’t have too aggressive multicolor backgrounds across the text of the image. 3. The image doesn’t have any kind’s objects in the background across the text of the image. |
| **Enhancement Scope:** The enhancement scope of this project are follows:   1. The accuracy of the model can increase with predefined models and powerful machine learning GPU processors can be used to attain a good percentage of accuracy. 2. In future we can use this algorithm with more than one particular language. |
| **References:**  https://software.intel.com/content/www/us/en/develop/training/course-artificial-intelligence.html https://software.intel.com/content/www/us/en/develop/training/course-machine-learning.html https://[www.python-course.eu/machine\_learning.php](http://www.python-course.eu/machine_learning.php)  https://numpy.org/doc/ https://pypi.org/project/pytesseract/  https://[www.geeksforgeeks.org/python-using-pil-imagegrab-and-pytesseract/](http://www.geeksforgeeks.org/python-using-pil-imagegrab-and-pytesseract/) [https://medium.com/analytics-vidhya/convolutional-neural-networks-cnn-explained-step-by-step- 69137a54e5e7](https://medium.com/analytics-vidhya/convolutional-neural-networks-cnn-explained-step-by-step-%2069137a54e5e7)  https://drive.google.com/drive/folders/13MX9gDLLMfBpQvdOY5ndA\_\_EfhXiBuxz?usp=sharing |
| Link to Code and executable file:  <https://github.com/aayush2019/TCS-iON-Internship/blob/master/TCS_iON_Aayush_Verma.ipynb> |