Shape

Description automatically generated with medium confidence

Supervisor:

**Prof. Nello Cristianini**

Student Number:

**2051421**

DEPARTMENT OF COMPUTER SCIENCE

An Exhaustive comparison of OCR algorithms under various conditions

A project plan submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering.

**Supporting Technologies**

* I used Python version 2.7 for older OCR tools like OCRopus and version 3.7 for other OCR tools.
* I used OpenCV computer vision library to perform all the image related tasks, such as, image processing, detection etc.
* I used git version control to keep my project files safe and secure with a track of all the updates made.
* I used various python libraries such as, *NumPy* ( <https://numpy.org/doc/stable/> ), *Pandas* ( <https://pandas.pydata.org/docs/> ), *Matplotlib* ( <https://matplotlib.org/stable/contents.html> ).
* I used licenced version of Microsoft Word as the processor to format my thesis.

**Notation, Acronyms, and Terminology**

OCR : Optical Character Recognition

NN : Neural Networks

ANN : Artificial Neural Network

CNN : Convolutional Neural Network

RNN : Recurrent Neural Network

LSTM : Long Short term Memory

CER : Character Error Rate

WER : Word Error Rate

VGG : Visual Geometry Group

ILSVR : ImageNet Large Scale Visual Recognition Challenge

**Acknowledgements**

First of all, I would like to thank my mentor and supervisor Prof. Nello Cristianini for providing continuous support and providing motivation at every point during this thesis. He has always worked hard to keep a check on my work and to validate the findings. His immerse knowledge has not only helped me shape the style of working but has also made me understand the smarter way of coding.

**Abstract**

Optical Character Recognition or OCR is a technological conversion of images of texts, either machine written or handwritten, or scanned copies into an editable or machine-readable format for further processing. OCR has gained a lot of popularity in the last few decades and there are various tools available like Adobe, Abbyy and Rossum; just to name a few. A few open-source tools like tesseract, vision, ocropus have a tremendous count of forks on Github and they lay a foundation of many other tools and frameworks available in the market. The accuracy of the image to text conversion is dependent on multiple factors like the quality of the image, tools used to carry out the process, pre-processing steps used, text category, etc. There are various constraints related to the conversion of images to digitized text as images tend to have multiple complexities, making it inefficient for machines to understand the text embedded into the image. A major chunk of OCR systems is used on character isolation. All the characters of a sentence are read individually to make it easier for the machines to understand and process the document. However, a low-resolution image makes it difficult to understand the text, which can further lead to problems like character mixing. Moreover, it is difficult to find the difference between the digits like 6 and 8 or 1 and 7, due to their writing structure. The background colour on which the text is printed also plays an important part in text recognition. Due to a lot of research, there are a variety of OCR tools available in the market with high accuracy. If the OCR tools have very high accuracy, do they differ in any way? Do different architectures give the same accuracy? Does the input image change the output for different OCR tools? This thesis goes a level up to the other previous research. Here we compare the outputs of the two most famous Optical Character Recognition engines: Tesseract and OCRopus, by using different quality, category, and format of images.

We start this thesis by introducing OCR and then moving towards the more technical side of the research. A thorough background is covered with the introduction of all the technical terms, introduction to all the conceptual terms, history of both the OCR tools, architecture of both the OCR tools, a comparison of both the tools and the resulting experiment. We use different graphical outputs throughout the thesis to compare the results of the tools. The experiment is performed using the Python programming language due to its vast collection of libraries and api support for using the OCR tools. And the integrated development environment used is the Jupyter notebook because it gives all the necessary formatting to compare and interpret the results either graphically or numerically.

The dataset that we use to experiment consists of a collection of images from different categories of media sources with varying dimensions and the quality of text captured. Agile methodology is used as the working method for model creation and comparison throughout the project.

In terms of accuracy, the OCRopus fails in comparison to tesseract for a clean document because of the integrated LSTM architecture above the classifier in the Tesseract engine. Moreover, tesseract has more number contributors and updates as compared to OCRopus, updating it with technology. But for irregular and trickier documents, both the tools failed to give a satisfying performance. There is no clear winner out of the two. The thesis concludes by stating the results of the comparative study of the tools. It gives an insight into the category of tool that should be used by initially analysing the condition of the input image. It also helps to understand the commercial usage of tools according to the size of the input document. Moreover, a detailed understanding of the architecture and the steps involved in the optical character recognition of both tools is gained by the end of the project. This thesis also lays the foundation for using the other OCR tools in the market as it compares the two most popular tools having complex architectures.

**Chapter 1**

**Introduction**

This chapter of the thesis gives an overview of the project and specifies the need for this project. It also clarifies the selection of the specific tools and their usage. Moreover, it states the need for the topic, its importance and outlines some of the common challenges related to the problem statement.

**1.1 Evolution of Optical Character recognition**

OCR is the procedure for the conversion of images of text to machine-readable format. Recognising the characters has been a challenging task for a long time and the roots of OCR lay back to the 1920s when physicist Emanuel Goldberg invented a mechanical hardware was used to detect text. This machine could read the text and convert it to telegraph code. These mechanical systems were very time consuming and less accurate. They used to work by recognising one character at a time, after many iterations, which made them less reliable for today’s era. After these hardware models, no progress was made towards their enhancement. Later on, after the development of high-capacity hardware, there were progressions made towards the research and development of OCR technologies. In 1970s, researcher Kurzweil introduced omni-font OCR, which was capable of processing any printed text. Later, in 2000, OCR became available on cloud technology with worldwide access, compatible with any device. It is in the last two decades that OCR has made tremendous progress in the technological field. Due to increased accuracy and less time-consuming tools, it has gained a lot of popularity. But even after so many years of research and innovation in the methodologies, outputs of OCR still possess errors.

Since the start of OCR systems till the present available systems, they undergo numerous calculations before the actual text recognition task takes place. The only difference is the processing time and the methods to carry the process. Today these calculations are carried out in a progressive manner, known as the pipeline. Every OCR tool has its own pipeline, which makes them different from one another in terms of speed of calculations, accuracy, efficiency, etc. A common pipeline for OCR consists of image pre-processing as the first step. Moving to binarization and page segmentation as the second and third steps. Line and word segmentation is performed as step four. As the last step of pipeline, process of text extraction is carried. The best OCR tools that are compared in this thesis work on highly complex recurrent neural network known as Long Short-Term Memory.

**1.2 Importance of the topic**

In the last 3 decades, OCR technologies have undergone a lot of research and exploration. Due to this continuous research, the market is flooded with a lot of tools to perform OCR. All the tools available in the market have their highs and lows, due to which it becomes important to analyse the accuracy and performance of the two most popular OCR tools. Moreover, the research being carried out are to make specific tools better and none of them is carried out to undergo a comparison of the existing models. This topic helps to identify the performance and accuracy of the best available OCR tools in the market. Moreover, it is beneficial to understand the important features of the input dataset that makes the system more or less dependable.

**1.3 Usage of OCR**

Due to a wide number of applications, OCR is a key area of interest for researchers. Banking is one of the biggest sectors where OCR plays a crucial role. The OCR technology is used to automate the processing of cheques by reading the text and the digits written on the cheque. This can reduce the time required by the customers for the transactions related to cheque by a huge percentage. OCR can also be used to archive millions of old books, newspapers, and other important documents so that they can be easily accessed online and can also be preserved. Moreover, this also reduces the physical space taken by these old books. This technology has heavy use in the medical sector, where it can be used to save the records of the patients and various kinds of forms in digital format. It can be used to automate the mailing industry by reading the contact details written on the parcel. OCR can be used to keep a track of all the bills and other financial documents for future references. In short, OCR has a wide range of applications for industrial as well as personal usage.

**1.4 Challenges to perform OCR**

Implementing OCR tools are always challenging. The domain requires a lot of research and knowledge to perform even a small task. Good quality hardware is important to run the algorithms of both the tools, as they are big and work on heavy machines. Since the dataset consists of only images, it is big data and requires a lot of disk space. Working on a small laptop is very time-consuming for OCR related projects. Moreover, there are many symbols and characters in Unicode and OCR tools are not trained to understand them all, which makes the evaluation difficult on character rich document. Languages also play an important role in the performance of the tool and possess heavy challenges for computation. Simple English text can easily be converted but when it comes to tougher languages like Chinese, Arabic, etc. OCR systems tend to fail due to the style of writing and difficulty to divide the sentences into words and characters. Typographic errors also make the conversion challenging. These errors can be due to the italics style writing, background colour error, errors due to the size of font etc. All these aspects make performing OCR challenging and difficult. And due to these challenges, the accuracy of OCR tools is not 100% and requires human intervention to complete any task.

**1.5 Why Tesseract and OCRopus**

Out of n-number of available OCR tools in the market, this research specifically focuses on the Tesseract and OCRopus. This selection is made by taking care of a few parameters. Firstly, the open source community of both the tools is taken into consideration. These tools have a large fan base and hence a very diverse community. Secondly the architecture of both the tools has been into development since a long time and has evolved in multiple stages. This makes the research more deep and gives a chance to understand the changes made in the architecture with the specific reasons and results of those changes. Moreover, both these tools use specialised recurrent neural network - LSTM as there base extractor, making them very advance and complex technology. And because of all the stated reasons, these tools hold a good amount of work that can be carried out in future to make them more accurate and reliable. This gives a scope for this research to be carried forward and create a reference for future researches/work being done on the these two tools.

**1.6 Thesis Overview**

This thesis comprises 5 chapters. These chapters are arranged in an easy to understand order. The first chapter gives the introduction of the whole thesis starting from the basic definition of OCR to its use-cases. The second chapter covers all of the technical concepts which are essential to experiment successfully. These concepts start from the basic architecture of the tools being used to their practical working. The third chapter carries all the execution part. In this, all the essential coding for both tools is done to implement the research. This section discusses the hurdles that arise in the execution and how those hurdles were solved to implement the OCR. The fourth chapter outlines the comparison of the outcomes of both tools. It categorises the tools according to their performance and accuracy. The last chapter lists the outcome of the experiment, with final results and gives a summary of the project. Due to continuous evolution in the OCR technologies and there is increased usage, this thesis also includes the scope of the future works that can be carried out using this research as a reference. Moreover, it also states how this research can also be extended to compare other commonly available and famous open-sourced OCR tools with comparable fan base and availability in the market. Finally, a reference list of all the resources used to carry out this research is attached, so as to give a better understanding and reliability to this project.

**Chapter 2**

**Motivations**

This chapter highlights the major points that brings out the motivation to perform the task involved in this thesis.

**2.1 Quality of tools**

With the advancement in technology and a wide research in the computer vision and neural networks, OCR tools available in the market have gained a lot of popularity. Due to diverse usage and increased demand, there has been a significant rise in the number of tools available. This has resulted in creation of various tools with low accuracy and improper quality of analysis.

**2.2 Variety of documents / Writings**

Since centuries, writings have been considered to be a consequence of cultural diffusion originating from the Mesopotamian civilisation from the early Bronze era. After multiple discoveries from various regions of the world, it has been confirmed that the origin of writing is not from one region or century rather there were multiple origins. These writings had varied structures and form, which ultimately gives a solid evidence of having multiple independent origins. With the continuous evolution from writing on stones to writing on paper and in recent times, with the advancement in technology, going paperless has become a new mantra. This lead to creation of billions of document types with varied writing styles. Converting a good amount of text from various image types having different structure and language style is a challenging task.

**2.3 Commercial Challenges**

In various work places, a vast amount of time is spent on unwanted tasks such as sifting through stacks of documents, performing manual entries, organising bills, matching documents etc. Due to these unproductive tasks, the demand of OCR technology is increasing in every domain of the industry. With the technological evolution and increased demand, there are various OCR tools available in the market and it becomes difficult to choose the best one according to the structure, quality and the quantity of the document.

**2.4 Diverse Industrial Uses**

OCR technologies help to streamline and reduce manual intervention in various sectors of the industry. It saves a lot of human efforts, finances and time required to perform the tedious data entry tasks. Some of the major industrial sectors that use OCR technology to perform tasks in a smoother way are Banking, Medical sector, Finance industry and insurance industry.

**2.4.1 Banking Industry**

Banking Industry uses OCR in various places due to a lot of paper work involved. This industry was amongst the early stage adopters of this technology where numbers on the cheque were recognised using OCR. At present not only numbers, but the user signature, place and name are also extracted using this technology. Another basic application of OCR is in the use of Automated Teller Machines, where the machine reads the number on the cards to fetch account details and deliver cash to the customers. After 2016, various neo banking platforms have evolved which validates the user data online with the help of OCR.

**2.4.2 Medical Sector**

Medical sector also has a lot of paper work involved, ranging from patients subscriptions to various medical records. Keeping a stack of all the historical medical data becomes a headache. OCR helps in converting all the records in the digital format, so that it can be easily accessed from any location at any time required. These digital documents make it easier to get a better overview of the patients health and to diagnose any new disease. Moreover, scientists and pharmacists can store all the information related to various drugs in the digital format and can access them easily from cloud platforms whenever required.

**2.4.3 Finance Industry**

Finance professionals, charted accountants and analysts can profit substantially from OCR technology since it allow them to focus on high-priority activities rather than processing data using outdated ways. Filing income tax returns at the year end becomes very hectic due to lots of hard copies of bills, expenses and other assets. Digitizing these documents can save a lot of time for financial planners and tax payers as it makes documents, bills and other financial statements error free.

**2.4.4 Insurance Industry**

Insurance industry is no exception when it comes to lengthy hardcopies and tremendous amount of manual paper work. To give better costumer experience, insurance companies have now started to make customer onboarding easier and faster using OCR technology by extracting essential verification information from the documents uploaded by the user.

**2.5 Existing Work**

Even after having many use cases, there are very less optical character recognition tools available to convert images into text in the market for free. The available OCR tools have their own set of steps to implement them on different images dataset. Due to these differences, the metrics used for evaluating the performance of these tools in dependent on multiple factors and vary according to the implementation and the results of the tool.

In terms of existing comparisons and predictions, Vijayarani [11] lists few of the common OCR tools available online for free, to compare their performance on the basis of accuracy and word error rate. She talks about text types and their performance issues while applying the conversion task. Her result specified that the OCR tools were not able to detect and convert the symbols and different kinds of font involved into text. The outputs in that space were usually blank.

Shivani [12] talks about the two open source OCR tools and compares them on the basis of two different parameters accuracy and precision. She told about the difficulty of comparing OCR performances for the handwritten images and takes into consideration only the printed documents. Their results specified that the tesseract out performs Google OCR for most of the documents but lags in some cases.

Marcin [13] compares the OCR tools amongst the highest market share holders. They highlight the importance of training the models to increase the performance of both the tools. Their comparison results marked the competitiveness in the OCR tools available.

**Chapter 2**

**Technical Background**

This chapter of the thesis introduces the concepts related to the OCR tools that are compared. It starts with important terminologies and a brief introduction of neural networks which form the base of the OCR tools. Then it moves towards the Recurrent neural network, which is a specific category of neural network used for our tools. Finally, it gives a brief history of both the tools followed by their architectural overview and working.

**2.1 Terminology**

**Pixel.** A pixel is the building block of an image. These pixels are responsible for given different colours to the image as they are made of variable intensities. A resolution of 1920 \* 1080 pixels resembles that there are 1920 pixels in every row of the image and 1080 pixels in every column of the image. The more the pixel density, the better is the image quality.

**Binary image.** A binary image also called a two-segment or bi-segment image is an image that consists of only two colours, black or white. This means that the image consists of only two types of bits: 0 or 1, hence the pixels will be stored as either 0 or 1. It is a very common term in image processing and

**Grayscale image.** A grayscale image, also known as a grey monochrome image, is made up of only shades of grey. It is different from black and white images as it consists of all the shades of grey and not just black and white colours.

**Digitized image.** It is an image that is converted into a format that can be used as an input to the machines.

**2.2 Neural Networks**

The brain is the most complex segment of the human body. Numerous studies are going on to understand its structure and working. Due to its vast abilities and small size, it attracts a lot of research and study. The beginning of neural networks, in artificial intelligence, to mimic the functions of a human brain through different sets of algorithms was marked by McCulloch and Pitts in 1943 (Hardesty, 2017) [1]. The roots were laid to solve computational problems in a way similar to the human brain. Later, with the advancement of technological hardware and reduction in cost, these were used to perform a variety of tasks. They carry a lot of importance to solve day to day problems with multiple types of complexities. These networks are made up of neurons that resemble the biological neurons in the brain cells and are used to perform complex tasks. There are majorly three types of neural networks known as the artificial neural network, convolutional neural network, and recurrent neural network.

**2.2.1 Artificial Neural Network**

Artificial Neural Network, commonly known as Feedforward Neural Network, is considered to be the simplest division of neural networks as it consists of a group of neurons working in a single direction. Its roots lay back to 1958 when psychologist Frank Rosenblatt invented the Perceptron to corelate it with the human brain (Rosenblatt, 1958) [2]. It does not perform well on image data as it takes input only in one dimension, which means that the images are required to be converted into one dimension. This conversion not only increases the parameters of the input but also removes the features of the image. Hence, ANN cannot function well for image data.

Diagram

Description automatically generated

Figure Artificial Neural Network

**2.2.2 Convolutional Neural Network**

Convolutional Neural Networks are generally used for visual tasks. There were many different models introduced to recognise patterns similar to the way a human brain recognises but all of them had distortion and positioning issues. The famous ones include a three-layered perceptron introduced by Frank Rosenblatt in 1962 [4] and a visual information processing model given by M. Kabrisky in 1966 [5]. In 1980, Kunihiko Fukushima, a Japanese computer scientist, talks about the self-organising ability of these neural networks to recognise patterns and calls them neocognitron [3]. The building blocks of CNNs are known as kernels. It easily captures the data from an image and finds the relationship between the individual pixels of that image. This helps in dividing and identifying different aspects of the image. The major drawback of this network is that it becomes slow when the number of hidden layers is increased. Moreover, it is very complex to build if the dataset has varied features.

**2.2.2.1 Architectures**

**Diagram

Description automatically generated**There are various traditional architectures for CNNs that have similar general principle of stacked layers of networks whereas the newer architectures try to find a more reliable and faster network layer to build CNN. Almost all of these networks follow the same design principles for the input. The famous LeNet5, introduced to recognise the handwritten digits by Yann Lecun et el in 1998 [6] is the historical CNN architecture which is trained on 60000 images dataset.

Figure- LeNet-5 architecture for digits recognition

The AlexNet architecture is based on the LeNet-5 architecture and was developed in 2012 by Alex Krizhevsky et al. to complete in the ImageNet competition.

Diagram

Description automatically generated

Figure- AlexNet Architecture

Diagram, engineering drawing

Description automatically generated

Figure VGG16 Architecture

Recurrent Neural Networks are built on artificial neural networks. A special type of RNN known as Associative Neural Network was first introduced by J. Hopfield in 1982 [7]. These networks can remember the sequence of inputs in their attached memory to improve their predictions. RNNs are computationally less expensive because they receive input from the previous layer, allowing them to perform predictions with less input data. It could be a difficult task to train and process long sequential data in a recurrent neural network.

All the three categories of neural networks are used for different purposes since they all have varying structures.

**2.3 Long Short Term Memory**

A very popular kind of gradient based Recurrent Neural Network known as Long Short Term Memory or LSTM was introduced by Hochreiter and Schmidhuber in the late 90s to solve the problem of information storage over extended time intervals by modifying the gradient [8]. It is a neural network that is much more advanced than a feedforward neural network. The main goal of creating LSTM was to increase the accuracy of RNN models and to solve the problem of long-term dependency.

Every LSTM unit consists of a memory cell which can store information for a specific duration of time. The LSTM architecture includes three gates: the input gate, the output gate, and the forgotten gate, which are in charge of information retrieval, storage, and deletion. The input gate decides the quantity of data to be stored from the previous layer. The output gate decides the quantity of data that should be sent out to the next layer. And the forgotten gate decides when the data should be removed. There are various kinds of LSTM models used with slight variations. A famous LSTM version was introduced by Gers & Schmidhuber in 2000, in which the state of the cell can be examined by gate layers. Another variation was introduced by connecting input and forgetting gates, where the judgments are made collaboratively rather than individually determining what to leave out and what to add. A model that is easier to understand than an ordinary LSTM model, known as the Gated Recurrent Unit, was introduced by Cho, et al. in 2014.

**2.4 Image Processing**

In simple terms, image processing can be defined as the process of enhancing the image for various kinds of information extraction. There are mainly two methods used for image processing, namely, analog image processing and digital image processing. The analog image processing method processes hard copies or physical images. Whereas, the digital image processing method processes digital images, that is, images stored on computers or other electronic devices with the help of various computational algorithms. There are a number of open-source image processing libraries available on the market, the most popular of which is OpenCV, which contains a number of libraries that are required for image processing tasks.

**2.5 History of OCR tools**

**2.5.1** **Tesseract**

The Tesseract is one of the most popular tools used for Optical Character Recognition. Its investigation was a research project that began in 1984 at Hewlett Packard Labs in Bristol and lasted until 1994. Some critical changes were made to the port's windows in 1996. In 2005, Tesseract was open sourced by Hewlett Packard and, since 2006, Google has been the lead sponsor for the development and testing of Tesseract. Version 4 of Tesseract was released in 2018, with a major update to the integration of the LSTM based recognition engine.

**2.5.2 OCRopus**

OCRopus is another famous OCR tool which was initially released in 2007 under the sponsorship of Google. Before the stable release in March 2010, there were multiple beta releases until 2009. It started as a PhD project at the German Research Center for Artificial Intelligence and the University of Kaiserslautern. The software was originally written in C++ and Python, but in 2012, version 5 was released with completely refactored code in Python.

**2.5.3 EasyOCR**

EasyOCR is a modern character recognition library released in July 2016. It is created in the name of a private organisation, which specialises in AI, registered as JaidedAI by using multiple research papers and open source code repositories with an aim to build a better performing character recognition tool. It is regarded as one of the easiest to use character recognition tools amongst available high accuracy tools. With the latest version released in 2021, this library supports 80+ languages which includes the most difficult writing scripts such as Chinese, Devanagari, Latin etc. The library is most suitable for python users as it is written in python with complete deep learning segment in Pytorch.

**2.6 Architecture of OCR tools**

**2.6.1 OCRopus**

The design of OCRopus was done by taking into consideration the modular approach as its objective was to digitize a huge volume of information contained in books. Its pipeline is composed of a feedforward network consisting of 3 main components. The first component is known as layout analysis, wherein the lines, blocks and columns of texts are identified by the use of the Gaussian Error Model’s maximum likelihood estimate. Line character recognition, also commonly understood as the second component in the OCRopus architecture, is used to recognize the characters in every line. Alternative recognition hypotheses are combined with prior knowledge of language, vocabulary, and the document's domain in statistical language modelling, making up the third component, known as statistical language modelling.

Layout Analysis

Line Character Recognition

Statistical Language Modelling

Figure 5: Architecture of OCRopus

**2.6.2 Tesseract**

The architecture of the Tesseract is a combination of 5 different segments. The first segment consists of Adaptive Thresholding, wherein the image is converted into its binary version. In the second segment, known as page layout analysis, the character outlines forming the individual blocks are extracted from the converted page. In the third segment, which is commonly known as the line recogniser segment, the outlines of the text are recognised and are converted to blobs. These are further divided into words by understanding the area or text size. The last two segments are used for the word recognition task. The first segment performs word recognition using a simple classifier. The words which cross a threshold level are recognised and saved as a training dataset for the classifier. Using this dataset, the second segment looks for the remaining words which were not recognised in the previous segment.

Adaptive Thresholding

Line Recognizer

Page Layout Analysis

Word Recognition 1

Word Recognition 2

Figure 6: Architecture of Tesseract

**2.6.3 Architecture EasyOCR**

The architecture of EasyOCR consists of multiple different segments taken from several research papers. The first segment known as the Detection part is based on Character Region Awareness for Text Detection or CRAFT framework. This framework was introduced by Baek et al. in 2019 with an aim to overcome the problem of character level annotation [9]. It is designed by using batch normalisation over the VGG16 architecture which is a convolution neural network. It gives two separate outcomes as the region score and the affinity score. The region score determines individual character level localisation while the affinity score is used to create a single instance by using each character. Figure 6 illustrates the schematic overview of the CRAFT framework with output scores.

The second segment known as the recognition model is based on Convolutional Recurrent Neural Network. This model was proposed by Shi et al. in 2017 to solve the problem of scene text recognition, which is amongst the most important image based sequence recognition [10]. This segment is a combination of three different modules, namely, feature extraction module, sequence labelling module and decoding module. The feature extraction module uses the resnet architecture so as to make it easier to train the deeper networks. This architecture was introduced by assembling the residual functions with reference to the input layer by removing the unreferenced residual functions from learning (He et al., 2015) [15]. The sequence labelling module is based on the LSTM. And the last, decoding module, uses the novel method of training the recurrent neural networks which works on the existing neural network classifier and its principles (Graves et al., 2006) [14].

**Diagram

Description automatically generated**

Figure- CRAFT Architecture

**2.7 Working of OCR tools**

**2.7.1 OCRopus**

The simple working pipeline of OCRopus consists of 3 major steps. In the first step, the image from the dataset is converted to a black and white image in grayscale, because different kinds of images can have varied light-dark levels. This step is known as the binarization step. A type of adaptive thresholding is employed to perform this step, in which the light-dark cut-off can change over time. Moreover, images can have an off-scale layout, i.e., there can be a slight positive or negative inclination. The images are rotated by a small amount to make them horizontal because inclined images have more variance, leading to incorrect output and less recognition accuracy. This is better known as skew estimation.

In the second step, the output image from the previous step is given as an input to this step. Here, individual lines of text are extracted from the whole image. The median of the dimensions of the connected components is calculated from the input image. The outliers, that is, the components which vary a lot from the median size, are removed. The top and bottom of the remaining components are calculated to make them blur. This step in the pipeline is known as segmentation.

After completing the two steps in the pipeline, the third step, known as character recognition, is performed on the output of the second step. A network of LSTM is used to perform this step because it is the most complex step in the complete OCRopus pipeline. This step can become more difficult if the above steps make the input image less readable. The network takes individual columns as an input, taking the probable scores as the output. By completing all the steps, the text is extracted from the input image.

**2.8 Character Error Rate**

Character Error Rate is a heavily used performance metrics to calculate the machine translation systems majorly OCR.

Character Error Rate can be calculated as:

where:

|  |  |  |
| --- | --- | --- |
| Variable | Meaning | Example |
| Sc | Number of substituted characters in the output. | ‘123490’ gets extracted as ‘1Z349O’, which means, ‘2’ and ‘0’ gets substituted as ‘Z’ and ‘O’. |
| Dc | Number of deleted characters in the output. | String ‘ABC1234’ gets extracted as ‘AC123’. Here ‘C’ and ‘4’ are deleted and not present in the output. |
| Ic | Number of inserted characters in the output. | A series of characters ‘94127’ gets converted into ‘g41Z’ at the output . Here, one character is missing which needs to be inserted to compare with the ground truth. |
| Nc | Total number of characters in the ground truth excluding the inserted characters as initially they were not available. |  |
| Cc | Number of correctly extracted characters in the output. | Sum of all the correctly extracted characters. |

The percentage of characters in the ground truth that were predicted inaccurately in the OCR output text is the result of this equation. The performance of the OCR is justified by the value of the CER as, lower the value of CER, better is the performance of the tool.

It is common to see a CER value crossing the 100% mark when the reference text size is large but the mistakes in the output are larger, or majorly when the number of time the insertion operation performed is high. To bring the value of CER in the range of 0-100%, a normalisation technique is used. The number of mistakes can be divided by the summation of the number of number of correct characters and the number of edit operations.

The normalised equation can be stated as:

Another important consideration made while calculating the CER are the blanks or white spaces as they play a crucial role in separating the words and making a meaningful sentence. Example, mangoes and man goes have a completely different meaning. Generally multiple white spaces are considered as one during the calculation.

There are various theories that define different ranges of CER value for a good OCR output. These values vary according to the structure of the document, quality, type of content, writing style of the text etc. A common reference taken to resolve this conflict and to set a benchmark for the newspaper style dataset is taken from an article from the Australian Newspaper Digitization Program (Holley, 2008) [16]. These benchmarks were:

* Poor OCR accuracy: The character error rate > 10%
* Average OCR accuracy: 2% < character error rate < 10%
* Good OCR accuracy: 1% < character error rate < 2%

These benchmarks are not valid for hand written text as it is very difficult to get consistency and the content can be a combination of symbols and out to vocabulary words. A character error rate of 20% can be considered as a satisfactory level for hand written data set.

**2.9 Word Error Rate**

Word Error Rate is another most widely used performance metrics to measure the machine translation systems. A common problem in calculating the performance of the OCR systems is that the recognised word can have different word size when compared with the ground truth. Calculations involved in WER are based on the Levenshtein distance. This approach cannot be directly implemented because of one of the important drawback that it does not provide with in-depth conversion errors which makes it difficult to understand the required changes to get higher accuracy of the OCR tool. A better approach to use WER as the performance metrics is to align the words that are recognised in the order similar to the ground truth and then perform the calculations. Character Error Rate and Word Error Rate are calculated using the same formula with the difference that the WER is applied on words and CER get in more granular and is applied on the character level. Word level is generally used with the data that has meaning, like paragraphs in a newspaper.

Word Error Rate can be calculated as:

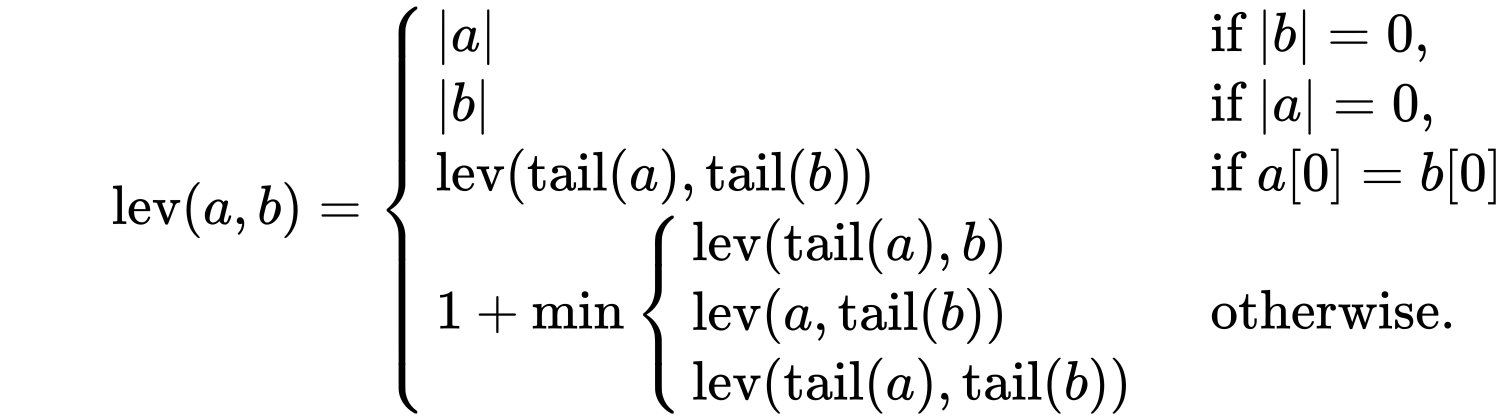
Where:

|  |  |  |
| --- | --- | --- |
| Variable | Meaning | Example |
| Sw | Number of substituted words in the output. | ‘Vitamin’ gets extracted as ‘Vltamin’, which means, ‘i’ gets substituted as ‘l’. |
| Dw | Number of deleted words in the output. | In sentence ‘World needs to control pollution’, ‘needs’ does not get extracted due to inefficient tool. |
| Iw | Number of inserted words in the output. | The word ‘Mangoes’ gets converted into ‘Man goes’ at the output. Here, the meaning of the word gets completely changed when compared with the ground truth. |
| Nw | Total number of words in the ground truth excluding the inserted words as initially they were not available. |  |
| Cw | Number of correctly extracted words in the output. | Sum of all the correctly extracted words. |

However, one disadvantage of adopting a general formula as stated above is that it ignores the impact that different sorts of faults may have on the likelihood of a successful conclusion, for example few errors can be heavier than others and few can be easy to remove. Moreover, it becomes difficult to segregate the errors by sum of deletion and insertion error from the substitution error. Thus, it is advised to derive a formula according to the problem to get the best accuracy calculations. For example, A new formula to calculate the word error rate was introduced by assigning different weights to the errors.

**2.10 Levenshtein distance**

Levenshtein distance is a string metric that is used to compare two string sequences. In other words it is the smallest number of required edits done on individual characters (insertions, deletions, or substitutions) to transform one word into the other. This distance is named after a Russian scientist and mathematician Vladimir Levenshtein, who used this distance metric in 1965 [17]. The distance computed between the two strings a,b having a length |a| and |b| can be given as:

****

where the tail of some string x is a string of all but the first character of x, and x[n] is the nth character of the string x, starting with character 0.

There are various upper and lower bounds for the Levenshtein distance which can be described as:

* For the two strings, it is at least the size difference between them.
* The highest distance should not be greater than the length of the longer string.
* If the length of the strings are equal, the distance is zero.
* The Levenshtein distance follows the triangle inequality law, that is, the sum of distance between the two strings should not be greater than the sum of Levenshtein distance from the third string.

**2.11 Confidence Value**

The error rates calculated help in predicting the confidence value for any OCR to evaluate its performance. These values are further used in industries to define a threshold value and to get an understanding of certain processes that can be automated end-to-end and those which require human interaction wherever needed throughout the process.

This can be easily understood by taking into consideration certain scenarios. For example, a pharmaceutical company involved in analysing the drug information from the label might need to introduce a lower threshold value so that it can introduce human interaction in the process for the granular verification of the data for the labels that qualify that threshold value. It is important to do this because incorrect information extraction from the drugs label can cost a lot of lives.

A second scenario can be considered where a loan application company needs OCR for the document processing task. It can set a medium confidence value threshold as human interaction will only be required to have an overall glance at the application.

**References**

[1] Hardesty, L. (2017). *Explained: Neural networks*. [online] MIT News. Available at: <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>.

[2] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), pp.386–408.

‌[3] Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4), pp.193–202.

‌[4] Rosenblatt, F. (1962). *Principles of neurodynamics; perceptrons and the theory of brain mechanisms.* Washingotn, D.C., Spartan Books.

‌[5] Kabrisky, M. (1966). *A proposed model for visual information processing in the human brain*. Urbana.

[6] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), pp.2278–2324.

[7] Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, [online] 79(8), pp.2554–2558. Available at: https://www.pnas.org/content/79/8/2554 [Accessed 1 Aug. 2019].

[8] Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), pp.1735–1780.

‌[9] Baek, Y., Lee, B., Han, D., Yun, S. and Lee, H. (2019). *Character Region Awareness for Text Detection*. [online] . Available at: https://arxiv.org/pdf/1904.01941.pdf [Accessed 12 Aug. 2021].

[10] Shi, B., Bai, X. and Yao, C. (2017). An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, [online] 39(11), pp.2298–2304. Available at: https://arxiv.org/pdf/1507.05717.pdf [Accessed 6 Apr. 2020].

[11] S, V. and A, S. (2015). Performance Comparison of OCR Tools. *International Journal of UbiComp*, 6(3), pp.19–30.

‌[12] Dhiman, S. and Singh, A. (2013). Tesseract Vs Gocr A Comparative Study. *International Journal of Recent Technology and Engineering (IJRTE)*, [online] (4), pp.2277–3878. Available at: https://www.ijrte.org/wp-content/uploads/papers/v2i4/D0788092413.pdf [Accessed 14 Aug. 2021].

[13] Heliński, M., Kmieciak, M. and Parkoła, T. (2012). *Report on the comparison of Tesseract and ABBYY FineReader OCR engines*. [online] . Available at: http://lib.psnc.pl/Content/358/PSNC\_Tesseract-FineReader-report.pdf.

‌[14] Graves, A., Ch, A., Fernández, S., Gomez, F., Schmidhuber, J. and Ch, J. (2006). *Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks*. [online] . Available at: https://www.cs.toronto.edu/~graves/icml\_2006.pdf.

‌[15] He, K., Zhang, X., Ren, S. and Sun, J. (2015). *Deep Residual Learning for Image Recognition*. [online] . Available at: https://arxiv.org/pdf/1512.03385.pdf.

‌[16] Holley, R. (2008). How Good Can It Get? Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs. [online] 15. Available at: http://www.nla.gov.au/ndp/project\_details/documents/ANDP\_HowGoodCanitGet.pdf.

‌

[17] B. And. Levenstein (1965). *В. И. Левенштейн, “Двоичные коды с исправлением выпадений, вставок и замещений символов”, Докл. АН СССР, 163:4 (1965), 845–848*. [online] www.mathnet.ru. Available at: http://www.mathnet.ru/links/765258e84f567bc0755d492983f5e1ba/dan31411.pdf [Accessed 17 Aug. 2021].

‌

‌

‌

‌

[1]

Z. Erlandsson, “IN DEGREE PROJECT COMPUTER SCIENCE AND ENGINEERING, SECOND CYCLE, 30 CREDITS,” 2019. Accessed: Jun. 21, 2021. [Online]. Available: https://kth.diva-portal.org/smash/get/diva2:1337270/FULLTEXT01.pdf.

[2]

C. Reul, C. Wick, U. Springmann, and F. Puppe, “Transfer Learning for OCRopus Model Training on Early Printed Books,” *027.7 Zeitschrift für Bibliothekskultur*, vol. 5, no. 1, pp. 38–51, Dec. 2017, doi: 10.12685/027.7-5-1-169.

[3]

S. S. Bukhari, A. Gupta, A. K. Tiwari, and A. Dengel, “High Performance Layout Analysis of Medieval European Document Images,” *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods*, 2018, doi: 10.5220/0006574603240331.

[4]

C. Patel, A. Patel, and D. Patel, “Optical Character Recognition by Open source OCR Tool Tesseract: A Case Study,” *International Journal of Computer Applications*, vol. 55, no. 10, pp. 50–56, Oct. 2012, doi: 10.5120/8794-2784.

[5]

R. Karpinski, D. Lohani, and A. Belaïd, “Metrics for Complete Evaluation of OCR Performance,” 2018. Accessed: Jun. 21, 2021. [Online]. Available: https://csce.ucmss.com/cr/books/2018/LFS/CSREA2018/IPC3481.pdf.

[6]

J. E. Tejaswini K, “Text Extraction from Images Using OCR,” *International Journal for Research in Applied Science and Engineering Technology*, vol. 8, no. 5, pp. 1805–1810, May 2020, doi: 10.22214/ijraset.2020.5291.

[7]

C. Wick, C. Reul, and F. Puppe, “Comparison of OCR Accuracy on Early Printed Books using the Open Source Engines Calamari and OCRopus,” 2018. Accessed: Jun. 21, 2021. [Online]. Available: https://jlcl.org/content/2-allissues/2-heft1-2018/jlcl\_2018-1\_4.pdf.

[8]

R. Smith, “An Overview of the Tesseract OCR Engine,.” Accessed: Oct. 15, 2020. [Online]. Available: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/33418.pdf.

[9]

R. W. Smith, “History of the Tesseract OCR engine: what worked and what didn’t,” *Document Recognition and Retrieval XX*, vol. 865802, no. SPIE 8658, Feb. 2013, doi: 10.1117/12.2010051.

[10]

S. Mori, C. Y. Suen, and K. Yamamoto, “Historical review of OCR research and development,” *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1029–1058, Jul. 1992, doi: 10.1109/5.156468.

[11]

C. Geetha, “Optical Character Recognition with Tesseract,” *JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES*, vol. 1, no. 2, Aug. 2019, doi: 10.26782/jmcms.spl.2019.08.00006.

[12]

N. Weichselbaumer, M. Seuret, S. Limbach, R. Dong, M. Burghardt, and V. Christlein, “New Approaches to OCR for Early Printed Books,” *DigItalia*, vol. 15, no. 2, pp. 74–87, Dec. 2020, doi: 10.36181/digitalia-00015.

**Appendix A**

**Risk Assessment**

This section of the project outlines the potential risks as well as the safeguards in place to reduce those risks. The potential dangers are grouped into four subcategories. It all starts with a qualitative risk assessment, which is used to evaluate the project's quality. Moving on to programming risk assessment, which takes into account all of the risks connected with coding. Timeline risk focuses on the risks associated with targets and the corresponding deadlines. Finally, dataset risk assessment examines the risks associated with the chosen dataset. If these risks are not addressed, they can lead to serious difficulties and incomplete research.

* **Qualitative Risk Assessment**
  + Description:

Qualitative risks are common in any type of project because there is no predefined matrix that can measure the value of this risk, and it consists of individual comments and perspectives. This master’s thesis involves a huge qualitative risk as:

* OCR is still a topic having wide dimensions of solutions and none of the solutions are perfect with an accuracy of close to 100%.
* There are numerous OCR tools on the market and individual’s perspective might differ depending on the software.
  + Mitigation:

All the official documentations are read carefully before starting the project, so that all the standards can be followed.

* **Programming Risk Assessment**
  + Description:

This category of risk plays a crucial role in projects involving heavy coding. Programming risks associated with this project are:

* The Python programming language is used to perform OCR in this project, which might not be the correct choice to perform OCR for the selected tools as they were initially written in different languages.
* Completing the pipelines involved in the project for both the tools with good accuracy can be difficult as they are long and complex.
  + Mitigation:
    - Because the software’s main architecture is based on python programming, it is an excellent choice for conducting research.
    - The pipelines involved are studied thoroughly from multiple sources/ researches before starting the project.
* **Timeline Risk Assessment**
  + - * Description:
        + These tools were introduced after a lot of research and even now, these are open-sourced, leading to a lot of innovation, and changes. This poses a risk of timely completion of the project as it needs to be submitted before a specific deadline.
      * Mitigation:
        + Before beginning the project, a weekly step by step chart was created to estimate the time time required for all tasks.
* **Dataset Risk Assessment**
  + - * Description:

The right choice of dataset is essential for the completion of this project as the OCR models need the correct dataset for testing. It is possible that the dataset chosen doesn’t work well during the testing part.

* + - * Mitigation:

Past researches using the dataset related to this project have been taken into consideration to get a better understanding of the dataset.

**Appendix B**

**Project Time-Plan**

* Week 1 (May 30 – June 5):
  + - * Research on Tesseract (History and use cases).
      * Explore related datasets.
* Week 2 (June 6 – June 12):
  + - * Research on OCRopus (History and use cases).
      * Explore related datasets.
* Week 3 (June 13 – June 19):
  + - * Work on Project Plan.
* Week 4 (June 20 – June 26):
  + - * Polish the Project Plan.
      * Divide the dataset according to the project requirements.
* Week 5 (June 27 – July 3):
  + - * Create pipeline for Tesseract.
      * Test the dataset selected.
* Week 6 (July 4 – July 10):
  + - * Create pipeline for OCRopus.
      * Test the dataset selected.
* Week 7 (July 11 – July 17):
  + - * Complete the Literature part for Tesseract.
* Week 8 (July 18 – July 24):
  + - * Complete the Literature part for OCRopus.
* Week 9 (July 25 – July 31):
  + - * Compare the outputs of both the models for various criteria.
* Week 10 (August 1 – August 7):
  + - * Write the working of both the tools.
* Week 11 (August 8 – August 14):
  + - * Write the outputs and comparison results.
* Week 12 (August 15 – August 21):
  + - * Complete the backlogs related to coding as well as writing.
* Week 13 (August 22 – August 28):
  + - * Write the final conclusion of the research.
      * List down the references.
* Week 14 (August 29 – Sept 4):
  + - * Write down the future work.
      * Prepare for oral presentation.
* Week 15 (Sept 5 – Sept 11):
  + - * Review the complete work and give final touches.