Offline handwriting recognition algorithms comparison based on efficiency and accuracy

Author: Vlad-Laurentiu Andreescu

Student ID: 6727817

1st Supervisor: Dr. Diana Hintea

2nd Supervisor: Dr. Matthew England

Bachelor of Computer Science

Coventry University

Faculty of Engineering, Environment and Computing

Abstract

In the last years, handwriting recognition became more and more popular and used. In the past handwriting recognition systems were used mainly by the big companies for reading and writing checks, validate signatures and reading postal cards. Nowadays, handwriting recognition is found from Health Care, where both patients and doctors can benefit from it, patients are not afraid because of the doctor’s sloppy handwriting, to automotive manufacturers and even in education. The objective of this thesis is to analyse and compare different handwriting recognition approaches based on accuracy, efficiency and adaptability. In the following chapters we are going to discuss about the background of handwriting recognition systems. Following we will analyse the Literature review in order to select only the best candidates into comparison. After that, the methodology will explain what the logic behind selected handwriting recognition approaches is. Following we will implement and tests the algorithm in order to respond to the thesis question: “Which provides the best offline handwriting recognition based on accuracy, efficiency and adaptability?”

In the final chapter we will covering project management conclusion and reflection upon this thesis.

Keywords: Handwriting, recognition, machine learning, neural network, algorithms;

Declaration of originality

I Declare that This project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: [illegible] Date: **12/04/2019**

First Name:	Vlad
Last Name:	Andreescu
Student ID number	6727817
Ethics Application Number	P88300
1st Supervisor Name	Dr. Diana Hintea
2nd Supervisor Name	Dr. Matthew England

Acknowledgements

I would like to thank Dr. Diana Hintea for her support, feedback and help she gave me during the project. She helped me find a direction for this project during meetings, her feedback helped me improve the project.

I would also like to thank Dr. Matthew England for his feedback during the presentation

Moreover, I would like to thank the persons who helped me build the dataset for Neural Network.

The responsibility for the final formulation and any errors that might concern are entirely mine.

Coventry University

Certificate of Ethical Approval

Applicant:

Vlad Andreescu

Project Title:

Which algorithm provides the most accurate recognition of offline handwriting?

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

26 February 2019

Project Reference Number:

P88300

Abbreviations

Handwriting Recognition Systems -> HRS

Optical Character Recognition -> OCR

Hidden Markov Models -> HMM

Intelligent Character Recognition -> ICR

Artificial Neural Network -> ANN

Convolutional Neural Network -> CNN

Recurrent Neural Network -> RNN

Connectionist Temporal Classification-> CTC

Rectified linear unit -> ReLU

Beam width -> BW

Figure -> Fig

Contents

List of Figures

1. Introduction

In the past, handwriting was the only way of communication, all historical documents we have left are handwritten and most of them indecipherable, handwriting recognition systems proved could be a useful allied to archaeologists in order to discover more about our history.

From reading postal addresses, bank's check amounts and forms to reading the text for blind people, handwriting recognition systems play a huge role in the society (MacKenzie and Tanaka-Ishii 2007). They translate handwriting sometimes sloppy, in digital words and sentences. A report from the National Academies of Science's Institute of Medicine (IOM) revealed that more than 7000 people are dying annually because of the doctor's sloppy handwriting. (van Leijen 2012).

Moreover, some handwriting recognition systems are using to detect counterfeit or illegal documents. As crime rate in the UK has increased, technologies like these should be always up to date with the current technology.

1.1 Background

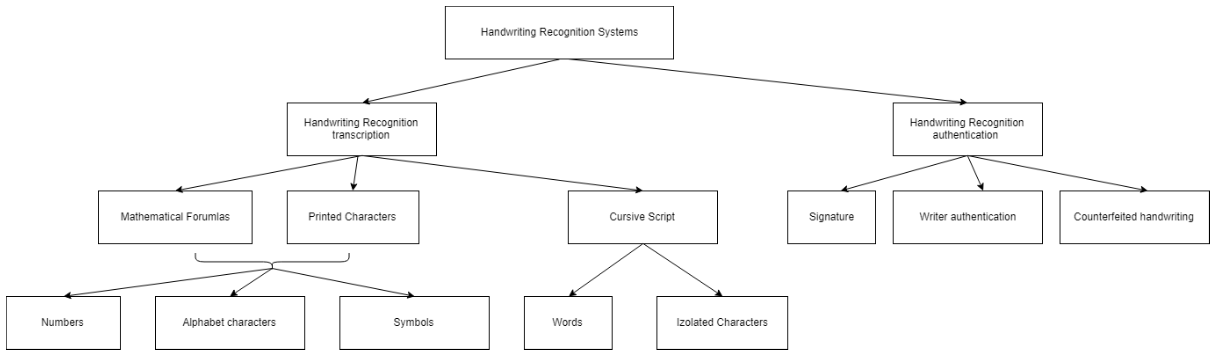


Fig. 1 Different Types of Handwriting Recognition Systems

The problem of handwriting recognition is well-studied by researchers since first computer was created. Optical Character Recognition (OCR) known to be the first handwriting recognition system was created in 1951 when M. Sheppard invented "GISMO" a robot capable of reading and writing (Verma, KULKARNI and BLUMENSTEIN 2008).

In 1967, big companies like IBM decided that OCR systems should be public for people's use. But they were very expensive, some of them only recognizing one font, therefore they could have been used only buy large companies.

As the problem of recognizing isolated character has been solved in the past, producing very good recognizing results, researchers are now focused on more advanced HRS which should recognize numbers, words and even sentences.

Different methods have been developed and applied by the researchers from simple segmentation algorithms to metaheuristic approaches using Genetic Algorithm.

Offline handwriting recognition systems will execute in four steps. First step is acquisition where the handwriting is scanned, and it is given as input for the following processes. Second step is segmentation where the handwriting is split into characters. Third step is recognition, during this step the characters are recognized using OCR algorithms. Final step is postprocessing, in this process, recognized characters are compared, using lexicons, with words in order to fix any error.

Before 2009, Hidden Markov Models techniques obtained the best handwriting recognition rates. HMM is based on probability and is using probabilistic tools for modelling sequential processes in order to recognize images containing handwritten words (Sueiras et al. 2018).

In 2009, a new method was introduced in this area of research: Deep Neural Network. Using recurrent neural network or feedforward neural network both containing bi-directional and multidimensional Long Short-Term Memory, the researchers have obtained the best recognition rates (Sueiras et al. 2018).

In 2015, Google released TensorFlow, a free and open source software library for developing and training machine learning models. It also contains some handwriting recognition algorithms (Vanilla Search, Best Path and Beam Search) (TensorFlow 2015).

1.2 Aims and Objectives

As no handwriting recognition algorithms comparison has been done in the past years since Neural Networks raised the bar in recognition rates, this thesis will aim to compare these algorithms not only by the accuracy (based on validation character error rate) of the recognition rates but also by the by the efficiency(time required to train and validate the dataset, difficulty of implementation) and adaptability(the ability to recognize characters from multiple datasets).

In order to achieve the aims we discussed before, we are going to set the main objectives that we have to accomplish:

1. The first objective is to obtain some datasets in order to train the handwriting recognition algorithms. Moreover, we are also going to use a personal dataset which will help during comparison.
   The dataset should contain images with handwritten words, characters and digits with different fonts, sizes and calligraphy.

2. Following, with the datasets gathered, I chose to implement a Neural Network using Google's TensorFlow and OpenCV using Python as programming language. Using this Neural Network, we will train and validate the models on different datasets. The testing phase will include the latest HR algorithms from TensorFlow libraries.
   I chose to use Neural Network as it is the most recent technology used by Handwriting Recognition Systems. Moreover, it is the only approach in which the hardware hardly influences the final results (having a slower PC might cause some efficiency problem).

3. The last objective is to analyse the results from the implemented algorithms and datasets. We will compare them with the results from different algorithms in order to have a response for this thesis' question.

1.3 Summary

So far, we introduced the problem of handwriting recognition and thesis’ aims and objectives. Following, we will organize the final thesis report five chapters. First chapter is Literature Review where we are going to analyse different studies and approaches of handwriting recognition systems. Second chapter is Methodology in which we are going to discuss the process of gathering the datasets, implementation of Neural Network and multiple algorithms pseudocodes. Third chapter comprises of testing and comparing the results in order to create a brief conclusion for this thesis. Forth chapter will refer to Project Management. The fifth and final chapter is the Conclusion where I summarize this thesis’ results and express my thoughts about it.

2. Literature Review

In this literature review we are going to examine research papers and documentation including methods and techniques of handwriting recognition systems, Neural Networks, Hidden Markov Models and also software and libraries used in HRS. This literature review will be organized chronologically, the reason is that this thesis will analyse how handwriting recognition technology evolved in time.
Handwriting analysis proved that handwriting is unique to every human, just as DNA, fingerprints, etc. There are several factors that determine the uniqueness of handwriting, such as the position in which you hold the pen, or the amount of pressure put on paper.

In the past few years, handwriting recognition received significant attention. This technology is used in diversified fields from healthcare to car manufactures and banks. Handwriting recognition is used to authenticate signatures or recognize different postcodes in letters. (P. Shankar Rao)

2.1 Online versus Offline handwriting recognition

Handwriting recognition is the process in which handwritten characters are identified. There are two concepts of handwriting recognition: online and offline.

Online Handwriting Recognition refers to the characters being identified in real time while they are being written on the surface (e.g. writing a letter on a touchscreen). For the other concept, Offline Handwriting Recognition, an image with a letter, digit or word is provided which needs to be recognized. (Kala R. et al. 2010)
For online handwriting recognition there is no need for a comparison as the current algorithms provide very good recognition rates. One of the best online handwriting recognition system is build by Google and implemented in GBoard (Google Keyboard). In order to recognize the word, it is converted in *Bézier* curves which are feed into the recurrent neural network. We could say that the Google's neural network is the most advanced approach not only because it generates great results, but it can recognize 82 different languages (Feuz and Gonnet 2019).

The offline approach reported a lower rate of recognition than the online approach, yet offline recognition technologies are efficient on domains where handwriting is highly used such as recognizing hand-written postal addresses on envelopes or reading bank checks therefore it has a huge economic impact. (P. Shankar Rao)

Because the word is not processed and recognized while it is written, it makes the process of recognizing more difficult. Because people have different handwriting styles, some offline handwriting recognition approaches such as neural network requires a huge training dataset.
My project will be focused on the second approach, Offline Handwriting Recognition.

2.2 Offline handwriting recognition approaches

There are several methods and algorithms used to develop and improve handwriting recognition. The most important ones will be discussed in this literature review and will be sorted chronologically. The first approach, developed around 1950, is Optical Character Recognition (OCR). The following approach is Binary Segmentation Algorithm. Third method proposed uses a metaheuristic algorithm, Graph Theory using Genetic Algorithm. Fifth approach will discuss about Hidden Markel Model. The sixth and final approach will analyse Neural Network approach using Recurrent Neural Network.

2.2.1 Optical Character Recognition

Optical Character Recognition, it is said it has been developed in the same period the first computer was build. But the developing of systems which use OCR started around 1951 with the invention of David Shepard, a device called '*Gismo*' which could read twenty-three characters from the alphabet and understand Morse Code

Optical Character Recognition is the process in which an image containing a handwritten word is processed and scanned than translated into digital text format.

Soumya Mishra, Debashish Nanda & Sanghamitra Mohanty (2010 December) mentioned that the implementation of an Optical Character Recognition is very difficult especially if it is developed to function under all possible conditions and gives highly accurate results. (D.P. et al., 2011).

The process of Optical Character Recognition includes two main elements. First element is scanning, during this this step, the system produces a digitalized version of the handwritten text and it is saved as a bitmap image file (Sheehan 2007). The second element is recognition in which the word is identified in the image blocks. Following, the word is split into characters which are recognized, then the word is identified using internal dictionaries, and correction is applied if needed. Finally, the word is saved in an editable format (Sheehan 2007).

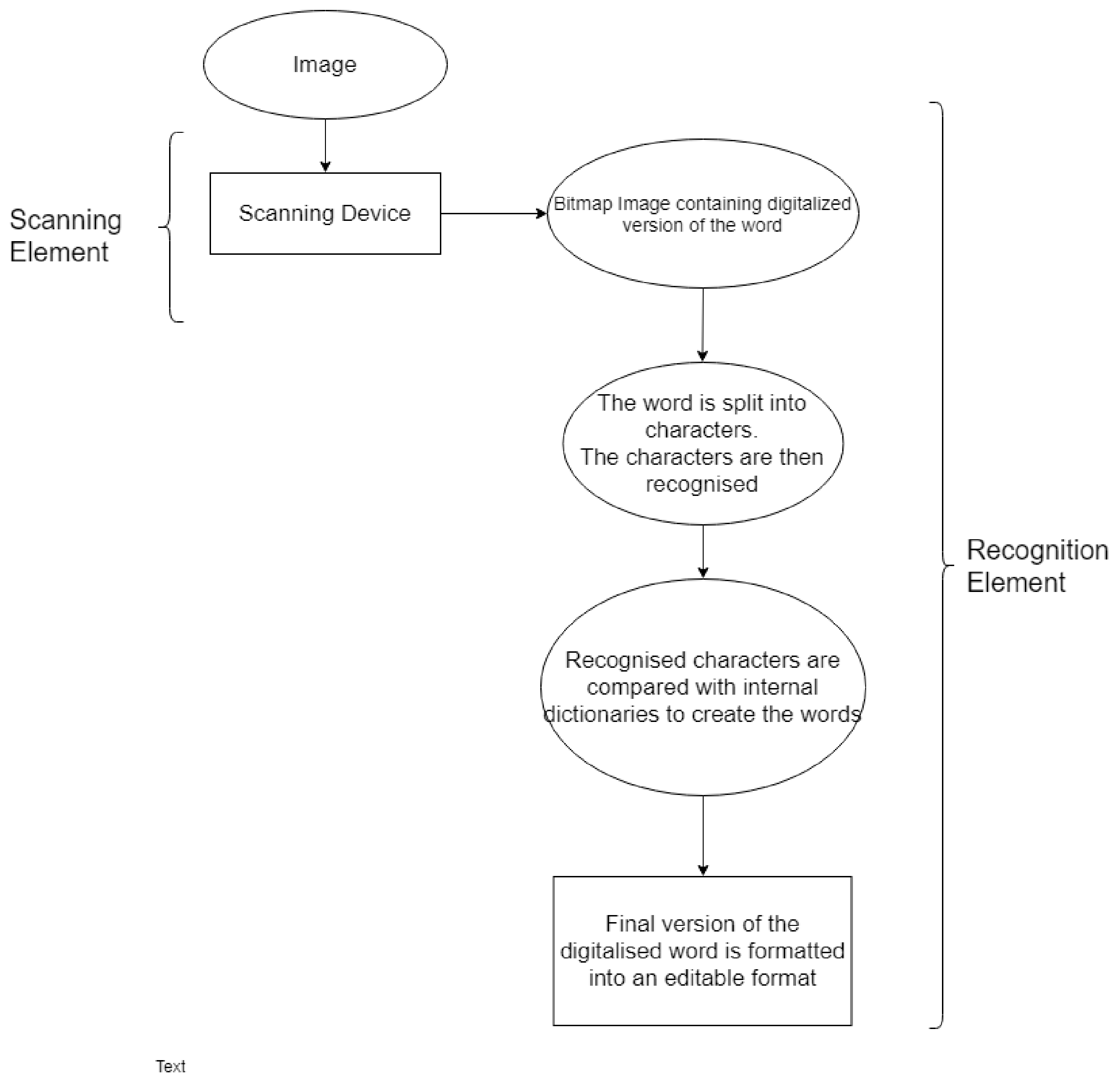


Image
Scanning Device
Scanning Element
Bitmap Image containing digitalized version of the word
The word is split into characters. The characters are then recognised
Recognised characters are compared with internal dictionaries to create the words
Final version of the digitalised word is formatted into an editable format
Recognition Element
Text

Fig. 2 OCR System functionality Diagram

When the OCR systems were released for the public, only big companies could afford them. And were used for productivity purposes only. US government has used Optical Character Recognition system in the mailing domain in order to digitalize and automate the mail handling process (Sheehan 2007).

The gap in Optical Character Recognition is that it is very sensitive to image noise. An image noise is calculated by the level of blur of an image. If the image is very blurry, it lowers the accuracy of the algorithm. Several denoising algorithms are used to reduce the noise and increase signal-to noise ration of an image. (D.P. et al., 2011)
First OCR systems could recognise one single font and only if it was properly scanned at a specific DPI. Modern OCR's can recognize any kind of font and they are available for any type of customer (form big companies to personal use). Although, they still struggle to recognize texts with poor quality (Verma, KULKARNI and BLUMENSTEIN 2008).

But as technology evolves, OCR systems were substitute with Intelligent Character Recognition (ICR) that uses pattern recognition and is based on the human brain learning model. It was proved that ICR systems were not very accurate and sometimes provided worse results than Optical Character Recognition systems. (Sheehan 2007).

In conclusion, Optical Character Recognition systems open the path of text recognition systems. Many researches and improvements have been made in Handwriting Recognition area using this approach.

2.2.2 Binary Segmentation Approach

The segmentation process in offline handwriting recognition has a huge effect on the recognition result generating performance.

By segmentation, a word is divided in multiple characters which represent the input for the recognition procedure. If the segmentation algorithm is not properly developed it could badly influence the recognition result. Therefore, some researchers skipped the segmentation step completely creating a "holistic" or segmentation-free approach (Lee and Verma, 2012). The holistic algorithm takes as an input the whole word and compare it with a dictionary.

Here the disagreement between researchers appears. In the English language there are 52 characters (lowercase and uppercase). In the segmentation approach the system must learn only 52 characters in order to be able to recognize a word. Contrarily, using holistic approach the system must learn all the English word which are more than 200,000. (Lee and Verma, 2012)

The principle of Binary Segmentation Algorithm (BSA) is to generate sub-images of the characters within a word until each generated image contains a known character by the system. There are two modules required by the algorithm to function properly: Suspicious Segmentation Points (SSPs) generation module and Segmentation Point (SP) which is selected from SSPs. Binary Segmentation Algorithm is applied to Segmentation Point which generate two sub-images. This process is repeated until each sub-image contains a character known by the system. (Lee and Verma, 2012)

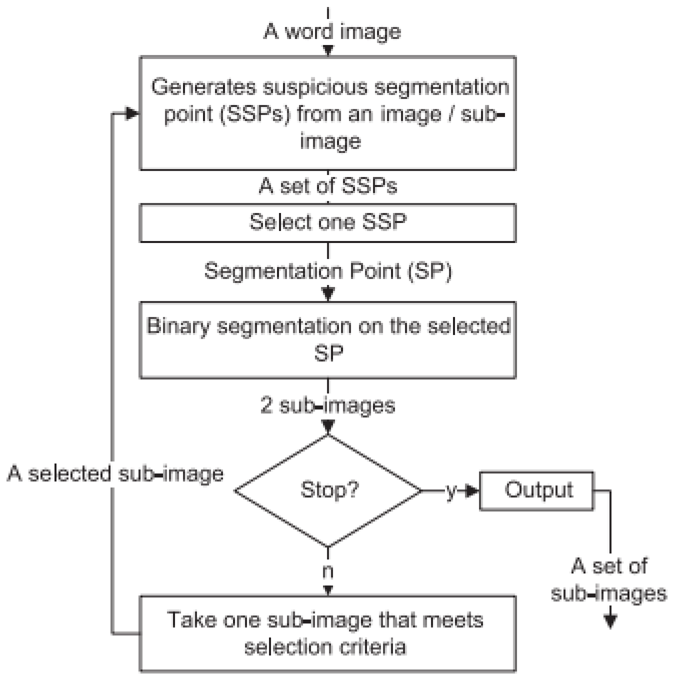

A word image
Generates suspicious segmentation point (SSPs) from an image / sub-image
A set of SSPs
Select one SSP
Segmentation Point (SP)
Binary segmentation on the selected SP
2 sub-images
A selected sub-image
Stop?
y
Output
n
A set of sub-images
Take one sub-image that meets selection criteria

Fig 3. Binary Segmentation Algorithm

This approach improves the handwriting recognition by lowering the number of characters that system must learn. The segmentation recognition results are calculate based on the number of characters successfully segmented and recognized, not the number of words recognized as other approaches might focus on. (Lee and Verma, 2012)
A gap in Segmentation approach could appear when the handwritten text is not in the best condition or it is very difficult to segment it.

In conclusion, Segmentation is one of the most important process in Handwriting Recognition systems because if it is not done correctly it will directly influence the final results.

2.2.3 Graph Theory using Genetic Algorithm

Graph Theory using Genetic Algorithms is a metaheuristic approach based on Artificial Neural Network. In order to train the system, for each input (image) a graph is created then a Genetic Algorithm is applied to mix the graphs and spot the most favourable solution.

The problem Kala R. et al. is trying to solve using this approach is the size of the dataset fed into the Neural Network. After the graphs have been created for each word in the dataset, the Genetic Algorithm is applied in order to mix and mutate the graphs which results in a bigger dataset that will provide a better handwriting recognition accuracy rate. Moreover, using this approach Kala R. et al. also solved the problem with the different handwriting styles. By combining the styles, it increases the algorithm probability to match the right text.

Graph Theory changed the problem of handwriting recognition to the problem of graph matching. Including Genetic Algorithm, the performance was highly improved.

The process of this approach follows the general handwriting recognition steps. First step is Segmentation in which the words in split into characters. It also provides information about boundaries of the characters. Second step is Pre-processing, during this step the image of the handwriting text is formatted in order to be fed into the recognition system (change the width of word's lines). Final step is Recognition in which the word is recognized using probability after the neural network was trained. The input for the recognition system is the processed image and the result is the digitally representation of the handwritten text.

Even if the dataset on which the ANN was trained contained only 69 characters from the English alphabet, the efficiency of this algorithm was 98.44% (Kala R. et al. 2010).

2.2.4 Hidden Markov Model

Hidden Markov Model approach was first developed for speech recognition. Modern HMM approaches have been successfully applied on image recognition systems. One of the first HMM implementation was developed by Kundu and Bhal in order to recognize Words from English dictionary. But the gap in their implementation was that the input (word) had to be very well segmented before the recognition process.

In order to improve the recognition accuracy, Kundu and Chen implemented an HMM system which uses an improved segmentation algorithm which split each character from the word into four parts. In this way, the features of characters can be seen and analyse more closely and precisely. This approach had a handwriting recognition accuracy of 74.5%

Current Hidden Markov Model systems have a recognition accuracy between 85% and 90% depending o the dataset size.

The HMM implementation presented by (Bunke, Roth and Schukat-Talamazzini n.d.) is focused more on image processing in order to extract as many features from the input image as possible. While the image is processed a graph is created for it, then the algorithm uses shape descriptors to calculate the high-level features from the graph's edges. (Bunke, Roth and Schukat-Talamazzini n.d.)
The results of this implementation showed a recognition accuracy of 98%. The algorithm was tested on two dictionaries both containing 150 words.

In conclusion, Hidden Markov Model approach was well designed and developed as it was one of the best approaches until 2009 when a new method was introduced: Deep Neural Network.

2.2.5 Neural Network using RNN and CTC

Artificial Neural Networks (ANN) require training the system with all characters before a result is processed. (Graves et al., 2008). After the ANN is trained, when it receives an input which might be a digit, letter or word, it compares the input with the learnt data set and chooses the most likely character based on generalization (Kala R. et al., 2010).

$$\mathrm{NN}: \underset{W\times H}{M} \rightarrow \underset{0\le n\le L}{(c_1, c_2, \ldots, c_n)}$$

Fig 4. Neural network formula of mapping an image to a set of characters

There are several Neural Network implementations in order to solve the handwriting recognition problem. The best results on word level recognition accuracy have been obtain by using five Convolutional Neural Network (CNN) layers, two Recurrent Neural Network (RNN) layers and one Connectionist Temporal Classification layer (CTC). CNN Layers take the image as input and extract all the features. RNN Layers transmit the important features along the layers. In order to obtain better training and validation results Long Short-Term Memory (LSTM) is used to implement RNN layers due to its ability to transmit the information on longer distances making the training more accurate. Final Layer is Connectionist Temporal Classification layer which takes as input the RNN matrix and the ground truth text (digitalized text of the image) in order to calculate the loss value during the Neural Network Training.

A gap in ANN is that if the discrepancy between the input and the learnt data set is large, the result might not be correct. Therefore, the data set needs to be well designed and big enough to cover all the possible character styles. (Kala R. et al., 2010).

2.3 Conclusion

From this literature review you can see that handwriting recognition is a well-studied process. There are many different approaches, some methods produce better results than others. Each algorithm has its own advantages and disadvantages. The aim of this research is to examine the algorithms and compare the results accuracy and the algorithms efficiency.

2.4 Handwriting recognition Software and Libraries

During this chapter we are going to discuss the main software and libraries used in order implement the algorithms and test the results.

2.4.1 Python

As this thesis' aims is to compare and find the best approach to solve the handwriting recognition problem, we are going to use Python 3.7 as our main programming language as it has great support for Artificial Neural Network development. We installed python through Anaconda because it comes with some useful libraries already installed such as numpy, matplotlib.

The editor we used is Visual Studio Code as it has a very well implemented linting extension called Python.

2.4.2 TensorFlow

TensorFlow is a free open source multi-programming language library implemented by Google in 2015. It offers multi-platform support for machine learning models developing. It is well documented and easy to install. (TensorFlow 2015) There are two ways of installing TensorFlow for Python:

1. If Python is installed through Anaconda, first important thing is to create a new environment for TensorFlow as we don't want to overload the root environment with all machine learning related libraries. Then, in the anaconda prompt type the following command: *conda install tensorflow=1.3* (we install version 1.3 as the newer 2.0 version is still in Alpha)

2. If Python was not installed through Anaconda, TensorFlow can be installed using pip command as following: *pip install tensorflow=1.3*

2.4.3 OpenCV

One important step in handwriting recognition systems is “Image Processing”. OpenCV is an open source library including computer vision and machine learning software.

Its main aim is to grant common infrastructure for computer vision and machine learning applications. Being open source, developers can use and modify its code.
The library contains more than 2500 computer vision and machine learning algorithms.

It has multi-programming language support (Python, C++, Java and MATLAB) and can operate on any operating system (Windows, Linux, Mac and Android) (Opencv.org, n.d.)

3. Methodology

In the literature review we explored different handwriting recognition systems developed since 1951. In this chapter, we are going to discuss and analyse the implementation theory of the handwriting recognition approaches which proved to provide similar results during experiments.

First, we will examine the datasets used for this comparison. Following, the implementation of the chosen neural network model will be explained. After that, we will analyse the meta-heuristic implementation of handwriting recognition system using Graph Theory and Genetic Algorithm developed by Kala R. et al.

3.1 Gathering the Datasets

In order to successfully complete this thesis' main objective, the comparison, we used three different datasets which were used to train the neural networks. The datasets that we used are: IAM Dataset, a dataset used by Kala R. et al. and a dataset I created myself.

3.1.1 IAM Dataset

IAM Handwriting Dataset contains more than 115,320 handwritten and labelled words as this was the official number in 2002. It was created in 1999 and it was purely created in order to train machine learning neural networks and perform handwriting validation and recognizing.

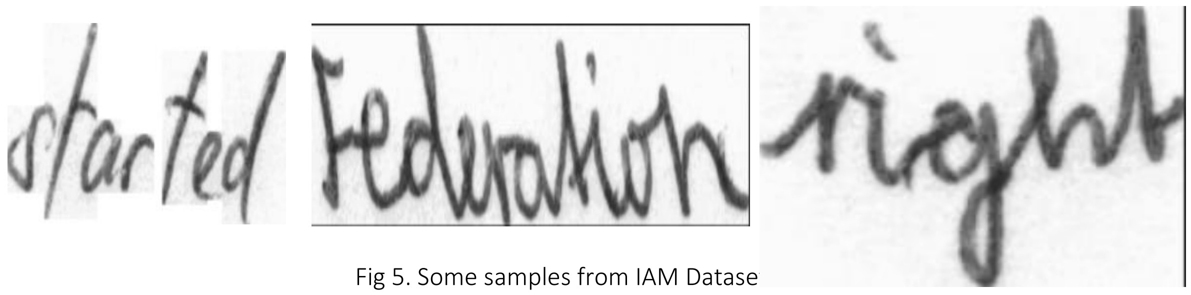


Fig 5. Some samples from IAM Datase

All images follow the same formatting, they were scanned at 300 DPI and saved as PNG files with a 256 gray scale.

3.1.2 Custom Dataset for meta-heuristic approach

This dataset was custom made based on the capital letters of English Language. Each character was handwritten twice and for some characters they include a shaking motion while writing so they can create diversity in the dataset. The final training dataset was consisted of 69 characters used for training.

3.1.3 Personal dataset

In order to expand the comparison process and to test the algorithms in under different circumstances. I asked two more students to help me and write some words on papers (around 400 each). Then I scanned the pages using university scanners at 300 DPI as it makes the picture more readable. Final step was cropping the words and assembly the dataset. After some cleaning we ended up with around 1000 words ready to be used in training.

In order to mimic Kala R. et al. dataset, we each wrote English alphabet with different styles in order to have a more robust comparison.

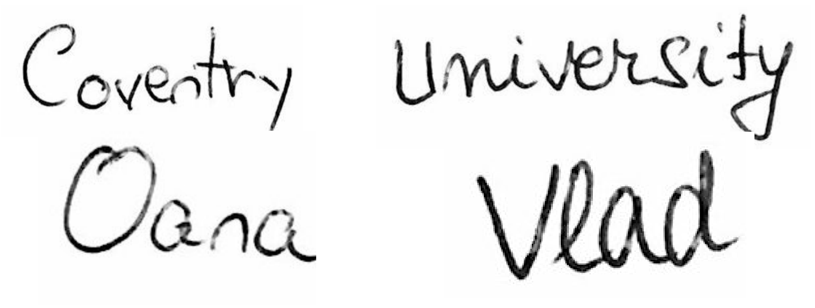


Fig 6. Samples from the custom dataset

There are some other datasets worth to be mentioned such as MINST dataset created by Jann LeCounm, Corinna Cortes and Christopher Burges. It contains handwritten digits and numbers, each image having size of 28x28 pixel square. (Learn Neural Networks n.d.)

3.2 Neural Network Approach

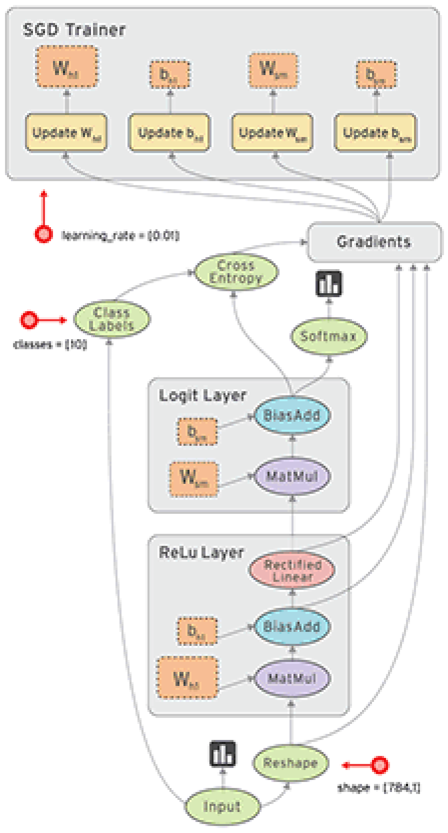


Fig 7. General Neural Network Schema

As this thesis objective in the comparison of handwriting recognition algorithms, for this approach we implemented the model initially created by Harald Scheidl. His model provided the highest recognition rate from the literature studied.
This handwriting recognition system is built on a neural network consisting in Convolutional Neural Network layers (CNNs), Recurrent Neural Network layers (RNN) and at the end one Connectionist Temporal Classification (CTC). (Scheidl 2018)

3.2.1 Convolutional Neural Network (CNN) layers

Each CNN layer contains three steps:

1. First step is the convolution operation, during this step first two layers apply a filter kernel with size 5x5 and the last three layers apply a 3x3 filter kernel.
2. Second step implies the application of non-linear RELU (rectified linear unit) function which is used as an activation function.
3. Last step is creating a pooling layer which encapsulate different regions of the image. An advantage of Neural Network is that it does not only learn the text of the image but also how the image looks like making it easier to recognize if the image was part of the training process (Scheidl 2018)

The output of these CNN layers is a feature map image of size 32x256. This output which represents a sequence of length 32, each time-step having 256 features. They are further fed into RNN for further processing. (Scheidl 2018)

3.2.2 Recurrent Neural Network (RNN) layers

Recurrent Neural Network layers receive the CNN's sequence as input. The RNN transmits the important information through the sequence by storing it in each RNN layer. This model uses Long Short-Term Memory (LSTM) implementation which helps RNN transmits the features information on longer distances and provide better training features than simple RNN's. With this in mind, the sequence is traversed starting from the front to back and again from back to end, creating two sequences of size 32x256. These sequences are later connected creating a sequence of size 32x512. (Scheidl 2018)

The output of RNN layers is represented by a 32x80 matrix which is transmit to the CTC layer.

3.2.3 Connectionist Temporal Classification (CTC) layer

CTC is the final layer of the Neural Network system. It has a double role in the system. During the training of Neural Network, it receives the RNN's output matrix and the ground truth (digitalized text of the current image processed in the batch) and calculates the difference between the learnt word and the original word, simply called loss value. During the validation process it receive only the RNN's matrix. Using decoding algorithms, CTC is able to output the final text. (Scheidl 2018)

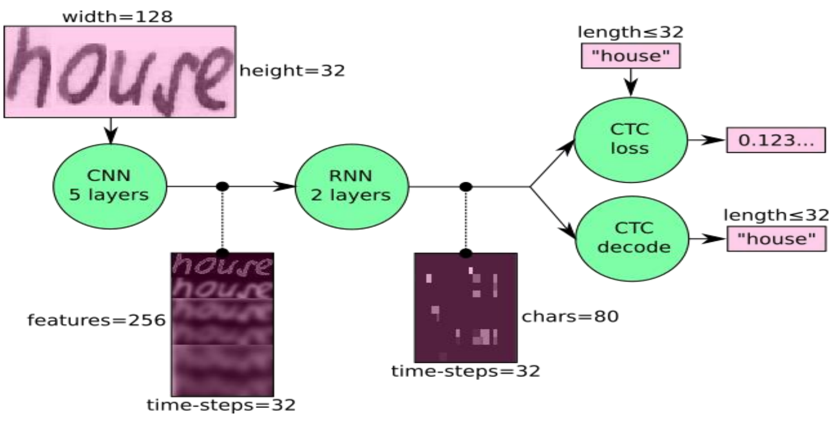


Fig. 8 Handwriting Recognition Neural Network Schema

3.2.4 Training the Neural Network

During the training process, the dataset is split into two parts. One part represents the training dataset and the second part represents the validation dataset with a ratio of 95% - 5% (depending on the dataset size).

The words from the training dataset are grouped into batches and fed into neural network. The model is training on multiple epochs. One epoch consists of training process where the batches are trained, and the loss value is calculated for each batch, and the validation process where a random number of images from the validation dataset are chosen in order to be analysed and recognized.
After one epoch is completed the model calculates the character error rate and word accuracy and checks if the current character error rate has improved since the last saved model.

The training process can take up to 24 hours if it is trained on the processor and far less time if it is trained on the graphics card. In order to check if the model is actually training we set a barrier: if the model has not been improved in five epochs the training should stop. But if the dataset is not as big as IAM dataset, removing the limit would be a good idea as the training process is very slow.

In order to complete this thesis's objective, we trained the neural network several times and modify its code to accept our personal dataset.

3.2.5 Decoding Algorithms

After the training process has been successfully completed. In order to validate an image, we have to pass the matrix, obtain from RNN layers to the CTC layer which will decode the matrix using one of the following decoding algorithms: best path decoding, vanilla beam search decoding and word beam search decoding. Following, we are going to discuss and analyse these algorithms and point any advantage and disadvantage that might exist.

3.2.5.1 Best Path decoding algorithm

Best Path decoding algorithm is included in CTC library from TensorFlow. It is a fast and resource efficient and requires two steps:

1. it recognizes the most appropriate character at each time-step in order to calculate the best path
2. after the word was roughly recognized, the algorithm removes the duplicate characters and all spaces in order to remain only the text. (Scheidl 2018)

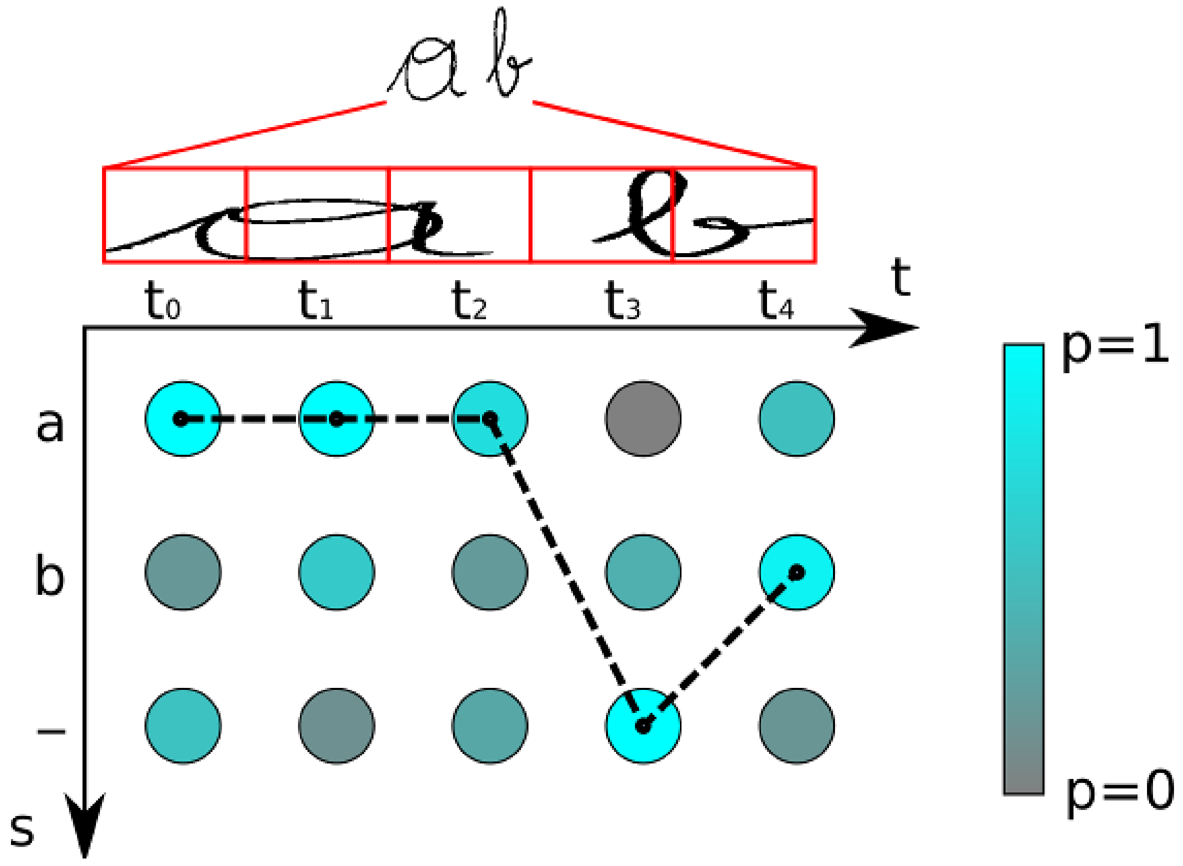


Fig 9. RNN output matrix being decoded by best path decoding (black dashed line)

The reason why we did not include Hidden Markov Model system in this comparison is that the algorithm behind HMM is very similar with best path decoding. Both algorithms operate under approximation and probability.

3.2.5.2 Vanilla Beam search algorithm

At each time-step this algorithm is creating text candidates (called beams) and score them.

Vanilla Beam search Pseudo-code:

t -> time step

b -> beam

```
Data: NN output matrix $mat$, $BW$
Result: decoded text
$beams = \{\varnothing\}$;
$scores(\varnothing, 0) = 1$;
for $t = 1 ... T$ do
    $bestBeams = bestBeams(beams, BW)$;
    $beams = \{\}$;
    for $b \in bestBeams$ do
        $beams = beams \cup b$;
        $scores(b, t) = calcScore(mat, b, t)$;
        for $c \in alphabet$ do
            $b' = b + c$;
            $scores(b', t) = calcScore(mat, b', t)$;
            $beams = beams \cup b'$;
        end
    end
end
return $bestBeams(beams, 1)$;
```

Fig 10. Vanilla Beam Searching algorithm (Scheidl 2018)

During each time-step, Vanilla Beam Search algorithm creates the beams and score them, only the beams with the best score is kept and further processed. The beam width (BW) set the maximum beams allowed to be stored. At the end after all time-steps, the best beam remained represents the digitalized version of the text. (Scheidl 2018)

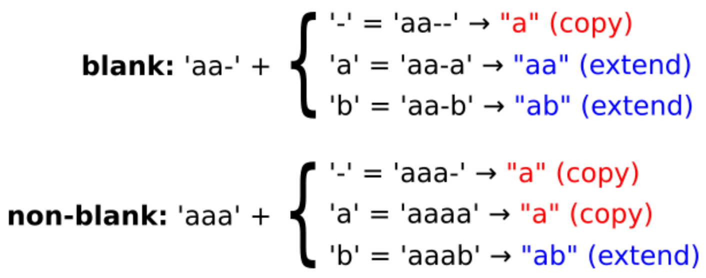
The beam scoring system is based on probability, it is separated into the score of paths ending with a blank space(“aa-”) and the score of paths ending with a non-blank(“aaa”). The algorithm calculates the probability of a beam to end with a blank and the probability of a beam to end with a non-blank. (Scheidl 2018)

The final score is calculated using the following formula:

$$\underline{P_{total}(b\,,\,t) = P_{b}(b\,,\,t) + P_{nb}(b\,,\,t)}$$

Where Pb is probability for a path to end with blank and Pnb is probability for a path to end with a non-blank.

During the process of recognition, some paths are extended. This can be caused only in three scenarios: extended by blank, extended by duplicating the previous character or extended by a new character. (Scheidl 2018)
If we break the extended paths in parts we observer that we either have unchanged beam (“a” -> “a”) or we have an extended beam (“a” ->”aa”). This information could be use as following: if the beam is extended we can figure out which paths will provide the best score. (Scheidl 2018)


blank: 'aa-' +
'-' = 'aa--' → "a" (copy)
'a' = 'aa-a' → "aa" (extend)
'b' = 'aa-b' → "ab" (extend)
non-blank: 'aaa' +
'-' = 'aaa-' → "a" (copy)
'a' = 'aaaa' → "a" (copy)
'b' = 'aaab' → "ab" (extend)
Fig 11. Extended paths on blank and non-blank scenarios (Scheidl 2018)

This algorithm can be further improved by adding a Character-level language model (LM) which will calculate not only the score for a single beam but a sequence of beams.

3.2.5.3 Word Beam search algorithm

Word Beam search algorithm is an extend version of Vanilla Beam search algorithm. The only difference is this algorithm will function differently if it recognizes a word or if the algorithm recognizes digits numbers or non-alphabetical characters. (Scheidl 2018)

After the algorithm has created the beam list, it adds a new state to each beam: it can be either in word state or non-word state.

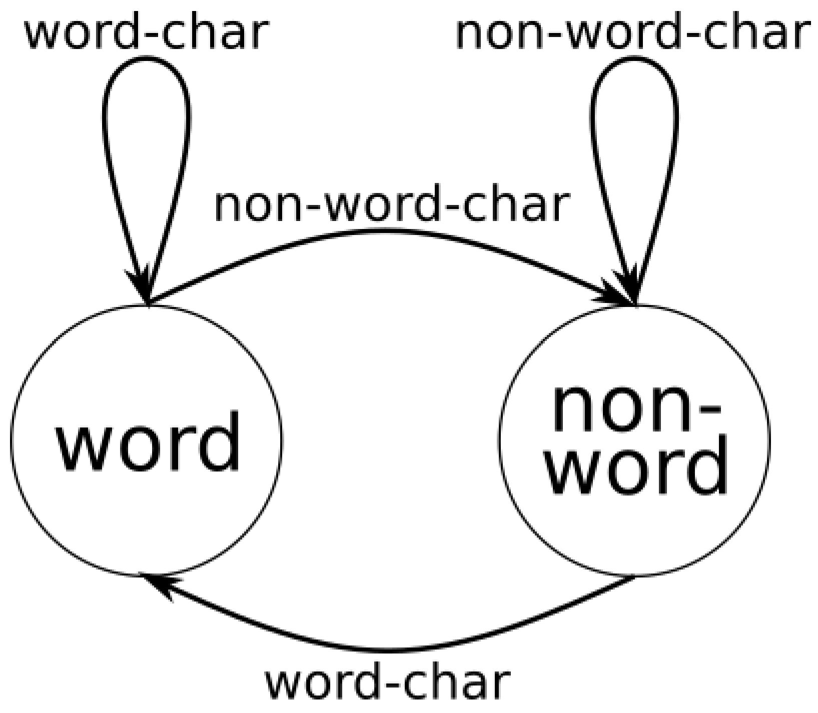


Fig 12. New state added for beams (Scheidl 2018)

When a beam is in word state, the algorithm only adds characters which will form new words. During this process, in each time-step we add a maximum of one character for each beam. As vanilla beam search algorithm only keeps the best beams at the end of each time-step using this new method will increase recognition accuracy because it only forms words contained in a dictionary. (Scheidl 2018)

3.3 Graph Theory using Genetic Algorithm

Kala R. et al. have a very interesting approach in solving the handwriting recognition problem. By applying Graph Theory, it switched from a handwriting recognition problem to a graph matching problem. Genetic Algorithm operates the best when it faces optimization problems.

The fitness function, in this scenario, compares the deviation of the graph of the ground truth text with the deviation of the graph of the unknown input. If the deviation is low, the result of the fitness function will be also low. The best-case scenario would be if the deviation is zero, meaning that the algorithm successfully recognized the word.

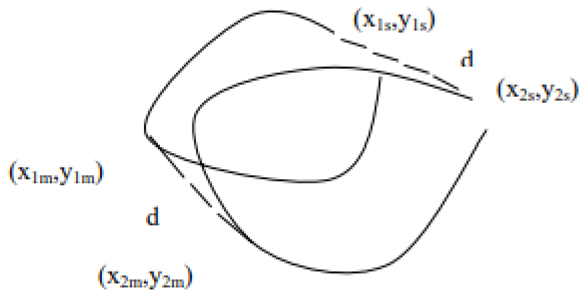


Fig 13. Deviation of Edges on two different graphs

Before being able to calculate the deviation of an entire graph, first the algorithm should calculate the deviation of each edge.

Following, the algorithm applies Crossover function which will mix two images in order to form new ones, even better than the previous versions. The crossover operation, in a graph problem, will be always applied between two vertices of the graph.

Before generating the Adjacency Matrix, the algorithm must match the points in the two graphs. Once the points have been matched, for this problem Kala R. et al. use adjacency matrix as graph representation. As this is an undirected and unweighted graph, the distance from x to y is the same length as the distance from y to x, as result the matrix will be symmetric.

Last step is graph generation. It converts the adjacency matrix into graphs represented as a list of edges.

Following I am going to present the pseudo-code as Kala R. et al. have written for the algorithm discussed above.

```
HandwritingRecognition(Language, TrainingData, InputImage)

Step1: For every character c in language
Step2:         For every input i for the character c in test data
Step3:                 Generate Graph gci of i
Step4: Generate graph t of input image
Step5: For every character c in language
Step6:         Use Genetic Algorithm to generate hybrid graphs
Step7: Return character corresponding to graph with the minimum most fitness function (out
of the graphs generated in any genetic operation)
```

The calculation of deviation between two graphs

```
Deviation(G1, G2)

Step1: dev <- 0
Step2: While G1 has no edges or G2 has no edges
Step3:         Find the edges e1 from first graph and e2 from second graph such that
        deviation between e1 and e2 is the minimum for any pair of e1 and e2
Step4:         Add its deviation to dev
Step5:         Remove e1 from first graph and e2 from second graph
Step6: For all edges e1 left in first graph
Step7:         Add deviation of e1 and null to dev
Step8: For all edges e2 left in second graph
Step9:         Add deviation of null and e2 to dev
Step10: Return dev
```

Crossover Function which will mix two graphs in order to find new handwriting styles

```
CrossOver(G1, G2)

Step1: match <- MatchPoints(G1, G2) (Find points in G1 corresponding to G2 and vice versa)
Step2: W1 <- GenerateAdjacency(G1) (Generates the Adjacency Matrix of the Graph)
Step3: W1 <- GenerateAdjacency(G2)
Step4: if W1 ≠ W2
Step5:          FindPaths(W1) (Finds the paths of all lengths between any pair of points)
Step6:          FindPaths(W2)
Step7:          p1 <- path between vertices v1 and v2 in G1 of length l, for all v1, v2 in W1 and length l = 1 or l = 2 or l = 3 or l = 4
Step8:          p2 <- path between vertices match[v1] and match[v2] in G2 of length l2=1 or l2= 2 or l2 = 3 or l2 = 4
Step9:          If p1 exists and p2 exists and p1 ≠ p2
Step10:                 Remove all edges from W1 that are found in p1
Step11:                 Add all edges in W1 that are found in p2
Step12:                 g <- MakeGraph(W1, W2) (Generate graph out of Adjacency matrix of W1)
Step13:                 Add g to solution set
Step14 else g <- MakeGraph(W1, W2)
Step15:                 Add g to solution set
```

Match points function

```
MatchPoints(G1, G2)

Step1: ver1 <- all unique points in first graph which are at least β units' distance apart from each other
Step2: ver2 <- all unique points in second graph which are at least β units' distance apart from each other
Step3: match <- null
Step4: While ver1 is not null or ver2 is not null
Step5:          match[s2] <- s1 such that s1 is a vertex in ver1 and s2 is a vertex in ver2 and distance between s1 and s2 is least for any combination of vertices in ver1 and ver2 and distance between s1 and s2 is less than 2ὺ units
Step6:          Remove s1 from ver1
Step7:          Remove s2 from ver2
Step8: For all vertex v in ver1
Step9:          match[v] <- v
Step10: For all vertex v in ver2
Step11:         match[v] <- v
```

Adjacency Matrix generation

```
GenerateAdjacency(G)

Step1: W <- 0
Step2: For all edges e in graph joining points i and j
Step3:          Wij <- Wij+1 (if e is a line)
Step4:          Wij <- Wij+2 (if e is a curve)
Step5:          Wji <- Wij
Step6: Return W
```

Last function represents the graph generation

```
GenerateGraph(W)

Step1: For all vertex i W
Step2           For all vertex j in W that come after or at i
Step3:                  If Wij > 2
Step4:                          add a curve from x to y
Step5:                          Wij <- Wij - 2
Step6:                  If Wij =1
Step7:                          add a line from x to y
```

3.4 Hardware and Software

In this section we are going to review the hardware and software used for this project

3.4.1 Hardware

We trained the Neural Network and validated the dataset on my personal desktop computer with the following specifications:

1. CPU: Intel i5 6600K overclocked at 4.00GHz

2. Graphics card: Nvidia GTX 1060 6 GB DDR5

3. RAM memory: 16 GB DDR4

These are the main components which can directly affect the training or validation process.

3.4.2 Software

Before implementing the neural network model, we had to install Python. We chose to use Anaconda as our Python framework as it comes with most of the libraries already installed, also the installation of other libraries is very easy, we discussed that in the literature review.

TensorFlow was installed so we can have access to any library needed for training the neural network

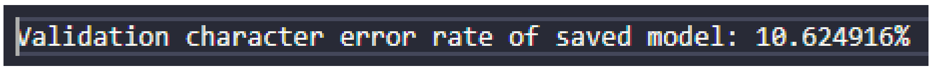
We used OpenCV purely for image processing in order to resize the picture and to thicken the text.

4. Experiments and Results

In this chapter we test the neural network with different datasets and different parameters in order to compare it from accuracy, efficiency and adaptability point of view.
This section will be split in parts as follows. First part will cover the experiments and results over a pre-trained model on IAM dataset. Second part will focus on the locally trained model on IAM dataset. In third section we will analyse the model trained on personal dataset. Forth section will be focusing on comparing the Neural Network results with Graph Theory using Genetic Algorithm Results.

4.1 Pre-Trained Model

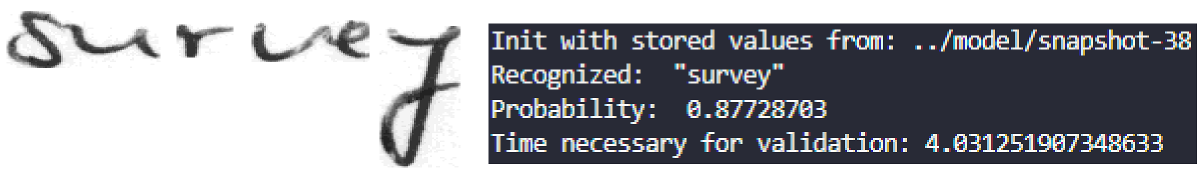
We used the model provided by Harald Scheidl which achieved a validation character error rate of 10.65%. The model was trained on IAM Dataset and validated with best path decoding algorithm.



```
Validation character error rate of saved model: 10.624916%
```

Fig 14. Validation Character error rate on pre-trained Model

We validated one photo from the trained dataset in 10 loops using best path decoding algorithm and the average recognition accuracy was 87%. We modified the code, so it can calculate the time necessary for validation. As we can see in Fig. 15 it takes 4.03 seconds to validate a character.

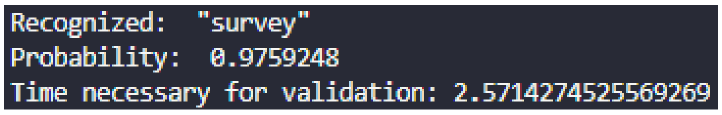


```
Init with stored values from: ../model/snapshot-38
Recognized:  "survey"
Probability:  0.87728703
Time necessary for validation: 4.031251907348633
```

Fig 15. Text to be recognized – the recognition process

In Fig. 16 we can observe the validation process and the results of validation over the trained Neural Network. The word accuracy on this model is 73.68%.

While we tested the validation process there was not difference on validating the word with Best Path decoding algorithm and Beam Search Algorithm. But when I validated the same picture as in Fig 15 but using Word Beam Search the results have drastically changed as we can see in Fig 16.

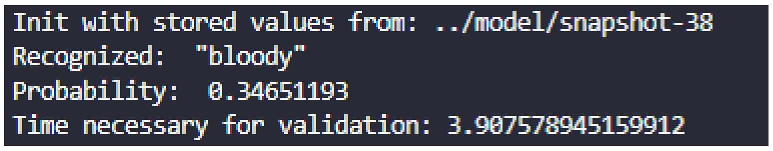


```
Recognized:  "survey"
Probability:  0.9759248
Time necessary for validation: 2.5714274525569269
```

Fig 16. IAM Dataset trained. Validated with Word Beam Search

We tested the algorithm with 250 pictures from the trained dataset and the accuracy was 100%. This means that the algorithm successfully recognized all words even if the probability was not higher than 93%.

The problem occurs when we try to validate an image that has not taken part in the training process. We can see in Fig 17 that the algorithm successfully recognized the word but with very low Probability: 0.34



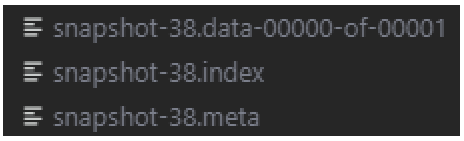
```
Init with stored values from: ../model/snapshot-38
Recognized:  "bloody"
Probability:  0.34651193
Time necessary for validation: 3.907578945159912
```

Fig 17. Validation process on the picture not trained in NN

The problem occurs when we try to validate an image that has not taken part in the training process. We can see in Fig 17 that the algorithm successfully recognized the word but with very low Probability: 0.34

4.2 Locally trained model on IAM Dataset

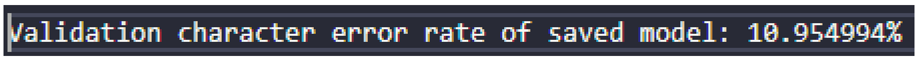
We trained the Neural Network with the IAM dataset over night. It started at 3:32:18 AM and it finished at 16:14:53 PM in the same day. After 53 epochs the training was stopped as it was not improving in 5 epochs.
The final model snapshot was at epoch number 38 as we can see in Fig. 18.



```
snapshot-38.data-00000-of-00001
snapshot-38.index
snapshot-38.meta
```

Fig 18. Last improved epoch

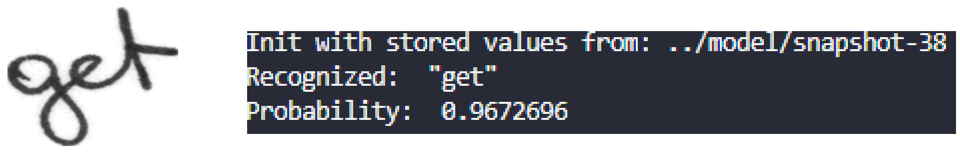
Our model has a validation character error rate 10.954994% which is similar with what Harald obtained on his model.



```
Validation character error rate of saved model: 10.954994%
```

Fig 19. Character error rate on locally trained model

We tested 250 images and the result was 100% recognition rate. The highest probability was obtained by the word "get" with 96% as we can observe from Fig 20.



```
Init with stored values from: ../model/snapshot-38
Recognized:  "get"
Probability:  0.9672696
```

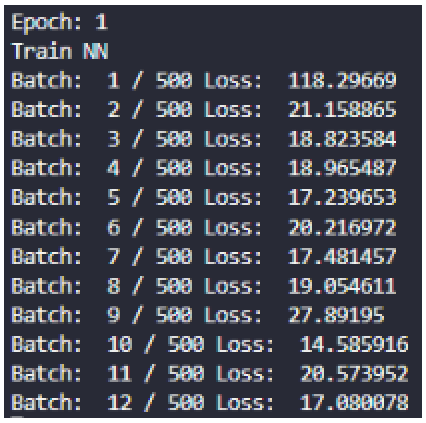
Fig 20. Highest recognition probability word

After the validation of entire dataset, our model achieved a word accuracy of 73.686957%.

So far, we could not compare the results because as expected they have similar results.

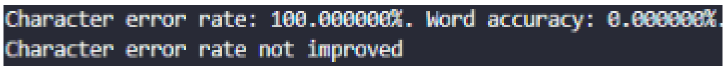
4.3 Locally trained model on personal dataset

After we created our dataset containing around 1000 personal pictures of handwritten text, we tried to train the neural network with the default values for splitting the dataset into two separate datasets: one dataset for training which consisted in 95% of the original dataset and another dataset for validation which contains the remaining 5%. The training has stopped after 5 epochs with no improvement.



```
Epoch: 1
Train NN
Batch:  1 / 500 Loss:  118.29669
Batch:  2 / 500 Loss:  21.158865
Batch:  3 / 500 Loss:  18.823584
Batch:  4 / 500 Loss:  18.965487
Batch:  5 / 500 Loss:  17.239653
Batch:  6 / 500 Loss:  20.216972
Batch:  7 / 500 Loss:  17.481457
Batch:  8 / 500 Loss:  19.054611
Batch:  9 / 500 Loss:  27.89195
Batch:  10 / 500 Loss:  14.585916
Batch:  11 / 500 Loss:  20.573952
Batch:  12 / 500 Loss:  17.080078
```

Fig 21. First attempt on training personal dataset

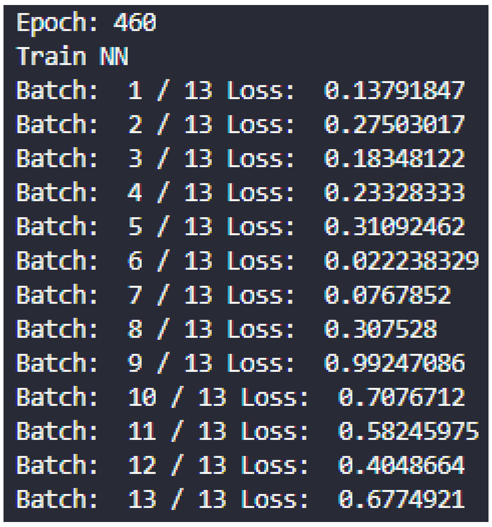


```
Character error rate: 100.000000%. Word accuracy: 0.000000%.
Character error rate not improved
```

Fig 22. Character error rate after 5 epochs

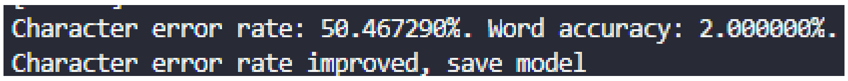
As we can observe from those two pictures above Fig 21 and Fig 22, the training dataset is too small to be split in 95% for training and 5% for validation as there will not remain enough pictures to validate the dataset correctly.

In Fig 23 and Fig 24 we removed the 5 epochs limit and analysed the model while training. We could have observed that the CTC loss has been reduced very close to 0 but the accuracy has not been improved, Moreover, the character error rate has been reduced by half while the word accuracy has improved only with 2%, this is happening because the validation dataset is too small.



```
Epoch: 460
Train NN
Batch:  1 / 13 Loss:  0.13791847
Batch:  2 / 13 Loss:  0.27503017
Batch:  3 / 13 Loss:  0.18348122
Batch:  4 / 13 Loss:  0.23328333
Batch:  5 / 13 Loss:  0.31092462
Batch:  6 / 13 Loss:  0.022238329
Batch:  7 / 13 Loss:  0.0767852
Batch:  8 / 13 Loss:  0.307528
Batch:  9 / 13 Loss:  0.99247086
Batch:  10 / 13 Loss:  0.7076712
Batch:  11 / 13 Loss:  0.58245975
Batch:  12 / 13 Loss:  0.4048664
Batch:  13 / 13 Loss:  0.6774921
```

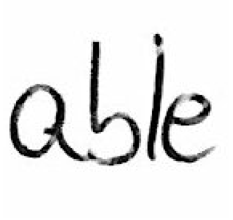
Fig 23. Lowest CTC loss after 460 epochs

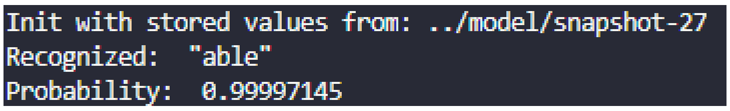


```
Character error rate: 50.467290%. Word accuracy: 2.000000%.
Character error rate improved, save model
```

Fig 24. Character error rate and word accuracy after 460 epochs

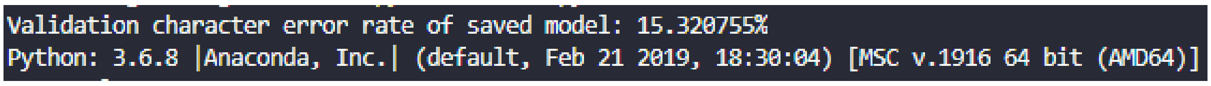
We solved this problem by splitting the dataset from 95%-5% in 80%-20% and as it can be seen in Fig 25, the word "able" was correctly recognize with a record probability of 99.99%. Moreover, being way smaller then IAM dataset, our dataset reached the minimum validation character error rate of 15.32% (Fig 26) in fewer epochs than IAM dataset (27 epochs versus 38 epochs).





```
Init with stored values from: ../model/snapshot-27
Recognized:  "able"
Probability:  0.99997145
```

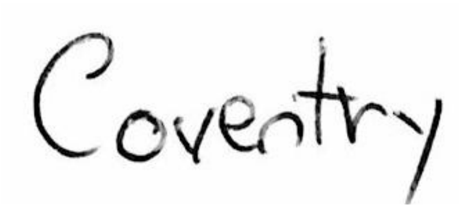
Fig 25. Probability record on personal Dataset

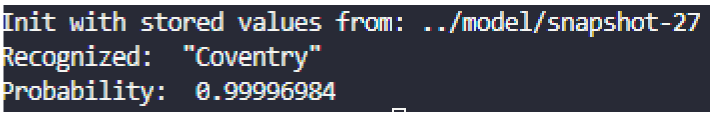


```
Validation character error rate of saved model: 15.320755%
Python: 3.6.8 |Anaconda, Inc.| (default, Feb 21 2019, 18:30:04) [MSC v.1916 64 bit (AMD64)]
```

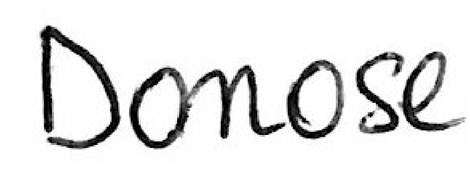
Fig26. Validation character error rate for personal dataset

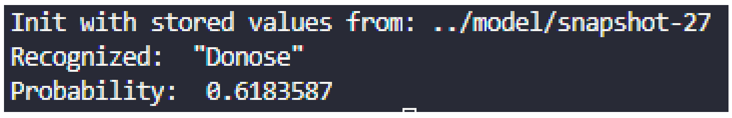
We solved this problem by splitting the dataset from 95%-5% in 80%-20% and as it can be seen in Fig 25, the word “able” was correctly recognize with a record probability of 99.99%. Moreover, being way smaller then IAM dataset, our dataset reached the minimum validation character error rate of 15.32% (Fig 26) in fewer epochs than IAM dataset (27 epochs versus 38 epochs).
We tested the Neural Network with 350 images and the algorithm successfully recognized 347 images rate was 99.14% with the highest probability being 99.99%. I attached some personalized pictures from my dataset with different handwriting in Fig 27 which were correctly validated and in Fig 28 some photos which were wrongly recognized. (from me and two other students who helped me designing the dataset).





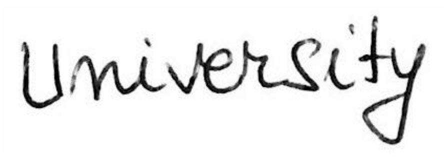
```
Init with stored values from: ../model/snapshot-27
Recognized:  "Coventry"
Probability:  0.99996984
```

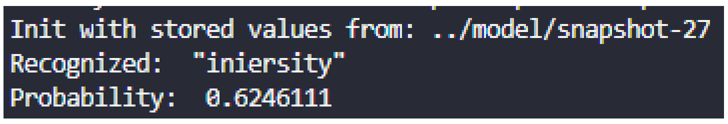




```
Init with stored values from: ../model/snapshot-27
Recognized:  "Donose"
Probability:  0.6183587
```

Fig 27. Successful validation on personalized photos from the dataset





```
Init with stored values from: ../model/snapshot-27
Recognized:  "iniersity"
Probability:  0.6246111
```

Fig 28. Wrong validation on personalized photos from the dataset

5. Conclusion

In this section we will summarize the pervious chapter Experiments and Results and we are going to answer this thesis' question: Which algorithm provides the best offline handwriting recognition based on efficiency, adaptability and accuracy.

But first I want to explain why I compared only two approaches: Neural Network implementation and Graph Theory using Genetic Algorithm. I chose these two approaches as they provided the best results as we pointed in Literature Review.

Graph Theory using Genetic Algorithm was trained on 69 images of alphabetical characters and tested on 385 images and the results shown that only 379 characters have been successfully recognized having a handwriting recognition rate of 98.44%. While, on the neural network using RNN and CTC approach, I used as training my dataset containing around 1000 images of full words (not characters) and it was tested on 350 images, the results were a little better, 347 images have been correctly recognized, unveiling a handwriting recognition rate of 99.14%.

As efficiency, both approaches are using Neural Network which takes time during the training (anywhere between 12 and 48 hours depending on the dataset size). I was able to calculate the time need for each decoding algorithm to validate an image and the results are as follow:

1. Best Path decoding algorithm: 4.03 seconds
2. Vanilla Beam Search decoding algorithm: 4.02 seconds
3. Word Beam Search decoding algorithm: 2.57 seconds

Word Beam Search decoding algorithm is by far the fastest algorithm included in CTC library from TensorFlow. We could not compare the results with Kala R. et al. and their approach as there was not any time efficiency specified.

We could see that the Neural Network implemented with RNN and CTC was able to recognize images outside the training dataset even with lower Probability.

Handwriting Recognition Systems	Neural Network using RNN and CTC	Graph Theory using Genetic Algorithm
accuracy	✓	
efficiency	✓	
adaptability	✓	

In conclusion, the answer to this thesis' questions: "which algorithm provides the best offline handwriting recognition based on efficiency, adaptability and accuracy?" is Neural Network using Recurrent Neural Network and Connectionist Temporal Classification

6. Project Management

In this section we are going to discuss the methods and techniques we applied in order to start and finish this project. First, we are going to analyse the version control system I used then how I planned and managed my time and finally we will discuss about the meetings and feedback.

6.1 Version Control

In order to keep every aspect under control while working on this project, we used GitHub as our version control. After creating the repository, we implemented GitFlow branching model, not because multiple people worked on this project or the project has too many features. The reason we chose to implement GitFlow is because we were training multiple Neural Networks models and we wanted to have a branch for each of them, this is also the reason why we did not merge everything back to master.

Unfortunately, the model trained, and the datasets were too big to be uploaded on GitHub, but we will attach links to Google Drive where you can download them if needed.

GitHub Link: https://github.coventry.ac.uk/andreesv/303COM_andreesv_HandwritingRecognition

Already Trained Model:
https://drive.google.com/file/d/1HVZAo6621xQeOec8VS6e06yLi1sLRVmC/view?usp=sharing

IAM Words dataset:
https://drive.google.com/file/d/1vopczcxTsFqRX2971wDfcTj_qHb41r1k/view?usp=sharing

Words ground truth:

https://drive.google.com/file/d/1JMAiqlrBgE1qmQVMaoLV58iKoGMTuZ8k/view?usp=sharing

As model and dataset were too big to upload on GitHub, in case you want to test the algorithm, I uploaded on google drive The IAM dataset (unrar the zip file intro /data/words folder of the project), the ground truth of the dataset (place words.txt into /data) and the model (if you want to train the neural network, delete everything inside /model, go to /src and run "*python ./main.py –train*" or copy my model into /model folder and run "*python ./main.py –validate" to validate the entire dataset or just "python ./main.py"* to validate the sample test.png inside /data folder. If there is not any test.png file you can copy a random image from dataset and rename it *"test.png".*

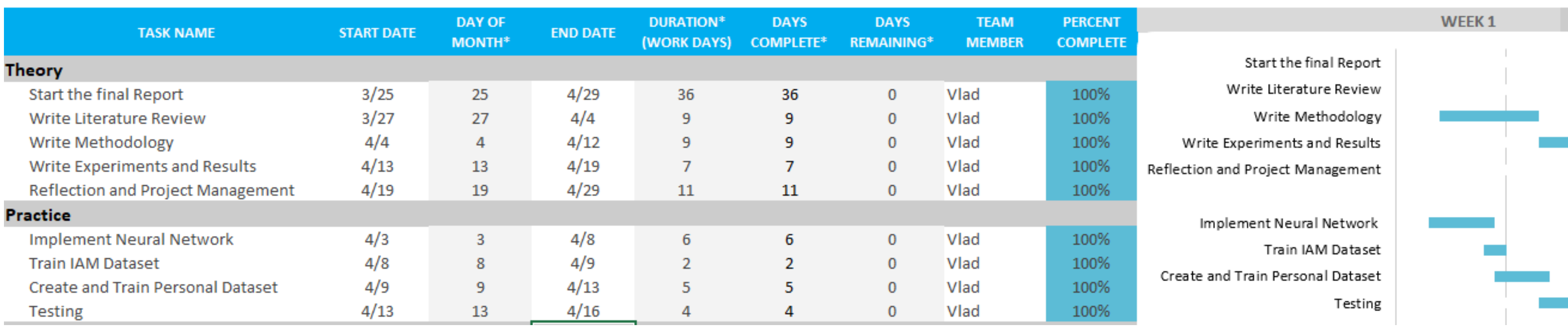
If you want to test train the algorithm using my dataset, please download the following rar which contains the dataset in "words" folder and the ground truth text in "words.txt":

https://drive.google.com/file/d/1xGkDw2JXBUjRFNtMJABxnT3mtZJbNIsb/view?usp=sharing

After the download was completed, copy both the file "words.txt" and the folder "words" into the /data folder in the project root.

6.2 Time Management

Because we wanted to have a good time management and to be every time on schedule we have created a Gantt Chart, so we know how much time to spend o a task in order to stay on track.



TASK NAME	START DATE	DAY OF MONTH*	END DATE	DURATION* (WORK DAYS)	DAYS COMPLETE*	DAYS REMAINING*	TEAM MEMBER	PERCENT COMPLETE
Theory								
Start the final Report	3/25	25	4/29	36	36	0	Vlad	100%
Write Literature Review	3/27	27	4/4	9	9	0	Vlad	100%
Write Methodology	4/4	4	4/12	9	9	0	Vlad	100%
Write Experiments and Results	4/13	13	4/19	7	7	0	Vlad	100%
Reflection and Project Management	4/19	19	4/29	11	11	0	Vlad	100%
Practice								
Implement Neural Network	4/3	3	4/8	6	6	0	Vlad	100%
Train IAM Dataset	4/8	8	4/9	2	2	0	Vlad	100%
Create and Train Personal Dataset	4/9	9	4/13	5	5	0	Vlad	100%
Testing	4/13	13	4/16	4	4	0	Vlad	100%

Fig 29. Gantt Chart

6.3 Meetings and Feedback

I had several meetings with my tutor where we discussed about the aims and objectives that my project should cover. My tutor helped me find a better name for my thesis and also supported me by providing feedback on different key tasks of this project such as the proposal and presentation.

During the mid-period presentation, I was lucky that I was the only one in my group that had second tutor's attention during the presentation.

Diana, I tried to be as critical as possible during the Literature review, highlighting some of the major gaps in some handwriting recognition approaches, I included risk assessment and Reflection in my report as you suggested. Moreover, I took your advice and paid more attention to referencing and bibliography.

Matthew, after you gave me the feedback, I had to rethink most of my thesis as I wanted to follow your suggestion and be as explicit as possible in my theory, explaining every decision I make and every path I take.

In the end, I would like to thank you both for your attention feedback and support.

6.4 Risk Assessment

As our project does not influence other people lives or business is worth looking at the risks that might be happening inside our project:

1. **Poor dataset**. In this scenario we are focusing on the dataset which is not big enough and we cannot split it into training dataset and validation dataset. This would result in an inefficient machine learning model.

2. **Wrong text recognized.** This scenario might be caused by several factors. First factor would be that the image fed in for recognition was not part of the training dataset. Second factor can occur if the text is larger than the maximum number of characters allowed by the Neural Network (increasing NN size will improve the recognition accuracy).

3. **Model does not train.** This problem is usually caused by bugs but there are scenarios where the code is perfect, but the dataset is not big enough (point 1).

4. **Damaged image in dataset.** This problem does not occur when the dataset personally created but for example in IAM dataset there are several hundreds of bad pictures such as the image is very small, the image does not contain the whole word, the image is not well scanned etc.

7. Reflection

In my opinion, I believe that this thesis has completed its objective and has answered its question: “Which algorithm provides the best offline handwriting recognition based on accuracy, efficiency and adaptability?”.

From the literature review, we select only the algorithms that proved to obtain the best algorithm. Following with Methodology we explain the logic behind these algorithms and their implementation. Both algorithms are based on Neural Network which is proved to be hard to master due to the number of variables that should be taken in consideration. From the Experiments and Results section, we discover how close those approaches are in terms of accuracy (98% for Graph Theory using Genetic Algorithm and 99% for Neural Network using Recurrent Neural Network (RNN) layers and Connectionist Temporal Classification (CTC) layer). But practically, Graph Theory using Genetic Algorithm is more difficult to implement than Neural Network approach.

As the technology is continuously growing, new methods and techniques solving this problem will be implemented, maybe faster, maybe more accurate. This thesis is a good start, it pointed out which algorithms are worth taking in consideration and which are not.

For future research I was thinking about combining both approaches using a Neural Network with Recurrent Neural Network Layer but instead of CTC layer to use Genetic Algorithm which can be applied to mutate and crossover some the features created By RNN layers.

8. Bibliography

1. Bunke, H., Roth, M. and Schukat-Talamazzini, E. (n.d.). *Off-line Cursive Handwriting Recognition using Hidden Markov Models*. [online] Pdfs.semanticscholar.org. Available at: https://pdfs.semanticscholar.org/bcc5/90498a13d19ee94c811b2c6ef41b52263f6b.pdf [Accessed 23 Apr. 2019].

2. D.P., G., GUNGE, Y., MUNDADA, R., BHARADWAJ, H. and PATIL, S. (2011). USER DEPENDENT HANDWRITING RECOGNITION USING OFF-LINE OCR FOR ARITHMETIC OPERATION AND TEXT PROCESSING. *i-Manager's Journal on Software Engineering*, [online] 6(1), p.9. Available at: https://search.proquest.com/docview/1473909263?accountid=10286&rfr_id=info%3Axri%2Fsid%3Aprimo [Accessed 28 Jan. 2019].

3. Feuz, S. and Gonnet, P. (2019). *RNN-Based Handwriting Recognition in Gboard*. [online] Google AI Blog. Available at: https://ai.googleblog.com/2019/03/rnn-based-handwriting-recognition-in.html [Accessed 23 Apr. 2019].

4. Graves, A., Fernandez, S., Liwicki, M., Bunke, H. and Schmidhuber, J. (2008). *Unconstrained Online Handwriting Recognition with Recurrent Neural Networks*. [online] People.idsia.ch. Available at: http://people.idsia.ch/~juergen/nips_2008.pdf [Accessed 1 Feb. 2019].

5. KALA, R., VAZIRANI, H., SHUKLA, A. and TIWARI, R. (2010). Offline Handwriting Recognition using Genetic Algorithm. *International Journal of Computer Science Issues (IJCSI)*, [online] 7(2), p.11. Available at: https://search.proquest.com/docview/223065287/fulltextPDF/FD3B7FE0657C49B7PQ/1?accountid=10286 [Accessed 26 Jan. 2019].

6. Learn Neural Networks. (n.d.). *Handwriting recognition by using multilayer perceptron | | Learn Neural Networks*. [online] Available at: http://learn-neural-networks.com/handwriting-recognition-using-multilayer-perceptron/ [Accessed 15 Apr. 2019].

7. Lee, H. and Verma, B. (2012). Binary segmentation algorithm for English cursive handwriting recognition. *Elsevier*, [online] 45(4), p.12. Available at: https://www.sciencedirect.com/science/article/pii/S0031320311003943 [Accessed 1 Feb. 2019].

8. MacKenzie, I. and Tanaka-Ishii, K. (2007). *Text Entry Systems*. Elsevier Science & Technology.

9. Opencv.org. (n.d.). *OpenCV*. [online] Available at: https://opencv.org/about/ [Accessed 19 Apr. 2019].

10. P. Shankar, R. and J., A. (n.d.). *Handwriting Recognition – "Offline" Approach*. [ebook] p.4. Available at: https://cs.stanford.edu/people/adityaj/HandwritingRecognition.pdf [Accessed 27 Jan. 2019].

11. Scheidl, H. (2018). *An Intuitive Explanation of Connectionist Temporal Classification*. [online] Towards Data Science. Available at: https://towardsdatascience.com/intuitively-understanding-connectionist-temporal-classification-3797e43a86c [Accessed 14 Apr. 2019].

12. Scheidl, H. (2018). *Build a Handwritten Text Recognition System using TensorFlow*. [online] Towards Data Science. Available at: https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5 [Accessed 11 Apr. 2019].

13. Scheidl, H. (2018). *Word Beam Search: A CTC Decoding Algorithm*. [online] Towards Data Science. Available at: https://towardsdatascience.com/word-beam-search-a-ctc-decoding-algorithm-b051d28f3d2e [Accessed 15 Apr. 2019].

14. Sheehan, E. (2007). *Optical Character Recognition*. [online] Www-bcf.usc.edu. Available at: http://www-bcf.usc.edu/~wdutton/comm533/OCR-SHEE.htm [Accessed 23 Apr. 2019].

15. Sueiras, J., Ruiz, V., Sanchez, A. and Velez, J. (2018). *Offline continuous handwriting recognition using sequence to sequence neural networks*. [online] sciencedirect. Available at: https://www.sciencedirect.com/science/article/pii/S0925231218301371 [Accessed 23 Apr. 2019].

16. TensorFlow. (2015). *TensorFlow Core | TensorFlow*. [online] Available at: https://www.tensorflow.org/overview/ [Accessed 23 Apr. 2019].

17. van Leijen, M. (2012). *Death by prescription: Doctors' handwriting causes 7,000 deaths a year*. [online] Emirates24|7. Available at: https://www.emirates247.com/news/emirates/death-by-prescription-doctors-handwriting-causes-7-000-deaths-a-year-2012-11-04-1.481418 [Accessed 3 Feb. 2019].

9. Appendix

9.1 Records of supervisor meetings

Supervisor: Dr Diana Hintea

Student: _Vlad Andreescu____

Date of meeting: _____17/10/2018_______________________________

Record of individual actions completed + notes:

Research an idea for the final year project

Key topics Discussed:

What topic is best for me

What machine learning will be challenging enough for this project

Individual action points for next meeting (no more than 3)

Research machine learning ideas

[illegible]
____

Date of next meeting: 07/01/2019

Supervisor: Dr Diana Hintea

Student: _Vlad Andreescu____

Date of meeting: _____07/01/2018_______________________________

Record of individual actions completed + notes:

Research different ideas of projects related to machine learning

Analyse different research papers related to machine learning and Neural Network

Focusing my area of research around handwriting recognition

Key topics Discussed:

I proposed my idea to Diana.

She recommends me the online university library and google scholar in order to do my research

Individual action points for next meeting (no more than 3)

Write the Detailed Proposal (from where I can decide which path I will approach)

Complete the ethics application.

[illegible]
____

Date of next meeting: 08/02/2019

Supervisor: Dr Diana Hintea

Student: _Vlad Andreescu____

Date of meeting: _____08/02/2018______________________________

Record of individual actions completed + notes:

Completed the Detailed Proposal

Sent a copy of Detailed Proposal to Diana in order to receive some feedback

Key topics Discussed:

We discussed about the ethics application.

She gave me some feedback on the proposal

Individual action points for next meeting (no more than 3)

Using the feedback received, modify and improve the Proposal

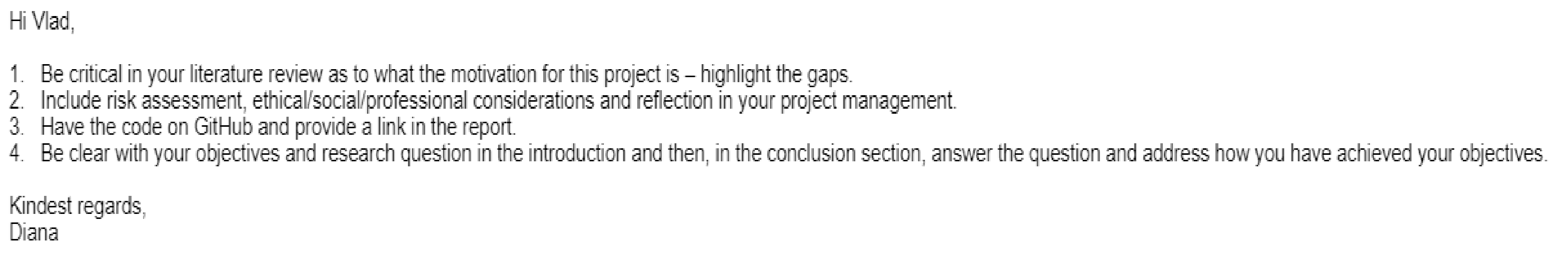
Wait for the ethics application to be accepted before starting to work on the project

[illegible]

____

Date of next meeting: 03/04/2019 … Presentation day

9.2 Supervisor Feedback



Hi Vlad,

1. Be critical in your literature review as to what the motivation for this project is – highlight the gaps.
2. Include risk assessment, ethical/social/professional considerations and reflection in your project management.
3. Have the code on GitHub and provide a link in the report.
4. Be clear with your objectives and research question in the introduction and then, in the conclusion section, answer the question and address how you have achieved your objectives.

Kindest regards,
Diana

9.3 Detailed Project Proposal

1 Research Question, Problem Statement or Topic for Investigation

Provide a clear outline of the research question, problem or investigation that you will be undertaking for your project. You should work with your supervisor to refine your idea into an achievable project outcome. Useful questions to address would be: what is the question or problem you are addressing, what evidence is there that this is a real problem, what approach or method are you going to take to address the problem. Try to ensure that your description of the question or problem is clear, specific and defined enough to base a short programme of primary research on it.

Which algorithm provides the most accurate recognition of offline handwriting?

2 Initial/Mini Literature Review

Using the University Library e-journal database, the ACM portal database or Google Scholar, identify and select between three and five research papers relating to your topic. Try and identify papers that are relatively current (within the last three years). A literature review is a select analysis of existing research which is relevant to your topic. It explains and justifies how your investigation may help answer some of the questions or gaps in this area of research. A literature review is not a straightforward summary of everything you have read on the topic and it is not a chronological description of what was discovered in your field. Use your literature review to:

- Show how your project will relate to previous studies.
- Compare and contrast different authors' views on an issue - note areas in which authors are in disagreement.
- Highlight current exemplary studies
- Highlight any gaps in research that may provide you with a starting point for your project
- Highlight any good approaches that may allow you to develop a project idea further

The key to the mini-literature review is your critical and evaluative perspective on the literature reviewed. ***Use the review to make a case/argument as to why your own research project is necessary/important.***

Handwriting analysis proved that handwriting is unique to every human, just as DNA, fingerprints, etc. There are several factors that determine the uniqueness of handwriting, such as the position in which you hold the pen, or the amount of pressure put on paper.

In the past few years, handwriting recognition received significant attention. This technology is used in diversified fields from healthcare to car manufactures and banks. Handwriting recognition is used to authenticate signatures or recognize different postcodes in letters. (P. Shankar Rao)

Handwriting recognition is the process in which handwritten characters are identified. There are two concepts of handwriting recognition: online and offline.

Online Handwriting Recognition refers to the characters being identified in real time while they are being written on the surface (e.g. writing a letter on a touchscreen). For the other concept, Offline Handwriting Recognition, an image with a letter, digit or word is provided which needs to be recognized. (Kala R. et al., 2010)

The offline approach reported a lower rate of recognition than the online approach, yet offline recognition technologies are efficient on domains where handwriting is highly used such as recognizing hand-written postal addresses on envelopes or reading bank checks therefore it has a huge economic impact. (P. Shankar Rao)

My project will be focused on the second approach, Offline Handwriting Recognition.

There are several methods and algorithms used to develop and improve handwriting recognition. The main ones this review will focus on are: Artificial Neural Network, Graph Theory using Genetic Algorithm, Binary segmentation algorithm and Optical Character Recognition.

Artificial Neural Networks (ANN) require training the system with all characters before a result is processed. (Graves et al., 2008). After the ANN is trained, when it receives an input which might be a digit, letter or word, it compares the input with the learnt data set and chooses the most likely character based on generalization (Kala R. et al., 2010).

A gap in ANN is that if the discrepancy between the input and the learnt data set is large, the result might not be correct. Therefore, the data set needs to be well designed and big enough to cover all the possible character styles. (Kala R. et al., 2010).

Graph Theory using Genetic Algorithms is a method based on ANNs. In order to train the system, for each input (image) a graph is created then a Genetic Algorithm is applied to mix the graphs and spot the most favorable solution.

Genetic Algorithms can be used to combine diverse styles of writing a character and produce new styles.

Graph Theory changed the problem of handwriting recognition to the problem of graph matching. Including Genetic Algorithm, the performance was highly improved.

The efficiency of this algorithm was 98.44% (Kala R. et al., 2010).

The segmentation process in offline handwriting recognition has a huge effect on the recognition result generating performance. By segmentation, a word is divided in multiple characters which represent the input for the recognition procedure. If the segmentation algorithm is not properly developed it could badly influence the recognition result. Therefore, some researchers skipped the segmentation step completely creating a "holistic" or segmentation-free approach (Lee and Verma, 2012). The holistic algorithm takes as an input the whole word and compare it with a dictionary.

Here the disagreement between researchers appears. In the English language there are 52 characters (lowercase and uppercase). In the segmentation approach the system must learn only 52 characters in order to be able to recognize a word. Contrarily, using holistic approach the system must learn all the English word which are more than 200,000. (Lee and Verma, 2012)

The principle of Binary Segmentation Algorithm (BSA) is to generate sub-images of the characters within a word until each generated image contains a known character by the system. There are two modules required by the algorithm to function properly: Suspicious Segmentation Points (SSPs) generation module and Segmentation Point (SP) which is selected from SSPs. Binary Segmentation Algorithm is applied to Segmentation Point which generate two sub-images. This process is repeated until each sub-image contains a character known by the system. (Lee and Verma, 2012)

This approach improves the handwriting recognition by lowering the number of characters that system has to learn.

Optical Character Recognition is the process in which an image containing a handwritten word is processed and scanned than translated into digital text format.

Soumya Mishra, Debashish Nanda & Sanghamitra Mohanty (2010 December) mentioned that the implementation of an Optical Character Recognition is very difficult especially if it is developed to function under all possible conditions and gives highly accurate results. (D.P. et al., 2011).

The gap in Optical Character Recognition is that it is very sensitive to image noise. An image noise is calculated by the level of blur of an image. If the image is very blurry, it lowers the accuracy of the algorithm. Several denoising algorithms are used to reduce the noise and increase signal-to noise ration of an image. (D.P. et al., 2011)

From this review you can see that handwriting recognition is a well-studied process. There are many different approaches, some methods produce better results than others. Each algorithm has its own advantages and disadvantages. The aim of this research is to examine the algorithms and choose the one which might be improved in order to receive the best results.

3 Client, Audience or Motivation:

To whom is this project important? A research project must address a research question that generates a small piece of new knowledge. This new knowledge must be important to a named group or specific client to make it worthwhile carrying out. This is the motivation for your project. In this section you should address who will benefit from your findings and how they will benefit. (note: 'greater knowledge about' is defined as a benefit – even if your audience is mainly other academic researchers, new data, if collected using scientific principles, adds to the body of knowledge about the topic).

Handwriting was the only way of communication until recent times in the history. A lot of history is lost because of indecipherable old handwriting. Handwriting recognition technologies can help archeologists find more information about our history.

One of the main reasons handwriting recognition is gaining so much attention is because it could play a huge role in Health Care System. A report from the National Academies of Science's Institute of Medicine (IOM) revealed that more than 7000 people are dying annually because of the doctor's sloppy handwriting. (van Leijen, 2012)

Both patients and doctors can benefit from Handwriting Recognition Technologies

In the past few years, Automotive Industry showed an interest in Handwriting Recognition Technology. After the study from August 2014 by J. D. Power which showed that 63% of the participants downvoted voice-recognition technology because they were unable to complete several tasks. (Read, 2014) Therefore, car manufactures shifted their focus on handwriting recognition technologies to create a communication between the driver and multimedia systems (e.g. writing a person name in Contacts).

Recently, the education sector has slowly adopted technology in their daily activities (e.g. from notebooks and books to laptops). Handwriting recognition technologies can help both teachers and students to improve the learning experience.

Currently Handwriting Recognition Technology is used for taking notes and translate teacher's sloppy handwriting. There is much more this technology can provide, such as interactive classes and simpler ways to share notes.

4 Primary Research Plan

This is the plan as to how you will go about answering your detailed research question - It must include a primary research method (an extended literature review is not an acceptable primary method). Think and plan logically. Primary methods may include experiments, applications or software demonstrators, process models, simulations, surveys, analysis of existing or generated data … you may wish to suggest a timeline (covering semester 2) or simply set out a sequence of tasks. Where you intend to collect data think about how much data you need and how long the collection process will take. Make reasonable assumptions about the amount of work you can do and try not to 'over-promise' on results – most scientific research is small scale and time limited, this is even more the case with student projects where you also have competing modules.

For my primary research plan, I will be researching some handwriting recognition algorithms that are already using to solve real world problems. After I analyzed some algorithms, the next step is to gather some datasets of handwritten letters digits or words. When I am happy with the data I will start the development phase following Agile Development Cycle. The last step is testing and retrieving the results. The time management will be organized using Gantt Chart. The following steps will be followed in order to reach the desired outcome:

1. Research handwriting recognition algorithms
   In this step, I will be focusing on artificial neural network framework, comparing and evaluating different handwriting recognition algorithms (e.g. Genetic Algorithm, Binary Segmentation, Feedforward) in order to create a starting point for my dissertation. Each algorithm has its own advantages and disadvantages, so I must take multiple factors in consideration (e.g. result accuracy, neural network dataset).

2. Data Collection
   During this step, I am searching for different datasets of pictures containing handwritten letters, digits or words in order to teach my artificial neural network. I am looking for predefined datasets but if I won't succeed, I will design my own datasets. From my research, I found several datasets containing approximately 70.000 28x28 pixel format pictures of handwritten letters provided by National Institute of Standards and Technology.

3. **Developing the software**
   With the data I collected, I can proceed in developing phase. I am developing a machine learning using Artificial Neural Network framework.
   First, I will be most likely choosing Python as main programming language (the other option would be C++). Then, I will be using TensorFlow and OpenCV as recognition libraries in Python.
   In order to have everything under control I will be using Git as version control system and a well-designed time management schema

4. **Testing and Results**
   This is the final step where I am testing the software using the dataset collected and retrieve the results. To test the algorithm, the first task is to teach the neural network with a predefined dataset of characters. After the teaching process is finished, I will provide an unknown input which will represent a character handwritten by me. The algorithm must match the input (character) with the digital representation of it. The results consist in mainly the accuracy of the algorithm, also the time required for each iteration.

5 Intended Project Outcome

Describe, as clearly as possible, what outcome your project aims to produce in relation to the original question, investigation or problem statement.

As handwriting recognition is more and more important in the society, my project outcome is to determine if the current used algorithms can be improved and if so how would they influence our daily basis activities. The results will consist in the accuracy of the algorithm, how many resources it will be using as well as the difficulty of implementation.

6 Bibliography (key texts for your literature review)

Please provide references, in correct Harvard style, for the research papers that have informed your literature review. The references should be recent and sufficiently technical or academic. Your markers will be looking for you to identify technical reports, conference papers, journal papers, and recent text books. Avoid *Wikipedia* entries, newspaper reports that do not cite sources, and general or introductory texts.

1. D.P., G., GUNGE, Y., MUNDADA, R., BHARADWAJ, H. and PATIL, S. (2011). USER DEPENDENT HANDWRITING RECOGNITION USING OFF-LINE OCR FOR ARITHMETIC OPERATION AND TEXT PROCESSING. *i-Manager's Journal on Software Engineering*, [online] 6(1), p.9. Available at: https://search.proquest.com/docview/1473909263?accountid=10286&rfr_id=info%3Axri%2Fsid%3Aprimo [Accessed 28 Jan. 2019].

2. KALA, R., VAZIRANI, H., SHUKLA, A. and TIWARI, R. (2010). Offline Handwriting Recognition using Genetic Algorithm. *International Journal of Computer Science Issues (IJCSI)*, [online] 7(2), p.11. Available at: https://search.proquest.com/docview/223065287/fulltextPDF/FD3B7FE0657C49B7PQ/1?accountid=10286 [Accessed 26 Jan. 2019].

3. Lee, H. and Verma, B. (2012). Binary segmentation algorithm for English cursive handwriting recognition. *Elsevier*, [online] 45(4), p.12. Available at: https://www.sciencedirect.com/science/article/pii/S0031320311003943 [Accessed 1 Feb. 2019].

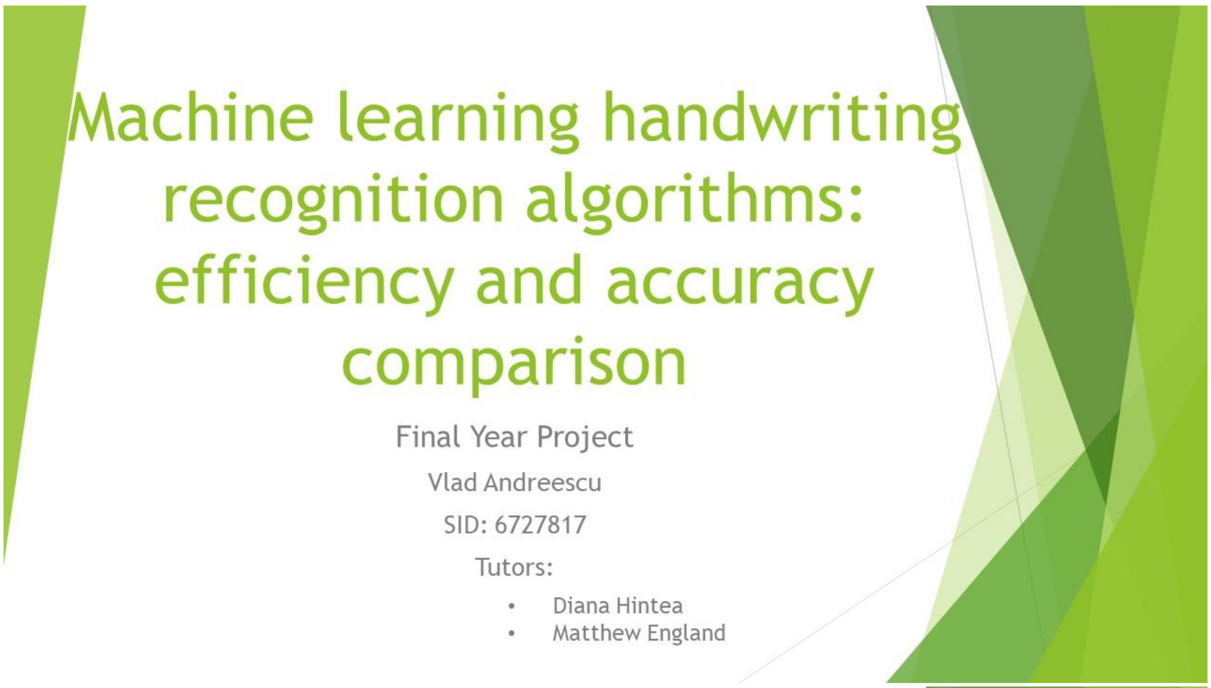
4. P. Shankar, R. and J., A. (n.d.). *Handwriting Recognition – "Offline" Approach*. [ebook] p.4. Available at: https://cs.stanford.edu/people/adityaj/HandwritingRecognition.pdf [Accessed 27 Jan. 2019].

5. Graves, A., Fernandez, S., Liwicki, M., Bunke, H. and Schmidhuber, J. (2008). *Unconstrained Online Handwriting Recognition with Recurrent Neural Networks*. [online] People.idsia.ch. Available at: http://people.idsia.ch/~juergen/nips_2008.pdf [Accessed 1 Feb. 2019].

6. van Leijen, M. (2012). *Death by prescription: Doctors' handwriting causes 7,000 deaths a year*. [online] Emirates24|7. Available at: https://www.emirates247.com/news/emirates/death-by-prescription-doctors-handwriting-causes-7-000-deaths-a-year-2012-11-04-1.481418 [Accessed 3 Feb. 2019].

7. Read, R. (2014). *J.D. Power Offers More Proof That Voice-Recognition Systems Are The Worst Things In Our Cars*. [online] The Car Connection. Available at: https://www.thecarconnection.com/news/1094106_j-d-power-offers-more-proof-that-voice-recognition-systems-are-the-worst-things-in-our-cars [Accessed 3 Feb. 2019].

9.4 Mid-term Presentation



Machine learning handwriting recognition algorithms: efficiency and accuracy comparison

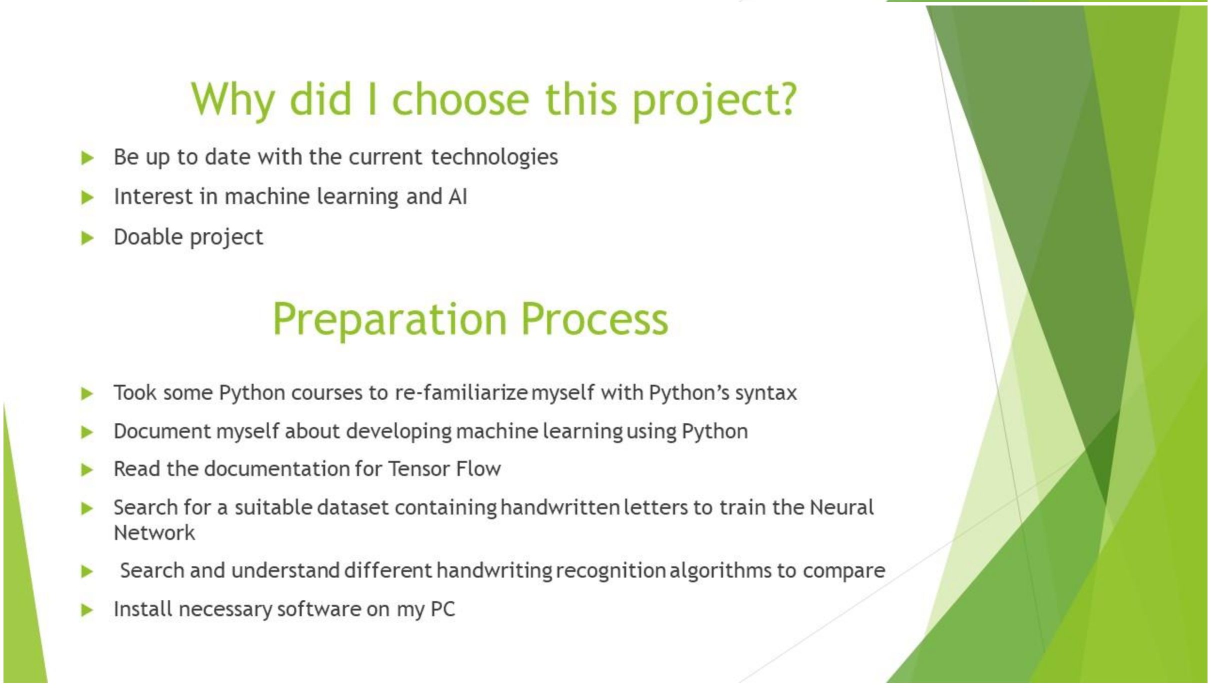
Final Year Project

Vlad Andreescu

SID: 6727817

Tutors:

- Diana Hintea
- Matthew England



Why did I choose this project?

- Be up to date with the current technologies
- Interest in machine learning and AI
- Doable project

Preparation Process

- Took some Python courses to re-familiarize myself with Python's syntax
- Document myself about developing machine learning using Python
- Read the documentation for Tensor Flow
- Search for a suitable dataset containing handwritten letters to train the Neural Network
- Search and understand different handwriting recognition algorithms to compare
- Install necessary software on my PC

Hardware and Software used

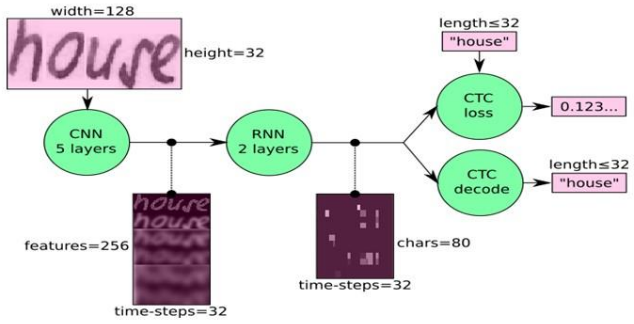
Hardware

- CPU: i5 6600k 3.50GHz
- GPU: GeForce GTX 1060 6GB
- RAM memory: 16 GB

Software

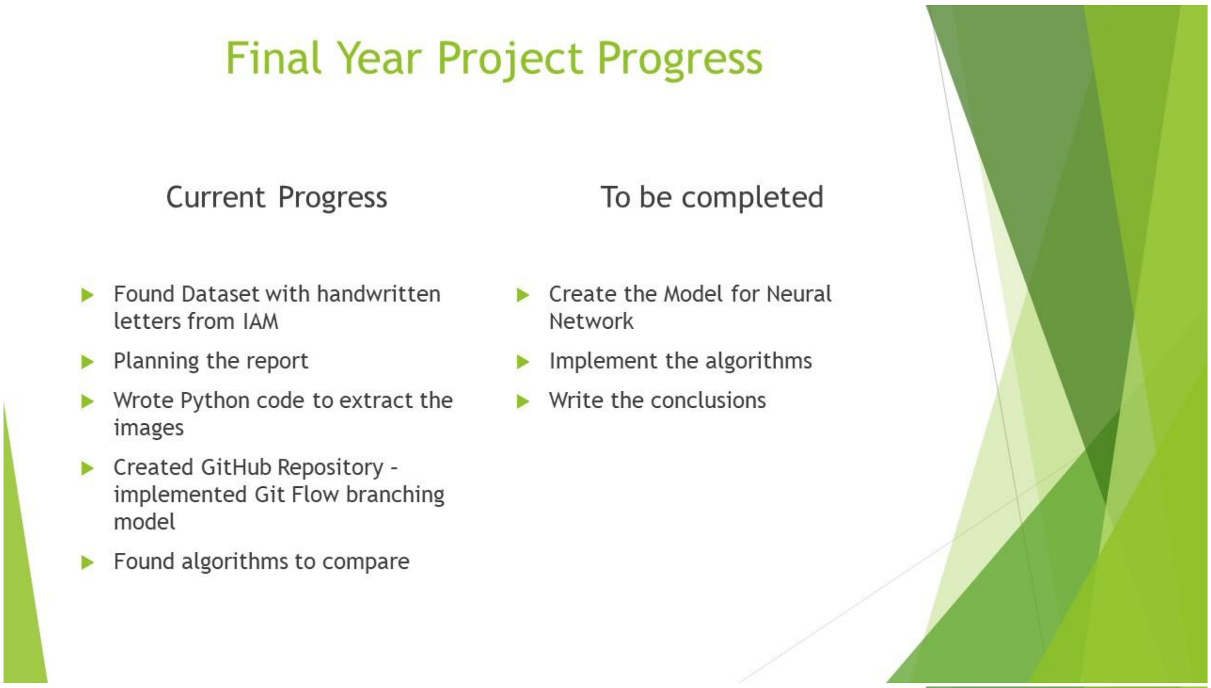
- Operating System: Windows 10 Pro 64-bit
- Python 3.7
- Tensor Flow 1.3
- Version Control: Git Bash

Neural Network design



CNN - Convolutional Neural Network
- Receive image as input
- Extract relevant features from the image

RNN - Recurrent Neural Network
CTC - Connectionist Temporal Classification



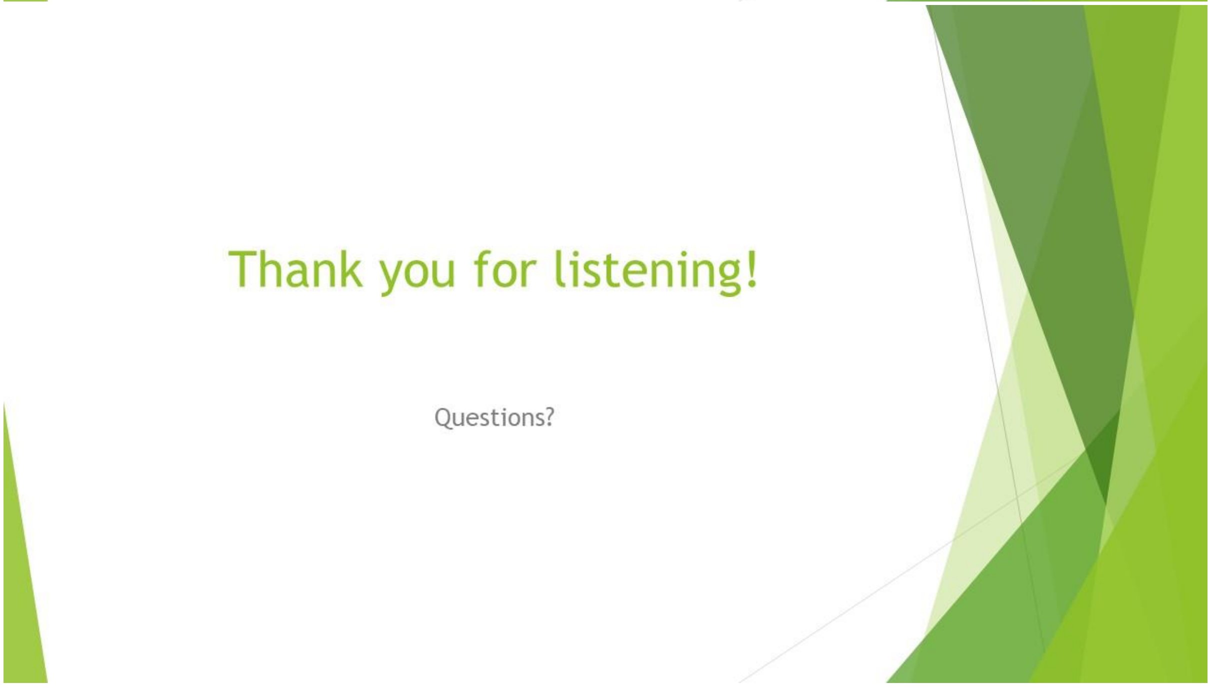
Final Year Project Progress

Current Progress

- Found Dataset with handwritten letters from IAM
- Planning the report
- Wrote Python code to extract the images
- Created GitHub Repository - implemented Git Flow branching model
- Found algorithms to compare

To be completed

- Create the Model for Neural Network
- Implement the algorithms
- Write the conclusions



Thank you for listening!

Questions?