Mittuniversitetet
MID SWEDEN UNIVERSITY

Form data enriching using a post OCR clustering process

Measuring accuracy of field names and field values clustering

Adil Aboulkacim

Computer Engineering BA (C), Final Project
Main field of study: Computer engineering
Credits: 15 credits
Semester, Year: Spring, 2022
Supervisor: Karl Petterson
Examiner: Patrik Österberg
Course code: DT099G
Programme: Master of Science in Engineering - Computer Engineering

Abstract

With OCR technologies the text in a form can be read, the position of each word and its contents can be extracted, however the relation between the words cannot be understood. This thesis aims to solve the problem of enriching data from a structured form without any pre-set configuration using clustering. This is done using the method of a quantitative measurement of a developed prototype counting correctly clustered text boxes and a qualitative evaluation. The prototype works by feeding an image of an unfilled form and another image of a filled form which contains the data to be enriched to an OCR engine. The OCR engine extracts the text and its positions which is then run through a post-processing step which together with a modified Euclidean and fuzzy string search algorithm, both together is able to cluster field names and field values in the filled in form image. The result of the prototype for three different form structures and 15 different images for each structure ranges from 100% to 92% accuracy depending on form structure. This thesis successfully was able to show the possibility of clustering together names and values in a form i.e., enriching data from the form.

Keywords: Optical Character Recognition, Form Processing, Data enrichment

Sammanfattning

Med OCR teknologier kan innehållet av ett formulär läsas in, positionen av varje ord och dess innehåll kan extraheras, dock kan relationen mellan orden ej förstås. Denna rapport siktar på att lösa problemet med att berika data från ett strukturerat formulär utan någon förinställd konfiguration genom användandet utav klustring. Detta görs med en kvantitativ metod där mätning av en utvecklad prototyp som räknar antal korrekt klustrade textrutor och en kvalitativ utvärdering. Prototypen fungerar genom att mata en bild av ett ofyllt formulär och en annan bild av ett ifyllt formulär och en annan bild av ett ifyllt formulär som innehåller informationen som ska berikas till en OCR-motor. Utdatan från OCR-motorn körs genom ett efterbearbetningssteg som tillsammans med en modifierad euklidisk algoritm och en oskarp strängsökningsalgoritm kan klustra fältnamn och fältvärden i den ifyllda formulärbilden. Resultatet av prototypen för tre olika formulärstrukturer och 15 olika bilder vardera gav en träffsäkerhet från 100% till 92% beroende på formulärstruktur. Denna rapport kunde visa möjligheten att grupper ihop fältnamn och fältvärden i ett formulera, med andra ord utvinna information från formuläret

Nyckelord: Optisk teckenläsning, Formulärbearbetning, Databerikning

Acknowledgements

First and foremost, I would like to thank Karl Petterson as my supervisor at Mid Sweden University, with his invaluable advice and guidance many mistakes were avoided, and a lot of time was saved in both creating the prototype and writing the report.

Secondly, I'd like to thank Magnus Eriksson at Mid Sweden University for the idea of extracting information from forms in the key-value pair format.

Contents

List of Figures

List of Tables

Terminology

API	Application Programming Interface
CNN	Convolutional neural network
EAST	Efficient and Accurate Scene Text detector
ICR	Intelligent Character Recognition
LSTM	Long short-term memory
OCR	Optical Character Recognition

1 Introduction

What do a postal office and a blind person have in common? Both want to have an easy and convenient way to read text written on physical mediums. Through transformation to an electronic system these mediums can more easily be understood and used. In our current time it is becoming more common to digitize and read our physical environment using a technique called optical character recognition (OCR) into machine-encoded text for an easier and more convenient usage using computers. Not only does digitizing make our common day to day lives easier, allows for automation and further ease of life improvements, but it also aids the most unfortunate ones of us who cannot easily read.

This question of how to understand text either electronically or mechanically has been studied since early 1900s [1]. OCR is a scientific subfield of the combination of pattern recognition, computer vision, and in recent times, artificial intelligence. It is the transformation of images which contains text of typed, handwritten, or printed character into machine-encoded text which can be understood and used by digitally.

1.1 Background and problem motivation

Humankind has since the ancient civilization of Sumer from 3200 BCE [2] been writing and printing symbols and text on various physical mediums to later be read and understood visually. Not only is physical text created for long-term usage but also for short-term in forms of letters, paper invoices, reports, which are sent to be consumed, processed, digitized, and then promptly discarded.

The constant growth of information in physical form and the need for digitization creates an increasing problem for both those who want to share information and those who want to consume information. Bridging the gap between producers and consumers of information whilst also including those with hindrance to consume information easily (Vision impairment, reading disability, etcetera) is a problem that exists in the present and has never been more current to be solved.

The OCR technology can save countless working hours in various job sectors, it can enable access to information for more people, and it can help those with reading hindrances in one form or another. As OCR increases its reliance on various artificial intelligence techniques such as neural networks which steadily becomes easier to implement, more accurate in prediction, and faster to compute, this future becomes closer to our present each day.

1.2 Overall aim

The aim of this project is to give an introductory understanding to why and how to enrich textual data from an OCR system, construct a prototype for automatically reading and enriching data from images containing text, as well as measuring the reliability of the prototype to enrich data accurately automatically from text found in images. This is to give an understanding if automatic data enriching from an OCR system is possible and to which degree it is possible through analytic measurements and higher-level discussion later in the report.

1.3 Problem statement

The thesis investigates the problem statement of whether it is feasible to use an automatic postprocess step, relying on OCR system, to enrich extracted data from images containing text in a structured form. To which degree it is achieved, which is measured and reported in results chapter while the over-aiming feasibility shall be answered and discussed in discussion chapter.

1.3.1 Goal 1

First goal is to be able to detect, recognize, and group text such that each field value in a form is grouped with the right field names from the form. Reaching this goal will allow for further development of clustering and trying to extract relations between field names and field values.

1.3.2 Goal 2

Next goal is of this project is to be able to some degree extract the relation between of field names and field values. Each group contains inputted information from the user in the form of handwriting, and field name that represents the field in which the information was written in. Measuring the accuracy of correctly separated information from the fields will reach goal 2.

1.4 Scope

The project scope is focused on a subsection of the subfield OCR, this is to solely focus on the automatic data enriching and not the workings of optical character recognition in terms of pattern recognition, computer vision, or artificial intelligence. Furthermore, this project will focus on providing a limited prototype to test and prove the feasibility of introducing automatic textual data enriching into OCR systems, this will not result in a fully developed software solution instead a steppingstone for further development and research.

1.5 Outline

Any figure with no source explicitly cited has been created by the author. Chapter 2 goes over the theory for OCR and ways of extracting data and information, as well as related works to give a better understanding of OCR and ways others have solved enriching data from forms. Chapter 3 describes the scientific and project method, as well as the two goals for the project, and lastly how the projects success will be evaluated. Chapter 4 the implementation will cover the workings of OCR engine and the postprocess clustering step which enriches data. Chapter 5 presents the results of running the prototype over different forms and a number of images. Chapter 6 discusses not only the results but the project as a whole from a subjective point of view. Lastly chapter 7 contains the conclusions drawn from finished work.

2 Theory

The aims of this chapter are to cover introductory information to understand the basics of OCR, information extraction, and how the reliability is going to be measured, which will give the reader enough understanding to interpret the rest of this thesis. Furthermore, this chapter is going to cover the theory needed to understand technical aspects of chosen methods, algorithms, libraries, and frameworks.

2.1 Optical character recognition

The subfield optical character recognition (OCR) is in simple terms the technique of trying to get a computer to understand the text on an image. This conversion from text on an image to machine-encoded text is the core function of OCR. How this task has been achieved has changed from conventional methods using pattern recognition, to more advanced methods within computer vision, and to most recent times now also using neural networks from the field of artificial intelligence. This technique bridges the gap between our physical world and digital world through conversion of text present physically to digitized text which can be further understood and processed by computers.

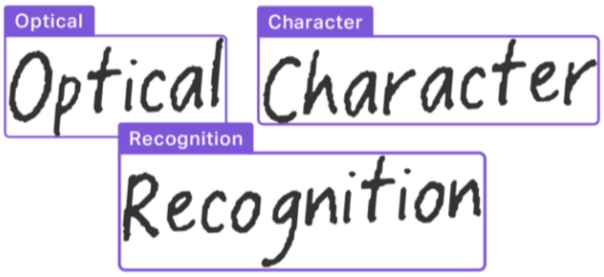


Figure 1: Optical Character Recognition [3]

OCR systems work by first detecting text and putting a bounding box around them. This box is processed by a text recognition module which outputs the predicted text [4]. Figure 1 visualizes the bounding box around a word and the recognized text.

2.2 OCR pipeline

In general, modern OCR applications contain four steps in succession that can be referred to as the OCR pipeline as seen in figure 2. These four steps are defined as technical operations with clear boundaries in duties and order. These four steps are pre-processing, text detection, text recognition, and post-processing. Some applications combine text detection and text recognition into a single step, and some call the final step restructuring instead.

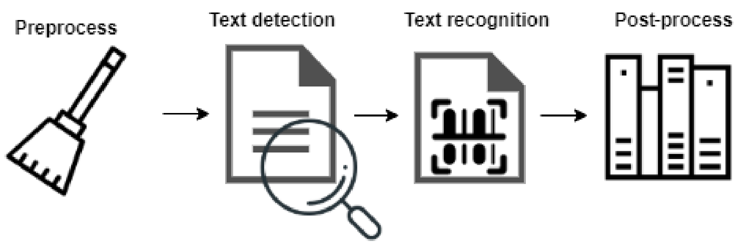


Figure 2: OCR pipeline visualized

Pre-processing usually contains conventional methods to convert a raw unedited image into an image more suitable for image processing. This step is essential to produce a better result in the later steps as the processed image is pipelined to the next step. Some operations that the pre-processing step may include are deskewing, grayscaling, binarization, noise reduction, and image scaling [5].

The grayscaling done in the implementation uses the library OpenCV [6] which converts the three-colour space RGB to a singular value to represent the amount of light.

$$RGBtoGray : 0.299R + 0.587G + 0.114B \rightarrow Y$$

Equation conversion from RGB to grayscale [7]

The second step is text detection which originally used conventional methods but now finds more success using artificial intelligence in the form of neural networks. This step's main purpose is to find regions of interest which may contain text. The recognized regions may differ in size and rotation with an accompanying score to dictate how likely that there is text present. Further conventional methods such as IoU (Intersection over Union) and thresholding is executed to produce more accurate results.

Third step is the most crucial which is text recognition which takes an image or subsection of an image and outputs a list of most likely character for each position. This is used to output the most recognized string of characters.

Lastly is the step of post-processing which is usually uses conventional methods to further improve the accuracy of the output. This includes methods of comparing recognized text to lexicon, which might be all the words of a certain language, or a specific subset of words of a field.

2.3 Data extraction

The difference from a scanner to an OCR system is that a scanner creates an image representing the paper, without being able to store, locate, or manipulate each individual character, word, or sentence. Without extracting the textual data using OCR the image data is mostly unusable. The OCR system extracts data from the image representing the paper without understanding the actual text. [8]

The process of retrieving data out of a data source is called data extraction. This enables further processes such as data processing or data storage on the extracted data. Usually, data is transformed and put into some context through the use of metadata before being exported to another program or directly to the user. The exported data is what is called unstructured data which means it is not arranged to any pre-set schema. [9]

2.4 Information extraction

The next more advanced and complex extraction is information extraction which automatically converts previously unstructured data into structured information. This is done for the end goal of allowing either further computation to be done or for a user to draw inferences based on logical reasoning. The overall goal of understanding the data can be divided into several different tasks and sub tasks. One of these tasks is template filling which aims to extract a set collection of attributes from a data source (text, document, article) into a hierarchical structure of one or more key-value pairs [10].

2.5 Implementation specific theory

This subsection will include theory specific for the implementation.

2.5.1 Euclidean distance

For the implementation the formula for Euclidean distance was used which calculates the distance of a straight path between two points on a plane. The formula as follows: $d(P,Q) = \sqrt{(x-a)^2 + (y-b)^2}$. Where the distance between point $P = (x,y)$ and point $Q = (a,b)$ are calculated, visualized in figure 3. In the following figure the Euclidean distance formula is visualized where lighter regions denote a higher distance to the origin.

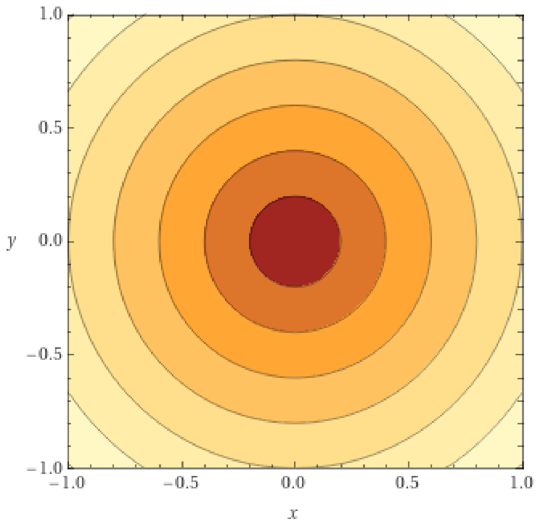


Figure 3: Euclidean distance visualized

2.5.2 Euclidean distance modified

The formula can be modified to give a higher weight to a certain axis. In the implementation a modification to the formula was done to give higher weight to the vertical axis. The formula is as follows, visualized in figure 4:

$$d(P,Q) = \sqrt{(x-a)^2 * 0.5 + (y-b)^2 * 2}$$

This is used in the implementation to allow individual text boxes containing field names to be clustered together. Using the formula two field names are inputted as points and if the weight is below a certain threshold the field names are considered part of the same cluster.

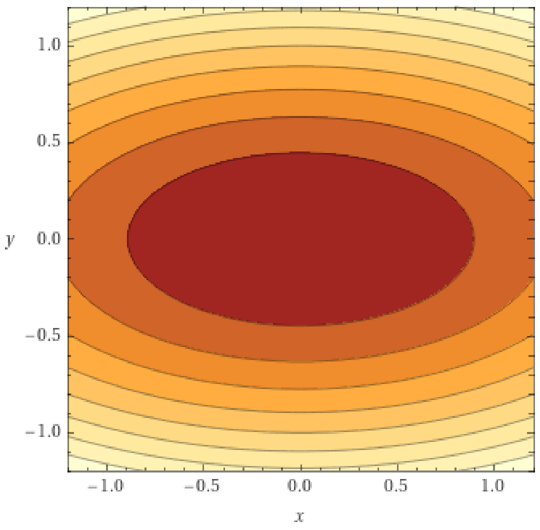


Figure 4: Modified Euclidean algorithm visualized

2.5.3 Piecewise Euclidean distance modified

The second modified Euclidean distance formula is created using a piecewise equation, visualized in figure 5. Where the weight is higher for negative x and/or positive y. This formula is used to cluster field values to field names. The formula has lowest weight for the fourth quadrant, medium weight for first and second quadrant, and highest weight for second quadrant. When clustering field values to field names this gives field values a lower weight if they are positioned to the bottom-right, then directly below and to the right, and lastly the highest weight if they are positioned top-left.

$$d(P,Q) = \begin{cases} \sqrt{(x-a)^2 * 2 + (y-b)^2 * 0.5} & x \leq 0 \, y \leq 0 \\ \sqrt{(x-a)^2 * 2 + (y-b)^2 * 2} & x \leq 0 \, y > 0 \\ \sqrt{(x-a)^2 * 0.5 + (y-b)^2 * 0.5} & x > 0 \, y \leq 0 \\ \sqrt{(x-a)^2 * 0.5 + (y-b)^2 * 2} & x > 0 \, y > 0 \end{cases}$$

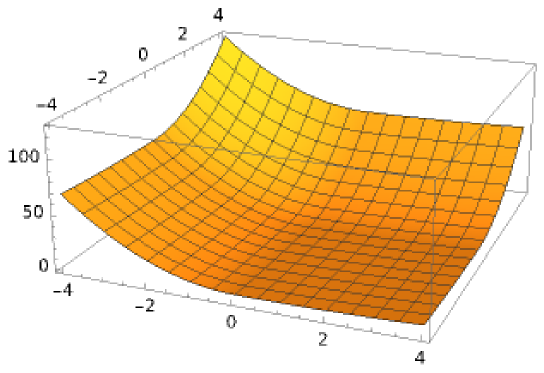


Figure 5: Piecewise modified Euclidean algorithm visualized

2.6 Intelligent character recognition

The successor to OCR is a more advanced system which incorporates learning new fonts and handwriting, this is called intelligent character recognition (ICR). While OCR systems may also contain more advanced techniques such as neural networks and learning, this approach was first used by ICR systems. It's considered that OCR is more suitable to read printed text and ICR more suitable to read handwritten text. [11] Generally however, the term OCR can be referred to both OCR and ICR systems as both techniques solve the same problem using similar methods.

2.7 Related work

This sub chapter contains the works which this thesis was inspired by. Furthermore, this sub chapter will go over related works found in the scientific community which covers the same or similar problems to give both a broader and deeper knowledge on the problem.

2.7.1 Android application for digitization of forms

Erik Fahlén wrote the thesis *"Tolkning av handskrivna siffror i formulär: Betydelsen av datauppsättningens storlek vid maskininlärning"* on digitizing forms using object recognition which was written at Mid Sweden University 2019 as a bachelor thesis. The purpose of the thesis was to see which combination of dataset and hardware using the machine learning library TensorFlow produced the best results in the metrics of accuracy, speed, and cost. This was done by training a model called Single Shot MultiBox Detector MobileNet version using Google Clouds Machine Learning Engine, after which the trained model was converted to a phone fitted model which was used in an android application. The model's purpose was to detect handwritten filled and unfilled boxes in physical forms [12]. While the thesis mostly covers which configuration provides best accuracy, speed, and cost, it also describes the methods of reading a form by using object recognition which is useful since not all forms exclusively only text as input.

2.7.2 Digitization of handwritten numbers on physical forms

Jonathan Manousian wrote the thesis *"Digitalisering av handskrivna siffror på fysiska formulär: Utvärdering av tillförlitlighet och träningstid"* on reading forms in specific reading handwritten numbers which proves to be very tricky for OCR engines to handle which was written at Mid Sweden University during 2020 as a bachelor thesis. The purpose of the thesis was to measure the accuracy of OCR tools found online compared to a model trained for the thesis using the TensorFlow framework. The model was fed manually cropped regions containing digits of photos taken or scanned of physical forms. Results of the thesis show that training a model to recognize specific handwriting yields higher accuracy of 80% on similar handwriting style as shown in figure 6, than the general-purpose OCR tools on handwritten digits which produced on average a lower accuracy of 37.5% [13]. This work describes the issues of handwritten text on forms which may give a lower than a workable accuracy when using traditional OCR tools compared to newer tools which use machine learning.

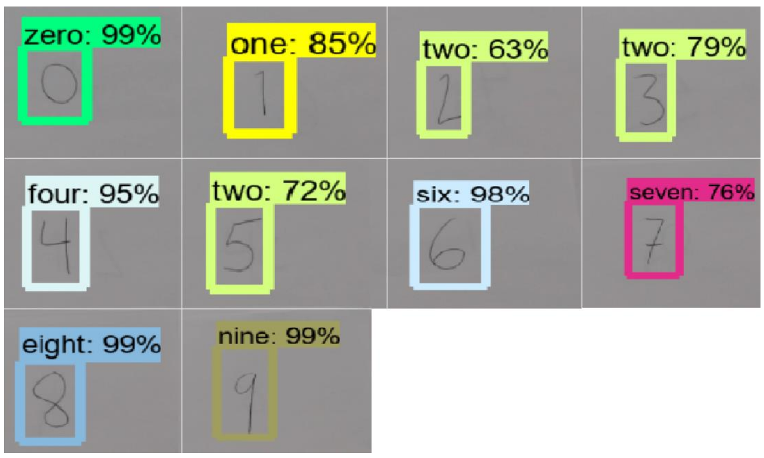


Figure 6: Trained model detecting handwritten digits in form [13]

2.7.3 Interpretation of handwritten numbers in forms

Engin Kirik wrote the thesis *"Tolkning av handskrivna siffror i formulär: Betydelsen av datauppsättningens storlek vid maskininlärning"* [14] on the growth in accuracy of an OCR engine using machine learning when reading handwritten numbers in a form which was written at Mid Sweden University during end of 2020 as a bachelor thesis. The purpose of the thesis was to measure the accuracy of a trained OCR model using different sizes of training dataset. This is to find out how important size of dataset affects results of object recognition. The model was created using both TensorFlow and PyTorch frameworks with a training size from 10 images to 30 images as shown in figure 7. This work proves the importance of a large dataset for training learning models in OCR for handwritten text.

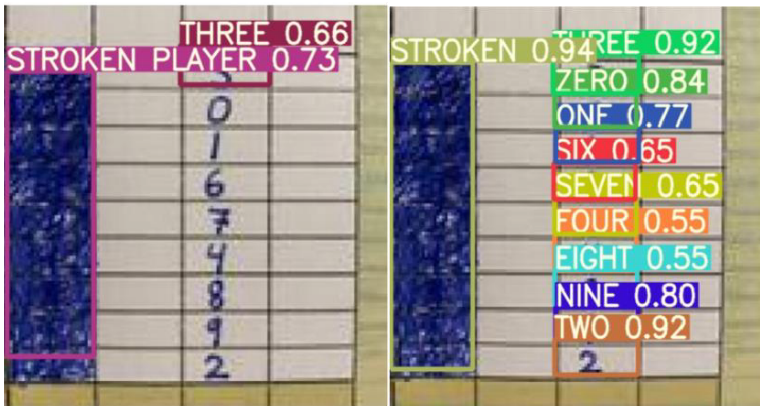
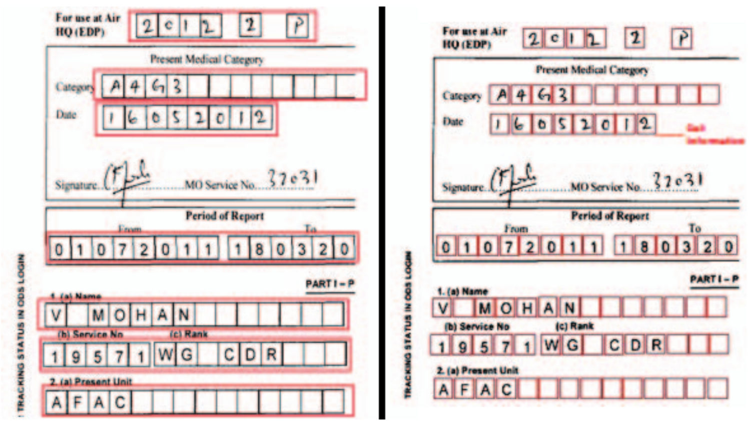


Figure 7: Results of two models trained with 10 and 30 images respectively [14]

2.7.4 Automatic Classification of Handwritten and Printed Text in ICR Boxes

One paper by Abhishek and Mohd at Newgen Software Technologies researched the possibility and the accuracy of a proposed method to automatically recognize zones of either printed or handwritten text. [15]. The idea is that by separating the work of recognizing characters of printed text to an OCR system and handwritten text to an ICR system the recognition rate and production rate can be increased compared to feeding both types of text to the same system. This is done by a manually selecting regions of interest which contains either printed or handwritten text as seen in figure 8. These regions are then divided into cells using ICR cell detection to isolate each potential character, the cells which previously resided in one region are now considered part of one zone. The zone is treated to be either printed or handwritten, however each cell may or may not contain a valid character.


TRACKING STATUS IN ODS LOGIN

For use at Air HQ (EDP) 2 0 1 2 2 P

Present Medical Category

Category A 4 G 3

Date 1 6 0 5 2 0 1 2

Signature [illegible] MO Service No. 32031

Period of Report

From 0 1 0 7 2 0 1 1 To 1 8 0 3 2 0

PART I – P

1. (a) Name V MOHAN

(b) Service No 19571 (c) Rank WG CDR

2. (a) Present Unit AFAC

TRACKING STATUS IN ODS LOGIN

For use at Air HQ (EDP) 2 0 1 2 2 P

Present Medical Category

Category A 4 G 3

Date 1 6 0 5 2 0 1 2 Cell Information

Signature [illegible] MO Service No. 32031

Period of Report

From 0 1 0 7 2 0 1 1 To 1 8 0 3 2 0

PART I – P

1. (a) Name V MOHAN

(b) Service No 19571 (c) Rank WG CDR

2. (a) Present Unit AFAC

Figure 8: Selected zones before and after ICR cell detection [15]

Firstly, for each cell in each zone, the possibility of a present character is predicted using an eight-neighbour connected component-labelling algorithm. Three features of valid characters in each zone are then extracted, these features are inter-character gap (ICG) shown in figure 9, character height, and baseline. If one of these features of every character in a zone fails to reach a certain threshold, the whole zone is considered to contain handwritten text. However, if all three features of all characters reach the threshold for acceptance, then the zone is considered to contain printed text.

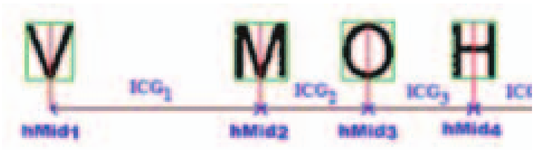


Figure 9: Identifying the midpoint and ICG of the characters [15]

This is important to discern between printed and handwritten text in each zone, as these characters are passed to an OCR system for printed text and ICR system for handwritten text. The implemented solution managed to properly detect handwritten zones with 96.24% accuracy and printed zones with 91.17 % accuracy. The paper suggests that by passing different texts to different engines the recognition rate can be increased, however this is not tested in the paper.

While this paper presents a novel idea on how to extract information from a form using predefined zones and designated character boxes, it also presents ways of identifying objects out of pattern using measurements and averages creating thresholds for said object being in the group or not. This is used as inspiration for the implementation of this project.

2.7.5 Azure form recognizer

Microsoft developed a tool to extract key-value pairs, text, and tables from images and documents. The tool uses machine-learning models to analyse forms and documents, extracts the text and data, maps field relations as key-value pairs, and lastly returns the output result in a computer friendly JSON format. [16] Azure form recognizer uses several underlying tools such as OCR, text analytics, and custom text classification. The underlying OCR tools works in both extracting text from printed and handwritten documents, and extract information to provide more structure and information to the text extraction. [17] The Azure form recognizer uses deep learning technology to extract tables from documents and images. This has been proven to be more successful than other methods in computer vision applications, which include OCR. [18] The neural network type which predicts tables is called Mask R-CNN which is an extension of Faster R-CNN by adding another branch to predict object mask and work in parallel with bounding box in an image. The overhead of another branch is relatively small while adding a more effective generalized approach. [19].

While an in-depth explanation of the inner workings of this proprietary OCR software is not available, the high accuracy Azure for recognizer provides is both a proof of such advanced OCR engines existing as well as the possible usage in future workings.

2.7.6 Tesseract

Originally developed by Hewlett-Packard but from 2006 and onwards by Google Tesseract is an open-source engine for OCR. Tesseract uses neural networks called Long short-term memory (LSTM) [20] which focuses on line recognition to recognize characters. While the engine is open-source, the package is distributed as a library or binary for out-of-the-box usage. [21] Tesseract does not solve form recognition but serves as a well-functioning alternative as OCR engine which could be used in future works.

3 Methodology

In this thesis both a qualitative and quantitative method will be conducted to measure the results of the project through the implementation. An iterative approach is taken to progress the thesis and the implementation simultaneously each cycle. Two milestones are noted, first one for recognizing groups of text containing information, and the second one for enriching the data for recognized groups. Lastly, the success of this thesis lays not in its degree of successful implementation, but to which degree this thesis helps the reader understand the problem and advantage of automatic data enrichment from OCR systems.

3.1 Scientific method description

To measure the success of the implementation a quantitative method is used. Measurements are done by counting the amount of correctly and incorrectly clustered text boxes from which an accuracy can be calculated. The measurements will be the basis of proving the feasibility of solving the problem presented in this thesis 1.3. This measurement will be unique for each type of form tested and how the measurement is done specifically is explained in 5.1

3.2 Project method description

The project that has been conducted is done through an agile methodology where an iterative approach is being taken. This will mean that most parts of the thesis will be continuously written, and the prototype will be developed simultaneously. This is to combine the scientific research and software development in a suitable iterative workflow that progresses the thesis incrementally each cycle. Problem of certain parts of the thesis not being able to advance due to it inherit nature of relying on previous chapters or parts is a possibility. Furthermore, could the technical complexity prove to be a problem for the scope.

3.3 Evaluation method

The degree of how much understanding into automatic data enrichment on OCR systems dictates if the thesis was successful or not. As this thesis does not aim to give a definite answer to the possibility of the chosen problem. No matter the results from the implementation, this thesis aims to show but a subsection of what could possibly be done with these technologies.

4 Implementation

The implementation is done using the Python language with the notable library OpenCV [6] as main image manipulation library. Implementation starts with inputting two images, one of which is of a blank form and the other of the same form but with filled field values called filled form. The two input images both go through the same OCR engine and then later go through a postprocess clustering step as shown in figure 10. Clustering the filled form image is done using the results from clustering the blank form image with a various conventional algorithm.

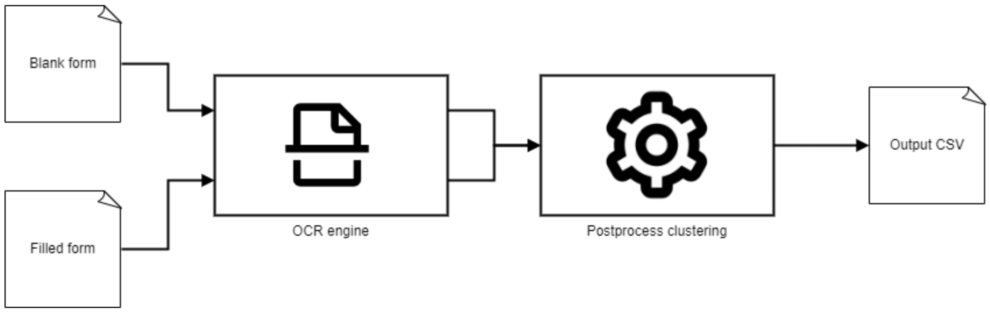
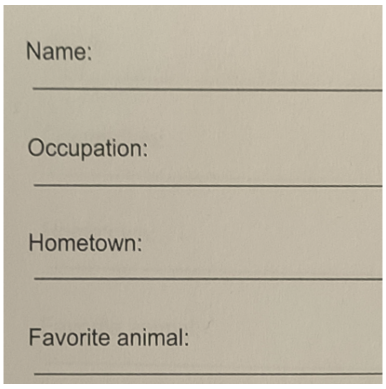


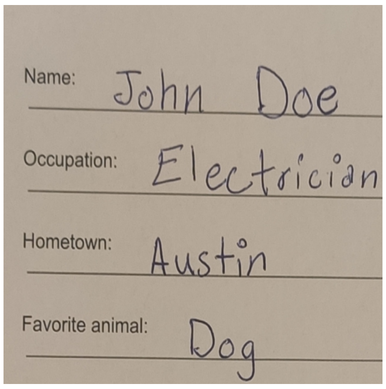
Figure 10: Application overview

The images used in the implementation is taken to be as alike as possible in terms of camera angle, lighting, scaling, etc. as shown in figure 11



Name:	
Occupation:	
Hometown:	
Favorite animal:	

(a) Blank form



Name:	John Doe
Occupation:	Electrician
Hometown:	Austin
Favorite animal:	Dog

(b) Filled form

Figure 11: Example of input images

4.1 OCR engine

The heart of any OCR engine is the text detection and text recognition parts, these parts together with pre-processing and postprocessing (such as error correction) make up the basic functioning OCR. However in this implementation no postprocess for the OCR engine is being used, as shown in figure 12

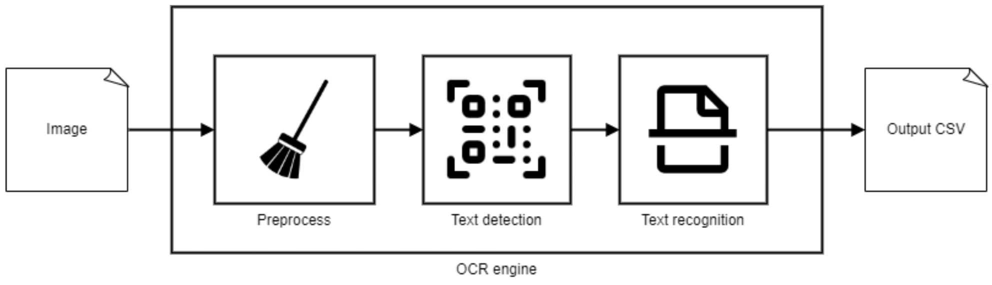


Figure 12: OCR engine

In this project three different approaches were considered to speed up development of a prototype: using a ready-built online API [22], using closed-source software or binaries, using and modifying open-source or creating new code. For the most control over the implementation and possibility to modify and analyse the code the last option with open code was chosen.

4.1.1 Pre-processing implementation

The first part of the implemented OCR engine is pre-processing which contains 3 separate steps to enhance the ability to detect and recognize text, as shown in figure 13. Pre-processing is the first contact the inbound image has with the engine. Simply this step has the input of a taken image and output is a modified image which is more suitable for text detection and text recognition.

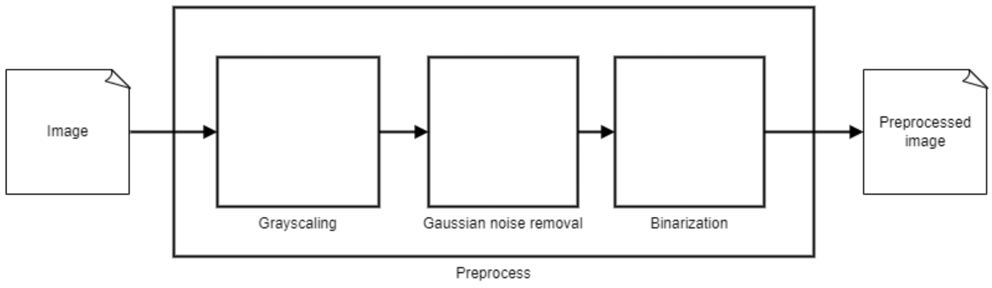


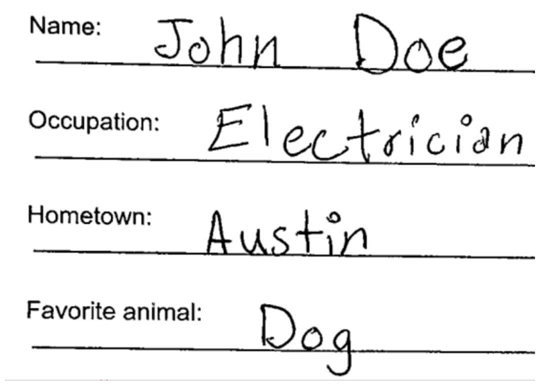
Figure 13: pre-processing

Colour in an OCR engine is not used as algorithms that detect and recognize text do so from 2D-arrays containing the intensity values.

Furthermore, running computations on all three colour channels (red, green, blue) increases computational costs with no benefit. The colour conversion implementation is using the OpenCV [6] colour space conversion functions which converts from RGB/BGR to grayscale which has been described in the theory chapter 2.2.

Images contains noise such as Gaussian noise and salt-and-pepper noise which both affect the ability to run any further detection and recognition on the image. To combat this the technique called Gaussian blurring was used which first blurs the image, then divide each pixel value with the blurred pixel value.

Lastly a thresholding algorithm called binarization is run on each pixel to either convert the pixel value to 0 (black pixel) or 255 (white pixel). This produces an image with strictly either black or white pixels with low to no noise as shown in 14.



Name: John Doe

Occupation: Electrician

Hometown: Austin

Favorite animal: Dog

Figure 14: Image after pre-processing

4.1.2 Text detection implementation

The neural network EAST (Efficient and Accurate Scene Text Detector) [23] was used for text detection for the reason of being both fast and accurate. First step is to prepare the image to fit the width and height into multiples of 32. The image is the sent through the EAST neural network which outputs geometry, also called bounding boxes, and a score for each box. This score is used in thresholding to remove low score boxes. Non-max suppression is used to remove overlapping boxes. Finally, the image is readjusted into original width and height. These five steps can be seen in figure 15. The values for the bounding boxes are sent to the next step in the pipeline.

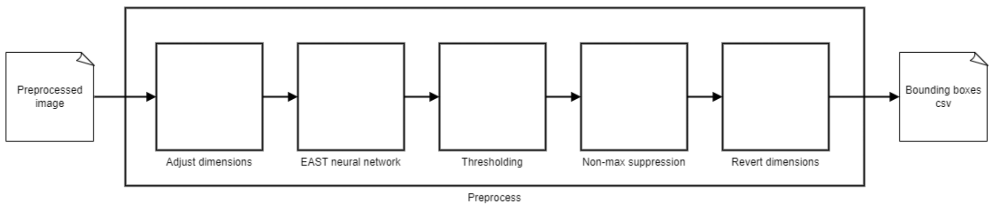


Figure 15: Text detection

An output image is also created for debugging and analysing purposes where each bounding box is visualized using a green rectangle over the pre-processed image as shown in 16.

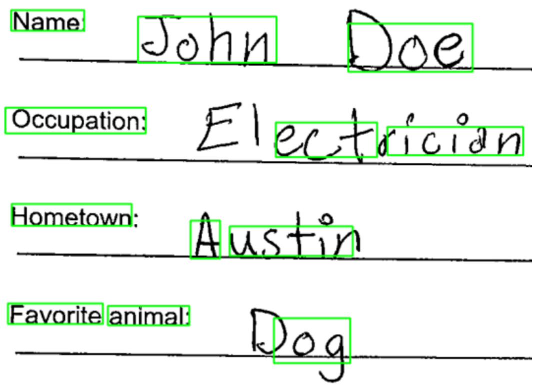


Figure 16: Bounding boxes visualized

4.1.3 Text recognition implementation

The image with bounding boxes is iterated over to recognize text in each box using a pre-trained convolutional recurrent neural network. Bounding boxes are used to cut sections of the image to be sent individually to the neural network for each to be recognized. The network produces a list for each character position with an ordered list containing the most likely character from the English alphabet and numerals in that position. In this implementation the most likely character was always chosen for that character position. For each bounding box an array of characters was predicted to be contained within. The output of this step is a list of box positions together with predicted text.

X	Y	text
222	136	name
471	166	johy
788	178	dioe
265	292	occupation
...	...	...

Table 1: Example of the text recognition output

4.2 Blank form clustering

This implementation inputs two images, one of which is an image of an unfilled form used to extract each word and cluster words that belong together. The first clustering is called blank form clustering in that it clusters words into individual or into other clusters. This is done using the prior steps of the OCR engine which results in blank form clustering receiving the box positions jointed with predicted text for each box. An example of a blank form with the bounding box for the boxes are drawn in green can be seen in figure 17.

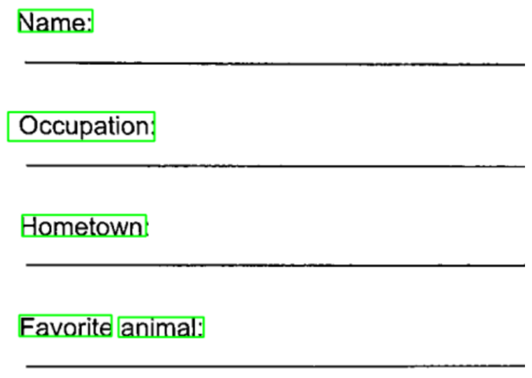


Figure 17: Example blank form

Clustering is first done by calculating the average distance between each box's midpoint and its closest neighbour. This is used as a reference point to determine if neighbouring boxes belong in the same cluster. Distance calculating is done using a modified variant of Euclidean distance which a higher weight on vertical distance than horizontal distance described in section 2.5.2. For another box to be contained in a cluster with another box they must be closer than the average distance with a certain threshold, current implementation uses 90% as threshold which means for a box to be clustered with another box it has to be closer than 90% of the mean. In the example of a blank form 17 the boxes containing "favorite" and "animal" got rightfully clustered together while the other boxes were put in individual clusters.

4.3 Filled form clustering

Clustering a filled form uses an image such as 16 where both field names and field values are present as shown in figure 18. Identified words from blank form clustering are used to verify which words in filled form are field names and not field values. The verification is through using a method called fuzzy string search [24]. As the words recognized in the blank and filled form are not always exactly equal e.g., "name" field name can get recognized as "names" due to semi-colon present. The function used is implemented in the standard library for Python called SequenceMatcher [25] which finds the longest contiguous matching sub-sequence between two strings.

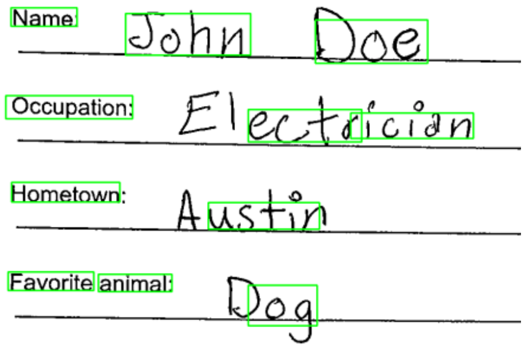

Name: John Doe
Occupation: Electrician
Hometown: Austin
Favorite animal: Dog

Figure 18: Example filled form

Secondly clustered field names get converted into a larger box which spans over the clustered boxes with its new midpoint being the new mid of the box. This is done to treat a multi-word field name as a single box in terms of position and recognized text.

Lastly a conventional clustering method is used where boxes in more common positions are preferred. This is done using a piecewise function with conditions if the field value box is horizontally to the right or left of field name box and if the field value box is vertically above or below of field name box described in section 2.5.3. In the code below the distance between an answer box (field value) and a question box (field name) is calculated which can be seen in appendix D

5 Result

This chapter presents the results of testing the systems accuracy on various types of forms. The system has been tested by inputting different images of handwritten filled in forms with the same image of a blank form. Furthermore, three different types of forms (left-oriented, right-oriented, down-oriented) were chosen to test a variety of different structures a form can follow. For each type of form 15 different images were taken and computed through the implementation. The input for each field value in the images are randomized to contain different words and different number of words.

5.1 Measurement

This project aims to evaluate if an automatic system to extract key-value pairs from forms is suitable to be used in practice through a quantitative measurement. The measurement will not focus on the degree of correctly recognized words nor the amount of correctly detected text as this depends on the OCR system in use which can be swapped or improved without any modification to the rest of the system. The measurements will solely focus on the degree of correctly clustered boxes which contain field value to the correct field name.

5.1.1 First measurement

First measurement is the amount of correctly number of clusters for boxes containing field names (Questions in the form) and the correct clustering for multi-word field names. The correctness of a multi-word field name will be determined by the degree of native contra non-native text boxes. In figure 19 the field name box containing 'Occupation' and 'Doctor' results in one correct and one wrong text box, while the field name box containing 'Favorite' and 'animal' results in two correct text boxes. The text boxes in the picture are 'Name', 'Occupation', 'Hometown', 'Favorite', 'animal', 'Doctor', these text boxes belong to one of 4 field name clusters. In total the picture has 5 correctly clustered text boxes and 1 wrongly clustered text box (Doctor does not belong to Occupation field name).

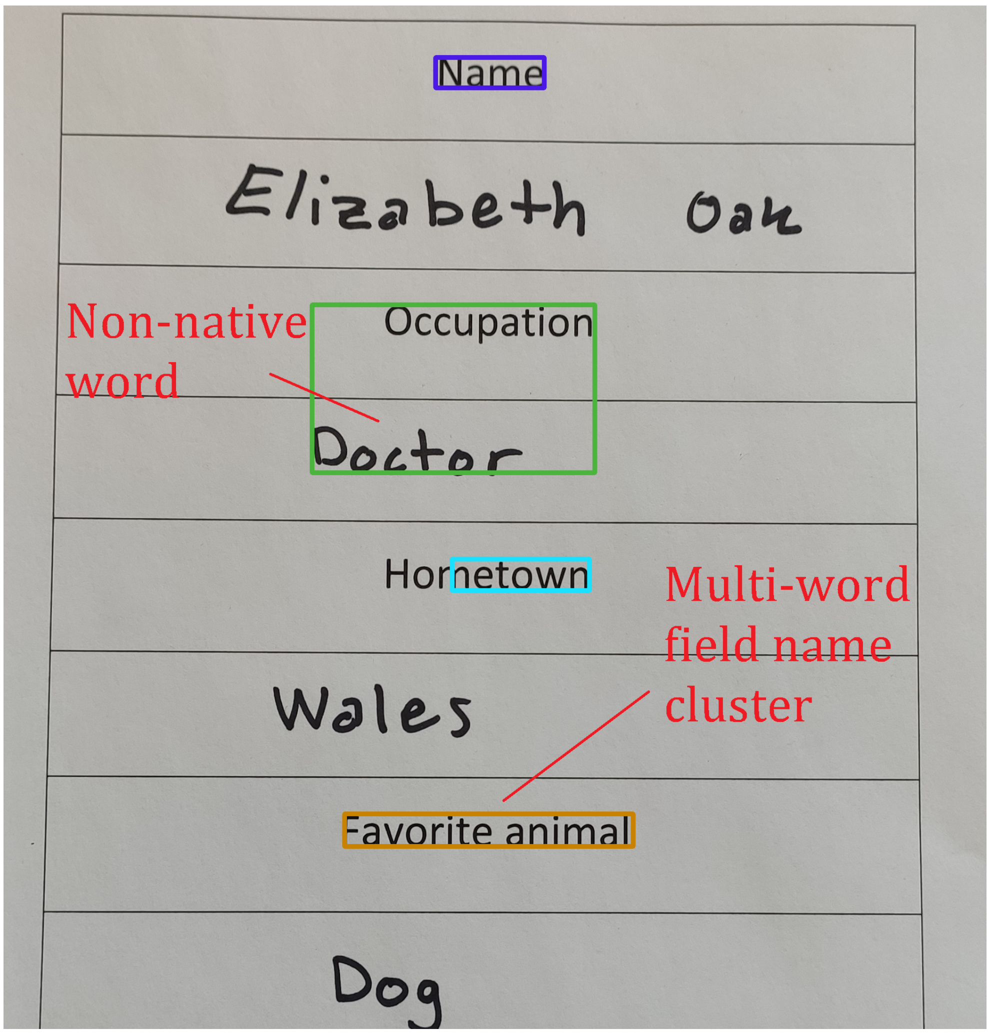


Figure 19: First measurement visualized

5.1.2 Second measurement

Second measurement is the percentage of boxes containing field values that is clustered to the correct field name box, where the order, box size, or recognized word is not considered. In figure 20 the field value box containing 'York' is wrongly clustered to 'Occupation' rather than 'Hometown. This in consideration to the other field value text boxes gives 8 correct and 1 wrongly clustered box.

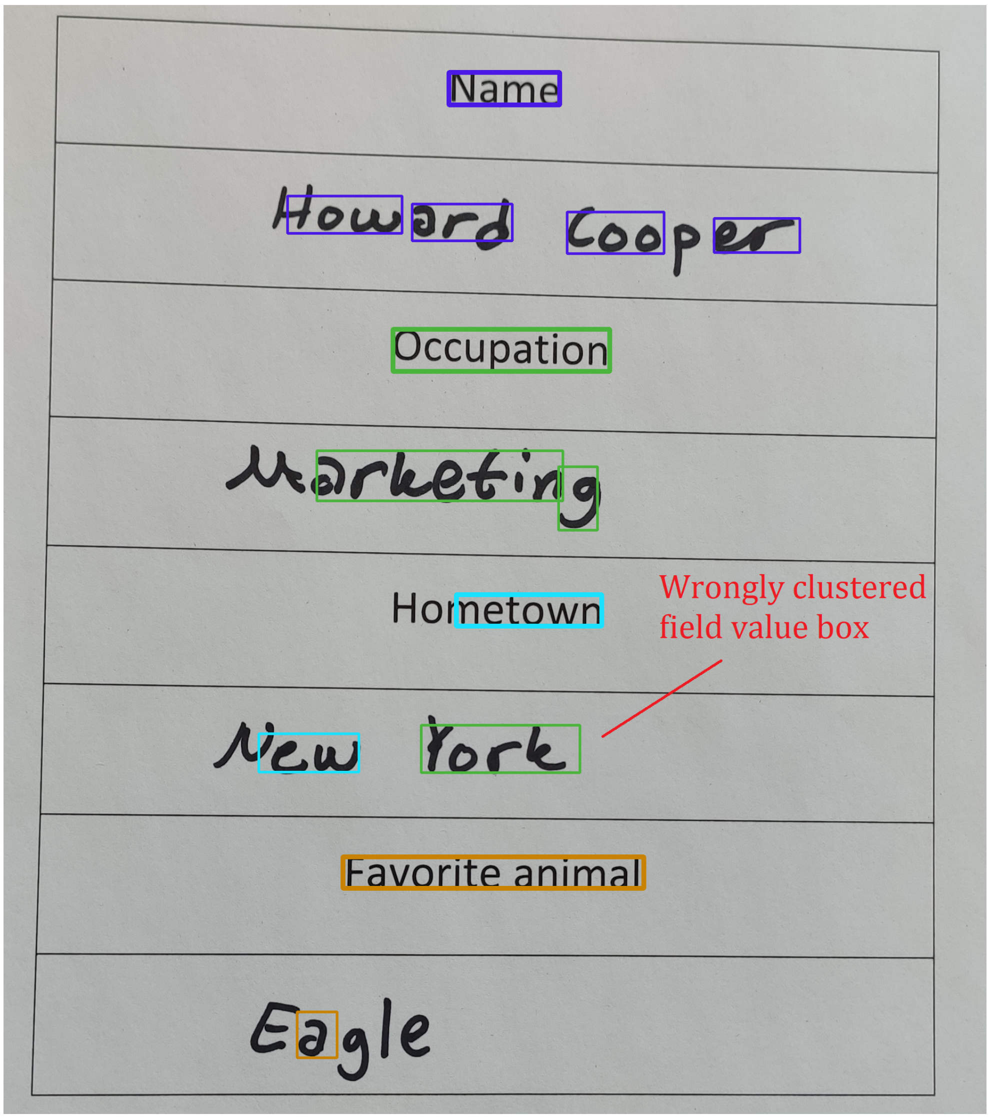


Figure 20: Second measurement visualized

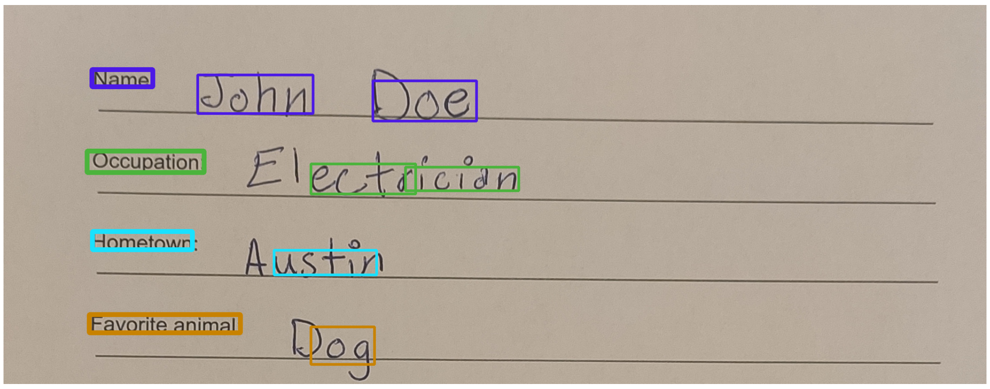
5.2 Accuracy results

Using the results of each form orientation the accuracy of correctly and wrongly clustered field names and field values can be derived as such. To calculate accuracy of each implementation the formula used was:

$$accuracy = \frac{\text{correctly clustered text boxes}}{\text{total amount of text boxes}}$$

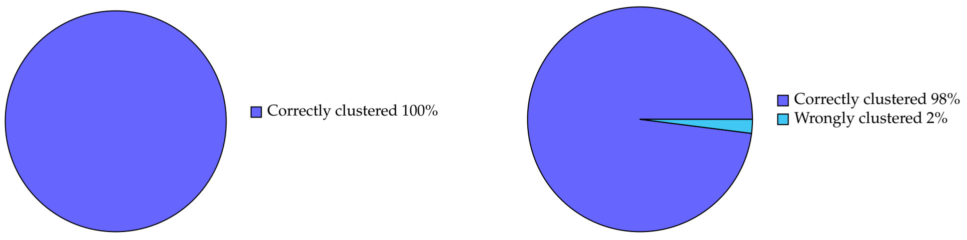
5.2.1 Left-oriented form

The left-oriented form follows the structure of most forms where the field name is present to the left of the area to input the field value. Sample of such form shown in 21 and the results of all left-oriented forms shown in 22.



Name	John Doe
Occupation	Electrician
Hometown:	Austin
Favorite animal	Dog

Figure 21: Sample of clustering a left-oriented form

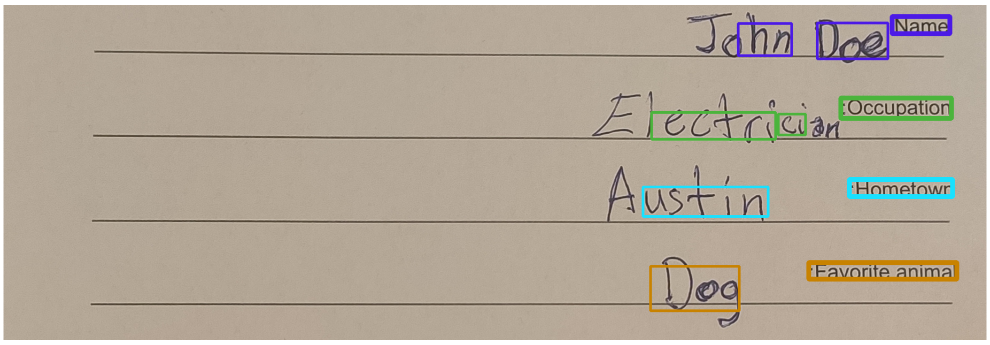


(a) Field name (b) Field value

Figure 22: Left-oriented form results from appendix A

5.2.2 Right-oriented form

The right-oriented form follows a structure not commonly used. This was done for testing purposes only with no practical usage as a goal. Sample of such form shown in 23 and the results of all left-oriented forms shown in 24.



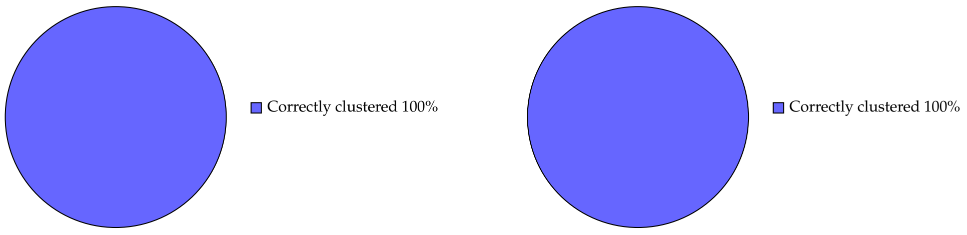
John Doe Name

Electrician :Occupation

Austin :Hometown

Dog :Favorite animal

Figure 23: Sample of clustering a right-oriented form

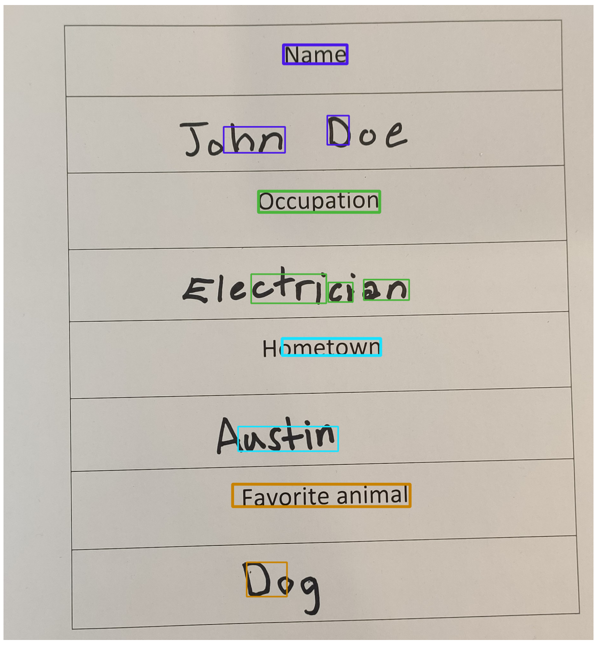


(a) Field name (b) Field value

Figure 24: Right-oriented form results from appendix B

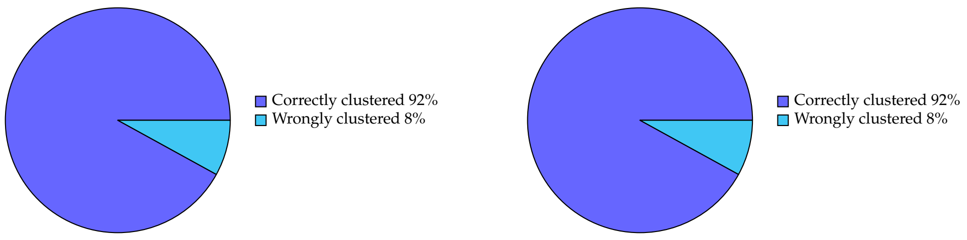
5.2.3 Bottom-oriented form

Bottom-oriented form is common for large open-space boxes for field values. Sample of such form shown in 25 and the results of all left-oriented forms shown in 4.



Name
John Doe
Occupation
Electrician
Hometown
Austin
Favorite animal
Dog

Figure 25: Results of clustering a bottom-oriented form



(a) Field name (b) Field value

Figure 26: Bottom-oriented form results from appendix C

6 Discussion

Here will the results be analysed while the methods, approaches, metrics, and milestones will be discussed. Furthermore, a scientific, ethical and societal discussion will also be presented.

6.1 Analysis and discussion of results

As shown in 5.2 the accuracy for different types of form structure differs. This shows clearly that the resulting application and solution has its strengths and weaknesses and is suitable for a subsection of form types. From the measurements we can conclude that solution performs better with left-oriented and right-oriented forms than bottom-oriented forms. The small difference of accuracy between left-oriented and right-oriented could be due to a small sample size (N = 15). To solve issue with low accuracy either the solution could be modified, or the form could be changed. Modifying the solution to better solve one of the form structures will in probability increase accuracy for that form but in return lower the accuracy for other form structures and as a result seize to be a solution for general purpose forms. Adapting the forms structure before reading it in as an image may also improve the accuracy, as attributes such as average text box distance and the intraclass cluster distance affect the results.

6.2 Project method discussion

The method used in this project will be discussed in both terms of thought behind the choices and in terms of the resulting impact of the choice.

6.2.1 Chosen method

Using the agile methodology for developing the project solution and thesis was critical to work in a limited time whilst also being able to produce a functioning solution. As problems using OCR may not be apparent at the start of the project it is important to get a working prototype and identify the weaknesses of OCR and the implemented solution which uses it. The problem that arose with an agile methodology was reaching the first working prototype took a lot more time than each future iteration.

6.2.2 Chosen approach

Approaching the problem two specific important decisions had to be made.

- Firstly, how to implement the OCR engine which outputs text box coordinates and recognized words. Three options were considered, firstly using an external API such as Azure OCR, see 2.7.5. Secondly using the Tesseract OCR engine, see 2.7.6, and thirdly implementing an own OCR engine using pre-trained neural network models. For this project the third option was chosen as an OCR engine for the reason of high code transparency and ability to modify as many parameters as possible. While the third option gave a deeper understanding to OCR, implementing the OCR engine proved to take longer time and have a lower accuracy than pre-made solutions.

- The second decision to approach the problem is what type of clustering algorithm to be used in the post-process step. Algorithms such as k-means and mean-shift were considered, however the time to implement the algorithm to the project problem and able to modify the algorithm to fit the project were considered too costly for this project, and instead implementing a custom Euclidean distance algorithm deemed more suitable.

6.2.3 Chosen metrics

Measuring the results of the solution were not straightforward as the images used for developing the solution and for measuring had no predefined correct solution to compare with. As the postprocess primarily works in two steps, firstly correctly clustering the field names and the correctly clustering the field values depending on the field names it was deemed suitable to measure both those parameters for accuracy. This was suitable for giving an overview for the effectiveness of the solution and its accuracy with different types of forms. In hindsight figuring out more metrics to measure would give a better insight into the degree of success for the prototype and chosen approach.

6.2.4 Goals

The two goals which were set which were both achieved by the technical solution. At the time of writing the goals a different type of clustering approach was considered but not pursued, which would allow for separately achieving goal 1 and then goal 2 in sequence. Such approach would focus on clustering first and secondly separation of field name and field value. However, the resulting approach works in reverse as the first step is separating the field name and then clustering the field names with field values, meaning goal 2 was reached before goal 1.

6.3 Scientific discussion

The main scientific knowledge gained from this work is proving that understanding the information present in forms read through OCR using clustering is possible without any pre-set configurations such as marking regions of interest on a form. Furthermore, a general understanding for OCR is presented in this thesis for the reader to better understand how OCR works and its limitations. As this thesis may be a steppingstone for further research into automatic data enrichment in OCR systems, the developed algorithms are too specific to be applied in a general field outside of structured forms.

6.4 Ethical and societal discussion

Ethically this project builds on previous technologies on the ethical aspects of those technologies. However, overlooking the underlying technologies and their ethical discussions, this new solution does not rely on previous data which could skew the results, nor does it save any data which could have been a privacy problem. In this regard, the solution is ethically sound without any doubt.

In terms of the societal discussion this solution is part of the OCR field which inherently impacts society in both positive and negative ways. As most uses of OCR allows for automation of simpler tasks which frees up time from manual work, helps people with disabilities that affect the ability to read and understand text. As most automation it could be argued that it takes away work from society leading to higher unemployment rate. This is however a false possibility of causation as the technology would not directly replace any work but only improve existing work, such as a data entry job.

As the risk of misreadings by an OCR engine and a postprocess clustering step is ever-present another postprocess step could be introduced to correct misreadings by the OCR engine through actions alike auto-correct. However, as no system is never completely foolproof, sensitive data should either be reviewed manually afterwards or never be read through an OCR application in firsthand and instead be transmitted digitally in machine-encoded text instead.

6.5 Problems with OCR

The main problem and which cripples any development of practical usage of data enrichment is the unreliability of OCR. Both in terms of text detection (finding text boxes) and text recognition (reading the text) the output various widely when used. As any step in the prototype relies on the steps before each degree of error not only propagates through the pipeline but gets exponentially larger with each degree of error. This was one of the problems down the road on the project, not taking advantage of well-engineered readymade solutions for as many steps in the pipeline as possible.

Various methods could be used to resolve this issue, both in the process of OCR with more error correction processes or simply a better OCR engine and on the physical form to create the best condition for OCR. One of the methods is to use an out-of-the-box ready solution as for example [17]

6.6 Indefinite forms

This implementation for form reading is able to read forms that has not be predefined using alignment features, lines, boxes, or any pre-sets. This is something that was really important when asking the scientific question and creating the implementation. A predefined form using boxes to designate regions of interest can be seen in figure 8.

Using an implementation for indefinite forms allows for both an easier creation and setup of forms and form collection, but also the possibility to read foreign forms which has not been created by the user. I think these two advantages can really save time especially if this work of reading indefinite forms is further researched on.

7 Conclusion

This work successfully to cluster field names and values in a form to a certain degree of accuracy (92% to 100%) depending on form structure. The overall aim of the project to provide an easy introduction of OCR and data enrichment together with its limitations is done through the report while the problem statement of examining feasibility of automatically extracting information is conducted through the project prototype.

While this project does not solve the problem of extracting information from forms completely, it provides a steppingstone into the direction of conventional clustering using techniques such as modified Euclidean distance and fuzzy string search.

7.1 Future Work

This project had to cut down on the amount of work and different approaches tried due to time constraints. Because of this constraint there is possibilities to further work on the project, and below is a list of a few of possible future work.

- A big outside factor of the solution accuracy depends on the OCR engine, further research into which OCR engine could provide a better accuracy is to be done in the future, possibly using the Azure form recognizer as OCR engine to offload problems with OCR in the project [17]
- Include increasing both the amount of different form structures as well as number of images to provide a better statistical result
- Using the aid of glyphs, lines, boxes, or markings to indicate the position of a grouping containing field name and field values. These aids could be used as anchor points, borders, additional weights for distance calculations, etc.
- As the Eucledian distance was calculated by using midpoints of each text box another solution could investigate the possibility of using points on the edges of text boxes for calculating the distance. This could provide a better result considering long and short words have drastically different midpoint.

7.2 Example of a future work

As this project used conventional algorithms to solve the problem a future work using technologies such as deep learning could provide a higher accuracy and function better as a general-purpose solution. This would be done by firstly creating data through manually inspecting images containing forms and creating accompanying files describing the relationship between field names and field values in the image. This data creation would provide the means necessary to establish a deep learning solution to the problem in terms of clustering names and values together.

References

[1] E. E. F. d'Albe. *On a type-reading optophone*. 1914. URL: https://royalsocietypublishing.org/doi/10.1098/rspa.1914.0061 (cit. on p. 1).

[2] Wikipedia. *History of writing — Wikipedia, The Free Encyclopedia*. 2022. URL: http://en.wikipedia.org/w/index.php?title=History%5C%20of%5C%20writing&oldid=1088553424 (cit. on p. 1).

[3] Medium. *Optical Character Recognition(OCR) — Image, Opencv, pytesseract and easyocr*. 2020. URL: https://medium.com/@nandacoumar/optical-character-recognition-ocr-image-opencv-pytesseract-and-easyocr-62603ca4357 (cit. on p. 4).

[4] Vijaysinh Lendave. *A guide to text detection and recognition using MMOCR*. 2022. URL: https://analyticsindiamag.com/a-guide-to-text-detection-and-recognition-using-mmocr/ (cit. on p. 4).

[5] Nicomsoft. *Optical Character Recognition (OCR) – How it works*. URL: https://www.nicomsoft.com/optical-character-recognition-ocr-how-it-works/ (cit. on p. 5).

[6] OpenCV team. *OpenCV modules*. URL: https://docs.opencv.org/3.4/index.html (cit. on pp. 5, 15, 17).

[7] OpenCV team. *Color conversions*. URL: https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html#color_convert_rgb_gray (cit. on p. 5).

[8] GRM. *What is OCR data extraction*. URL: https://www.grmdocumentmanagement.com/what-is-OCR-data-extraction (cit. on p. 6).

[9] Levity. *What is data extraction & how does it work?* URL: https://levity.ai/blog/what-is-data-extraction (cit. on p. 6).

[10] Raja Kalpana et al. *Template Filling, Text Mining*. URL: https://doi.org/10.1007/978-1-4419-9863-7_173 (cit. on p. 6).

[11] Fullsailpartners. *What you need to know about ocr and icr technologies*. URL: https://www.fullsailpartners.com/fspblog/what-you-need-to-know-about-ocr-and-icr-technologies (cit. on p. 9).

[12] Erik Fahlén. *Androidapplikation för digitalisering av formulär*. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-35623 (cit. on p. 10).

[13] Jonathan Manousian. *Digitalisering av handskrivna siffror på fysiska formulär*. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-39343 (cit. on p. 10).

[14] Engin Kirik. *Tolkning av handskrivna siffror i formulär*. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-41291 (cit. on p. 11).

[15] Abhishek Jindal and Mohd Amir. "Automatic classification of handwritten and printed text in ICR boxes". In: (2014) (cit. on pp. 11, 12).

[16] Microsoft. *What is Azure Form Recognizer?* URL: https://docs.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/overview (cit. on p. 13).

[17] Microsoft. *Form Recognizer FAQ*. URL: https://docs.microsoft.com/en-us/azure/applied-ai-services/form-recognizer/faq#how-is-form-recognizer-related-to-ocr (cit. on pp. 13, 30, 31).

[18] Lei Sun, Neta Haiby, Cha Zhang, et al. *Enhanced Table Extraction from documents with Form Recognizer*. URL: https://techcommunity.microsoft.com/t5/ai-cognitive-services-blog/enhanced-table-extraction-from-documents-with-form-recognizer/ba-p/2058011 (cit. on p. 13).

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, et al. *Mask R-CNN*. URL: https://arxiv.org/abs/1703.06870 (cit. on p. 13).

[20] Sepp Hochreiter and Jürgen Schmidhuber. *Long Short-term Memory*. Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735 (cit. on p. 13).

[21] Google. *Tesseract Open Source OCR Engine (main repository)*. URL: https://github.com/tesseract-ocr/tesseract (cit. on p. 13).

[22] Microsoft. *Computer vision*. URL: https://azure.microsoft.com/en-gb/services/cognitive-services/computer-vision/#overview (cit. on p. 16).

[23] Xinyu Zhou, Cong Yao, He Wen, et al. "EAST: An Efficient and Accurate Scene Text Detector". In: *CoRR* abs/1704.03155 (2017).

arXiv: 1704.03155. URL: http://arxiv.org/abs/1704.03155 (cit. on p. 17).

[24] Esko Ukkonen. "Algorithms for approximate string matching". In: *Information and Control* 64.1 (1985). International Conference on Foundations of Computation Theory, pp. 100–118. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(85)80046-2. URL: https://www.sciencedirect.com/science/article/pii/S0019995885800462 (cit. on p. 20).

[25] Python Software Foundation. *Helpers for computing deltas*. URL: https://docs.python.org/3/library/difflib.html (cit. on p. 20).

A Left-oriented results

Image number	Correct name	Wrong name	Correct value	Wrong value
1	5	0	6	0
2	5	0	5	1
3	5	0	4	0
4	5	0	4	0
5	5	0	5	0
6	5	0	4	0
7	5	0	6	0
8	5	0	6	0
9	5	0	5	0
10	5	0	6	0
11	5	0	5	0
12	5	0	4	0
13	5	0	4	0
14	5	0	4	0
15	5	0	5	0
Total	75	0	73	1

Table 2: Table of results using the left-oriented form

B Right-oriented results

Image number	Correct name	Wrong name	Correct value	Wrong value
1	5	0	6	0
2	5	0	7	0
3	5	0	4	0
4	5	0	5	0
5	5	0	5	0
6	5	0	4	0
7	5	0	4	0
8	5	0	4	0
9	5	0	4	0
10	5	0	4	0
11	5	0	3	0
12	5	0	4	0
13	5	0	4	0
14	5	0	4	0
15	5	0	4	0
Total	75	0	66	0

Table 3: Table of results using the right-oriented form

C Bottom-oriented results

Image number	Correct name	Wrong name	Correct value	Wrong value
1	5	0	7	0
2	5	0	7	0
3	5	1	8	2
4	5	0	5	0
5	5	0	6	0
6	5	0	7	0
7	5	0	7	0
8	4	1	2	3
9	5	0	7	0
10	5	0	6	0
11	4	2	3	2
12	5	1	6	0
13	5	1	5	0
14	5	0	8	1
15	5	0	10	0
Total	73	6	94	8

Table 4: Table of results using the bottom-oriented form

D Piecewise Euclidean distance modified in Python

```python
# Calculate distance between two text boxes
# question box = field name
# answer box = field value

# Set weights for Y-axis
if answer_box_pos[1] < question_box_pos[1]:
    y_weight = 2
else:
    y_weight = 0.5

# Set weights for X-axis
if answer_box_pos[0] < question_box_pos[0]:
    x_weight = 2
else:
    x_weight = 0.5

# Calculate weighted distance
    weighted_eucledian_distance = math.sqrt(
        pow((answer_box_pos[0] - question_box_pos[0])*
                                x_weight, 2)
        + pow((answer_box_pos[1] - question_box_pos[1])*
                                y_weight, 2)
    )
```