UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Optical Character Recognition for Know Your Customer solution

Author:
Oleksandr VAITOVYCH

Supervisor:
Andrii KLESOV

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2021

Declaration of Authorship

I, Oleksandr VAITOVYCH, declare that this thesis titled, "Optical Character Recognition for Know Your Customer solution" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Optical Character Recognition for Know Your Customer solution

by Oleksandr VAITOVYCH

Abstract

KYC stands for Know Your Client. In other words it is a set of rules that help financial companies to check potential clients. The main purpose is to collect enough information about an individual to detect whether he/she would do some illegal actions like money laundering. Usually KYC takes place in banks offline. The customer should be present and have some documents. Often a passport or ID card is enough but this procedure can vary and depends on a particular company and services that person wants to get. The point of KYC is to process documents, check if they are genuine and belong to the customer, and whether the customer has everything to get a needed service. This work researches the KYC verification, using Tesseract and Google OCR approaches. Here is the demonstration link.

Acknowledgements

First of all, I would like to thank my supervisor Andrii Klesov for supervising my work, sharing his knowledge and giving valuable advises.

I am grateful to my family for their support through all the studying.

I am also thankful to Ukrainian Catholic University and to my classmates for an interesting studying process.

Contents

List of Figures

List of Tables

List of Abbreviations

MRZ	**M**achine **R**eadable **Z**one
OCR	**O**ptical (it) **C**aracter **R**ecognition
SaaS	**S**oftware **as a** **S**ervice
KYC	**K**now **Y**our **C**lient
CNN	**C**onvolutional **N**eural **N**etwork
YOLO	**Y**ou **O**nly **L**ook **O**nce
SSD	**S**ingle **S**hot **D**etector
LSTM	**L**ong **S**hort **T**erm **M**emory
OSD	**O**orientation **S**cript **D**etection
RGB	**R**ed **G**reen **B**lue

Chapter 1

Introduction

The traditional way of KYC process is the following: you come to a bank or another financial company in order to get some service, you give them needed documents, the manager should put your data into a database, check it's consistency, validity, legality, check whether they are really yours. All this is done manually.

Nowadays, in the technological epoch, time is our main resource, and the process which requires a client to spend a lot of time going to the bank and then turning back to home should be interchanged in all the institutions. Now, in the Covid-19 time, the idea of transferring KYC to the Internet sounds as relevant as ever before. An online KYC solves three issues simultaneously: people can go through a process from home, which saves their time for the ride and their health due to a smaller chance to get infected. It is also useful for bank workers. In the online KYC process, they will not need to put the customer data into databases manually. The machine will do it by itself.

The aim of this work was to implement a document recognition and scoring part. For this, I used Tesseract and Google Vision OCR engines in pair with regular expressions. I also implemented something similar to key-value pair extraction [4]. For the scoring part, I used Convolutional Neural Networks [14]. The advantages of my approach compared to other existing ones is easy to use and to integrate. It is also offered together with other services for financial companies like face anti-spoofing and anti-money laundering. All these services in one make infrastructure building much easier, eliminating the need to communicate with multiple providers and creating some programs to make services work together.

Chapter 2

Related works and background information

2.1 OCR

OCR stands for Optical character recognition. In other words, it is a process of extracting text from images of text. It is widely used in our everyday life and has various applications like digitizing printed texts, automatic number plate recognition, book scanning, real-time translation, document recognition, information extraction and many others.

Nowadays, in general, OCR is a well-studied task, and modern engines show very high recognition accuracy. M. Ryan and N. Hanafiah in [6] used template matching for character recognition. N. Thanh Cong, N. Dinh Tuan and T. Quoc Long used Convolutional recurrent neural network to recognize sequence of unknown length on the image [7]. W. Satyawan, M Octaviano Pratama, R. Jannati, G. Muhammad, B. Fajar, H. Hamzah, R. Fikri, K. Kristian in [13] implemented Tesseract-based approach. Besides a standard deep learning approach with detectors for each character like YOLO or SSD, there exist novel approaches with attention mechanism [15].

2.2 Tesseract

Tesseract is an open-source LSTM-based OCR engine. It was developing by Hewlett-Packard from 1984 to 1995 [9]. After that, it was bought by Google in 2006 to continue development. It supports more than 100 languages and various data formats. It also supports the usage of multiple languages. The latest version of Tesseract uses LSTM models, but there is a possibility to use the legacy version, which recognizes character patterns. Here is the list of possible page segmentation models [9]

0. Orientation and script detection (OSD) only.
1. Automatic page segmentation with OSD.
2. Automatic page segmentation, but no OSD, or OCR. (not implemented)
3. Fully automatic page segmentation, but no OSD. (Default)
4. Assume a single column of text of variable sizes.
5. Assume a single uniform block of vertically aligned text.
6. Assume a single uniform block of text.
7. Treat the image as a single text line.

8. Treat the image as a single word.
9. Treat the image as a single word in a circle.
10. Treat the image as a single character.
11. Sparse text. Find as much text as possible in no particular order.
12. Sparse text with OSD.
13. Raw line. Treat the image as a single text line, bypassing hacks that are Tesseract-specific.

For my task, Sparse text model(number 11) was the most appropriate one because ID carts do not have some line structure across all the image. I also used a single language at once because of better accuracy.

2.3 Google Cloud OCR

This engine developed by Google currently achieves the best accuracy among all OCR engines. Google cloud vision OCR is not open source and charges fees for the usage. Also, google OCR is a cloud service, so it is able to work with an internet connection only. There is a small amount of information about how this engine works, but I found one article [5] which gives a high-level explanation.

2.4 Morphological filters

Morphological filters [12] are applied to binary images in order to shrink or grow some regions, connect or disconnect some close groups of pixels.

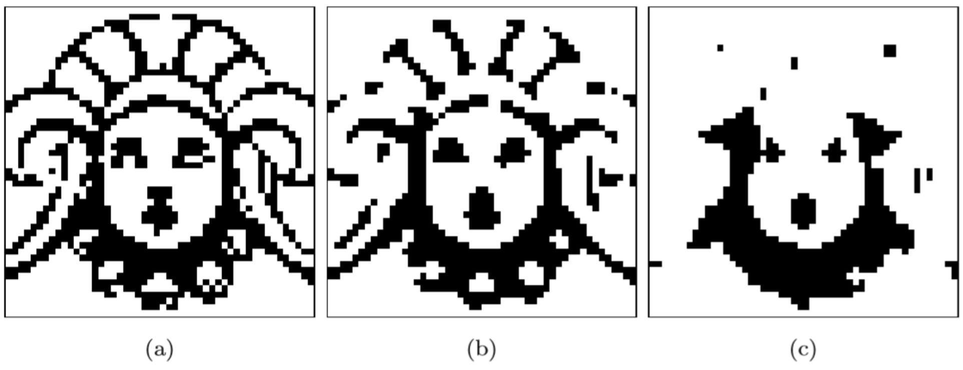


FIGURE 2.1: Example of morphological filters usage. Source [12]

There are four operations - erosion, dilation, opening, and closing. Where erosion stands for a shrinking process,

$$I \ominus H \equiv \{ \boldsymbol{p} \in \mathbb{Z}^2 \mid (\boldsymbol{p}+\boldsymbol{q}) \in I, \text{ for every } \boldsymbol{q} \in H \}.$$

FIGURE 2.2: Erosion formula. Source [12]

$$I \oplus H \equiv \big\{(\boldsymbol{p}+\boldsymbol{q}) \mid \text{for every } \boldsymbol{p} \in I, \boldsymbol{q} \in H\big\}.$$

FIGURE 2.3: Dilation formula. Source [12]

dilation stands for a growing process,
opening stands for an erosion followed by dilation,

$$I \circ H = (I \ominus H) \oplus H.$$

FIGURE 2.4: Opening formula. Source [12]

and closing stands for a dilation followed by erosion [12].

$$I \bullet H = (I \oplus H) \ominus H.$$

FIGURE 2.5: Closing formula. Source [12]

2.5 Lab color space

Lab color space is considered to be the most accurate color space[img]. L stands for lightness, a for green to red ratio, and b for blue to yellow ratio. One more exciting thing is that Lab color space is device independent, so it is easier to get the same color on different devices [2].

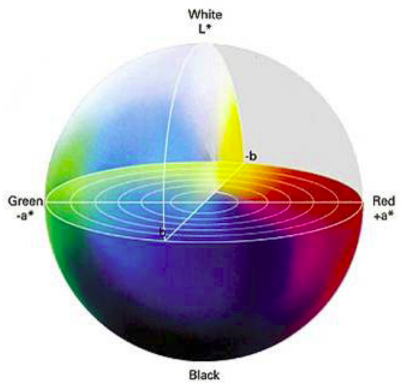


FIGURE 2.6: Lab color space specifies a color using a 3-axis system [2]. Source

2.6 Adaptive Thresholding

Adaptive thresholding is used for transforming RGB or grayscale images into binary ones. Instead of using global thresholding where all pixels are modified by a single threshold for the whole picture, adaptive thresholding assigns threshold value locally. It can be very different for some parts. There are two main approaches [8]: the first is to find a threshold for each pixel and the second is to split an image into overlapping areas, threshold it, and interpolate the results.

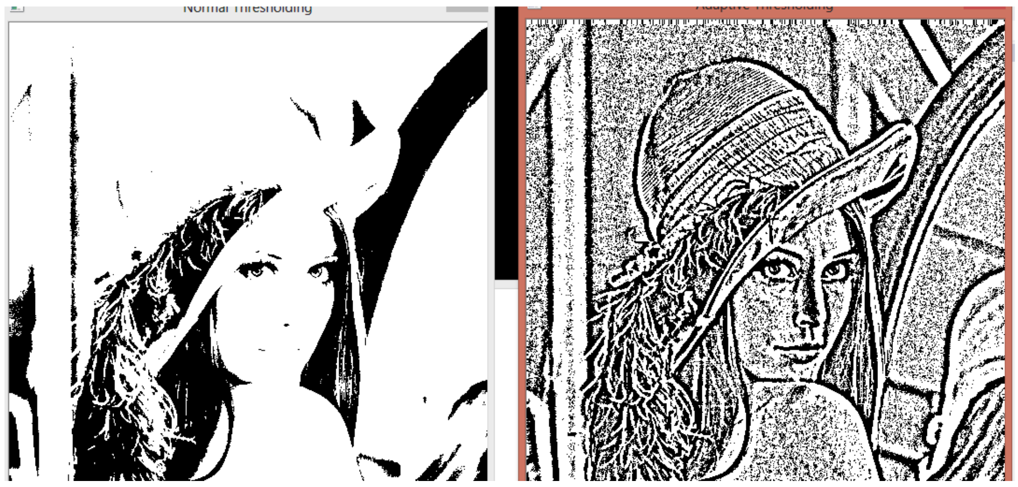


FIGURE 2.7: Visual difference between global thresholding (left) and adaptive thresholding (right). Source

2.7 Otsu thresholding

Otsu thresholding is a particular case of adaptive thresholding. It finds an optimal threshold value by going through all possible ones and choosing a value with minimal within-class variance. It showed the best results in image preprocessing for OCR. The work of Sunil L. Bangare, Amruta Dubal, Pallavi S. Bangare and Dr. S. T. Patil provides more evidence in this [10].

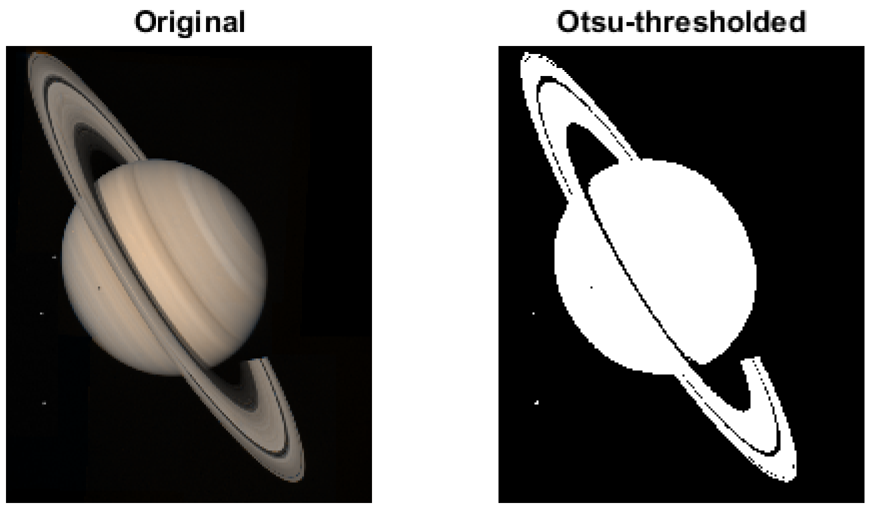


FIGURE 2.8: Example of Otsu thresholding. Source

2.8 Levinshtein distance

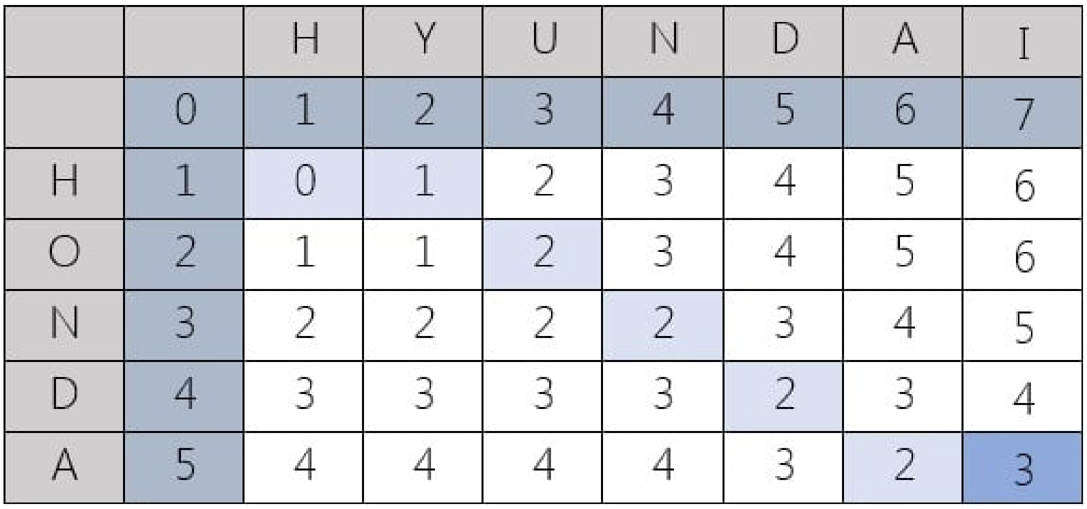
Levenshtein distance[3] is a popular metric to measure the difference between two strings. It calculates the smallest number of edits needed to transform one string to another.

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j) + 1 \\ \operatorname{lev}_{a,b}(i,j-1) + 1 \\ \operatorname{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

FIGURE 2.9: Formula for calculating Levinshtein distance. Source

There are three edit operations - substitution, insertion, and deletion. Here is a dynamic programming approach [3]:

1. A matrix is initialized measuring in the (m, n) cell the Levenshtein distance between the m-character prefix of one with the n-prefix of the other word.
2. The matrix can be filled from the upper left to the lower right corner.
3. Each jump horizontally or vertically corresponds to an insert or a delete, respectively.
4. The cost is normally set to 1 for each of the operations.
5. The diagonal jump can cost either one, if the two characters in the row and column do not match else 0, if they match. Each cell always minimizes the cost locally.
6. This way the number in the lower right corner is the Levenshtein distance between both words.



		H	Y	U	N	D	A	I
	0	1	2	3	4	5	6	7
H	1	0	1	2	3	4	5	6
O	2	1	1	2	3	4	5	6
N	3	2	2	2	2	3	4	5
D	4	3	3	3	3	2	3	4
A	5	4	4	4	4	3	2	3

FIGURE 2.10: Example of Levinstein distance between "Honda" and "Hyundai" Source

2.9 Polyglot lib

Polyglot is a lib for doing operations with text in multiple languages. It supports [1]:

- Tokenization (165 Languages)
- Language detection (196 Languages)
- Named Entity Recognition (40 Languages)
- Part of Speech Tagging (16 Languages)
- Sentiment Analysis (136 Languages)
- Word Embeddings (137 Languages)
- Morphological analysis (135 Languages)
- Transliteration (69 Languages)

I used this lib for transliteration between English and other languages. The project has a very simple interface and supports all European languages. Here are examples for French and Ukrainian to English transliterations:

```
>>>from polyglot.transliteration import Transliterator

>>>transliterator = Transliterator(source_lang="fr", target_lang="en")

>>>print(transliterator.transliterate(u'Anaïs'))

anais
```

FIGURE 2.11: French to English transliteration

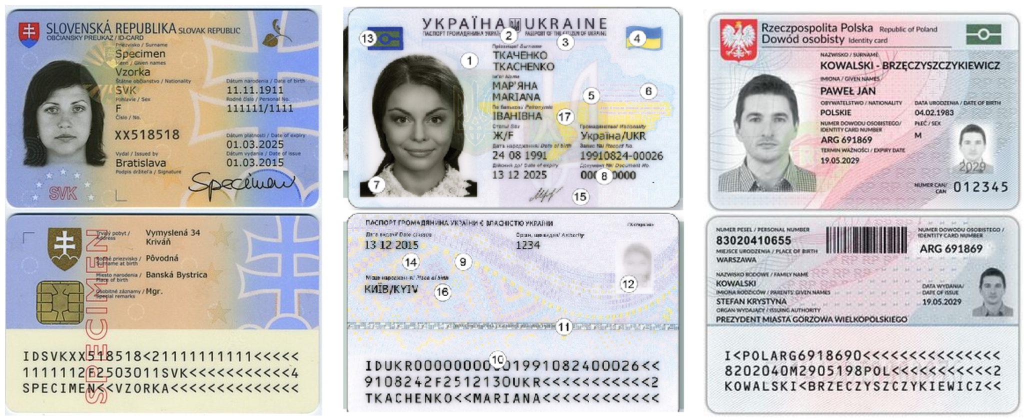
```
>>>from polyglot.transliteration import Transliterator

>>>transliterator = Transliterator(source_lang="uk", target_lang="en")

>>>print(transliterator.transliterate(u'Андрій'))

andrius
```

FIGURE 2.12: Ukrainian to English transliteration

Chapter 3

Data description

3.1 Document structure


SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID-CARD
Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature
Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015
SVK

Trvalý pobyt / Address
Vymyslená 34
Kriváň
Rodné priezvisko / Surname at birth
Pôvodná
Miesto narodenia / Place of birth
Banská Bystrica
Osobitné záznamy / Special remarks
Mgr.
SPECIMEN

```
IDSVKXX518518<21111111111<<<<<
1111112F2503011SVK<<<<<<<<<<<<4
SPECIMEN<<VZORKA<<<<<<<<<<<<<<<
```

УКРАЇНА UKRAINE
ПАСПОРТ ГРОМАДЯНИНА УКРАЇНИ PASSPORT OF THE CITIZEN OF UKRAINE
Прізвище/ Surname
ТКАЧЕНКО
TKACHENKO
Ім'я/ Name
МАР'ЯНА
MARIANA
По батькові/ Patronymic
ІВАНІВНА
Стать/ Sex
Ж/F
Громадянство/ Nationality
Україна/UKR
Дата народження/ Date of birth
24 08 1991
Запис №/ Record No.
19910824-00026
Дійсний до/ Date of expiry
13 12 2025
Документ №/ Document No.
000000000

ПАСПОРТ ГРОМАДЯНИНА УКРАЇНИ Є ВЛАСНІСТЮ УКРАЇНИ
Дата видачі/ Date of issue
13 12 2015
Орган, що видав/ Authority
1234
Місце народження/ Place of birth
КИЇВ/KYIV

```
IDUKR000000000<01991082400026<<
9108242F2512130UKR<<<<<<<<<<<<2
TKACHENKO<<MARIANA<<<<<<<<<<<<<
```

Rzeczpospolita Polska Republic of Poland
Dowód osobisty Identity card
NAZWISKO / SURNAME
KOWALSKI - BRZĘCZYSZCZYKIEWICZ
IMIONA / GIVEN NAMES
PAWEŁ JAN
OBYWATELSTWO / NATIONALITY
POLSKIE
DATA URODZENIA / DATE OF BIRTH
04.02.1983
NUMER DOWODU OSOBISTEGO/ IDENTITY CARD NUMBER
ARG 691869
PŁEĆ / SEX
M
TERMIN WAŻNOŚCI / EXPIRY DATE
19.05.2029
2029
NUMER CAN/ CAN
012345

NUMER PESEL / PERSONAL NUMBER
83020410655
MIEJSCE URODZENIA / PLACE OF BIRTH
WARSZAWA
NUMER DOWODU OSOBISTEGO / IDENTITY CARD NUMBER
ARG 691869
NAZWISKO RODOWE / FAMILY NAME
KOWALSKI
IMIONA RODZICÓW / PARENTS' GIVEN NAMES
STEFAN KRYSTYNA
DATA WYDANIA/ DATE OF ISSUE
19.05.2029
ORGAN WYDAJĄCY / ISSUING AUTHORITY
PREZYDENT MIASTA GORZOWA WIELKOPOLSKIEGO

```
I<POLARG6918690<<<<<<<<<<<<<<<
8202040M2905198POL<<<<<<<<<<<2
KOWALSKI<BRZECZYSZCZYKIEWICZ<<
```

FIGURE 3.1: Example of Slovakian (left), Ukrainian (center) and Polish (right) ID cards

All the documents have a similar structure. Every ID card has standard fields like name, surname, date of birth, etc. However, some of them also have specific areas like height, eyes color, Parents' names. In this project, I focused on the mutual fields only and haven't processed specific ones due to their redundancy in the KYC systems. The critical difference between ID cards is a field's font size and a field name's font size. The best case is when the field is in lowercase, and the field name is in an uppercase or vice versa. If both are the same case, it becomes harder to distinguish them. Documents with such a feature showed worse results. One more valuable part of ID cards is an MRZ (Machine Readable Zone). Usually, it is three lines of text at the bottom of the back page of the card. It contains a mark "10" on the Ukrainian ID card (Fig 3.1). This zone was created for better readability for the machines and had almost all primary information about the document. It always has a neutral background and is recognized with a high accuracy.

3.2 Preprocessing

In Tesseracts paper [9], there is a mention that this engine works best with binary black-on-white text like any other OCR system. Since all the input images are colored, they should be preprocessed.

There are many ways of image preprocessing, but I chose the most popular ones:

- Morphological filters

  - Erosion
  - Dilation
  - Opening
  - Closing
- Blurring
  - Average blurring
  - Gaussian blurring
  - Median blurring
  - Bilateral blurring
- Thresholding
  - Adaptive thresholding
  - Otsu thresholding
- Equalizing

I tried all the combinations of these methods with different filter sizes. Here are several examples

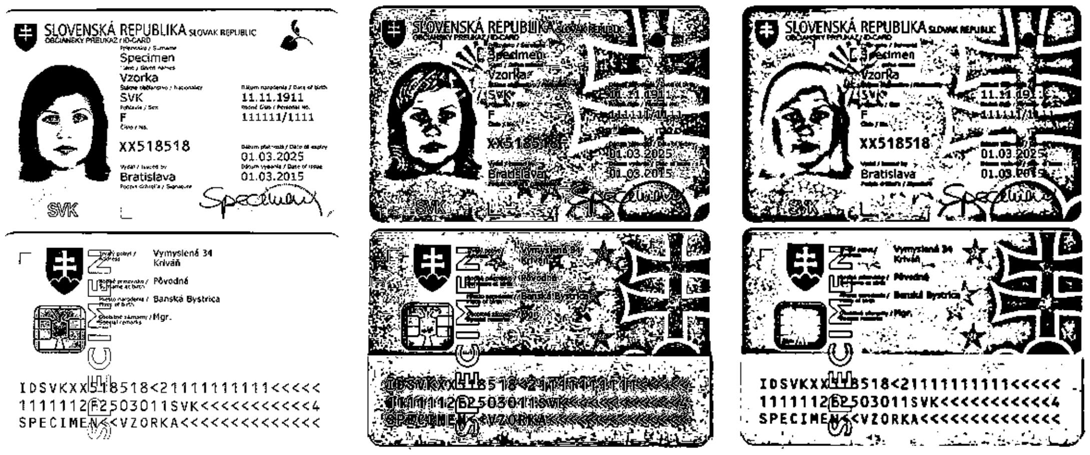


FIGURE 3.2: Example of preprocessed image

All the pictures were converted to grayscale at the beginning of processing.

The picture on the left was binarized with Otsu threshollding.

The central one was equalized, blurred and binarized with adaptive thresholding.

The rightmost picture equalized, blurred, equalized one more time and thresholded with adaptive thresholding.

I tested the results by comparing them with actual text and considering confidences from the Tesseract engine. Simple grayscale images showed the best results by a small margin.

Chapter 4

Methodology

4.1 Parsing

At the initialization of the OCR instance, we should choose an engine: Tesseract or Google OCR. On the one hand, Google OCR works significantly better, but it is a SaaS (Software as a Service) product that needs an internet connection and should be paid for. On the other hand, the Tesseract engine is open-source, free-to-use, installed locally, and works in a secure environment.

I have started with a Tesseract-based approach, and its possibilities were enough for the images of relatively good quality. Otherwise, Google OCR is preferred. Another difference between engines is that Google OCR detects text languages and structure automatically and outputs text as it is in the document. For instance, the Ukrainian card has two languages: Ukrainian and English, so that the output will contain words in both languages. However, Tesseract needs to get language and text structure as parameters. This engine can also work with several languages simultaneously, but the results of this approach are much worse than with a single language. So, in the case of using Tesseract, I have to extract text two times for both language versions. For instance, it should be extracted for both English and Ukrainian versions. ID cards have two sides which I processed sequentially - front page and then back page.

4.2 Frontpage

4.2.1 Removing redundant parts

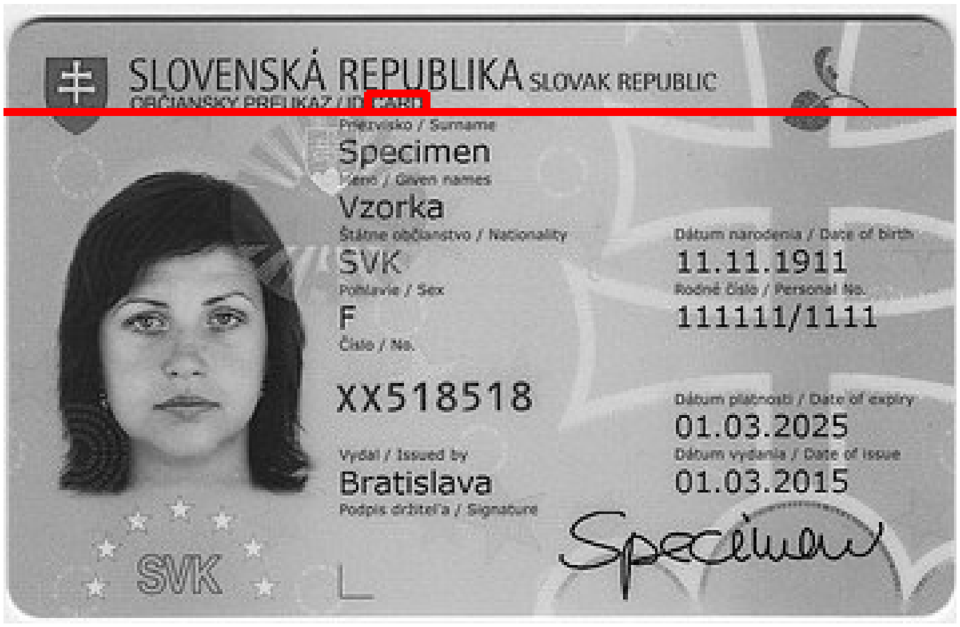

SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID CARD
Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature
Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015
SVK

FIGURE 4.1: Position of "CARD" in a bouding box. The part above the red line should be removed

All the cards contain pieces of information that are useless for my approach. Appropriate fields are usually situated below the "Surname" position. Thus, all the text above, such as ’Passport’, ’Ukraine’ (for Ukrainian ID card), could be removed. To execute this, I parsed both texts by regular expressions and found the last word before the "surname" field. For English text, it usually is a "CARD". For non-English text, it may vary.

4.2.2 Best match for "Surname"

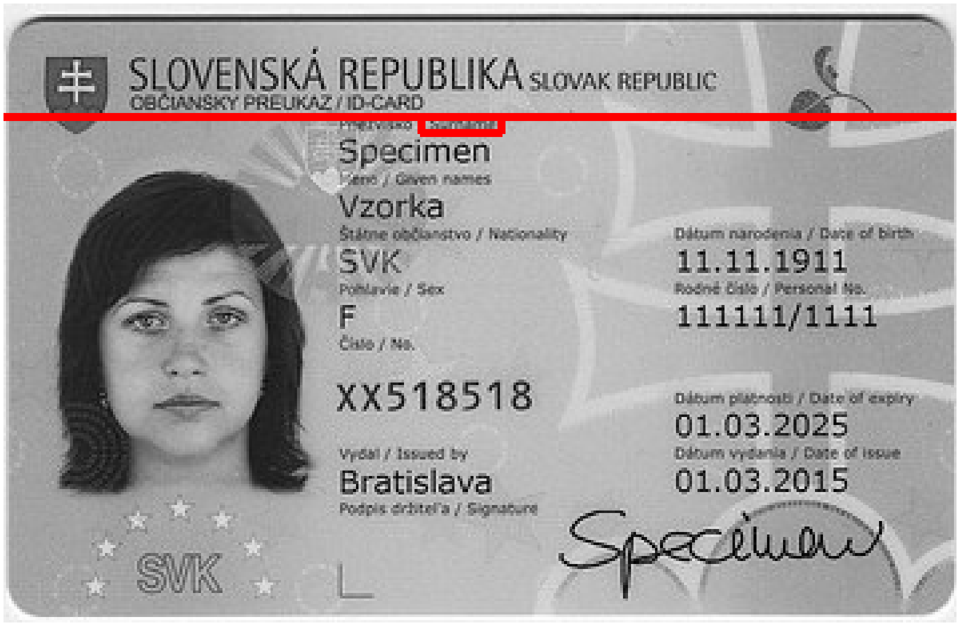
The next step has the same idea as the previous one but different implementation. The main purpose remained the same - cut the upper part of the text, which is useless. Occasionally, the approach of cutting images by a single constant word may work appropriately. Nevertheless, it strongly depends on the recognition quality and accuracy. In order to increase the chance of removing the upper part of texts, I used an approach that finds not a precise word but the best existing match, based on Levenshtein distance.

The idea was the following: find the closest match to the word "Surname" in English text or analog in another language and cut off everything above that word. At first, the text should be split into lines, then Levinstein distance calculated between the word and every line. When the top five matches are found, we move to the next step - partial matches. It uses the same Levinstein distance but between their parts instead of the distances between the whole words. Those parts match sequences of characters in both words supplemented to the length of the shorter of two compared words. Example:

- Word1: qwerty

- Word2: 111wert11

Both words contain the “wert” part. The distance is calculated between the shorter word and part of the longer one of the same length. So, the distance between “qwerty” and “1wert1” is calculated. If there were several such blocks, the result would be the smallest distance.



SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID-CARD

Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature
[illegible]

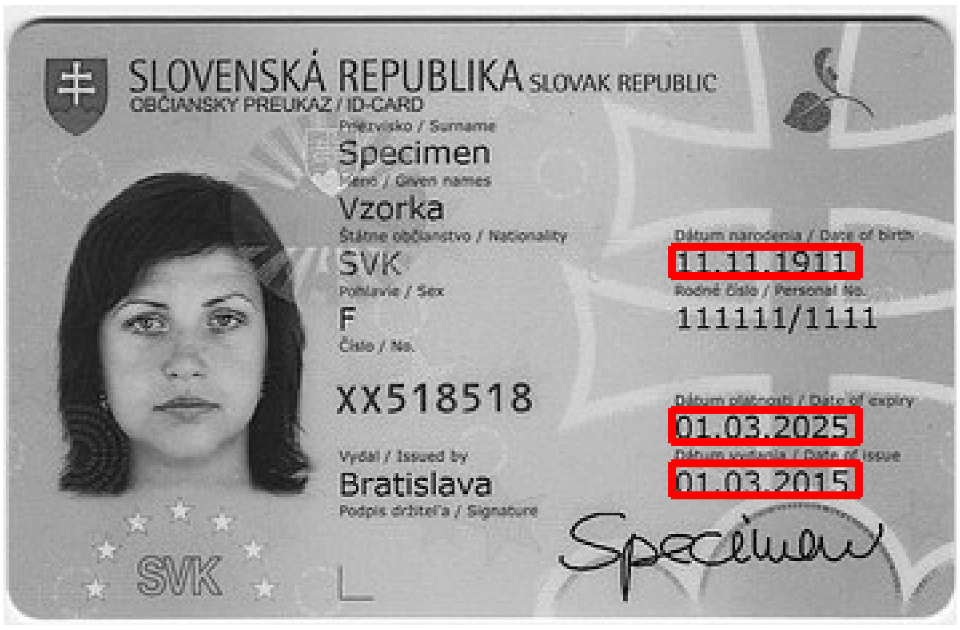
Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015

SVK

FIGURE 4.2: Position of "Surname" in a bouding box. The part above the red line should be removed

After getting full and partial matches, a match with the smallest distance among all is chosen as the best match. If the distance between the match and the matched word is lower than the threshold, the text is trimmed to the beginning of the match.

4.2.3 Dates



SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID-CARD

Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature

Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015

Specimen

SVK

FIGURE 4.3: Dates are in bounding boxes

Among all the ID cards, dates could be in ten formats: Examples for the first January of 2020:

- 01.01.20
- 01.01.2020
- 01/01/20
- 01/01/2020
- 01 01 20
- 01 01 2020
- 01 Jan 2020
- 01 Jan 20
- 01 JAN/JAN 20
- 01 JAN/JAN 2020

Usually, there is only one date format on the particular document, but rarely there could be two. Recognition is not always accurate, so there could be some additional spaces or digits. After parsing with regular expressions, all the dates are processed to remove double spaces if it is split by space or to remove spaces at all if it is split by dots or slashes. The dates with a month abbreviation are converted into a format “dd mm YYYY”. After that step, the dates are assigned to date fields of birth, issue, and expiry based on year.

4.2.4 Entities

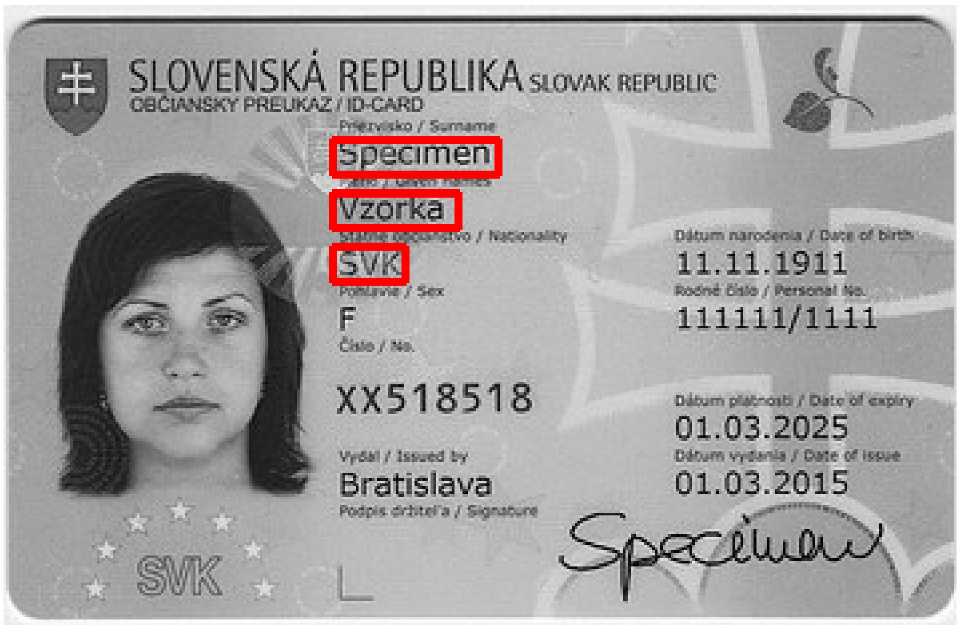

SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID-CARD
Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature
Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015
SVK

FIGURE 4.4: Entities are in bounding boxes

One of the most important things to extract are entities. Usually, they are 3 fields:

1. surname
2. name
3. nationality

But for countries with non-Latin languages, there can be four fields:

1. surname in English
2. surname in that language
3. name in English
4. name in that language

Moreover, for the Ukrainian ID card, it has a length of 5:

1. surname in Ukrainian
2. surname in English
3. name in Ukrainian
4. name in English
5. patronymic

In different documents, entities could be uppercase or lowercase with a capital letter. Usually, documents with uppercase entities achieve better results because it is easier for the engines to recognize more distinguishable uppercase letters. That entities, or candidates to be entities, are extracted by regular expressions, forming an entities list using just a word structure. For instance, it could be a word with all uppercase letters with leading space and ending space. In some cards, it is enough because there are absent uppercase words but entities. However, it is a rare case. This approach is often confused with some words like "REPUBLIC". Undoubtedly, in a perfect case, it should be removed by cutting at the beginning of processing, but it is not always possible due to recognition quality.

Moreover, if some particular entities are lowercase, they could be confused with words like "Name". To avoid it, I used stop words - a list with potentially confusing words. The stop words list is divided into two parts: the first one contains words that could be met only before the very first entity, the second one includes words that could be met between entities.

There is also an allowed words list for each document. It contains words with a small Levenshtein distance to some of the stop words, which could be treated so. For instance, it could be the word "POLSKIE" from the Polish ID card example (Fig 3,1). It is the third entity but could be confused with "POLSKA", which is in a stop words list.

As one may see in the Polish example, the entity is not necessarily a single word. Some people may have a double or even triple name or surname — that is why entities can contain spaces. Though, along with the name of multiple words, there can be a name with leading useless words or stop words. To avoid removing this kind of entry in the future, I checked every word of the entity for being a stop word and removed this word if it was present in the stop words list. Then I filtered out the words of length two and of length three if they do not have three different characters, ones with no uppercase letter. Lowercase entities have a capital letter, so it is acceptable for all cases. After that, I looped through the entities list and marked every word that has a smaller Levenshtein distance than the threshold with words from the first part of stop words. Then I deleted all entries from the entities list before the index of the last marked word. I executed this because all the words from the first part of the stop words list are definitely situated before the first entity, so they should be excluded. The next step was to filter the entities list with the second part of stop words to remove the remaining redundant findings. In the end, the first three entities from the entities list have been taken as a surname, name, and nationality (it also could be four or five entities in some cases). All those steps were executed for both languages.

4.2.5 Additional fields

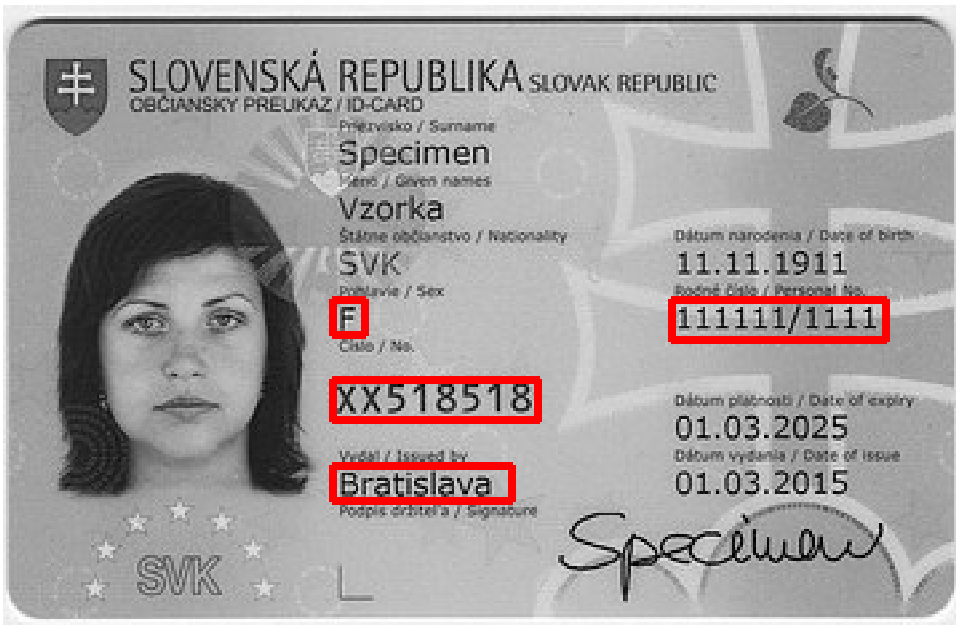

SLOVENSKÁ REPUBLIKA SLOVAK REPUBLIC
OBČIANSKY PREUKAZ / ID-CARD
Priezvisko / Surname
Specimen
Meno / Given names
Vzorka
Štátne občianstvo / Nationality
SVK
Pohlavie / Sex
F
Číslo / No.
XX518518
Vydal / Issued by
Bratislava
Podpis držiteľa / Signature
Dátum narodenia / Date of birth
11.11.1911
Rodné číslo / Personal No.
111111/1111
Dátum platnosti / Date of expiry
01.03.2025
Dátum vydania / Date of issue
01.03.2015
Specimen
SVK

FIGURE 4.5: Sex, record_No, passport_No and authority are in bounding boxes

After extracting dates, name, surname, and nationality, I used regular expressions to extract other fields: passport number, record number, place of birth, sex, and authority. Names of all these fields may vary in different ID cards. Some of them can be absent at all. I have extracted place of birth and sex from both languages because they can vary. I also have extracted the passport number, record number, and authority from English text only because it is mostly numeric. I chose the English version of the text because engines work best with it. In my opinion, the reason for better English fit is the high amount of available training data for this language.

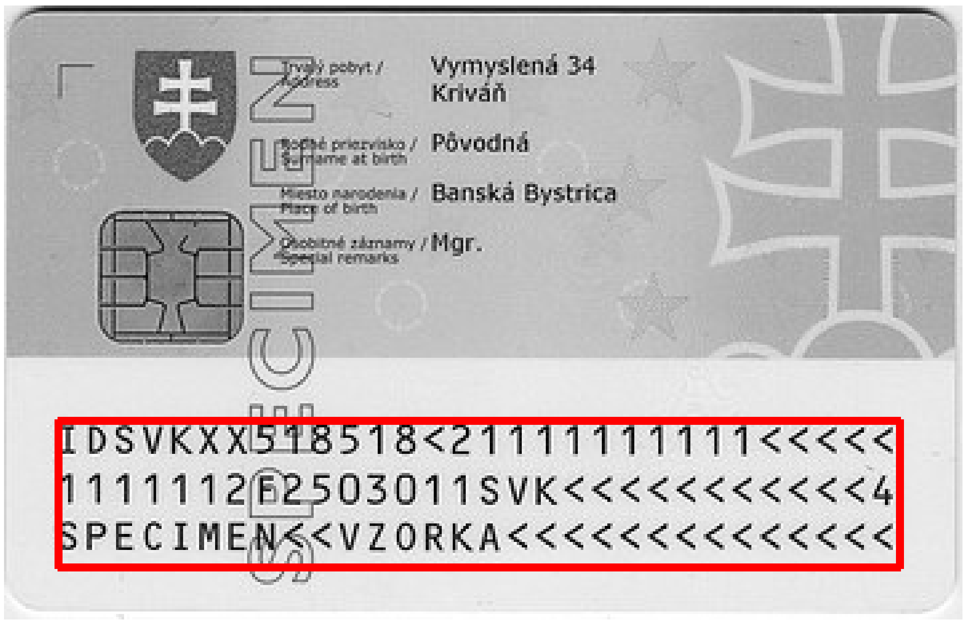
The Ukrainian ID card is specific, and nationality is not extracted during the entity parsing process so that it is extracted during the parsing of additional fields.

4.2.6 Positions

All the previous steps primarily relied on regular expressions and text matches only. Additionally, there is one more option to get better results - use the positions of fields that are relative to others. In order to use this approach, I resorted to Levenshtein distance and "the best match approach" described above. I found the best matches for every field name like 'Surname', 'Name', 'Date of birth' and others. Next, I used it to correct the fields I have already extracted. The idea is that the position of the field in the text is situated definitely after the position of the appropriate field name and definitely before the position of the name of the next field. For instance, the word "TKACHENKO" on the Ukrainian sample is situated lower than "Surname" and upper than "Name". So, if both positions and Levenshtein distance between matches and actual field names are smaller than the threshold, the already detected field should be checked. If the field is in between the positions, it is an appropriate result. In another case, the field should be updated with a new one that has been detected between those positions.

4.3 Backpage

4.3.1 MRZ


Trvalý pobyt / Address Vymyslená 34 Kriváň
Rodné priezvisko / Surname at birth Pôvodná
Miesto narodenia / Place of birth Banská Bystrica
Osobitné záznamy / Special remarks Mgr.

```
IDSVKXX518518<21111111111<<<<<
1111112F2503011SVK<<<<<<<<<<<4
SPECIMEN<<VZORKA<<<<<<<<<<<<<<
```

FIGURE 4.6: MRZ on backpage of the ID card

MRZ or Machine Readable Zone is designed for rapid machine-reading and contains all the most essential information about the ID cards. Since it has been created for machines but not for the human eye, it is usually a black-on-white text which is easily distinguishable from the background. Due to this, recognition of the MRZ often shows better results than recognition of other parts. Almost always, MRZ has three lines.

The first one contains document type, country code, passport number, a record number.

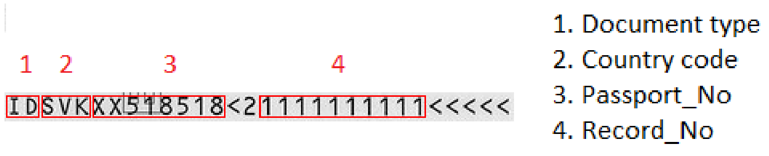


FIGURE 4.7: Description of the first line of MRZ

The second one includes a date of birth, sex, date of expiry, country code.

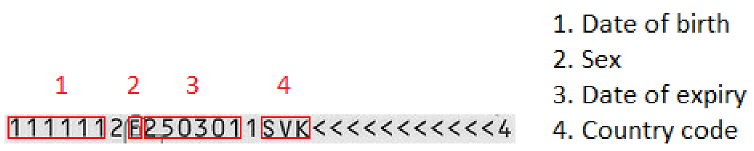


FIGURE 4.8: Description of the second line of MRZ

And the third one provides a name and surname.

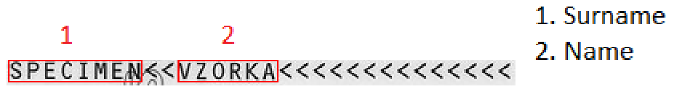
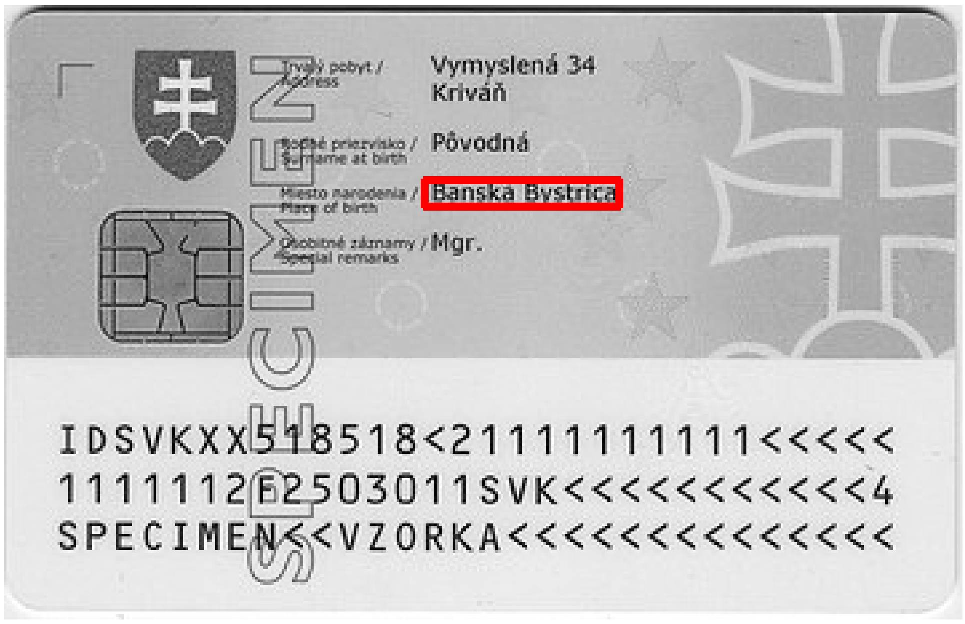


FIGURE 4.9: Description of the third line of MRZ

4.3.2 Additional parsing


Trvalý pobyt / Address Vymyslená 34 Kriváň

Rodné priezvisko / Surname at birth Pôvodná

Miesto narodenia / Place of birth Banska Bystrica

Osobitné záznamy / Special remarks Mgr.

```
IDSVKXX518518<2111111111<<<<<
1111112F2503011SVK<<<<<<<<<<<4
SPECIMEN<<VZORKA<<<<<<<<<<<<<<
```

FIGURE 4.10: On this particular card there is only a Place of birth field on the backpage

On the back page of the ID cards, only authority usually leaves. . In some documents, like Ukrainian, authority is represented as a number, but it also could be multiple words like in the Polish card (Fig 3.1).

Independent of the form, it is parsed using regex, which is customized for every country discreetly. Nevertheless, some documents have a lot of information on the back page. Sometimes there can be almost all data except surname, name, and nationality. That is why I customized an additional parsing function for every document, which has more than just authority on the back page. The most popular thing to be on the back page is the date of issue or date of expiry.

4.3.3 Dates

At this stage, it is time to end up with dates parsing. As you may see, there are several date fields in MRZ, and they are in a different format than dates from the front page (see the Image X). After getting all the dates parsed, they are converted into the same form, "YYYY-mm-dd". It will be helpful for comparing dates from MRZ with dates from the front page with other parts of the project.

4.4 Transliteration

For the following name and surname checking, transliteration is required to get better performance. For transliteration of most languages, I used the polyglot library [1], described in the background information section. However, for the Ukrainian language, it demonstrated too bad results, so I developed Ukrainian to English and English to Ukrainian transliterators by myself. For that purpose, I used an official Ukrainian transliteration table [11]. Besides that, I defined special cases like

```
з: zgh
```

and cases for the beginning of the word like

```
є: ye
```

After testing, I also added some of my own rules that helped me to get the correct results for all the samples.

4.5 Checks

4.5.1 Validating words

Engines sometimes produce a noise. I defined several rules for a word to be considered as actual words but not just a set of characters.

1. Al least one vowel and one consonant
2. At least three characters
3. If the word of length three it cant have two consonant in a row
4. No more than three vowels in a row
5. No more than five (four for the Ukrainian) consonants in a row
6. No more than two non-alphabetic characters

So, validating words is an important operation that allows avoiding many cases with setting some senseless sequence of letters.

4.5.2 Country check

Id cards have from two to three occurrences of country or country code. All of these can be used to determine an actual country. I scraped all the countries and country codes in a .csv file and used Levinstein distance to get the closest match for all country names and country codes. If any of them has a 100% match, I considered it as a valid country and modified all the country fields with this country name and code.

4.5.3 Name check

There are at least two name entries in documents: on the front page and in MRZ. Entry from the front page is parsed two times - in English and local language. Thus, I already have three entries. Sometimes one or more of those entries could be incorrect. For example, it can be replaced with another word, be just noise or have a couple of misspellings. However, in the case of having three distinct entries, there is roughly a triple chance to get a proper name. Next, I will explain in detail how I achieved this.

First, I have validated names with the rules described above (Section4.5.1). If some of the fields are not valid - it is set to be empty. After that, I checked every field for similarity to two lists of words using Levinstein distance. The first list contains country names. It is helpful to remove accidentally trapped country names in the names fields. The second list includes words from the document like "Card" or "Date". It is used in order to be absolutely sure that these words will not be in the final result. If the checked word has a high similarity to one of the words from these lists, it is also set to an empty field. The next step was to get the closest match for each name in the list of names for a particular country. I have scrapped names for every country and save them and their transliteration to .csv files. After getting the best matches, there was a check for every two matches for being the same (all the names are transliterated to English before this step). If they are present, I chose that name as the correct one. If they are absent, I select one with the highest similarity score as a correct one with a condition of at least 80% similarity. In another way, all the names remained the same. I did not t use matches in every case because someone could have a name that is not popular and absent in my database.

4.5.4 Surname check

I did almost the same operation for surnames, except scrapping surnames for each country and looking for the most similar ones. There are too many different surnames for creating a database, and I didn’t find existing databases available on the Internet.

4.5.5 Patronymic check

For countries with a patronymic (Ukraine and Bulgaria), I did for the patronymic field something similar to what I did with names. Through, the point is that there is only one field with a patronymic. So, I converted databases with names to databases with patronymics and looked for the closest match. If the similarity score was high enough - the new match used, if not - the old one.

4.6 Scoring

Since it is a KYC but not only a document parsing system, there should be some scoring or fraud check. I developed a simple scoring system that can filter some common naive mistakes. The first one is the most common - expired document. Actually, it is not necessarily a fraud, and a person with such a document should not always be included in some blacklists. However it is the case that should not be passed. The next check bears on the country. Sometimes scammers can put down a non-existing country or, even more straightforward - they can make a mistake in the country name. Fraudsters are not always professionals, and they could make any mistake you can imagine. The check, which also can be considered as evidence, is a digits consistency. Fields like passport_No, record_No, dates occur at least two times. For instance, the date of birth can occur four times in some cards. In such an abundance of digits, it is pretty easy to make a one-digit mistake which will be caught by this check. One more thing to check is the sex field's consistency. This is also a quite stupid mistake, but it happens. And the last check works only for the Ukrainian ID card - check for visual marks.

4.6.1 Flag check

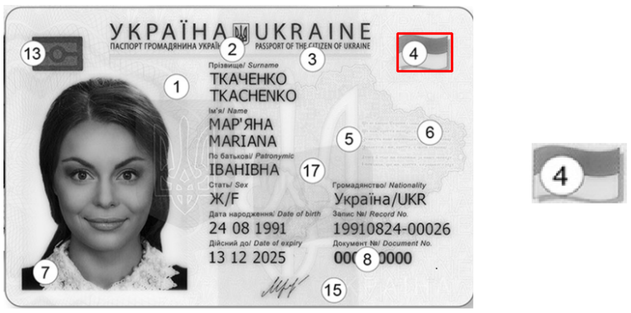


FIGURE 4.11: Image with flag (left) and template of the flag (right)

For this check, I used a small image of a flag as a template. It was cut from a document with good lighting conditions. The best matches for this template were found on the image and filtered by a threshold. After that, duplicated regions were removed, and the remaining are fed to a simple CNN.

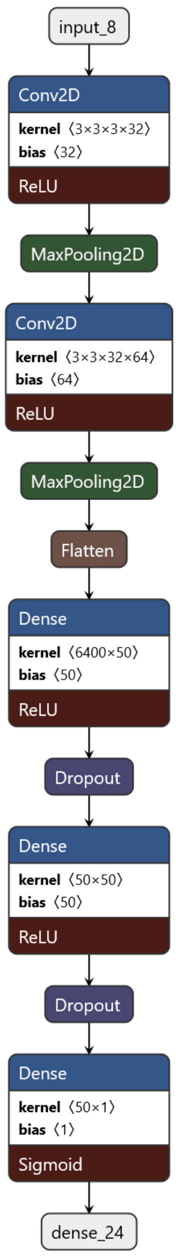


FIGURE 4.12: CNN architecture visualization

Which predicts either it is an flag or not. The dataset I have created and annotated by myself. It consists of 450 photos with flag and 550 photos without flag. Train set included 70% of images. Validation and test sets included 15% of images each. I trained for 10 epochs using Adam optimizer with learning rate 0.001. This model achieved 99% accuracy on the validation set and 98% on the test set.

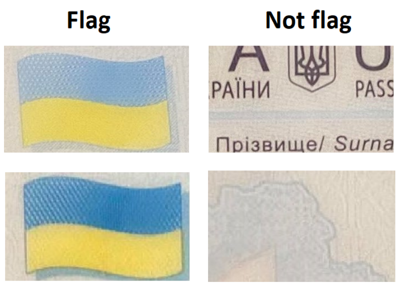


FIGURE 4.13: Images with flag and without flag

The data contains images of flags and pictures with no emblems but with some other ID card parts. This is quite a simple binary classification task.

4.6.2 Sign check

The second visual check is similar to the previous one. Here I tried another approach for detecting a part of the image to be fed into the classification model. As one may see, the sign is yellow and blue. If we convert it into the lab color space and extract the b channel, which shows blue to the yellow ratio, we see that the part with a sign is easily distinguishable.

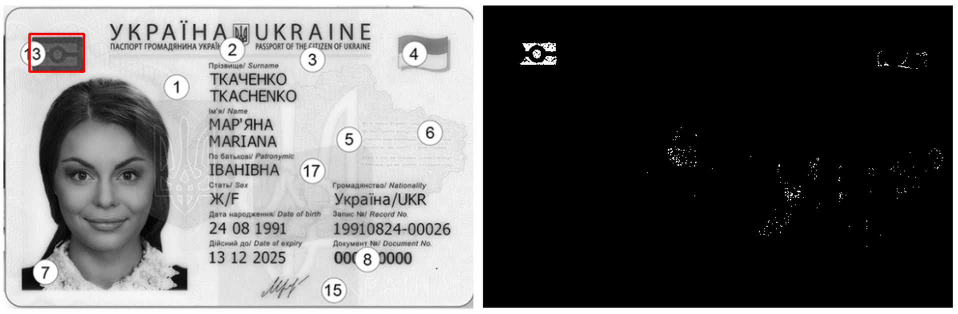


FIGURE 4.14: Image with sign in bounding box (left) and thresholded a chanel of the image (right)

Actually, there are several of these spots in some photos, and I should deal with them. In the beginning, I filter up all the spots that are too small and ones that are near the border because the ID card sign can not be close to the frame. After that, I merge the spots that are small and close to each other. It has been done because sometimes not all the signs can be blue and yellow enough due to bad lighting conditions. The last step was extracting the coordinates from spots, cutting probable signs, and forwarding them through the model. If at least one region is a sign the check was passed. The dataset consists of 450 photos with sign and 550 photos without sign. Model architecture and training configuration is the same as in the previous check. This model achieved 97.5% accuracy on the validation set and 98% on the test set.

4.6.3 Scoring details

TABLE 4.1: Scores for each check

Scores		
Check	Score description	Maximal points
country check	1 point for texts in each language	2
expiration check	2 points for each expiry field	4
record_No consistency	2 points	2
passport_No consistency	2 points	2
dates of birth consistency	1 point for each pair (4 fields, 6 pairs)	6
sex consistency	2 points	2
flag presence	2 points	2
sign presence	2 points	2
Total		22

All these checks have their own impact on the result score (Table 4.1). Additionally, to avoid the low scores due to poor recognition quality, there is a rule for not considering the field as unacceptable if it is absent. For this purpose, I also added a maximum_score variable along with score var. The resulting score is a real number from zero to one, and it is a score/max_score. For instance, if the expiry date is later than the current date, one is added to both max_score and score. If the date is before the current date, one is added to the max_score only. If the field is absent at all, it is not the influence of the scores. Here is also one more rule: if the overall max_score is lower than five - the score is set to zero. I added this rule because if such a small number of fields are adequately recognized, one could not be sure that unrecognized fields are acceptable and the document is valid.

Chapter 5

Results

I have tested both Tesseract and Google OCR on each field and got accuracies for all the fields. Accuracy for a particular field was calculated only on the documents in which this field exists. The field is considered to be correctly recognized only if it exactly matches a genuine one. If there is even one character mismatch, the field is labeled as a wrongly recognized one.

TABLE 5.1: Accuracy results of both engines for each field

Results		
Field name	Tesseract	Google Vision OCR
doc_type (MRZ)	0.84	0.94
country_code (MRZ)	0.87	0.96
passport_No (MRZ)	0.88	0.96
record_No (MRZ)	0.89	0.93
date_of_birth (MRZ)	0.89	0.95
sex (MRZ)	0.82	0.94
date_of_expiry (MRZ)	0.90	0.96
country_code2 (MRZ)	0.88	0.95
surname (MRZ)	0.91	0.90
name (MRZ)	0.93	0.98
country_code	0.79	0.84
country	0.87	0.96
country_eng	0.87	0.96
surname	0.82	0.90
surname_eng	0.84	0.91
name	0.91	0.96
name_eng	0.93	0.97
patronymic	0.77	0.88
date_of_birth	0.88	0.95
date_of_issue	0.86	0.93
date_of_expiry	0.89	0.95
passport_No	0.80	0.90
record_No	0.77	0.90
sex	0.52	0.77
sex_eng	0.54	0.78
place_of_birth	0.31	0.61
place_of_birth_eng	0.33	0.65
authority	0.43	0.70

Chapter 6

Summary

To conclude, the engine-based approach works relatively well. There are some issues with a Greek ID card only due to inappropriate structure for OCR. All the European Union countries and Ukrainian ID cards supported. I have implemented multiple algorithms to increase recognition quality. As one may see, the accuracy for name fields is relatively high, but the accuracy for a place of birth is relatively low. The reason is that the name occurred three times, and it always has the same structure. The place of birth occurred only one time and may have a very different structure. One more reason is that name probably is the most important field, and I developed many algorithms to increase the accuracy of name recognition. For the place of birth, I used only several standard approaches that are not enough to achieve such a good result. In further work, I will fix it.

The main bottleneck of this approach is engine usage. The next step will be to train my own OCR model using attention OCR [15].

Bibliography

[1] Rami Al-Rfou. *polyglot*. 2015. URL: https://github.com/aboSamoor/polyglot. [Accessed: May 9,2021].

[2] Joris Hermans. *What Is Lab Color Space? (And How to Use It in Photoshop)*. URL: https://expertphotography.com/lab-color-photoshop/. [Accessed: May 8,2021].

[3] Cuelogic Insights. *The Levenshtein Algorithm*. 2017. URL: https://www.cuelogic.com/blog/the-levenshtein-algorithm. [Accessed: May 9,2021].

[4] Raghuveer Kanneganti. *A Training-Free and Layout-Agnostic Approach to Extract Key-Value Pairs from Business Documents*. 2020. URL: https://blogs.sap.com/2020/11/06/a-training-free-and-layout-agnostic-approach-to-extract-key-value-pairs-from-business-documents/. [Accessed: May 6,2021].

[5] Edward Ma. *Secret of Google Web-Based OCR Service*. 2019. URL: https://towardsdatascience.com/secret-of-google-web-based-ocr-service-fe30eecedd01.

[6] Novita Hanafiah Michael Ryan. *An Examination of Character Recognition on ID card using Template Matching Approach*. 2015. URL: https://www.researchgate.net/publication/283041809_An_Examination_of_Character_Recognition_on_ID_card_using_Template_Matching_Approach. [Accessed: May 10,2021].

[7] Tran Quoc Long Nguyen Thanh Cong Nguyen Dinh Tuan. *INFORMATION EXTRACTION FROM ID CARD VIA COMPUTER VISION TECHNIQUES*. 2018. URL: https://eprints.uet.vnu.edu.vn/eprints/id/eprint/3281/2/technical_report_v1_Cong_Tuan.pdf. [Accessed: May 10,2021].

[8] A. Walker R. Fisher S. Perkins and E. Wolfart. *Adaptive Thresholding*. URL: https://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm. [Accessed: May 8,2021].

[9] Ray Smith. *An Overview of the Tesseract OCR Engine*. 2007. URL: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/33418.pdf. [Accessed: May 7,2021].

[10] Pallavi S. Bangare Dr. S. T. Patil Sunil L. Bangare Amruta Dubal. *Reviewing Otsu's Method For Image Thresholding*. 2015. URL: https://www.researchgate.net/publication/282282124_Reviewing_Otsu%27s_Method_For_Image_Thresholding. [Accessed: May 8,2021].

[11] *TABLE of transliteration of the Ukrainian alphabet into Latin*. 2010. URL: https://www.kmu.gov.ua/npas/243262567. [Accessed: May 9,2021].

[12] Pitchaya Thipkham. *Image Processing Class 6 — Morphological Filter*. 2018. URL: https://towardsdatascience.com/image-processing-class-egbe443-6-morphological-filter-e952c1ec886e. [Accessed: May 7,2021].

[13] Rini Jannati Gibran Muhammad Bagus Fajar Haris Hamzah Rusnandi Fikri Kevin Kristian Wira Satyawan M Octaviano Pratama. *Citizen Id Card Detection using Image Processing and Optical Character Recognition*. 2019. URL: https://iopscience.iop.org/article/10.1088/1742-6596/1235/1/012049/pdf. [Accessed: May 10,2021].

[14] Jianxin Wu. *Introduction to Convolutional Neural Networks*. 2017. URL: https://cs.nju.edu.cn/wujx/paper/CNN.pdf. [Accessed: May 6,2021].

[15] Dar-Shyang Lee Kevin Murphy Qian Yu Yeqing Li Julian Ibarz Zbigniew Wojna Alex Gorban. *Attention-based Extraction of Structured Information from Street View Imagery*. 2017. URL: https://arxiv.org/abs/1704.03549. [Accessed: May 6,2021].