esprit
Se former autrement

Internship Report
Optical Charater Recognition

ROMDHANI Ichrak

2020/2021

Table des matières

Chapter I:

State of the art

I.1 Problematic :

A correct reading of the check is essential for issuing and recieving payment. Although recognising diffrent and unclear handwrting may be a difficult task that Consumes time and ressources

I.2 Objectif :

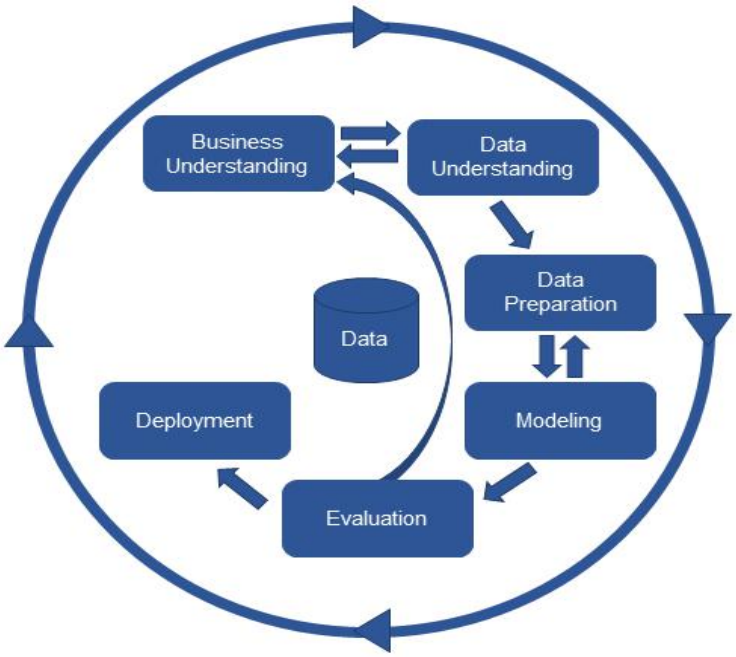
We aim to automate the extraction of information from checks so we gain time and resourses, and also we avoid confusion.

Chapter II:

Methodology

CRISP DM:

The **CR**oss **I**ndustry **S**tandard **P**rocess for **D**ata **M**ining is a process model with six phases that naturally describes the data science life cycle.
This process helps to plan, organize and implement data science project.



II.1 Business understanding:

Focuses on understanding the objectives and requirements of the project

- Extract informations from checks such as the payer's name and the amount of money ...

II.2 Data understanding:

It builds on top of the business understanding phase and drives the focus to identify, collect, and analyze the data sets that can help accomplish the project goals.

2.1 Collect initial data:

- **NIST Special Database19 :** database of A-Z characters

- **The MNIST :** database of handwritten digits.

- **Unreadable:** dataset of check images scrapped using Selinuim

=> Load both the datasets for MNIST 0-9 digits and Kaggle A-Z letters from disk and combine them into a single, unified character dataset.

2.2 Describe data:

-Kaggle A-Z dataset:

Dataset that covers 62 ASCII hexadecimal characters corresponding to the digits *0-9*, capital letters *A-Z*, and lowercase letters *a-z*.

This dataset takes the capital letters *A-Z* from **NIST Special Database 19** and rescales them to be *28 x 28* grayscale pixels to be in the same format as our MNIST data.

- The MNIST datbase:

The MNIST dataset will allow us to recognize the digits *0-9*
It has a training set of **60,000** examples, and a test set of **10,000** examples.
It's a subset of a larger set available from NIST.
The digits have been size-normalized and centered in a fixed-size image.
Each of these digits is contained in a *28 x 28* grayscale image

.

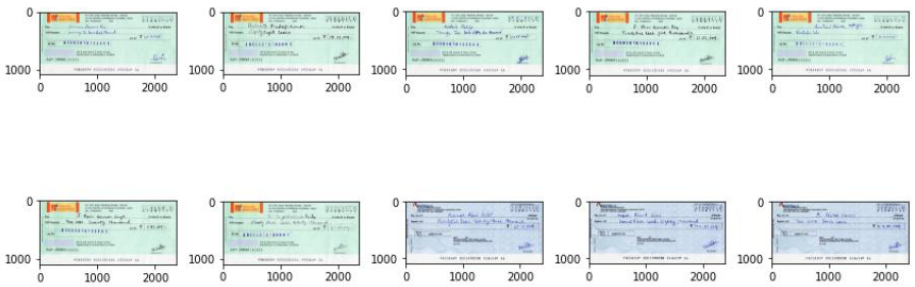
– Data:

Dataset that conatins 120 images of checks with format .jpg and diffrent sizes.

```
img.shape
```

```
(1079, 2365, 3)
```

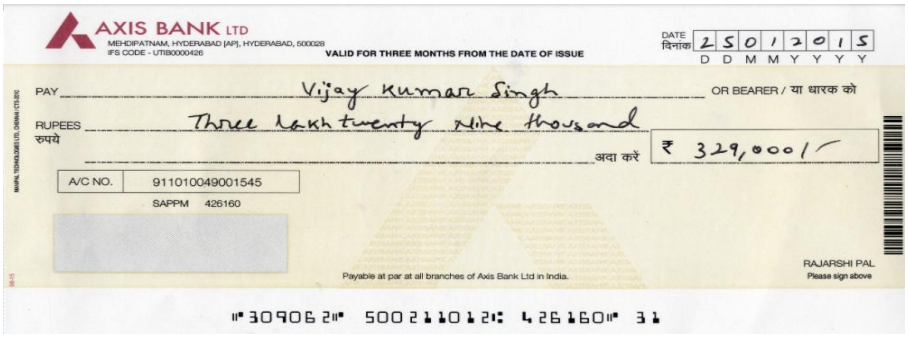
```
[ ] plot_imgs(image_foler_path+'unreadable')
```



II.3 Data Preparation:

Prepares the final data set(s) for modeling. It has five tasks:

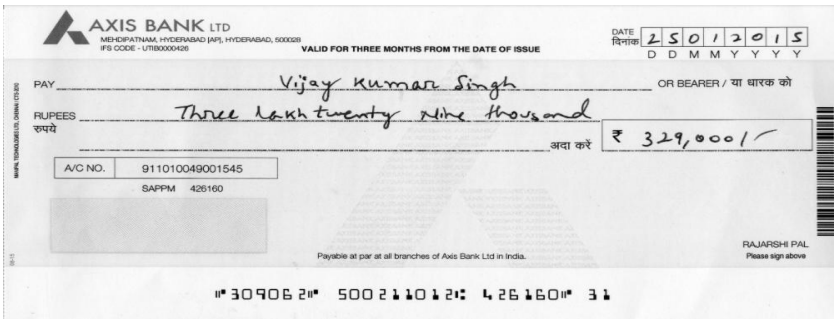
Sample of original data:



The original image has multiple colors.
We need to binirase it in order to eliminate unicessary information and reduce pixels to simplify calculs.

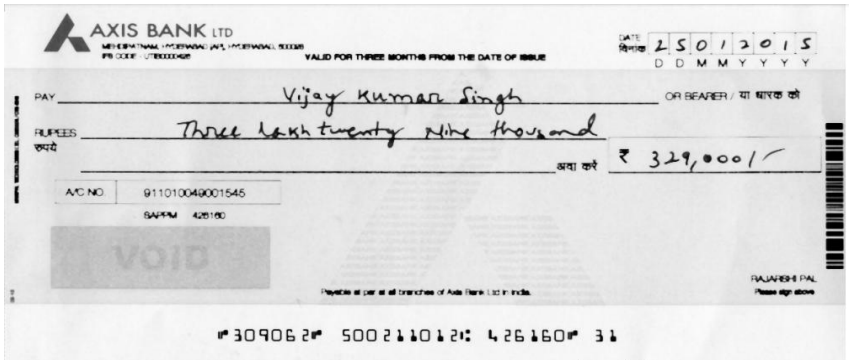
3.2 GrayScale:

Convert RGB image to grayscale image: Less information needs to be provided for each pixel.



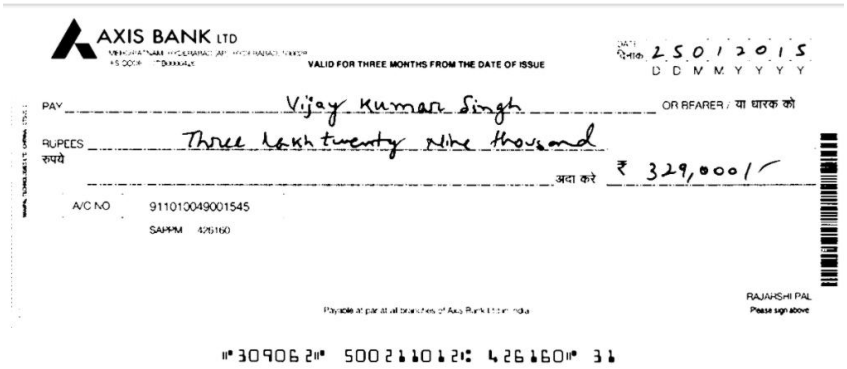
3.4- Denoising:

To smooth the image,I applied **cv2.medianBlur()** which computes the median of all the pixels under the kernel window and the central pixel is replaced with this median value. This is highly effective in removing salt-and-pepper noise and which is our case.



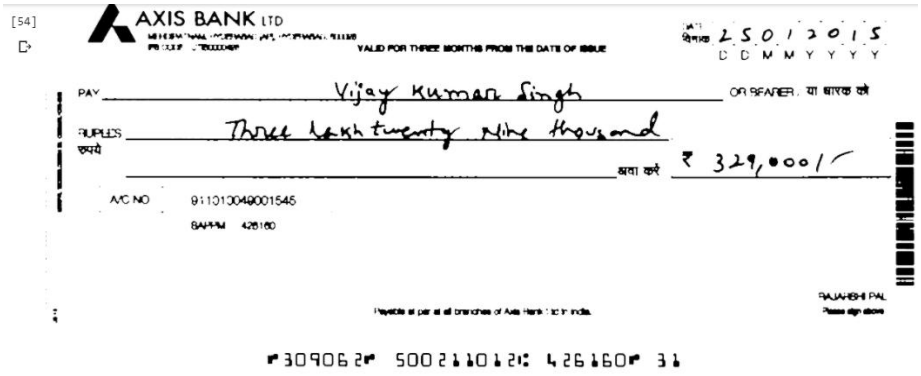
3.5- Binirasation:

cv2.threshold: conevert the grayscale image to black and white in order to show more interest to the writing and neglect the background.



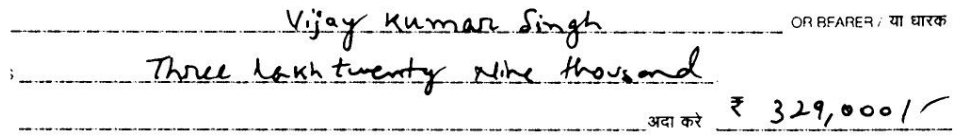
3.5 Deskewing:

Correct the skeweing of the image so that the handwriting model does not confuse between letters or numbers that they looks the same (1 and 7 for example)



3.6 Cropping:

Keep the most important part of the check.



II.4 Modeling:

Build and assess various models based on several different modeling techniques.

4.1 Select modeling techniques:

* **Keras**, **Tensorflow** and an implementation of the deep learning architecture, **ResNet** to classify letters and numbers from the **MNIST** and **A-Z Kaggle datasets**.

* **EasyOCR**: pretrained model
Uses a neural network system based on LSTMs, with major accuracy gains.

4.2 Generate test design:

Partition the data into training and testing splits using 80% of the data for training and the remaining 20% for testing.
(trainX, testX, trainY, testY) = train_test_split(data, labels, test_size=0.20, stratify=labels, random_state=42)

4.3 Build Model:

Train a Keras and TensorFlow model on the combined dataset
model.fit(aug.flow(trainX, trainY...))

II.5 Evaluation:

*Handrwiting.model:

Classification report shows the precision, the recall, f1-score of each character.
We notice that for 0 those values are really weak, and this is because of the confusion between 0 and O.

	precision	recall	f1-score	support
0	0.40	0.66	0.50	1381
1	0.99	0.98	0.98	1575
2	0.94	0.96	0.95	1398
3	0.99	0.99	0.99	1428
4	0.94	0.96	0.95	1365
5	0.76	0.93	0.84	1263
6	0.97	0.98	0.97	1375
7	0.98	0.99	0.99	1459
8	0.98	0.99	0.98	1365
9	0.99	0.98	0.99	1392
A	1.00	0.99	0.99	2774
B	0.99	0.99	0.99	1734
C	0.99	0.99	0.99	4682
D	0.92	0.98	0.95	2027
E	0.99	0.99	0.99	2288
F	0.97	0.99	0.98	232
G	0.96	0.96	0.96	1152
H	0.98	0.98	0.98	1444
I	0.96	0.99	0.97	224
J	0.97	0.98	0.98	1699
K	0.98	0.99	0.99	1121
L	0.98	0.98	0.98	2317
M	0.99	0.99	0.99	2467
N	0.99	0.99	0.99	3802
O	0.95	0.87	0.91	11565
P	1.00	0.99	0.99	3868
Q	0.97	0.98	0.98	1162
R	0.99	0.99	0.99	2313
S	0.99	0.96	0.97	9684
T	0.99	0.99	0.99	4499
U	0.99	0.99	0.99	5802
V	0.97	1.00	0.98	836
W	0.99	0.98	0.99	2157
X	0.98	0.99	0.99	1254
Y	0.98	0.96	0.97	2172
Z	0.95	0.96	0.95	1215
accuracy			0.96	88491
macro avg	0.95	0.97	0.96	88491
weighted avg	0.97	0.96	0.96	88491

-As we can see, our Keras/TensorFlow OCR model is obtaining **~96% accuracy** on the testing set.

The training history can be seen below:

The metric: accuracy
The loss: categorical_crossentropy

*Randomly test characters:

The character ‘o’ or ‘0’ is uncorrectly predicted.

We can see from the sample output that our Keras and TensorFlow OCR model is performing quite well in identifying our character set.

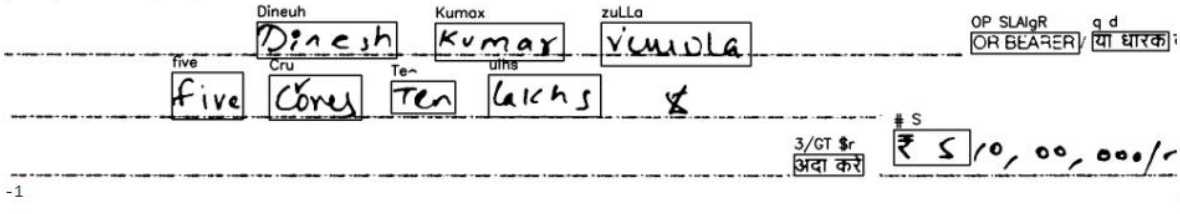
***EasyOCR:**

A pretrained model based on Tessract ocr and the neural network LSTM: Long Short Term Memory.
First we initialize a Reader wich will support the english language.
This reader accept the processed image as an input.
The reader tries to recognize each character in each word by caluliting The probabilities.

```
reader = Reader(['en'], gpu = True)
results = reader.readtext(deskewed)
```

Test:

```
[INFO] 0.1404: Dineuh
[INFO] 0.0131: zuLLa
[INFO] 0.0744: Kumax
[INFO] 0.0475: q d
[INFO] 0.0105: OP SLAIgR
[INFO] 0.2546: five
[INFO] 0.1449: ulhs
[INFO] 0.8606: Cru
[INFO] 0.4820: Te^
[INFO] 0.0391: # S
[INFO] 0.0038: 3/GT $r
```




```
-1
```

II.6 Deployment:

Django:

Is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.
Since Django is written in Python it makes it a great choice of web framework for deploying machine learning models.

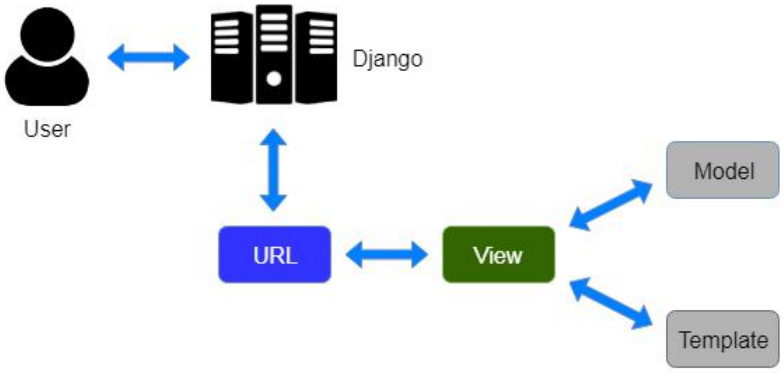
Django MVT architecture:

The MVT (Model View Template) is a software design pattern.
It is a collection of three important components:
Model View and Template.

*The **Model** helps to handle database.
It is a data access layer which handles the data.

*The **Template** is a presentation layer which handles User Interface part completely .

*The **View** is used to execute the business logic and interact with a model to carry data and renders a template.

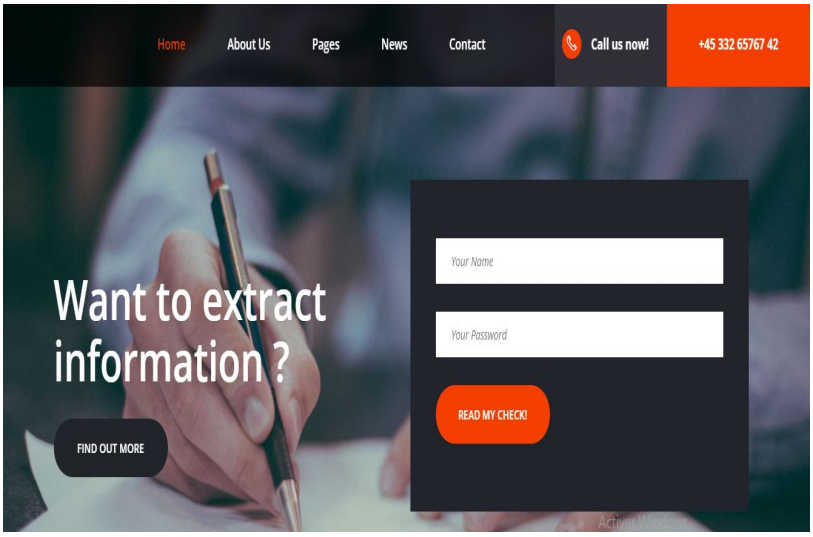


Here, a user **requests** for a resource to the Django, Django works as a controller and check to the available resource in URL.

If URL maps, **a view is called** that interact with model and template, it renders a template.

Django responds back to the user and sends a template as a **response**.

I've implemented the Template part and I'm still working on the model because it gives me bad results.



III. Conculsion:

Many companies wants to automate the task of reading documents.
Tessarct, EasyOCR can resolve that with a really accurate result.
But, reacognzing and reading diffrent handwriting is still challenging until this day.