COMPUTER VISION PROJECT

Tunisian ID card OCR web Application

Hamza Boulahia

September 2021

1 Project presentation

1.1 Problem statement

Identification Cards commonly referred to as IDs are used world wide to verify that the person holding it is indeed who she claims to be. In most cases nowadays, the registration processes that require the information contained in the ID card are conducted manually using a registration forms. And the forms are then digitized by retyping them on computer machines.
For some businesses, this process can become tedious and time-waster when a large number of forms needs to be treated in a daily basis, which affect the productivity of such businesses.

Since, the final form that is required is a digital one, then it would be much more sensible to extract the information from an ID card using a camera and storing it directly in the desired digital format.

1.2 Project objective

The aim of this project is to automate the information extraction process from a Tunisian ID card also known as CIN (Carte d'Identité Nationale). The proposed solution is a web application that uses deep learning models to perform optical character recognition and store the extracted data in a specific digital format.

1.3 Functional requirements list

The functional requirement for this project are the following:

- Supervision feature: The web application should have a feature that allows human interaction with regard to the extracted information, so that when the extracted information doesn't exceed a certain confidence threshold. The supervisor being the only actor interacting with the system, is required to decide whether to accept or manually modify the information.
- ID image privacy: The collected ID image should be deleted once the information extraction process ends. Only the collected text data are allowed to be stored.

1.4 Non-functional requirements list

The non-functional requirement for this project are the following:

- User-friendliness: The user interface should be simple and easy-to-use.
- Speed & performance: The web app should perform the information extraction in no more than 10 seconds with minimal no errors, and the accuracy should be no less than 95%
- Auditability: The system should offer logs inn order to facilitate the error tracing and control procedures.
- Scalability: The application should be able to handle many images simultaneously with different sizes.

1.5 Main process

The main process that is executed via the proposed web application is presented in the following figure:

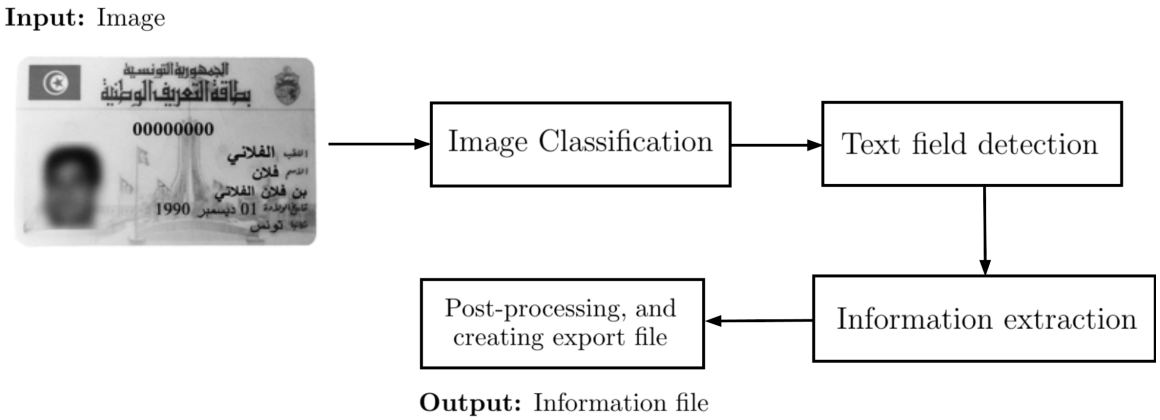

Figure 1.1: Global architecture of the CIN OCR process

It is important to note that the input image could be either showing one side of the ID card or both at the same time. The text field detection step as well as the information extraction step are usually included in a single model that performs OCR. However, given the poor results of the both publicly available OCR libries (Tesseract and EasyOCR) on our CIN dataset. I've used a separate model for text fields detection, which drastically improved the accuracy of the text extraction operation.

1.6 Tools and technologies

This projects consists of two principal components:

- A web application.
- An information extraction model.

For the creation of the information extraction deep learning model, I've used the following tools:

(a) Python logo (b) Tensorflow logo (c) Pytorch logo (d) YOLOv5 logo (e) EasyOCR logo

- **Python:** Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. It supports multiple programming paradigms, including procedural, object-oriented and functional programming. It's is the most used programming language for developing robust machine learning models.
- **Tensorflow:**TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.
- **Pytorch:** PyTorch is an open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab.

- **YOLOv5:** YOLOv5 is a family of object detection architectures and models pretrained on the COCO dataset, and represents Ultralytics open-source research into future vision AI methods, incorporating lessons learned and best practices evolved over thousands of hours of research and development.
- **EasyOCR:** EasyOCR is an open-source library written in pyton, which present a Ready-to-use OCR model with 80+ supported languages and all popular writing scripts including: Latin, Chinese, Arabic, Devanagari, Cyrillic, etc.

For the creation of the web application, I've used the following tool:

(a) Streamlit logo

- **Streamlit:** Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. It present a fast way to create a web application prototype that can also be easily deployed on Streamlit Cloud.

1.7 Project milestones

The milestones of this project are the following:

1. Creating an image classifier for Tunisian IDs.
2. Creating an object detection model to detect text field on a Tunisian ID image.
3. Evaluating OCR results from EasyOCR and Tesseract.
4. Creating a web application that incorporates the created models.

2 Work description

2.1 Data collection

Basically, the first step in the development of the Tunisian ID OCR web app is the collection of data. Since, identification cards are private information and can not be collected without the consent of its owner. I opted for synthetic data generation, to create a small dataset of CIN images, and then use transfer learning to train the required machine learning models. During the training phase of each model, a data augmentation technique was utilized to achieve higher accuracy.
The creation of synthetic data was conducted obeying certain conditions:

- The created images need to be low quality 72 dpi(dots per inch).
- The created images should have different sizes.
- Different types of backgrounds should be used to create synthetic images.
- Incorporate skew, blur, noise, black&white, and other effects to some of the created images.

The created dataset will be used to train both an object detection and an image classifier model. The labeling for the classification task was done manually, whereas for the object detection task, I used a the open source LabelImg tool to create the label files.
LabelImg is a graphical image annotation tool. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO and CreateML formats.

2.2 Models creation

2.2.1 CIN classification model

The created model is used to predict whether the input image is of a CIN or it is not of a CIN. The model is basically a convolutional neural network, with a pre-trained EfficintNetB0 as backbone and a fully connected head.

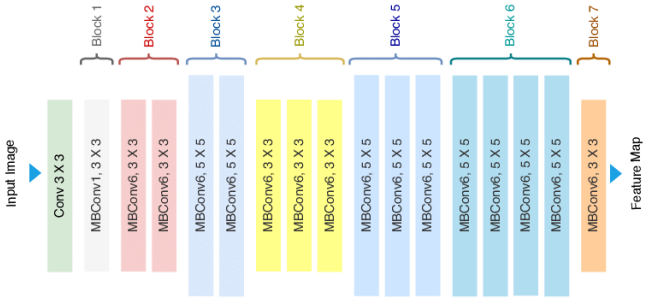

Figure 2.1: The EfficientNet B0 network architecture

The model is trained on a small dataset to perform binay classification, while freezing all the layers of the backbone layers.
This transfer learning technique produce remarkable results especially in case of a small training dataset. The data augmentation technique used in the training process is a standard one, but only zoom, rotation, and skew functions are used.

2.2.2 Text field detection model

The deep learning model used for text field detection i the YOLOv5, which is an open-source model for object detection pre-trained on the COCO dataset. As for the classification task, a similar transfer learning method is used for detecting the text fields on the CIN card images.
A hyper-parameter technique is also incorporated in the training process to optimize the resulted

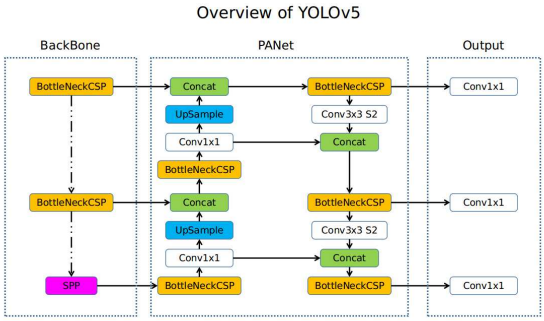

Figure 2.2: The YOLOv5 network architecture

model accuracy. This technique uses a genetic algorithm to optimize the parameters where a base scenario is used at start and by each evolution step we seek to improve it. This technique is less computationally expensive than the classical grid-search and it is more efficient when the model has a large number of hyper-parameters, which is the case with YOLOv5.
The dataset used in this training process was labeled using the open-source labeling tool LabelImg, which provides a GUI to label objects manually and export the label data on a file compatible with the YOLOv5 label format.

2.2.3 Optical Character Recognition

This is the final step in the process of information extraction from the CIN. I tested both open source OCR models available online and which support Arabic language, and then used the one that produced better results in the final implementation of the whole process pipeline. The OCR models that have been tested are Tesseract and EasyOCR. The latter showed better results on extracting Arabic texts especially when numerals are present, whereas Tesseract failed to accurately extract most of the date of birth and CIN number fields. Even though EasyOCR has its own integrated

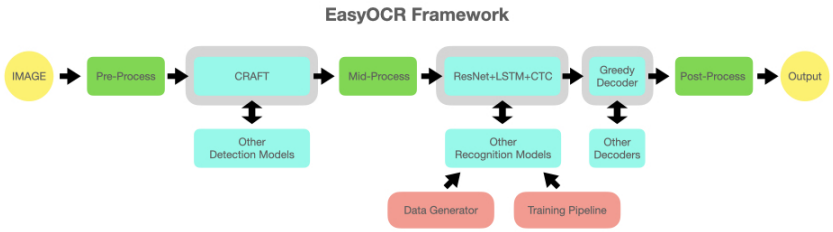

Figure 2.3: The EasyOCR architecture

pre-processing algorithm which help making the input image more suitable for the character segmentation and extraction, it is always advised to pre-process the input image before passing it as input for the EasyOCR model. The pre-rocess step showed huge improvement in the accuracy of the EasyOCR results. The following figure shows the pre-processing techniques used to render the image more suitable for text extraction, such that the model can easily distinguish a character from the background during the segmentation phase.

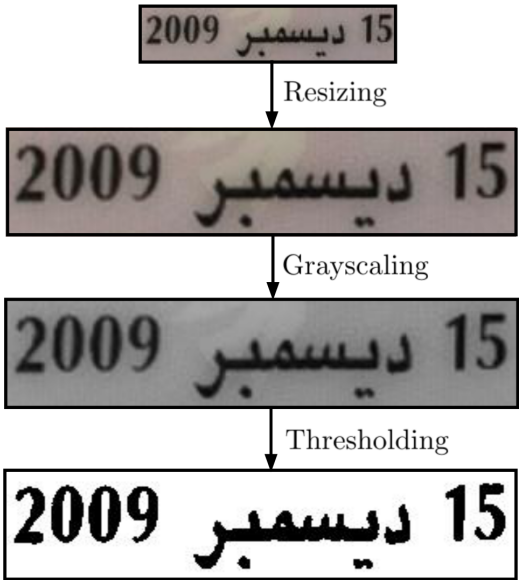

Figure 2.4: Pre-processing Steps for a detected text field

2.3 Web application development

The development of the web application is done completely with Python, using a library called Streamlit. Basically the whole application is built in a single Python file, and it calls many other Python files that are required for the CIN OCR process.
The list of files are the following:

- App.py: The main file of the web application.
- cin_ocr.py: Used to create the page where the CIN OCR process can be conducted.
- info.py: Used to create the home page of the application.
- best.pt: Contains trained weights for the YOLOv5 model.
- hubconf.py: Configuration file used to import YOLOv5 model from Pytorch hub.
- EffNetB0_CIN.h5: Contains trained weights of the VGG16 model.
- arabicocr.py: Used to perform EasyOCR.
- MyUtils.py: Contains helper functions for pre-processing and post-processing.

In the Github repository of this web application, we can find other files than the ones mentioned above, however those are only created for the deployment of the application on Streamlit Cloud.

2.4 Deployment

The deployment on Streamlit Cloud is straight forward and it is done by through the Github repository of the web application. The deployed application is then updated automatically, whenever a new version is available on Github.

3 Conclusion & Perspectives

In the current project, a web application is created to perform OCR on the Tunisian ID card using python. The created application is able to perform this operation with an accuracy of 81%, and the whole process on a single image takes 7 secs, when performed locally.
There are many things that can be done to improve both the accuracy and the speed of the application, such as using more training data and using more semantic post-processing techniques.
Moreover, the web application can be upgraded to be able to perform the whole process on live stream video instead of uploaded images.

References

[1] Ultralytics, YOLOv5 Github repository, `https://github.com/ultralytics/yolov5`

[2] EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, Mingxing Tan and Quoc V. Le, 2020

[3] Jaided AI, EasyOCR Github repository, `https://github.com/JaidedAI/EasyOCR`