

**USSI2Y Données temporelles et spatiales**

**Détection de la nationalité d’un individu**

**à partir de documents officiels (Visa/passeport/ID)**





****

****

**Rédigé par :**

Adrien Golebiewski

Quentin Lachaussée

Vladimir Guignard **Surveillé par :**

Youssef Balti Hoang Nam Ho

Introduction

Aujourd’hui de la numérisation des documents papiers est un enjeu important dans toutes les entreprises. Pouvoir accéder, exploiter et stocker les informations issues de ces documents constitue un enjeu primordial dans la quête de développement et d’augmentation de profits. La Data Science et le Big Data au sens large participe à cette évolution pour une meilleure exploitation des données issues de documents.

Dans le cadre de l’UE, “Données temporelles et spatiales”, nous avons réalisé cette année un projet Data Science qui intervient dans cette problématique.

L’enjeux est de pouvoir proposer une solution industrialisée d’identification de la nationalité d’un individu à partir d’un document officiel. Nous avons restreint le nombre de documents aux documents suivants : visa, une carte d’identité ou bien un passeport.

L’intérêt de ce projet est de pouvoir automatiquement identifier la nationalité des individus sans passer par des processus manuels, longs et répétitifs qui peuvent exister aujourd’hui dans certaines entreprises. Le gain en temps et en réactivité est donc important, et c’est cela que nous voulons mettre en avant dans ce rapport.

Pour cela, nous avons utilisés différents packages du langage Python afin de répondre à notre problématique complexe, à la fois d’un point de vue technique sur la reconnaissance mais aussi lors de phase d’industrialisation à travers le développement d’une interface de visualisation. Les langages HTML et CSS ont également été utilisé afin de détaillés de manière visuelle notre interface de la solution.

La réalisation du projet nous a également mené à élargir notre problématique en tentant d’identifier les informations personnelles (nom et prénom) de l’individu à travers un de ces documents officiels et de faire une validation de ces données auprès des données officielles de l’INSEE.

Ce rapport présente donc dans un premier temps l’aspect organisationnel du projet avec l’organisation et la répartition des différentes tâches. Ensuite, est présentée la construction de notre Dataset de documents qui constitue notre base de développement. Puis, la reconnaissance de la nationalité par le texte, un croisement avec les données de l’INSEE et la construction d’une interface de test seront expliqués de manière technique et argumentée. Enfin, un bilan de nos démarches et des perspectives d’amélioration du projet conclu ce rapport.

Table des matières

[I - Organisation et gestion des tâches 4](#_Toc67418421)

[II - Construction du Dataset 6](#_Toc67418422)

[III – Processus de reconnaissance de la nationalité 9](#_Toc67418423)

[A) Reconnaissance par le texte brut 9](#_Toc67418424)

[a) Reconnaissance sans traitement de l’image 9](#_Toc67418425)

[b) Reconnaissance avec traitement de l’image 15](#_Toc67418426)

[B) Reconnaissance de la nationalité par la forme 20](#_Toc67418427)

[IV – Généralisation du processus d’identification 25](#_Toc67418428)

[V – Industrialisation de notre solution à l’aide de l’outil Flask 25](#_Toc67418429)

[VI - Conclusion et perspectives 25](#_Toc67418430)

# I - Organisation et gestion des tâches

Dans tout projet, l’aspect managérial est important à assimiler et à reproduire entre les différents membres du groupe afin d’avancer de manière collective.

Au cours des 4 mois de travaux, nous avons eu plusieurs durant lesquelles nous avons avancé de manière rigoureuse et progressive. Voici ci-dessous un bref aperçu des différentes tâches réalisées à chaque séance :

**27 novembre 2020 :**

Vladimir-Quentin : Création du Dataset (récupération d’images de cartes d’identité/passeports/Visas)

Adrien - Youssef : Familiarisation et découverte des packages Python Open-cv et Pytesseract.

**4 décembre 2020 :**

Vladimir-Quentin : Finalisation de la création du Data Set (Ajout supplémentaire de photos de documents, cadrées et non cadrées afin de varier les données de notre Dataset).

Adrien – Youssef : Familiarisation des packages Python Open CV et Pytesseract. Début de tests d’algorithmes d’internet sur des images random.

**10 décembre 2020 :**

Quentin : Indexation des données (Carte d’identité, Visas, passeports). Récupération des textes/formes sur les documents par nationalité. Conception de tableaux excels récapitulatifs. Reconnaissance de la nationalité par texte avec l’utilisation des librairies Python Pytesseract et Passport Eye

Youssef : Reconnaissance de la nationalité par texte avec l’utilisation des librairies Python Pytesseract et Passport Eye. En duo avec Quentin.

Adrien-Vladimir : Reconnaissance de la nationalité par la forme des objets présents dans les documents. Familiarisation de la méthode de seuillage d’images/méthode d’Otsu pour récupérer les formes contenues dans l’image. Test sur les images des cartes d’identité.

Adrien : Familiarisation, découverte du classifieur d’objets Haar Cascade par apprentissage.

**12 janvier 2021 :**

Présentation orale intermédiaire de l’avancée du projet.

Début de rédaction du rapport du projet.

**10/17 février 2021 :**

Généralisation de la reconnaissance des informations d’un document officiel, récupération des informations d’un individu sous forme de Data Frame.

Réflexion autour de notre interface de visualisation/test de notre algorithme de reconnaissance 🡪 Test et début de démarches autour de lu Package FLASK.

**23 mars 2021 :**

Finalisation de l’interface de visualisation de test de reconnaissance sur FLASK. Détails graphiques et fonctionnels ajoutés.

Ces différentes séances de travail ont été complétées par des séances individuelles dont l’objectif était de poursuivre les démarches entamées durant nos séances collectives. En début de projet, les démarches ont été réalisées en duo puis notre organisation a évolué de manière à cibler le « reste à faire » tout en continuant à rédiger et préparer nos présentations d’avancement et notre rapport.

La répartition des tâches peut être résumée de la manière suivante :

|  |  |  |  |
| --- | --- | --- | --- |
| **Quentin** | **Youssef** | **Vladimir** | **Adrien** |
| * Construction Data Set * Indexation des données * Développement Python Pytesseract-PassportEye * Développement interface Flask | * Développement Python Pytesseract-OpenCV * Développement Interface Flask | * Construction Data set * Indexation des données * Développement Python Open CV | * Développement Python Open CV (seuillage) * Indexation des données * Familiarisation Classifieur Haar Cascade/Apprentissage (reconnaissance par la forme) |

# II - Construction du Dataset

Pour bien entamer un projet de Data Science, il faut récupérer la donnée en elle-même et en construire un data set.  Cette étape est souvent la plus longue mais la plus importante dans tout projet orienté Data.

C’est donc avec l’aide de plusieurs moteurs de recherche comme Google images, mais aussi Eclosia que nous avons obtenu des images de Passeport, de Visa et de cartes d’identités, pour pouvoir obtenir un DataSet complet.



C’est ainsi que nous avons récolté 32 cartes d’identité de 8 nationalités distinctes avec 4 personnes de chaque nationalité.

Mais également 32 passeports avec ces mêmes 8 nationalités et 4 autres personnes par nationalités, qui sont belges, Anglais, Français, Allemands, Italiens, Polonais, Portugais et Espagnols. Pour finir on a également récolté 35 visas , avec 7 chinois , 6 anglais, 8 français , 4 allemands, 2 russes et 8 américains en destination de plein de pays différents.

Avant d’aller plus loin dans le sujet, nous tenons à informer que la plupart de ces documents ne sont pas réels. C’est-à-dire que les informations précisées dans ces documents ne sont pas attachées à une personne physique identifiée ou identifiable.  Mais ce sont des vraies images de passeports, cartes d’identité et visas qui constituent notre Date Set. Elles ont cependant été générées en utilisant des faux noms, adresses etc … afin que nous, développeurs puissions les manipuler librement tout en respectant les normes du RGPD.

Cette centaine de documents se caractérise par une grande diversité. En effet, les documents sont tous différents les uns des autres, notamment dû au fait que ces images ont été pris d’internet. Certaines sont donc très petites et d'autres très grandes comme on peut le remarquer avec ces deux images.



à droite une image taille réel, à gauche une image

 2 fois plus petite ,

D’autres images constituent des photos du document et ont donc une vue complète sur celui-ci. Ainsi, des éléments extérieurs peuvent apparaître en dehors du document comme par exemple des doigts pour pouvoir tenir la pièce en question, ou alors plusieurs pièces d’identité en une seule image, comme le montre ces image.



La construction d’un Dataset est très importante pour un projet data science, mais il est également important que celui-ci soit très hétérogène. Mais le traitement d’un projet aux données hétérogènes avec une grande variété de nationalité demande une création d’une documentation impeccable.

C’est ainsi que pour faciliter la création de notre dataset et favoriser sa variété de données, un fichier Excel a été créé dans le but de regrouper et repérer les éléments indicatifs pouvant permettre de repérer la nationalité dans tous les documents que ce soit par les visas, cartes d'identité et les passeports.



*Document excel regroupant les différentes données textuelles permettant d’identifier la nationalité*

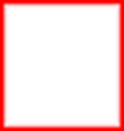


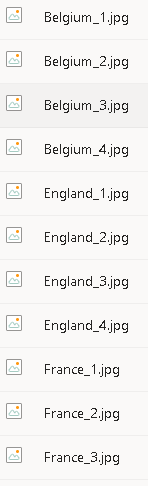
*Document excel regroupant les différentes données visuelles permettant d’identifier la nationalité*

****Comme expliquer juste avant, Il y existe deux types d’options pour le traitement de détection.

****

Le premier est le traitement par le texte, Par exemple une carte d’identité Allemande sera toujours décrite avec la phrase du haut indiquant « carte d’identité allemande » ou bien par la catégorie « Nationalité » noté sur la carte avec « Deutsch ».

****Le deuxième correspond aux différentes formes et contours des objets présents sur la carte. Par exemple, on peut retrouver sur une carte d’identité  italienne un drapeau italien  tout le long en bas de celle-ci.

 La dernière phase de construction de notre data set consiste à renommer nos données selon la nationalité correspondante et à définir un système d’incrémentation pour pouvoir compter le nombre de documents par nationalité (par exemple :  Belgium\_1 Belgium\_2,etc.).

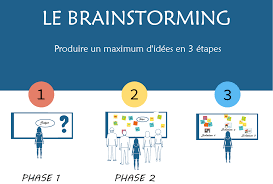
Cette étape a été essentielle pour nos traitements futurs.

Une fois le DataSet construit, nous sommes passés à la partie développement Python concernant la reconnaissance de la nationalité par le texte et la forme des objets le composant.

# III – Processus de reconnaissance de la nationalité

## A) Reconnaissance par le texte brut

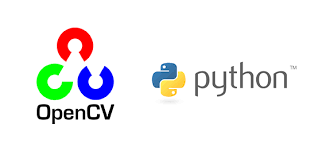
### a) Reconnaissance sans traitement de l’image

Avant de commencer chaque projet, chaque mission, ou même chaque cas d’usage, il est nécessaire de lance une phase de réflexion qui doit-être, au moins, aussi longue que la phase de développement.

Hormis la partie « préparation du DataSet » avec les photographies des documents d’identité à récupérer, il fallait identifier tous les axes d’analyse qui aurait pu nous aider à mieux appréhender le sujet.

Chaque méthode, chaque technique ou mêmes chaque programme algorithmique de tout langage était bon à prendre pour aboutir à un résultat des plus satisfaisant.

Pour la partie développement, il était intéressant de se pencher sur la richesse du langage Python et de sa communauté, afin d’identifier les librairies qui auraient pu nous être utiles.

Nous avions déjà une première librairie “Opencv”, spécialisée dans le traitement d'images, mais nous y reviendrons plus tard.

Toutefois, nous étions persuadés qu’il existait d’autre librairies qui concernait le traitement d’image, voire même plus précisément le traitement des documents officiels.

En partant donc d’”Opencv”, de recherche en recherche par similarité, que ce soit sur Google ou Youtube, pléthore d'autres librairies ont émergé.



Le site regroupant toutes les librairies est le suivant : <https://pypi.org/>

Ce dernier contient toutes les documentations, ainsi que les lignes de code à taper pour installer les dites librairies.

En ce qui concerne le traitement d’image par la lecture de texte, on a recensé près d’une dizaine librairies telles que :

* + - PasseportEye
    - Pyetesseract
    - Card
    - Id\_card\_extractor
    - Etc…

Malheureusement, par manque de documentation (ou de tutoriel) sur la plupart d’entre elles, nous avons décidé de ne conserver que les 2 premières librairies, qui se sont révélées par la suite très efficaces, et donc, très utiles dans notre développement.

**PasseportEye :**

La librairie « PasseportEye » permet de détecter ce qu’on appelle le « MRZ ». Cet acronyme vient de l’anglais « Machine Readable Zone ». Il s'agit de 2 lignes présentes au bas de chaque passeport et chaque visa qui permet de regrouper au même endroit, toutes les informations utiles à la validation du document et à la reconnaissance de son titulaire.

On peut y retrouver par exemple le nom, le prénom, le sexe…



MRZ

La librairie « PasseportEye » permet en premier lieu de détecter cette zone de lecture et ensuite d’extraire le texte à l’aide d’un OCR (Optical Character Recognition).

Une fois le texte prélevé, « PasseportEye » va identifier les 88 caractères des 2 lignes et supprimer les chevrons qui font office d’espace supplémentaire (présent, par exemple, pour les personnes qui ont plusieurs nom/prénom, ou lorsque ces derniers sont de grandes tailles).



Le texte va ensuite être segmenté pour identifier chaque information selon sa position.

Nom du titulaire

Prénom du titulaire

Sexe du titulaire

Date d’expiration

Nationalité du document

Type du document

P BEL LOGAN REASE

EH520227 6 BEL 820627 6 M 240912 7 07

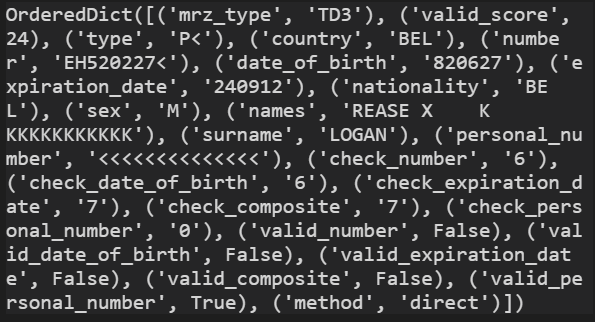
Date de naissance

Pays d’origine

Numéro du passeport

Ces données sont incorporées dans un dictionnaire de données, et il n’y a plus qu’à aller chercher l’information voulue, en l’occurrence, le pays d’origine.

La théorie étant expliquée, voyons la pratique dans le programme python.



Tout d’abord, on lit le « PasseportEye ».

Tout d’abord, on lit le fichier à l’aide de la fonction « read\_mrz(*file*) » dans la librairie « PasseportEye ».

Ensuite on le transforme en dictionnaire de données grâce à la mention « to\_dict() » et en affichant le résultat on retrouve bien les informations mentionnées précédemment.

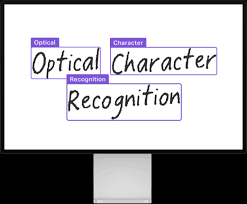
On remarque que c’est la mention « country » qui permet d’aller chercher le pays d’origine du titulaire (à ne pas confondre avec « nationality » qui elle donne le pays du document).

En revanche, la sortie de « mrz\_data["country"] » ne donne pas le nom du pays, mais un code au format ISO 3 (par exemple “FRA” pour France, “PRT” pour Portugal etc.).

Pour pallier à ce problème, nous avons construit une base de données répertoriant chaque pays avec son code ISO 3.

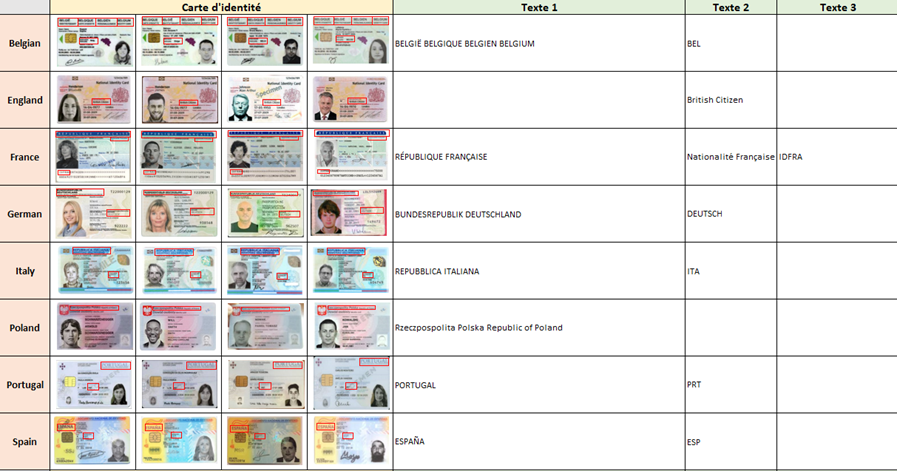
Grâce à cette base de données générée à l’aide de différentes sources comme l’INSEE ou encore Wikipédia, il sera plus simple d’afficher les résultats obtenus.

**Pytesseract :**

En ce qui concerne la librairie « Pytesseract », elle est utilisée dans « PasseportEye » car il s’agit simplement d’une lecture basique de tous les caractères présents dans une image (ligne par ligne).

Elle est donc très utile pour les cartes d’identité (qui ne contiennent pas toutes le fameux « MRZ » évoqué précédemment).

Ainsi, c’est à l’aide de chaque information relevée sur chaque type de carte d'identité, qu’on a mis en place une indexation d’informations. C’est-à-dire qu’on a répertorié chaque mot ou groupe de mots qui pourrait nous être utile pour l’identification de la nationalité.



On a identifié les groupes de mot en 3 catégories :

* **Texte 1**: se situant en haut, qui mentionne des phrases évoquant une particularité concernant le pays.

Par exemple : « BELGIE BELGIQUE BELGIEN BELGIUM » pour la Belgique

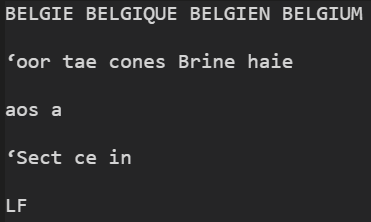
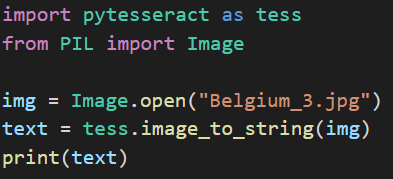
* **Texte 2** : se situant dans le corps, qui mentionne clairement le pays par son nom, ou son code ISO 3.

Par exemple : « DEUTSCH » pour l’Allemagne ou « PRT » pour le Portugal

* **Texte 3** : se situant au bas, qui mentionne la présence du « MRZ »

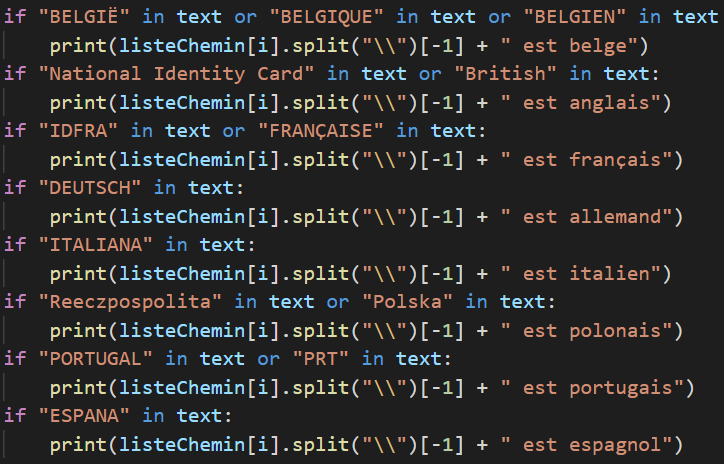
Par exemple : « IDFRA » pour la carte d’identité Française (ID puis FRA)

On a plus qu’à vérifier si les mots clés sont présents dans le texte total pour obtenir un résultat favorable ou non.

Nous avons avec Python, les lignes de code suivantes :

Tout d’abord, on lit le fichier à l’aide de la fonction « Image.open(*file*) » dans la librairie « PIL ».

Ensuite on le transforme en chaine de caractères grâce à la mention « image\_to\_string() » dans la library « Pytesseract », et en affichant le résultat on retrouve bien les informations présentes sur la carte d’identité.

Une fois le texte entre nos mains, il ne reste plus qu’à identifier les mots clés pour déterminer le pays d’origine du titulaire.

Cette méthode a ses limites car la conclusion peut parfois s’avérer un peu rapide et raccourcie.

Prenons l’exemple du Portugal : imaginons que notre carte d’identité soit Polonaise, que mon nom de famille soit « PRTOLE » mais que l’image ne soit pas très lisible. La présence du code ISO 3 du Portugal donnera la nationalité portugaise, ce qui est erroné !

C’est pour cela qu’il est très important de ne pas se contenter de lire l’image, mais bien de la rendre la plus lisible possible, avant d’effectuer un éventuel traitement.

Toutefois, grâce à cette première approche, sans modification de l’image, nous arrivons aux résultats suivants :

* 46,75% de réussite pour les cartes d’identité ;
* 43,72% de réussite pour les passeports ;
* 32% de réussite pour les Visas.

Ce qui constitue, sans traitement de l’image, des résultats plutôt satisfaisants.

### b) Reconnaissance avec traitement de l’image

Les programmes que nous avons développés précédemment représentent principalement deux méthodes différentes pour l’identification de la nationalité.

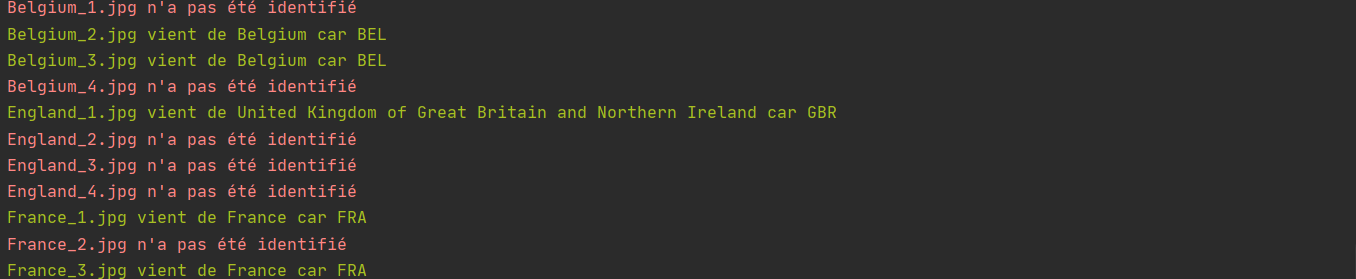
La première méthode se repose seulement sur la librairie Pytesseract. On extrait tout le texte présent dans ce document et qui est détectable par l’OCR Tesseract.



Une fois le texte récupéré, on essaie de chercher des mots clés dans ce texte qui peuvent indiquer la nationalité de la personne comme BELGIUM ou BELG pour une personne de nationalité Belge, etc...

La deuxième méthode utilise le package PasseportEye. Avec cette librairie le programme lit seulement la partie MRZ presente dans le document et extrait directement la nationalité de la personne à l’aide de la mention Country.

Suite à l’exécution de nos programmes en utilisant les deux différentes méthodes, on a constaté que notre programme était incapable d’identifier la nationalité pour certains documents (Passeport, Visa ou carte d’identité.) sans traitement de l’image.



Nous avons donc essayé de chercher l’origine du problème qui pourrait empêcher notre programme de d’identifier la nationalité, en regardant et analysant les images qui n’ont pas été détectées de plus près, unes par unes.

Nous avons conclu que cet échec de détection du texte par l’algorithme peut être due à une ou plusieurs causes suivantes :

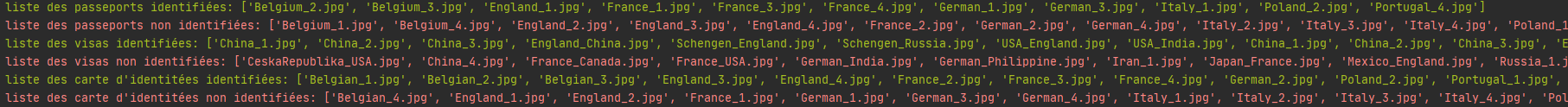
- La partie MRZ du document a été découpée ou a été partiellement couverte. - un mauvais scan qui résulte en un texte trop flou - les dimensions de l’image sont très petites

C'est dans ce contexte que le rôle du package OpenCV intervient. Cette librairie riche contient plusieurs méthodes qui nous permettent de faire un traitement sur nos images afin d’avoir un meilleur résultat et une image plus claire et plus nette pour que notre programme puisse lire correctement le texte présent dans l’image.



Notre première approche était d’effectuer cette transformation sur toutes les images afin de les uniformiser. Nous avons donc appliqué toutes ces modifications sur l’ensemble des documents et on a réexécuté notre programme. Nous avons constaté ensuite, que notre programme a réussi à détecter des nouvelles nationalités pour les passeports et cartes identités, mais il a aussi la perdu la capacité d’identifier des nationalités qui étaient identifiables avant l’application de ces transformations. Pour cela, nous avons employé une nouvelle démarche.

On a créé deux listes pour chaque catégorie de document. Chaque catégorie va avoir une liste « identified » et une autre liste « unidentified ». La liste identified contiendra les noms des documents pour lesquels on a pu identifier la nationalité, et la liste « unidentified », contrairement à « identified », contiendra les documents pour lesquelles la nationalité est encore inconnue. Prenons l’exemple de la liste des nationalités identifiées pour les visas : liste des visas identifiés: ['China\_1.jpg', 'China\_2.jpg', 'China\_3.jpg', 'England\_China.jpg', 'Schengen\_England.jpg', 'Schengen\_Russia.jpg', 'USA\_England.jpg', 'USA\_India.jpg']



Nous avons redimensionné ces images, en récupérant seulement les chemins pour les images non identifiées grâce aux listes que nous avons créées.

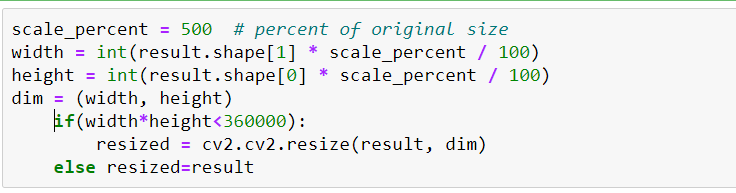
Certaines images avaient des dimensions très petites (i.e. : inférieure à 200\*200 Pixels), ce qui rendait la lecture et la détection du texte très difficile pour le programme de reconnaissance optique.

Nous avons constaté parfois l’impossibilité de détection n’est pas due à une dimension d’image qui est petite. Parfois la taille de l’image est très large, i.e. 2517\*1830 pixels, mais toute la partie MRZ est censuré.

Une augmentation de la taille de l’image n’est donc pas nécessaire dans ce cas et parfois elle peut même entrainer une erreur qui arrête l’exécution de notre programme.

« *DOS attacks caused by “decompression bombs*»

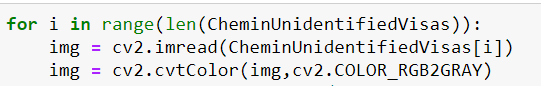
Nous avons donc appliqué le redimensionnement seulement pour les images qui ne dépassent pas une taille de 360 000 pixels (hauteur \* largeur) afin d’éviter tout problème d’exécution.



Dans un second temps, on a converti les images en Grayscale (Niveau de gris).

Une image Grayscale est constituée de seulement des nuances de gris où chaque pixel dans cette image représente seulement la quantité de lumière présente. La valeur 0 d’un pixel représente la couleur noire et la valeur 255 représente la couleur blanche. Toute valeur qui se situe entre ces deux dernières représente une nuance de gris.

Cette étape est nécessaire afin de pouvoir passer à l’étape suivante : le Thresholding.



L’image ci-dessous représente la première transformation appliquée pour une image d’un passeport.





La méthode de thresholding a ensuité été utilisée. Cette technique est une technique de binarisation d’image qui consiste à transformer une image en niveau de gris à une image en noir et blanc. Cela veut dire plus précisément que les pixels dans nos images ne peuvent être que des 0 ou des 1 (noir et blanc)

Il existe deux manières pour utiliser le thresholding: global thresholding et adaptive/local thresholding. Nous avons privilégié l’utilisation du deuxième parce qu’on a eu des meilleurs résultats sur nos images avec le adaptive thresholding qu’avec le global thresholding et cela est expliqué par leurs différents fonctionnement.

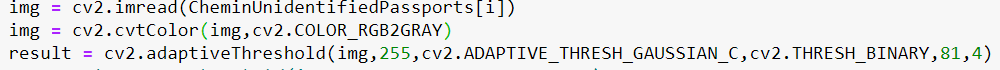
Le **Global Thresholding** fixe un seuil pour tous les pixels dans l’image. C’est-à-dire, que toute valeur de pixel dépassant cette valeur sera transformée en un pixel blanc et tout pixel ayant une valeur inferieur à ce seuil va être transformé en un pixel noir.

Le **adaptive thresholding** ou **local thresholding** permet de donner un seuil différent pour chaque pixel en se basant sur les pixels qui l’entourent.

Il y a deux algorithmes pour la méthode adaptive thresholding dans la librairie de openCV, chacune a une manière différente pour calculer ce seuil :

* Le « ADAPTIVE\_THRESH\_MEAN\_C » fixe la valeur du seuil pour un pixel donnée en calculant la moyenne des valeurs des pixels qui l’entourent. D’où le nom « mean » qui veut dire moyenne en anglais.
* Le « ADAPTIVE\_THRESH\_GAUSSIAN\_C » fixe la valeur du seuil pour un pixel donnée en calculant la somme pondérées des valeurs de pixels voisins en utilisant une fonction gaussienne.

Pour ce projet, nous avons utilisé seulement l’algorithme : adaptive\_thresh\_gaussian\_c.



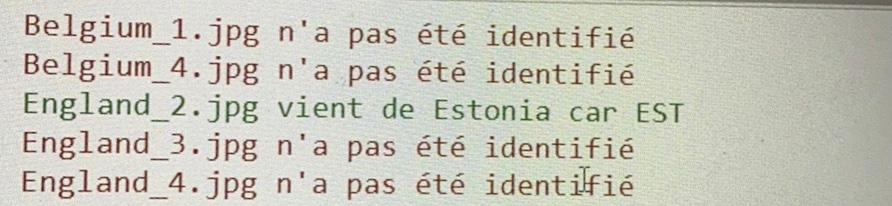
 L’image ci-dessous représente la transformation appliquée sur une image par le thresholding.

Grace à cette première tentative de modification, nous avons réussi à récupérer la nationalité pour une bonne quantité de documents. Mais il reste un nombre important de documents pour lesquels on ne connait pas la nationalité.

La table ci-dessous résume les taux de réussite pour chaque catégorie de documents avant et après la transformation des images :

|  |  |  |
| --- | --- | --- |
|  | **Sans Traitement** | **Avec Traitement** |
| Passeports | 46,75% | 56% |
| Visas | 32% | 32% |
| Cartes d’identité | 43,72% | 47% |

On remarque une très bonne amélioration pour le taux de réussite des passeports suite aux traitements des images. Le programme a pu détecter 3 nouvelles nationalités en plus. Mais parmi ces 3 nouveaux cas détectées, on a remarqué qu’une nationalité a été mal détectée ‘’False positive error’’. La nationalité qui a été affiché n’est pas conforme à la nationalité réelle de la personne. Cela veut dire que la précision du modèle n’est pas parfaite et peut être largement améliorée. On ne peut donc pas faire complètement confiance à notre programme.



Il est donc nécessaire de s’appuyer sur d’autres éléments que contiennent ces documents officiels.

En effet, tous les documents officiels contiennent des formes, des couleurs et des tailles différentes. Il serait donc judicieux de définir des éléments pertinents (autres que le texte) qui pourraient permettre d’identifier la nationalité pour chaque document.

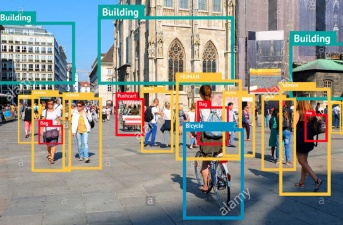
# 

## B) Reconnaissance de la nationalité par la forme

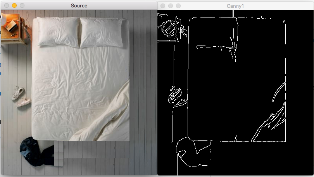
## 

La détection de la nationalité par le texte étant compliquée et hasardeuse pour le cas de la carte d’identité, il est donc nécessaire de trouver une solution adéquate sans passer par le texte.

L’option par le texte éliminée, nous nous sommes donc rattachés à la détection par la forme et contours d’objets.



Changement de méthode



Par la forme

Par le texte

Notre objectif est donc devenu plus précis car nous souhaitons récupérer une forme d’un objet de la carte d’identité et ainsi en déduire à partir de cette forme la nationalité de l’individu représentée sur sa carte.

Tout d’abord, il a fallu sélectionner les formes/objets pour chaque nationalité présente sur les cartes.

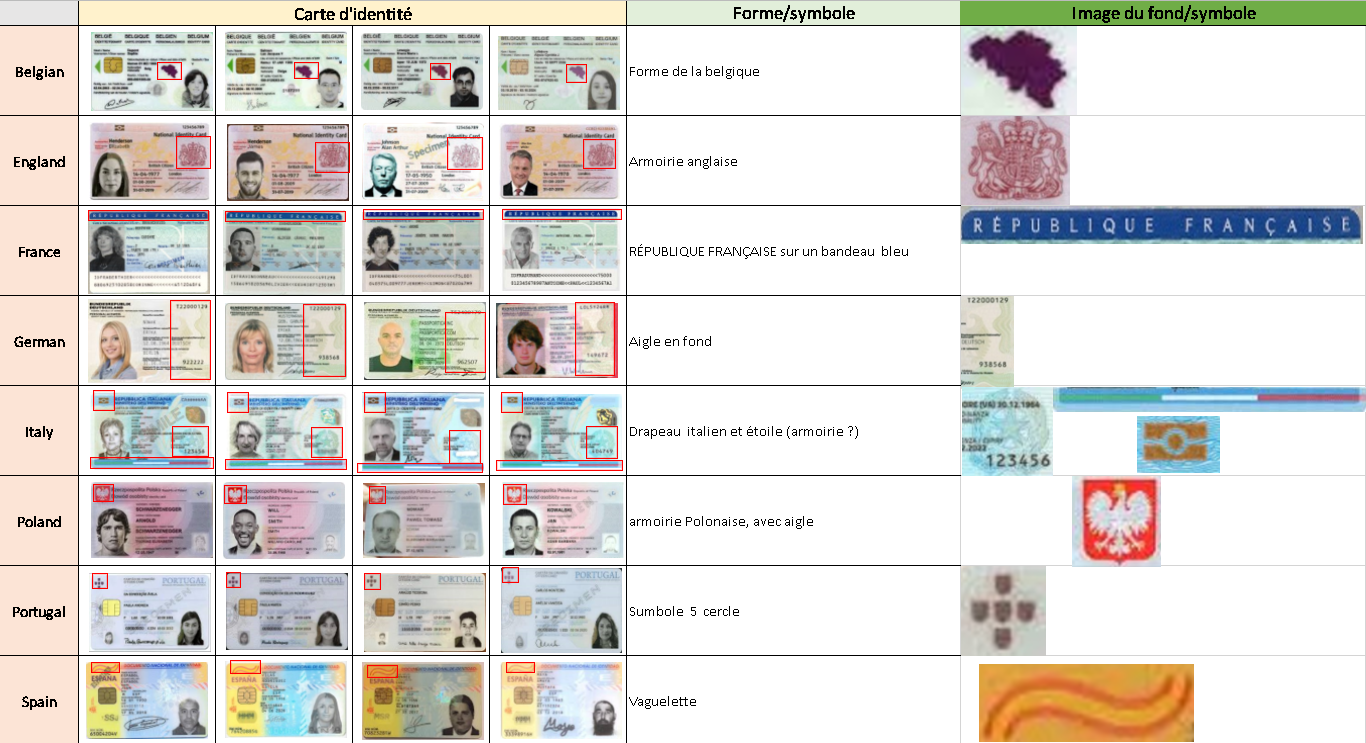
Un travail de sélection a donc été réalisé en amont. Nous avons sélectionné 1 objet par nationalité. Cet objet, de manière visuelle, représente le mieux, la nationalité pour une carte d’identité donnée.

Ex, l’aigle royal pour la Pologne ou encore le symbole de vaguelettes pour la carte d'identité espagnole.



*Carte d’identité espagnole - Identification de la nationalité par l’objet « vaguelettes » présent dans le coin supérieur gauche de la carte*

L’ensemble des objets identifiant les nationalités ont été regroupés sous forme de tableau qui nous sert de support dans notre démarche :



*Tableau récapitulatif des identifications d’objets/formes pour chaque nationalité*

A partir de cette méthode, différentes études ont donc été lancées :

Premièrement, la méthode d’Otsu, méthode de seuillage d’images afin de reconnaître et d’isoler les différents objets graphiques présents dans une image.

Le seuillage est une opération qui permet de transformer une image en niveau de gris en image binaire (noir et blanc), l'image obtenue est alors appelée masque binaire. Pour chaque masque, 2 images seront obtenues. Une ne contenant que le fond, et une ne contenant que les objets.

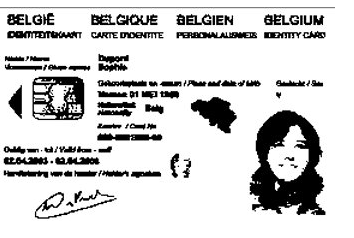
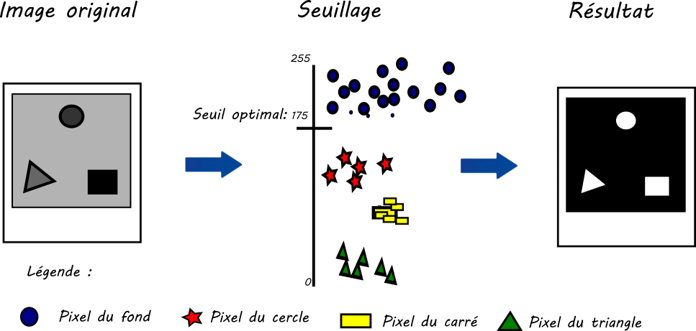
Il existe 2 méthodes pour seuiller une image, le seuillage manuel (on détermine le seuil à la main) et le seuillage automatique. Ce dernier est appelé le seuillage d’Otsu.

En pratique, le seuil optimal est considéré comme étant celui qui sépare le mieux le fond et les objets présents sur l’image.

La variance interclasse, entre les pixels du fond et ceux des objets, est calculée et le seuil du masque pour lequel l'on obtient la plus grande variance est retenu comme seuil optimal.

Ainsi nous avons testé le seuillage d’Otsu sur une carte d’identité d’une belge comme montré ci-dessous. En reprenant notre objectif initial, l’enjeu est de pouvoir reconnaître l’objet associé à la nationalité belge. En reprenant notre tableau d’objets, il s’agit des contours du pays de la Belgique situés dans la partie droite de la carte.

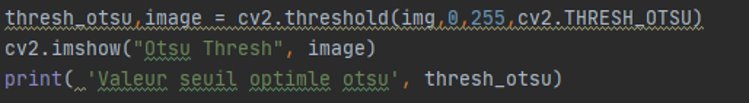
De manière générale, on remarque que les « objets » et autres formes sont situés en premier plan et sont mis en avant en noir en détriment du fond de la carte d’identité en blanc. Nous observons bien la forme et les contours du pays de la Belgique.

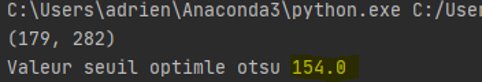
 

*Application du seuillage d’Otsu à une carte d’identité afin de mettre en avant la forme de la zone encadrée en rouge*

La fonction python permettant d’utiliser cet algorithme de seuillage d’image est la fonction «  thresold ». Elle classifie l’intensité du pixel.

Elle prend en paramètre, img : l’image à traiter, seuil : la valeur du seuil , couleur : la couleur que l’on souhaite attribuer à la zone objet et enfin le type de seuillage que l’on souhaite affecter à cette zone





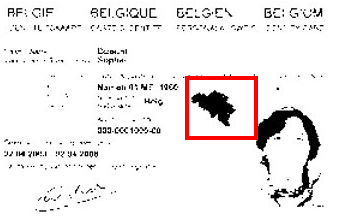
Ci-dessus, le code Python employé pour obtenir le résultat de seuillage.

Le dernier paramètre est “Tresh OTSU” indiquant que la technique du seuillage d’OTSU est utilisée.

On applique plus précisément le seuillage d’OTSU sur chaque canal de l’image (de 0 à 255). Nous obtenons un seuil optimum de 154.0. Contrairement au seuillage manuel, ce seuil est calculé automatiquement par la fonction « THRESH\_OTSU ». En effet, en fonction des caractéristiques des pixels de l’image, l’algorithme d’OTSU détermine automatiquement le seuil correspondant à la variance (écart inter-pixel) la plus grande sur la totalité de l’intervalle [0,255].

Visuellement, on remarque que les contours du pays de la Belgique sont lisses et nets, et donc proches de la situation des frontières réelles de la Belgique présente sur la carte d’origine.

Nous n’obtenons pas de quelconques pics ou autres déformations au niveau des frontières comme on peut le voir ci-dessous avec le seuillage manuel appliqué à la carte (seuil = 65).



*Mise en évide des disparités de forme au niveau des frontières de la Belgique prouvant que le seuillage manuel est moins pertinent que le seuillage d’Otsu*

De cette phase de test, nous pouvons conclure que le seuillage d’OTSU est pertinent à utiliser dans notre cas d’étude pour pouvoir récupérer les différents contours de nos objets à identifier. En effet, nous obtenons une reconnaissance des contours de l’objet adaptée et surtout réelle par rapport à notre objet au sein de l’image originale. La Belgique et ses frontières sont bien mises en valeur.

Cependant, nous nous sommes interrogés par la suite sur l’intérêt de cette technique pour pouvoir reconnaître la nationalité à partir de ce contour.

C'est-à-dire, comment notre algorithme en récupérant ce type de contour de l’objet dans l’image peut-il identifier la nationalité ?



**Nationalité ?**

Identification du contour de l’objet



C’est dans cette perspective que nous avons pensé pour la suite de notre projet à un processus d’apprentissage qui permettrait à un classificateur de classer les différentes images de cartes d’identité en différentes nationalité selon les objets identifiés.

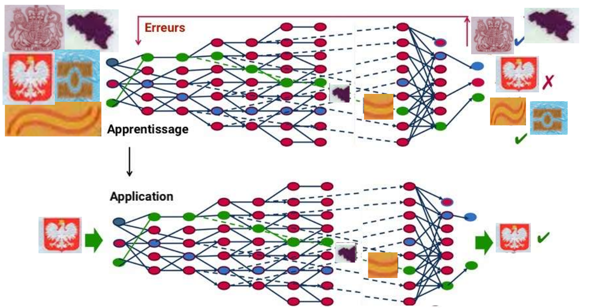
Dans le cadre de notre étude, nous allons nous intéresser principalement à la détection d'objets par Deep Learning, qui suit donc ce fonctionnement bien précis.

Pour mieux illustrer cela, considérons un exemple concret de détection des vaguelettes présentes sur une image donnée de carte d’identité. Les vaguelettes présentent sur l’image permettent de classer cette carte d’identité en tant que « espagnole ».

Afin de pouvoir détecter les vaguelettes présentes sur les images, l'algorithme doit être capable d'identifier toutes les vaguelettes, peu importe leur position ou leurs couleurs. Pour ce faire, il faut nécessairement au préalable, compiler plusieurs images des vaguelettes, associées à d'autres images contenant des objets différents en guise "d'entraînement" au Deep Learning pour l'algorithme.

Ainsi de manière générale, chaque image compilée est convertie en données et transmise sur un réseau de neurones ; ces derniers attribuent un poids à chaque nœud du réseau, et la couche finale émet une réponse à toutes les informations collectées et identifie ou non les vaguelettes présentes sur l'image.

Cette identification est ensuite comparée aux résultats donnés par l’utilisateur : les "bonnes" réponses sont gardées en mémoire afin d'être utilisées pour les tests autonomes suivants, et les "mauvaises" sont étudiées à nouveau jusqu'à corriger les problèmes rencontrés lors de la détection (en modifiant l'assignation des poids aux noeuds par exemple).



*Apprentissage par réseaux de neurones afin de définir les nationalités de nos cartes d’identité*

Cette méthode est appelée "apprentissage supervisé", car elle nécessite la participation et la validation par l'humain. Ce sera cette méthodologie qui sera employée par la suite.

Aussi, afin de détecter l'objet de notre choix, il sera nécessaire de développer notre propre classificateur. La détection d'objets à l'aide de classificateurs en cascade basés sur les fonctionnalités Haar constitue en autres une sérieuse piste pour améliorer notre score de précision sur la carte d’identité.

# IV – Généralisation du processus d’identification

# V – Industrialisation de notre solution à l’aide de l’outil Flask

# VI - Conclusion et perspectives

Comme évoqué tout le long de ce dossier, nous avons tenté de récupérer la nationalité à travers nos documents de deux différentes manières : par le texte et les formes et contours des objets les composant.

La première partie concerne donc l’exploitation par le texte, avec ou sans modification de l’image, allant de l’agrandissement, jusqu’à la technique du Grayscalling, afin d’avoir les meilleurs résultats possibles. Les librairies Pytesseract et passortEye nous ont permis d’obtenir un taux de reconnaissance allant jusqu’à 56% pour les cartes d’identité, 47% de réussite pour les passeports et 32% pour les Visas. Ces résultats sont encore insuffisants mais restent tout de même correctes sur 2 mois de travaux.

La deuxième partie nous a permis d’identifier la nationalité à travers les formes et autres contours des objets présents sur les cartes d’identité. Elle vient en complément de la première partie, notamment pour l’identification sur les cartes d’identité. A l’aide de la méthode d’OTSU les contours des objets ont pu être identifiés.

Cependant, l’exploitation des formes et contours des objets reste encore incomplète. Pour faire apprendre notre algorithme à reconnaitre les formes des objets et à les faire correspondre à des nationalités, il sera nécessaire par la suite de développer un processus d’apprentissage par Deep Learning. Cette technique d’apprentissage par réseaux de neurones sera pertinente dans notre cas pour pouvoir associer de manière automatisée une nationalité à une carte d’identité par l’intermédiaire d’un objet présent sur celle-ci.

De plus, au-delà de la nationalité, il serait pertinent de récupérer l’ensemble des informations présentes sur un document officiel et d’en ressentir pour chaque individu, l’ensemble des informations associées sous un format de Data Frame facilement exploitable et présentable. L’intérêt est d’aller encore plus loin qu’une simple recherche de de la nationalité mais de récupérer l’ensemble des caractéristiques d’un individu à travers un document officiel.

Ce projet nous plaît et il reste encore d’innombrables développements à réaliser.