PROJET FÉDÉRÉ
2022/2023

DÉTECTION DES PIÈCES MONNAIS POUR LES AVEUGLES

Presenté par

YASSINE HATTAY &
SAMAR HAMRAOUI

Encadrant

MR KAMEL CHAIEB

Plan

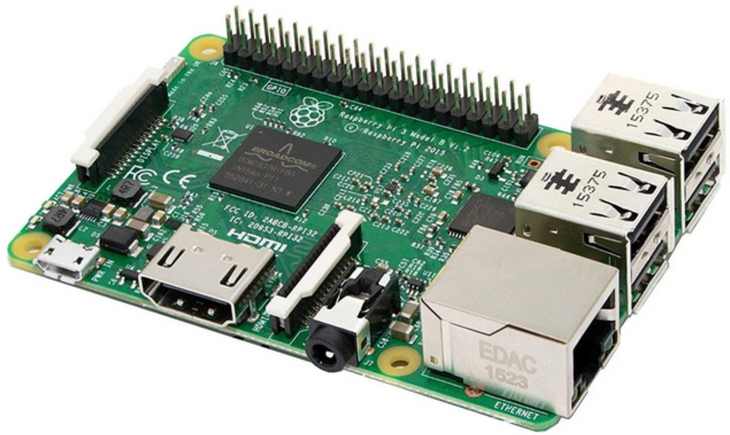
- Contexte
  - Problématique
  - Solution
- Analyse
  - Analyse
  - code python sur la carte
  - code python du serveur
- Realisation
  - Choix matériel
  - choix de model
  - Préparation de données
  - Entrainement du modèle
- Clôture

Contexte

Problématique

Les personnes aveugles rencontrent souvent des difficultés lorsqu'il s'agit de différencier les différents valeur des pièces de monnaie en leur possession

Solution



envoi d'une photo des pièces →

← réception des résultats de détection

Analyse

Comment l'utilisateur déclenche-t-il l'envoi de l'image et la réception des résultats ?

et comment le processus se déroule-t-il en détail ?

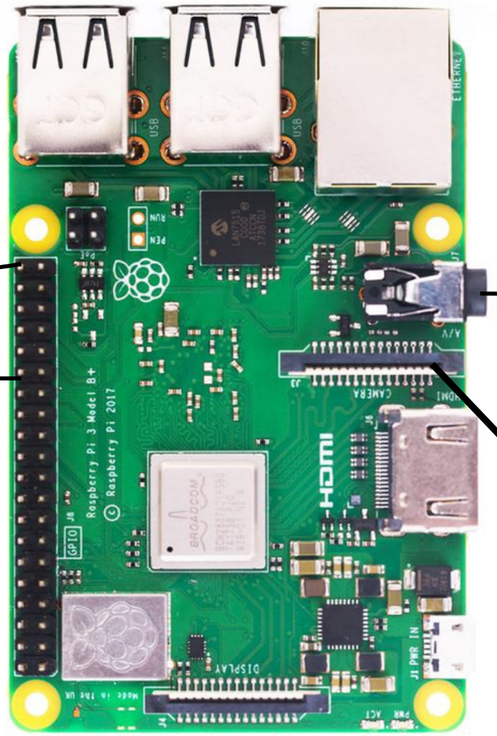
Analyse du carte

Envoi du photo au serveur 3

4 Réception des résultat

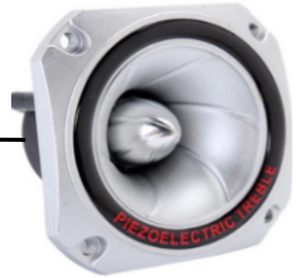
Cliquer sur le bouton

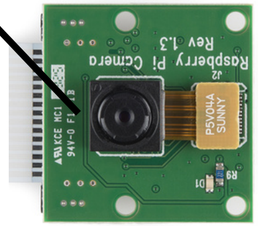
1



Annonce auditif

5





Prendre une photo

2

Code python sur la carte

1-importation : `**requests**` , `**picamera**`,`**RPI.GPIO**` et `**pyttsx3**`

2-Configuration du camera

3-Configuration du broches

4-Definition de la fonction du rappel

5-Détection d'événement sur la broche du bouton

Analyse du serveur

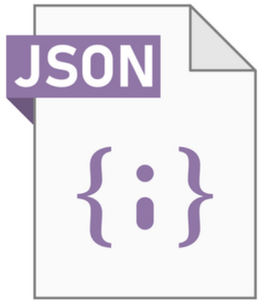
Détection les types des pièces et le nombre de chaque type

Serveur

Réception du photo 3

4 **Envoi du résultat en format json**

Code de serveur



1- Importation: `**argparse**`, `**json**`, `**torch**`, `**Flask**` et `**Popen**`

2- Configuration de l'application Flask

3- Définition de les routes

- **a**- Route d'index : index.html
- **b**- Route de détection d'objets

4- Fonction principale

PyTorch

Flask

Réalisation

Choix materiél

pour la choix matériel en raison de la disponibilité de ressources en ligne et de tutoriels, la carte Raspberry paraissait être un bon choix pour ce projet .

Pourqoi un serveur ?

En utilisant un serveur, nous pouvons fournir les informations nécessaires à l'utilisateur dans un temps raisonnable avec l'aide d'un modèle d'apprentissage profond qui serait sinon plus lent et moins précis dans une carte raspberry.

Choix du model

Algorithme d'apprentissage profond

- -Rapide
- -Bonne précision
- -Polyvalent

Preparation des données

Script python

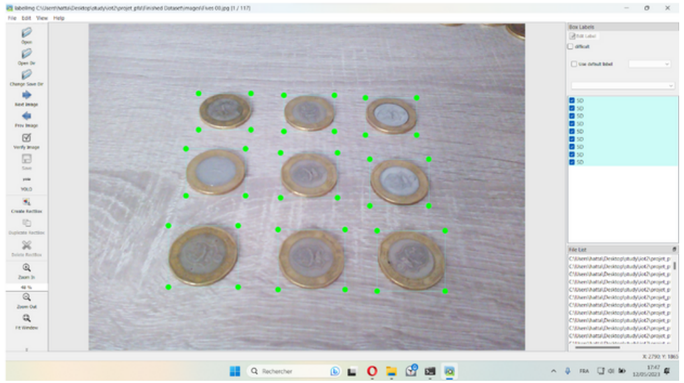
.XML

.TXT

la préparation des données

étiquetage des photos

LabelImg



Résultat

Fives 00.txt

Fives 00.txt

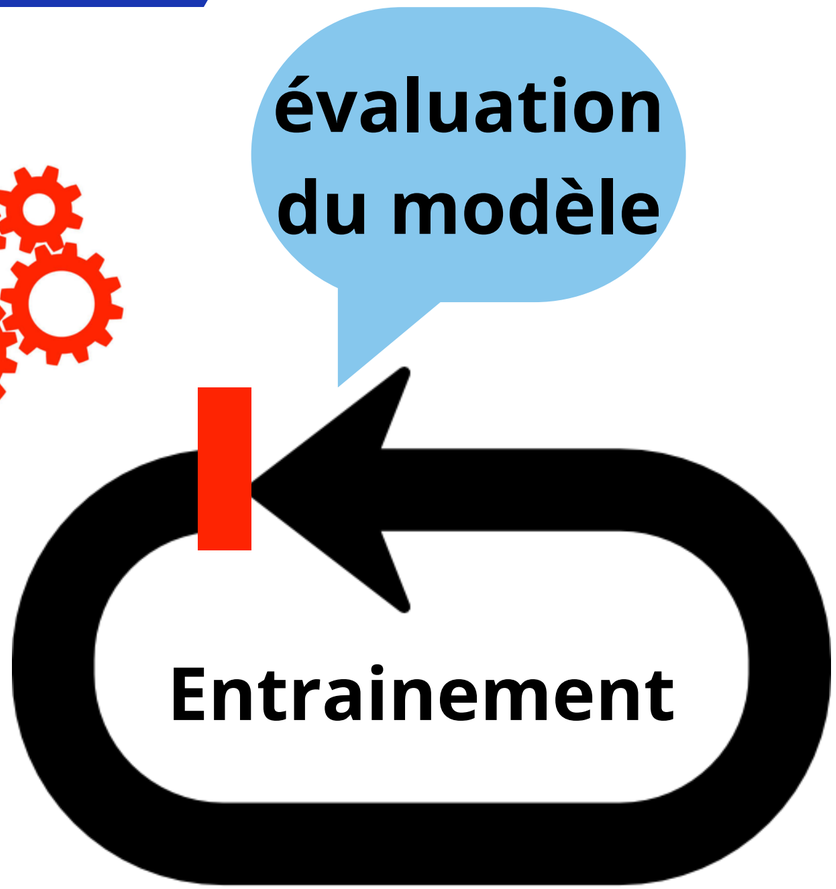
Fichier Modifier Affichage

```
0 0.3119212962962963 0.26954732510288065 0.11998456790123457 0.1111111111111111
0 0.5054012345679012 0.2749485596707819 0.11574074074074074 0.11059670781893004
0 0.6929012345679012 0.28420781893004116 0.11728395061728394 0.11368312757201646
0 0.29128086419753085 0.4549897119341564 0.13657407407407407 0.14351851851851852
0 0.5050154320987654 0.4642489711934156 0.13040123456790123 0.13734567901234568
0 0.7118055555555556 0.48636831275720166 0.13503086419753085 0.14248971193415638
0 0.26273148148148145 0.7121913580246914 0.1597222222222222 0.18775720164609053
0 0.5108024691358025 0.7219650205761317 0.15354938271604937 0.18364197530864199
0 0.7388117283950617 0.7260802469135802 0.1558641975308642 0.18467078189300412
```

Entrainement du modèle:

PyTorch

configuration des paramètres



Clôture

Probléme

Ce modèle détecte parfois des types des pièces incorrects

Comment le rendre plus performant?

1
Augmentation des données

2
Équilibrage des données

3
Ensembles de modèles

Merci pour votre attention

N'hésitez pas à poser vos questions