**1. Collecte et préparation des données**

**Données d'entraînement :**

* **Images des doigts et chiffres :** Rassemblez un ensemble d'images qui montrent des mains avec les doigts formant des chiffres (0 à 9) et des signes d'opérations (+, -, \*, /). Vous pouvez prendre ces photos vous-même ou rechercher des jeux de données existants.
* **Annotation des données :** Utilisez un outil comme [LabelImg](https://github.com/tzutalin/labelImg) pour annoter les images en créant des fichiers au format YOLO. Chaque image doit avoir un fichier .txt correspondant contenant les coordonnées des boîtes englobantes et les étiquettes (chiffres ou opérateurs).

**2. Configuration de l'environnement**

**Outils et bibliothèques nécessaires :**

* **Python 3.x**
* **YOLOv8 :** Installez la bibliothèque YOLOv8, qui peut être faite en clonant le dépôt depuis GitHub :

bash

Copier le code

git clone https://github.com/ultralytics/yolov8.git

cd yolov8

pip install -r requirements.txt

**3. Entraînement du modèle YOLOv8**

1. **Préparez votre ensemble de données :** Assurez-vous que vos images et annotations sont organisées de manière appropriée.
2. **Configurez les paramètres d'entraînement :** Créez un fichier de configuration pour YOLOv8 spécifiant vos classes (dix chiffres et quatre opérations). Exemple de fichier data.yaml :

yaml

Copier le code

train: ./data/train # chemin vers vos images d'entraînement

val: ./data/val # chemin vers vos images de validation

nc: 14 # nombre total de classes (10 chiffres + 4 opérations)

names: ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '+', '-', '\*', '/']

1. **Démarrez l'entraînement :** Exécutez la commande suivante pour commencer l'entraînement :

bash

Copier le code

python train.py --img 640 --batch 16 --epochs 50 --data data.yaml --weights yolov8n.pt

**4. Détection en temps réel**

1. **Utilisez votre modèle entraîné :** Une fois le modèle entraîné, vous pouvez l'utiliser pour détecter les chiffres et les opérations. Utilisez le code suivant pour la détection en temps réel via une webcam :

python

Copier le code

import cv2

from yolov8 import YOLO

model = YOLO('runs/train/exp/weights/best.pt') # Chargez le modèle entraîné

cap = cv2.VideoCapture(0) # Ouvrir la webcam

while True:

ret, frame = cap.read()

results = model(frame)

annotated\_frame = results[0].plot() # Afficher les résultats

cv2.imshow('Detection', annotated\_frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

**5. Traitement des résultats**

1. **Interpréter les résultats :** Une fois que les chiffres et les opérations sont détectés, vous devez les convertir en une expression mathématique. Vous pouvez utiliser les coordonnées de chaque boîte englobante pour déterminer l'ordre des chiffres et des opérations.
2. **Calculer le résultat :** Une fois l'expression mathématique construite, utilisez eval() en Python pour calculer le résultat :

python

Copier le code

expression = '3 + 5' # Exemple d'expression

result = eval(expression)

print(f"Resultat : {result}")

**6. Améliorations possibles**

* **Améliorer le modèle :** En fonction des résultats de détection, vous pourriez avoir besoin d'ajuster votre ensemble de données ou d'augmenter les données pour de meilleures performances.
* **Interface utilisateur :** Vous pouvez développer une interface utilisateur simple pour rendre le projet plus interactif.