Autó típus felismerő modell

Fehér Máté (HWYLDB)

1. Bevezetés

A project egy általunk feltöltött kép alapján megpróbálja felismerni (kisebb nagyobb sikerrel), hogy a három megadott autótípus (SUV/Sedan/Sport) közül, melyikre hasonlít a legjobban. Az adatokat a Roboflow segítségével raktam össze, ez segít a képek felismerésében. A felület megjelenítést a Streamlit segítségével valósítottam meg.

2. Modell tanítás (Yolo)

Először cmd-ben telepítettem, az ultralytics-et ami kell a Yolo-hoz. Ezt az alábbi kóddal csináltam:

```
pip install ultralytics
```

Ezután a Robotflow oldalon létrehoztam egy projectet, ahova feltöltöttem képeket a háromfajta autó típusról. Ezután be kellett őket csoportosítani, hogy melyik kép melyik osztályhoz tartozik, és az adott osztályhoz hozzárendelt az oldal egy feliratot, pl.: az SUV autókat egy zöld kerettel kijelölte, és belerakta a suv osztályba.

Ezután belehetett állítani, hogy többféle tulajdonságot is figyelembe vegyen a képek vizsgálatakor, pl.: forgatva van a kép, fekete fehér a kép, homályos a kép. Ezután már csak exportálni kellett az adatokat ezek találhatók meg a Car_types_Yolo mappában.

Ebben a mappában található a train mappa, amiben vannak a képek és a hozzá tartozó feliratok. Itt látható az oldal által szerkesztett kép, amivel tudjuk tanítani az alkalmazást.

Minden képhez tartozik egy label is, ami leírja, hogy a kép melyik osztályba tartozik, az objektum (azaz a bekeretezett rész) középpontját (x,y), magasságát és szélességét. Az alábbi példánál a 0. azt jelenti, hogy az SUV osztályba tartozik, a következő két tizedes szám az objektum középpontját határozzák meg (középpont x-koordinátája / kép szélessége | középpont y-koordinátája / kép magassága), ezután látható, hogy az objektum szélessége a kép szélességének 97%-a, a magassága pedig a kép magasságának 88%-a.

```
0 0.5109375 0.4428969359331476 0.978125 0.8857938718662952
```

A következő lépés az volt, hogy készítettem egy betanitas.py fájlt, ami az általunk generált adatok alapján betanítja a modellt a Yolov8 segítségével.

```python
from ultralytics import YOLO

# Modell betöltése
model = YOLO('yolov8n.pt')

# Tanítás
model.train(data='./Car_types_Yolo/data.yaml', epochs=50, imgsz=640)
```

A data.yaml fájlban vannak a feltanításhoz szükséges adatok, és a link a robotflow-hoz, ahol eltudja érni az általam generált adatbázist.

```yaml
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 3
names: ['sedan', 'sport', 'suv']

roboflow:
  workspace: fehermate852
  project: car-types-adznc
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/fehermate852/car-types-adznc/dataset/1
```

3. Felhasználói felület (Streamlit)

Ahhoz, hogy működjön a felület megjelenítés, le kellett telepíteni a streamlit-et a cmd-ben a következő kóddal: `pip install streamlit`

Ezután létrehoztam a beadando.py fájlt, amiben importálni kellett a szükséges modulokat (streamlit, yolo, image, tempfile (a konvertálás miatt)).

```
import streamlit as st
from ultralytics import YOLO
from PIL import Image
import tempfile
```

Ezután betölt a modell az előzőleg lefuttatt betanitas.py fájl által generált adatokból. A következő kódrészlet az volt, hogy felépítettük a felületet, megjelenik egy cím, egy kép feltöltő ablak és egy üzenetet megjelenítő rész. A kép feltöltésénél kizárólag .jpg, .png, és .jpeg formátumokat fogad el a mező, viszont került bele egy .jpg-vé konvertálás, mert nem minden képnél volt sikeres a feltöltés.

```
# Modell betöltése
model = YOLO('runs/detect/train/weights/best.pt')

st.title("Autótípus felismerő")
uploaded_file = st.file_uploader("Tölts fel egy autó képet", type=['jpg', 'png', 'jpeg'])

if uploaded_file:
    try:
        # Kép betöltése
        image = Image.open(uploaded_file)

        # Átmeneti fájlba mentés
        with tempfile.NamedTemporaryFile(delete=False, suffix='.jpg') as temp:
            image.save(temp.name)
            temp_path = temp.name
```

Ezután megjelenik a felhasználó által megadott kép, és ez a rész után kezdődik el, a kép vizsgálata (predikció). Megpróbálja felismerni a feltöltött képet, és ezután kiír egy bizonyosságot, hogy szerinte melyik típusra passzol a leginkább a három közül.

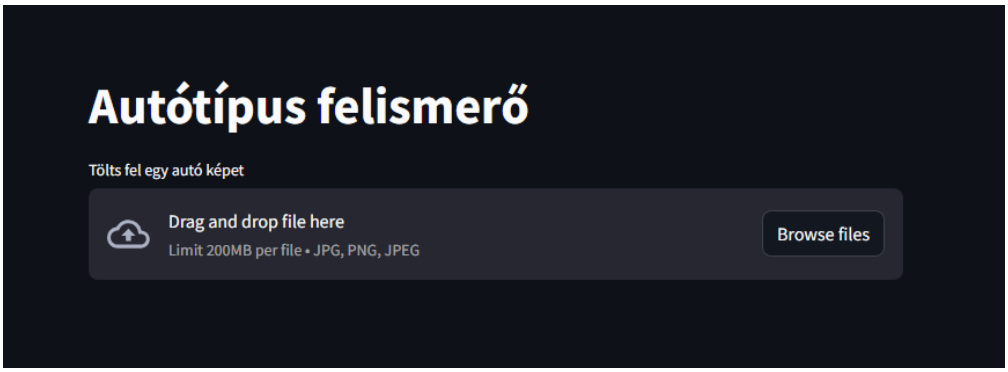
```
        # Predikció
        results = model.predict(source=temp_path)
        st.image(image, caption='Feltöltött kép', use_column_width=True)
        st.write("Eredmények:")
        for box in results[0].boxes:
            cls = box.cls.cpu().numpy().item()
            confidence = box.conf.cpu().numpy().item()
            st.write(f"Osztály: {model.names[int(cls)]}, Bizonyosság: {confidence:.2f}")

    except Exception as e:
        st.error(f"Hiba történt: {e}")
```

4. Tesztelés

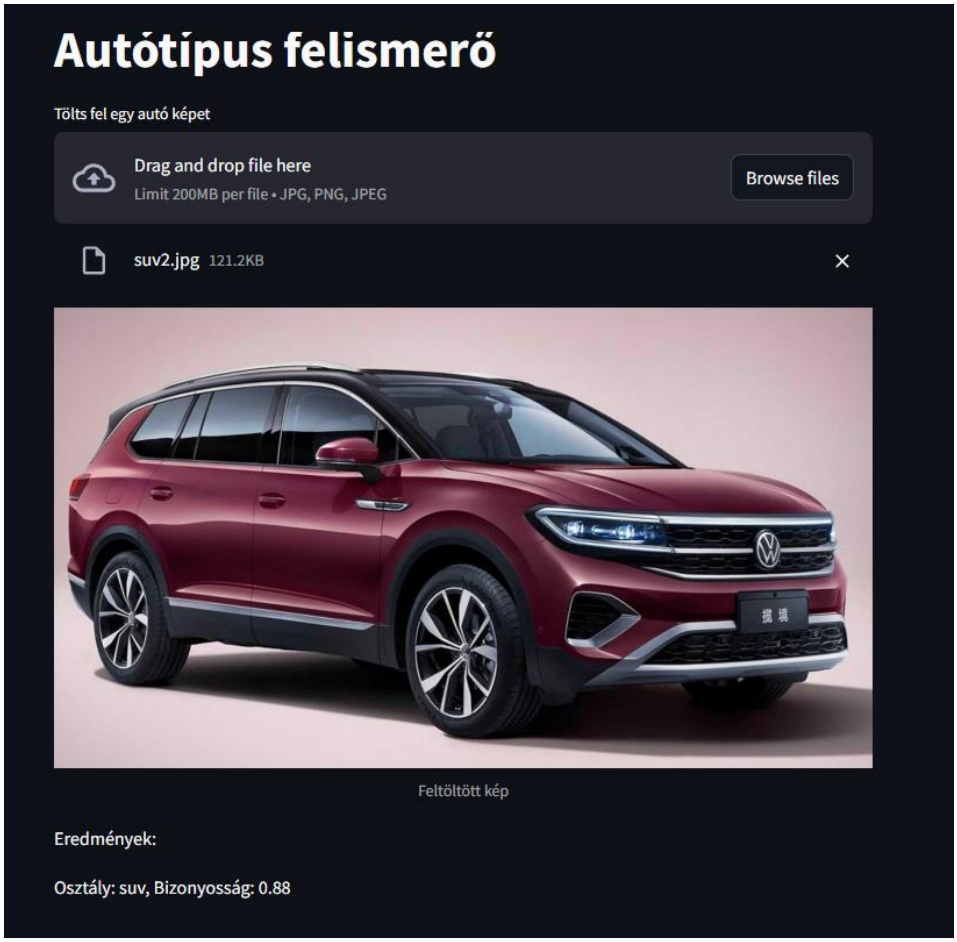
Tesztelni úgy tudtam, hogy a cmd-be beírtam a következő parancsot:

```
streamlit run beadando.py
```

Ezután kapunk egy sikeres üzenetet, hogy létrejött az alkalmazás az alapértelmezett web böngészőnkön. A felület így néz ki:



Itt fel tudunk tölteni egy képet, amit megjelenít és ha talál hasonlóságot, akkor alá kiírja, hogy mennyire hasonlót az adott osztályra (0-1). Amennyiben nem hasonlít, nem ír ki semmit.



5. Használati útmutató

Először a github-ról ki kell másolni a linket, és ezt klónozni, pl.: Visual Studio-ban. Le kell tölteni egy python verziót (pl.: 3.10), majd cmd-ben telepíteni a következő parancsokat.:

- pip install ultralytics
- pip install streamlit

(Ha a pip nem működik ez a kód segíthet előtte: py -m pip install apache-airflow)

Következő lépés, a %appdata% (roaming) mappában az Ultralytics mappán belül a settings.json file-ból ki kell törölni a „\\dataset” -et hogy ennyi maradjon a végén:

(A piros rész az elérési út, ez minden esetben más, attól függ, hova mentjük a projectet)

Ezután le kell futtatni a betanitas.py fájlt, ez eltarthat néhány percig, amint végzett le kell futtatni utána a beadando.py fájlt is.

Ezek után a cmd-be be kell írni a következő kódot: streamlit run beadando.py

(Fontos, hogy a megadott mappába legyünk amikor beírjuk a kódot, különben nem fogja megtalálni a fájlt.)

Ameddig a cmd (ahova a streamlit run beadando.py kódot írtuk) meg van nyitva, addig fut a weboldal is, amint az bezárjuk, már nem lesz elérhető, ezután újra meg kell nyitni.

Ha ezt beírtuk meg fog jelenni a weboldal, ahol már tudjuk is tesztelni a programot egy kép feltöltésével. Vannak előre letöltött képek a letöltött fájlok között van egy „kepek” mappa, ahol a kép neve a kocsi típusát is jelzi. Ezekkel is le tudjuk tesztelni a program működését.

Ha probléma merül fel a kép megjelenítésénél, vagy figyelmeztetést kapunk, akkor ezt a sort módosítani kell:

```
st.image(image, caption='Feltöltött kép', use_container_width=True)
```

A „use_container_width”-t problémázhat a python verzió miatt, ugye attól függ, milyen van a gépen, ha ez felmerül át kell írni erre:

```
st.image(image, caption='Feltöltött kép', use_column_width=True)
```

A „use_column_width” megoldaja ezt a problémát, és ez érvényes oda-vissza verziófüggően.