Dokumentacja techniczna projektu zaliczeniowego

Programowanie w języku Python 2

Inez Małecka (52738)

Stworzono przy użyciu LaTeXw programie TeXmaker
na licencji GNU GPL.

1 Opis projektu

1.1 Opis projektu, wprowadzenie

Program stanowiący przedmiot niniejszego opracowania powstał jako projekt na zaliczenie zajęć "Programowanie w języku Python 2"prowadzonych w trakcie drugiego semestru informatyki w trybie niestacjonarnym (nabór zimowy 2023/2024).

Z racji iż do zaliczenia projektu wymagane było wykorzystanie modułów obsługujących uczenie maszynowe, autorka czuła zobowiązana się wymyślić oraz zaimplementować program który realnie znalazłby zastosowanie w jej pracy zawodowej - branża budownictwo drogowe.

W zamyśle program ma zliczać ilość przejeżdżających pojazdów w celu stworzenia analizy ruchu i tworzenia kartogramów na podstawie uzyskanych danych. W związku z ograniczoną ilością czasu, autorka podjęła się niecnej decyzji jaką było użycie gotowego modelu detekcji obrazów dostępnego na portalu *TensorFlow Hub*. W związku z tym funkcjonalność programu znacznie się ograniczyła, a program jest w stanie określać jedynie przybliżoną ilość pojazdów jadących na prostej drodze. W przyszłości przewiduje się stworzenie autorskiego modelu YOLOv8 który wykrywałby nie tylko różne typy pojazdów a również kierunki ich poruszania.

Uwaga! Program testowano jedynie na komputerach z systemem MacOS oraz na przeglądarce Safari

1.2 Funkcje zaimplementowane

Z racji na wczesne stadium w/w programu, nie wszystkie funkcje zostały dotychczas zaimplementowane, niektóre zostały zaimplementowane jedynie w zubożałej formie. Obecnie program umożliwia:

- Dostęp do prostej aplikacji internetowej,
- Możliwość wgrania filmu w formatach .mp4, .avi, .mov.
- Identyfikacja obiektów na nagraniu poprzez dzielenie nagrania na pojedyńcze klatki, na których przy pomocy modelu uczenia maszynowego przerysowany jest zakres wykrycia obiektu, wraz z nazwą obiektu oraz pewnością algorytmu dot. obiektu.
- Możliwość odtworzenia nagrania z poziomu interfejsu użytkownika.
- Wyświetlenie przybliżonej ilości pojazdów na nagraniu.

1.3 Planowane implementacje oraz aktualizacje

- Stworzenie własnego modelu na bazie YOLO v8, służącego do specjalistycznej identyfikacji obiektów na nagraniu.
- Dodanie opcji modyfikacji kolorów oznaczeń identyfikacji z poziomu panelu użytkownika.
- Dodanie możliwości tworzenia kartogramów na otrzymanych od modelu danych.
- Możliwość pobrania przetworzonego filmu oraz wygenerowanych kartogramów.
- Dodanie paska postępu przetwarzania wraz ze średnim czasem oczekiwania i procentem przetworzenia na panelu użytkownika.

Uwaga! przed podjęciem się dalszych prac, należy porównać środowiska programistyczne oraz dostępne języki programowania. Należy dokonać ewaluacji spójności kodu oraz przestrzeni czasowej oraz pamięciowej algorytmów.

1.4 Środowisko programistyczne

Projekt ten został stworzony w zintegrowanym środowisku programistycznym (IDE) PyCharm 2023.3.2 (Professional Edition) z wykorzystaniem języka Python3.12.

2 Informacje szczegółowe

2.1 Deklaracja bibliotek wykorzystanych w projekcie

Wycinek z app.py - biblioteki

```
import os
from flask import Flask, request, redirect, url_for, render_template,
                    send_from_directory
from przetwarzanie import przetworz_wideo
from installrequirements import zainstaluj_biblioteki
```

Wycinek z przetwarzanie.py - biblioteki

```
import os
import cv2
import numpy as np
import tensorflow as tf
import tensorflow_hub as hub
```

Wycinek z installrequirements.py - biblioteki

```
import os
import subprocess
import sys
```

Na powyższym wycinku kodu można zaobserwować wszystkie biblioteki jakie zostały wykorzystane do stworzenia niniejszego programu. Wszystkie są niezbędne do prawidłowego działania programu w sposób zaprojektowany przez autorkę.

- **os**: Biblioteka os umożliwia interakcję z systemem operacyjnym.
- **sys**: Moduł sys zapewnia dostęp do niektórych zmiennych i funkcji związanych ze środowiskiem uruchomieniowym Pythona.
- **flask**: Framework Python do tworzenia aplikacji internetowych. Jest to lekki i elastyczny framework, który pozwala na szybkie i łatwe tworzenie prostych i bardziej zaawansowanych aplikacji webowych.
- **cv2**: Biblioteka funkcji wykorzystywanych podczas obróbki obrazu.
- **numpy**: Biblioteka obsługująca wielkoformatowe tabele oraz macierze.

- **tensorflow**: Biblioteka wykorzystywana w ML i sieciach neuronowych
- **tensorflow hub**: Biblioteka służąca pobieraniu i używaniu gotowych modeli dostępnych na w/w portalu.

3 Procedury testowe

3.1 Opis procedur testowych

Procedury testowe stanowią kluczowy element procesu tworzenia oprogramowania. Opracowywane są one w celu przetestowania różnych aspektów systemu podczas jego rozwijania. Procedury testowe obejmują zestawy kroków i testów, które są wykonywane w celu sprawdzenia, czy oprogramowanie działa zgodnie z oczekiwaniami i spełnia określone wymagania.

Podczas opracowywania procedur testowych uwzględniono różne scenariusze i warunki, które mogą wystąpić podczas użytkowania programu. Procedury te obejmują testowanie wszystkich funkcji dotychczas zaimplementowanych, interakcji użytkownika, obsługi błędów oraz wydajności systemu.

Schemat opracowanych procedur testowych znajduje na końcu dokumentu. Do testów poproszono osobę nie związaną z kodem by przeprowadziła test zgodnie z przedstawionym schematem oraz próbowała używać funkcji programu w sposób jak najbardziej zbliżony do naturalnego użytkowania kodu by sprawdzić czy wszystkie procedury zostały odpowiednio zawarte.

Tester aplikacji przeprowadził 20 prób na różnej rozdzielczości ekranu na 2 komputerach z systemem MacOS - Macbook Air M1 oraz Macbook Pro M2. Tester zgłosił znaczną rozbieżność między ilością wyświetlanych kwadratów detekcji a liczbą pojazdów wyświetlonych na nagraniu - po głębszej analizie, należałoby użyć lepiej wytrenowanego i bardziej wyspecjalizowanego modelu ML.

3.2 Zestawienie testów

Numer testu	Rozdzielczość ekranu	Błąd w procedurze	Jakość obrazu
1	1280 x 800	NIE	wysoka
2	1280 x 800	TAK	wysoka
3	1280 x 800	TAK	wysoka
4	1280 x 800	NIE	wysoka
5	1280 x 800	TAK	wysoka
6	1140 x 900	NIE	wysoka
7	1140 x 900	NIE	wysoka
8	1140 x 900	TAK	wysoka
9	1140 x 900	NIE	wysoka
10	1140 x 900	NIE	wysoka
11	1650 x 1050	TAK	niska
12	1650 x 1050	NIE	niska
13	1650 x 1050	TAK	niska
14	1650 x 1050	NIE	niska
15	1650 x 1050	NIE	niska
16	1680 x 1050	TAK	średnia
17	1680 x 1050	NIE	średnia
18	1680 x 1050	TAK	średnia
19	1680 x 1050	TAK	średnia
20	1680 x 1050	NIE	średnia

3.3 Schematy procedur testowych