VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Projektová dokumentácia k predmetu SUR

Detektor osoby

19. apríla 2020

Dominik Boboš (xbobos00)
Tomáš Kender (xkende01)

1 Úvod

Zadaním projektu je natrénovať detektor jednej osoby z obrázku tváre či hlasovej nahrávky. Detektor je implementovaný v programovacom jazyku `Python 3.8`.

2 Spustenie a používanie detektoru

2.1 Inštalácia knižníc

Na spustenie všetkých súčastí programu je potrebné mať nainštalované všetky potrebné knižnice a súčasti, ktoré sú obsiahnuté v súbore `requirements.txt`.

Ich inštalaciu je možné uskutočniť pomocou `pip`:

```
$ pip install -r requirements.txt
```

Prípadne:

```
$ pip3 install -r requirements.txt
```

2.2 Dátová štruktúra

Program očakáva nasledujúcu dátovú štruktúru.

```
-- |-- data              // dáta určené k~trénovaniu a testovaniu detektoru
       |-- target_dev
       |-- non_target_dev
       |-- target_train
       |-- non_target_train
   |-- eval              // dáta určené ku klasifikácií
   |-- src               // zdrojové kódy
-requirements.txt
```

2.3 Spustenie jednotlivých súčastí detektoru

Spustenie detektoru je možné pomocou príkazu.

```
$ python src/detector.py
```

Prípadne:

```
$ python3 src/detector.py
```

Po spustení sa natrénujú modely trénovacími dátami, modely sa uložia do priečinku `src` a následne sa uskutoční klasifikácia dát v `eval` a uložia sa výsledky do .txt súboru. Funkcie detektoru je možné upravovať pomocou prepínačou na príkazovom riadku.

```
--verbose   // informácie pre ladenie
--plot      // vykreslí informácie získané počas trénovania
--evalonly  // vyhodnotia sa len dáta v~/eval (potrebné mat' uložené modely)
--trainonly // trénovanie modelu bez vyhodnotenia
--cnn       // použitie CNN modelu pre spracovanie obrázkov
```

3 Použité modely

3.1 Modely pre spracovanie obrázkov

Pred samotným trénovaním modelov je dôležité získať čo najviac trénovacích a testovacích dát. Toto je docielené tým, že na trénovanie sa používajú všetky *.png* súbory v priečinku `data`. Obrázky sa načítajú pomocou knižnice *Pillow*. Target dátam sa nastaví label 1, ostatným label 0. Pomocou *ImageDataGenerator* z knižnice *Keras* sa vytvoria ďalšie trénovacie dáta a to vďaka modifikácií ako posun, rotacia, skrivenie, prevrátenie. Takto získané dáta smerujú modelom na trénovanie.

3.1.1 Model CNN

CNN je model konvolučných neuronových sietí, bol vybraný na základe zvedavosti, či CNN dokáže na akceptovateľnej úrovni rozpoznať tváre. Tento model získava features obrázkov z rôznych vrstiev, známe aj ako *feature maps*, pomocou *pooling-u* postupne matice „zmenšujeme“, respektíve podvzorkujeme. Nakoniec použijeme funkciu *flatten* čím vytvoríme 1D pole. Ako optimalizátor je použitý **SGD** - *Stochastic-Gradient Descent*, a aby sme zabránili pretrénovaniu, dáta sa predkladajú v náhodnom poradí a zo všetkých dát sa náhodne vybere 20%, ktoré sa budú používať ako testovacie dáta. Trénovanie modelu potom prebieha v desiatkách *Epochs* a samotné trénovanie trvá pomerne dlho. Model sa po trénovaní uloží `src/modelCNN.h5`.

3.1.2 Lineárny model SVM s SGD

Ako druhý model bol zvolený **SGDclassifier** použitý z knižnice *sklearn.linear_model*. Používa lineárny klasifikátor **SVM** - *Support Vector Machines* s trénovaním dát SGD. Model bol vybraný hlavne ako o dosť rýchlejšia alternatíva k CNN modelu vzhľadom na trénovanie modelu a pritom so stále akceptovateľnou úspešnosťou.