Projektová dokumentace

Strojové učení a rozpoznávání

Rozpoznávání obličeje

Vstupní data

Vstupní data jsou načtena pomocí funkce „png2fea“ z knihovny *ikrlib*. Následně převedena do tensorů. Data jsou škálována do tvaru (počet dat, šířka obrázku, výška obrázku, kanály). Šířka i výška jsou nastaveny na 80px. Jelikož se jedná o RGB obrázky, jsou tedy 3 kanály. Upravená data jsou přetypována na hodnoty typu *float32* a normalizována do hodnot [0-1]. Data jsou rozdělena do dvou tříd. Jako cílová data a necílová data.

Počet trénovacích dat: 152, z toho 20 cílových a 132 negativních

Počet validačních dat: 70, z toho 10 cílových a 60 negativních.

Model

Typ modelu: konvoluční neuronovou síť.

Hierarchie sítě: Síť je sestavena z 13 vrstev.

Aktivační funkce: *LeakyReLU*, pro poslední vrstvu použita funkce *softmax*

Kompilace modelu: Jako ztrátová funkce je použita funkce *categorical_crossentropy*, optimalizace pomocí *rmsprop*

Trénování modelu: Trénování probíhá v 7 epochách po 64 dávkách. Tyto hodnoty byly experimentálně upravovány pro dosažení co nejlepších výsledků. Pro trénovací data síť dosahuje úspěšnosti 85,71%.

Natrénovaný model je uložen jako „cnn_image.h5“

Rozpoznávání hlasu

Vstupní data

Vstupní data jsou načtena pomocí funkce „wav16khz2mfcc“ z knihovny *ikrlib*. Následně jsou data přetypována na hodnoty typu *float32* a normalizována do hodnot [0-1]. Třídy pro trénovací i validační označení jsou vektrorizována pomocí funkce „to_categorical“. Data jsou rozdělena do dvou tříd. Jako cílová data a necílová data.

Počet trénovacích dat: 152, z toho 20 cílových a 132 negativních

Počet validačních dat: 70, z toho 10 cílových a 60 negativních.

Model

Typ modelu: neuronovou síť.

Hierarchie sítě: Síť je sestavena z 5 vrstev.

Aktivační funkce: *relu,* pro poslední vrstvu použita funkce *softmax*

Kompilace modelu: Jako ztrátová funkce je použita funkce *categorical_crossentropy*, optimalizace pomocí *rmsprop*

Trénování modelu: Trénování probíhá v 6 epochách po 64 dávkách. Tyto hodnoty byly experimentálně upravovány pro dosažení co nejlepších výsledků. Pro trénovací data síť dosahuje úspěšnosti 89,74%.

Natrénovaný model je uložen jako „nn_voice.h5“

Predikce pro testovací data

Pro predikci dat, je nutné systémy nejprve natrénovat. Poté lze uložené modely nahrát ve skriptu *predict.py*.

Fáze predikce

Predikce obrázků: Pomocí funkce *predict_proba,* je pro jednotlivá evaluační data určena pravděpodobnost, zda je obličej shodný. Práh pro tvrdé rozhodování je nastaven na 1.0.

Výsledky pro jednotlivá data jsou uložena do souboru „predict_image_output.txt“.

Predikce hlasu: Pomocí funkce *predict_proba,* je pro jednotlivá evaluační data určena pravděpodobnost, zda je hlas shodný. Práh pro tvrdé rozhodování je nastaven jako aritmetický průměr všech pravděpodobností.

Výsledky pro jednotlivá data jsou uložena do souboru „predict_voice_output.txt“.

Celková predikce: Pravděpodobnost pro testovací data je určena jako $P = P(A) * P(B)$. Následně normalizována. Práh pro tvrdé rozhodování je nastaven na 0.5.

Výsledky pro jednotlivá data jsou uložena do souboru „predict_person_output.txt“.

Práce se skripty

Závislosti pro správný běh systémů: Numpy, Keras, ikrlib, python3.6, matplotlib

Vyvíjeno a testováno na Ubuntu18.04.4 LTS

Spouštění jednotlivě: Pro správné spuštění skriptů, je nutné nastavit cesty k datům. Toto nastavení se provádí v jednotlivých skriptech. Implicitně je zvolena cesta: „ ../data/(podle spouštěného skriptu)“.

Spouštění pomocí příkazu make:

make img: Provede natrénování sítě pro rozpoznávání obličeje.

make voice: Provede natrénování sítě pro rozpoznávání hlasu.

make predict: Provede predikci a zápis výsledků do souborů.

make all: Provede sekvenčně vše výše uvedené.

Zlepšení systémů

Pro lepší výsledky obou systémů by jistě nejvíce pomohlo mít k dispozici větší množství trénovacích dat. Bylo by také možné využít metody *rozšíření dat (data augmentation)*. Tato metoda využívá různé transformace, které postupně aplikuje na sadu dat a tím vytváří nová data. Další možností by bylo využití předtrénované sítě. Tyto sítě jsou trénovány na velkých množinách dat, tudíž většinou (ne nutně) dosahují lepších výsledků. Dále by šlo jistě lépe nastavit parametry jednotlivých sítí a jejich vrstev. Jelikož máme malou validační sadu dat, šlo by také použít *k-násobnou křížovou validaci (K-fold cross-validation)*, která by nám mohla pomoci lépe určit tyto parametry.

Literatura

Deep learning v jazyku Python Knihovny Keras, TensorFlow Chollet François, ISBN: 978-80-247-3100-1