Vysoké Učení Technické v Brně
Fakulta informačních technologií

Strojové učení a rozpoznávaní 2020

Dokumentácia k projektu

Detektor jedné osoby z obrázku obličeje a hlasové nahrávky

Marek Žiška **(xziska03)**
Martin Osvald **(xosval03)**

Brno, 30.04.2020

1 Audio systém

1.1 Popis systému

1.1.1 Spracovanie nahrávok

Na vytvorenie systému som použil deep learning knižnicu Keras. Na vyhodnocovanie som sa rozhodol použiť extrakciu príznakov z audio nahrávok. Na extrakciu príznakov som najprv potreboval vyfiltrovať audio nahrávky zo zložiek. Tieto audio nahrávky som načítaval pomocou funkcie *parse_audio_files* z jednej zložky *train* v ktorej boli zložky *non_target* a *target*. Jednotlivé cesty k nahrávkam som si uložil do zoznamu, z ktorého som následne mohol spracovávať jednotlivé nahrávky.

1.1.2 Extrakcia príznakov

Extrakcia prebieha vo funkcií *get_features*. V tejto časti som dosť experimentoval. Na jednoduchú extrakciu príznakov som použil knižnicu *librosa*. Pri návrhu môjho prvého modelu som z audio nahrávok extrahoval len MFCC príznaky, ktoré mapujú charakteristiky ľudského hlasu. Pri prvom modely som z týchto príznakov vytváral spektogramy, na ktorých som som trénoval konvolučnú neurónovú sieť. Túto metódu som nakoniec ale nepoužil z dôvodu nepresných výsledkov a použil som metódu, v ktorej som si vyextrahované príznaky, názov osoby a nahrávacieho sedenia uložil do csv súbora. Vo funkcií *get_set_of_data* som z csv súbora vytvoril dátové sety:
Train_data - numpy pole reprezentujúcé príznaky nahrávok pre dané osoby. Jednotlivé príznaky som s *StandardScaler* vyštandardizoval aby boli dáta viac menej normálovo distribuované a teda znížime možnosť neočakávane zlého natrénovania neúrónovej siete.
Train_labels - pole veľkosťou korešpondujúce *Train_data*, obsahujúce hodnoty 0 pre neznáme osoby a 1 pre hľadanú osobu.
Pri extrakcií a klasifikácií MFCC príznakov som sa ale nedostal k dobrým výsledkom, klasifikátor klasifikoval všetky nahrávky ako neznáme poprípade rozpoznal len zopár cieľových nahrávok. Rozhodol som sa skúsiť rozšíriť trénovací dataset. Rozmýšľal som nad transformáciami nahrávok alebo extrahovaním ďalších príznakov z audio nahrávok. Ako prvé som skúsil extrahovať ďalšie príznaky. Po ktorých pridaní som už dostal rozumnejšie výsledky a k transformáciam nahrávok som sa nedostal. Okrem MFCC som teda skúšal príznaky ako:
Spektrálny centroid - ukazuje na frekvenciu, okolo ktorej je sústredené najvačšie množstvo energie signálu.
Spektrálny rollof - Určuje, kde je uložených 90% energie spektra.

Spektrálny bandwidth - Ukazuje na rozloženie frekvencie.
Zero crossing rate - Počet prechodov signálu nulou.
Chroma príznak - Indikuje koľko energie je v každej triede tónu.
Postupne som tieto príznaky pridával/kombinoval a najväčšiu presnosť som dosiahol pri zahrnutí všetkých príznakov.

1.1.3 Návrh modelov neuronovej siete

Prvý návrh modelu pozostával z dvoch Dense vrstiev, vstupný tvar dát bol počet príznakov extrahovaných z náhravky, aktivačnú funkciu prvej vrstvy som zvolil *relu.* s tým že výstupnú dimenzionalitu som zvolil na 32, druhá vrstva používala aktivačnú funkciu *softmax*, ktorý sa typicky používa pre koncové vrstvy. Výstupná dimenzionalita bola 2, keďže klasifikátor rozdeloval dve triedy(target/non_target). Model som kompiloval s optimalizačnou funkciou *adam*, ktorý funguje na princípe stochastického poklesu gradientu a je nenáročný na výkon. Loss funkciu som zvolil *sparse_categorical_crossentropy*, ktorá je vhodná pre systémy s dvoma a viac výslednými triedami. Pri trénovaní modelu som sa viedol tým že počet epôch budem optimalizovať tak aby sedeli na práve taký počet, kedy sa presnosť modelu na trénovacích dátach blížil k jednej. Bolo to hlavne z dôvodu aby sa model nepretrénoval. Tento model avšak nepriniesol žiadne dobré výsledky, všetky nahrávky označoval za neznáme. Následne som skúšal zvyšovať počet skrytých vrstiev v návrhu modelu a to v takom zmysle aby počty výstupných dimenzionalít regresívne klesali. Taktiež som pri trénovaní zaviedol validačný set dát a všimol som si že, model už dávať presnejšie výsledky. Pri týchto obmenách som sa dopracoval až k výslednému modelu, z ktorého presnosťou som bol uspokojený.

1.1.4 Generovanie výsledkov

Pri generovaní výsledkov som použil funkcie:
model.predict_classes - ktorá vrátila tvrdé rozhodnutie 0 alebo 1 podľa veľkosti apriórnych pravdepodobností tried
mode.predict_proba - vráti tuple apriórnych pravdepodobností pre každú triedu. Z týchto pravdepodobností som generoval skóre. Skóre teda predstavovalo s akou pravdepodobnosťou si je systém istý tvrdým rozhodnutím.
A z týchto dát som vytvoril požadovaný finálny formát.

1.1.5 Čo by som zmenil

Pri trénovaní by som skúsil nahrávky transformovať a vygenerovať tak viacero nahrávok najmä pre hľadanú osobu, aby sa počty nahrávok pre jednotlivé triedy

mierne vyrovnali. Pri trénovaní by sa model natrénoval presnejšie, plus validačný set by dával väčší zmysel, keďže zastúpenie oboch tried by v ňom bolo väčšie nie ako to je v prípade odovzdaného systému, kde pri trénovaní prevažovala trieda *non_target*.

1.2 Zoznam použitých externých nástrojov

1.3 Návod na zreprodukovanie výsledkov.

Ako prvé je samozrejme potrebné nainštalovanie všetkých nástrojov, uvedených v README. Následne bude potrebné vytvoriť zložku *train*, v ktorej budú dve zložky *non_target* a *target*. V zložkách sú obrázky a nahrávky nehľadaných a hľadanej osoby, na týchto dátach bude systém natrénovaný. Zložku s vyhodnocovacími/nevidenými dátami. *eval*. Trénovanie neurónovej siete som vykonával v jupyter notebooku *audio_classifier_MLP.ipynb*. Keďže pri trénovaní neurónovej siete vznikajú rôzne váhy, som teda model na ktorom som vyhodnocoval evaluačné dáta uložil separátne, aby bolo možné totožné zreprodukovanie výsledkov(súbor *audio_classifier_MLP.py*).

2 Image systém