Biometrický identifikačný systém

Strojové učení a rozpoznávání

Fakulta Informačních technologií VUT v Brně
2020/2021
Anton Firc (xfirca00), Tomáš Lapšanský (xlapsa00)

1 Dataset

Všetky dáta ktoré systém používa pre tréning a vyhodnotenie sa nachádzajú v zložke `dataset`. Do tejto zložky je potreba nakopírovať zložky `eval`, `dev` a `train` z poskytnutého datasetu. Následne je ešte potreba vytvoriť zložku `train_big` ktorá vznikne zlúčením zložiek `dev` a `train` tak aby vzniklo najväčšie možné množstvo trénovacích dát. Túto zložku je možné vytvoriť spustením skriptu `dataset_big.py`, ktorý je umiestnený v koreňovej zložke projektu.

Obsah zložky `dataset` teda vyzerá nasledovne:

```
dataset
├── dev
│   └── speaker folders 1-31
├── eval
│   └── speaker folders 1-31
├── train
│   └── speaker folders 1-31
└── train_big
    └── speaker folders 1-31
```

2 Rozpoznávanie reči

Táto sekcia popisuje skúšané a zvolené prístupy k spracovaniu nahrávok a rozpoznávaniu podľa reči.

2.1 Predspracovanie reči

Prvým krokom spracovania datasetu bolo odstránenie ticha z nahrávok. Za použitia nástroja `sox` boli z nahrávok odstránené dlhšie segmenty ticha, ako na začiatku a konci, tak aj uprostred nahrávky.

Okrem odstránenia ticha sme trénovací dataset augumentovali. Pre tento účel bol znovu použitý nástroj `sox` pomocou ktorého sme každú nahrávku mierne spomalili a zrýchlili.

Experimentovali sme ešte s odstránením šumu v pozadí použitím nástroja `sox`, no takáto úprava datasetu nepriniesla zlepšenie v presnosti klasifikátorov, práve naopak.

2.2 Klasifikácia pomocou GMM

Vstupom klasifikátora modelujúceho reč použitím GMM sú nahrávky očistené od ticha a následne augumentované. Po extrakcii MFCC koeficientov je ku každému koeficientu prepočítaná delta, a výsledný 48-príznakový vektor použitý pre trénovanie a klasifikáciu.

V priebehu experimentov sa ukázal ako najvhodnejší počet gaussoviek v každom GMM ako 2. Pre každú osobu prebehne tréning jemu priradeného GMM.

Ohodnotenie modelu a predikcia na nevidených dátach používa rovnaký princíp, kde sa vyberie GMM s najvyššou log-pravdepodobnosťou a podľa jeho indexu sa určí výsledná trieda.

2.3 Klasifikácia pomocou NN

Modelovanie rozpoznávania podľa reči sme skúšali modelovať aj pomocou neurónovej siete.

Pre spracovanie nahrávok, je potreba nahrávky rozdeliť na rovnako dlhé segmenty, tak aby dáta vstupujúce do siete mali vždy rovnaký tvar. Z tohoto dôvodu sme všetky trénovacie dáta rozdelili na segmenty o dĺžke 1 sekundy.

Architektúru siete a pôvodné parametre sme prevzali z tutoriálu dostupného na: `https://keras.io/examples/audio/speaker_recognition_using_cnn/`.

Oproti pôvodnému návrhu používame inú aktivačnú funkciu (softmax), learning rate scheduler a optimizer SGD.

Tréning modelu prebiehal vo viacerých iteráciách, kde sme striedali pôvodné dáta a dáta s pridaným hlukom v pozadí. Vo výsledku sme pri tréningu dosiahli presnosť cez 95% a loss 0.2.

2.4 Zhodnotenie

Pre rozpoznávanie reči sme implementovali 2 systémy z ktorých sa oba ukázali ako funkčné. Najviac pozornosti a experimentov bolo venovaných príprave dát, od rôznych metód odstraňovania ticha, cez odstraňovanie šumu, až po augumentáciu dát.

3 Rozpoznávanie tváre

Táto sekcia popisuje skúšané a zvolené prístupy k rozpoznávaniu podľa tváre.

3.1 PCA a LDA analýza

Nad trénovacími dátami sme sa pokúsili vykonať redukciu dimenzionality pomocou PCA a následne LDA analýzy. Tento prístup sa však v našom prípade ukázal ako neefektívny. Pri klasifikovaní dát z *dev* setu sa nám podarilo dosiahnuť úspešnosť len v rádovo pár desiatkach percent. Tento jav mohol nastať pre niekoľko dôvodov, ako je napríklad nízke rozlíšenie obrázkov, nerovnomernosť natočenia jednotlivých tvárí,...

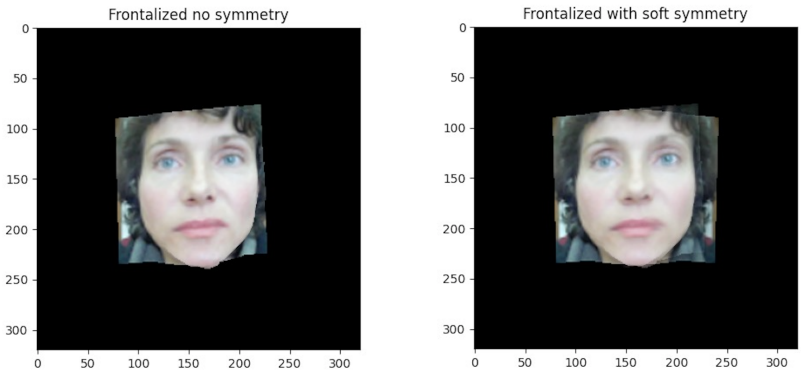
Pre vylepšenie výsledkov sme sa rozhodli do riešenia zakomponovať aj *NSGT (Non Stationary Gabor Transform)* pre lepšiu extrakciu hlavných rysov tváre, a taktiež využitie *GMM*, no pre slabú úspešnosť dosiahnutých výsledkov sme sa rozhodli toto riešenie nepoužiť.

3.2 Frontalizácia

Frontalizácia je procesom vytvárania syntetického zobrazenia tváre z prednej strany so zarovnaním na stred. Tento proces by mal zvýšiť presnosť biometrických identifikačným systémov. Ako implementáciu sme zvolili model dostupný v nasledujúcom repozitári: `https://github.com/dougsouza/face-frontalization` . Úprava vstupných dát pomocou frontalizácia však nepriniesla razantné výsledky. Na obr. 1 môžete vidieť príklad synteticky vytvorených dát. Dáta dosiahli po frontalizácia dvojnásobné rozmery v porovnaní s pôvodným datasetom (160 x 160px) a zároveň sú vyobrazené iba v strede snímku. Nie všetky dáta však boli v konzistentnom a použiteľnom stave, zároveň nástroj nedokázal spracovať všetky vstupné dáta, takže nepriniesli požadovaný prínos.

3.3 VGGFace2

Vrámci experimentovania s rozpoznávaním tvári sme sa rozhodli otestovať sériu modelov predtrénovaných na datasete *VGGFace2*. VGGFace2 je dataset obsahujúci 3.3 milióna fotografií známych osobností (9131 osobností). Keras obsahuje modul *keras_ vggface*,



(a) (b)

Obr. 1: Výsledky frontalizácie tváre. Obrázok naľavo ukazuje frontalizovanú tvár bez symetrie, obrázok napravo frontalizovanú tvár so symetriou.

kde existuje niekoľko predtrénovaných modelov na tomto datasete, ako napríklad *VGG16* a *Resnet50*. Po vložení požadovanej fotky do modelu je fotka pretransformovaná do vektoru o veľkosti 512 príznakov, ktorý reprezentuje charakteristiku tváre. Následne sme schopní porovnávať kosínusové vzdialenosti týchto vektorov a na ich základe rozhodnúť, či je na dvoch snímkoch rovnaká osoba.

Pri tomto prístupe sme narazili na niekoľko problémov. Pri tejto implementácií sa zväčša využíva threshold hodnota (v rádoch desatín), pod ktorú ak sa hodnota kosínusovej vzdialenosti dostane, môžeme osoby vyhlásiť za totožné, a však na dodanom datasete sme dosahovali hodnoty vzdialenosti v rádoch 10^{-3} a menej, takže boli všetky osoby vyhodnotené ako totožné. Rozhodli sme sa preto vybrať osobu, ktorej hodnota vzdialenosti je najmenšia.

Najlepšie vyhodnotenie sme dosiahli pri modeli *VGG16*, kde sme dosiahli presnosť 20.97%. Pri tomto modeli sme však ako pri jedinom mohli pracovať s neupravenými dátami, zatiaľ čo *Resnet50* a *SENet50* vyžadovali vyššie rozlíšenie dát, preto bolo potrebné vstupné dáta umelo zväčšiť na rozlíšenie 224x224px, čo mohlo taktiež spôsobiť nižšiu presnosť natrénovaných modelov. Resnet50 dosiahol presnosť 16.13% a SENet50 12.9%.

3.4 MTCNN

Pre zvýšenú presnosť rozpoznávania tvári sme sa pokúsili na pridelený dataset aplikovať techniku detekcie tváre, keďže v datasete existujú fotky tvárí, ktoré v značnej miere obsahujú pozadia s kanceláriami alebo inými obytnými zónami. Pri výbere modelu sme sa rozhodovali medzi niekoľkými variantami, ako je *MTCNN*, *Haar Cascade* a *Dlib* modely ako *CNN* a *HOG*. Z vybraných modelov sme zvolili *MTCNN*, keďže podľa dostupných informácií poskytuje najpresnejšiu extrakciu tváre.

Sieť MTCNN si však s prideleným datasetom nedokázala poradiť, výsledkom extrakcie tvári bola prázdna množina, pravdepodobne pre nízke rozlíšenie snímkov. *MTCNN* vyžaduje veľkosť tváre minimálne 80x80px, čo je v našom prípade veľkosť celého snímku, nie len tváre na ňom.

3.5 Modul face recognition

Ako posledný experiment sme zvolili modul `face-recognition`[1] pre Python. Tento modul obsahuje metódy pre extrakciu príznakov z tvárí a následne porovnávanie extrahovaných 120-príznakových vektorov pomocou euklidovskej vzdialenosti.

Modul dokáže porovnávať dve tváre medzi sebou, nakoľko však dataset obsahuje viac trénovacích dát rozhodli sme sa kombinovať trénovacie fotky do jednej "priemernej tváre"pre každú triedu. Toto priemerovanie je vykonané nad extrahovanými vektormi.

Následne predikcia trieda porovnáva vzdialenosť tváre od priemerných tvárí uložených pre každú triedu. Výsledkom porovnania je vzdialenosť od každej triedy, táto hodnota je následne invertovaná a pomocou softmax funkcie sú spočítané pravdepodobnosti pre každú triedu, index s najväčšou pravdepodobnosťou udáva predikovanú triedu.

Pri testovaní nad dev dátami sme dostali presnosť 100% už na prvý pokus, bez akéhokoľvek predspracovania vstupných obrázkov.

3.6 Zhodnotenie

Pre rozpoznávanie tváre sme implementovali dokopy 4 systémy, z ktorých dosiahol rozumné výsledky len jeden. Predpokladáme, že výborné dosiahnuté výsledky posledného implementovaného systému spočívajú v extrakcii príznakov priamo z rysov tváre a nie celého obrázku.

4 Multimodálny systém

Táto sekcia popisuje skúšané a zvolené prístupy k rozpoznávaniu podľa oboch modalít - reči a tváre.

4.1 Kombinované pravdepodobnosti

Pre kombináciu systémov sme sa rozhodli vyskúšať kombinovanie predpovedaných pravdepodobností. Kombinujeme výstupy implementované systému pre rozpoznávanie podľa tváre a systému pre rozpoznávanie reči pomocou GMM.

Pre každú nahrávku a fotku získame predpovedané log-pravdepodobnosti. Pomocou softmax funkcie ich prevedieme na pravdepodobnosti a tie medzi sebou vynásobíme. Z takto vynásobených pravdepodobností potom vyberieme index triedy s najvyššou pravdepodobnosťou.

Takto kombinovaný systém vykázal 100% úspešnosť na dev dátach.

4.2 Zhodnotenie

Implementovali sme jeden multimodálny systém ktorý sa ukázal, že funguje na dodaných dátach veľmi dobre. Počas jeho implementácie sme experimentovali s reprezentáciou výstupov z oboch klasifikátorov tak aby bolo možné ich čo najlepšie kombinovať a dosiahnuť vysokú presnosť.

[1] `https://pypi.org/project/face-recognition/`

5 Spustenie odovzdaných systémov

Odovzdané riešenie obsahuje implementáciu štyroch systémov: dva pre hlasovú identifikáciu, jeden pre identifikáciu pomocou tváre a jeden multimodálny systém. Pre každý z týchto systémov je v koreňovej zložke pripravený demo skript ktorý slúži pre reprodukciu odovzdaných výsledkov.

5.1 Požiadavky

Všetky skripty sú implementované v jazyku `Python 3.6`. Všetky potrebné Python balíky sú uvedené v súbore `requirements.txt`, jediná závislosť mimo tento súbor je nástroj `sox`, verzia 14.4 alebo vyššia.

5.2 Systém: Hlas - GMM

Detaily tohoto systému sú popísané v sekcii 2.2. Demo skript `demo_speech_gaussian.py` umiestnený v koreňovej zložke projektu obsluhuje prípravu dát z datasetu, tréning a vyhodnotenie GMM a následné vytvorenie súboru s výsledkami `speech_gaussian.txt` v koreňovej zložke projektu.

5.3 Systém: Hlas - Keras

Detaily tohoto systému sú popísané v sekcii 2.3. Demo skript `demo_speech_keras.py` umiestnený v koreňovej zložke projektu zavolá pripravený skript pre vyhodnotenie natrénovaného modelu neurónovej siete. Pre klasifikáciu dát sú potreba dva súbory, `model.h5` a `class_names.h5`. Prvý súbor obsahuje uložený model, druhý súbor mapuje neuróny výstupnej vrstvy na naučené triedy. Oba súbory sú je možné stiahnuť spustením skriptu `download_model.sh`. Nakoniec demo skript vytvorí súbor s výsledkami `speech_keras.txt` v koreňovej zložke projektu.

5.4 Systém: Tvár - FaceRecognition

Detaily tohoto systému sú popísané v sekcii 3.5. Demo skript `demo_photo_face_rec.py` umiestnený v koreňovej zložke projektu obsluhuje tréning a vyhodnotenie modelu a následne vytvorenie súboru s výsledkami `photo_norm-dist.txt` v koreňovej zložke projektu.

5.5 Multimodálny Systém: pravdepodobnosti

Detaily tohoto systému sú popísané v sekcii 4.1. Demo skript `demo_multimodal_probs.py` umiestnený v koreňovej zložke projektu obsluhuje prípravu dát a tréning systémov pre rozpoznávanie pomocou hlasu a tváre. Následne oba systémy vyhodnotí a nakoniec vytvorí súbor s výsledkami `multimodal_probabilities.txt` v koreňovej zložke projektu.