# SUR

VUT-FIT SUR projekt

xgalba03, xpomyk04

## Klasifikátor obličeje

Pro vytvoření modelu klasifikátoru obličeje jsme se rozhodli využít konvoluční neuronovu síť, která má následující strukturu.

1. konvoluční vrstava - 16 filtrů (3x3)
2. max pooling (2x2) a dropout (0.3)
3. konvoluční vrstava - 32 filtrů (3x3)
4. max pooling (2x2) a dropout (0.2)
5. plně propojená vrstva (Relu) - 256 neuronů
6. plně propojená vrstva (Relu) - 64 neuronů
7. plně propojená vrstva (softmax) - 2 neurony

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 78, 78, 16)        448

 max_pooling2d (MaxPooling2D  (None, 39, 39, 16)       0
 )

 dropout (Dropout)           (None, 39, 39, 16)        0

 conv2d_1 (Conv2D)           (None, 37, 37, 32)        4640

 max_pooling2d_1 (MaxPooling  (None, 18, 18, 32)       0
 2D)

 dropout_1 (Dropout)         (None, 18, 18, 32)        0

 flatten (Flatten)           (None, 10368)             0

 dense (Dense)               (None, 256)               2654464

 dense_1 (Dense)             (None, 64)                16448

 dense_2 (Dense)             (None, 2)                 130

=================================================================
Total params: 2,676,130
Trainable params: 2,676,130
```

Tento model je implementován a natrénován pomocí rozhraní keras, který pracuje nad frameworkem tensorflow (framework pro práci s n. sítěmi). Implementace modelu je ve zdrojovém souboru img_class_nn.py

## Předzpracování trénovacích dat

Data která dostupná pro trénování se skládají z:

1. 160 obrázků pro trénování (20 non-target a 140 non-target)
2. 30 obrázků pro validaci (20 non-target a 10 target)

Tyto data jsou postupně načteny pomocí knihovny cv2, převedeny a převedeny do RGB formátu. Při načítání datasetu je z každého obrázku syntetizováno pomocí transformací několik dalších (knihovna albumentations). Tímto dojde k rozšíření datasetu. až na 13140 trénovacích dat a 2670 validačních dat. Obrázky jsou uloženy jako pole (80x80x3) matic s hodnotami normalizovanými do rozsahu (0-1) pro rychlejší zpracování neuronovou sítí.

## Trénování

Neuronová síť je trénována pomocí batchů o velikosti 20 na 11 epoch. Dosažené kvality neuronové sítě na validačních datech je možno vidět na následujicím obrázku.

```
Epoch 1/11
657/657 [==============================] - 51s 77ms/step - loss: 0.1751 - accuracy: 0.9513 - val_loss: 0.2284 - val_accuracy: 0.9247
Epoch 2/11
657/657 [==============================] - 52s 80ms/step - loss: 0.0507 - accuracy: 0.9848 - val_loss: 0.2583 - val_accuracy: 0.9071
Epoch 3/11
657/657 [==============================] - 58s 88ms/step - loss: 0.0323 - accuracy: 0.9919 - val_loss: 0.6085 - val_accuracy: 0.9176
Epoch 4/11
657/657 [==============================] - 55s 84ms/step - loss: 0.0283 - accuracy: 0.9938 - val_loss: 0.5359 - val_accuracy: 0.9109
Epoch 5/11
657/657 [==============================] - 55s 84ms/step - loss: 0.0228 - accuracy: 0.9947 - val_loss: 0.5696 - val_accuracy: 0.9101
Epoch 6/11
657/657 [==============================] - 55s 84ms/step - loss: 0.0144 - accuracy: 0.9965 - val_loss: 0.4165 - val_accuracy: 0.9378
Epoch 7/11
657/657 [==============================] - 58s 89ms/step - loss: 0.0219 - accuracy: 0.9959 - val_loss: 0.4545 - val_accuracy: 0.9397
Epoch 8/11
657/657 [==============================] - 59s 89ms/step - loss: 0.0155 - accuracy: 0.9970 - val_loss: 0.8089 - val_accuracy: 0.9285
Epoch 9/11
657/657 [==============================] - 57s 87ms/step - loss: 0.0122 - accuracy: 0.9971 - val_loss: 0.5354 - val_accuracy: 0.9303
Epoch 10/11
657/657 [==============================] - 61s 92ms/step - loss: 0.0123 - accuracy: 0.9966 - val_loss: 0.6760 - val_accuracy: 0.8880
Epoch 11/11
657/657 [==============================] - 64s 97ms/step - loss: 0.0108 - accuracy: 0.9976 - val_loss: 0.2852 - val_accuracy: 0.9551
Training dataset size: 13140
Validation dataset size: 2670
```

### Klasifikace

Při testování na jednotlivých obrázích jsem zjistil že klasifikátor funguje celkem pěkně, ale někdy se "sekne" a označí jako target osobu non-target data s velkou pravděpodobností (>0.9). Bohužel se mi toto nepodařilo nijak vyřešit. Hard decision práh pro klasifikaci jsem experimentélně určil na 0.9 (pro target) tak že > 0.9 -> hard decision = 1

## Klasifikátor reči

Klasifikátor rečových dát vo formáte .wav bol implementovaný pomocou Gaussian Mixture Modelu (GMM) vyuýívajúceho EM algoritmus na jednotlivé ktorý výpočtu

Štruktúra zdrojového kódu: -načítanie vstpuných dát na trénovanie a klasifikáciu -vytvorenie MFCC koeficientov z dát a ich transponovanie a uloženie do dátovej štruktúry -určenie vstupných koeficientov GMM modelu (prebieha náhodne) -iterácia EM algoritmu pokiaľ dáta neskonvergujú -využitie natrénovaných koeficientov na odhadnutie/určenie klasifikácie testovacích dát -uloženie výsledkov do súboru .txt

### Klasifikace

```
Pri prehodnotení hodnôt vyplívajúcih z GMM boli hodnoty prevedené do intervalu 0-100. Kvôli veľkému rozsahu pôvodných hodnôt bol testovaním učený prah pozitívnej klasifikácie na hodnotu 0.98. (hard decision)
```

## Použití

Řešení produkuje celkem 3 result soubory, jeden pouze ze zpracování png, druhý ze zpracování wav a třetí z kombinace zpracování png a wav souboru.

1. image_CNN.txt - soubor s výsledky ze zpracování png souborů (spuštění skriptu img_class_nn.py). Skript očekává složku s trénovacími daty ve stejném adresáři. Pro spuštění skriptu je třeba nejprve doinstalovat všechny moduly, které potřebuje pomocí 'pip -install'

2. audio_GMM.txt - súbor s výsledkami hodnotenia audio nahrávok, vytvára sa pomocou spustenia skriptu audio_GMM.py a očakáva zložku testovacích dát s názvom 'eval' v rovnakom adresári ako zdrojový kód. Trénovacie dáta sa nachádzajú v zložke train_data/

3. results.txt - súbor obsahujúci výsledky z kombinovanej klasifikácie - audio + obrázok. Pre vyhodnotenie úspešnej klasifikácie musí násobok pravdepodobností dielčich skriptov dosahovať hodnotu 89% - 0.89