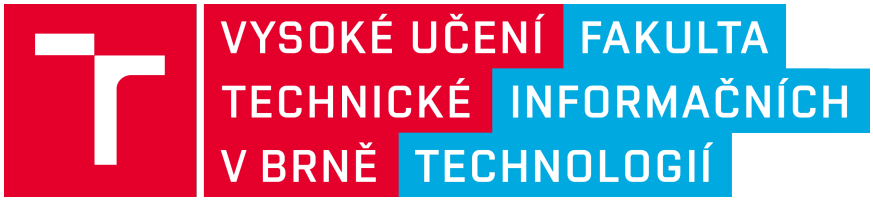

SUR – projekt 2024/2025

Klasifikace osob z obrázků a řeči

Vojtěch Orava (xorava02)

4. května 2025

1 Úvod

Tento dokument slouží jako dokumentace k projektu do předmětu Strojové učení a rozpoznávání (SUR) na FIT VUT v Brně. Jsou zde popsány implementované klasifikátory (1 obrázkový, 2 řečové) a všechny náležitosti k nim příslušející (jak kódy klasifikátorů spustit, kde najít výsledky, kam vložit vstupní data a jaké nástroje je potřeba nainstalovat). U každého klasifikačního systému je popsán způsob implementace a techniky či rozhodnutí, které byly vyzkoušeny nebo aplikovány za účelem zvýšení přesnosti klasifikace osob.

Data poskytnutá k trénování byla rozdělena do 2 složek:

- train – celkem 6 obrázků a audio nahrávek pro každou osobu,
- dev – celkem 2 obrázky a nahrávky pro každou osobu.

Řečové klasifikátory byly natrénovány nejprve pouze na datech ze složky **train** a soubory ze složky **dev** byly použity k validaci/testování. Poté bylo k trénovacím datům přidáno po 1 souboru pro každou osobu ze složky **dev** a byly natrénovány nové klasifikační modely (celkem tedy 7 trénovacích obrázků/nahrávek a 1 testovací obrázek/nahrávka). Z

takto různě natrénovaných modelů byl vybrán ten s vyšší přesností a byla jím provedena klasifikace na ostrých datech.

Dále jsou v dokumentaci popsány vytvořené klasifikační systémy.

2 Obrazový klasifikátor – CNN

Obrazový klasifikátor je vytvořen formou konvoluční neuronové sítě (CNN). Tato síť je složena z konvolučních, maxpooling, batch normalization, dropout a plně propojených vrstev. Implementace je napsána v souboru `image.ipynb` a využívá primárně framework **Tensorflow**.

Byly vyzkoušeny různé architektury s různým počtem konvolučních a plně propojených vrstev, podobně bylo postupováno s přidáváním a odebíráním normalizačních a dropout vrstev. Nakonec se jako model s nejvyšší přesností ukázal ten, jenž je odevzdán (4 konvoluční vrstvy následované batch normalizací a dropoutem s pravděpodobností 0,3 a dvě plně propojené vrstvy). Velikost batch byla nastavena na hodnotu 16. Aplikace normalizačních a droupout vrstev se pozitivně projevila v tom, že se model nepřeučil na trénovacích datech ani po 500 epochách. Bylo experimentováno s hodnotami pravděpodobnosti pro dropout vrstvy a nejlepších výsledků bylo dosaženo s hodnotou 0,3.

Na trénovací obrázky je před trénováním aplikována augmentace, aby se dosáhlo vyšší generalizace modelu. Experimentálně byly vybrány tyto augmentační nastavení:

- náhodné převrácení přes osu Y,
- náhodná rotace (pravděpodobnost 0,25),
- náhodné přiblížení (pravděpodobnost 0,25),
- náhodný kontrast (pravděpodobnost 0,25),
- náhodná úprava jasu (pravděpodobnost 0,25).

Při trénování je ukládán nejlepší model (nejvyšší validační přesnost) pomocí Tensorflow callback objektu. Tento model je pak použit k predikci na ostrých datech.

Neuronová síť není úplně nejvhodnější řešení pro klasifikaci s tak malým počtem trénovacích obrázků. Pokud by byl ke klasifikaci použit model založený na Support Vector Machine (SVM), pravděpodobně by bylo dosaženo vyšší přesnosti klasifikace.

3 Řečový klasifikátor – Logistická regrese

Klasifikátor je implementován ve skriptu `audio_LR.ipynb`. V programu jsou nejprve načteny soubory ze složek **train** a **dev** a pomocí funkce `wav16khz2mfcc` z knihovny **ikrlib** jsou vypočteny hodnoty MFCC. Každý audio signál je také knihovnou **librosa** převeden do energetické formy a jsou z něj odstraněny segmenty s nízkou energií (ticho). Jako práh pro rozlišení mezi užitečným a nepotřebným signálem byla stanovena hodnota 20 %

maximální energie. Trénovací data jsou před spuštěním trénování promíchána (shuffle). Jako model je použit `LogisticRegression` z knihovny **sklearn**. Byly vyzkoušeny různé parametry **solver** a nakonec byly v kódu ponechány solvery „newton-cg“ a „lbfgs“. Modely dosáhly přesnosti cca 70 % na testovacích datech, bez ohledu na použití standardní nebo rozšířené trénovací množiny.

4 Řečový klasifikátor – GMM

Pro dosažení vyšší přesnosti byl implementován i klasifikátor založený na směsi Gaussovských rozložení (GMM). Tento klasifikátor se nachází v souboru `audio_GMM.ipynb`. Využívá `GaussianMixture` z knihovy **sklearn**. Signál je opět předzpracován stejným způsobem jako u modelu výše. Byly vyzkoušeny různé parametry GMM:

- počet Gaussovských komponent: 4, 8, 16, 32, 64,
- maximální počet iterací EM algoritmu (mezi 100 až 2000),
- metoda pro startovní nastavení parametrů (kmeans, k-means++),
- různé hodnoty počátečního stavu (**random_state**),
- různé typy kovariance (diag, full).

Nejlépe se nakonec projevily parametry, které má model nastaveny v odevzdaném skriptu (16 komponent, kmeans a 1000 iterací EM, „full“ kovariance). Velmi záleží jak jsou inicializovány parametry jednotlivých GMM, protože se podařilo dosáhnout se stejným nastavením různě vysokých přesností od 70 % výše. Nejvyšší přesnost které bylo dosaženo na testovacím datasetu je 91 %.

Byla vyzkoušena i implementace neuronových sítí s využitím LSTM a GRU jednotek, ovšem s tak omezeným trénovacím datasetem se přesnost klasifikace nedostala přes 50 %.

5 Technické náležitosti

Klasifikátory jsou implementovány v jazyce Python a odevzdány ve formě **Jupyter notebooků**. Pro správné fungování projektu je potřeba mít nainstalovaný framework **Tensorflow 2** a knihovny **numpy, matplotlib, librosa, sklearn** (všechny požadavky viz soubor *requirements.txt*). Řešení rovněž používá zdrojové kódy z poskytnuté knihovny **ikrlib**. Kód knihovny musel být lehce upraven (přetypování na int v některých funkcích), tak aby fungoval s Pythonem 3.10.12 a vyšším. Proto je do odevzdávaného archivu zahrnuta i tato knihovna.

Jupyter notebooky je možné spustit klasickým způsobem, klasifikátory očekávají trénovací a validační/testovací data ve složkách **train** a **dev** ve stejném adresáři. Ostrá data je nutno umístit do složky s názvem **main_data** (bez dalších podadresářů). Výsledky v

předepsaném formátu jsou zapsány do souborů s příponou **.txt**. Pro obrázkový klasifikátor se jedná o soubor `image_CNN.txt`, pro řečové pak `audio_GMM.txt` a `audio_LR.txt`, dle použitého přístupu.

K jednotlivým buňkám v rámci notebooků jsou přiloženy komentáře, takže by neměl být problém se v kódu zorientovat.