SUR projekt 2025

Radek Zobaník (xzoban02)

Michal Zobaník (xzoban01)

Zvuk

Pro klasifikaci řečníků ze zvukových nahrávek byl zvolen model na bázi GMM. Pro každého mluvčího byl natrénován jeden samostatný GMM model. Pro každý klasifikovaný vstup je vypočítána hodnota log-likelihood, která říká, s jakou pravděpodobností pochází daný hlas z této Gaussovy, tedy s jakou pravděpodobností patří tomuto mluvčímu. Pro klasifikaci se pak vybere mluvčí s nejvyšší log-likelihood pravděpodobností.

Nejprve však bylo nutné zpracovat hlasové nahrávky tak, aby jsme je co nejlépe mohli využít pro učení a následnou klasifikaci. Byla k tomu využita knihovna *librosa,* pomocí které jsme extrahovali mfcc příznaky. Po sérii testů jsme zvolili 30 extrahovaných příznaků. Pro lepší výsledky jsme též vypočítali první a druhou derivaci zmíněných příznaků, které zachycují změnu příznaků v čase, respektive míru jejich zrychlení.

Tyto příznaky jsou poté použity pro trénování parametrů GMM modelu, vždy pro daného mluvčího. U GMM bylo nutné najít optimální parametry pro naše data. Bylo zvoleno 24 gaussovkých komponent, které dávaly nejlepší výsledek na validačních datech. Též bylo nutné model trochu stabilizovat, kdy pro některé řečníky byl problém natrénovat jejich GMM kvůli nedostatku informací. Toto řídí parametr *reg_covar=1e-2,* který přičítá malou hodnotu k diagonále kovarianční matice. Díky všem nastavením dosahuje model přesnosti 79% na validační sadě.

Pro lepší výsledky by bylo nutné získat více dat a buď natrénovat tento model s vícero daty, anebo již použít nějakou neuronovou síť, která těch trénovacích dat potřebuje mnohem více.

Obraz

Pro klasifikaci pomocí obrázků obličejů byla vybrána konvoluční neuronová síť. K její implementaci byla použita knihovna *Keras.* Jako základní architektura knn byla vybrána

architektura sítě AlexNet. Síť AlexNet používá jako vstup obrázek, který má mnohem větší rozměry než ty, které jsou k dispozici pro trénování. Parametry sítě proto byly upraveny pro tyto menší rozměry. Při použití tohoto modelu přesnost na evaluační sadě dosahovala kolem 25 %. Přesnost na sadě trénovací ale byla 100%. Model byl příliš velký pro dodaný počet dat a špatně generalizoval.

Rozhodli jsme se že místo zmenšení modelu získáme větší množství dat. Na dodané trénovací obrázky byla použita augmentace pro vytvoření obrázků nových, ale podobných. Při použití stejného modelu jako předtím a augmentace došlo ke zvýšení presnosti přibližně na 45% a nedocházelo k přetrénování sítě.

K zpřesnění klasifikace byla použita knihovna *Air Tune*, která slouží k nalezení hyperparametrů, pro které bude mít síť nejlepší výsledky. Pomocí této knihovny jsme zkoušeli různé počty filtrů pro jednotlivé konvoluční vrstvy nebo plně propojené vrstvy, velikost učící konstanty nebo vynechání některých vrstev. Takto získaný model dosahoval přesnosti kolem 60 %. Parametry tohoto modelu je možné vidět v souboru *SRC/image_cnn.py* v proměnné *model_params*. Úspěšnost by se dala ještě zvetšit vyzkoušením jiných architektur sítí nebo vyzkoušením většího množství kombinací parametrů.

Spuštění

Pro spuštění je potřeba mít nainstalovaný Python 3.x. Dále je nutné nainstalovat potřebné knihovny například pomocí příkazu: *pip install -r requirements*. Program se potom spouští z root adresáře s dokumentací, např. jako: *python 3 SRC/sur_classification.py.* Program je možné spustit s několika argumenty:

- **-m {a, i, b}** - Výběr, které modely budou použité (a - zvuk, i - obrázky, b - modely pro zvuk i obrázky, Výchozí: b
- **--audio_model_location *AUDIO_MODEL_LOCATION*** - *AUDIO_MODEL_LOCATION* je umístění souboru s modelem pro zvuk, Výchozí: voice_gmm.pkl
- **--image_model_location *IMAGE_MODEL_LOCATION*** - *IMAGE_MODEL_LOCATION* je umístění souboru s modelem pro obrázky, Výchozí: image_cnn_model.keras
- **--train_data *TRAIN_DATA*** - *TRAIN_DATA* je umístění složky s trénovacími daty pro trénování modelu, která obsahuje podadresáře s daty pro jednotlivé třídy, Výchozí: dataset/train
- **--eval_data *EVAL_DATA*** - *EVAL_DATA* je umístění složky s validačními daty používaných při trénování, která obsahuje podadresáře s daty pro jednotlivé třídy, Výchozí: dataset/dev
- **-c *CLASSIFICATION*** - *CLASSIFICATION* je umístění složky se soubory pro klasifikaci
- **-t** - Provede se trénování vybraných modelů

Všechny argumenty programu jsou volitelné. V případě že nebude použit *-t* ani *-c* program neudělá nic, při použití obou budou nejprve vybrané modely natrénovány a potom pomocí nich provedena klasifikace.

V souboru *SRC/image_cnn.py* je dále možné změnit hodnotu proměnné *param_tune* na začátku souboru. Při použití *-t* a nastavení hodnoty *param_tune* na *True* dojde místo trénování modelu k hledání nejlepších parametrů modelu.

Reprodukce výsledků

Pro získání odevzdaných výsledků je možné použít přiložený makefile. Pro jeho použití je potřeba rozbalit archiv s daty pro klasifikaci do složky s dokumentací, tak aby zde vznikla složka eval. Potom stačí zavolat příkaz *make run.* Soubory s výsledky *audio_gmm* a *image_cnn* se po skončení programu vytvoří ve složce s dokumentací.