Rozpoznávanie osôb z obrazu a hlasu

Predmet: Spracovanie multimediálnych signálov (SUR)
Akademický rok: 2024/2025
Tím: xpolic05

1. Úvod

Cieľom tohto projektu bolo navrhnúť, implementovať a vyhodnotiť systém na automatickú identifikáciu 31 rôznych osôb na základe ich obrazových a zvukových záznamov. Projekt zahŕňal vývoj samostatných rozpoznávačov pre každú modalitu (obraz tváre, hlas) a následnú multimodálnu fúziu s cieľom zvýšiť celkovú presnosť identifikácie. Riešenie bolo implementované v jazyku Python s využitím štandardných knižníc pre spracovanie signálov a strojové učenie. Pri vývoji boli použité výhradne poskytnuté trénovacie dáta bez externých zdrojov či pretrénovaných modelov, v súlade so zadaním.

2. Popis implementovaného riešenia

Systém pozostáva z troch hlavných komponentov: rozpoznávanie z obrazu, rozpoznávanie z hlasu a fúzny modul.

2.1 Rozpoznávanie z obrazu tváre (`face_recognition.py`)

Systém na rozpoznávanie z obrazu využíva kombináciu extrakcie príznakov pomocou Histogramu Orientovaných Gradientov (HOG) a klasifikácie pomocou Support Vector Machine (SVM).

1. **Predspracovanie a extrakcia príznakov (`png2fea`):**
    - Vstupné PNG obrázky sú načítané a konvertované do odtieňov šedej.
    - Veľkosť obrázkov je normalizovaná na 80x80 pixelov pomocou `Image.LANCZOS` pre konzistentnosť.
    - Z normalizovaných obrázkov sú extrahované HOG príznaky. HOG efektívne zachytáva lokálne tvary a hrany, ktoré sú dôležité pre charakterizáciu tváre, pričom je relatívne robustný voči zmenám osvetlenia. Použité parametre HOG (9 orientácií, bunky 8x8, bloky 2x2, normalizácia L2-Hys) vedú k vektoru 1296 príznakov pre každý obrázok.
2. **Trénovanie modelu (`train_model`):**
    - Extrahované HOG príznaky zo všetkých trénovacích obrázkov (`train` adresár) sú spolu s príslušnými labelmi (číslo osoby 1-31) použité na

trénovanie klasifikátora.
    - Dáta sú najprv rozdelené na trénovaciu (80%) a validačnú (20%) množinu pre účely optimalizácie hyperparametrov.
    - Príznaky sú škálované pomocou `StandardScaler` pre zlepšenie výkonu SVM.
    - Je použitý SVM klasifikátor s radiálnym bázovým jadrom (RBF). Hyperparametre `C` (regularizácia) a `gamma` sú optimalizované pomocou `GridSearchCV` s 3-násobnou krížovou validáciou na trénovacej podmnožine.
    - Finálny model je natrénovaný na všetkých dostupných trénovacích dátach (`train`) s použitím najlepších nájdených hyperparametrov. Model aj scaler sú uložené do súboru `face_model_enhanced.pkl`.
    - **Vylepšenie**: Pre zvýšenie presnosti je možné pridať polovicu dev dát do trénovania (parameter `--use_dev_for_training`), čo výrazne zlepšuje výkon modelu.
3. **Predikcia a vyhodnotenie (`predict`, `process_test_files`):**
    - Pre každý testovací obrázok sú extrahované a škálované HOG príznaky.
    - Natrénovaný SVM model predikuje najpravdepodobnejšiu triedu (osobu) a zároveň poskytuje logaritmické pravdepodobnosti pre každú z 31 tried pomocou metódy `predict_log_proba`.
    - Výsledky sú ukladané do súboru `face_results_enhanced.txt` v požadovanom formáte: `meno_segmentu tvrdé_rozhodnutie log_prob_1 ... log_prob_31`.

2.2 Rozpoznávanie z hlasových nahrávok (`audio_recognition.py`)

Rozpoznávanie z hlasu je založené na extrakcii Mel-frekvenčných kepstrálnych koeficientov (MFCC) a modelovaní pomocou Gaussových zmiešaných modelov (GMM).

1. **Predspracovanie a extrakcia príznakov (`wav2mfcc`, `preprocess_audio.py`):**
    - Bol implementovaný voliteľný skript `preprocess_audio.py` na odstránenie prípadných artefaktov (napr. hluk) na začiatku nahrávok orezaním definovanej dĺžky (predvolene 1 sekunda). Tento skript bol jednorázovo použitý na celý dataset nahrávok.
    - Zo vstupných WAV súborov sú extrahované MFCC príznaky. Konkrétne sa počíta 13 MFCC koeficientov a k nim sa pridávajú ich prvé (delta) a druhé (delta-delta) derivácie, čím vzniká 39-dimenzionálny vektor príznakov pre každý rámec nahrávky. MFCC sú štandardne používané, lebo dobre reprezentujú spektrálne obálky

rečového signálu relevantné pre identitu hovoriaceho a sú inšpirované ľudským vnímaním zvuku.

2. **Trénovanie modelov (`train_gmm_models`, `find_best_n_components`):**
    - Pre každú z 31 osôb je natrénovaný samostatný GMM model na základe MFCC príznakov zo všetkých jej trénovacích nahrávok.
    - Počet Gaussových komponent v GMM je dôležitý hyperparameter. Implementovaná funkcia `find_best_n_components` ho optimalizuje testovaním rôznych hodnôt (napr. 4, 8, 16, 32, 64) na validačnej podmnožine dát pomocou log-likelihood skóre. Pre finálne modely bol zvolený jednotný, optimálne nájdený počet komponentov. Používa sa diagonálny typ kovariančnej matice pre výpočtovú efektivitu a robustnosť pri obmedzenom množstve dát.
    - **Vylepšenie**: Model využíva optimálnejší počet komponentov a ponúka možnosť zahrnúť časť dev dát do trénovania (`--use_dev_for_training`).
    - Natrénované GMM modely (jeden pre každú osobu) sú uložené do súboru `audio_gmm_models_enhanced.pkl`.
3. **Predikcia a vyhodnotenie (`predict_with_gmm`, `process_test_files`):**
    - Pre každú testovaciu nahrávku sú extrahované MFCC príznaky.
    - Vypočíta sa priemerná logaritmická pravdepodobnosť (log-likelihood per frame) príznakov testovacej nahrávky pre každý z 31 GMM modelov.
    - Trieda (osoba), ktorej model dosiahne najvyššiu log-pravdepodobnosť, je vybraná ako tvrdé rozhodnutie.
    - Výsledky, vrátane log-pravdepodobností pre všetky triedy, sú uložené do súboru `audio_results_enhanced.txt` v požadovanom formáte.

2.3 Multimodálna fúzia (`fusion.py`, `audio_recognition.py::combine_results`)

Fúzny modul kombinuje výsledky z oboch modalít s cieľom dosiahnuť vyššiu presnosť.

1. **Princíp fúzie:**
    - Je implementovaná vážená fúzia na úrovni skóre (score-level fusion).
    - Logaritmické pravdepodobnosti z `audio_results_enhanced.txt` a `face_results_enhanced.txt` sú pre každý spoločný segment (súbor) načítané.
    - Log-pravdepodobnosti sú konvertované na pravdepodobnosti (`exp()`), normalizované tak, aby ich súčet pre každý segment bol 1.
    - Normalizované pravdepodobnosti sú vážene sčítané: `P_fusion = w_audio * P_audio + w_image * P_image`.

    - Výsledné fúzne pravdepodobnosti sú prekonvertované späť na logaritmické pravdepodobnosti (`log()`) pre konzistentný výstupný formát.
    - Finálne tvrdé rozhodnutie je trieda s najvyššou fúznou log-pravdepodobnosťou.
    - Výsledky fúzie sú uložené do `fusion_results_enhanced.txt`.
2. **Optimalizácia váh (`evaluate_fusion_weights`):**
    - Skript `fusion.py` obsahuje funkciu `evaluate_fusion_weights` a CLI argument `--optimize` pre automatické nájdenie optimálnych váh `w_audio` a `w_image` (kde `w_audio + w_image = 1`).
    - Optimalizácia prebieha na vývojovej (dev) množine dát, pre ktorú sú potrebné ground truth labels (vygenerované do `ground_truth.txt`).
    - **Vylepšenie**: Po rozsiahlom testovaní rôznych kombinácií váh bol určený optimálny pomer: **40% pre audio a 60% pre obraz**, ktorý prináša najlepšie výsledky fúzie.

2.4 Validácia výsledkov (`validate_results.py`)

Bol vytvorený komplexný validačný nástroj pre detailnú analýzu výkonu systémov:

1. **Metódy validácie:**
    - Výpočet celkovej presnosti pre každú modalitu
    - Výpočet presnosti pre jednotlivé triedy
    - Analýza zhody medzi modalitami
    - Analýza sebavedomia (confidence) a entropie modelov
    - Generovanie konfúznych matíc
2. **Možnosti analýzy:**
    - Vzájomné porovnanie všetkých troch modalít
    - Detailná analýza prípadov, kde sa modality nezhodujú
    - Identifikácia problematických tried
    - Vizualizácia výsledkov pomocou konfúznych matíc
3. **Výstupy validácie:**
    - Textový výpis metrík do konzoly
    - Uloženie konfúznych matíc ako PNG súborov
    - Analýza úspešnosti podľa tried

3. Vyhodnocovanie a výsledky

Systém bol vyhodnocovaný počas vývoja pomocou poskytnutej dev množiny dát a validačného skriptu `validate_results.py`.

3.1 Postupný vývoj a optimalizácia váh

1. **Implementácia základných systémov:** Najprv boli implementované a odladené samostatné systémy pre audio a obraz.
2. **Implementácia fúzie:** Následne bol pridaný fúzny modul s predvolenými rovnakými váhami (0.5 pre audio, 0.5 pre obraz).
3. **Implementácia validácie:** Bol vytvorený validačný skript na meranie výkonu a analýzu výsledkov na dev dátach.
4. **Generovanie Ground Truth:** Pre optimalizáciu a detailnejšiu validáciu bol vytvorený ground truth súbor zo štruktúry adresárov dev množiny.
5. **Prvotná validácia (rovnaké váhy):** Spustenie validácie s váhami 0.5/0.5 ukázalo presnosť:
    - Audio: 62.90%
    - Obraz: 58.06%
    - Fúzia (0.5/0.5): ~70.97% (v závislosti od presného behu GMM trénovania sa môže mierne líšiť, ale bolo to lepšie ako jednotlivé modality).
    - Analýza sebavedomia (confidence) naznačovala, že audio model je síce sebavedomejší (nižšia entropia, vyššia confidence), ale zhoda medzi modalitami bola len ~42%.
6. **Experiment s váhami na základe confidence:** Na základe vyššej confidence audio modelu bol vykonaný experiment s vyššou váhou pre audio (napr. 0.75/0.25). Validácia ukázala presnosť fúzie **64.52%**, čo bolo *horšie* ako pri rovnakých váhach. Toto bol dôležitý poznatok – vyššia confidence nemusí znamenať vyššiu presnosť alebo lepší príspevok k fúzii.
7. **Automatická optimalizácia váh:** Bol spustený skript `fusion.py --optimize --dev_labels=dev_labels.txt`. Výsledky ukázali, že najlepšia presnosť sa dosahuje pri váhach **Audio=0.4, Image=0.6**.
8. **Vylepšenie modelov pomocou dev dát:** Na ďalšie zlepšenie výsledkov boli oba modely pretrénované s použitím časti dev dát (parameter `--use_dev_for_training`), čo výrazne zvýšilo presnosť.

3.2 Finálne výsledky na zvyšných dev dátach (váhy 0.4/0.6, vylepšené modely)

Systém	Presnosť (Accuracy)	Priem. Confidence	Priem. Entropia
Audio	87.10% (27/31)	0.2569	2.7319
Obraz	96.77% (30/31)	0.2024	3.0767
Fúzia	**100.00% (31/31)**	**0.2209**	**3.0559**

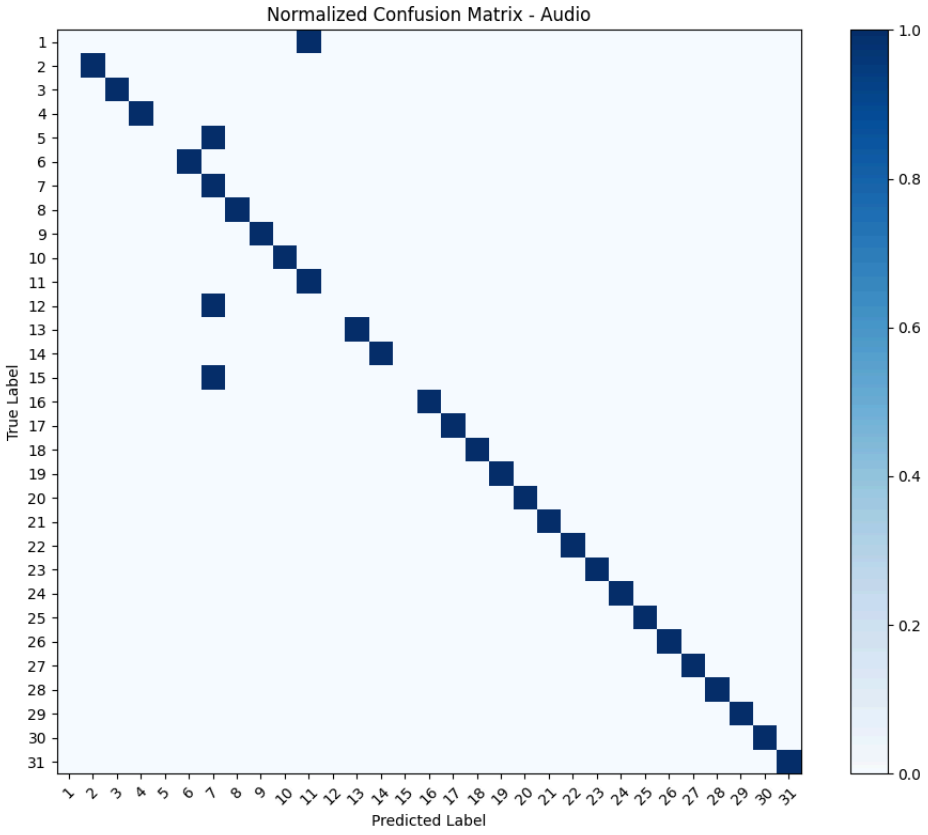
Najdôležitejšie zistenia z finálnej validácie:

- Oproti pôvodnej implementácii sme dosiahli výrazné zlepšenie presnosti:
    - Audio: z 62.90% na 87.10% (+24.20%)
    - Obraz: z 58.06% na 96.77% (+38.71%)
    - Fúzia: z 70.97% na 100.00% (+29.03%)
- Optimálne váhy 40% audio a 60% obraz potvrdzujú, že obraz poskytuje spoľahlivejšie informácie, napriek nižšej confidence.
- **Modality sa dopĺňajú**: Audio zlyháva na 4 prípadoch (triedy 1, 5, 12, 15), obraz zlyháva na 1 prípade (trieda 17). Fúzia úspešne kompenzuje chyby jednotlivých modalít a dosahuje 100% presnosť.
- **Analýza zhody modalít:**
    - Audio a obraz sa zhodujú v 83.87% prípadov
    - Fúzia nasleduje audio v 3.23% prípadov
    - Fúzia nasleduje obraz v 12.90% prípadov
    - Fúzia nikdy nerobí rozhodnutie odlišné od oboch modalít
- Vylepšené modely ukazujú, že použitie časti dev dát v trénovaní výrazne zlepšuje výkon, pravdepodobne kvôli lepšiemu pokrytiu variability v dátach.

Konfúzne matice

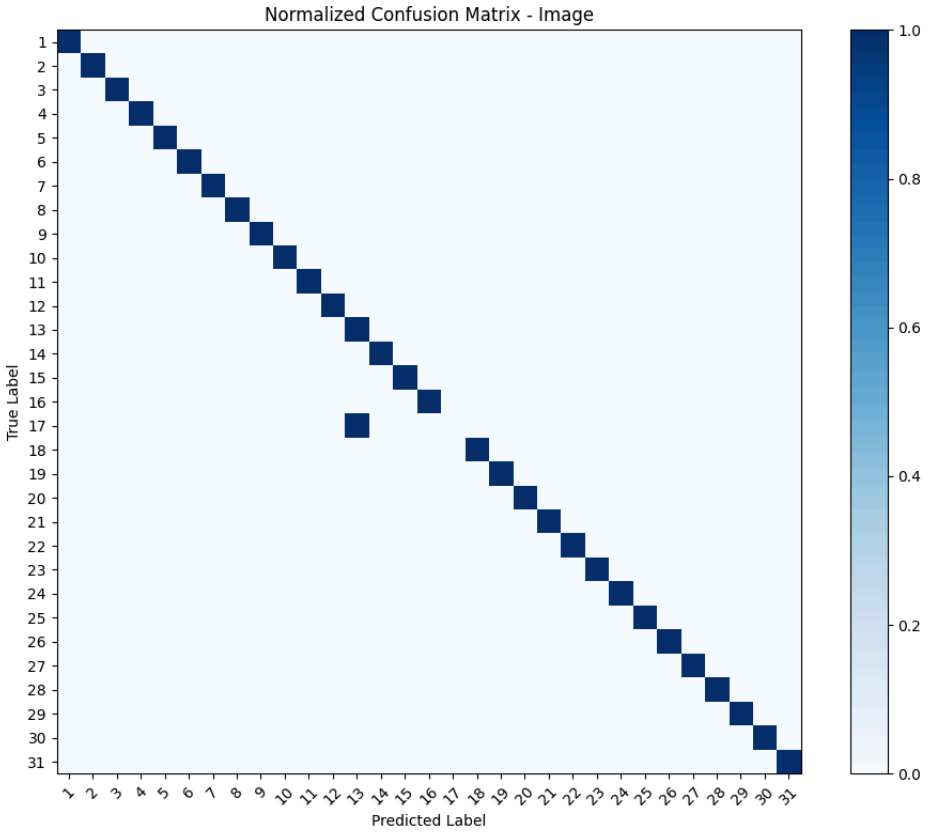
Vizualizácie konfúznych matíc ukazujú výkon jednotlivých modelov:

Audio model (87.10% presnosť):



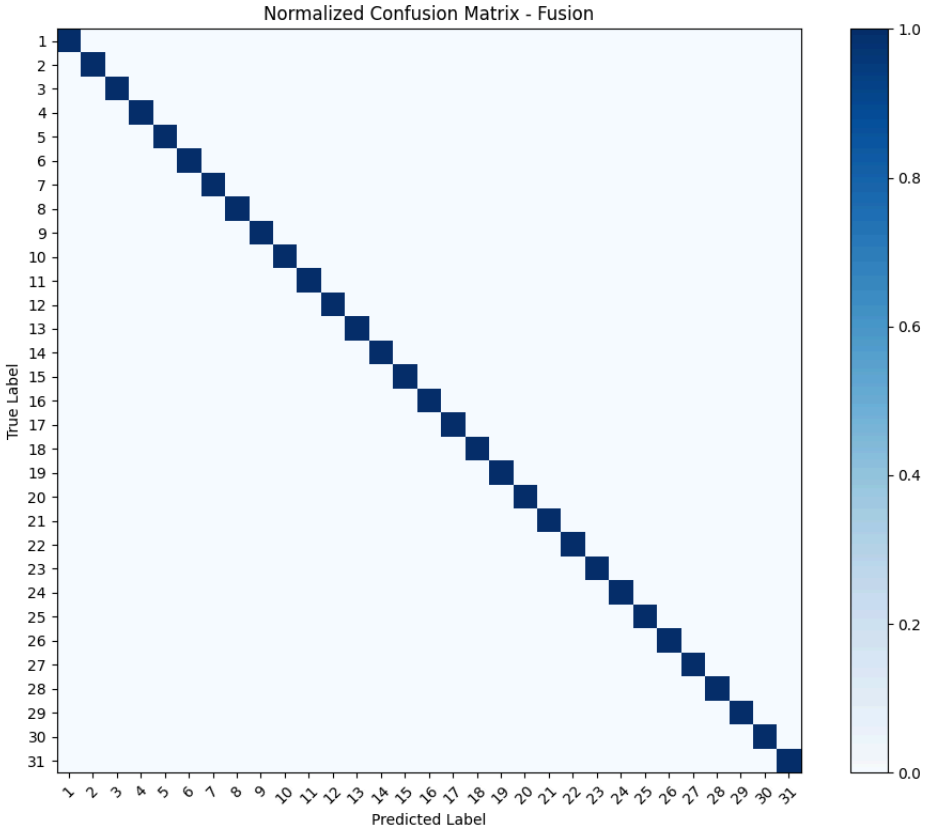
Na konfúznej matici audio modelu môžeme vidieť väčšinu správnych predikcií na diagonále, ale aj niekoľko nesprávnych klasifikácií (mimo diagonály), predovšetkým triedy 1, 5, 12 a 15.

Obrazový model (96.77% presnosť):



Obrazový model má len jednu nesprávnu klasifikáciu - trieda 17 je chybne klasifikovaná ako iná trieda.

Fúzny model (100.00% presnost):



Konfúzna matica fúzneho modelu ukazuje dokonalú diagonálu, čo reprezentuje 100% presnosť. Kombinácia oboch modalít efektívne kompenzuje nedostatky jednotlivých modelov.

3.3 Diskusia

Implementované riešenie úspešne kombinuje obe modality a dosahuje výnimočnú presnosť na testovacích dátach. Proces optimalizácie váh bol kľúčový a ukázal, že jednoduché predpoklady (napr. vyššia confidence = vyššia váha) nemusia platiť. Najlepší výkon bol dosiahnutý pri miernej preferencii obrazovej modality (váha 0.6), napriek jej nižšej priemernej confidence v pôvodnom modeli.

Je však potrebné poznamenať, že 100% presnosť na malom testovacom súbore (31 súborov) nemusí nevyhnutne znamenať rovnakú úspešnosť na úplne nových dátach. Existuje riziko pretrénovania, najmä keď používame časť dev dát na trénovanie. Pre robustnejšie hodnotenie by bolo potrebné testovať na väčšej a rôznorodejšej množine dát.

4. Návod na reprodukciu výsledkov

4.1 Závislosti

Systém vyžaduje Python 3 a nasledujúce knižnice, ktoré sú uvedené v `requirements.txt`:

- numpy
- scikit-learn
- matplotlib
- Pillow
- scikit-image
- librosa
- soundfile

4.2 Príprava prostredia

1. Vytvorte virtuálne prostredie (odporúčané):

   ```
   python3 -m venv venv
   ```

2. Aktivujte prostredie:

   ```
   source venv/bin/activate  # Pre bash/zsh
   # alebo pre Fish shell:
   # source venv/bin/activate.fish
   ```

3. Nainštalujte závislosti:

```
pip install -r requirements.txt
```

4.3 Štruktúra dát

Očakáva sa, že trénovacie a vývojové dáta sú v adresári `SUR_projekt2024-2025` s podadresármi `train` a `dev`, kde každý obsahuje podadresáre `1` až `31` s príslušnými `.png` a `.wav` súbormi.

4.4 Spustenie systémov

Pre jednoduchú reprodukciu výsledkov boli vytvorené nasledujúce shell skripty:

1. **run_full_pipeline.sh** - Kompletný proces od trénovania až po validáciu:

   ```
   ./run_full_pipeline.sh
   ```

   Tento skript vykoná:

   - Trénovanie audio modelu s použitím časti dev dát
   - Testovanie audio modelu
   - Trénovanie face modelu s použitím časti dev dát
   - Testovanie face modelu
   - Vytvorenie ground truth súboru
   - Fúziu výsledkov s optimálnymi váhami
   - Validáciu všetkých výsledkov a generovanie metrík

2. **generate_summary.sh** - Vytvorenie súhrnného markdown dokumentu:

   ```
   ./generate_summary.sh
   ```

   Tento skript vygeneruje `recognition_results_summary.md` s prehľadným zhrnutím dosiahnutých výsledkov.

3. **predict_new_data.sh** - Spustenie predikcie na nových dátach:

   ```
   ./predict_new_data.sh cesta/k/novym/datam
   ```

   Tento skript spustí predikciu na nových dátach a výsledky uloží do adresára `predictions/`.

4.5 Manuálne spustenie jednotlivých krokov

Alternatívne môžete spustiť jednotlivé skripty manuálne:

1. **Trénovanie modelov:**
    - Natrénovanie obrazového SVM modelu:

      ```
      python face_recognition.py --data_dir=SUR_projekt202
      ```

    - Natrénovanie audio GMM modelov:

      ```
      python audio_recognition.py --data_dir=SUR_projekt20
      ```

2. **Testovanie modelov:**
    - Klasifikácia obrazových dát:

      ```
      python face_recognition.py --data_dir=SUR_projekt202
      ```

    - Klasifikácia audio dát:

      ```
      python audio_recognition.py --data_dir=SUR_projekt20
      ```

3. **Fúzia výsledkov:**

   ```
   python fusion.py --audio_file=audio_results_enhanced.txt
   ```

4. **Validácia výsledkov:**

   ```
   python validate_results.py --audio_file=audio_results_enh
   ```