Klasifikace hudebního žánru

Petr Lorenc
Oleksandra Liutova

Popis projektu

Cílem projektu je vytvoření aplikace schopné klasifikovat vstupní nahrávku do jednoho z předem nadefinovaných žánrů. Definice žánrů probíhá tak, že je zvoleno několik klasických skladeb daného žánru, čímž se vytvoří referenční množina skladeb. Při dotazu jsou identifikovány nejpodobnější databázové skladby a na jejich základě je dotaz zařazen do daného žánru.

Způsob řešení

K účelům semestrálního projektu byl použit korpus 1000 skladeb o délce 30 vteřin. Data byla rozdělena do 10 žánrů (100 skladeb pro každý žánr). Data byla stažena z http://marsyasweb.appspot.com/download/data_sets/ .

Audio signál je popsatelný různými deskriptory. Známá sada deskriptorů, popisující audio z různých pohledů, jsou např. MPEG7 deskriptory. V současné době se ale využívají spíše MFCC (Mel-frequency cepstral coeffients), které zvuk lépe popisuje z hlediska lidského sluchu. Jeho vznik zahrnuje Fourierovu transformaci, škálování pomocí Melovy stupnice a také diskrétní kosinovu transformaci. Způsob tvorby tohoto deskriptoru je mimo rozsah této práce. Jisté seznámení s možnostmi extrakce příznaků nám poskytla i bakalářská práce Podobnostní vyhledávání klavírních skladeb[3].

Práce také zahrnuje grafický výstup. Jako výstup se proto používá webová stránka, kde je přehledně zobrazen výsledek našeho dotazu.

Implementace

Celý projekt je implementován v jazyce Python 3.
Knihovny:

- Librosa
- Scikit-learn
- Numpy
- Flask[7]

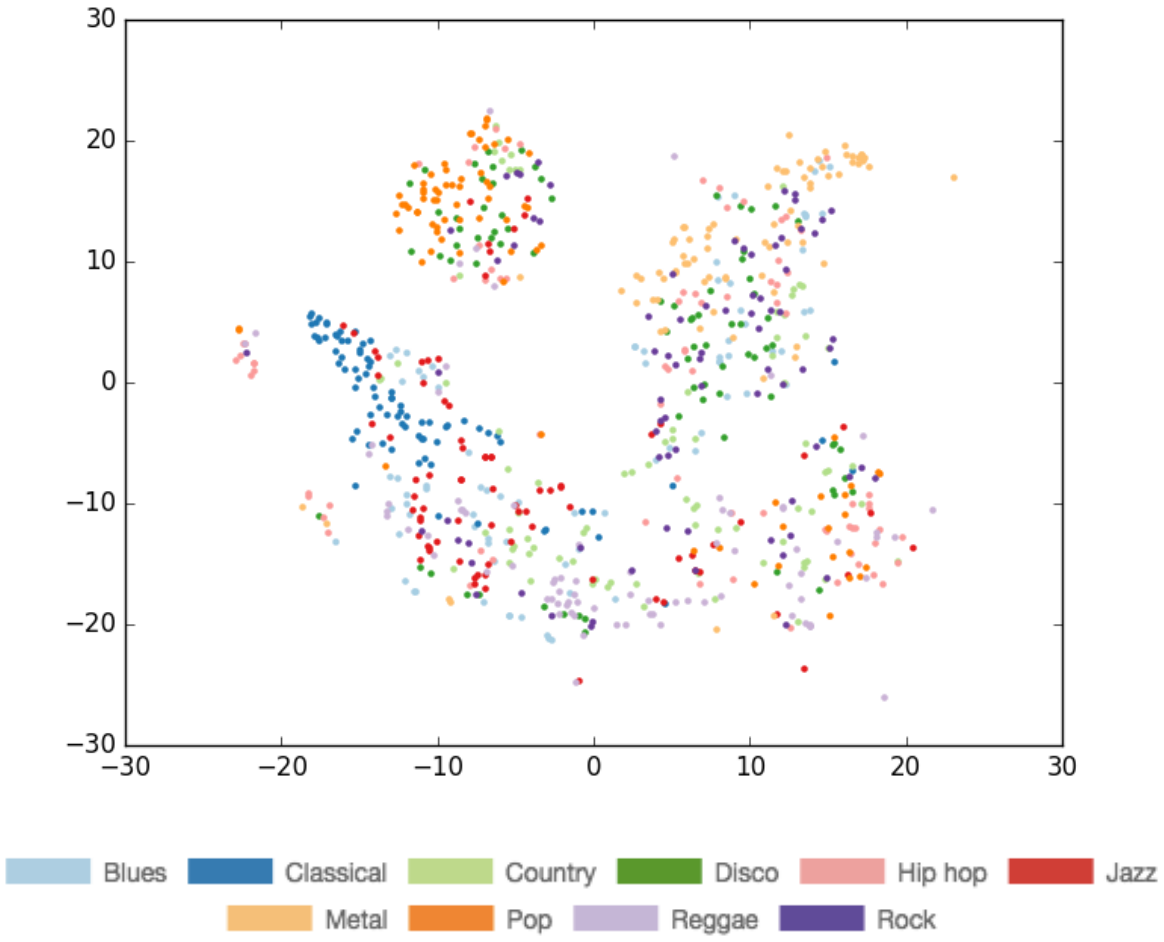
K extrakci příznaků z hudební skladby je používaná knihovna librosa, která nám umožňuje extrahovat MFFC. Z trénovacích dat vznikne matice 20x1293. Tuto množinu se snažíme redukovat na co nejmenší reprezentativní prvek. V literatuře[1] můžeme najít třeba způsob který zprůměruje složky jednotlivých 1293-složkových vektorů. Tak nám vznikne prostor o 20 dimenzích, který už se hodí pro další zpracování, třeba za pomoci metody k-Nearest Neighbours[4] (dále jen k-NN).

V experimentální fázi se budeme zabývat i jinými příznaky, které nabízí knihovna librosa. Dále budeme zkoušet různé metriky pro měření vzdálenosti mezi skladbami.

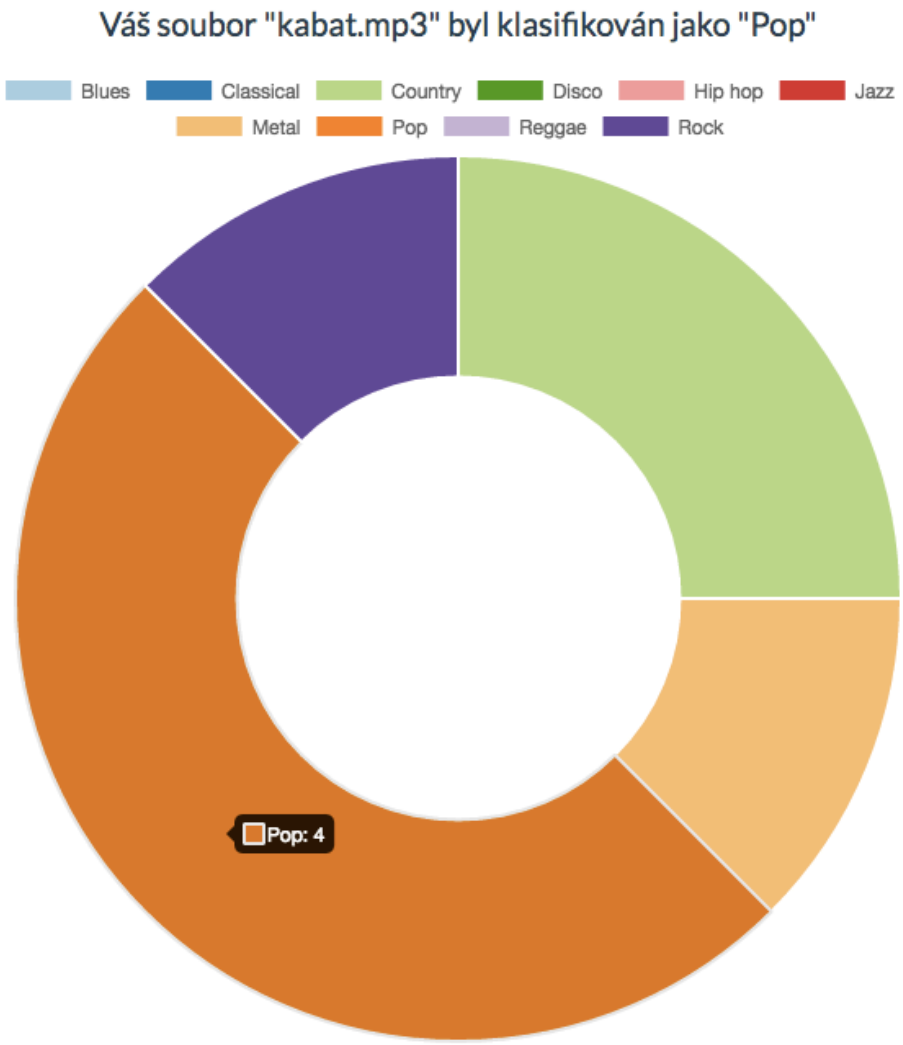
Výstup bude zobrazen pomocí knihovny Flask, kde budou implementovány 2 stránky. Na první bude možnost volit skladbu a na druhé se zobrazí výsledky v podobě grafu (uvidíte v příkladu výstupu).

Vizualizace příznaků

V rámci hledání vhodných příznaků jsme implementovali jejich vizualizaci pro lepší názornost. Využili jsme metodu t-SNE, která umí zobrazit data libovolné dimenze v 2D prostoru. Jednotlivé žánry jsou odlišeny barvou. Pro tento graf bylo použito jako příznaky průměry z 13 MFCC vektorů, zero-cross a odhad tempa. Je vidět závislost mezi prostorovými klusterami a jejich barvami.



Příklad výstupu



Experimentální sekce

Experimentálně bylo zjištěno, že 1 vteřina záznamu vygeneruje matici o velikosti 20x44. Toho bylo využito pro zprůměrování po sekundách a poté zprůměrování přes všech 20 řádků matice. Dosaženo bylo stejných výsledků jako když se jednotlivé řádky v matici (tj. 1293 členů) zprůměrují. Bylo proto využíváno tohoto druhého přístupu který je přehlednější a rychlejší. Pro výběr klasifikátoru a vhodného k pro k-NN bylo využito Eukleidovy metriky.

Pro účely testování bylo využito techniky cross-validace. Celková množina 1000 skladeb byla rozdělena na trénovací a testovací části. Po natrénování na trénovacích datech bylo zkoušena predikce na testovacích. K testovacím účelům bylo použito knihovny scikit-learn[2], která poskytuje třídu GridSearchCV[6]. Tato třída postupně zkouší různé parametry klasifikátoru a vrací data na níž je vidět úspěšnosti jednotlivých kombinací.

Tyto parametry byly testovány na k-NN klasifikátoru:

```
tuned_parameters = [
    {
        'n_neighbors': [1,3,5,7,9,11,13,15],
        'weights': ["uniform", "distance"],
        "metric" : ["euclidean","manhattan","chebyshev"]
    },
    {
        'n_neighbors': [1, 3, 5, 7, 9, 11, 13, 15],
        'weights': ["uniform", "distance"],
        "algorithm" : ["ball_tree", "kd_tree" ,"brute"],
        "leaf_size": [10,20,30,40,50],
        "metric": ["euclidean", "manhattan","chebyshev"]
    },
    {
        'n_neighbors': [1, 3, 5, 7, 9, 11, 13, 15],
        'weights': ["uniform", "distance"],
        "algorithm": ["ball_tree", "kd_tree", "brute"],
        "leaf_size": [10, 20, 30, 40, 50],
        "metric": ["minkowski"],
        "p": [1,2,3,4,5,6,7]
    }
]
```

Zde jsou vypsány nějaké konkrétnější výsledky.

- **Random Forest**
  - Úspěšnost 44.5 procenta
  - Dobré určení Reggae, Hip Hop, Metal a Classical (přes 80%)
- **k-NN – K=3**
  - Úspěšnost 40 procent

    - Dobré určení Blues a Classical (přes 80%)
- **KNN – K=5**
    - Úspěšnost 42 procent
    - Dobré určení Blues, Jazz a Classical (přes 80%)
- **KNN – K=7**
    - Úspěšnost 47 procent
    - Dobré určení Blues a Classical (přes 80%)
- **KNN – K=13**
    - Úspěšnost 46 procent
    - Dobré určení Blues, Classical a Metal (přes 80%)

Pro experimenty se vzdálenosti funkcí byl tedy vybrán 7-NN klasifikátor. U něho bylo vyzkoušeno několik metrik:

- **Kullback-Leibner distance**
    - Měří vzdálenost mezi dvěma pravděpodobnostními rozděleními – zde tedy nevhodné použití
    - np.sum(np.where(p != 0, p * np.log(p / q), 0))
    - Úspěšnost 15 %
- **Minkowski** kde p=3
    - Sum(abs(x-y)^p)^(1/p)
    - Úspěšnost 43%
- **Minkowski** kde p=4
    - Úspěšnost 43,5%
- **Chebyshev**
    - Max(abs(x - y))
    - Úspěšnost 43%
- **Eukleidova**
    - Sqrt(sum((x-y)^2))
    - Úspěšnost 40%
- **Manhatanská**
    - Sum(abs(x - y))
    - Úspěšnost 46%

Dále byla testována hodnota úspěšnosti klasifikace u Support Vector Machine[5] klasifikátoru (dále jen SVM). S těmito různými parametry:

```
tuned_parameters = [
    {
        "C": [0.1, 0.5, 1, 2, 4, 8, 10, 100],
        "kernel": ["linear"],
        "probability": [True]
    },
    {
        "C": [0.1, 0.5, 1, 2, 4, 8, 10, 100],
        "kernel": ["poly"],
        "degree": [2, 3, 4, 5],
```

```
        "coef0": [0.0, 0.2, 0.4, 0.6, 0.8, 1.0, 2, 4, 8, 16],
        "probability": [True],
    },
    {
        "C": [0.1, 0.5, 1, 2, 4, 8, 10, 100],
        "kernel": ["sigmoid"],
        "probability": [True],

    },
    {
        "C": [0.1, 0.5, 1, 2, 4, 8, 10, 100],
        "kernel": ["rbf"],
        "probability": [True],
    }
]
```

Zde bychom chtěli vyzdvihnout výsledek který dosáhl **SVM** pro parametry C=4 a kernel="linear". Jeho výsledky dosahovali hranice **úspěšnosti přes 55%.** Dobře určil Blues, Classical, Disco, Hip hop, Metal a Pop. Proto byl vybrán jako model, který bude používat webová aplikace.

Diskuze

Zvažovali jsme použití i jiných metod měření podobnosti. Z přednášek jsme získali dlouhý seznam (jako třeba Kosinova podobnost, Kosinova vzdálenost, Úhlová vzdálenost – tyto metriky jsme ale nemohli použít, protože nesplňovaly požadavky na metriku – nejčastěji trojúhelníkovou nerovnost)

Závěr

Práce nám rozšířila povědomí o možnostech vyhledávání podobností hudebních skladeb. Určování hudebního žánru je obtížný problém, protože mnoho skladeb ho nemá jasně definovaný a můžou proto pokrývat mnoho skupin. Z experimentální části je vidět, že rozlišení mezi Blues, Classical, Metal a Hip Hop je snadnější než třeba rozlišit Rock-Pop nebo Reaggue-Pop. Vyzkoušeli jsme si také několik vzdálenostních funkcí, tyto vědomosti se nám určitě budou hodit v jiných oblastech. V neposlední řadě jsme si také rozšířili povědomí o audiu a jeho způsobu zpracování – seznámili jsme se z MPEG-7 deskriptory a také se spektrálními charakteristikami jako MFCC.

Literatura

1. http://www.cp.jku.at/research/papers/A%20fast%20audio%20similarity%20retrieval%20method.pdf
2. http://scikit-learn.org/
3. https://edux.fit.cvut.cz/courses/MI-VMM.16/_media/tutorials/maiducan_2016bach.pdf
4. http://scikit-learn.org/stable/modules/neighbors.html#classification
5. http://scikit-learn.org/stable/modules/svm.html
6. http://scikit-learn.org/stable/modules/grid_search.html
7. http://flask.pocoo.org