Klasifikace hudebního žánru

Tomáš Rokos, Jan Rudolf

December 31, 2020

Contents

1 Zadání úlohy a úvod

Úkolem této semestrální práce je prozkoumat různé přístupy pro klasifikaci hudebního žánru. U každého přístupu bude vysvětleno jak funguje a uvedena jeho přesnost klasifikace (accuracy) na testovacích datech.

1.1 Struktura projektu

V root složce projektu se nachází složka **audio_classification**, ve které se nachází všechny zdrojové kódy všech částí této aplikace. Jsou zde implementace klasifikátorů a ve složce **run** poté frontend i backend webové aplikace.

Jednotlivé klasifikátory byly testovány v Python Jupyter noteboocích nacházejících se v root složce projektu. V každém je nejdříve načten dataset, naučena neuronová síť a poté vidět její výsledky v klasifikaci žánru.

1.2 Instalační příručka

Nasadili jsme demo na server pro vyzkoušení (více v sekci Webová aplikace). Kdyby ale bylo třeba vyzkoušet něco na lokálním vývojovém prostředí nainstalujte prosím pomocí následující příručky.

Pro spuštění jednotlivých příkazů je třeba mít nainstalovaný make, Python 3.8 a jako balíček přidanou knihovnu venv. Setup projektu probíhá pomocí **Makefile**, ve kterém jsou příkazy pro spuštění prostředí a instalaci knihoven. Stáhne také dataset GTZAN a udělá potřebný preprocessing. Lze pomocí něho také spustit webový server aplikace. Server v základu běží na portu 4666.

Pro setup projektu zavolejte příkaz **make setup-dev**. Pro stažení datasetu zavolejte příkaz **make get-dataset**. Ten potřebuje ale ještě preprocessing, proto prosím zavolejte **make preprocess**.

Jupyter notebook lze spustit pomocí příkazu **make ntb**, který ho spustí s správným prostředím venv.

Frontend serveru je zbuilděný již v gitu, protože je opravdu miniaturní.

1.3 Dataset

Využili jsme dataset GTZAN (dostupný z adresy http://opihi.cs.uvic.ca/sound/genres.tar.gz).

Tento dataset obsahuje 100 skladeb pro každý z jeho deseti hudebních žánrů ve formátu .wav. Každá ze skladeb je dlouhá 30 sekund. Všechny skladby mají 22050Hz, jsou pouze Mono s 16-bitovou reprezentací.

Dataset měl bohužel pozůstatky po uživatelově filesystému, proto jsme si napsali skript pro jeho preprocessing. Dataset lze stáhnout i preprocessovat pomocí make příkazů zmíněných výše.

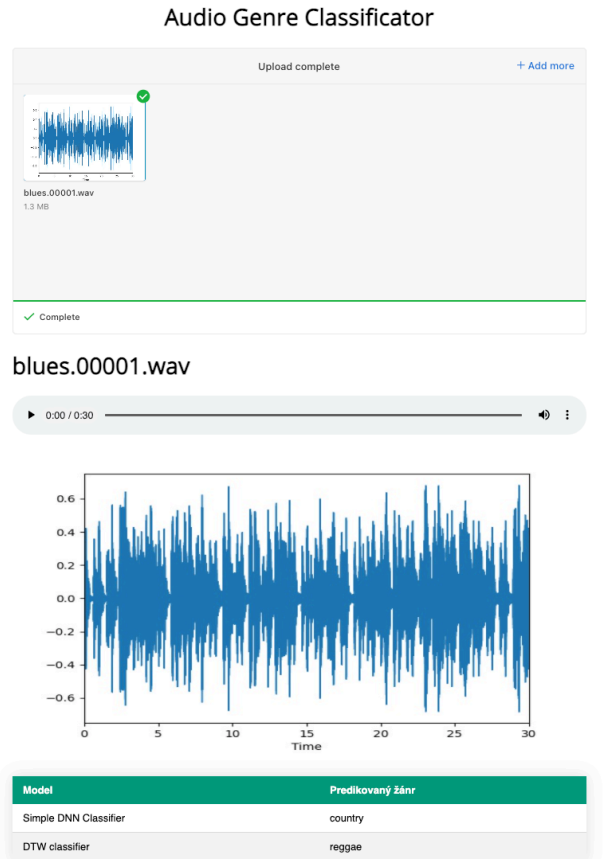
Dataset obsahuje následující žánry, náš projekt tedy vždy uploadovaný wav soubor klasifikuje do následujících žánrů.

- blues
- classical
- country
- disco
- hiphop
- jazz
- metal
- pop
- reggae
- rock

1.4 Webová aplikace

Demo webové aplikace je dostupné z adresy `http://134.209.243.49/`.

Frontend aplikace pouze nahraje soubor na backend. Ten se chová jako API a vrátí žánr pro všechny metody zmíněné v sekci Přístupy klasifikace.



Po nahrání dané skladby na server je rozřezána na překrývající se úseky o velikosti 30 sekund. Tyto úseky poté necháme klasifikátorem predikovat. Převažující predikovaný žánr prohlásíme za výsledný žánr dané vstupní skladby.

Odtud také přichází omezení na délku skladby. **Musí mít alespoň 30 s jinak jí bohužel naše aplikace nedokáže predikovat.** Pro vstupní soubor lze zvolit cokoliv, co podporuje knihovna librosa, nejlépe však **.wav** soubory.

2 Rešerše použitých přístupů

Obecně při aplikaci strojového učení na audio data je třeba zvolit přístup, jakým získáme z dat užitečné příznaky (feature extraction) a vhodný model strojového učení, který umí s těmito příznaky pracovat tak, aby došel ke správným závěrům.

Mezi nejpopulárnější feature extraction metody pro audio patří MFCC a MPEG7. MFCC je dnes upřednostňován pro práci s hudbou, zatímco MPEG7 je vhodnější pro obecné audio a speech recognition.

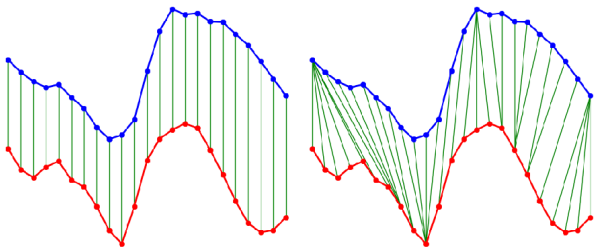
2.1 MFCC

Lidské ucho je citlivější na nižší frekvence. Frekvence zvuku tedy vnímáme nelineárně, proto je zde logaritmická Melova škála, které převádí vyšší frekvence logaritmicky na nižší. MFCC využívá tuto škálu v kombinaci s překrývajícími se zvukovými stopami. Na tyto překrývající se stopy je poté použita Fourierova transformace. Jednotlivé koeficienty tedy reprezentují spektrum logaritmicky upravené stopy.

Hlavním parametrem pro feature extraction MFCC je počet, kolik takových koeficientů chceme z získat. Doporučená hodnota je 13 - 20. Čím vyšší index koeficientu tím méně významný jako reprezentace úseku je. Když je MFCC použit na skladbu získáme matici, kde ve sloupcích jsou identifikátory daných časových stop a v řádcích je i-tý koeficient.

2.2 Dynamic Time Warping

První přístup ke klasifikaci je **Dynamic Time Warping**. Dynamic time warping je algoritmus pro zjištění podobnosti mezi dvěma časovými sekvencemi, které můžou být různě posunuté nebo se lišit v jejich rychlostech.



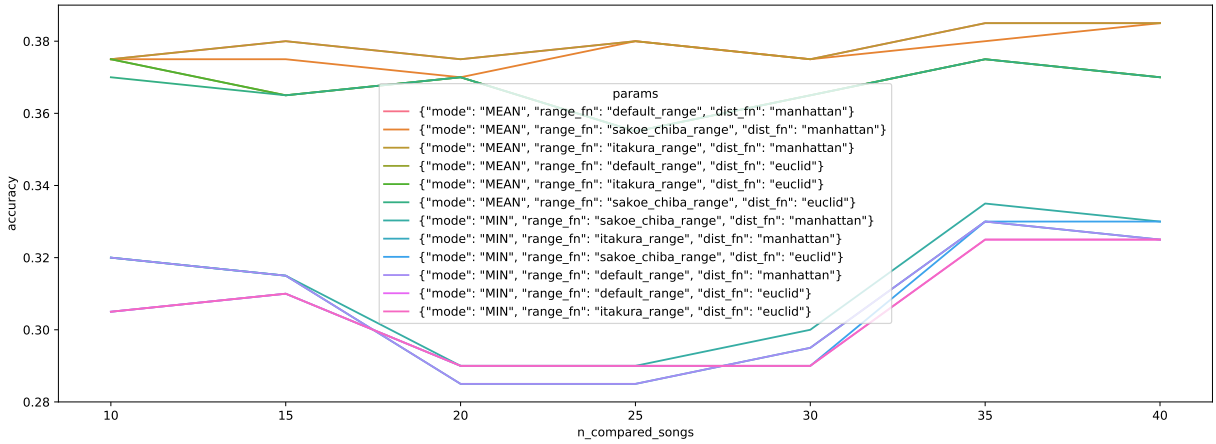
V předchozím obrázku se použilo dtw pro dvě časové řady v dimenzi 2, ale obecně může být využit pro vektory jakýchkoliv délek. MFCC dobře reprezentuje danou skladbu, ale kdybychom jí porovnávali s jinou, která by byla časově posunutá, tak tradiční Minkowskiho vzdálenosti nejsou dobré. Využijeme tedy DTW jako vzdálenostní funkci MFCC koeficientů.

Bohužel s dobrou porovnávací funkcí přichází také její výpočetní náročnost. Kvůli její výpočetní náročnosti přišly různé metody pro ořezu prohledávaného prostoru. Implementoval jsem dvě z nich - **itakura_range** a **sakoe_chiba_range**. Tyto metody oříznout prohledávaný prostor podle diagonály a tím zrychlí průběh algoritmu.

Klasifikaci hudebního žánru s DTW provádíme pomocí porovnání vstupní skladby s nějakým počtem skladeb od každého žánru. Počet skladeb je určen podle parametru **n_compared_songs**.

Po porovnání dvou skladeb máme vzdálenost mezi jednotlivými MFCC koeficienty těchto skladeb. Po porovnávní s n skladbami poté chceme nějakou agregační funkci, pro zvolení žánru. Pro pokusy jsem využil dvě - průměrná vzdálenost od těchto n písní a nejkratší soused (minimum). Tato hodnota je určena parametrem **mode**.

V následujícím grafu jsou vidět jednotlivé parametry DTW a podle nich jeho accuracy. Na ose x je počet porovnávaných skladeb a na ose y accuracy.



Z tohoto grafu je vidět, že nejlépe vychází dobře dtw nastavené s právě parametry `"mode": "MEAN", "range_fn": "itakura_range", "dist_fn": "manhattan"`. Stačí mu porovnat pouze 15 písní pro velmi dobrou accuracy.

2.3 Jednoduchá hluboká neuronová síť

Další možností klasifikace je využití neuronové sítě. Pro vytvoření features znovu použijeme MFCC, ale tentokrát budeme potřebovat o moc méně features. Celý vektor by určitě nebyl dobrým vstupem kvůli narůstající komplexnosti neuronové sítě.

Každý vektor jednotlivých koeficientů tedy zprůměrujeme a tuto hodnotu použijeme jako hodnotu dané feature. Pro každou píseň máme tedy N features (na základě počtu koeficientů, které zadáme MFCC).

Tyto features jsou předány vstup neuronové sítě a přidáme pouze pár plně propojených hidden vrstev. Mezi nimi Droupout vrstvy pro zabránění přeučení. Výstupní vrstvu použijeme aktivační funkci softmax pro její dobré vlastnosti pro klasifikaci. Výstupní vrstva má deset neuronů podle deseti žánrů.

Tento přístup nám ihned dal testovací accuracy 57% což je oproti DTW velmi dobrý výsledek.

2.4 ResNet-18

Dále jsme zkusili aplikovat konvoluční neuronovou síť, konkrétně 18 vrstvou síť s architekturou ResNet a preaktivačními bloky. Tuto neuronovou síť aplikujeme na MFCC.

Každá konvoluční vrstva používá 3x3 filtry a jejich počet ve vrstvě se snižuje s každou redukcí velikosti feature map po vzoru běžných ResNet sítí. Redukce velikosti je provedena pomocí strides. Výstup konvolučních vrstev agregujeme pomocí max poolingu a tyto features jsou klasifikovány ve dvou fully connected vrstvách.

Síť jsme trénovali po 40 epoch s použitím optimalizačního algoritmu Adam, learning rate 1e-3 a batch size 32. Model získal accuracy 77% na testovací množině.

2.5 ResNet-18 embedding s KNN klasifikací

Stejnou architekturu jsme použili jako embedding model, s výjimkou výstupní vrstvy, která má 128 neuronů. Výstup embeddingu klasifikujeme pomocí KNN, kde n jsme zvolili 12.

Embedding jsme trénovali pomocí jako siamské sítě s Semi Hard Triplet loss funkcí, jak byla představena v originálním paperu. Idea za touto metodou je použití tří stejných sítí se sdílenými váhami na tři samply: anchor sample, positive sample (sample ze stejné třídy jako je anchor) a negative sample (sample z jiné třídy než anchor). Optimlizace podle triplet loss funkce pak znamená maximalizování vzdálenosti mezi anchor a negative samplem a minimalizování vzdálenosti mezi anchor a positive samplem.

Výstupem embedding modelu jsou samply v 128 dimenzionálním prostoru. Na predikované samply nejdřív aplikujeme embedding a výstup klasifikujeme pomocí KNN.

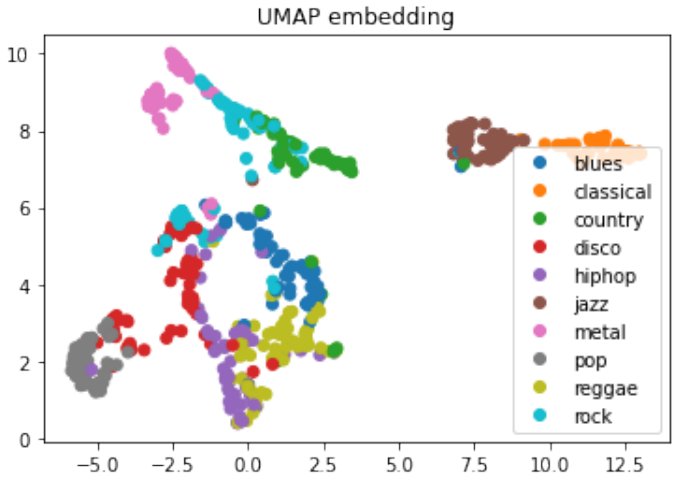


Figure 1: Výstup embedding modelu aplikovaného na trénovací množinu. Dimenze je reukována pomocí UMAP algoritmu

Síť jsme trénovali po 40 epoch s použitím optimalizačního algoritmu Adam, learning rate 1e-3 a batch size 32. Výsledný model získal accuracy 65% na testovací množině. Výsledek je horší než samotatný ResNet pro klasifikaci, nicméně tato metoda dovoluje dělat predikce i pro žánry neznámé během trénování bez nutnosti úprav modelu.

2.6 VGG-16

Po dobrých výsledcích ResNetu bylo příhodné vyzkoušet i jiné konvoluční neuronové architektury. Tato architektura úspěšně vyhrála v roce 2014 ILSVR(Imagenet) competition. Vyznamenala se hlavně na obrázcích s vysokým rozlišením. My tak velké vstupní data nemáme, ale stejně by bylo dobré neuronovou síť vyzkoušet.

VGG-16 využívá 3x3 konvoluční filtry se stride 1 a pokaždé využívá stejný padding a maxpool vrstvu (2x2 se stride 2). Toto uspořádání používá konzistentně celou architekturou.

Na konci se poté nachází 2 fully connected vrstvy se softmaxem. Mezi nimi jsou poté dropouty pro zamezení přeučení a rychlé konvergenci.

U VGG 16 označuje právě 16 vrstvev, pro které trénujeme váhy.

Síť jsme trénovali po 32 epoch s použitím optimalizačního algoritmu Adam, learning rate 1e-3 a batch size 32. Model získal pouze accuracy 47% na testovací množině.

3 Literatura

Čerpali jsme z přednášek na VMM a z následujících článků:

The dummy's guide to MFCC

Mel Frequency Cepstral Coefficient (MFCC) tutorial

Understanding the Mel Spectrogram

Dynamic Time Warping

ResNet paper

TripletLoss paper

4 Repozitář

https://gitlab.fit.cvut.cz/rudolja4/ni-vmm-music-genre-classification

5 Závěr

Úloha byla zajímavá a mohli jsme na ní vyzkoušet nové techniky pro učení neuronových sítí.

Výsledkem práce je funkční nasazený prototyp. Bohužel je ale nasazen na trochu starším hardware tak predikce chvíli trvá.