



# Klasifikace hudebního žánru

Jan Rudolf

January 29, 2021

## Contents

<b>1</b>	<b>Zadání úlohy a úvod</b>	<b>2</b>
1.1	Struktura projektu . . . . .	2
1.2	Instalační příručka . . . . .	2
1.3	Dataset . . . . .	2
1.4	Webová aplikace . . . . .	3
<b>2</b>	<b>Rešerše použitých přístupů</b>	<b>4</b>
2.1	MFCC . . . . .	4
2.2	ResNet-18 klasifikace . . . . .	4
2.3	ResNet-18 embedding s KNN klasifikací . . . . .	4
<b>3</b>	<b>Literatura</b>	<b>5</b>
<b>4</b>	<b>Repozitář</b>	<b>5</b>
<b>5</b>	<b>Závěr</b>	<b>5</b>

# 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 notebookách 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 uživatelském filesystému, proto jsme si napsali skript pro jeho preprocessing. Dataset lze stáhnout i preprocessovat pomocí make příkazů zmíněných výše.

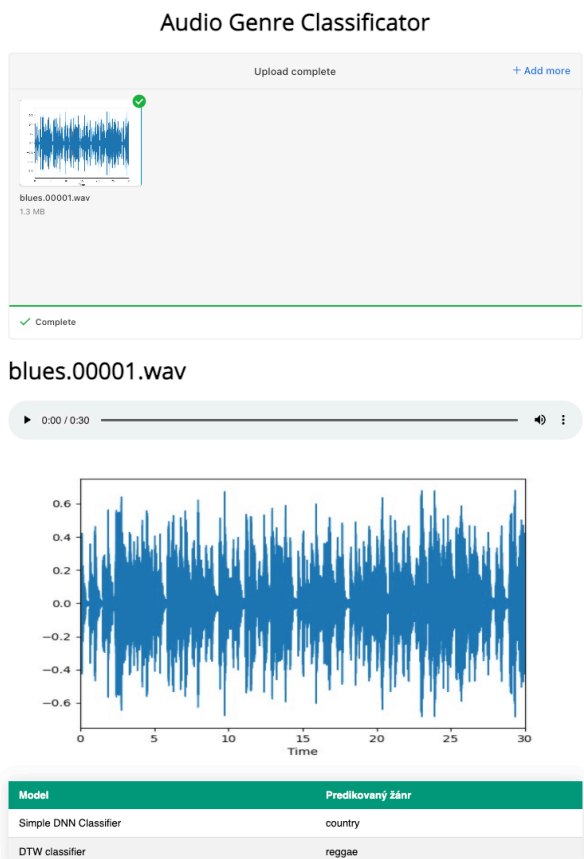
Dataset obsahuje následující žánry, náš projekt tedy vždy uploadovaný wav soubor klasifikuje do následujících žánrů.

- 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 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 ResNet-18 klasifikace

Aplikovali jsme konvoluční neuronovou síť, konkrétně 18 vrstvou síť s architekturou ResNet a praktičtějšími bloky. Vstupem této sítě jsou MFCC.

Každá konvoluční vrstva používá  $3 \times 3$  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 poolingů 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.

### 2.3 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 function, jak byla představena v originálním paperu. Idea za touto metodou je použití tří stejných sítí se sdílenými váhami na tři samplly: anchor sample, positive sample (sample ze stejné třídy jako je anchor) a negative sample (sample z jiné třídy než anchor). Optimalizace 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 samplly v 128 dimenzionálním prostoru. Na predikované samplly nejdříve aplikujeme embedding a výstup klasifikujeme pomocí KNN. KNN obsahuje příznaky trénovací sady.

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 samotný ResNet pro klasifikaci, nicméně tato metoda dovoluje dělat predikce i pro žánry neznámé během trénování bez nutnosti úprav modelu.

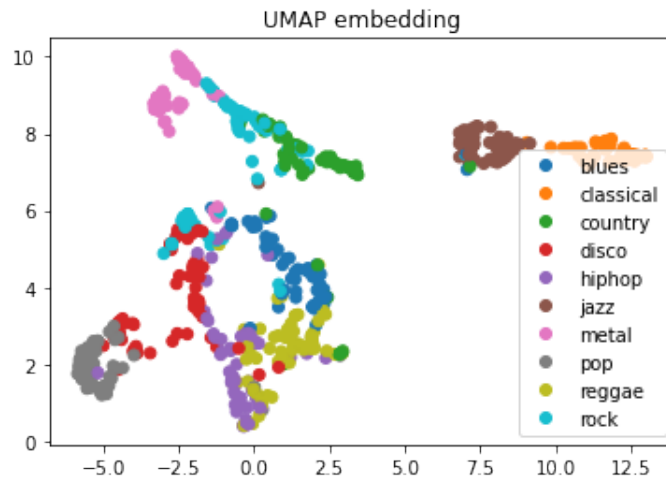


Figure 1: Výstup embedding modelu aplikovaného na trénovací množinu. Dimenze je reukována pomocí UMAP algoritmu

### 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

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

Model	Accuracy
Dynamic Time Warping	0.39
Jednoduchá hluboká neuronová síť	0.57
<b>ResNet-18</b>	0.77
<b>ResNet-18 embedding s KNN klasifikací</b>	0.65
VGG-16	0.47

Table 1: Výsledky dosažené našimi modely. Zvýrazněné modely pochází od autora reportu

Nejlépeších výsledků dosáhl model ResNet pro klasifikaci. Ovšem i přístupy založené na podobnosti mají podle nás dobré uplatnění, díky tomu, že se dají využít i pro další žánry.