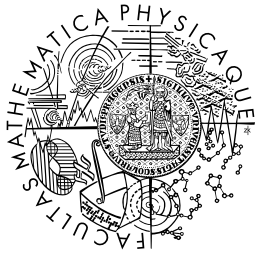


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Jiří Balhar

Extrakce melodie pomocí hlubokého učení

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce:	Mgr. Jan Hajič Ph.D.??
Studijní program:	Informatika
Studijní obor:	Programování a softwarové systémy

Praha 2019

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Poděkování.

.

Název práce: Extrakce melodie pomocí hlubokého učení

Autor: Jiří Balhar

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Jan Hajič Ph.D.??, ústav

Abstrakt: Abstrakt.

Klíčová slova: klíčová slova

Title: Melody Extraction using Deep Learning

Author: Jiří Balhar

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Jan Hajič Ph.D.??, institute

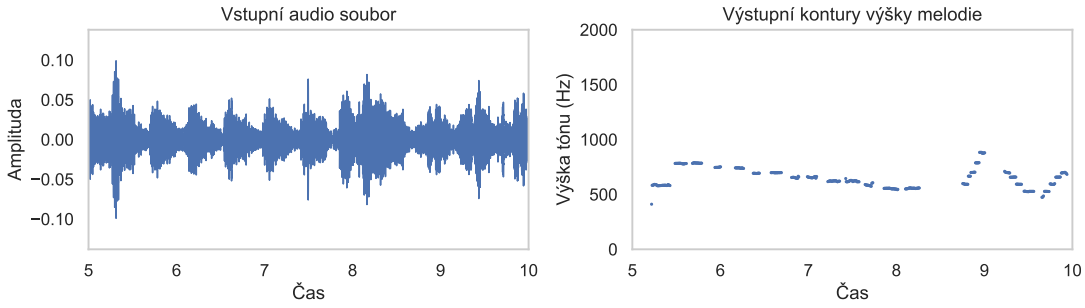
Abstract: Abstract.

Keywords: key words

Obsah

1. Úvod

Spolu s harmonií a rytmem představuje melodie základní stavební kámen většiny existující hudby. V průběhu vývoje od folklórních zpěvů přes orchestrální skladby po soudobou elektroniku si melodie téměř vždy zachovávala své dominantní postavení nositele esence jednotlivých písní. Melodie je to hlavní, co si člověk po poslechu skladby odnáší a nejsnadněji vybaví, a její důležitost je zejména v našem kulturním kontextu natolik jednoznačná, že se hudba, která ji postrádá, zřídkakdy dostává do širokého povědomí.



Obrázek 1.1: Příklad vstupu a výstupu metody pro extrakci melodie. přidat zvukový přílkad do přílohy

Tato práce se zabývá metodami odhadu fundamentální frekvence melodie ze zvukové nahrávky. Jinými slovy je naším cílem získat v každém bodě vstupní skladby informaci o tom, zda melodie v daném okamžiku zní a její případnou výšku. Jde o jednu z nejdůležitějších a zároveň nejtěžších úloh z oboru *Music Information Retrieval* (MIR), jejíž rozsah využití v této doméně pokrývá významnou část aktivně řešených, otevřených problémů.

Spolehlivý přepis melodie by usnadnil vyhledávání v hudebních datech, ať už na základě notového zápisu (*Symbolic Melodic Similarity*), pomocí nekvalitní nahrávky z rádia (*Audio Fingerprinting*), pomocí broukání (*Query by Singing/Humming*) nebo dokonce pomocí coveru hledané písně (*Audio Cover Song Identification*). Mimo vyhledávání by byl algoritmus užitečný pro další zpracování zvukového signálu, ať už pro manipulaci a úpravu melodického hlasu (například software Melodyne), nebo naopak jeho odstranění a vytvoření karaoke doprovodu (*Informed Source Separation*). V neposlední řadě by extrakce melodie pomohla při kategorizaci hudebních dat, například podle žánru (*Genre Classification*) nebo podle zpěváka (*Singer Characterization*). A konečně široké spektrum využití by nalezla i v muzikologii (případně etnomuzikologii) pro kvantitativní i kvalitativní studii hudebních motivů a postupů (V jazzu například Pfleiderer a kol.).

Extrakce melodie však nemusí sloužit pouze jako mezikrok pro řešení jiné úlohy, užitečný je i samotný výstup algoritmu, znázorněný na obrázku 1.1. Motivačním příkladem použití může být pomoc při transkripci. Představíme-li si začínajícího hráče na saxofon, který si chce do not přepsat své oblíbené jazzové sólo, výstup algoritmu mu dá užitečnou informaci o tom, jaký tón zní v jakou chvíli. Z této reprezentace už pak hráči zbývá nalezené tóny projít a zapsat je do notové osnovy.

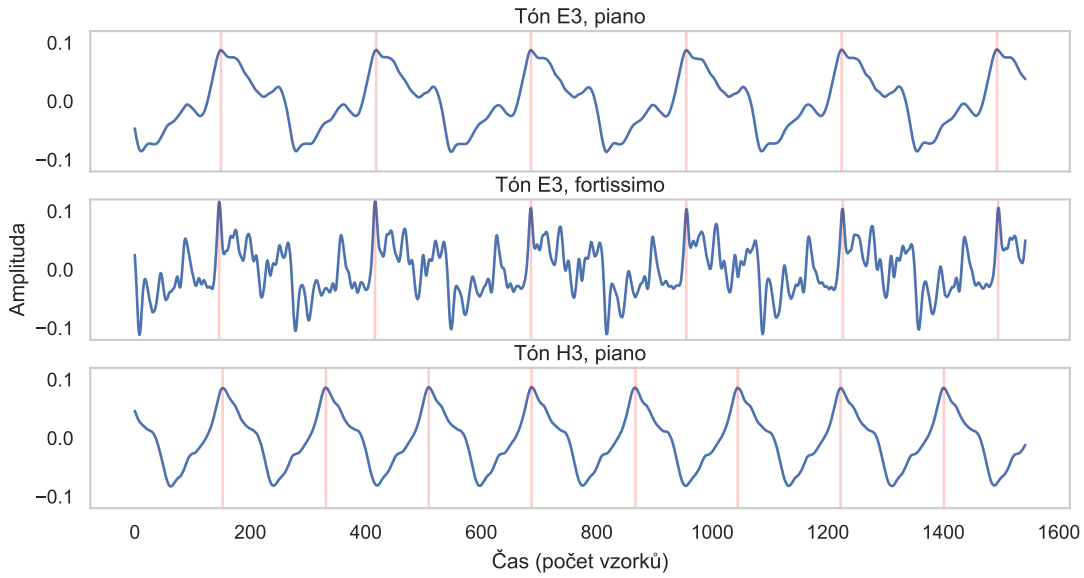
1.1 Analýza hudebního signálu

Proč je ale extrakce melodie otevřený problém? Příbuzná úloha, která spočívá v přepisu nahrávky jednoho izolovaného nástroje, je v podstatě vyřešena (Mauch a Dixon, 2014), proč se tato úloha po přidání hudebního doprovodu stává výrazně obtížnější? Pro vysvětlení zásadního problému, který se s přepisem nahrávky pojí, musíme nejdříve přiblížit vůbec povahu zvuku a možnosti jeho zkoumání.

Naše zkušenost se zvukem probíhá primárně skrze sluch. Teprve na hlasitém koncertu však člověk pocítí, že zvuk je ve své fyzikální podstatě změna tlaku vzduchu, putující od zdroje k posluchači. Díky sluchu z těchto vibrací dokážeme oddělit jednotlivé zdroje a identifikovat v nich i velmi jemné rozdíly. Ačkoli jde o subjektivní vjemy, zvuky lze částečně rozřadit podle toho, jak snadno v nich rozeznáme nějakou konkrétní výšku.

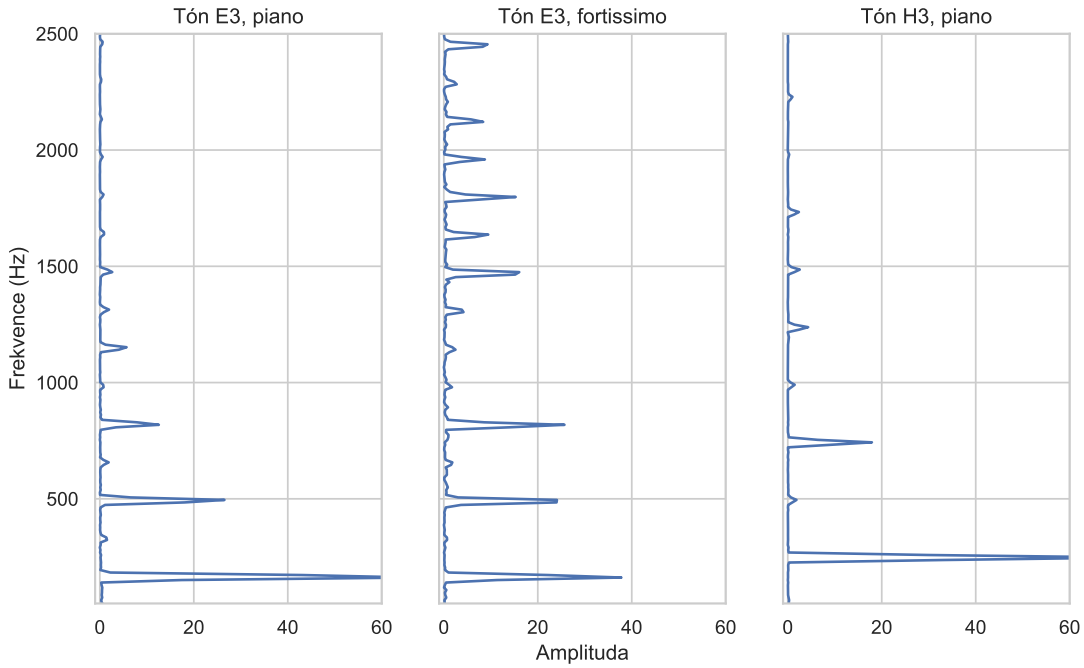
Čtenář této práce si nyní může postupně vybavit: hrající violoncello, odbíjení kostelního zvonu, cinknutí příboru, štěkot psa, plynutí potoka, šelest listí stromů, trhání papíru, tlesknutí a prasknutí balónku.

Se ztrácející se zřetelností výšky nejprve přijdeme o možnost zpívat společně se zdrojem zvuku v harmonii a posléze i o možnost si představit „vyšší“ a „nižší“ instance toho samého zvuku (jak zní vysoké a nízké prasknutí balónku?). To, co mají první z uvedených příkladů společné, je výrazná a stabilní periodicita jejich signálu — daný tlakový průběh se opakuje v čase. Díky sluchu tuto periodicitu interpretujeme jako výšku, přičemž různé výšky se od sebe liší frekvencí, se kterou se signál opakuje. Hudební nástroje jsou jedním ze zdrojů těchto pravidelných vibrací, jejichž frekvenci lze zpravidla měnit (pomocí klapek, pohybu prstu po struně, atd.). Hlas nástroje však není charakteristický pouze svou výškou, nýbrž i barvou. Ta je určena podobou signálu v rámci jedné periody.



Obrázek 1.2: Zvuk klarinetu, tóny s různou výškou a dynamikou, 35 milisekund signálu se vzorkovací frekvencí 44 100 Hz. nějak uvést zdroj zvuku https://www.philharmonia.co.uk/explore/sound_samples/clarinet?p=3

Na obrázku 1.2 můžeme srovnat tři tóny hrané klarinetem, první dva mají stejnou výšku, jsou ale zahrané s různou intenzitou (dynamikou) je tohle správně formulované, vím že dynamika není intenzita, ale „zahrané s různou dynamikou“ mi zní divně?. Jejich vizuální rozdíl částečně odpovídá i rozdílu v barvě tónu, první tón má příjemný, měkký zvuk; druhý je výraznější a hrubší. Třetí tón se od zbylých liší svou výškou, což lze pozorovat na kratší periodě signálu, která je na obrázku vyznačená úsečkami.



Obrázek 1.3: Zvuk klarinetu, absolutní hodnota výstupu Fourierovy tranformace signálu délky 4096 s oknem typu Hamming.

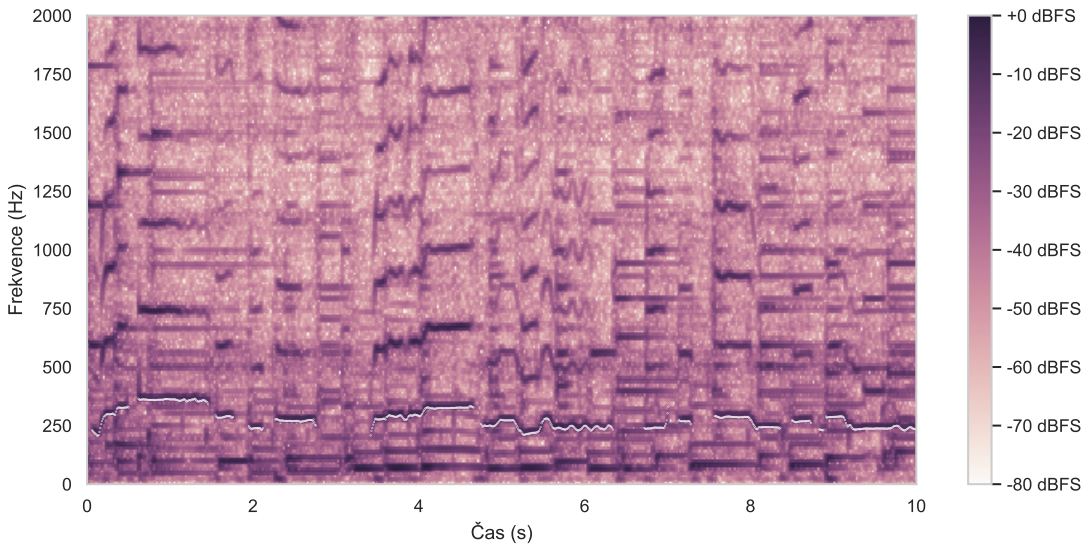
Jedním ze způsobů analýzy zvukového signálu je pomocí Fourierovy transformace (DFT). Základní myšlenkou je, že na signál lze hledět jako na vážený součet jednodušších signálů. Podobně, jako když se barvy na obrazovce míchají ze tří základních, libovolný zvuk můžeme smíchat ze sady sinusoid. Výslednou kombinaci všech frekvencí, ze kterých se zvuk skládá, označujeme *zvukové spektrum*. Na obrázku 1.3 vidíme část výsledku Fourierovy transformace zvuků klarinetu z předchozího příkladu. To zásadní, co na spektru tónu můžeme pozorovat, je jeho podstata jakožto součet *harmonických složek*. Tón, kterému posluchač přisoudí výšku f_0, se zpravidla skládá ze součtu sinusoid, jejichž frekvence je celočíselným násobkem základní frekvence f_0 (jinak také nazývaná *fundamentální frekvence*). Například tedy tón E3 se na obrázku 1.3 skládá ze složek o frekvenci 165 Hz, 330 Hz, 495 Hz, . . . , zároveň intenzita těchto harmonických frekvencí určuje barvu hlasu.

Ukazuje se, že práce s touto reprezentací zvuku je pro analýzu signálu užitečnější, než práce s nezpracovaným signálem. Ze spektrální reprezentace je například na první pohled zřejmý vztah fundamentálních frekvencí porovnávaných signálů, který odpovídá lidské intuici o výšce zvuků — tón H3 je na obrázku 1.3 opravdu „výše“ než tón E3. Díky spektrální analýze lze také pozorovat charakteristiky hlasů různých nástrojů. Pro hlas klarinetu platí, že liché harmonické

frekvence jsou mnohem výraznější než sudé (na obrázku 1.3 vypadají sudé harmonické jako malé vrcholky mezi výraznými lichými), naopak například lidský zpěv je charakteristický výraznějšími sudými harmonickými složkami. Další výhodou je možnost hledání rozdílů v barvě tónů — na spektru vidíme, že vyšší harmonické jsou u tónu hraném fortissimo mnohem výraznější než u tónu hraném piano. Tyto vyšší frekvence způsobují zmiňovanou hrubost tónu. a tohle se dá říct? Tón hraný fortissimo/piano. Moje znalosti hudební terminologie jsou nulové

Harmonická struktura, která je vlastní lidskému hlasu a téměř všem zvukům hudebních nástrojů, je zásadní pro metody extrakce melodie. Je to vlastnost, která zvuky potenciálně nesoucí melodii odlišuje od bubnového doprovodu, od šumu nebo od jiných nemelodických rušení. Díky ní se také můžeme pokoušet rozložit souzvuk různě vysokých tónů na jejich původní, čisté signály.

spektrogram hudby - basa, zpěv a bubny. Pod tím obrázek kompletního přepisu melodických kontur



Obrázek 1.4: Spektrogram zpěvu s doprovodem piana, basy a perkusí; zpívaná melodie je vyznačena bílým obrysem.

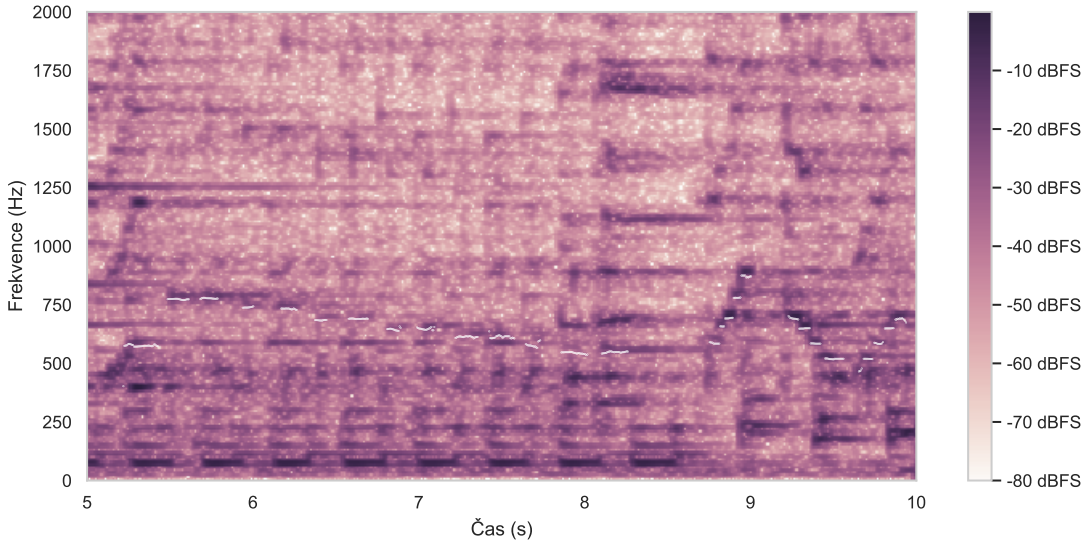
Obrázek 1.4 vznikl pomocí opakované Fourierovy transformace, která byla aplikována na po sobě jdoucí, krátké časové úseky vstupní nahrávky, přičemž intenzity frekvenčních složek v každém časovém okamžiku zvuku jsou nyní znázorněny odstínem barvy. Časově-frekvenční reprezentaci signálu nazýváme obecně *spektrogram*, a jeho výpočet je prvním krokem většiny metod pro extrakci melodie.

Na spektrogramu 1.4 můžeme pozorovat harmonické struktury tónů — vyznačenou konturu hlasu, která se na frekvenční ose pohybuje volněji, a pak klavírní a basový doprovod, charakteristický frekvenční stabilitou a v čase slábnoucí amplitudou. Na tomto jednoduchém příkladu lze melodii zpozorovat poměrně snadno, nese ji velmi výrazný, v poměru k doprovodu nejsilnější hlas. Lze na něm však prezentovat první ze základních problémů extrakce melodie.

Frekvence tónů, ze kterých se skládá hudební skladba, jsou uspořádány do stupnic, které definují pevně dané poměry (hudební *intervaly*), ve kterých se tyto tóny ve skladbě mohou vyskytovat. Principem libozvučnosti jsou však takové

intervaly, které způsobují, že harmonické frekvence jednotlivých tónů se překrývají a ve výsledné směsi pak není zřejmé, zda-li daná harmonická frekvence patří k jednomu, či více hlasů. Hudební doprovod, pro lidské ucho znějící „pod melodií“, tedy často svými harmonickými frekvencemi zasahuje do melodie samotné, což je zřejmé ze spektrální analýzy.

Dekompozice signálu na jednotlivé znějící hlasy, která je pro člověka přirozená podobně, jako porozumění řeči v rušné kavárně, se kvůli této harmonické povaze tónů a intervalů stává pro algoritmy extrakce melodie obtížným problémem. To, co pro nás činí hudbu zajímavou pro poslech, ji činí obtížně analyzovatelnou pro počítač.



Obrázek 1.5: Spektrogram orchestrální skladby s obtížně detekovatelnou melodií.

1.2 Definice melodie

Rozpoznání melodie v rámci hudební skladby je pro většinu posluchačů intuitivní schopností, která je součástí prožitku poslechu hudby, a která jejímu poslechu vůbec dává smysl. Ačkoli je melodie tedy termín, který je subjektivně jasný, formální, obecně přijímanou muzikologickou definici, která by se zpětně neodkazovala k posluchači, nemá.

Z tohoto důvodu si výzkumné týmy zabývající se automatickou transkripcí melodie volí pragmaticky spíše užší definice melodie, se kterými se v jejich kontextu pracuje lépe. Práce Goto a Hayamizu (1999), která je považována za jednu z prvních prací v oboru, chápe melodii jako „konturu fundamentální frekvence sestávající se z nejsilnějších tónů hrajících v omezeném frekvenčním rozsahu“. Práce se tedy omezuje na poměrně úzké chápání melodie, obecně se totiž tóny melodie jistě mohou vyskytovat i mimo autory specifikovaný frekvenční rozsah a také nemusí být vždy v poměru s doprovodem nejhlasitější složkou signálu. Z technického hlediska však umožnila autorům implementaci algoritmu běžícího v reálném čase, který poskytoval sémanticky bohatý popis vstupních nahrávek. Navazující články již pracují s volnějšími definicemi, které lépe reflektují podstatu melodie.

Kompromisem mezi subjektivní a praktickou definicí se na dlouhou dobu stala „extrakce základní frekvence hlavního, neměnného, melodického hlasu“. Ačkoli melodii v reálném hudebním materiálu obvykle nese více hlasů, které se v hraní střídají (například píseň se zpěvem a kytarovým sólem), v letech 2005 – 2015 se v soutěži MIREX provádí evaluace pouze nad krátkými výňatky, kde tato definice není omezující. Přestože se může na první pohled zdát, že tato definice pouze uměle zjednodušuje celou úlohu, její formulace vede k rozvoji nových a zajímavých přístupů, které se sice formálně soustředí právě na extrakci pouze jednoho neměnného hlasu, ale ve výsledku překvapivě dobře fungují i na složitější skladby. Příkladem nového směru může být extrakce melodie pomocí modelování hudebního záznamu jako součtu signálu jednoho hlasu a doprovodu (práce Durrieu a David (2010) nebo Bosch a Gómez (2016)) s přesahem do příbuzné úlohy oddělení hlasů (source separation). Některé práce se zaměřují ještě konkrétněji na separaci lidského zpěvu a doprovodu (Hsu a Jang (2010), Ikemiya a kol. (2016)). Nově se také objevují práce, které „hlavní“ melodický hlas neinterpretují nutně jako „nejsilnější“ a k jeho rozlišení využívají dalších rysů, jako je barva, vibrato nebo délka not. Například Salamon a Gomez (2012) využívají těchto rysů pro finální výběr mezi extrahovanými kandidátními konturami.

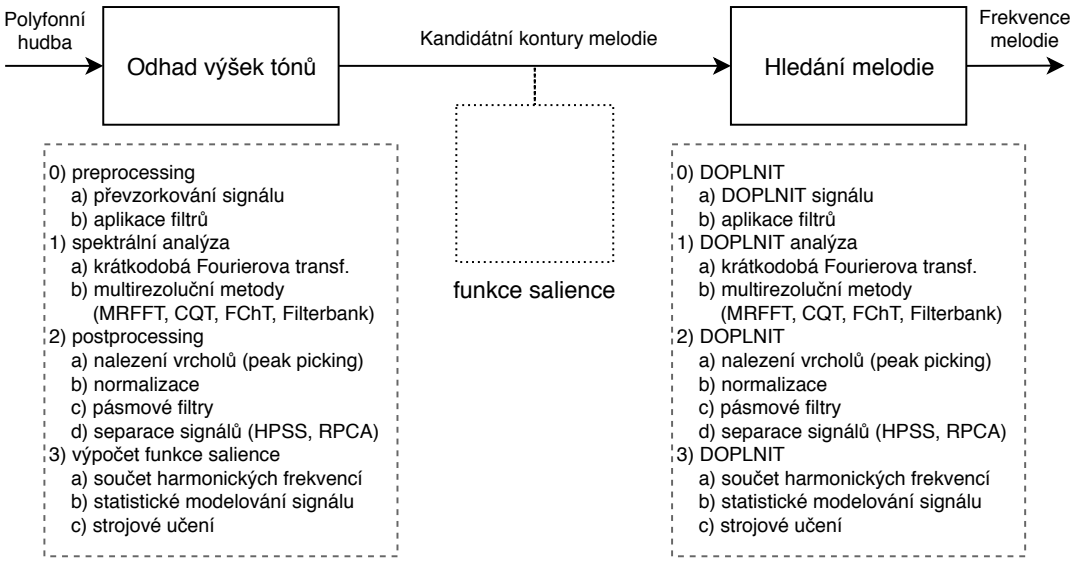
Ve svém přehledovém článku Salamon a kol. (2014) dochází k závěru že výzkum začal v letech 2009–2012 stagnovat, nová data jsou proto pro další vývoj oboru zásadní. Výrazným posunem v rámci MIR komunity proto bylo zveřejnění nových datasetů MedleyDB (Bittner a kol., 2014) a ORCHSET (Bosch a kol., 2016), oba obsahují data, ve kterých již melodii nenese pouze jeden hlas po celou dobu skladby. V porovnání s do té doby dostupnými daty jde také o mnohem rozmanitější kolekce a v případě MedleyDB jde o první volně dostupný dataset, ve kterém se objevují celé skladby, nikoli pouze výňatky.

Bosch a kol. (2016) pro práci na datasetu ORCHSET definuje melodii jako „jednohlasou sekvenci tónů, kterou bude posluchač nejspíše reprodukovat, pokud jej požádáme o zapískání či zabroukání příslušné skladby“ (na základě článku Poliner a kol. (2007)), pro sestavení kolekce dat proto opravdu využívá skupiny posluchačů, které po poslechu krátkých ukázek orchestrální hudby následně žádá o přezpívání melodie. V případě MedleyDB se na anotacích melodie podílí skupina profesionálních hudebníků a vznikají tři různé přepisy melodie s různě volnými formulacemi definice melodie.

Celkový směr výzkumu je tak ve výsledku velmi podmíněn dostupnými daty. Ta tvoří jakýsi protipól k ryze technickým a objektivním cílům algoritmických metod. Nadějí je, že tato dialektika vývoje algoritmů a práce na datech vyústí jednak v metody extrakce, které věrně zachycují podstatu subjektivního prožitku porozumění hudbě, a jednak v celkovém důsledku také snad v lepší porozumění pojmu melodie obecně.

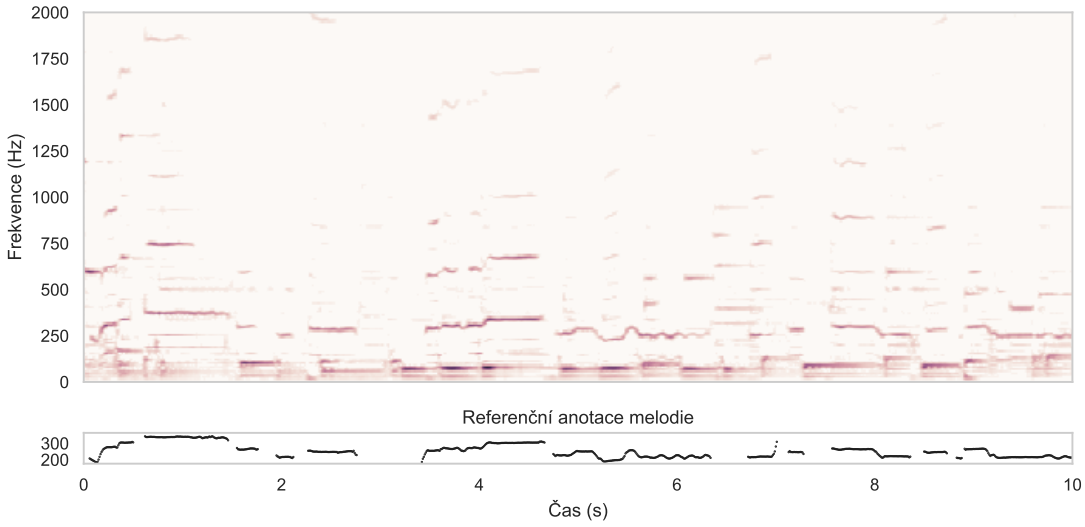
1.3 Metody extrakce melodie

Základním a společným přístupem k problému extrakce melodie je dekompozice na podproblémy odhadu výšek všech znějících hlasů v signálu a následného výběru melodické linie z těchto kandidátních kontur. Jednotlivé metody se pak liší ve způsobech řešení těchto podproblémů, mimo jiné také v míře abstrakce od původního signálu, které při zpracovávání dosahují — zatímco některé přístupy



Obrázek 1.6: Diagram obvyklého návrhu metod pro extrakci melodie.

po celou dobu pracují pouze se signálem jako takovým a důmyslnými způsoby ho transformují tak, aby získaly co nejpřesnější odhad výšky melodie, jiné metody při výpočtu vytváří symbolický popis jednotlivých not a následně i celých frází a melodii pak hledají v tomto vysokoúrovňovém popisu nahrávky. Shrnutí používaných přístupů, blíže popsaných v kapitole Související práce, můžeme vidět na diagramu 1.6.



Obrázek 1.7: Příklad výstupu výpočtu salienční funkce pomocí váženého sčítání harmonických frekvencí. Ačkoli je zpěv velmi zvýrazněn a salienční funkce na většině nahrávky dobře zachycuje výšku znějící melodie, doprovod kolem čtvrté sekundy nahrávky má vyšší hodnotu než zpěv, což neodpovídá lidskému vnímání zpěvu jakožto nejdůležitější složky signálu.

Prvním krokem všech existujících metod pro extrakci melodie je převod zvukového signálu do frekvenční domény, ať už pomocí zmiňované STFT nebo použitím jiných metod vyvinutých pro analýzu harmonického signálu (MRFFT, CQT,

FChT, ...). Jednou z hlavních výhod těchto dalších metod je logaritmická osa frekvence, díky které je snadné pracovat s harmonickými poměry a hudebními intervaly nezávislé na výšce frekvence. Abychom tuto vlastnost ilustrovali, uvedeme příklad — tóny A4 a E5 jsou vzdálené o kvintu (na klavíru od tónu A4 musíme postupně zmáčknout 7 bílých a černých klapek, abychom se dostali k tónu E5), tóny A5 a E6 jsou také vzdálené o kvintu. Rozdíl frekvencí těchto tónů je však 220 Hz mezi první dvojicí a 440 Hz mezi druhou dvojicí, tudíž vzdálenost frekvencí daného intervalu závisí na tónu, od kterého se tento interval počítá. Protože je hudební interval jistý *poměr* mezi dvěma tóny, použitím logaritmické osy frekvence se tyto poměry budou jevit jako absolutní rozdíly ($\log nf_0 = \log n + \log f_0$).

Zpracováním spektrogramu vstupu pak vzniká tzv. *funkce salience*, která každé znějící frekvenci v signálu přiřazuje jisté ohodnocení, které vyjadřuje poměrnou důležitost dané frekvence k ostatním slyšitelným složkám. Funkce salience je tedy v jistém slova smyslu speciální frekvenčně-časová transformace, která podává zejména informace o znějící melodii, nikoli o celkové kompozici signálu. Na obrázku 1.7 můžeme srovnat funkci salience vypočtenou pomocí váženého součtu harmonických frekvencí se vstupním spektrogramem 1.4.

Druhým krokem je pak výběr melodie na základě funkce salience. Triviálním řešením je výběr takových frekvencí, které mají nejvyšší ohodnocení. Problémem tohoto přístupu je však to, že jakmile signál obsahuje více podobně ohodnocených tónů, výstup tohoto řešení má tendenci mezi těmito kandidáty často „přeskakovat“. Algoritmy pro extrakci melodie proto volí různě pokročilé metody vyhlazování, případně metody hledání nejpravděpodobnějšího průchodu posloupností stavů (například pomocí Viterbiho algoritmu).

1.4 Hluboké učení

Motivací pro použití metod strojového učení je překonání limitů člověkem navržených, rigidních, pravidlových systémů. Cílem je automatické nalezení optimálního postupu pro řešení úlohy, na základě množství dat, ve kterých strojové učení dokáže nalézt a využít jejich pravidelností. V našem případě pak po metodě založené na strojovém učení požadujeme, aby na základě příkladů z trénovací množiny vytvořila funkci salience.

Výhodou tohoto přístupu je, že o vstupních datech nemusíme dělat žádné předpoklady. Vzniklá metoda pak může v praxi zohledňovat více faktorů ovlivňujících přítomnost melodie, jako je její barva, frekvenční modulace (vibrato, glissando) nebo hlubší vzájemné srovnání současně znějících tónů. Na základě trénovacích příkladů může být tato metoda robustnější vůči většímu spektru barev hlasů nástrojů — zatímco předešlé metody pro extrakci melodie často uvažují signály s postupně se snižujícím podílem harmonických frekvencí, opravdové signály hudebních nástrojů často tento předpoklad nesplňují (viz obrázek 1.3).

První pokus o využití těchto metod představili Poliner a Ellis (2005), vstupní signál transformovali pomocí krátkodobé Fourierovy transformace a část spektra po jednoduché normalizaci použili jako vstupní data pro metodu podpůrných vektorů (SVM). Jejich metoda měla své limitace, výstup byl kvantizován na úroveň jednoho půltónu a tudíž metoda nedokázala dobře postihnout například vibrata. I přesto však tým dosáhl srovnatelných výsledků s ostatními metodami.

Po roce 2005 jakékoli pokusy o aplikaci strojového učení ustávají a na nové

metody se čeká až do roku 2016, jedním z důvodů byl jistě nedostatek dat, tuto situaci zlepšil například dataset MedleyDB (Bittner a kol., 2014) nebo dnes již zaniklý iKala (Chan a kol., 2015). Zájem o strojové učení však znovu stoupá, také díky úspěšnému využití hlubokých neuronových sítí napříč ostatními obory. Na konferenci ISMIR 2016[1] objevují dva články týmů Kum a kol. (2016) a Rigaud a Radenen (2016), založené právě na hlubokém učení. V roce 2017 publikuje své metody Bittner a kol. (2017) (ISMIR), Balke a kol. (2017) (ICASSP[2]), následující rok přináší metody D Basaran, S Essid (2018) (ISMIR), Bittner a kol. (2018). Všechny zmíněné popisujeme v kapitole Související práce. V oboru lze tedy od roku 2016 vidět velmi výrazný trend právě směrem k hlubokému učení, a stejný směr je patrný i v příbuzných úlohách přepisu hudby. Tým z laboratoře Google Brain dokázal výrazně zlepšit přepis klavírních skladeb pomocí kombinace konvoluční a rekurentní architektury (Hawthorne a kol., 2017). Neuronové sítě také zlepšují výsledky na poli oddělení signálů (Stoller a kol., 2018).

V této práci se pokusíme navázat na zmiňované práce a otestovat nové architektury hlubokých neuronových sítí pro úlohu extrakce melodie, zejména pak pro hledání nových způsobů výpočtu funkce salience, v menší míře také pro detekci melodie.

1.5 Přínosy práce

TODO

1.6 Struktura práce

napsat signpost

[1] International Society for Music Information Retrieval Conference
[2] International Conference on Acoustics, Speech, and Signal Processing

2. Související práce

Pokusy o vytvoření automatické metody pro kompletní transkripci hudby se podle Poliner a kol. (2007) objevují již od sedmdesátých let, z důvodu značné obtížnosti této úlohy, která strmě roste s každým dalším přidaným hlasem ve zkoumaném signálu, však dodnes jedná o otevřený problém. Z tohoto důvodu se od devadesátých let objevují práce, které se pokouší alespoň o částečný automatický popis některých muzikálních aspektů skladeb.

Jednou z prvních je práce týmu Goto a Hayamizu (1999), který se záměrně omezuje na identifikaci jedné, nejhlasitější, spojité křivky fundamentní frekvence hlasu (F0) v omezeném frekvenčním rozsahu. Vzniklé transkripce pak sice nejsou kompletní, na druhou stranu je jejich získání výpočetně nenáročné a přitom poskytují sémanticky bohatý popis nahrávek, který je poměrně často shodný s melodií. Ustanovením úlohy pojmenované jako „Predominant-F0 Estimation“ (PreFEst), byly položeny základy pro vznik navazujících prací a soutěží zabývající se automatickým přepisem melodie.

Největší rozkvět v oboru začal od roku 2004. Uspořádáním první soutěže pro porovnání systémů pro automatický popis hudby v rámci konference ISMIR (ISMIR 2004 Audio Description Contest), se ustanovily priority, formalizovaly podmínky evaluace a byly sestaveny první kolekce dat pro testování algoritmů (Downie a kol. (2010)). Soutěž se v následujícím roku přerodila do samostatné každoroční události, v níž soutěží stále více týmů v rostoucím počtu úloh.

V této kapitole představíme existující metody a společné přístupy k řešení úlohy extrakce melodie. Nejprve úlohu dekomponujeme na podúlohy a pro tyto podúlohy uvedeme příklady přístupů existujících metod. V závěru kapitoly pak provádíme kvantitativní srovnání existujících metod na základě dat ze soutěže MIREX pro výběr replikovaných metod v této práci.

2.1 Průzkum existujících metod

Jen do soutěže MIREX se od roku 2005 přihlásilo 45 týmů s 62 různými metodami pro extrakci melodie, s různou mírou přesnosti přepisu. Mezi přístupy k tomuto problému tedy existuje veliká rozmanitost, jejíž kompletní popis přesahuje rámec této práce. Zaměříme se proto na společné rysy a celkové trendy v oboru.

Shrnující práce od Poliner a kol. (2007) a Salamon a kol. (2014) se při charakterizaci systémů pro transkripci odkazují na příbuznou úlohu odhadu fundamentální frekvence monofonní nahrávky. Algoritmy pro monofonní tracking na základě vstupního signálu $x_{mono}(t)$ počítají *funkci salience* $S_{x_{mono}}(f_\tau, \tau)$ pro každý krátký časový okamžik (okno) τ a frekvenci f_τ. Výsledkem této funkce je relativní ohodnocení (příp. pravděpodobnost) jednotlivých frekvencí obsažených ve vstupním signálu, které značí, zda-li je daná frekvence fundamentální frekvencí znějícího hlasu.

Pro zvýšení robustnosti vůči šumu, přeslechu, dozvuku a jiným vlivům, které zhoršují kvalitu odhadu salience, se využívá také spojitosti fundamentální frekvence. Pro zajištění kontinuity extrahovaných frekvenčních kontur se zohledňuje také faktor temporálních závislostí $C(\mathbf{f})$, jejímž vstupem je kandidátní kontura $\mathbf{f}$ a výstupem je ohodnocení této celé kontury na její spojitost. Například tato

funkce může penalizovat odhady, ve kterých výstupní F0 často přeskakuje o oktávu, což je u skutečného signálu nepravděpodobné a naopak jde o častou chybu při výpočtu salienční funkce signálů se silnými sudými harmonickými frekvencemi.

Výstupem monofonního trackingu je posloupnost frekvencí s maximální saliencí a spojitostí, tedy posloupnost frekvencí, které jsou nejlépe ohodnocenými kandidáty na fundamentální frekvenci a zároveň tato celá sekvence má také vysoké ohodnocení spojitosti.

$$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmax}} [\sum_{\tau} S_{x_{mono}}(f_\tau, \tau) + C(\mathbf{f})]$$

Přejdeme-li k úloze extrakce melodie, obecně se vstupní polyfonní signál $x(t) = x_m(t) + x_d(t)$ skládá ze směsi melodického hlasu $x_m(t)$ a hudebního doprovodu $x_d(t)$, cílem metod pro extrakci je z pohledu přepisu fundamentální frekvence zvýšení robustnosti algoritmu vůči mnohem výraznějšímu druhu šumu - hudebnímu doprovodu $x_d(t)$. Výstupem našeho systému tedy bude posloupnost odhadů frekvence v každém časovém okně vstupního signálu, reprezentovaná vektorem $\hat{\mathbf{f}}$:

$$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmax}} [\sum_{\tau} S'_x(f_\tau, \tau) + C'(\mathbf{f})]$$

kde f_τ je frekvence na pozici τ ve vektoru $\mathbf{f}$. $S'_x(f_\tau, \tau)$ je upravená funkce salience, která při výpočtu zohledňuje vliv doprovodu a složka $C'(\mathbf{f})$ představuje ohodnocení celého průběhu melodie.

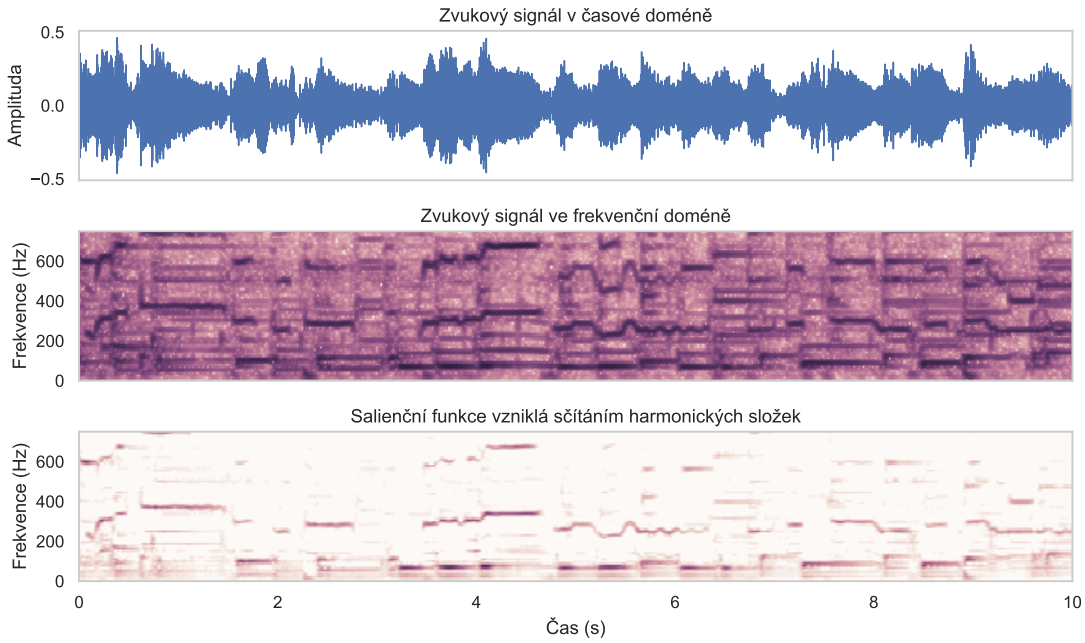
Spolu s odhadem frekvencí by také měl systém na výstupu určit úseky, ve kterých v nahrávce melodie zní a kdy nikoli. K výstupu tedy patří také vektor $\hat{\mathbf{v}}$, se stejným počtem složek jako $\hat{\mathbf{f}}$, který indikuje přítomnost melodie v každém časovém okně τ.

Většina existujících metod sdílí podobnou základní strukturu při řešení extrakce, která se zakládá na popsané formalizaci. Prvním krokem je transformace zvuku do frekvenční domény a následný odhad znějících výšek tónů v polyfonním signálu (výpočet funkce salience), druhým krokem je pak zpracování těchto odhadů a výběr melodie (tedy zpřesnění výsledné $\hat{\mathbf{f}}$ pomocí $C'(\mathbf{f})$). Přístupy k řešení těchto dvou kroků již s konkrétními příklady nastíníme v dalších sekcích.

2.1.1 Spektrální analýza

Zvuk hraného tónu na melodickém nástroji je z fyzikálního pohledu periodická změna tlaku vzduchu. Perioda tohoto signálu se nazývá fundamentální frekvence (označujeme F0) a zpravidla je tento signál složen ze součtu řady sinusoid, jejichž frekvence jsou celočíselným násobkem fundamentální frekvence. V čase měnící se amplitudy těchto *harmonických frekvencí* udávají hlasitost a barvu hlasu, výška první harmonické frekvence (tj. výška fundamentální frekvence) pak ve většině případů odpovídá posluchačem vnímané výšce tónu.

Prvním krokem metod pracujících s hudebním signálem je proto provedení spektrální analýzy, jde o převod zvuku do frekvenční reprezentace, která odhaluje tyto harmonické struktury tónů a umožňuje s nimi dále pracovat.



Obrázek 2.1: Znázornění kroků spektrální analýzy a výpočtu funkce salience.

Krátkodobá Fourierova transformace

Přístupů ke spektrální analýze je více, přímočará a podle Dressler (2016) nejčastěji používaná metoda je *krátkodobá Fourierova transformace* (STFT). Jejím principem je rozdělení vstupního signálu na množinu překrývajících se oken konstantní délky a výpočet Fourierovy transformace těchto krátkých zvukových úseků. Komplexní výsledek transformace umocníme a získáme tzv. výkonové spektrum signálu, které obsahuje informaci o poměrech energie frekvencí, ze kterých se signál v okně skládá. Spektrogram $X(f, \tau)$ vypočteme v čase τ a frekvenční složce f jako:

$$X(f,\tau) = \left| \sum_{n=-\infty}^{\infty} x(n) w(n-\tau) e^{-2\pi i \cdot n \cdot f} \right|^2$$

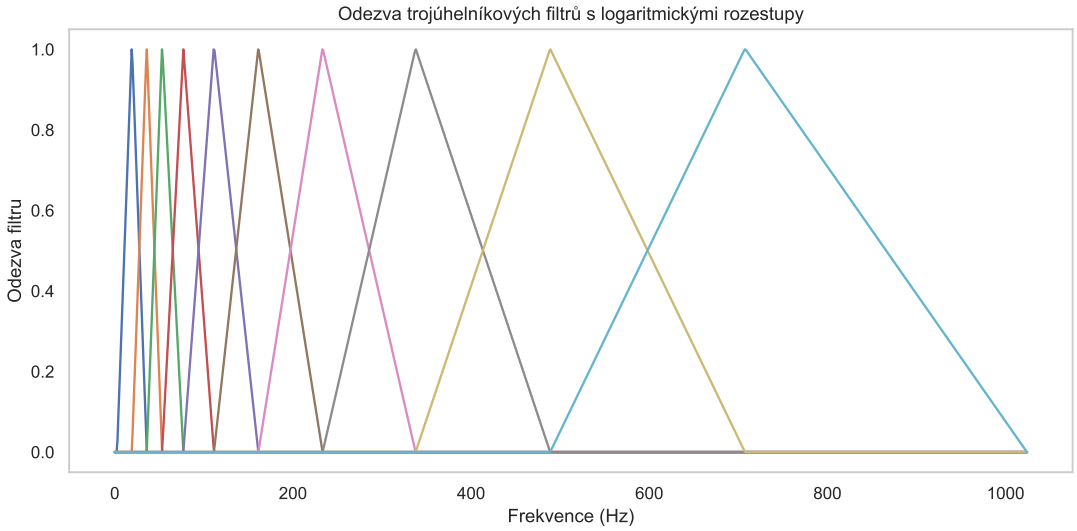
Pro diskrétní vstupní signál $x(n)$ a okénkovou funkci $w(n)$. V této práci budeme pro výpočty spektrogramů používat Hannovo okno, které omezuje tzv. prosakování ve spektru.

$$w_{hann}(n) = \begin{cases} \cos^2 \frac{\pi n}{N}, & \text{pro } -\frac{N}{2} \leq n \leq \frac{N}{2}, \\ 0, & \text{jinak.} \end{cases}$$

Kde N je požadovaná velikost okna STFT tranformace (kvůli obvyklé praxi výpočtu STFT pomocí algoritmu rychlé Fourierovy transformace (FFT) se hodnota N volí z $N \in \{512, 1024, 2048, \dots\}$).

Logaritmická osa spektrogramu

Výsledkem STFT je rozložení signálu na jednoduché frekvenční složky (sinusoidy) s konstantně vzdálenými frekvencemi. Jinými slovy frekvenční osa spektrogramu vytvořeném metodou STFT je lineární. Jak jsme již nastínili v úvodu,



Obrázek 2.2: Příklad trojúhelníkových filtrů pro transformaci frekvenční domény na logaritmickou škálu.

povaha hudebních intervalů a harmonických struktur tónů spočívá v tom, že téměř všechny periodické signály se v hudební skladbě vyskytují ve vzájemných poměrech (v případě intervalů v poměrech $2^{\frac{n}{12}}$ a v případě harmonických frekvencí v celočíselných). K tónu, jehož základní frekvence je rovna 440 Hz, patří také harmonické složky s frekvencemi 880 Hz, 1320 Hz, ..., tedy absolutní vzdálenosti na spektrogramu STFT mezi frekvencemi harmonických složek jsou závislé na výšce základní frekvence. Toto způsobuje obtíže při analýze signálů, jelikož všechny uvažované frekvenční rozdíly jsou relativní. Častým druhem zpracování STFT je proto převod frekvenční osy na logaritmickou, na té pak platí pro vzdálenost libovolné fundamentální frekvence f_0 a její libovolné h-té harmonické frekvence $f_h = f_0 \cdot h$:

$$\log f_h - \log f_0 = \log (h \cdot f_0) - \log f_0 = \log h + \log f_0 - \log f_0 = \log h = \text{const.}$$

Tedy absolutní vzdálenosti uvnitř harmonické struktury tónů na spektrogramu s logaritmickou osou frekvence zůstávají konstantní nezávisle na výšce fundamentální frekvence. Tento přepočet se obvykle provádí pomocí banky filtrů s trojúhelníkovou odezvou, pro transformovaný signál $X(f, \tau)$ s N frekvenčními složkami spočteme nový spektrogram s logaritmickou osou následovně:

$$X_{\log}(\omega, \tau) = \sum_{f=-\infty}^{\infty} g_\omega(f) X(f, \tau)$$

Přičemž $g_\omega(f)$ značí odezvu trojúhelníkového filtru pro výstupní frekvenční složku ω.

Multirezoluční transformace

Na libovolnou metodu převodu diskretizovaného signálu na frekvenční doménu se vztahuje Gaborův limit, který popisuje závislost přesnosti lokalizace signálu ve frekvenční a časové doméně (Gabor a Member, 1945). Volbou délky vstupního

okna transformace zpřesňujeme buď frekvenční nebo časové rozlišení výsledné spektrální reprezentace. Zvolíme-li krátké vstupní okno, zvyšujeme časové rozlišení (krátké okno lépe zachycuje rychlé změny průběhu signálu), avšak ztrácíme přesnost na frekvenční ose, opačný vztah platí pro volbu delšího okna.

Tato limitace je markatní zejména pokud STFT používáme pro hudební data. , u vyšších tónů jsou proto vzdálenosti mezi frekvencemi signálů větší než u nižších tónů. Frekvenční rozlišení STFT je však konstantní na celém výstupním frekvenčním rozsahu a volba velikosti okna transformace zajistí dobrý poměr frekvenčního a časového rozlišení jen pro část rozsahu. Ve výsledku je pak buď pro vyšší frekvence okno příliš velké (zbytečně detailní frekvenční rozlišení na úkor časového rozlišení) a nebo naopak pro nižší frekvence je okno nedostačující (rozlišení frekvence nemusí být pro basy ani na úrovni půltónů).

Z tohoto důvodu existují vedle STFT i další metody, jejichž cílem je nabídnout lepší kompromis frekvenčně-časového rozlišení v kontextu melodických dat. Goto a Hayamizu (1999) používají MRFFT (Multi-Resolution Fast Fourier Transform), principem je opakovaný downsampling signálu (převzorkování na nižší vzorkovací frekvenci) a aplikace Fourierovy transformace na každý vzniklý signál; s každou iterací spektrum obsahuje čím dál podrobnější informace o nižších frekvencích, protože vyšší frekvence se při downsamplingu ztratí. Brown (1990) popsala metodu Constant-Q Transform (CQT), která spočívá v použití proměnné délky okna Fourierovy transformace pro výstupní frekvenční pásma, která rovnoměrně pokrývají logaritmickou osu frekvence. Cancela a kol. (2010) kombinuje CQT a Chirp Z-transform, jedná se o zobecnění diskrétní Fourierovy transformace, ve které je signál rozložen na tzv. lineární čerpy — sinusoidy s proměnnou frekvencí. Díky tomu transformace dokáže s lepším rozlišením zachytit signály, které rychle mění výšku tónu, v hudbě například vibrato. Paiva a kol. (2004) napodobují mechanismy lidského sluchu pomocí banky pásmových filtrů (Cochleagram) s logaritmicky rozmístěnými mezními frekvencemi a sumy autokorelací na jednotlivých frekvenčních pásmech signálu (Summary correlogram).

I přes uvedené důvody se Salamon a kol. (2014) a Dressler (2016) domnívají, že metoda zpracování signálu příliš neovlivňuje výslednou přesnost algoritmů pro přepis melodie. Tvrzení dokládají jednak celkovým srovnáním výsledků metod ze všech ročníků soutěže MIREX a jednak neochvějnou převahou využití krátkodobé Fourierovy transformace, jakožto efektivní a dostačující metody pro spektrální analýzu.

Postprocessing spektrogramu

Po převodu signálu na frekvenční reprezentaci následuje u většiny metod některý druh úpravy celého spektrogramu, předcházející samotnému výpočtu *funkce salience*. Výsledkem tohoto kroku může být potlačení šumu a nemelodických částí signálu, zpřesnění informace o výšce znějících frekvencí nebo normalizace či jiná úprava amplitud spektrogramu.

Nejčastější úpravou je nalezení lokálních maxim (vrcholů). Potlačením nemaximálních oblastí se zbavíme velkého množství nemelodických složek signálu, přitom informaci o těch melodických neztratíme. Výhodou práce s množinou vrcholů je, že jejich frekvenci lze na základě spektra dále zpřesnit pomocí parabolické interpolace (Rao a Rao (2010)) a nebo využitím úhlové frekvence (Salamon a Gomez (2012), Dressler (2009)).

Jiným druhem úpravy jsou různé způsoby normalizace, ať jednoduché aplikace logaritmu na jednotlivé hodnoty spektrogramu (Cancela (2008), Bittner a kol. (2017)) nebo složitější strategie normalizace, které se aplikují na celá výstupní okna krátkodobé Fourierovy transformace (spectral whitening, Ryynänen a Klapuri (2008)), či které používají pohyblivé průměry nebo jinou metodu, beroucí v potaz širší zvukový kontext. Cílem normalizace je zvýraznění slabších harmonických frekvencí a potlačení celopásmových zvuků (například perkusí). Principiálně podobným krokem je aplikace pásmového filtru (Goto a Hayamizu (1999)) pro zvýraznění frekvencí obsahující melodii. Případně využití psychoakustických filtrů modelující lidské vnímání hlasitosti (Salamon a Gomez (2012), Ikemiya a kol. (2016)). Ze signálu lze také oddělit melodické nástroje a perkusivní doprovod pomocí metod separace signálů (source separation). Používanými metodami jsou například Harmonic/Percussive Sound Separation (HPSS) (Tachibana a kol. (2010)) nebo Robust principal component analysis (RPCA) (Ikemiya a kol. (2016)).

2.1.2 Funkce salience

Salience tónu vyjadřuje míru důležitosti či nápadnosti ke svému hudebnímu okolí. Nejvíce ji ovlivňuje hlasitost v poměru ke zbylým znějícím hlasům, vliv má ale také řada dalších charakteristik hraní. Dressler (2016) mezi příklady uvádí například frekvenční modulaci, jako je vibrato nebo glissando, zejména oproti frekvenčně stálému hudebnímu doprovodu (piano, kytara). Velký vliv má samozřejmě také i barva hlasu. Lidský zpěv nebo obecně zvuky se silnějšími vyššími alikvótními frekvencemi snadněji upoutají pozornost. V případě vícehlasu mají obecně posluchači potíže rozeznat výšky tónů uvnitř souzvuku. Pokud má posluchač přiřadit pociťovanou výšku tónu akordu, obvykle volí nejvyšší či nejnižší ze znějících frekvencí.

Výstupem funkce salience je ohodnocení každé výšky tónu v každém časovém okamžiku nahrávky, které co nejlépe odpovídá v relativních poměrech výše popsané zvukové salienci. Jelikož neexistují žádné studie, které by se zabývaly měřením a kvantifikací salience v hudbě, nelze posoudit, jak dobře výsledky obvyklých způsobů výpočtu funkce salience korelují s mírou, ze které vychází. Bittner (2018) se však domnívá, že odhad bude velmi hrubý, většina postupů totiž do výpočtu zahrnuje pouze hlasitost hlasu, a tedy vynechává řadu jiných důležitých faktorů, které salienci ovlivňují.

Přístupy k výpočtu by se daly zařadit do tří kategorií - sčítání harmonických frekvencí, odhad parametrů modelujících vstup a metody strojového učení.

Sčítání harmonických frekvencí

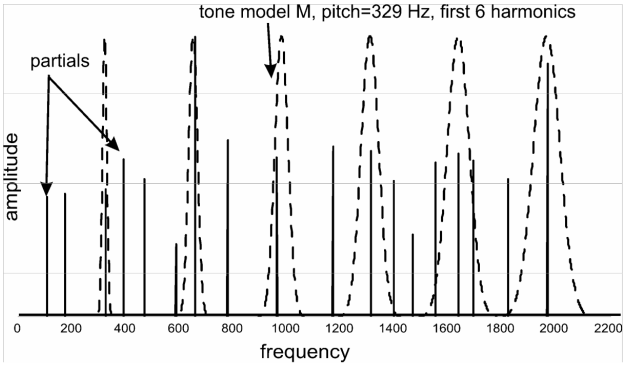
Metody založené na sčítání harmonických frekvencí jsou principiálně nejjednodušší skupinou. Vychází z práce Hermes (1988), jejich podstatou je využití harmonické struktury zvuku tónů. Ohodnocení frekvence f_τ získáme váženou sumou amplitud všech jejích harmonických frekvencí $h \cdot f_\tau$. Pro spektrogram $X(f, \tau)$ signálu x, funkci vah $g(f_\tau, h)$ a N_h počet zahrnutých harmonických frekvencí v sumě:

$$S_x(f_\tau, \tau) = \sum_{h=1}^{N_h} g(f_\tau, h) \, |X(h \cdot f, \tau)|$$

Pro ilustraci uvažme jednohlasý harmonický signál s fundamentální frekvencí $f^\star$. Hodnoty spektrogramu $X(f, \tau)$ tedy budou vyšší kolem frekvencí $H_{f^\star} = \{1 \cdot f^\star, 2 \cdot f^\star, 3 \cdot f^\star, \dots\}$ a jinde nulové. Funkce $S_x(f, \tau)$ bude tedy pro $f \notin H_{f^\star}$ nulová a v $f^\star$ bude nabývat globálního maxima. Příklad výstupu této metody pro polyfonní nahrávku je na obrázku 2.1.

Dressler (2011) tuto metodu vylepšuje zpracováváním dvojic vrcholů vstupního spektrogramu, její výsledný salienční spektrogram obsahuje méně kandidátů na fundamentální frekvenci. Cancela (2008) se pokouší zmenšovat chybné hodnoty salienční funkce pomocí vyhodnocení subharmonických frekvencí.

Statistické modelování signálu



Obrázek 2.3: Ilustrace modelu tónu spolu se signálem

Jiným přístupem k počítání funkce salience, který používá Goto a Hayamizu (1999), je modelování okna spektrogramu váženým součtem harmonických struktur (modelů tónů). Snažíme se vrcholy ve spektru rozdělit mezi různě silně znějící tóny tak, aby v součtu co nejlépe odpovídaly měřeným intenzitám. Přístup se jinými slovy snaží zjistit, jaké tóny musely v danou chvíli znít, aby vzniklo dané spektrum.

Vstupem metody je okno normalizovaného spektrogramu $p_X^{(t)}(x)$ v čase t. Okno se pokusíme modelovat jako hustotu pravděpodobnosti $p(x; \theta^{(t)})$ vzniklou váženou směsí modelů všech možných tónů melodie v definovaném rozsahu frekvencí v intervalu $[F_l, F_h]$. Hustotu pravděpodobnosti jednoho z tónů s fundamentální frekvencí F označíme jako $p(x|F)$ (obrázek 2.3 ukazuje jeden z možných modelů tónu), a jako $w^{(t)}(F)$ označíme váhu, kterou model tónu $p(x|F)$ přispívá do celkové smíšené hustoty pravděpodobnosti. Pak $p(x; \theta^{(t)})$ definujeme jako:

$$p(x; \theta^{(t)}) = \int_{F_l}^{F_h} w^{(t)}(F) p(x|F) \mathrm{d}F$$

$$\theta^{(t)} = \{\, w^{(t)}(F) \mid F_l < F < F_h \,\}$$

Cílem pak je nalezení takových parametrů $\theta^{(t)}$, aby model $p(x; \theta^{(t)})$ dobře popisoval pozorovaní $p_X^{(t)}(x)$. K tomu Goto a Hayamizu (1999) využívá Expectation-Maximization (EM) algoritmus. Výsledné parametry $\theta^{(t)}$ jsou pak hodnoty salienční funkce.

TODO: popsat Durrieu, jakožto jiný přístup k modelování signálu

Metody strojového učení

K výpočtu funkce salience můžeme využít také metody strojového učení. První práci využívající těchto metod představují Poliner a Ellis (2005), kteří úlohu formulují jako klasifikační. Vstupní okno signálu jejich metoda klasifikuje do jednotlivých tříd tónů melodie, výstup je tedy kvantizován na rozlišení jednoho půltónu. Pro tento účel využívají metodu podpůrných vektorů (SVM) přičemž vstupními příznaky je část spektrogramu signálu, konkrétně vektor s 256 složkami, který získávají pomocí krátkodobé Fourierovy transformace podvzorkovaného signálu. Metoda dosahovala průměrných výsledků v soutěžích MIREX 2005 a 2006. Strojové učení se v oboru příliš neuchytilo, z důvodu nedostupnosti dat, ale možná také i kvůli limitacím, které Poliner a Ellis (2005) ve své práci prezentují.

V roce 2016 na tyto nedostatky odpovídá práce Kum a kol. (2016). Augmentací dat a zaměřením se na odhad výšky zpěvu místo melodie úspěšně překovávají problém nedostatku dat. Místo klasifikátoru SVM používají hluboké neuronové sítě (3 skryté plně propojené vrstvy) a problém kvantizace na úroveň půltónu řeší natrénováním tří nezávislých sítí s různě jemným rozlišením výstupu, které pak spojí v jeden výstup. V jejich případě platí, že jemná síť má sice lepší výstupní rozlišení, celkově má ale horší přesnost, naopak je tomu u sítě s hrubým rozlišením, proto jejich výsledky kombinují a dosahují tehdejších state-of-the-art výsledků na vokálních datech. Podobný postup v tomtéž roce podniká i tým Rigaud a Radenen (2016), podobně jako v práci Kum a kol. (2016) se tým omezuje na odhad výšky zpěvu, používá augmentaci dat, přechází na hlubokou neuronovu síť (2 skryté plně propojené vrstvy) a používá jemnější rozlišení výsledného vektoru pravděpodobnostního rozdělení znějící výšky. Rozdílem je ale předzpracování dat pomocí dekompozice na harmonické a perkusivní složky pomocí HPSS. Práce Balke a kol. (2017) používá neuronovou síť s jednou skrytou vrstvou pro anotaci jazzových sól ze spektrogramu s logaritmickou osou frekvence.

Bittner a kol. (2017) představuje první pokus o použití hlubokých sítí na úlohu extrakce melodie bez zaměření na zpěv. Úlohu formuluje jako odstranění šumu obrázku, cílem je ze vstupního spektrogramu pomocí hluboké sítě s konvolučními vrstvami vytvořit salienční funkci. Vstupní a výstupní data mají tedy stejné měřítko, výstup však v ideálním případě obsahuje pouze informace o fundamentálních frekvencích znějící melodie. Velmi přínosný je také popis vstupní spektrální reprezentace HCQT, která spočívá ve výpočtu několika CQT spektrogramů (kanálů), jejichž počáteční frekvence jsou vzdálené v harmonických poměrech, tudíž (protože CQT spektrogram používá logaritmickou osu frekvence) všechny související harmonické složky na celém spektru jsou na těchto spektrogramech zarovnané nad sebou v ose kanálů. Tento koncept je hlouběji popsán v kapitole Experimenty v sekci HCNN, kde tento nápad aplikujeme nejen na vstupní spektrogram, ale nově také na propojení celé neuronové sítě. Dalším důležitým přínosem práce je úprava reprezentace cílové salienční funkce pomocí rozostření hranic cílové výšky tónu. Zatímco předchozí metody trénovali sítě jako diskrétní klasifikaci s jednou správnou výstupní třídou, v této práci je cíl trénování gaussián se střední hodnotou uprostřed výšky tónu.

Práce Bittner a kol. (2017) dokázala překonat state-of-the-art metody pouhým výpočtem salienční funkce. Metoda tedy úplně přeskakuje vyhlazování odhadů v čase a používá pouze nejzákladnější metody pro detekci melodie pomocí prahování. Zároveň však její salienční funkce pro odhad jednoho okna délky $\approx 11\,\text{ms}$

zpracovává přibližně 150 ms vstupního okna, tedy v porovnání s existujícími metodami její salienční funkce zpracovává velmi široký kontext, proto není vyhlazování jejich výsledků nezbytné.

D Basaran, S Essid (2018) na této práci buduje a jednak prozkoumává možnosti použití jiné vstupní spektrální reprezentace (založené na práci Durrieu a kol. (2011)) a jednak do architektury sítě zabudovává rekurentní neuronové sítě, které zajišťují zmiňované vyhlazování odhadů. Jeho metoda překonává výsledky Bittner a kol. (2017), nevýhodou jeho přístupu je však hrubý výstup s rozlišením na půltóny.

V rámci soutěže MIREX také přibývají od roku 2016 nové metody založené na hlubokých sítích, ty se však buď stále zaměřují pouze na extrakci zpěvu (například Su (2018)) případně k nim nelze dohledat související článek (v roce 2016 metody účastníka Zhe-Cheng Fan a v roce 2018 metoda od týmu Sanguen Kum, Juhan Nam).

2.1.3 Hledání melodie

Po výpočtu funkce salience $S_x(f_\tau, \tau)$ máme k dispozici odhady fundamentálních frekvencí v signálu, z těchto ohodnocení pak musíme vybrat výslednou konturu melodie. V závislosti na způsobu výpočtu funkce salience tyto ohodnocení frekvencí více či méně odpovídají jejich důležitosti v signálu. Triviálním řešením zpracování těchto hodnot by bylo vybrat frekvence s maximální saliencí pro každé časové okno $\hat{f}_\tau = \text{argmax}_{f_\tau} S_x(f_\tau, \tau)$, tento jednoduchý přístup však nevolí mnoho metod, protože jeho výstup má u složitějších skladeb tendenci „přeskakovat“ mezi doprovodem a melodií.

Obecně se dají přístupy rozdělit na pravidlové metody a statistické metody, dále se pak metody liší v tom, zda na základě salienční funkce vytváří abstraktnější popisy obsahu - ať už na úrovni jednotlivých celistvých konturů, tónů nebo celých frází.

Goto a Hayamizu (1999) pro sledování melodie používá množinu „agentů“, pohybujících se v čase po výstupu salienční funkce a na základě předem definovaných pravidel jejich pohyb zaručuje kontinuitu výstupní fundamentální frekvence. Podobné sledování kontur v čase na základě salienční funkce využívá i Dressler (2009). Jinou pravidlovou metodou je opakované nalezení globálního maxima, jeho iterativní prodlužování v obou směrech časové osy a následné vymazání této nalezené kontury z výstupu salienční funkce, čímž dovolíme nalezení nového globálního maxima (Cancela (2008), Salamon a Gomez (2012)). Z extrahovaných kontur následně můžeme vybrat ty, které splňují kritéria pro melodické kontury.

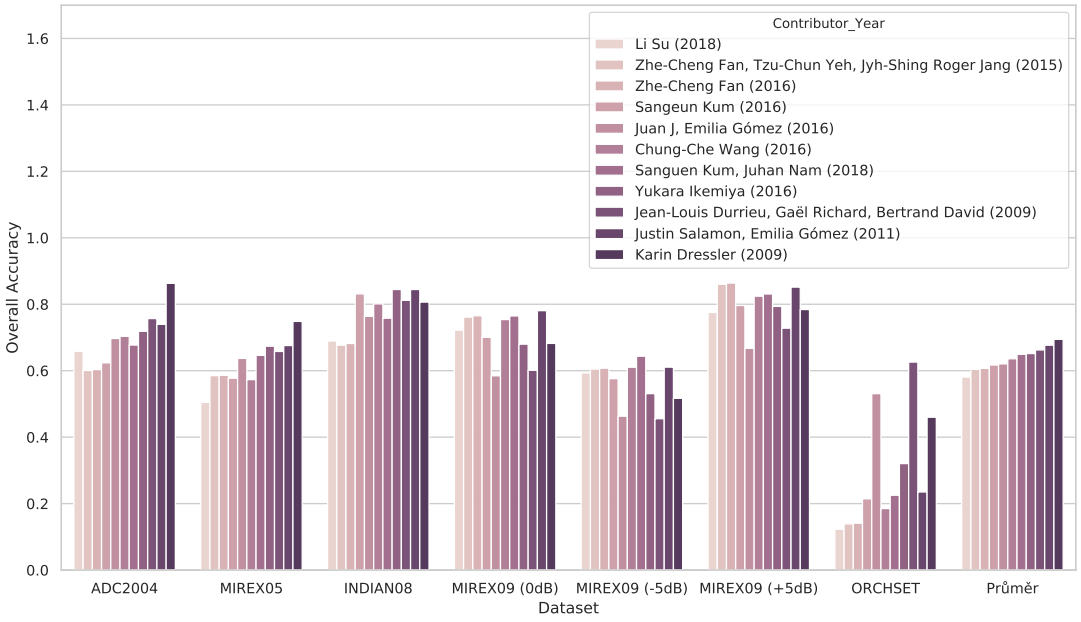
Jiným přístupem k hledání melodie je použití statistických metod, jako je například modelování pomocí skrytých Markovových modelů. Na salienční funkci tyto metody pohlíží jako na sérii pozorování a pomocí hledání nejpravděpodobnější cesty skrz stavy modelu s vhodně nastavenými či z dat získanými pravděpodobnostmi přechodu získávají vyhlazenou konturu melodie. Tyto modely mohou být velmi komplexní a můžou zahrnovat modely průběhu not (Ryynänen a Klapuri, 2008) nebo naopak velmi minimalistické, zahrnující pouze stavy pro jednotlivé tóny (Yeh a kol. (2012))

Přítomnost melodie (voicing)

Důležitou součástí algoritmů pro extrakci melodie je detekce melodie v signálu. Většina metod tento krok provádí na konci vyhodnocování pomocí pevně nastaveného či dynamického prahování, jiné metody detekci melodie řeší filtrováním melodických kontur (Salamon a Gomez (2012)). V případě statistických metod je stav neznějící melodie často přímo zabudován do statistického modelu (Ryynänen a Klapuri, 2008).

2.2 Srovnání existujících metod

Pro celkové kvantitativní srovnání metod jsme zpracovali výsledky všech ročníků soutěže MIREX. V této sekci prezentujeme shrnutí metod, pro které existují výsledky na všech dostupných datasetech, tedy vybíráme převážně z metod od roku 2015. Díky práci Bosch a Gómez (2014), který starší metody spustil na svém datasetu ORCHSET, můžeme ke srovnání přidat také výsledky metod Dressler (2009), Salamon a Gomez (2012) a Durrieu a David (2010). Celkové srovnání nalezneme v tabulce 2.4. Na základě těchto výsledků vybíráme metodu Salamon a Gomez (2012) jakožto zástupce metod, které nejsou založeny na strojovém učení[1].

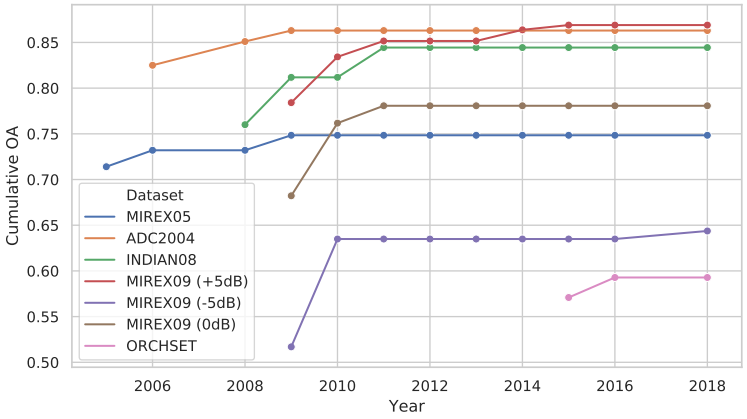


Obrázek 2.4: Výsledky metod v soutěži MIREX v letech 2015-2018 s vybranými metodami ze starších ročníků.

Na základě grafu 2.4 také vidíme, že největší variabilitu mají výsledky na datasetu ORCHSET, zde mají metody velký prostor pro zlepšení, naopak u datasetů INDIAN08 a variant MIREX09 je zřejmá jistá hranice kterou je pro metody obtížné překonat. Salamon a kol. (2014) ve svém přehledovém článku dochází k závěru, že vývoj metod extrakce melodie začal od roku 2009 stagnovat, na obrázku 2.5 znázorňujeme maximální dosaženou celkovou přesnost metod na jednotlivých datasetech od počátku soutěže MIREX. Bohužel musíme konstatovat,

[1] metoda Dressler (2009) nemá zveřejněnou implementaci.

že v rámci soutěže MIREX stagnace pokračuje doposud, přitom výzkum metod stále pokračuje a zejména díky strojovému učení se obor posouvá. V rámci MIREXu však nebyly vyhodnoceny nové stěžejní metody Bittner a kol. (2017) a D Basaran, S Essid (2018) a zájem o soutěž v této kategorii postupně upadá (v roce 2017 nesoutěžily žádné týmy, v roce 2018 pouze dva). Důvodem může být právě nedostatečný prostor pro zlepšení kvůli nedostatku nových, zajímavých dat.



Obrázek 2.5: Stagnující vývoj metod pro extrakci melodie.

2.2.1 Replikace výsledků

Pro srovnání metod představovaných v této práci spouštíme metody Salamon a Gomez (2012), Bittner a kol. (2017) a D Basaran, S Essid (2018) na testovacích množinách. Všechny tři metody mají volně dostupnou implementaci, první ve formě VAMP plug-inu[2], zbylé jsou implementovány v jazyce Python a používají standardní knihovny určené pro hluboké učení[3][4]. Výsledky těchto metod uvádíme v kapitole Výsledky.

Poznamenáme, že implementace algoritmu Salamon a Gomez (2012) existují dvě, druhá v rámci knihovny Essentia[5], tato implementace však v porovnání s implementací VAMP podávala výrazně horší výsledky napříč datasety. V kapitole Výsledky proto používáme implementaci VAMP.

Pokusili jsme se také o replikaci výsledků Bosch a Gómez (2014), bohužel se nám ale kvůli nekompatibilitě mezi verzemi knihoven nepodařilo tuto 5 let starou metodu spustit. Teprve nedávno tuto chybu autoři odstranili[6], výsledky již ale do práce nezahrnujeme z časových důvodů. je ok přiznat, že nestíhám?

Pro srovnání jsme se také pokusili spustit metodu Durrieu a David (2010), délka běhu algoritmu je však při zachování výchozího nastavení parametrů neúnosně dlouhá, 23 minut dat (ORCHSET) nám trvalo zpracovat dva dny. Tudíž její spuštění na větší korpusy (MedleyDB, WJazzD) je mimo možnosti autora práce. tohle je asi blbě formulované?

[2] `https://www.upf.edu/web/mtg/melodia`

[3] `https://github.com/rabitt/ismir2017-deepsalience/`

[4] `https://github.com/dogacbasaran/ismir2018_dominant_melody_estimation`

[5] https://essentia.upf.edu/documentation/

[6] `https://github.com/juanjobosch/SourceFilterContoursMelody/commit/6f88e709c470f1423dc429198cb3c261a772c66c`

3. Datasety

Nedostupnost dostatečného množství dat pro automatickou transkripci melodie představuje zejména pro metody strojového učení otevřený problém. Zatímco pro vzdáleně příbuznou úlohu automatického přepisu mluveného slova existuje tisíce hodin nahrávek (například dataset LibriSpeech, který vznikl na základě audioknih), největší dataset s přepsanou melodickou linkou MedleyDB má celkovou délku pod šest hodin. Do roku 2014, kdy MedleyDB vznikl, existovaly datasety, které byly buď rozmanité, ale příliš krátké (ADC04, MIREX05, INDIAN08) nebo naopak celkově větší, ale žánrově a hudebně homogenní (MIREX09, MIR1K, RWC). V roce 2015 byl vydán dataset Orchset, který obsahuje 23 minut výňatků z orchestrálních skladeb různých období. Za dataset pro extrakci melodie se také dá považovat Weimar Jazz Database, který je sice primárně zaměřený na využití v muzikologii, nicméně obsahuje přes 450 přepsaných jazzových sól. Novinkou z roku 2017 je vydání datasetu MDB-melody-synth, který byl automaticky vygenerován základě vstupní vícestopé hudby (převzaté z MedleyDB), existuje tedy naděje, že současný korpus pro přepis melodie by se mohl v budoucnu rozšířit o velkou část automaticky přesyntetizovaných, veřejně dostupných vícestopých nahrávek.

Co se týče blízké úlohy transkripce hudby, velikost největších datasetů se pohybuje v řádu desítek hodin, tudíž jde stále o omezené kolekce. Mezi největší se řadí MusicNet (orchestrální, 34 hodin), MAPS (klavír, 18 hodin), MDB-mf0-synth (multižánrový, 4,7 hodin), GuitarSet (kytara, 3 hodiny) a URMP (komorní orchestr, 1,3 hodiny). I když jde o úlohu, která je lépe definovaná (na rozdíl od extrakce melodie zde nehraje roli subjektivita volby hlavního hlasu), s použitím polyfonních nástrojů vyvstává problém náročné ruční anotace.

Vytváření nových datasetů je obecně velmi pracné a nákladné. Obvyklý postup totiž zahrnuje buď kompletní ruční přepis nahrávky nebo alespoň ruční opravu výstupu automatického přepisu, přičemž tuto práci odvedou kvalitně pouze zaškolení hudebníci. Každá vzniklá anotace se také musí překontrolovat, a to nejlépe jiným hudebníkem. Dalším problémem je vůbec identifikace melodie - jelikož je určení hlavní melodické linie subjektivní, musí se na výsledné anotaci shodnout co nejvíce posluchačů. Ve výsledku se proto do datasetů buď vybírají takové nahrávky, které nejsou sporné, nebo na každé anotaci pracuje celý tým, který melodii společně určí. S tím souvisí také zavedení a pečlivé dodržování anotační politky u komplexnějších skladeb (například orchestrálních), kde může melodii nést více hlasů zároveň současně či střídajíc se. Také množství výchozích dat pro vznik datasetů není velké. Jednak musí být skladby šiřitelné, pokud má být dataset volně dostupný a jednak by k nim měly být dostupné *audio stopy* (nahrávky samostatných hlasů), ze kterých je smíchán finální mix, jelikož ruční anotace finálního mixu je mnohem náročnější než anotace oddělených stop.

Existence dostatenčně velkých datasetů je obecně vzato zásadním předpokladem pro využití metod strojového učení pomocí hlubokých neuronových sítí, zejména pak pro netriviální úlohy, jakou je například přepis melodie, jelikož dovoluje zvětšení celkové kapacity modelu, aniž by docházelo k přeučení. Také pro evaluaci metod, například i v soutěži MIREX, jsou potřeba takové datasety, které dobře reprezentují reálná data, přitom dataset MedleyDB vznikl mimo jiné z dů-

vodu, že stávající datasety nestačily ani pro účel evaluace.

Napsat signpost

3.1 Struktura dostupných dat a jejich přehled

Dataset, který chceme použít pro řešení úlohy extrakci melodie, musí obsahovat soubory se zvukem a k nim příslušící anotace melodie. Standardním formátem zvukových souborů je jedno- nebo více kanálový formát WAVE, se vzorkovací frekvencí 44 100 Hz. Anotace melodie je uložena jako CSV soubor s dvěma sloupci — časem a frekvencí. Výška melodie je tedy určena její fundamentální frekvencí a je specifikována pro každý časový okamžik v nahrávce. Délka anotačního okna je standardně 10 ms, případně $\frac{256}{44\,100} = 5.8$ ms. Nepřítomnost melodie se označuje hodnotou 0.

Výjimkou je dataset ORCHSET, který neobsahuje přesné anotace fundamentální frekvence melodie, ale pouze frekvence not. Tedy frekvence nejsou spojité, nýbrž jsou omezené na přesnost jednoho půltónu (V tabulce 3.1 je tato informace zohledněna řádkem MIDI melodie). Důležitou poznámkou je, že zde nejde o diskretizaci původní spojité křivky, ale opravdu jde o anotaci not, tedy pokud melodii nese nástroj hrající vibrato a svou výškou se dostane nad rozsah jednoho půltónu, v anotaci tato skutečnost není zaznamenána.

Tento formát, který byl zaveden v rámci soutěže MIREX, dodržují všechny dostupné datasety a ačkoli existují pokusy o změnu tohoto formátu (Humphrey a kol., 2014), MIREX formát je natolik jednoduchý a prozatím dostačující, že k přechodu na sofistikovanější formáty zatím nedošlo. Pro ilustraci přikládáme část referenční anotace ženského zpěvu, v anotaci se vyskytuje krátká pomlka mezi znějícími tóny. Grafické znázornění můžeme nalézt v úvodu na obrázku 1.1.

```
8.568     381.349
8.574     379.959
8.580     378.229
8.586     376.067
8.591     372.236
8.597     369.793
8.603     0.000
8.609     0.000
8.615     0.000
8.620     0.000
8.626     0.000
8.632     0.000
8.638     352.272
8.644     338.922
```

Datasety však mohou obsahovat více informací či audio souborů. Užitečné jsou například přiložené audio stopy, ze kterých je vytvořena výsledná píseň (mix), informace o všech znějících výškách (Multi-F0) nebo notách (MIDI), o melodické prioritě jednotlivých audio stop nebo o instrumentaci skladby.

V tabulce 3.1 nalezneme přehledné shrnutí obsahu všech dostupných datasetů.

	MedleyDB	Orchset	ADC04	MIREX05 train	MDB-synth	WJAZZD	MIR-1K	RWC
Audio	Ano	Ano	Ano	Ano	Ano	Ne[1]	Ano	Ano[2]
F0 melodie	Ano	Ne	Ano	Ano	Ano	Ano	Ano	Ano
MIDI melodie	Ne	Ano	Ne	Ne	Ne	Ano	Ne	Ne
Audio stopy	Ano [3]	Ne	Ne	Ne	Ano	Ne	Ano[4]	Ne
Multi-F0	Ne[5]	Ne	Ne	Ne	Ano	Ne	Ne	Ne
MIDI	Ne	Ne	Ne	Ne	Ne	Ne	Ne	Ano
Priorita stop	Ano	Ne	Ne	Ne	Ano	Ne	Ne	Ne
Informace o instrumentaci	Ano	Ano	Ne	Ne	Ano	Ano	Ne	Částečné
Celková délka	7.3 h[6]	23.4 m	6.1 m	6.5 m	3.19 h	8.85 h	2.22 h	—
Poměr znějící melodie	60.9%	93.69%	85.7%	63.1%	50.4%	62.8%	—	—
Počet nahrávek	122[7]	64	20	13	65	299	1000	315
Webová stránka	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
Žánr	mnoho-žánrový	klasika	pop,jazz, opera,midi	pop, midi	mnoho-žánrový	jazz	karaoke	pop, jazz klasika
Účel v této práci	Trénování Validace Testování	Testování	Testování	Testování	Testování	Testování	Žádný	Žádný

Tabulka 3.1: Souhrnná tabulka se základními informacemi o veřejně dostupných datasetech.

TODO: Tabulka splitů a použití v práci TODO: Tabulka datasetů v MIREXu

3.2 MedleyDB

Bittner a kol. (2014)

Multimodální, vícestopý dataset obsahující 122 nahrávek, k 108 z nich je dostupná anotace melodie. Kromě té dataset obsahuje také metadata o všech písní s informacemi o žánru a instrumentaci. S celkovou délkou 7.3 hodiny jde o nejdelší dataset, který se zaměřuje na více hudebních žánrů. O rozmanitosti svědčí i to, že se v datasetu vyskytuje řada nástrojů mimoevropského původu, a že jen přibližně polovina písní obsahuje zpěv. Na rozdíl od ostatních datasetů jsou nahrávky ve většině případů celé písně, tedy nejde pouze o krátké výňatky, a ke každé jsou poskytnuty audiostopy, ze kterých je vytvořen výsledný mix. Na základě diskuze, kterou shrnuji v kapitole o definici melodie, autoři poskytují tři verze anotací, na základě různě obecných definic:

[1] Autoři audio poskytují neveřejně pro výzkumné účely

[2] Přístup k datasetu je zpoplatněn

[3] Část stop obsahuje přeslech ostatních nástrojů, informace o přeslechu je současí metadat každé skladby.

[4] Oddělený zpěv a karaoke doprovod

[5] Je dostupný přepis všech znějících melodií, viz Definice 3 v sekci MedleyDB.

[6] 5.59 h s anotací melodie

[7] 108 s anotací melodie

[8] `https://medleydb.weebly.com/`

[9] `https://www.upf.edu/web/mtg/orchset`

[10] `http://ismir2004.ismir.net/melody_contest/results.html`

[11] `https://labrosa.ee.columbia.edu/projects/melody/`

[12] `http://synthdatasets.weebly.com/mdb-melody-synth.html`

[13] `https://jazzomat.hfm-weimar.de/`

[14] `https://sites.google.com/site/unvoicedsoundseparation/mir-1k`

[15] `https://staff.aist.go.jp/m.goto/RWC-MDB/`

1. Základní frekvence nejvýraznějšího melodického hlasu, jehož zdroj zůstává po dobu nahrávky neměnný. [16]

2. Základní frekvence nejvýraznějšího melodického hlasu, jehož zdroje se mohou měnit.

3. Základní frekvence všech melodických hlasů, potenciálně pocházejících z více zdrojů.

Ačkoli třetí definice umožňuje, aby v anotaci znělo více melodických linek zároveň, v datasetu se nejedná o kompletní přepis nahrávek (použitelný pro úlohu *multif0 estimation*), ten autoři neposkytují.

Dataset vznikl obvyklou cestou ruční anotace, ze shromážděného vícestopého materiálu byly vybrány stopy s potenciálním výskytem melodie, stopy s přeslechem byly filtrovány pomocí source-separation algoritmu s ručně doladěnými parametry pro každou jednotlivou stopu, následně byl na monofonní stopy spuštěn pitch-tracker pYIN a výsledné automaticky získané anotace opravilo a vzájemně zkontrolovalo pět anotátorů s hudebním vzděláním.

3.3 MDB-synth

Hlavním přínosem práce Salamon a kol. (2017) je navržení způsobu anotace základní frekvence monofonních audiostop takovým způsobem, že výsledná dvojice zvukové stopy a anotace nevyžaduje další manuální kontrolu. Anotace stopy probíhá ve dvou krocích, nejprve získáme pomocí libovolného monopitch trackeru křivku základní frekvence a poté na základě této křivky, která může obsahovat chybné úseky, syntetizujeme novou stopu, která zachovává barvu nahrávky, ale výšku tónu určuje právě tato anotace. Díky tomu je pak přesnost anotace pro tuto novou, syntetickou nahrávku stoprocentní, přitom (v ideálním případě) neztrácí charakteristiky původní nahrávky.

Pro vytváření datasetu je toto významné zjednodušení, protože tím algoritmus odstraňuje časově nejnáročnější část práce - ruční kontrolu anotací audiostop. Pokud by se ukázalo, že syntéza významně neubírá na kvalitě dat, použitím navrhované metody by mohlo vzniknout velké množství nových dat (například repozitář Open Multitrack Testbed obsahuje stovky vícestopých nahrávek, které by šlo využít). Autoři v článku provádí kvantitativní analýzu pomocí srovnání state-of-the-art algoritmů pro extrakci melodie a prokazují, že výsledky těchto metod na syntetických datech se významně neliší od výsledků na původních, tím je podle autorů potvrzená možnost použití dat jak pro trénování tak pro evaluaci nových metod.

Metoda má ale bohužel svá omezení, mezi ty zásadní patří, že se dá aplikovat pouze na stopy, které obsahují monofonní signál, vstupní data tedy nesmí obsahovat přeslech a nahrávaný nástroj může hrát pouze jednohlase, v důsledku nelze zpracovat klavír či kytara, které hrají zpravidla vícehlas. To nevadí tolik u generování datasetu pro přepis melodie, jelikož melodii často hraje jeden hlas a doprovod hrají ostatní, velkým nedostatkem je toto spíše pro generování multif0 datasetů.

[16] Tato definice je shodná pro evaluační datasety používané v soutěži MIREX, s výjimkou Orchsetu

Dále k článku není zveřejněná kompletní referenční implementace algoritmu, tudíž algoritmus nelze snadno spustit na nových datech. Ve výsledku je tudíž největším praktickým přínosem nová sada syntetických datasetů pro úlohy přepisu melodie, basy, monofonních stop a kompletní partitury, každý dataset obsahuje destíky nahrávek. Vícestopá data použitá pro syntézu byla převzata z MedleyDB, tudíž ve výsledku nové datasety nerozšiřují celkový hudební záběr, pouze zpřesňují ten již existující.

TODO obrázek? Porovnání spektrogramů syntetické a původní nahrávky

Z kvalitativního pohledu je na výstupních syntetických nahrávkách poznat, že jsou syntetické. Autoři sice prokazují, že současné metody na těchto datech dosahují stejných výsledků, nicméně v článku chybí diskuse o tom, zda-li v datech algoritmus nevytváří nové umělé artefakty, které by mohly zneužít metody strojového učení pro spolehlivější výsledky (které by však negeneralizovaly na reálná data). Při pohledu na spektrogram je například zřejmé, že syntetická nahrávka obsahuje mnohem více výrazných alikvótních frekvencí

3.4 Orchset

Dataset vytvořený týmem Bosch a kol. (2016) orientovaný na orchestrální repertoár pocházející z různých historických období včetně 20. století. Obsahuje 64 výňatků délky od 10 do 32 sekund. Výňatky byly vybírány tak, aby obsahovaly zřejmou melodii, dataset tedy obsahuje v porovnání málo pasáží bez melodie (6% z celkové délky). Vzhledem k komplexitě uvažovaných žánrů autoři vycházejí z kombinace rozšířené definice melodie podle Bittner a kol. (2014) a definice Poliner a kol. (2007). Melodii ve výňatcích proto zpravidla nese více hudebních nástrojů (nebo celých sekcí), které se v průběhu střídají, případně mohou části hrát společně v rozdílných oktávách (nebo jiných intervalech, tvoříce tak harmonický doprovod).

Pro zjištění melodie se v takto vrstveném materiálu autoři uchylují k úplnému základu definice melodie (Poliner) a nechávají si skupinou čtyř posluchačů výňatky přezpívávat. Tato hrubá data pak autoři sumarizují a odebírají z datasetu ty výňatky, na jejichž melodii se posluchači neshodli. Přezpívané tóny bylo nutné ručně opravit, aby načasováním přesně seděly na výňatek. Lidský hlas také samozřejmě nemá rozsah plného orchestru, proto bylo dalším krokem transponovat anotace tak, aby zněly ve správných oktávách. Zde se opět může vyskytnout problém subjektivity, pokud melodii hrají dva různé nástroje, pouze v jiných oktávách, pak je sporné, který nástroj označit jako hlavní (v některých případech taková otázka ani nedává příliš smysl.). Částečným řešením je zvolit libovolnou anotační politiku a tu konzistentně dodržovat (žádná společná v komunitě MIR neexistuje), v případě Orchsetu byla snaha minimalizovat skoky v melodické kontuře, což zároveň respektuje obecné pozorování, že v melodii se vyskytují mnohem častěji malé skoky (nejčastěji prima a malá/velká sekunda) než větší. Tedy například pokud pasáži hrané ve dvou různých oktávách předcházela pasáž hraná v jedné, anotace obou pasáží lze transponovat do společné oktávy tak, abychom na rozhraní minimalizovali skok v anotaci.

Dataset obsahuje pouze hrubé anotace tónů melodie, nikoli přesnou základní frekvenci nástroje, který v danou chvíli melodii hraje. Článek o tomto rozhodnutí příliš nediskutuje, vychází ale opět logicky z volby dat. U orchestrálních dat je

tento abstraktnější pojem melodie mnohem méně sporný. Pokud hraje melodii sekce nástrojů v unisonu, přesná základní frekvence není dobře definovaná, jelikož se základní frekvence znějících hlasů vzájemně překrývají.

3.5 Weimar Jazz Database

Weimar Jazz Database Pfleiderer a kol. obsahuje přes 450 transkripcí jazzových sól ze všech období vývoje jazzu. Data původně zamýšlená pro muzikologické studie využívající statistické metody ale lze využít i pro potřeby extrakce melodie, jelikož uvažované nahrávky spadají zřejmě pod nejrestriktivnější definici melodie (definici používanou v soutěži MIREX) - melodii nese jistě právě jeden, sólový nástroj, a po celou dobu výňatku je jistě nejvýraznější. Výběr sólových nástrojů se omezuje pouze na jednohlasé, jelikož ruční anotace vícehlasých je příliš obtížná. Hlavním problémem při využívání je restriktivní licence, která platí na nahrávky, tudíž zdrojové audio, na základě kterého anotace vznikaly, není veřejně přístupné. Jelikož pro data neexistují jednotlivé stopy, ruční anotace probíhala přímo z finální nahrávky, což je obtížný úkol -

4. Metody evaluace

4.1 MIREX

Soutěž MIREX (Music Information Retrieval Evaluation eXchange) probíhá již od roku 2005 a v MIR komunitě zastává hlavní postavení jakožto každoroční událost pro nezávislé, objektivní srovnání state-of-the-art metod a algoritmů pro řešení širokého spektra úloh souvisejících se zpracováním hudebních dat. Mezi tyto úlohy patří například *rozpoznání žánru*, *odhad tempa*, *odhad akordů*, *identifikace coveru* a samozřejmě také *extrakce melodie*.

Na rozdíl od jiných úloh, kde debata o zvolení nejvhodnějších objektivních metrik pro porovnávání stále probíhá, metriky pro extrakci melodie se ustanovily již v prvním ročníku (na základě dřívějších zkušeností) a zůstaly neměnné dodnes Raffel a kol. (2014). Naopak data použitá pro testování se postupně kumulují a dnes soutěž probíhá již s řadou datasetů (ADC04, MIREX05, MIREX08, MIREX09, ORCHSET), které blíže popisuje kapitola o dostupných datech.

4.2 Trénovací, validační a testovací množina

Z dostupných dat, které pro úlohu máme k dispozici, musíme vyhradit množiny pro trénování, validaci a testování, aby byly metody porovnatelné jak mezi sebou, tak se stávajícími state-of-the-art metodami. Pro trénování se jeví jako nejvhodnější dataset MedleyDB, jednak pro svou délku a jednak pro žánrovou rozmanitost, proto je použit pro většinu popsaných experimentů. Rozdělení na tři části vychází z práce Bittner a kol. (2017) a D Basaran, S Essid (2018), aby byly metriky přímo porovnatelné s výsledky v uvedených článcích.

Dalším užitečným zdrojem dat je dataset MDB-melody-synth, který je přesyntetizován z vícestopých nahrávek MedleyDB. Proto se nabízí použít stejné rozdělení dat, jaké se používá pro MedleyDB, ze stejných důvodů uvedených v předchozím odstavci. Jelikož dataset neobsahuje veškerá data, ale pouze jejich podmnožinu, i v experimentech používané rozdělení dat obsahuje pouze podmnožinu z původního rozdělení datasetu MedleyDB.

Posledním velkým datasetem, používaným pro trénování, je Weimar Jazz Database. Zde žádný doporučený postup ani výběr rozdělení dataestu v relevatní literatuře neexistuje, proto jsem dataset rozdělil podle metody Bittner a kol. (2017) na tři části (v celkové délce nahrávek na části v poměrech 63%, 14% a 23%). Skladby jsou rozděleny do částí podle interpretů tak, aby se každý interpret vyskytoval právě v jedné části datasetu. Toto omezení na podmnožiny Bittner a kol. (2017) nediskutuje, lze však doložit (práce Sturm (2013)), že pro úlohu rozpoznání žánru metody založené na strojovém učení vykazují po trénování a validaci na datech bez tohoto filtru výrazně lepší výsledky než stejné metody spuštěné na roztříděných datech, takové zlepšení výkonu je ale jistě umělým důsledkem špatné volby trénovací množiny.

Ostatní datasety (ADC04, MIREX05, ORCHSET) jsou v práci použity pouze jako testovací data, díky tomu lze korektně výsledky přímo srovnávat s žebříčky úlohy Melody Extraction v soutěži MIREX.

4.3 Kvalitativní příklady

Pro lepší porozumění hranic testovaných metod je vhodné studovat také výsledky na kvalitativních ukázkách. Modely byly při práci vyhodnocovány na několikaminutových množinách výňatků z validačních a testovacích dat. Metodika výběru spočívala v poslechu nahrávek a ručním hledáním zajímavých hudebních jevů a také v seřazení nahrávek podle úspěšnosti přepisu stávajícími metodami a výběrem výňatků právě z těchto nejproblematičtějších příkladů.

Omezení plynoucí z potřeby zkrátit výňatky na minimum,

4.4 Metriky

Celkovou kvalitu metody pro extrakci melodie určuje její schopnost určit výšku tónu hrající melodie (*odhad výšky melodie*) a také rozpoznat části skladby, které melodii neobsahují (*detekce melodie*). Jelikož jsou tyto podúlohy na sobě nezávislé, standardní sada metrik zahrnuje jak celkové vyhodnocení přesnosti, tak dílčí vyhodnocení pro *odhad výšky* a *detekci melodie*.

4.4.1 Formát výstupu

Obvyklý formát výstupu algoritmů je CSV soubor se dvěma sloupci. První sloupec obsahuje pravidelné časové značky, druhý sloupec pak odhad základní frekvence melodie. Některé algoritmy uvádí i odhady výšky základní frekvence mimo detekovanou melodii (může jít například o doprovod, který zní i po hlavním melodickém hlasu). Aby tyto odhady byly odlišené od odhadů hlavní melodie, jsou uvedeny v záporných hodnotách. Díky tomu pak lze nezávisle vyhodnotit přesnost *odhadu výšky* a *detekce melodie*. Odhad výšky se vyhodnocuje podle absolutní hodnoty frekvence ve všech časových oknech, ke kterým existuje anotace, detekce melodie pak na všech hodnotách vyšších než 0.

4.4.2 Definice metrik

Většina metrik je definována na základě porovnávání jednotlivých anotačních oken - tedy typicky srovnáním odhadovaných a pravdivých výšek melodie po konstantních časových skocích. Datasety používané pro vyhodnocování v soutěži MIREX používají časový skok délky 10 ms. V definicích budu vycházet ze značení v práci Salamon a kol. (2014).

Označme vektor odhadovaných základních frekvencí $\mathbf{f}$ a cílový vektor $\mathbf{f}^*$, složka f_τ je buď rovna hodnotě f_0 melodie, nebo 0, pokud v daném čase melodie nezní. Obdobně zaveďme vektor indikátorů $\mathbf{v}$, jehož prvek na pozici τ je roven $v_\tau = 1$, pokud je v daném časovém okamžiku detekována melodie a $v_\tau = 0$ v opačném případě. Podobným způsobem zavedeme i vektor cílových indikátorů melodického hlasu $\mathbf{v}^*$ a také vektor indikátorů absence melodie $\bar{v}_\tau = 1 - v_\tau$.

přidat obrázky příkladů chyby

Voicing Recall rate (Úplnost detekce)

Poměr počtu časových oken, které byly správně označené jakožto obsahující melodii, a počtu časových oken doopravdy obsahujících melodii podle anotace.

$$\mathrm{VR}(\mathbf{v}, \mathbf{v}^*) = \frac{\sum_\tau v_\tau v_\tau^*}{\sum_\tau v_\tau^*}$$

Voicing False Alarm rate (Nesprávné detekce)

Poměr počtu časových oken, které byly nesprávně označené jako melodické, k počtu doopravdy nemelodických oken.

$$\mathrm{FA}(\mathbf{v}, \mathbf{v}^*) = \frac{\sum_\tau v_\tau \bar{v}_\tau^*}{\sum_\tau \bar{v}_\tau^*}$$

upozornit, že nižší je lepší!

Raw Pitch Accuracy (Přesnost odhadu výšky)

Poměr správně odhadnutých tónů k celkovému počtu melodických oken. Výška správně určeného tónu se může lišit až o jeden půltón.

$$\mathrm{RPA}(\mathbf{f}, \mathbf{f}^*) = \frac{\sum_\tau v_\tau^* v_\tau \mathcal{T}[\mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*)]}{\sum_\tau v_\tau^*}$$

kde $\mathcal{T}$ je prahová funkce

$$\mathcal{T}[a] = \begin{cases} 1 & \text{pro} |a| \leq 0.5 \\ 0 & \text{jinak} \end{cases}$$

přidat zmínku o možnosti změnění práhu

a $\mathcal{M}$ je funkce zobrazující frekvenci f na reálné číslo počtu půltónů od nějakého referenčního tónu f_{ref} (například od 440 Hz, tedy komorního A4).

$$\mathcal{M}(f) = 12 \log_2 (\frac{f}{f_{\mathrm{ref}}})$$

Raw Chroma Accuracy (Přesnost odhadu výšky nezávisle na oktávě)

Počítá se podobně jako *Přesnost odhadu tónu*, výstupní a cílové tóny jsou však mapovány na společnou oktávu. Metrika tedy ignoruje chyby odhadu způsobené špatným určením oktávy tónu.

$$\mathrm{RCA}(\mathbf{f}, \mathbf{f}^*) = \frac{\sum_\tau v_\tau^* v_\tau \mathcal{T}[\langle \mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*) \rangle_{12}]}{\sum_\tau v_\tau^*}$$

Nezávislost na oktávě zajistíme pomocí zobrazení rozdílu cílového a výstupního tónu na společnou oktávu.

$$\langle a \rangle_{12} = a - 12 \lfloor \frac{a}{12} + 0.5 \rfloor$$

Overall Accuracy (Celková přesnost)

Celková přesnost měří výkon algoritmu jak v odhadu melodie tak v detekci melodie. Počítá se jako podíl správně odhadnutých oken a celkového počtu oken.

$$\mathrm{OA}(\mathbf{f}, \mathbf{f}^*) = \frac{\sum_\tau v_\tau^* v_\tau \mathcal{T}[\mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*)] + \bar{v}_\tau^* \bar{v}_\tau}{L}$$

Poznámka k definicím metrik

Definice RPA, RCA a OA zde uvedené se mírně liší od výchozích v práci Salamon a kol. (2014), jejich přímá implementace podle vzorce totiž vede kvůli nedostatečně dobře zadefinovanému vektoru frekvencí $\mathbf{f}$ k chybě, která se týká zejména metriky RCA. Ta původní definici chybně zahrnovala jako správné tóny ty, které algoritmus odhadl jako nulové (tedy neznějící) a zároveň jejich pravdivá hodnota byla po zobrazení na jednu společnou oktávu blízká nule (tedy původní tón byl blízký nějakému násobku referenční frekvence). Kvůli zobrazení na společnou oktávu se stanou „neznělé nulové odhady“ a tóny blízké referenčním frekvencím nerozlišitelné a byly nesprávně považované za korektní. [1]

přepsat footnote ještě trochu

4.4.3 Další metriky

Protože princip vnitřního fungování neuronových sítí často není zřejmý, je užitečné mít co nejvíce různých indikátorů, abychom měli při porovnávání jednotlivých modelů alespoň podrobnou informaci, v jakých ohledech se síť zlepšuje nebo zhoršuje. Pro tento účel jsem při práci implementoval další metriky, které při hledání architektur sítí pomáhaly.

Chroma Overall Accuracy

Počítá se obdobně jako Overall Accuracy, ale tóny jsou mapovány na společnou oktávu.

Raw Harmonic Accuracy

Metrika počítá odhadovaný tón jako správný, pokud se trefil do některé z harmonických frekvencí tónu. Protože je harmonických frekvencí teoreticky nekonečné množství, parametrem metriky je do jakého celočíselného násobku se ještě odhad počítá.

$$\mathrm{RHA}(\mathbf{f}, \mathbf{f}^*, n) = \frac{\sum_{k=1}^{n} \sum_\tau v_\tau^* v_\tau \mathcal{T}[\mathcal{M}(f_\tau) - \mathcal{M}(k f_\tau^*)]}{\sum_\tau v_\tau^*}$$

[1] která byla přítomna i v nejpoužívanější, veřejné implementaci MIR metrik *mir_eval*. V praxi chyba této metriky na datasetu MedleyDB mohla dosahovat až sedmi procentních bodů, na repozitáři hostovaném na serveru Github jsme již spolu s autory chybu odstranili (odkaz na Github issue: `https://github.com/craffel/mir_eval/issues/311`). Opravný patch bude zahrnut do další verze balíku. Výsledky v této práci jsou počítány s opravenou verzí.

Matice záměn not

Pro podrobnější souhrnný přehled četností chyb se pro klasifikační úlohy používá matice záměn. Sloupce označují správné noty, řádky odhadované. Buňka na pozici (x,y) má pak hodnotu podle četnosti odhadu noty y místo správné noty x.

5. Experimenty

V této kapitole prezentujeme návrhy nových architektur hlubokých neuronových sítí pro extrakci melodie a jejich výsledky. Modely byly trénované nad datasetem MedleyDB. Data byla rozdělena na trénovací, validační a testovací množinu tak, aby skladby od jednoho interpreta náležely právě do jedné z množin. Využili jsme existujícího rozdělení používaného v článcích Bittner a kol. (2017) a D Basaran, S Essid (2018), validační i testovací výsledky jsou tedy díky tomu porovnatelné s výsledky uvedenými v článcích.

Těžiště experimentů leží v hledání nové salienční funkce, která by odlišovala znějící melodii od doprovodu lépe, než existující metody.

Pro výpočet salienční funkce porovnáváme architektury inspirované pracemi Kim a kol. (2018), van den Oord a kol. (2016a) a Bittner a kol. (2017). V prvním případě jde o architekturu CREPE původně navrženou pro sledování výšky tónů v jednohlasých nahrávkách. van den Oord a kol. (2016a) používají architekturu WaveNet složený z dilatovaných konvolucí pro generování lidské řeči. Bittner a kol. (2017) používá hlubokou konvoluční síť pro kompletní přepis nahrávek i pro přepis melodie. líp popsat moje architektur

Pro detekci melodie pak srovnáváme jednoduchou metodu práhování a složitější samostatný modul, založený na hlubokých neuronových sítích.

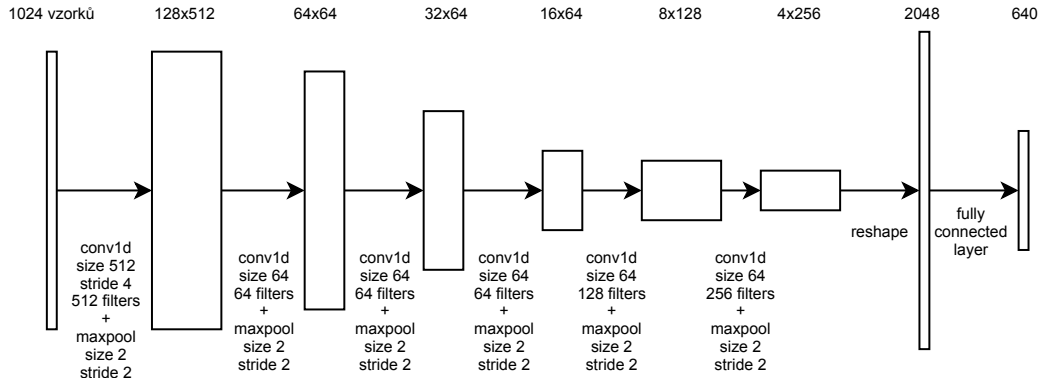
5.1 Architektura CREPE

První sada experimentů se zakládá na architektuře popsané v článku od Kim a kol. (2018) použité pro sledování jednohlasu. Jak blíže popisujeme v kapitole Související práce, cílem monopitch trackingu je určit konturu základní frekvence melodického nástroje v jednohlasé nahrávce. Tato nahrávka se zpravidla skládá ze směsi čistého signálu hlasu a šumu v pozadí. Pokud však rozšíříme pojem šumu v pozadí tak, aby zahrnoval i melodický doprovod, pak dostáváme polyfonní signál, tedy vstupní signál pro metody extrakce melodie.

Jinými slovy je sledování jednohlasu speciálním případem extrakce melodie a tudíž přinejmenším stojí za zkoušku pokusit se tuto architekturu pro extrakci využít. Mimo to jednohlasé stopy často obsahují přeslech ostatních nástrojů, pokud nahrávka vznikala při společném hraní ve studiu, tudíž by model trénovaný na vícehlasé mixech mohl být robustní vůči tomuto druhu rušení.

Architektura CREPE se skládá ze šesti konvolučních a pooling vrstev, pro regularizaci používá batch normalization a dropout po každé konvoluční vrstvě, jako nelineární aktivace je použita funkce ReLU. Po konvolucích následuje výstupní plně propojená vrstva, jako finální aktivační funkce je použita sigmoida. Vstupem modelu je okno o velikosti 1024 vzorků jednokanálového audio signálu, převzorkovaného na 16 kHz. Před první konvolucí je signál normalizován tak, aby každé jednotlivé vstupní okno mělo střední hodnotu 0 a směrodatnou odchylku 1. Podrobnější popis modelu je naznačen na obrázku.

Výsledný vektor o 640 složkách aproximuje pravděpodobnostní rozdělení výšky základní frekvence uprostřed vstupního okna, přičemž tento vektor pokrývá rozsah od noty C_{-1} po G_9, mezi dvěma sousedními predikovanými tóny je vzdálenost 20 centů. Výšky tónů v centech označíme $¢_1, ¢_2, \ldots, ¢_{640}$. Rozsah tedy bezpečně



Obrázek 5.1: Diagram architektury CREPE, multiplikační koeficient 16x.

pokrývá obvyklé hudební nástroje a na jednu notu připadá 5 složek (tónů) výsledného vektoru.

$$\text{ç}(f) = 1200 \log_2 \frac{f}{f_{\text{ref}}}$$

Pro trénování modelu potřebujeme také cílové diskrétní pravděpodobnostní rozdělení základní frekvence tónu. Jako cílovou pravděpodobnostní funkci použijeme normální rozdělení se střední hodnotou v bodě cílové základní frekvence $\text{ç}(f_{\text{ref}})$ a se směrodatnou odchylkou 25 centů. Toto rozdělení diskretizujeme tak, aby měl cílový vektor stejné dimenze jako odhadovaný.

$$y_i = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\text{ç}_i - \text{ç}_{\text{ref}})^2}{2\sigma^2}\right)$$

Převod z pravděpodobnostní reprezentace výstupního vektoru na konkrétní hodnotu výšky noty provedeme pomocí výpočtu střední hodnoty výstupní distribuce. Jelikož by při výpočtu střední hodnoty ale hodnotu výsledné výšky tónu ovlivňoval i doprovod, který se na výstupním vektoru objevuje, počítáme střední hodnotu pouze z okolí maxima výstupu. Tím zajistíme, že získáme střední hodnotu gaussiánu náležícímu pouze jednomu tónu.

$$\hat{\text{ç}} = \sum_{i,|\text{ç}_i - \text{ç}_m|<50} \hat{y}_i \text{ç}_i \Big/ \sum_{i,|\text{ç}_i - \text{ç}_m|<50} \hat{y}_i \ , m = \text{argmax}_i(\hat{y}_i)$$

Optimalizovaná ztrátová funkce modelu (loss funkce) $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})$ se počítá jako vzájemná korelace mezi vektorem cílových pravděpodobností y a výstupním vektorem $\hat{y}$.

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{i=1}^{640} \left(-y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i)\right)$$

Optimalizace probíhá pomocí algoritmu Adam (Kingma a Ba, 2014) s parametrem learning rate 0.0002.

5.1.1 Replikace výsledků CREPE

Abychom ověřili správnost implementace architektury CREPE pro sledování jednohlasu, spustíme model na syntetických, jednohlasých datech MDB-stem-

	1.	2.	3.	4.	5.	6.	Celk. parametrů
CREPE 4x	128	16	16	16	32	64	558 240
CREPE 8x	256	32	32	32	64	128	177 1200
CREPE 16x	512	64	64	64	128	256	6 163 200

Tabulka 5.1: Počty filtrů v konvolučních vrstvách v architektuře CREPE v závislosti na multiplikačním koeficientu.

synth, která byla zvěřejněná spolu s článkem od Salamon a kol. (2017).

Na rozdíl od článku Kim a kol. (2018), ve kterém autoři používají pro celkové vyhodnocení architektury pětinásobnou křížovou validaci, jsme použili pouze jednu trénovací a testovací množinu. Zásadní rozdíly mezi implementacemi modelu jsme na základě článku a veřejně dostupného kódu neobjevili.

Po jedné epoše trénování model dosáhl na testovací množině 98.6% přesnosti odhadu výšky. Kim a kol. (2018) uvádí přesnost modelu 97%. V jejich případě jde o průměrný výsledek pěti nezávislých běhů trénování a testování na různě rozdělených datových množinách. Rozdíl mezi dosaženými přesnostmi přičítáme odlišné evaluační strategii.

Metrika	Práh	Průměrná hodnota	Hodnota Kim a kol. (2018)
RCA	50 centů	0.988	0.970
RPA	50 centů	0.986	0.967
RPA	25 centů	0.975	0.953
RPA	10 centů	0.937	0.909

Tabulka 5.2: Výsledky pokusu o replikaci. Přesnosti nejsou přímo srovnatelné kvůli různým evaluačním strategiím.

Při replikaci experimentu jsme narazili na důležitost správného promíchání dat. Framework Tensorflow použitý pro trénování promíchává data vždy pomocí bufferu pevné velikosti pro dvojice vstupů a cílových výstupů. V praxi je však potřeba buď nastavit buffer na velikost větší než je celková velikost datasetu, a nebo implementovat vlastní míchání přes všechna dostupná data. Při nedostatečně promíchaných datech totiž trénovací dávka (batch) není reprezentativní pro celý dataset, ale pouze pro jeho podmnožinu, což se negativně projevuje kolísající validační přesností modelu.

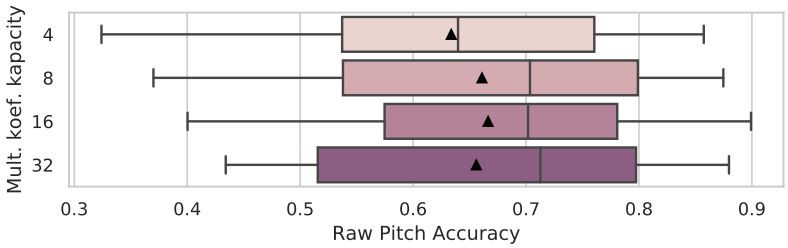
5.1.2 CREPE pro extrakci melodie

Jako první experiment extrakce melodie z polyfonních dat spustíme nezměněnou architekturu CREPE, v následujících experimentech se tuto baseline pokusíme překonat. Abychom urychlili trénování následujících experimentů, přesnost určíme pro sítě s různou kapacitou. Pokud se výsledky při různých kapacitách nebudou příliš lišit, můžeme experimenty provádět s architekturou s nižší kapacitou a tím snížit trénovací čas. Kapacity upravíme pomocí multiplikačního koeficientu počtu filtrů u všech konvolučních vrstev, počty filtrů jsou uvedeny v tabulce 5.2.

Z validačních výsledků po 200 000 iteracích (přibližně 6 epoch) vidíme, že se výsledek modelů CREPE 8x a CREPE 16x liší řádově o desetiny procentních

Mult. koef. kapacity	RPA	RCA
4	0.634	0.753
8	0.661	0.766
16	0.666	0.771
32	0.656	0.753

Tabulka 5.3: Výsledky experimentu s různými kapacitami modelu.



Obrázek 5.2: Výsledky experimentu s různými kapacitami modelu.

bodů. Přitom model s větší kapacitou se trénuje o 35% delší dobu. Proto pro další srovnávání zvolíme architektury s multiplikačním koeficientem 8, modely s dobrými výsledky případně přetrénujeme s vyšší kapacitou.

5.1.3 Vliv rozlišení diskretizace výšky noty

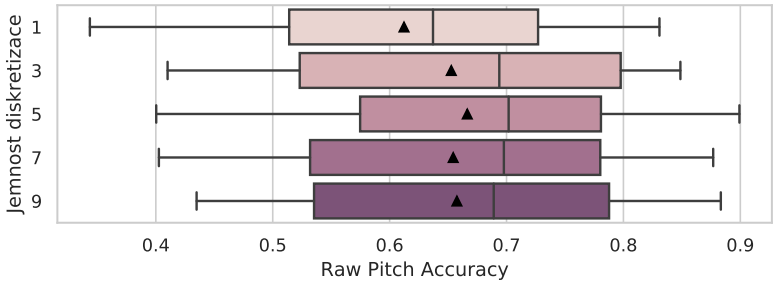
Otestujeme nastavení granularity výstupního vektoru. V článku Kim a kol. (2018) se totiž důvod volby pěti frekvencí na notu nediskutuje. Intuitivně by však mělo vyšší rozlišení spíše pomáhat, důvodem je, že nástroje a zejména lidský hlas se často při hraní odchylují od přesných, definovaných frekvencí hraných not a vyšší rozlišení tyto odchylky může lépe zachytit. Ve výsledku by pak síť s jemnějším výstupem měla dělat méně chyb, kde se skutečná a výstupní hodnota liší o jeden půltón.

Jemnost diskretizace	RPA	RCA
1	0.612	0.711
3	0.653	0.760
5	0.666	0.771
7	0.654	0.763
9	0.658	0.760

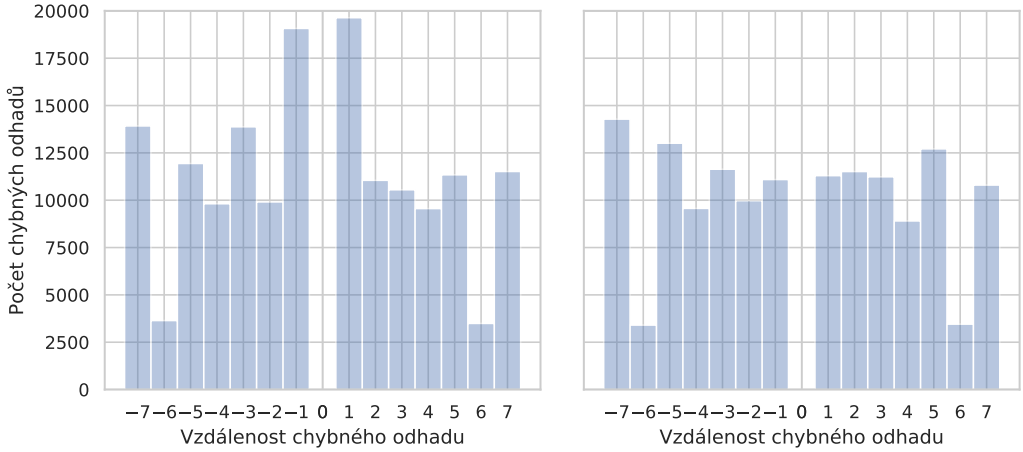
Tabulka 5.4: Architektura CREPE s různou jemností diskretizace.

Jak můžeme pozorovat na výsledných hodnotách, jemná granularita výstupu jednoznačně zlepšuje přesnost sítě. Abychom ověřili domněnku, že vyšší rozlišení pomáhá zmenšit počet chyb o půltón, vytvoříme histogram vzdáleností cílového a odhadovaného tónu. V tomto histogramu by pak měl být zřetelný pokles v příslušných třídách. Podle histogramu se počet chyb o půltón mezi zkoumanými modely liší téměř o polovinu, zlepšení tohoto druhu chyb je tedy podstatné.

odsud dolů přepsat



Obrázek 5.3: Architektura CREPE s různou jemností diskretizace.



Obrázek 5.4: Histogramy vzdálenosti chybného odhadu, výstup prvního modelu má rozlišení 50 centů, výstup druhého 10 centů.

5.1.4 Vliv rozptylu cílové pravděpodobnostní distribuce výšky noty

Podle Bittner a kol. (2017) pomáhá cílová distribuce s vyšším rozptylem snížit penalizaci sítě za téměř korektní odhady výšek tónů. Mimo to u dostupných dat často nejsou anotace naprosto perfektní, jisté rozostření hranice anotace tudíž pomáhá i v případě nepřesné cílové anotace, síť pak není tolik penalizována za svou případnou správnou odpověď.

V článku se však nediskutuje, proč bylo zvoleno nastavení směrodatné odchylky na 20 centů. Kim a kol. (2018) používá odchylku 25 centů a není na první pohled zřejmé, jaká je optimální hodnota. Příliš vysoký rozptyl způsobí, že síť bude tolerovat více chyb o půltón, příliš nízký rozptyl naopak penalizuje i téměř správné odhady. Intuitivně se nejlepší nastavení pravděpodobně bude pohybovat kolem používaných 25 centů, jelikož to je hranice chybné klasifikace, na druhou stranu optimální hodnota jistě bude závislá na nastavení rozlišení výstupního vektoru, jelikož nižší rozlišení bude jistě vyžadovat vyšší hodnotu rozptylu (v extrémním případě rozptylu blížícího se k nule a cílové frekvence mimo kvantizační hladiny by vzniklý cílový vektor nemusel obsahovat žádné ostré maximum).

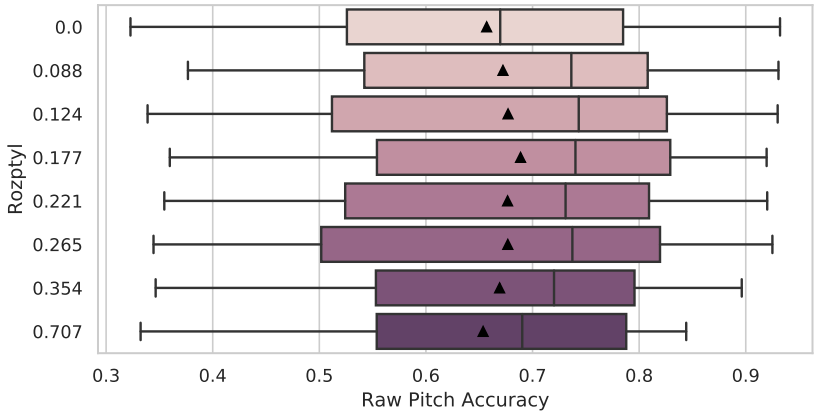
Poznamenám také technický detail, který je důležitý při samotné implementaci. Přestože jsem cílový výstup sítě zadefinoval jako diskrétní pravděpodobnostní rozdělení, při trénování je tento vektor hodnot pronásoben koeficientem

tak, aby $\max(\mathbf{y}) = 1.0$ a tedy součet prvků vektoru není roven jedné (a o pravděpodobnostní rozdělení se doopravdy nejedná). Důvodem je použití aktivační funkce *sigmoid* u výstupní vrstvy, která nezaručuje výstup korektního rozdělení. Díky tomu se na výstupu může objevit různé množství stejně pravděpodobných kandidátů na melodii.

Testovaná síť má vstupní okno široké 4096 vzorků, používá multiplikátor kapacity 16x a vstup zpracovává 6 různě širokými konvolučními vrstvami (viz experiment *Vliv násobného rozlišení první konvoluční vrstvy*).

Rozptyl	RPA	RCA
0.000	0.657	0.759
0.088	0.672	0.775
0.124	0.677	0.773
0.177	0.689	0.784
0.221	0.677	0.771
0.265	0.677	0.770
0.354	0.669	0.773
0.707	0.654	0.757

Tabulka 5.5: Architektura CREPE, vliv rozptylu cílové distribuce.



Obrázek 5.5: Architektura CREPE, vliv rozptylu cílové distribuce.

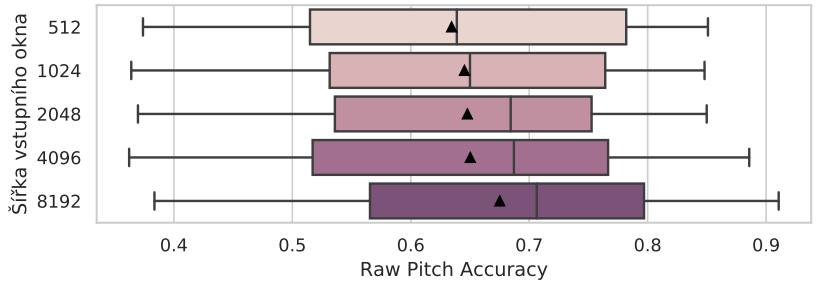
Z experimentů vyplývá, že optimální směrodatná odchylka se pohybuje kolem hodnoty 0.177, tedy níže než v porovnávaných pracích.

5.1.5 Vliv šířky vstupního okna

Architektura CREPE byla navržena pro monopitch tracking, dá se předpokládat, že jelikož je v monofonních nahrávkách oproti polyfonním daleko méně (melodického) šumu, není pro určení výšky tónu potřeba větší kontext než použitých 1024 vzorků (při vzorkovací frekvenci 16kHz toto odpovídá 64 milisekundám audia). To ale nemusí platit pro složitější signály, kde by síť mohla z delšího kontextu těžit. Otestujeme tedy vliv většího vstupního okna na výslednou přesnost.

Šířka vstupního okna	RPA	RCA
512 (32 ms)	0.634	0.748
1024 (64 ms)	0.645	0.763
2048 (128 ms)	0.648	0.760
4096 (256 ms)	0.650	0.762
8192 (512 ms)	0.675	0.775

Tabulka 5.6: Architektura CREPE, vliv šířky vstupního okna.



Obrázek 5.6: Architektura CREPE, vliv rozptylu cílové distribuce.

5.1.6 Vliv násobného rozlišení první konvoluční vrstvy

Podle Kim a kol. (2018) se přesnost CREPE snižuje s výškou tónu. Autoři si tuto skutečnost vysvětlují neschopností modelu generalizovat na barvy a výšky tónů neobsažených v trénovací množině, generalizaci by ale mohla pomoci také úprava modelu. Protože k rozpoznání vyšších frekvencí stačí méně vzorků než pro rozpoznání nižších, mohli bychom se pokusit upravit první konvoluční vrstvu sítě, která tento úkol zastává, a rozdělit ji na množiny různě širokých konvolucí, jejichž kanály následně sloučíme zpět do jednotné vrstvy. To by mělo mít za následek, že rozpoznávání vysokých tónů budou zastávat užší konvoluce a jejich kernel bude jednodušší než široké kernely s vysokou mírou redundance.

První vrstvu s kernelem s 256 filtry (tj. počet filtrů první vrstvy s multiplikátorem 8x, viz první experiment) jsem rozdělil na vícero různě širokých kernelů s menším počtem filtrů, tak aby kapacita sítě zůstala přibližně stejná a sítě byly porovnatelné.

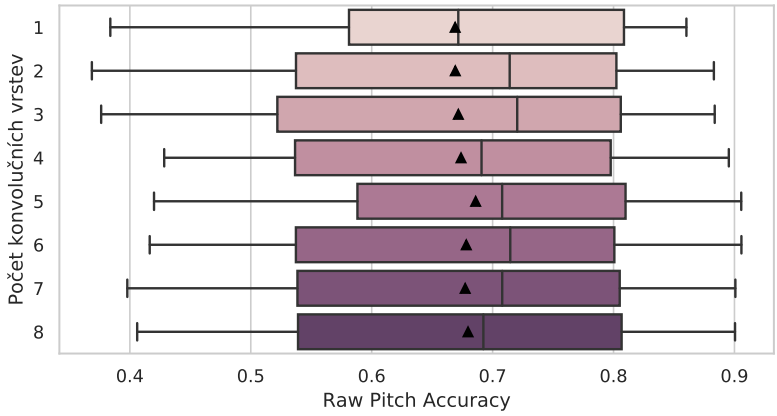
Šířka vrstev Počet vrstev	512	256	128	64	32	16	8	4	Počet parametrů
1	256								2098880
2	128	128							2066112
3	85	85	85						2041918
4	64	64	64	64					2029248
5	51	51	51	51	51				2016350
6	42	42	42	42	42	42			2001944
7	36	36	36	36	36	36	36		1996184
8	32	32	32	32	32	32	32	32	2000448

Tabulka 5.7: Počet filtrů prvních vrstev multirezoluční vstupní konvoluční vrstvy.

Experiment jsem provedl na síti se vstupním oknem 2048 vzorků, multiplikátorem kapacity 8,

Počet konvolučních vrstev	RPA	RCA
1	0.669	0.779
2	0.669	0.773
3	0.672	0.773
4	0.674	0.778
5	0.686	0.781
6	0.678	0.780
7	0.677	0.779
8	0.680	0.778

Tabulka 5.8: Architektura CREPE, vliv multirezoluční vstupní konvoluční vrstvy.



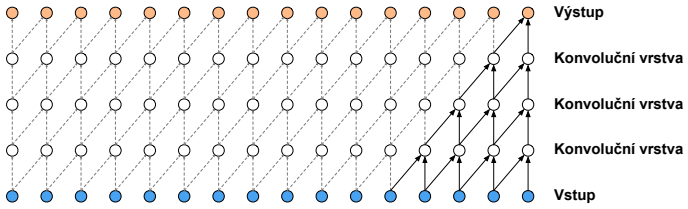
Obrázek 5.7: Architektura CREPE, vliv multirezoluční vstupní konvoluční vrstvy.

Zlepšení výsledků se pohybuje v řádu desetin procentních bodů, tedy není příliš vysoké. Zlepšení je nejvíce patrné v případě pěti různě širokých konvolučních vrstev, kde dosahuje 1.3 procentního bodu. Analýzou výsledků přesnosti podle výšky noty se mi nepodařilo prokázat domněnku, že by konvoluce s více rozlišeními pomáhala u odhadu not vyšších frekvencí. Její přínos je drobný a projevuje se na většině frekvenčních pásem.

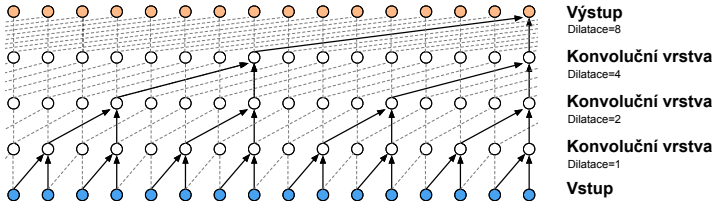
5.2 Architektura WaveNet

Generativní model WaveNet popsaný týmem van den Oord a kol. (2016a) je architektura navržená pro generování zvukového signálu. Autoři však v článku zmiňují, že se architekturu pokusili využít i pro převod mluvené řeči na text (dataset TIMIT), a podařilo se jim dosáhnout výsledků srovnatelných se state-of-the-art. Architektura spočívá ve vrstvení dilatovaných konvolucí s rozšiřujícím se rozsahem. Díky exponenciálně rostoucím dilatacím se také exponenciálně zvětšuje receptivní pole jednotlivých konvolučních vrstev. Díky této vlastnosti pak například stačí pro pokrytí 1024 vzorků vstupu pouze 9 vrstev s šířkou kernelu 2 a dilatacemi 1,2,4,8 ... 512. Pokud bychom stejného receptivního pole chtěli

dosáhnout pomocí obvyklých konvolucí počet potřebných vrstev by byl lineární vzhledem k šířce pole. Vrstvení konvolucí je porovnáno na obrázcích 5.8 a 5.9. Síť tedy velmi snadno pokryje široký kontext, což je vlastnost, která je pro zpracování zvukového signálu užitečná.



Obrázek 5.8: Vrstvení obyčejných konvolucí s lineárně rozšiřovaným dosahem, obrázek převzat z van den Oord a kol. (2016a).

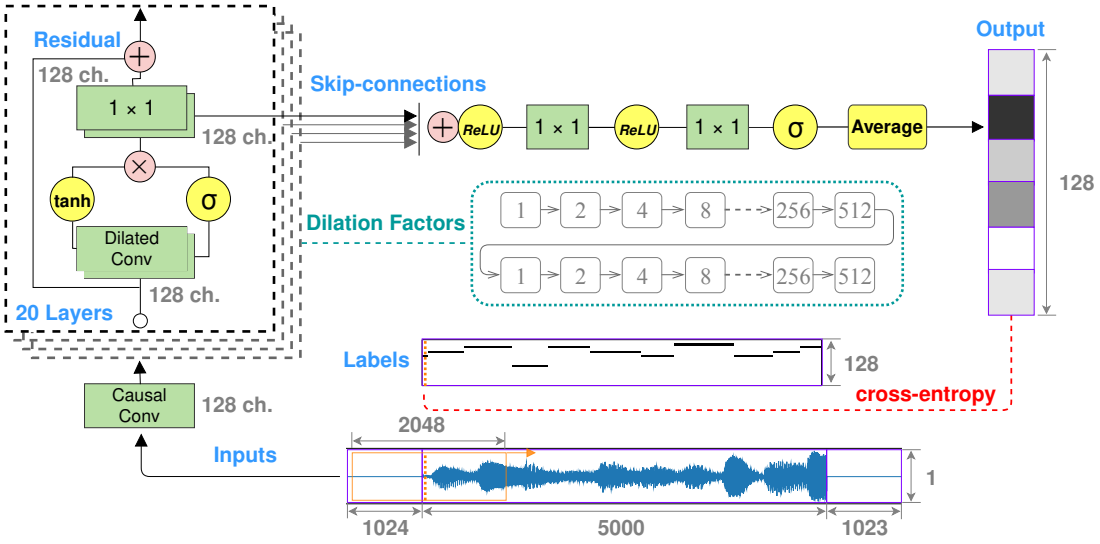


Obrázek 5.9: Vrstvení dilatovaných konvolucí s exponenciálně rozšiřovaným dosahem, obrázek převzat z van den Oord a kol. (2016a).

Síť se pro Music Information Retrieval úlohy od svého zveřejnění příliš neuchytila. Její použití se v oblasti hudby se omezuje na generativní úlohy (Hawthorne a kol. (2018), Yang a kol. (2017), Engel a kol. (2017) a další), případně pro source-separation (Stoller a kol., 2018). Jediný publikovaný pokus s použitím architektury WaveNet pro příbuznou úlohu kompletního automatického přepisu podnikli Martak a kol. (2018) s použitím datasetu MusicNet. Jejich model však netestovali na standardních evaluačních datasetech ze soutěže MIREX, tudíž není zřejmé, jakých výsledků v porovnání s existujícími metodami autoři dosáhli.

5.2.1 Baseline na základě Martak a kol. (2018)

Pro extrakci melodie využijeme jako výchozí model upravenou architekturu od Martak a kol. (2018), jejíž struktura je naznačena na obrázku 5.10. Vstupem je okno velikosti 4894 vzorků audia převzorkovaného na 16 kHz, toto okno nejprve zpracujeme standardní konvoluční vrstvou s šířkou kernelu 2 a 128 výstupními kanály, takto zpracovaný signál dále prochází dvěma bloky po desíti vrstvách, které obsahují dilatované konvoluce. Vrstvy obsahují dvě dilatované konvoluce, jednu s aktivací hyperbolickým tangens a jednu s aktivační funkcí sigmoid. Výstupy těchto konvolucí jsou po složkách vynásobeny, díky čemuž konvoluce s aktivací sigmoid funguje jako nastavitelná propust signálu (Gated activation unit, van den Oord a kol. (2016b)), poté je výstup opět zpracován dvěmi konvolucemi, tentokrát se šířkou kernelu 1x1. Výstup první sečteme s původním vstupem celé vrstvy (jedná se o residuální propojení poprvé popsané v práci He a kol. (2015)) a zpracujeme ho nadcházející vrstvou. Výstup druhé konvoluce (autoři tuto cestu



Obrázek 5.10: Architektura WaveNet upravená pro kompletní přepis skladeb, upraveno na základě Martak a kol. (2018).

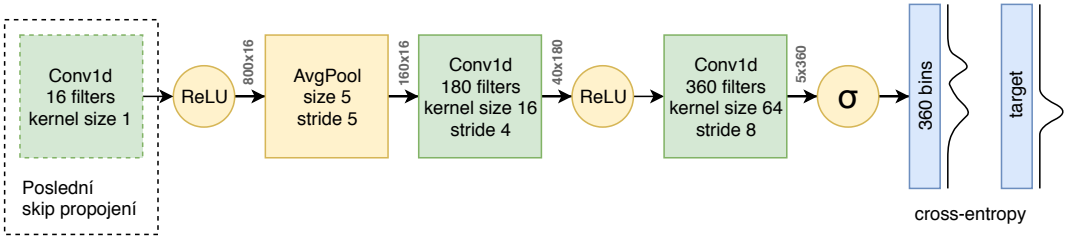
nazývají *skip propojení*) zařadíme mezi výstupy skip propojení všech ostatních vrstev.

Pro výpočet odhadů výšky tónu sečteme všechna skip propojení, tento součet zpracujeme dvěma konvolucemi, druhá z nich má na výstupu 128 kanálů zpracovaných sigmoidou, což odpovídá rozsahu odhadovaných not. Jelikož všechny popsané transformace zachovávají šířku vstupu, dostáváme po této konvoluci odhady výšek not pro každý vstupní vzorek zvuku, taková anotace je zbytečně podrobná, tudíž výsledné anotace podvzorkujeme pomocí pooling vrstvy počítající průměr hodnot.

Po 20 000 iteracích s velikostí dávky 20 a parametrem learning rate 0.001 síť dosahuje na validačních datech přesnost odhadu tónu 0.583 (RPA) a přesnost odhadu tónu nezávisle na oktávě 0.692 (RCA). V porovnání s výsledky architektury CREPE jsou tyto výsledky výrazně nižší, krom toho učení sítě trvá velmi dlouho (14 minut na 1000 iterací), zejména kvůli obrovské kapacitě a topologické komplexitě modelu (2 009 728 trénovatelných parametrů). Veliká kapacita modelu také způsobuje přeučení sítě, pokusíme se tedy zrychlit trénování a zabránit přeučení výrazným snížením počtu filtrů dilatačních a skip propojení ze 128 na 16, a použitím pouze jednoho dilatačního bloku. Také snížíme velikost trénovací dávky na 8 pro další zrychlení učení. Nová síť obsahuje 10 vrstev s dilatacemi $(1, 2, 4, 8, \ldots, 512)$, a tedy 34 736 parametrů a dosahuje RPA 0.598 a RCA 0.696 po 100 000 iteracích při rychlosti trénování 30 sekund na 1000 iterací. Síť tedy dosahuje stejných výsledků ve výrazně kratším čase.

Z předchozích experimentů na architektuře CREPE víme, že jemnější výstupní reprezentace pomáhá snížit počet chyb o půltón, upravíme tedy síť tak, aby na jeden půltón připadalo 5 výstupních složek vektoru, zmenšíme výstupní frekvenční rozsah a hodnoty cílového vektoru změníme z ostré predikce konkrétního tónu na „rozmlžený“ gaussián se směrodatnou odchylkou 18 centů (pro podrobnější popis viz 5.1). Síť po této úpravě dosahuje RPA 0.629 a RCA 0.731 po 100 000 iteracích.

Předběžným hledáním výchozí architektury pro nadcházející sadu experimentů jsme došli k následujícím úpravám. Dilatované konvoluce ve všech vrstvách mají



Obrázek 5.11: Úprava posledních vrstev WaveNet architektury.

šířku 3 místo 2, jejich vstup tedy závisí na všech okolních vzorcích, nikoli jen na předchozích, jak je naznačeno na obrázku 5.9. Dále jsme odstranili konvoluční vrstvu, která zpracovává vstup, ten je tedy přímo zpracován dilatačním blokem. Ze skip propojení se zpracovává pouze poslední, nedochází ke sčítání všech, což je úprava, kterou pro přepis řeči používají původní autoři článku WaveNet van den Oord a kol. (2016a). Tento výstup je následně zpracován pooling vrstvou a dvěmi konvolucemi se skoky (stride), viz obrázek 5.11. Tato síť dosahuje RPA 0.655 a RCA 0.759 po 100 000 iteracích a je základem pro následující experimenty.

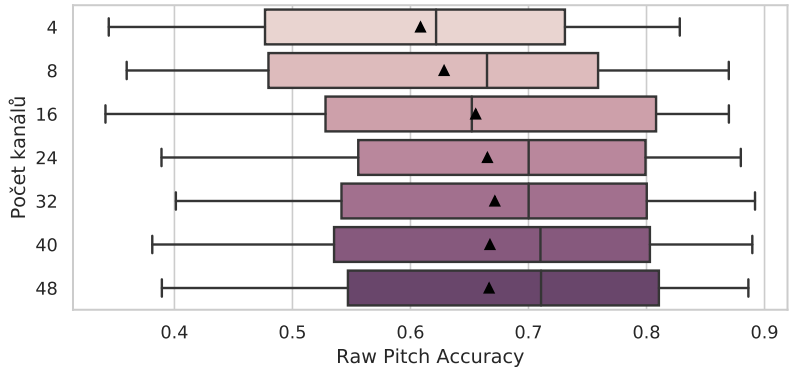
5.2.2 Vliv počtu filtrů dilatačních vrstev a skip propojení

Jedním ze zásadních faktorů ovlivňujících kapacitu sítě je počet filtrů dilatačních vrstev a skip propojení. Tyto kanály nesou informaci o vzorku a jeho okolí, v závislosti na vrstvě dilatace. Výstup poslední vrstvy má tedy délku počtu vstupních vzorků 2848 a 16 kanálů. Každý výstupní vzorek nese informaci o svém okolí délky 2047. Podobně jako v případě architektury CREPE nalezneme vhodnou kapacitu sítě tak, aby nedocházelo k přeučení (overfitting) ani k nedoučení (underfitting) na trénovací množině.

Počet kanálů	RPA	RCA
4	0.609	0.714
8	0.628	0.739
16	0.655	0.759
24	0.665	0.764
32	0.671	0.771
40	0.667	0.766
48	0.667	0.764

Tabulka 5.9: Architektura WaveNet, vliv počtu filtrů dilatačních vrstev a skip propojení.

Na základě výsledků volíme pro další experimenty nastavení 16 filtrů pro dilatované konvoluce a skip propojení. Při nastavení 24 a více filtrů dosažená přesnost sítě stagnuje, 16 filtrů je kompromisem z hlediska rychlosti trénování.



Obrázek 5.12: Architektura WaveNet, vliv počtu filtrů dilatačních vrstev a skip propojení.

5.2.3 Systematické prohledávání počtu dilatačních vrstev a bloků

Velikost zpracovávaného kontextu lze v architektuře WaveNet ovlivnit třemi různými hyperparametry. Jde o počet vrstev v bloku n_{layers}, počet dilatačních bloků poskládaných nad sebou n_{stacks} a šířka kernelu dilatací n_{width}. Přesně lze dosah vypočítat jako

$$\text{receptive\_field} = (n_{\text{width}} - 1) \cdot \Big(\sum_{d}^{n_{\text{layers}}} 2^{(d-1)} \Big) \cdot n_{\text{stacks}} + 1$$

Dosud jsme testovali sítě s jedním blokem o deseti vrstvách s dilatacemi (1,2,4,8,16,...,512) s šířkou konvolucí 3, její dosah je tedy 2047 vzorků. Systematickým prohledáváním prozkoumáme vliv delšího kontextu měněného pomocí n_{layers} a n_{stacks}. Při vyhodnocování experimentu je nutné vzít v úvahu, že přidání nové vrstvy, případně celého nového bloku, zároveň zvyšuje kapacitu modelu, což také ovlivňuje výslednou přesnost modelu. Šířky kontextu a kapacity jsou uvedené v tabulce 5.10.

V architektuře modelu jsme pro tento experiment provedli změnu ve zpracování skip propojení. V této sadě se nebere poslední skip propojení, nýbrž všechna se spojí v ose kanálů a slouží jako vstup pro následující pooling vrstvu a konvoluci. Z toho důvodu má model mnohem více parametrů, na druhou stranu je zajištěno, že poslední vrstvy mohou využít informace ze všech skip spojení k odhadu výšek. Na základě srovnání tréninkové a validační ztrátové funkce však modely nevykazují známky přeučení.

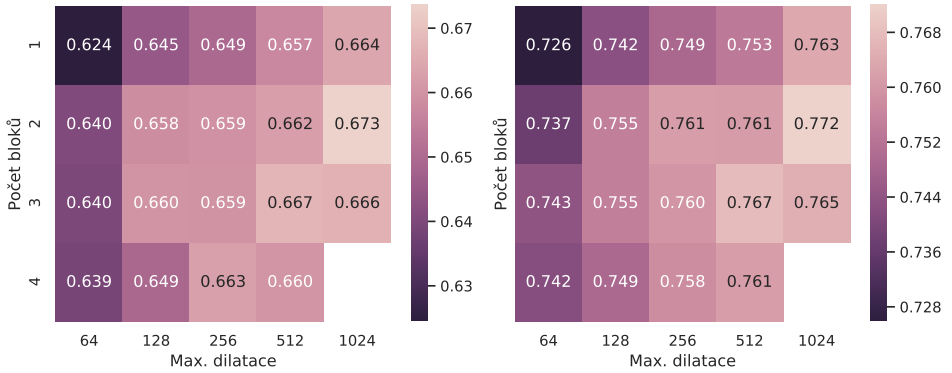
Z tabulky lze dojít k pozorování, že přidávání počtu vrstev zvyšuje výslednou přesnost víceméně vždy, to neplatí o počtu bloků, u kterých se zdá, že nejvhodnější počet je 2. Pokud se zaměříme na srovnání výsledků s podobným dosahem, zdá se že výsledky jsou poměrně podobné, zejména pro širší dosahy, pro kratší se zdá lepší využít spíše více vrstev než více bloků.

5.2.4 Vliv velikosti šířky kernelu dilatací

Jiným způsobem, jak zvýšit velikost zpracovávaného kontextu, je volba velikosti šířky kernelu dilatací n_{width}. Provedeme čtyři experimenty s různou šířkou.

Max. dilatace Počet bloků	64	128	256	512	1024
1 (dosah)	255	511	1023	2047	4095
1 (kapacita)	4 483 644	4 531 836	4 580 028	4 628 220	4 676 412
2 (dosah)	509	1021	2045	4093	8189
2 (kapacita)	4 820 988	4 917 372	5 013 756	5 110 140	5 206 524
3 (dosah)	763	1531	3067	6139	12 283
3 (kapacita)	5 158 332	5 302 908	5 447 484	5 592 060	5 736 636
4 (dosah)	1017	2041	4089	8185	—
4 (kapacita)	5 495 676	5 688 444	5 881 212	6 073 980	—

Tabulka 5.10: Architektura WaveNet, dosah a kapacita v závislosti na dilatačních počtu vrstev a bloků.



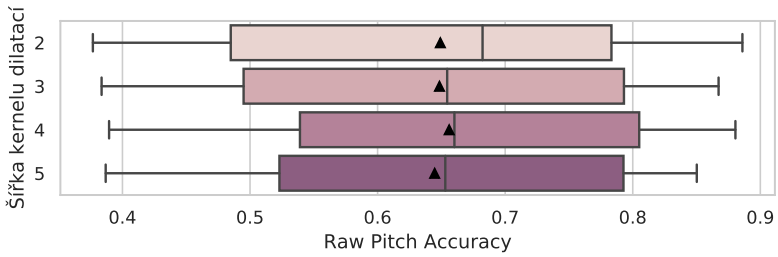
Obrázek 5.13: Architektura WaveNet, systematické prohledávání počtu dilatačních vrstev a bloků, vlevo hodnoty RPA, vpravo RCA.

Šířka 2 je zvolena v původním článku z toho důvodu, že konvoluce se tím stávají „kauzální“ — jejich výstup závisí pouze na vzorcích před nimi (viz obrázky 5.8, 5.9). To je výhodné pro generativní úlohy, u kterých chceme, aby hodnota nového generovaného vzorku v audio signálu nezávisela na budoucích, zatím neexistujících vzorkách. U klasifikačních úloh takto omezeni nejsme, tudíž můžeme s nastavením experimentovat.

Šířka kernelu dilatací	RPA	RCA
2	0.649	0.754
3	0.648	0.755
4	0.656	0.759
5	0.645	0.756

Tabulka 5.11: Architektura WaveNet, vliv velikosti šířky kernelu dilatací.

Z tabulky 5.11 a grafu 5.14 vyplývá, že tato síť při změnách šířky kernelu dilatace svou výslednou přesnost na validačních datech nemění.



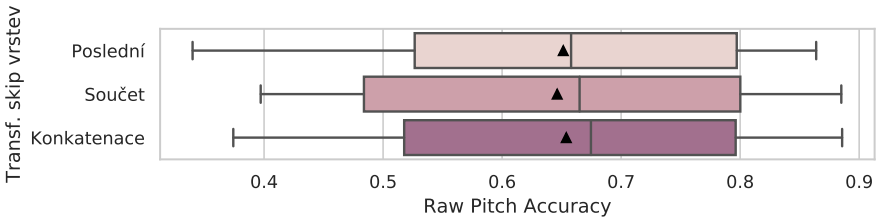
Obrázek 5.14: Architektura WaveNet, vliv velikosti šířky kernelu dilatací.

5.2.5 Vliv výstupní transformace skip propojení

Volba transformace skip propojení se v článku týmu van den Oord a kol. (2016a) nediskutuje, Martak a kol. (2018) také přejímají součet všech skip výstupů. Vyzkoušíme proto dvě další možnosti práce se skip propojeními - výběr posledního skip propojení a dále spojení všech skip propojení do mnohakanálového výstupu. V případě výběru posledního propojení či součtu jde o transformace, které lze vyjádřit jako speciální případ konvoluce nad spojenými skip propojeními. Dalo by se tedy říct, že výběr a součet jsou v tomto případě speciálními případy spojených výstupů. Testovaná možnost výběr poslední vrstvy skip propojení naopak přináší jinou výhodu - pro výpočet výsledku sítě nejsou potřeba zbylá skip propojení, což urychluje trénování.

Transf. skip vrstev	RPA	RCA
Konkatenace	0.654	0.754
Poslední	0.651	0.754
Součet	0.646	0.753

Tabulka 5.12: Architektura WaveNet, vliv výstupní transformace skip propojení.



Obrázek 5.15: Architektura WaveNet, vliv výstupní transformace skip propojení.

Všechny tři transformace vedou ke stejné přesnosti. Síť tedy netěží z větších možností práce s dřívejšími vrstvami a nejdůležitejší informace se objevují až na vrchu bloků dilatovaných konvolucí.

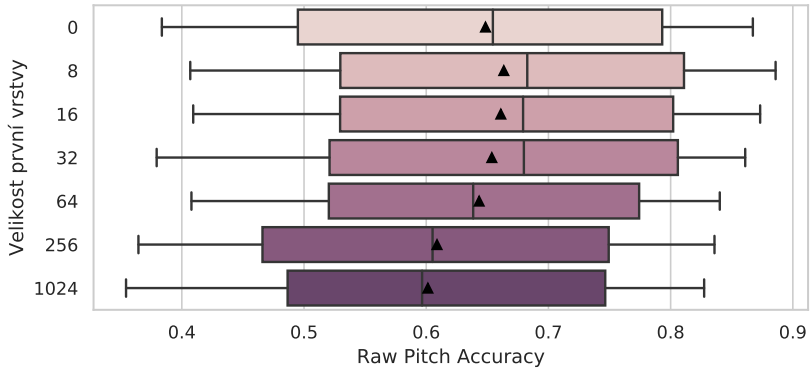
5.2.6 Vliv velikosti první konvoluce

Tým van den Oord a kol. (2016a) používá pro předzpracování zvukového signálu před vstupem do dilatovaných bloků obvyklou konvoluci, nespecifikuje však

její šířku. Veřejná implementace architektury WaveNet pracuje s šířkou 32 [1]. Martak a kol. (2018) používají šířku 2, čímž fakticky jen duplikují první dilatační vrstvu. První vrstva může sloužit jako banka filtrů, podobně jako první vrstva v architektuře CREPE. Aby však pro tento účel mohla fungovat plnohodnotně, musela by být široká minimálně 512 vzorků, aby mohla zachytit periodu i té nejnižší frekvence ve výstupním rozsahu. Pomocí sady experimentů se pokusíme zjistit, zda má první konvoluční vrstva pozitivní vliv na výslednou přesnost.

Velikost první vrstvy	RPA	RCA
0	0.648	0.755
8	0.663	0.762
16	0.661	0.765
32	0.654	0.754
64	0.643	0.756
256	0.609	0.735
1024	0.601	0.734

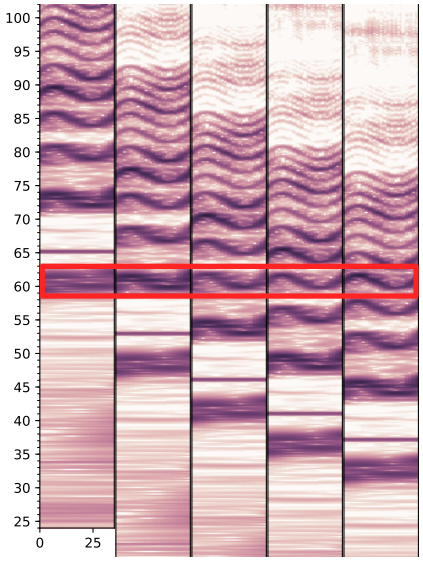
Tabulka 5.13: Architektura WaveNet, vliv velikosti první konvoluce.



Obrázek 5.16: Architektura WaveNet, vliv velikosti první konvoluce.

Sítě, které mají možnost využít první konvoluce jako banky filtrů, dosahují výrazně horší přesnosti, než sítě s menší konvolucí. Přeskočení tohoto předzpracování však také není nejlepším nastavením pro tuto architekturu. Vhodná šířka první konvoluce se pohybuje kolem 8 vstupních vzorků, do takové šířky se vejde jedna perioda signálu frekvence 2000 Hz, jinými slovy vrstva jistě nemůže zastávat ani z části funkci banky filtrů. Mohla by však popisovat sklon křivky vstupních dat, ze kterých se v dalších vrstvách dilatovaných bloků složí již celý frekvenční popis vstupu.

vizualizace první vrstvy? Co tam vlastně může být, když pomáhá šestnácti samplová konvoluce?



Obrázek 5.17: Znázornění harmonických závislostí na spektrogramu s logaritmickou osou frekvence.

5.3 Architektura HCNN

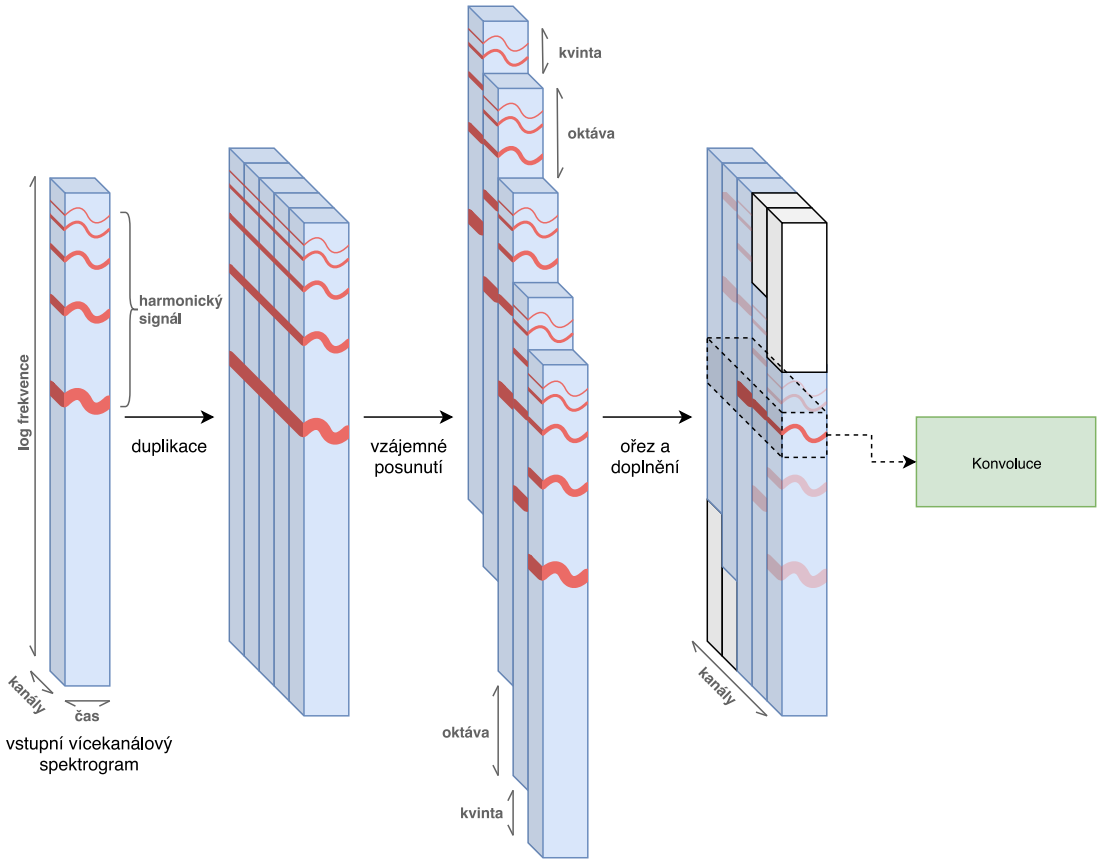
V práci prezentujeme také novou architekturu pro zpracování harmonicky strukturovaných zvukových signálů nazvanou *Harmonic Convolutional Neural Network*. Inspirujeme se prací Bittner a kol. (2017), která problém výpočtu salienční funkce zasazuje do rámce úlohy odstranění šumu v obrázku pomocí konvolučních sítí. Její architektura vstupní spektrogram zpracovává několika konvolučními vrstvami, výsledkem pak je nový spektrogram s odstraněným „šumem hudebního doprovodu“.

Dalším přínosem její práce je popis vstupní spektrální reprezentace HCQT. CQT spektrogramy mají logaritmickou osu frekvence, tato reprezentace má tu vlastnost, že dvě libovolné harmonické frekvence jednoho tónu mají na spektrogramu mezi sebou konstantní vzdálenost, nezávislou na fundamentální frekvenci. HCQT spočívá ve výpočtu několika CQT spektrogramů, jejichž frekvenční rozsahy jsou vzájemně posunuty právě o tyto konstatní vzdálenosti. Pokud pak tyto spektrogramy poskládáme za sebe v nové ose kolmé na osu frekvence a času, harmonické frekvence se překryjí (viz obrázek 5.17). Tento vícekanálový spektrogram pak může sloužit jako vstup pro konvoluční vrstvu, která má při výpočtu jedné frekvenční složky k dispozici také informace o souvisejících harmonických složkách.

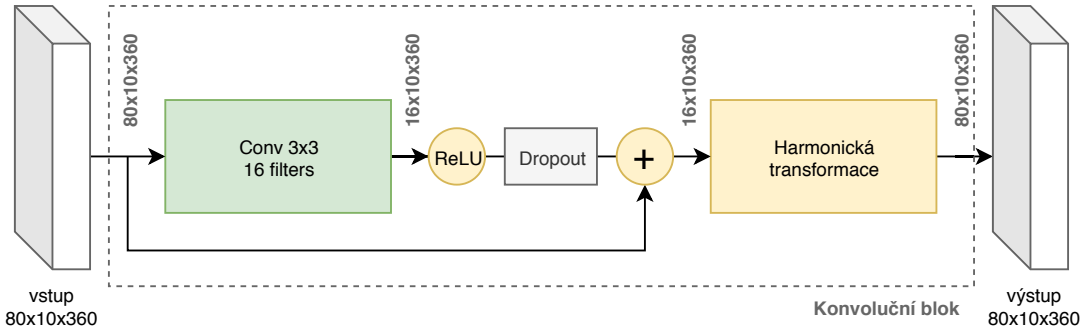
Z této myšlenky úpravy vstupní reprezentace pro rozšiřování receptivního pole konvoluce na související harmonické složky vychází naše architektura HCNN. Síť je koncipována jako obvyklé vrstvení konvolučních vrstev, každému konvolučnímu bloku však předchází úprava vstupu takovým způsobem, aby do výpočtu nadcházející konvoluce byly zahrnuty i potenciální harmonické složky dané frekvence. Naše metoda tedy nepracuje pouze s výpočtem různých CQT spektrogramů na vstupu jako práce Bittner a kol. (2017), upravujeme totiž mezivýsledky všech vrstev v síti a naše transformace spočívá v pouhém posunutí, nikoli přepočítávání.

[1] `https://github.com/ibab/tensorflow-wavenet/blob/master/wavenet_params.json`

Na obrázku 5.18 tuto úpravu vstupu znázorňujeme; výstup předchozí konvoluční vrstvy duplikujeme a vzájemně posuneme o pevně dané počty půltónů tak, aby harmonicky související složky byly umístěny „nad sebou“ v dimenzi kanálů. Následně výstupy v této dimenzi spojíme a chybějící složky doplníme nulami.

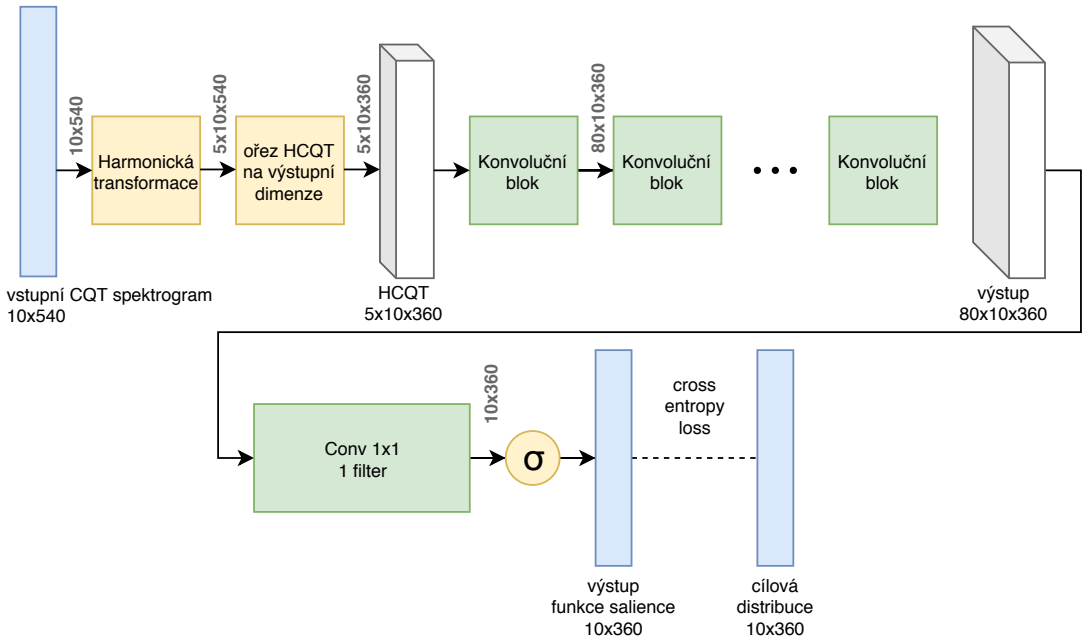


Obrázek 5.18: Diagram transformace vstupu konvoluční vrstvy pro zachycení harmonických souvislostí.



Obrázek 5.19: Diagram konvolučního bloku architektury HCNN.

Zbývající prvky architektury již čerpají ze zavedených postupů. Diagram 5.19 znázorňuje strukturu jednoho konvolučního bloku; po každé konvoluční vrstvě s aktivační funkcí ReLU následuje dropout vrstva a kolem konvoluce vedeme residuální propojení pro umožnění trénování hlubokých architektur. Celková architektura (obrázek 5.20) pak spočívá v harmonické transformaci vstupní CQT (čímž



Obrázek 5.20: Diagram celkového propojení architektury HCNN.

efektivně vytvoříme reprezentaci odpovídající HCQT bez přepočítávání každého CQT zvlášť), aplikaci několika konvolučních bloků a aplikaci konvoluce 1x1 pro transformaci výstupu zpět na jeden dvoudimenzionání výstup.

Vstupní CQT počítáme z původních audio souborů se vzorkovací frekvencí 44 100 Hz, používáme implementaci algoritmu z balíku `librosa` a parametry transformace jsou: počáteční frekvence (`fmin`) = 32.7 (Tón C1), počet složek na oktávu (`bins_per_octave`) = 60 (vychází 5 složek na půltón, jako v předchozích sekcích), počet složek (`n_bins`) = 540 (vychází na 9 oktáv, po první harmonické transformaci však spektrogram ořízneme na výšku 360, zbylé složky jsou přítomny, aby zaplnily chybějící hodnoty při harmonických posunech), `hop_size`=256 (tudíž jedno anotační okno MedleyDB odpovídá jednomu sloupci vstupního spektrogramu). Dále pak vstup převedeme na logaritmickou škálu dBFS (funkce `librosa.core.ampl`

Cílový výstup počítáme v souladu s implementacemi v předchozích sekcích, tedy použijeme normální rozdělení se střední hodnotou rovnou výšce znějící melodie a rozptylem 18 centů.

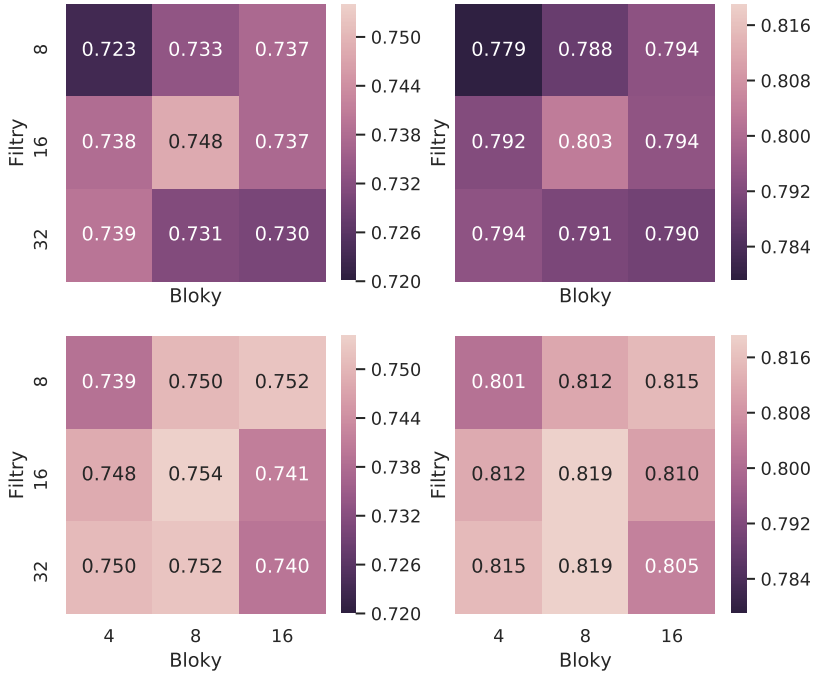
5.3.1 Harmonické transformace

Abychom prokázali pozitivní vliv harmonické transformace, natrénujeme tři sady modelů. Jednu sadu bez jakýchkoliv transformací, sadu s transformací s posunutím o +-12 půltónů (zachycení druhé harmonické složky) a třetí sadu s posunutím o +-12 a o +-19 půltónů (zachycení druhé a třetí harmonické složky). Vstupem všech sítí bude HCQT (implementovaný efektivně pomocí harmonických posunů), aby byl posouzen vliv transformací uvnitř sítě, nikoli také na vstupu.

5.3.2 Velikost okna

Metoda Bittner a kol. (2017) pracuje se spektrogramy CQT vypočítanými s velikostí okna ≈ 11 ms. Data MedleyDB, která v práci používáme, jsou však

anotována s velikostí okna ≈ 5.8 ms. To znamená, že při trénování musíme anotace v trénovací množině MedleyDB podvzorkovat a podobně při vyhodnocování výsledků sítě musíme výstup převzorkovat zpět na původní časové rozlišení datasetu. V obou případech převzorkování můžeme ztratit informace, které se projeví v horší výsledné přesnosti. Je také otázkou, zda-li je nejlepší postup, pokud síť budeme trénovat na neoriginálních, převzorkovaných datech. Výhodou většího okna je větší receptivní pole konvoluce, kernel o šířce větší než 1 má na podvzorkovaných datech dvojnásobný dosah. Tento efekt lze nicméně napodobit nastavením dvojnásobné dilatace konvoluce. Na experimentech s kernely šířky 1 posoudíme dopad převzorkování bez této zmiňované výhody.

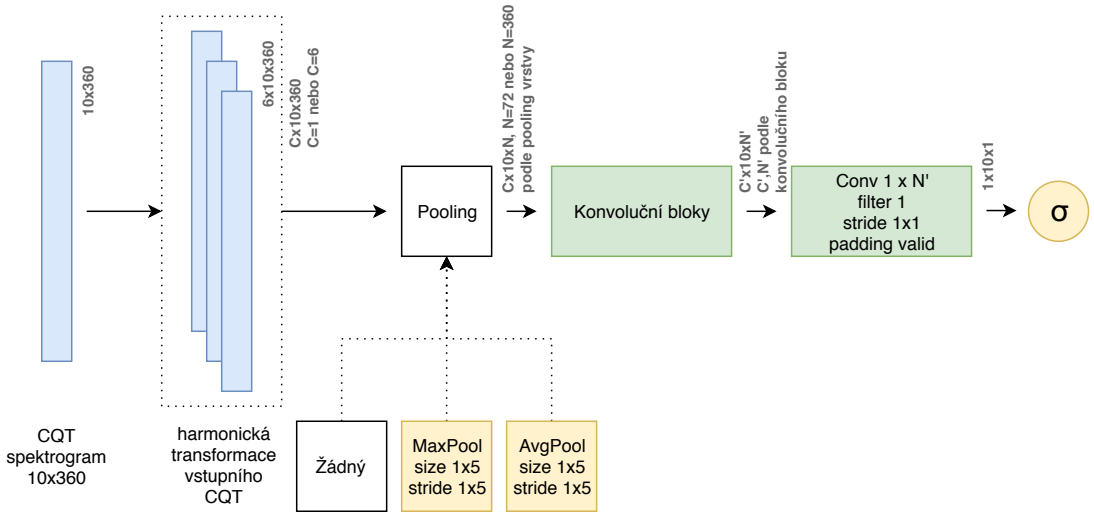


Obrázek 5.21: Vliv velikosti okna, horní okno velikosti ≈ 11 ms, dolní okno ≈ 5.8 ms. Vlevo jsou výsledky metriky RPA, vpravo RCA.

5.4 Detekce melodie

Podobně jako v práci Rigaud a Radenen (2016) jsme otestovali možnosti oddělení modulu pro odhad výšky melodie a pro její detekci. V této sekci proto nejprve provedeme sadu experimentů, ve kterých návrhy na modul pro detekci trénujeme samostatně a poté otestujeme jeho přesnost také v kombinaci s modulem pro odhad výšky.

Při hledání nejlepší architektury pro detekci jsme postupovali iterativním prohlubování návrhů sítí. Společná struktura všech experimentů je znázorněna na diagramu 5.22. Vstupem modelů je CQT spektrogram, podobně jako pro modely HCNN. Ten v některých experimentech transformujeme pomocí harmonického posunu (viz diagram 5.18). Dále tento spektrogram potenciálně zmenšíme pomocí jedné pooling vrstvy tak, aby na jeden půltón připadala jedna složka výstupu, u většiny experimetů však tuto transformaci vynecháváme. Tento upravený vstup



Obrázek 5.22: Diagram společné struktury modelů pro detekci melodie.

pak zpracováváme jedním z konvolučních bloků znázorněných na obrázku 5.23. Výstup bloku pomocí konvoluce a následné sigmoidy zredukujeme na výstupní hodnoty označující přítomnost melodie v každém ze vstupních oken.

V diagramu konvolučních bloků 5.23 obsahují bloky `dilated_block`, `octave_block` a `note_block` podsoučást `reg. blok`, tedy regularizační blok. Ten se skládá z kombinace dropout vrstvy a batch normalizace, nejvýhodnější kombinaci budeme ověřovat experimentálně. Dále také na obrázku používáme proměnné $\mathbf{T_{CTX}}$ pro parametrizaci šířky konvolucí a $\mathbf{K}$ multiplikační koeficient kapacity. Nastavení těchto proměnných budeme také porovnávat v následujících podsekcích. Dále se v diagramech používá proměnná N, která je vázaná na šířku vstupu daného bloku.

Úlohu detekce formulujeme jako binární klasifikaci pro každé uvažované vstupní okno signálu. Síť je trénována pomocí vzájemné korelace mezi výstupem a cílovou množinou binárních veličin, které indikují přítomnost melodie.

5.4.1 Vliv počáteční pooling vrstvy

Motivací pro použití pooling vrstvy je zmenšení nejspíše zbytečně podrobné frekvenční dimenze vstupu. Protože výstup úlohy detekce melodie nemusí obsahovat přesnou informaci o výšce tónu, pouze binární informaci o její přítomnosti v daném časovém okamžiku, jemná frekvenční osa se zdá redundantní a ponechání plného rozlišení vede ke zbytečně větším modelům. Pomocí experimentů na sadě architektur se proto pokusíme ověřit vliv této redukce.

Z tabulky 5.14 a obrázku 5.24 vidíme, že u většiny experimentů zpracování vstupu pooling vrstvou spíše zhoršilo, v dalších experimentech proto zpracování pooling vrstvou nepoužíváme.

5.4.2 Vliv použití dropout jako regularizace

Sadou experimentů prozkoumáme účinnost regularizace pomocí dropout vrstvy po každé konvoluční vrstvě v modelu. Parametr pravděpodobnosti vynechání složky jsme nastavili na 0.3.

Model	Pooling	VR	VFA	VA
full_1layer	None	0.818	0.606	0.616
full_1layer	avg	0.820	0.600	0.616
full_1layer	max	0.830	0.589	0.628
note_1layer	None	0.722	0.451	0.636
note_1layer	avg	0.751	0.547	0.605
note_1layer	max	0.733	0.477	0.626
note_dilated	None	0.720	0.441	0.636
note_dilated	avg	0.680	0.473	0.595
note_dilated	max	0.668	0.413	0.617
note_note	None	0.701	0.401	0.648
note_note	avg	0.722	0.519	0.602
note_note	max	0.682	0.436	0.619
octave_1layer	None	0.789	0.523	0.638
octave_1layer	avg	0.793	0.527	0.634
octave_1layer	max	0.809	0.561	0.628
octave_octave	None	0.588	0.337	0.617
octave_octave	avg	0.709	0.454	0.624
octave_octave	max	0.740	0.473	0.632

Tabulka 5.14: Detekce melodie, vliv počáteční pooling vrstvy.

Model	Dropout	VR	VFA	VA
full_1layer	0.0	0.782	0.565	0.618
full_1layer	0.3	0.825	0.632	0.610
octave_1layer	0.0	0.610	0.357	0.625
octave_1layer	0.3	0.832	0.586	0.629
note_1layer	0.0	0.691	0.418	0.634
note_1layer	0.3	0.791	0.519	0.642
octave_octave	0.0	0.588	0.337	0.617
octave_octave	0.3	0.824	0.564	0.636
note_note	0.0	0.743	0.438	0.648
note_note	0.3	0.758	0.424	0.666
note_dilated	0.0	0.720	0.441	0.636
note_dilated	0.3	0.686	0.362	0.656

Tabulka 5.15: Detekce melodie, vliv dropout vrstvy.

Výsledky experimentů můžeme porovnat na obrázku 5.25 a v tabulce 5.15, dropout vrstva výrazně pomáhá zvýšit přesnost modelu, zejména pak u architektur se dvěma vrstvami. V následujících experimentech dropout proto používáme.

5.4.3 Vliv použití batch normalizace jako regularizace

Sadou experimentů prozkoumáme účinnost regularizace pomocí batch normalizace po každé konvoluční vrstvě v modelu.

Z výsledků na obrázku 5.26 a v tabulce 5.16 vyplývá, že batch normalizace

Model	Batch normalizace	VR	VFA	VA
full_1layer	Ne	0.849	0.678	0.598
full_1layer	Ano	0.825	0.632	0.610
note_1layer	Ne	0.767	0.485	0.644
note_1layer	Ano	0.791	0.519	0.642
note_dilated	Ne	0.720	0.426	0.646
note_dilated	Ano	0.686	0.362	0.656
note_note	Ne	0.761	0.439	0.659
note_note	Ano	0.758	0.424	0.666
octave_1layer	Ne	0.841	0.557	0.647
octave_1layer	Ano	0.832	0.586	0.629
octave_octave	Ne	0.845	0.607	0.631
octave_octave	Ano	0.824	0.564	0.636

Tabulka 5.16: Detekce melodie, vliv batch normalizace.

přesnost sítí mírně zlepšuje, proto ji v následujících experimentech používáme společně s dropout regularizací.

5.4.4 Vliv kapacity sítě

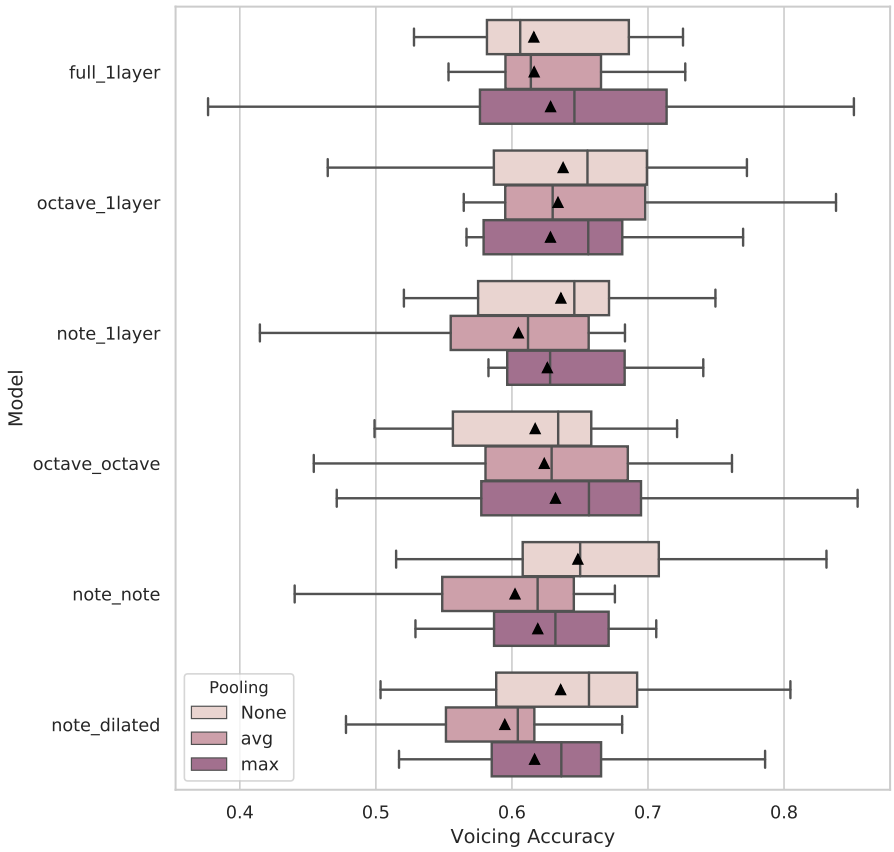
Konstrukcí a natrénováním sítí s různou kapacitou nastavenou pomocí parametru **K** na diagramu 5.23 ověříme, jaký počet konvolučních filtrů je pro úlohu a danou architekturu vhodný.

Model	Mult. Koef. Kapacity	VR	VFA	VA
full_1layer	8	0.825	0.632	0.610
full_1layer	16	0.923	0.765	0.601
full_1layer	32	0.858	0.690	0.600
note_1layer	8	0.791	0.519	0.642
note_1layer	16	0.845	0.583	0.641
note_1layer	32	0.829	0.565	0.641
note_dilated	8	0.686	0.362	0.656
note_dilated	16	0.766	0.474	0.649
note_dilated	32	0.694	0.395	0.646
note_note	8	0.758	0.424	0.666
note_note	16	0.711	0.368	0.662
note_note	32	0.676	0.338	0.659
octave_1layer	8	0.832	0.586	0.629
octave_1layer	16	0.842	0.588	0.636
octave_1layer	32	0.859	0.612	0.634
octave_octave	8	0.824	0.564	0.636
octave_octave	16	0.799	0.487	0.659
octave_octave	32	0.648	0.343	0.640

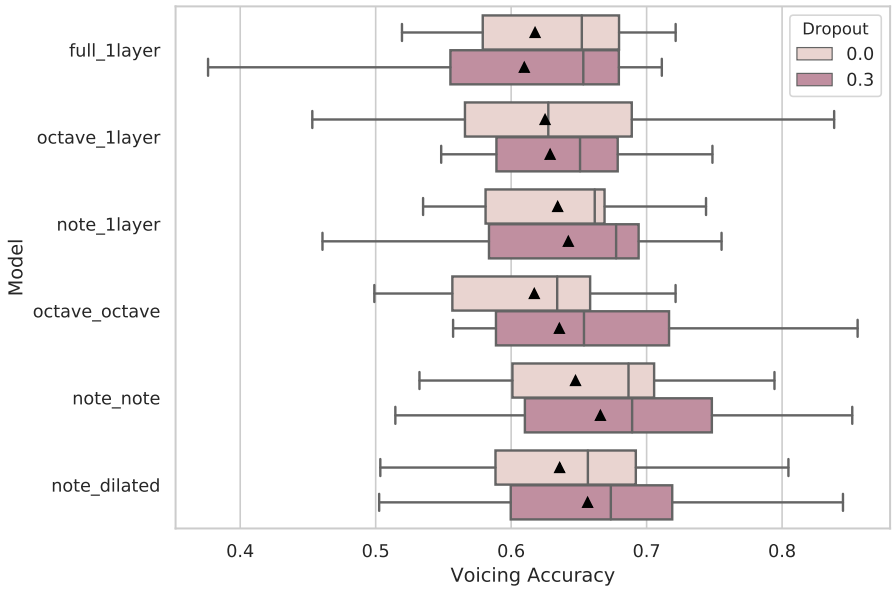
Tabulka 5.17: Detekce melodie, vliv batch normalizace.

5.4.5 Vliv kontextu

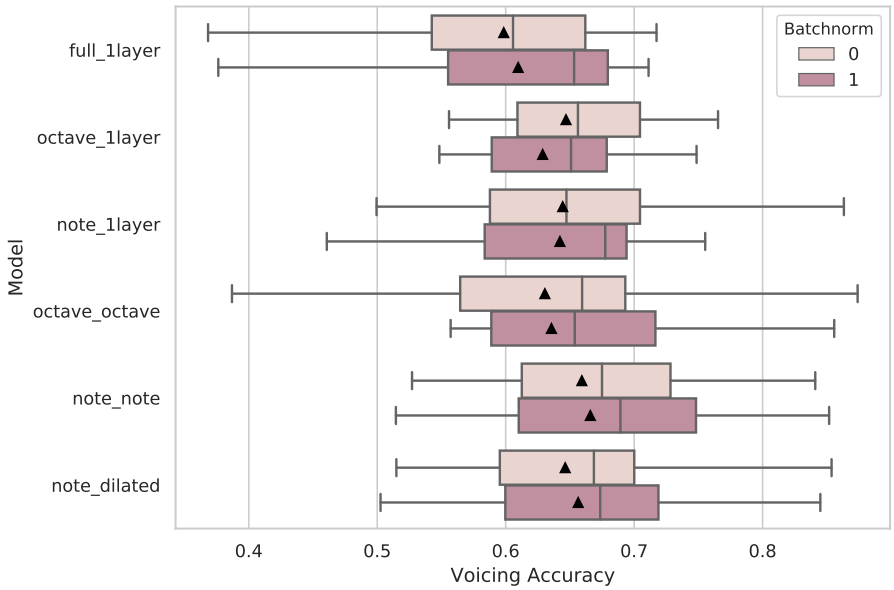
Obrázek 5.23: Diagram testovaných druhů konvolučních bloků pro detekci melodie.



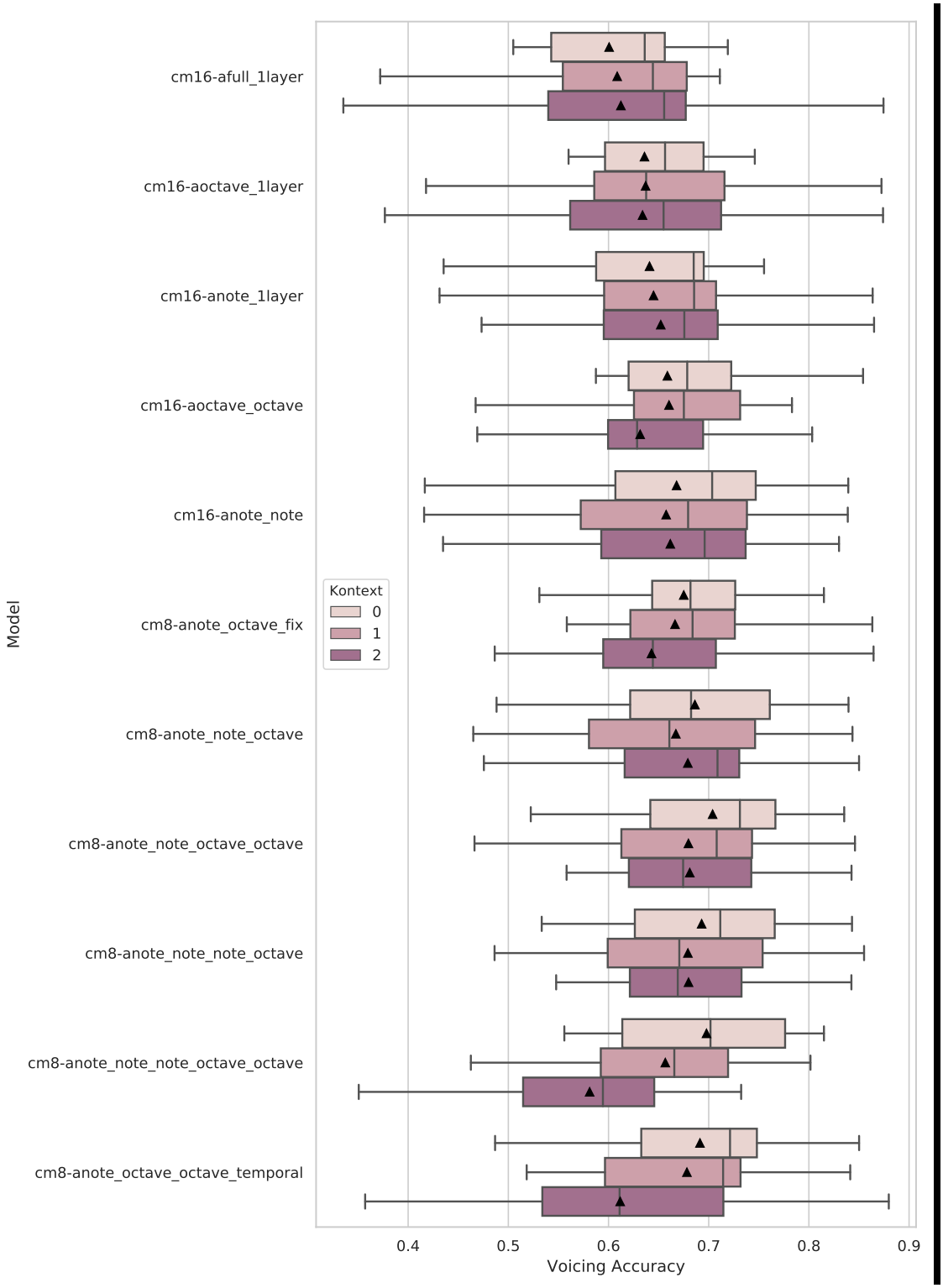
Obrázek 5.24: Detekce melodie, vliv počáteční pooling vrstvy.



Obrázek 5.25: Detekce melodie, vliv dropout vrstvy.



Obrázek 5.26: Detekce melodie, vliv batch normalizace.



Obrázek 5.27: Detekce melodie, vliv kontextu.

6. Výsledky

V této kapitole prezentujeme výsledky vybraných architektur z kapitoly Experimenty na testovacích datech a provádíme kvantitativní a kvalitativní srovnání se state-of-the-art systémy pro extrakci melodie, představené v kapitole Související práce.

6.1 Výběr testovaných modelů

Pro srovnání jsme vybrali nejlepší modely od každé testované architektury, pomocí provedených experimentů prezentujeme v tabulkách pro každou architekturu nejlepší nalezené nastavení hyperparametrů.

6.1.1 Architektura CREPE

Zvolený model dosáhl na validačních datech přesnosti přepisu výšky 0.689 a přesnosti přepisu výšky nezávisle na oktávě 0.784. Počet trénovatelných parametrů tohoto modelu je 7 816 640, trénovaní probíhalo téměř 9 hodin, každý trénovací příklad byl síti předložen 10 krát (proběhlo 10 epoch trénování).

Prohledávané parametry		Ostatní parametry	
Parametr	Hodnota	Parametr	Hodnota
Multiplikační koef. kapacity	16	Velikost dávky	32
Rozlišení diskretizace výšky noty	5	Iterace trénování	360000
Rozptyl distribuce výšky noty	0.177	Learning rate	0.001
Šířka vstupního okna	4096		
Násobné rozlišení první vrstvy	6 vrstev		

6.1.2 Architektura WaveNet

TODO vybrat

6.1.3 Architektura HCNN

TODO dotrénovat nějaký lol

6.1.4 Architektura pro detekci melodie

- modul detekce melodie bude mít k dispozici výstup HCNN

6.2 Kvantitativní srovnání

Tohle je tak trochu placeholder, protože ještě čekám na nějaká čísla

Metoda	ADC04	MDB-mel-s test	MIREX05 train.	MDB test	ORCHSET	WJazzD test
Salamon	0.767	0.514	0.761	0.526	0.281	0.693
Bittner	0.814	0.606	0.807	0.670	0.519	0.774
Basaran	0.793	0.733	0.798	0.706	0.635	0.767
CREPE	0.793	0.542	0.783	0.607	0.395	0.784
WaveNet	0.799	0.532	0.790	0.595	0.363	0.761
Bittner Improved	0.856	0.707	0.855	0.734	0.595	0.806

Tabulka 6.1: Výsledky přesnosti přepisu (Raw Pitch Accuracy).

Metoda	ADC04	MDB-mel-s test	MIREX05 train.	MDB test	ORCHSET	WJazzD test
Salamon	0.807	0.639	0.805	0.659	0.568	0.757
Bittner	0.855	0.666	0.824	0.735	0.694	0.785
Basaran	0.820	0.766	0.807	0.757	0.776	0.776
CREPE	0.853	0.614	0.817	0.713	0.606	0.810
WaveNet	0.850	0.604	0.826	0.706	0.573	0.796
Bittner Improved	0.894	0.746	0.872	0.787	0.762	0.815

Tabulka 6.2: Výsledky přesnosti přepisu nezávisle na oktávě (Raw Chroma Accuracy).

6.3 Kvalitativní srovnání

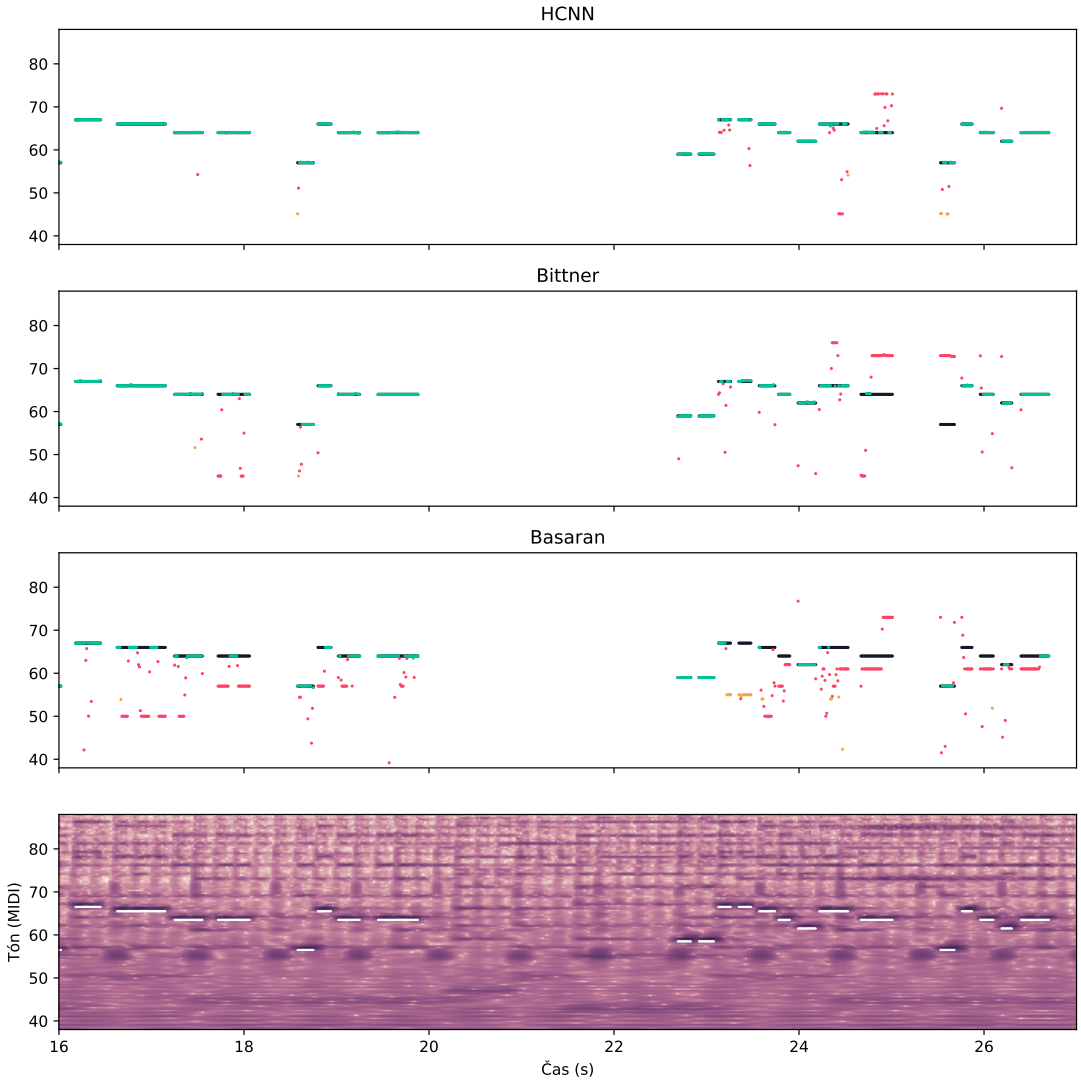
- celkově moc není příklad, na kterém je HCNN výrazně lepší než obě metody (krom train10 mirex) - Basaran je horší v jazzu kvůli kvantifikaci - zase je lepší v orchsetu kvůli lepší vstupní reprezentaci -

Pro srovnání sítí na jednotlivých příkladech jsme vybrali na základě kvantitativního vyhodnocení metody Bittnerové a Basarana. Z testovaných architektur vybíráme HCNN, jelikož výsledky této metody v přepisu melodie předčily stávající metody na většině testovacích datasetů. Pro srovnání vybíráme příklady, ve kterých se výstupy sítí nejvíce lišily.

Metrika (Metoda)	train10
RPA (HCNN)	0.917
RCA (HCNN)	0.934
RPA (Bittner)	0.863
RCA (Bittner)	0.868
RPA (Basaran)	0.512
RCA (Basaran)	0.572

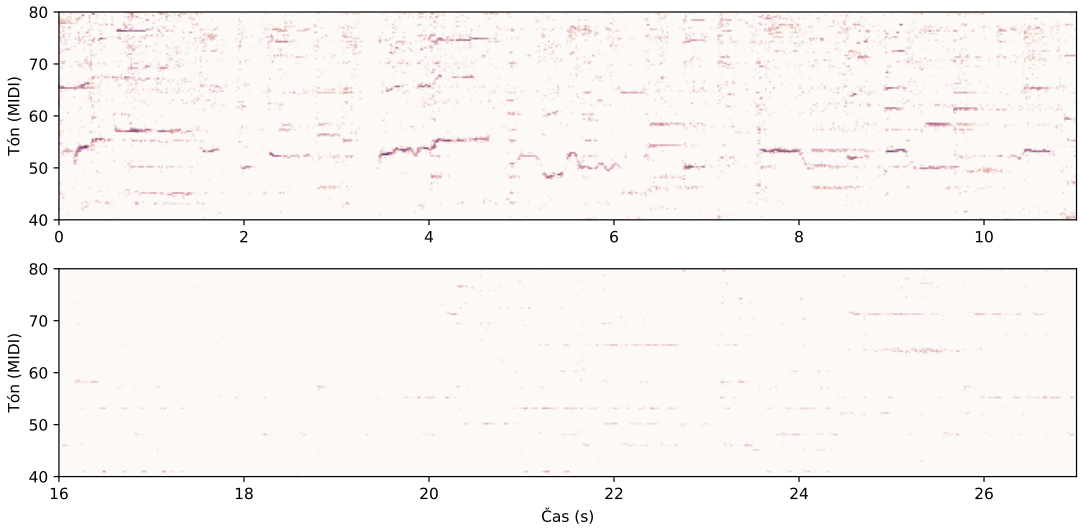
Tabulka 6.3: Přesnost metod na testovacím souboru `train10` z datasetu MIREX05.

Největší slabinou sítě HCNN se stala podle očekávání časová kontinuita odhadů. Protože síť pro odhad výšek uvažuje vždy pouze 5.8 ms a po vytvoření



Obrázek 6.1: Výstup metod na testovacím souboru `train10` z datasetu MIREX05.

funkce salience na odhady tónů neaplikujeme žádné způsoby vyhlazování, odhady jednotlivých časových oken na sebe nenavazují. To se nejeví jako zásadní problém v případech, kdy ve skladbě melodii nenese více hlasů v souzvuku (viz výstup 6.1). U některých orchestrálních skladeb však vzniká problém, například pokud melodii nese zároveň sekce smyčců a dechů v různých oktávách. Jak můžeme vidět na výstupu algoritmů 6.3, HCNN pak „přeskakuje“ mezi oktávami. Problém je také dobře vidět na výstupní funkci salience 6.4, na které vidíme tři totožné kontury posunuté o oktávu. Mírné zlepšení tohoto problému vidíme na výstupech metody Bittner a kol. (2017), která podobně jako naše metoda výsledek funkce salience žádným způsobem nezpracovává, na druhou stranu její metoda pro odhad výšky uvažuje okno délky 150 ms. Na obrázku 6.3 vidíme, že množství oktávových chyb je výrazně menší — díky většímu kontextu může metoda vybrat v čase navazující odhady. Metoda týmu D Basaran, S Essid (2018) odhad výšky tónů vyhlazuje pomocí rekurentní architektury GRU, tudíž jejich výstup obsahuje nejméně skoků. Použití rekurentní sítě dovoluje zachytit ještě dlouhodobější závislosti a výstupní kontura pak jednak často obsahuje nejmenší množství velmi krátkých, chybných skoků mimo hlavní melodii, které se na ob-



Obrázek 6.2: Srovnání vstupní frekvenčně-časové reprezentace $\mathbf{H}^{F_0}$ Basaranovy metody testovacích souborů `train01` a `train10` z datasetu MIREX05.

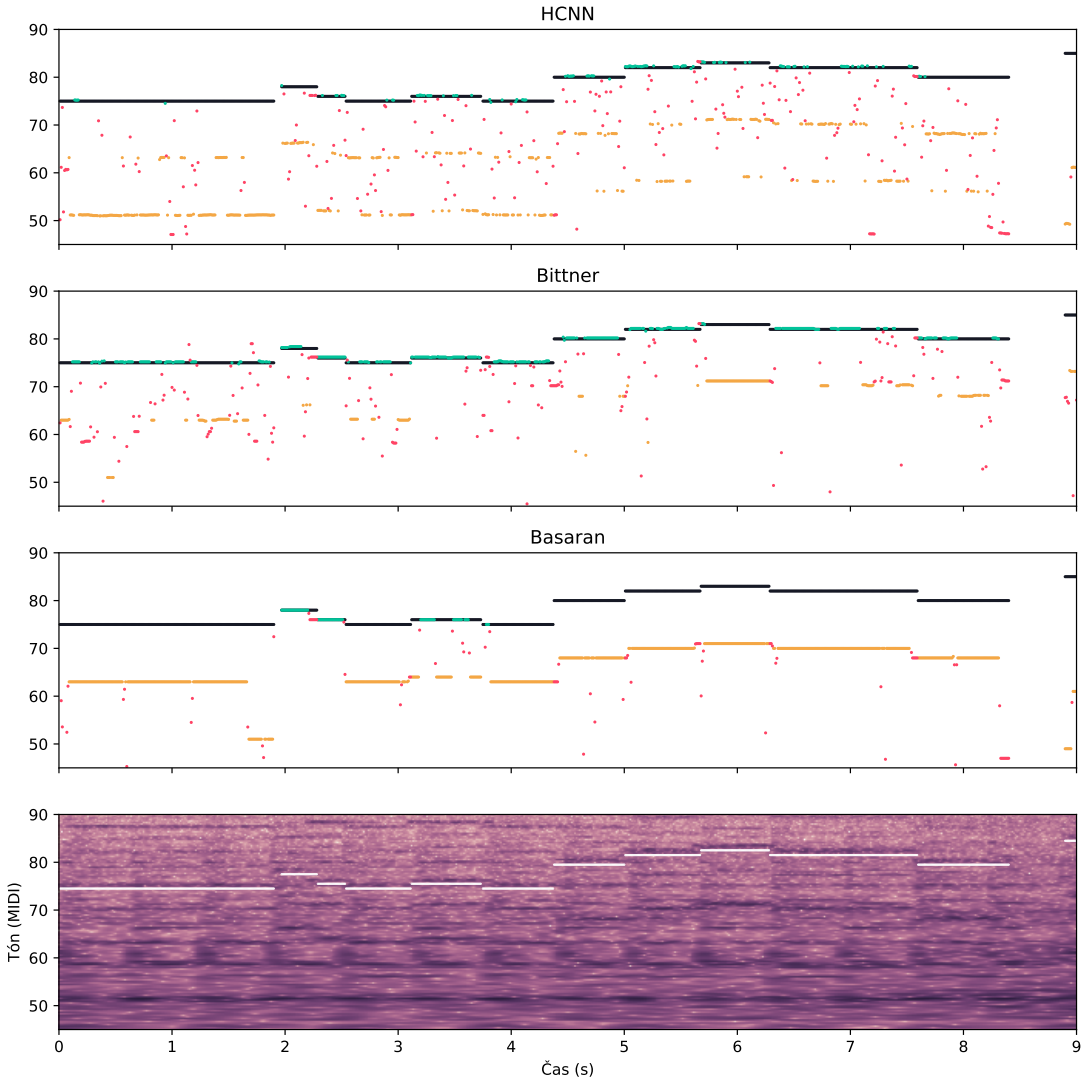
rázku 6.3 hojně vyskytují u metod bez vyhlazování a jednak je celkový průběh výsledné kontury často koherentnější. Další výraznou slabinu jsme z analýzy jednotlivých výstupů sítí neregistrovali. U výstupů, ve kterých se sítě nejvíce lišily, byl tento rozdíl nejčastěji způsoben právě těmito diskontinuitami, případně pak zachycením jiného než hlavního nástroje v pozadí. Největší rozdíly ve výsledcích jsou proto na datasetu ORCHSET, ve kterém je výskyt polyfonie nejčastější.

Metrika (Metoda)	Musorgski-Ravel-PicturesExhibition-ex6
RPA (HCNN)	0.125
RCA (HCNN)	0.725
RPA (Bittner)	0.397
RCA (Bittner)	0.826
RPA (Basaran)	0.040
RCA (Basaran)	0.914

Tabulka 6.4: Přesnost metod na testovacím souboru `Musorgski-Ravel-PicturesExhibition-ex6` z datasetu ORCHSET.

Basaranova metoda pro tuto kontinuitu tónů však obětovala přesné znějící výšky tónů. Jak jsme již prezentovali v kapitole Experimenty, výstup metod kvantizovaný na půltóny obsahuje množství chyb, jelikož často selhává v zachycení frekvenčních modulací. Na obrázku 6.6 vidíme další limitaci takového výstupu — pokud je obsah skladby laděný podle jiného referenčního tónu než jaký byl použit pro trénování sítě, znějící tóny vycházejí výškou „mezi“ výstupní složky. Z tabulky 6.5 a obrázku 6.5 je pak zřejmé, že kvůli této kvantizaci síť nedosahuje srovnatelných výsledků, přestože na jiných, žánrově shodných datech, které jsou laděny na správný referenční tón, podává kompetitivní výsledky. Limitace se proto týká zejména jazzových nahrávek pocházejících z období před rokem 1955, před zavedením referenčního tón A4=440Hz ve standardu ISO16.

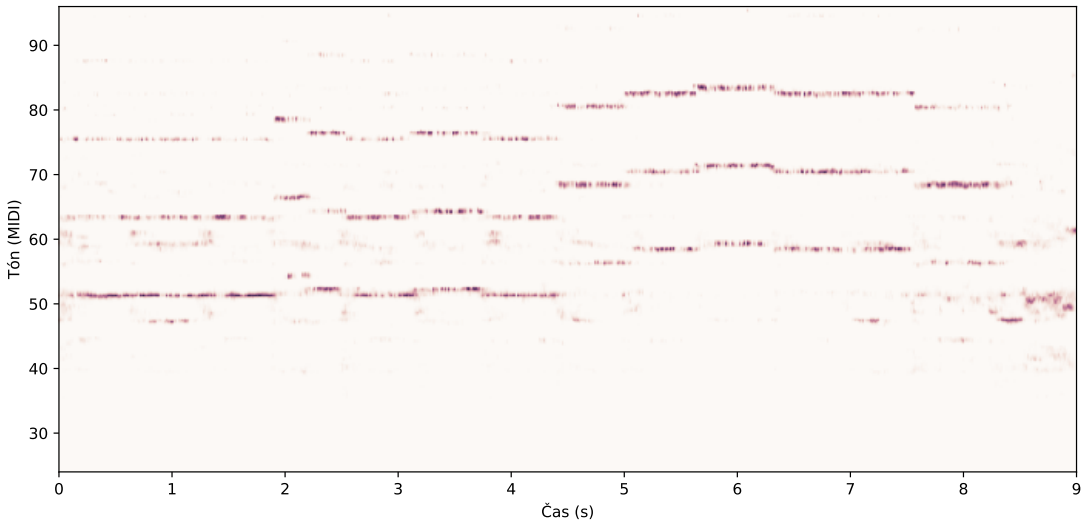
Dalším problémem Basaranovy metody je zhoršená schopnost extrakce na syn-



Obrázek 6.3: Výstup metod na testovacím souboru `Musorgski-Ravel-PicturesExhibition-ex6` z datasetu ORCHSET.

tetických datech. Příklady `midi2REF`, `midi3REF`, `train10REF` z datasetů ADC04 a MIREX05 jsou syntetizovány na základě MIDI pomocí základních zvukových fontů, nahrávky proto zní velmi uměle. Jak vidíme na obrázku 6.1, zatímco metody Bittnerové a HCNN si s touto syntetickou barvou hlasu dokáží poradit, výstup Basaranovy metody obsahuje šum a skoky k doprovázejícím nástrojům. Příčinou může být jiná vstupní reprezentace signálu, která je založena na práci Durrieu a David (2010) a spočívá na modelování hlavního hlasu pomocí zdroje a filtrů. Na obrázku 6.2 srovnáváme tuto reprezentaci pro vstupní signál s lidským zpěvem (nahoře, `train01`) a pro signál se syntetickou flétnou (dole, `train10`). Je zřejmé, že zatímco lidský zpěv tato reprezntace dokáže zachytit, syntetický hlas na reprezentaci téměř zachycen není.

Rozdílem mezi HCNN a metodou Bittnerové jsou zejména jiné priority, které přiřazují barvám hlasů, lze tudíž nalézt mnoho příkladů, kde HCNN zároveň přepisuje melodii a nesprávně místy přeskakuje nástroji v doprovodu, zatímco metoda Bittnerové na stejném příkladu tuto chybu nedělá. Podobně existují i opačné příklady. Zajímavým případem je soubor `MatthewEntwistle_FairerHopes` z ko-



Obrázek 6.4: Výstupní salience metody HCNN na testovacím souboru `Musorgski-Ravel-PicturesExhibition-ex6` z datasetu ORCHSET.

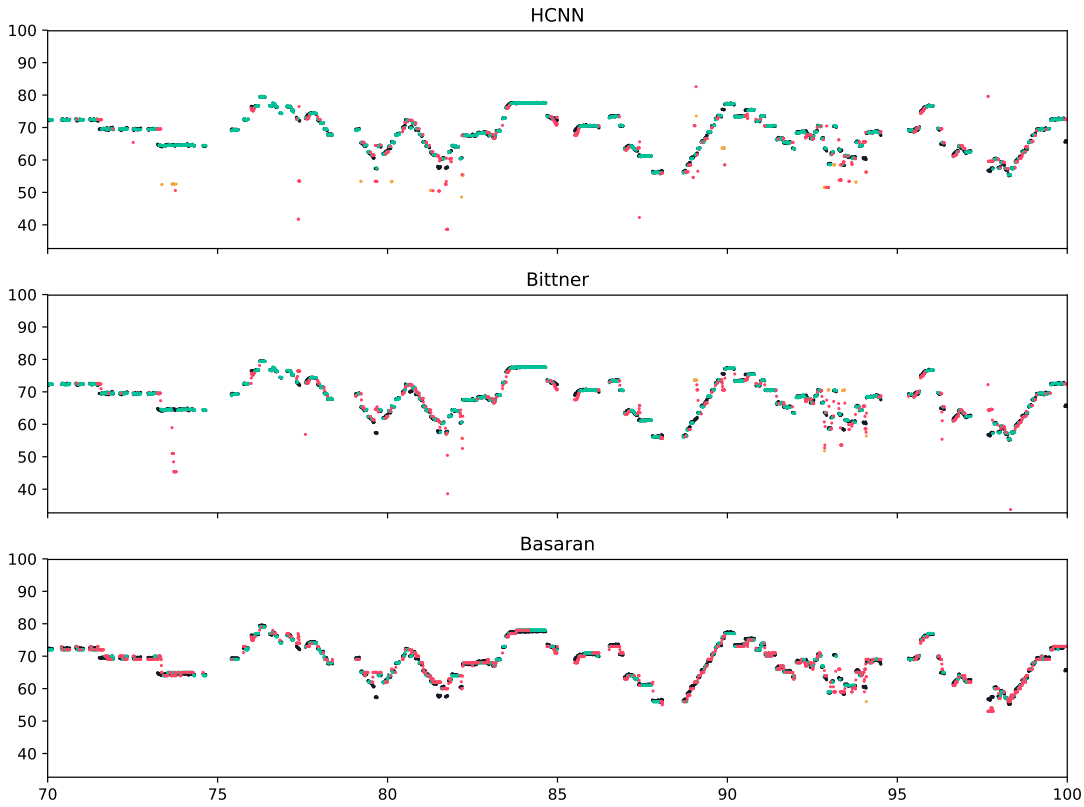
Metrika (Metoda)	CannonballAdderley_SoWhat
RPA (HCNN)	0.850
RCA (HCNN)	0.861
RPA (Bittner)	0.828
RCA (Bittner)	0.833
RPA (Basaran)	0.653
RCA (Basaran)	0.656

Tabulka 6.5: Přesnost metod na testovacím souboru `CannonballAdderley_SoWhat` z datasetu WJazzD.

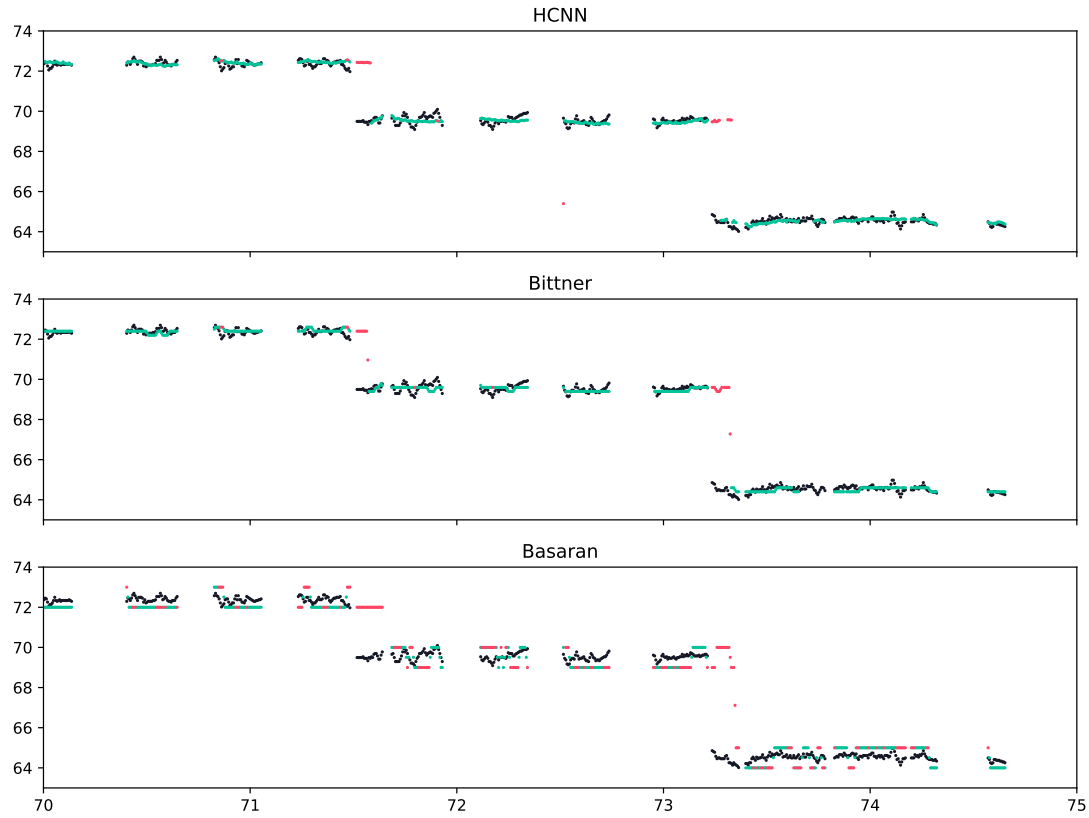
lekce MedleyDB, ve kterém melodii hraje harfa, která se však nevyskytuje v množině trénovacích dat. Zatímco metoda HCNN zvládá generalizovat i na dosud neslyšené barvy, metoda Bittnerové tóny ignoruje (viz obrázek 6.7).

Metrika (Metoda)	MatthewEntwistle_FairerHopes
RPA (HCNN)	0.451
RCA (HCNN)	0.626
RPA (Bittner)	0.118
RCA (Bittner)	0.423
RPA (Basaran)	0.544
RCA (Basaran)	0.661

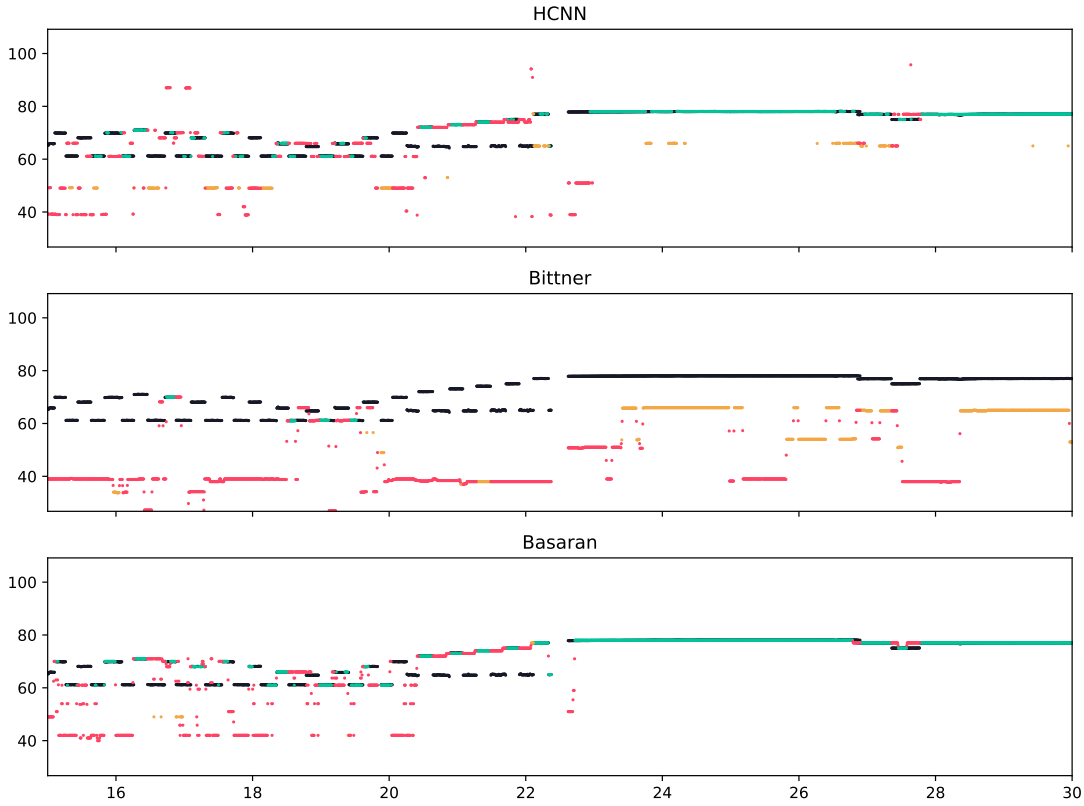
Tabulka 6.6: Přesnost metod na testovacím souboru `MatthewEntwistle_FairerHopes` z datasetu MedleyDB.



Obrázek 6.5: Výstup metod na testovacím souboru `CannonballAdderley_SoWhat` z datasetu WJazzD.



Obrázek 6.6: Detail přepisu metod na testovacím souboru `CannonballAdderley_SoWhat` z datasetu WJazzD.



Obrázek 6.7: Výstup metod na testovacím souboru `MatthewEntwistle_FairerHopes` z datasetu MedleyDB.

Závěr

Seznam použité literatury

Balke, S., Dittmar, C., Abesser, J. a Muller, M. (2017). Data-driven solo voice enhancement for jazz music retrieval. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 196–200. ISSN 15206149. doi: 10.1109/ICASSP.2017.7952145. URL https://www.audiolabs-erlangen.de/content/05-fau/assistant/balke/01-publications/2017{_}BalkeDAM{_}SoloVoiceEnhancement{_}ICASSP.pdf.

Bittner, R., Salamon, J., Tierney, M., Mauch, M., Cannam, C. a Bello, J. (2014). MedleyDB: A multitrack dataset for annotation - intensive mir research. *International Society for Music Information Retrieval Conference*.

Bittner, R. M. (2018). Data-Driven Fundamental Frequency Estimation.

Bittner, R. M., Mcfee, B., Salamon, J., Li, P. a Bello, J. P. (2017). Deep Salience Representations for F0 Estimation in Polyphonic Music. *Ismir*, pages 23–27. URL https://bmcfee.github.io/papers/ismir2017{_}salience.pdf.

Bittner, R. M., McFee, B. a Bello, J. P. (2018). Multitask Learning for Fundamental Frequency Estimation in Music. Technical report. URL http://arxiv.org/abs/1809.00381.

Bosch, J. J. a Gómez, E. (2014). Melody Extraction in Symphonic Classical Music: a Comparative Study of Mutual Agreement Between Humans and Algorithms. *Proceedings of the Conference on Interdisciplinary Musicology*. URL http://phenicx.upf.edu/system/files/publications/cim14{_}submission{_}114{_}ready.pdf.

Bosch, J. J. a Gómez, E. (2016). Melody Extraction Based on a Source-Filter Model Using Pitch Contour Selection. *13th Conference on Sound and Music Computing*, pages 67–74.

Bosch, J. J., Marxer, R. a Gómez, E. (2016). Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, **45** (2), 101–117. ISSN 17445027. doi: 10.1080/09298215.2016.1182191. URL https://repositori.upf.edu/bitstream/handle/10230/26985/Bosch{_}NewMusic{_}Eval.pdf?sequence=1{&}isAllowed=y.

Brown, J. C. (1990). Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, **89**(1), 425–434. ISSN 0001-4966. doi: 10.1121/1.400476. URL http://academics.wellesley.edu/Physics/brown/pubs/cq1stPaper.pdf.

Cancela, P. (2008). Tracking melody in polyphonic audio. *4th Music Information Retrieval Evaluation eXchange*. URL https://pdfs.

semanticscholar.org/226e/a7b870fd229149fae2e6c8b15d8a3d4f9bb8.pdf.

Cancela, P., López, E. a Rocamora, M. (2010). FAN CHIRP TRANSFORM FOR MUSIC REPRESENTATION. *Audio*, (1), 1–8. URL https://iie.fing.edu.uy/publicaciones/2010/CLR10/CLR10.pdf.

Chan, T. S., Yeh, T. C., Fan, Z. C., Chen, H. W., Su, L., Yang, Y. H. a Jang, R. (2015). Vocal activity informed singing voice separation with the iKala dataset. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, **2015-August**, 718–722. ISSN 15206149. doi: 10.1109/ICASSP.2015.7178063.

D Basaran, S Essid, G. P. (2018). Main melody extraction with source-filter nmf and crnn. *Ismir*, pages 82–89. URL http://ismir2018.ircam.fr/doc/pdfs/273{_}Paper.pdf.

Downie, J. S., Ehmann, A. F., Bay, M. a Jones, M. C. (2010). eXchange : Some Observations and Insights. *Advances in Music Information Retrieval*, pages 93–115. URL https://pdfs.semanticscholar.org/3836/15607f345a8cfbc5e884eaa5ca04f3c4b139.pdf.

Dressler, K. (2009). Audio melody extraction for mirex 2009. *Evaluation eXchange (MIREX)*, pages 1–3. URL http://www.idmt.fraunhofer.de/content/dam/idmt/de/Dokumente/Produktflyer/QbH/white{_}paper{_}fraunhofer{_}idmt{_}audio{_}melody{_}extraction{_}mirex{_}200
pdf.

Dressler, K. (2011). Pitch estimation by the pair-wise evaluation of spectral peaks. *42nd AES Conference*, pages 1–10. URL http://www.aes.org/e-lib/browse.cfm?elib=15960.

Dressler, K. (2016). Automatic Transcription of the Melody from Polyphonic Music. URL https://www.db-thueringen.de/servlets/MCRFileNodeServlet/dbt{_}derivate{_}00038847/ilm1-2017000136.pdf.

Durrieu, J.-l. a David, B. (2010). Source / Filter Model for Unsupervised Main Melody. **18**(3), 1–12. URL https://www.irit.fr/{~}Cedric.Fevotte/publications/journals/ieee{_}asl{_}voice{_}extrac.pdf.

Durrieu, J. L., David, B. a Richard, G. (2011). A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE Journal on Selected Topics in Signal Processing*, **5**(6), 1180–1191. ISSN 19324553. doi: 10.1109/JSTSP.2011.2158801. URL http://durrieu.ch/publis/durrieuDavidRichard{_}musicallyMotivatedRepresentation{_}JSTSP2011.pdf.

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Eck, D., Simonyan, K. a Norouzi, M. (2017). Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. URL http://arxiv.org/abs/1704.01279.

Gabor, B. D. a Member, A. (1945). THEORY OF COMMUNICATION * Part 1 . THE ANALYSIS OF INFORMATION. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, **93** (1946), 429–441.

Goto, M. a Hayamizu, S. (1999). A real-time music scene description system: Detecting melody and bass lines in audio signals. *IJCAI-99 Workshop on Computational Auditory Scene Analysis*, (August), 31–40. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.1085{&}rep=rep1{&}type=pdf.

Hawthorne, C., Elsen, E., Song, J., Roberts, A., Simon, I., Raffel, C., Engel, J., Oore, S. a Eck, D. (2017). Onsets and Frames: Dual-Objective Piano Transcription. ISSN 09259902. doi: 10.1007/s10844-013-0258-3. URL https://arxiv.org/pdf/1710.11153.pdfhttp://arxiv.org/abs/1710.11153.

Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., Elsen, E., Engel, J. a Eck, D. (2018). Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset. pages 1–12. URL http://arxiv.org/abs/1810.12247.

He, K., Zhang, X., Ren, S. a Sun, J. (2015). Deep Residual Learning for Image Recognition. URL http://arxiv.org/abs/1512.03385.

Hermes, D. J. (1988). Measurement of pitch by subharmonic summation. *The Journal of the Acoustical Society of America*, **83**(1), 257–264. ISSN 0001-4966. doi: 10.1121/1.396427. URL http://asa.scitation.org/doi/10.1121/1.396427.

Hsu, C.-l. a Jang, J.-s. R. (2010). Singing Pitch Extraction by Voice Vibrato/Tremolo Estimation and Instrument Partial Deletion. *11th Int. Soc. for Music Info. Retrieval Conf.*, (Ismir), 525–530. URL http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=9BE69D930B1BEADFCD07E4534BA4E7E0?doi=10.1.1.231.4599{&}rep=rep1{&}type=pdf.

Humphrey, E. J., Salamon, J., Nieto, O., Forsyth, J., Bittner, R. M. a Bello, J. P. (2014). JAMS: A JSON Annotated Music Specification for Reproducible MIR Research. *Proceedings of the 15th International Society for Music Information Retrieval Conference*, (September), 591–596.

Ikemiya, Y., Itoyama, K. a Yoshii, K. (2016). Singing Voice Separation and Vocal F0 Estimation Based on Mutual Combination of Robust Principal Component Analysis and Subharmonic Summation. *IEEE/ACM Transactions on Audio Speech and Language Processing*, **24**(11), 2084–2095. ISSN 23299290. doi: 10.1109/TASLP.2016.2577879.

Kim, J. W., Salamon, J., Li, P. a Bello, J. P. (2018). Crepe: A Convolutional Representation for Pitch Estimation. *ICASSP, IEEE International*

Conference on Acoustics, Speech and Signal Processing - Proceedings, **2018-April**, 161–165. ISSN 15206149. doi: 10.1109/ICASSP.2018.8461329. URL https://arxiv.org/pdf/1802.06182.pdf.

Kingma, D. P. a Ba, J. (2014). Adam: A Method for Stochastic Optimization. pages 1–15. URL http://arxiv.org/abs/1412.6980.

Kum, S., Oh, C. a Nam, J. (2016). Melody Extraction on Vocal Segments Using Multi-Column Deep Neural Networks. *Proceedings of the 17th International Society for Music Information Retrieval Conference, (ISMIR)*, (August), 819–825. URL https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/119{_}Paper.pdf.

Martak, L. S., Sajgalik, M. a Benesova, W. (2018). Polyphonic Note Transcription of Time-Domain Audio Signal with Deep WaveNet Architecture. *International Conference on Systems, Signals, and Image Processing*, **2018-June**, 2–6. ISSN 21578702. doi: 10.1109/IWSSIP.2018.8439708. URL https://vgg.fiit.stuba.sk/wp-uploads/2018/09/iwssip2018{_}wavenet.pdf.

Mauch, M. a Dixon, S. (2014). PYIN: A fundamental frequency estimator using probabilistic threshold distributions. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, **1**(1), 659–663. ISSN 15206149. doi: 10.1109/ICASSP.2014.6853678. URL https://www.eecs.qmul.ac.uk/{~}simond/pub/2014/MauchDixon-PYIN-ICASSP2014.pdf.

Paiva, R. P., Mendes, T. a Cardoso, A. (2004). An Auditory Model Based Approach for Melody Detection in Polyphonic Musical Recordings. (May 2014), 21–40. doi: 10.1007/978-3-540-31807-1_2.

Pfleiderer, M., Frieler, K., Abesser, J., Zaddach, W.-G. a Burkhart, B. *Inside the Jazzomat*. ISBN 9783959831246. URL http://schott-campus.com/wp-content/uploads/2017/11/inside{_}the{_}jazzomat{_}final{_}rev{_}oa4.pdf.

Poliner, G. E. a Ellis, D. P. W. (2005). A Classification Approach to Melody Transcription. *Proc. 6th Int. Conf. on Music Inform. Retrieval*, (1), 161–166.

Poliner, G. E., Ellis, D. P. W., Ehmann, A. F., Gómez, E., Streich, S. a Ong, B. (2007). Melody transcription from music audio: Approaches and evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, **15** (4), 1247–1256. ISSN 15587916. doi: 10.1109/TASL.2006.889797. URL https://academiccommons.columbia.edu/doi/10.7916/D8NC69RK/download.

Raffel, C., Mcfee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D. a Ellis, D. P. W. (2014). mir_eval: A Transparent Implementation of Common MIR Metrics. *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, pages 367–372. URL https://bmcfee.github.io/papers/ismir2014{_}mireval.pdf.

Rao, V. a Rao, P. (2010). Vocal melody extraction in the presence of pitched accompaniment in polyphonic music. *IEEE Transactions on Audio, Speech and Language Processing*, **18**(8), 2145–2154. ISSN 15587916. doi: 10.1109/TASL.2010.2042124. URL https://www.ee.iitb.ac.in/course/{~}daplab/publications/PrePrintForWebsite.pdf.

Rigaud, F. a Radenen, M. (2016). Singing Voice Melody Transcription Using Deep Neural Networks. *Ismir*, pages 737–743. URL https://s18798.pcdn.co/ismir2016/wp-content/uploads/sites/2294/2016/07/163{_}Paper.pdf.

Ryynänen, M. P. a Klapuri, A. P. (2008). Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *Computer Music Journal*, **32**(3), 72–86. ISSN 0148-9267. doi: 10.1162/comj.2008.32.3.72.

Salamon, J. a Gomez, E. (2012). Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech and Language Processing*, **20**(6), 1759–1770. ISSN 15587916. doi: 10.1109/TASL.2012.2188515.

Salamon, J., Gómez, E., Ellis, D. P. a Richard, G. (2014). Melody extraction from polyphonic music signals: Approaches, applications, and challenges. *IEEE Signal Processing Magazine*, **31**(2), 118–134. ISSN 10535888. doi: 10.1109/MSP.2013.2271648. URL http://www.justinsalamon.com/uploads/4/3/9/4/4394963/salamon{_}gomez{_}ellis{_}richard{_}melodyextractionreview{_}ieeespm{_}2013.pdf.

Salamon, J., Bittner, R. M., Bonada, J., Bosch, J. J., Gomez, E. a juan pablo Bello (2017). An Analysis/Synthesis Framework for Automatic F0 Annotation of Multitrack Datasets. *Proceedings of the International Society for Music Information Retrieval {(ISMIR)} Conference*, pages 71–78.

Stoller, D., Ewert, S. a Dixon, S. (2018). Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation. pages 334–340. ISSN 13514180. doi: arXiv:1806.03185v1. URL http://arxiv.org/abs/1806.03185.

Sturm, B. L. (2013). Classification accuracy is not enough. *Journal of Intelligent Information Systems*, **41**(3), 371–406. ISSN 0925-9902. doi: 10.1007/s10844-013-0250-y. URL https://link.springer.com/content/pdf/10.1007{%}2Fs10844-013-0250-y.pdf.

Su, L. (2018). Vocal melody extraction using patch-based CNN. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, **2018-April**, 371–375. ISSN 15206149. doi: 10.1109/ICASSP.2018.8462420. URL https://arxiv.org/pdf/1804.09202.pdf.

Tachibana, H., Ono, T., Ono, N. a Sagayama, S. (2010). Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, (May 2014), 425–428. ISSN 15206149. doi: 10.1109/ICASSP.2010.5495764.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. a Kavukcuoglu, K. (2016a). WaveNet: A Generative Model for Raw Audio. pages 1–15. URL http://arxiv.org/abs/1609.03499.

van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A. a Kavukcuoglu, K. (2016b). Conditional Image Generation with PixelCNN Decoders. URL http://arxiv.org/abs/1606.05328.

Yang, L.-C., Chou, S.-Y. a Yang, Y.-H. (2017). MidiNet: A Convolutional Generative Adversarial Network for Symbolic-domain Music Generation. URL http://arxiv.org/abs/1703.10847.

Yeh, T. C., Wu, M. J., Jang, J. S. R., Chang, W. L. a Liao, I. B. (2012). A hybrid approach to singing pitch extraction based on trend estimation and hidden Markov models. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, (August), 457–460. ISSN 15206149. doi: 10.1109/ICASSP.2012.6287915. URL https://www.researchgate.net/profile/Jyh-Shing{_}Jang/publication/261113746{_}A{_}hybrid{_}approach{_}to{_}singing{_}pitch{_}extraction{_}base
links/55d5446f08ae6788fa352c8e/A-hybrid-approach-to-singing-pitch-extraction
pdf.

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

Přílohy