VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÉ ROZPOZNÁVÁNÍ HUDEBNÍHO ZÁPISU POMOCÍ NEURONOVÝCH SÍTÍ

NEURAL NETWORKS FOR OPTICAL MUSIC RECOGNITION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VOJTĚCH VLACH**
AUTHOR

VEDOUCÍ PRÁCE **Ing. MICHAL HRADIŠ, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

146120

Ústav: Ústav počítačové grafiky a multimédií (UPGM)
Student: **Vlach Vojtěch**
Program: Informační technologie
Specializace: Informační technologie

Název: **Automatické rozpoznávání hudebního zápisu pomocí neuronových sítí**
Kategorie: Počítačové vidění
Akademický rok: 2022/23

Zadání:

1. Prostudujte základy konvolučních neuronových sítí, sítí založených na attention a autoregresivních modelů.
2. Vytvořte si přehled o současných metodách automatického rozpoznání hudebního zápisu z obrazu.
3. Navrhněte metodu schopnou automaticky rozpoznávat hudební zápis z obrazu nebo upravte vhodnou existující metodu.
4. Obstarejte si databázi vhodnou pro experimenty. Můžete rozšířit existující databázi.
5. Implementujte navrženou metodu a proveďte experimenty nad datovou sadou.
6. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
7. Vytvořte stručné video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Edirisooriya et al.: An Empirical Evaluation of End-to-end Polyphonic Optical Music Recognition. in Proc. of the 22nd Int. Society for Music Information Retrieval Conf., 2021.
- Mayer Jiří: Semi-supervised learning in Optical Music Recognition. thesis, Charles University, 2022.
- Ríos-Vila et al.: On the Use of Transformers for End-to-End Optical Music Recognition. Iberian Conference on Pattern Recognition and Image Analysis, 2022.
- Shatri, Fazekas: Optical Music Recognition: State of the Art and Major Challenges. International Conference on Technologies for Music Notation and Representation, 2020.

Při obhajobě semestrální části projektu je požadováno:

- Body zadání 1 - 4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Hradiš Michal, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.
Datum zadání: 1.11.2022
Termín pro odevzdání: 10.5.2023
Datum schválení: 3.5.2023

Abstrakt

Tato práce řeší problém rozpoznání hudebních zápisů z obrázku do textové podoby pomocí umělé inteligence a neuronových sítí. Zaměřuje se konkrétně na tištěnou polyfonní hudbu (více not a hlasů naráz). Cílem práce je vytvořit model schopný rozpoznat složité zápisy a jeho úspěšnost porovnat s předchozí literaturou a známými modely.

Zvolený problém jsem vyřešil díky využití architektury Vision-transformer, kde jsem testoval několik variant sítě za účelem nalezení té nejvýkonější, a vytvoření nového datasetu s polyfonní hudbou. Práce představuje proces vytvoření datasetu pomocí syntetizování obrázků z formátu MusicXML programem MuseScore.

Nejúspěšnější varianta architektury Vision-Transformer dosahuje minimální chybovosti pouze 7,86 %, což je velmi slibné pro další vývoj a využití. Hlavním zjištěním je, že architektura má potenciál dominovat na tomto poli stejně jako na jiných polích výzkumu a pro konkrétní úlohu rozpoznání polyfonních hudebních zápisů existuje funkční řešení, což bylo doteď předmětem debaty.

Abstract

This thesis consideres the problem of optical music recognition from images to text using Artificial inteligence and neural networks. I have choosed particularly the field of printed polyphonic music (more notes and voices at the same time). The goal of this thesis is to create a model capable of recognising complex notations and its accuracy compare with previous literature and other known models.

I solved the chosen problem by utilizing the Vision Transformer architecture, where I tested several network variants to find the most powerful one. And by creating a new dataset with polyphonic music. The work presents the process of creating the dataset by synthesizing images from MusicXML format using the MuseScore program.

The most successful variant of the Vision Transformer architecture achieves an error rate of only 7.86 %, which is very promising for further development and utilization. The main finding is that the architecture has the potential to dominate in this field, just as it does in other areas of research, and there is a functional solution for the specific task of polyphonic music notation recognition, which has been only up for a debate until now.

Klíčová slova

Počítačové vidění, neuronové sítě, transformer, rozpoznání hudebních zápisů, OMR, polyfonní hudba, příprava trénovacích dat

Keywords

Computer vision, neural networks, transformer, optical music recognition, OMR, polyphonic music, preparation of training data

Citace

VLACH, Vojtěch. *Automatické rozpoznávání hudebního zápisu pomocí neuronových sítí.* Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Hradiš, Ph.D.

Automatické rozpoznávání hudebního zápisu pomocí neuronových sítí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Michala Hradiše, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Vojtěch Vlach
10. května 2023

Poděkování

Děkuji mému vedoucímu za věcné připomínky a vedení postupu správným směrem. Výpočetní zdroje byly poskytnuty projektem e-INFRA CZ (ID: 90140), podporovaným Ministerstvem školství, mládeže a tělovýchovy České republiky, kterému tímto děkuji. V neposlední řadě děkuji mé snoubence, která mě neúnavně podporovala a nikdy to se mnou nevzdala.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Automatické rozpoznání hudebního zápisu (anglicky Optical Music Recognition – dále OMR) je obor, který má za cíl zjistit co nejvíce informací z prosté fotky hudebního zápisu. Může to být zápis poslední rychlovky od Pokáče nebo ručně psaná partitura Smetanovy Vltavy. Úkol OMR je automaticky (bez pomoci lidí) zpracovat fotku zápisu a vytvořit z ní nahrávku, přepsat noty do editačního softwaru nebo zjistit, do jakého stylu se daná skladba zařazuje. OMR je rozmanitá oblast, kde hlavní část spočívá v rozpoznání co nejvíce symbolů pro účely přepisu do libovolného formátu pro další zpracování člověkem nebo počítačem.

OMR má spoustu možných využití, například zpracování a zpřístupnění historických skladeb v archivech a knihovnách, kde by jinak byly k dispozici pouze omezeně. Dále lze OMR použít ke kategorizaci skladeb a zjištění informací neviditelných na první pohled, např. styl, odhad doby vytvoření, nebo dokonce určení autora. V minulosti se již mnoho lidí tyto úlohy pokoušelo vyřešit pomocí různých kroků a v každém dalším článku se přidávají nové a nové funkce a možnosti vývoje. Stále však zbývá spousta úloh otevřených a zatím neexistuje vyhovující řešení.

Jedna z hlavních úloh je přepis obrázku do textové podoby, která může být dále zpracování, a rozpoznání jednotlivých not, taktů, akordů atd. pomocí umělé inteligence. Na tuto oblast se zaměřuje tato práce. Přesto, že už existují řešení, jsou zatím omezená na jednodušší monofonní zápisy (pouze jedna nota naráz). Tato práce má za cíl vyzkoušet přístup k rozpoznání složitějších a komplexnějších hudebních zápisů polyfonní hudby (více not a více hlasů). Výsledkem práce nebude hotová aplikace, ale spíše průzkum, které metody a principy stojí za to použít v budoucnosti. Pro téma práce jsem se rozhodl, protože hudba je mojí vášní a trávím s ní spoustu času a aplikace pro vytváření, úpravu a rozpoznání not by mi velmi pomohla. Na konci této práce chci položit základ pro další postup v této oblasti, abych jednoho dne mohl takovou aplikaci sám používat.

Práce se skládá z několika kapitol. Kapitola Rozpoznání hudebního zápisu definuje oblast, ve které se práce pohybuje, uvádí důležitou literaturu, ze které bude práce vycházet, a ukazuje prostředky k dosažení ozkoušených cílů.

Kapitola Neuronové sítě představuje vybrané pojmy umělé inteligence, které se budou dále využívat. Vysvětluje princip fungování konvolučních neuronových sítí, rekurentních neuronových sítí a představuje tzv. architekturu transformer a její variantu Vision-Transformer.

V kapitole Návrhy řešení navrhuji dva způsoby pro řešení úlohy. Jeden je osvědčený z předchozích studií, druhý vytvářím zcela nový.

V kapitole Implementace popisuji, co jsem vytvořil a využil pro dosažení cílů práce. Představuji navržené architektury a proces vytvoření unikátního datasetu pro trénování.

Nakonec v kapitole Experimenty popisuji, čeho jsem dosáhl, a jaké má moje zjištění důsledky.

Kapitola 2

Rozpoznání hudebního zápisu

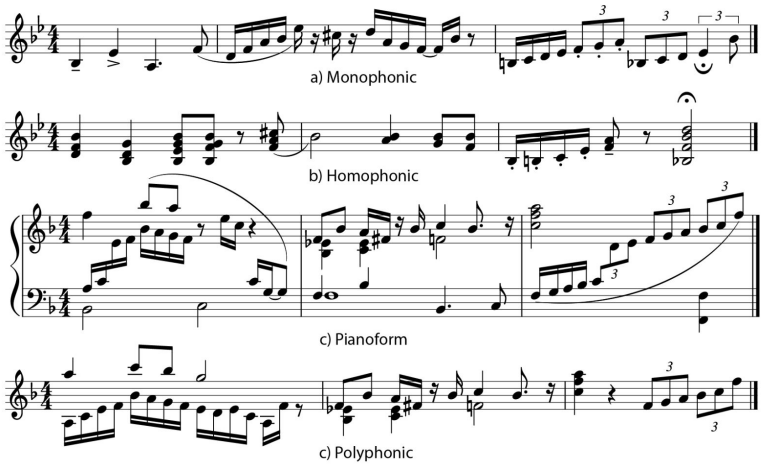
Optical Music Recognition (dále OMR) je oblast výzkumu, která se zabývá rozpoznáváním notových zápisů v dokumentech. Klasický přístup je tzv. OMR pipeline (viz kapitola 2.1), kde úlohu provádí více samostatných systémů. V posledních letech se začíná používat tzv. End-to-end přístup (viz kapitola 2.2, který k dané úloze přistupuje jako k celku. Využívá komplexnější architektury a potřebuj velké množství dat. Pro End-to-end přístup existuje např. dataset PrIMuS [5] s monofonní hudbou (naráz maximálně jedna nota). Stabilní dataset s polyfonní hudbou (více not naráz) zatím nebyl zveřejněn, ale například článek *Empirical evaluation...* [9] představuje způsob generování nového datasetu podle vlastních představ a potřeb.

Hlavní cíl oblasti OMR je rozpoznání a porozumění notovému zápisu. Dále je to snaha o uchování historických výtvorů, aby neupadly v zapomnění. OMR tak může být brána jako analogie k Optickému rozpoznávání znaků (Optical Character Recognition – dále OCR), která umožnila automatické zpracování psaných textů a vyzývá k automatizaci. OMR se zabývá automatizací této úlohy „čtení“ v kontextu hudby. Nicméně, zatímco hudebníci dokážou číst a interpretovat velmi složité notové záznamy, stále neexistuje počítačový systém, který by to dokázal.

Následující definice a její vysvětlení byly převzaty z Calvo-Zaragoza, Hajič a Pacha: *Understanding optical music recognition* [4].

Definice OMR *Rozpoznání hudebních zápisů je oblast výzkumu, která zkoumá, jak automaticky číst notový zápis v dokumentech.*

OMR není pouze úloha nebo proces, je to oblast výzkumu, protože se pod tento pojem řadí několik úloh a procesů, lišících se jak formáty vstupu a výstupu, tak primárním cílem. Pojem „automaticky“ odlišuje oblast od muzikologie. OMR nezkoumá systém notací jako takový, staví na znalostech hudební teorie s cílem naučit počítač její pravidla a fungování. Poslední část definice „číst notový zápis v dokumentech“ se snaží inkluzivně pojmout všechny varianty očekávaných vstupně-výstupních párů různých variant systémů. Vstupy mohou být tisknuté/skenované, ručně psané nebo symbolické. Výstupy mohou být kódované pro znovupřehrátí (zvukový soubor MIDI), pro znovu vytvoření grafické podoby, extrakce metadat (styl hudby, doba vytvoření) a další.



Obrázek 2.1: Příklady čtyř nejpoužívanějších kategorií taxonomie hudebních zápisů podle obtížnosti, převzato z Shatri et al. *Optical music recognition: State of the art and major challenges* [21]

Kategorizace hudebních zápisů podle komplexnosti se v různých zdrojích liší, ale lze mezi nimi najít společný průnik. Například Shatri E. et al. [21] uvádí následující běžně používané rozdělení (viz obr. 2.1):

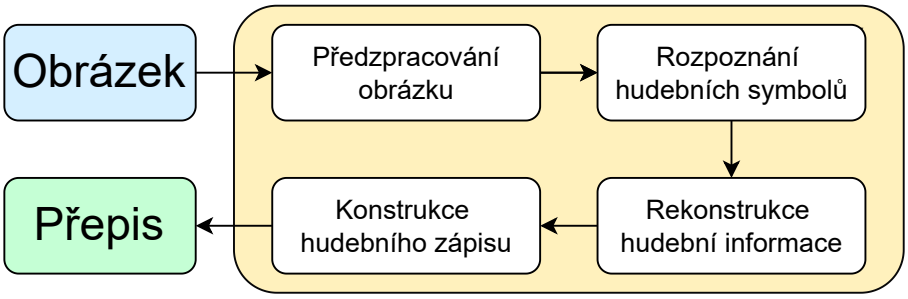
1. Monofonní hudba – jedna notová osnova, ve stejný čas pouze jedna nota
2. Homofonní hudba – jedna notová osnova, více not naráz pouze ve formě akordů (viz 2.4)
3. Pianoformní hudba – dvě notové osnovy typicky s více hlasy a s významnou interakcí mezi osnovami
4. Polyfonní hudba – jedna notová osnova, více not naráz (akordy nebo více hlasů)

Byrd a Simonsen [3] kromě výše zmíněných kategorií uvádí striktně monofonní zápisy na více osnovách. Jedná se tedy o hudbu pro více nástrojů naráz (sbor, komorní uskupení, orchestr).

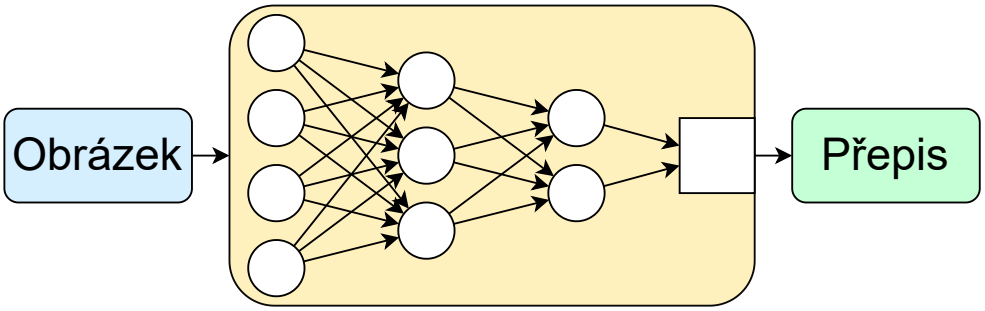
2.1 Klasický přístup k OMR

Klasický přístup k řešení OMR je tzv. OMR pipeline (viz Obr. 2.3). Jedná se o standardizovanou sadu kroků, které provádí jednotlivé podúlohy. *Rebelo et al.* [19] ji popisuje v těchto krocích:

1. Předzpracování obrázku
2. Rozpoznání hudebních symbolů
3. Rekonstrukce hudební informace pro vytvoření logického popisu hudební notace
4. Konstrukce modelu hudebního zápisu k symbolické reprezentaci



Obrázek 2.2: *Rebelo et al.* [19] definuje původní přístup k OMR jako 4 fáze tvořící tzv. OMR pipeline.



Obrázek 2.3: Vizualizace End-to-end přístupu k OMR.

Rebelo et al. [19] dále vysvětluje jednotlivé kroky. V první fázi se používá několik technik, jak zefektivnit celý systém. Jsou to např. odstranění šumu a zkosení nebo binarizace obrázku na bílé a černé pixely. Další fáze – Rozpoznání hudebních symbolů – je většinou rozdělena na tři části: 1) detekce a odstranění taktových čar, 2) segmentace symbolových primitiv, 3) rozpoznání samotných symbolů. Ve fázi rekonstrukce probíhá se pracuje se sémantickým významem symbolů a řeší se nejasnosti. Detekované symboly jsou interpretovány s hudebním významem. V poslední fázi se vytváří hudební popis celé informace. Výstupem je soubor využitelný jinými programy, jako MIDI nebo MusicXML.

Vytvoření OMR pipeline umožňuje samostatné řešení podproblémů, což otevírá dveře postupnému vývoji na poli výzkumu. Jedná se zároveň o původní řešení a bylo zde už od počátku OMR. Přesto, že jednotlivé kroky postupují ke stále lepším řešením, když přijde na jejich spojení, není to jednoduché a zatím neexistuje jeden celistvý systém, který by to uměl. [4]

Jednou z výhod tohoto přístupu je, že potřebuje relativně málo dat na trénování a testování, stačí stovky až jednotky tisíců označených dat. Nevýhodou je ovšem to, že na každou podúlohu musí být specializovaná data na trénování a metrika úspěšnosti na vyhodnocení.

2.2 OMR end-to-end

Popis a zhodnocení nového přístupu k OMR bylo převzato z Calvo-Zaragoza, Hajič a Pacha: *Understanding optical music recognition* [4].

Nedávný pokrok na poli umělé inteligence také celkově diverzifikoval způsob, jakým se k OMR přistupuje. Vznikly alternativní přístupy s vlastním probíhajícím výzkumem, které se snaží čelit celému procesu v jednom kroku. Toto holistické paradigma, označované také jako end-to-end systémy, dominuje současnému „state-of-the art" v jiných úlohách, jako je rozpoznávání textu, řeči nebo matematických vzorců [6], [7], [25]. Nicméně vzhledem ke složitosti odvozování hudební sémantiky z obrazu je (zatím) obtížné formulovat ji jako naučitelný optimalizační problém. Ačkoli end-to-end systémy pro OMR existují, jsou stále omezené přinejlepším na podmnožinu hudebních notací.

Od doby vydání tohoto článku [4] (rok 2020) bylo zveřejněno několik výzkumů na toto téma, které většinou potvrzují výše uvedenou hypotézu, že OMR end-to-end systémy před sebou mají ještě dlouhou cestu [20]. Přesto se však zveřejňují další, protože lze mezi články sledovat stále další a další pokrok [9].

Pro implementaci systémů se většinou využívají konvoluční neuronové sítě spojené s rekurentními sítěmi a chybovou funkcí CTC [5], nebo modely typu enkodér-dekodér (sequence-to-sequence, viz 3.2) např. transformery [20]. Zatím se většinou pohybují v doméně monofonní hudby [5], ale objevují se i výzkumy s hudbou polyfonní [9].

Nevýhodou využití složitějších achitektur (Neuronové sítě a transformery) je potřeba velkého množství trénovacích dat, což je jedna z hlavních příčin současného pomalého pokroku. Objevují se však nové způsoby, jak tento problém vyřešit pomocí generování vlastních tištěných dat (notových osnov) [5], [9], nebo dokonce manipulace tištěných dat tak, aby napodobovaly ručně psané podoby notové osnovy [17].

Nejstarší dílo na téma end-to-end OMR, které jsem našel, je VAN DER WEL et al. *Optical music recognition with convolutional sequence-to-sequence models* [24]. V tomto článku autor představuje unikátní myšlenku syntetizování vlastního datasetu pomocí volně dostupných souborů a nástrojů MuseScore (viz 5.2) a trénování pomocí sequence-to-sequence modelu (viz 3.2). Modely trénuje a testuje na vlastním monofonním datasetu o neznámé velikosti a bližších parametrech. Dosahuje slibných hodnot chybovosti přepisů méně než 1 %. Chybovost hodnotí na základě poměru správně detekovaných hudebních symbolů k jejich celkovému počtu a používá tzv. Rhythm Symbol Error Rate a Pitch Symbol Error rate (viz kapitola 6.1).

V článku *Calvo-Zaragoza, Rizo: End-to-End Neural Optical Music Recognition of Monophonic Scores* [5] autoři představují nový stabilní dataset použitelný právě k End-to-end OMR s výhradně monofonní hudbou jménem PrIMuS [5][1] (Printed Images of Music Staves) a experimentují s ním pomocí kombinace konvoluční sítě (s klasickým VGG jádrem 3×3) a obousměrně propojenou rekurentní sítí (BLSTM – Bidirectional Long Short-Term Memory, viz **??**). Zároveň stanovují metriky hodnocení trénovaných modelů a dosahují slibných výsledků, které ale stále mají prostor pro zlepšení. Článek působí jako velmi informačně nabitý úvod do OMR end-to-end problematiky.

Dále stojí za zmínku článek EDIRISOORIYA et al. *An empyrical evaluation of end-to-end polyphonic optical music recognition* (dále jen *Empirical evaluation...* [9]), publikovaný na konferenci ISMIR 2021[2]. Článek opět syntetizuje vlastní datovou sadu podobným způsobem jako předchozí, tentokrát ale poprvé pro polyfonní hudbu. Dále experimentuje se sítěmi typu seq2seq (přesněji tzv. RNNDecoder s chybovou funkcí CTC) a podle předem daných hodnotících metrik dosahuje slibných výsledků. Opět nechává prostor pro zlepšení.

[1] Dataset PrIMuS [5] ke stažení zde: grfia.dlsi.ua.es/primus/

[2] ISMIR - International Society for Music Information Retrieval, ismir.net

Článek se stal inspirací pro tuto práci, která na něj v některých aspektech navazuje. Přebírá klíčové myšlenky a zdokonaluje jak vytvoření vlastní datové sady, tak architekturu modelů řešících úlohu.

Nakonec stojí za zmínku článek Ríos-Vila et al. *On the use of transformers for end-to-end optical music recognition* [20], kde autor opět experimentuje s monofonními datasety a hodnotí 4 modely převzaté z jiných oborů oproti svojí baseline s architekturou v podobě CRNN sítě s chybovou funkcí CTC (viz kapitola 3. Zkoumané modely jsou kombinace transformerového enkoderu a tzv. Vision transfomeru [8] s chybovou funkcí CTC nebo transformerovým dekodérem. Z výsledku experimentů zjišťuje, že pro samostatnou úlohu monofonního OMR nové modely nepřinesly žádné zlepšení. Jaká je situace u polyfonních zápisů, se snaží zjistit tato práce v dalších kapitolách.

2.3 Datové sady

OMR je poměrně specializovaný obor, kterému se věnuje malá množina výzkumníků, což je důvod proč neexistuje mnoho datových sad. Většina sad plní pouze jeden specifický účel.

Celistvý seznam všech datových sad lze nalézt na stránce Optical Music Recognition Datasets [18]

Kromě jedno-účelových datasetů existuje i několik sad využitelných pro end-to-end OMR. Jsou to například DeepScores, CVC-MUSICMA, MUSCIMA++.

Dataset PrIMus. PrIMuS[3] (Printed Images of Music Staves) [5] obsahuje přes 80 000 jednořádkových obrázků not s korespondujícími přepisy ve 2 standardizovaných formátech (MEI: open-source XML standard[4] a MIDI[5]) a dále dvou nových formátech, se kterými může přímo pracovat neuronová síť, tzv. Sémantické a Agnostické kódování. Agnostické kódování zachycuje symboly bez významu, pouze rozpoznání jednotlivých elementů vzhledem k nejbližšímu okolí (např. nota na druhé lince, křížek, půlová pomlka). Sémantické kódování uvádí symboly s jejich významem (např. půlová nota C1 v houslovém klíči). Detailní definice obou kódování je k nalezení v článku, kde byl dataset představen [5]. Dataset PrIMus obsahuje pouze monofonní tištěnou hudbu.

Ta pravá výzva přichází s hudbou polyfonní. Nedostatek datasetů s polyfonní hudbou vedl výzkumníky k využití open-source nástrojů na pomoc při vytváření vlastních datasetů.

K tomuto účelu se hodí např. jeden z nejpoužívanějších nástrojů na editaci hudebního zápisu MuseScore[6], který má možnosti přidat do aplikace vlastní moduly a tím doplnit funkce, které v základní aplikaci nejsou dostupné. Vice o využití MuseScore je k nalezení v kapitole 5.2.

Navržený postup v článku Edirisooriya et al. *An Empirical Evaluation of End-to-End Polyphonic Optical Music Recognition* umožňuje vytvoření nového datasetu pomocí nastavení fixní výšky stránky, ze které se exportují obrázky. Obrázky se tak automaticky ořízou na přibližnou oblast kolem první notové osnovy stránky a malé útržky okolních osnov/partů.

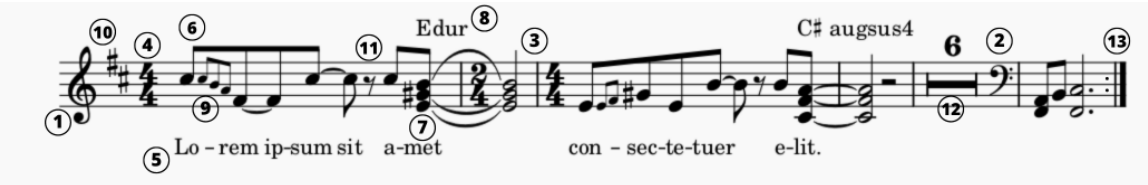
[3] Ke stažení například zde: grfia.dlsi.ua.es/primus/
[4] https://music-encoding.org/
[5] Musical Instrument Digital Interface, midi.org
[6] musescore.org

Tento proces má ovšem poměrně velkou ztrátovost a nevyužívá data optimálně. Hlavní příčinou je použití ořezu stránky pouze na první systém. Celý zbytek stránky tak zůstává nevyužit.

Nový způsob navržení datasetu (viz kapitola 5.2) má za cíl lépe využít data ze souborů MuseScore za použití většího porozumnění samotného kódování musicxml.

2.4 Vybrané části z hudební teorie

Tato kapitola uvádí vybrané části hudební teorie potřebné k pochopení pojmů zmiňovaných v souvislosti jejich rozpoznávání. Inspirací pro tuto sekci byly knihy *The everything reading music book...* [16] a *Hudební nauka pro malé i větší muzikanty* [15]



Obrázek 2.4: Příklad hudební osnovy s označenými symboly.
1. houslový klíč, 2. basový klíč, 3.taková čára, 4. taktové označení, 5. text písně, 6. nota, 7. akord, 8. akordová značka, 9. melodické ozdoby, 10. předznamenání, 11. pomlka, 12. vícetaktová pomlka, 13. repetice

Notová osnova. Prvním pojmem je notová osnova. Ta se skládá z pěti linek a čtyř mezer, které tvoří prostor pro noty. Jak nota vizuálně stoupá po stupnici, stoupá zároveň její výška. Na začátku notové osnovy se píší notové klíče. Tyto znaky definují, jak se jednotlivé noty v notové osnově nazývají. Dva nejznámější jsou houslový a basový klíč. Houslový klíč (1) je základním klíčem a také tím nejpoužívanějším. Využívá se zejména pro vysoké tóny. Také se mu říká G klíč. Basový klíč (2) se používá pro nízké tóny. Díky jeho poloze je možné definovat mnohem nižší tóny než v houslovém klíči. Také se mu říká F klíč.

Takty. Notová osnova je dále dělena taktovými čarami (3) na takty. Každý takt mívá stejný počet dob, který určuje taktové označení (4). Píše se hned za notovým klíčem. Vrchní číslo udává počet dob v taktu, spodní číslo pak určuje počítací jednotku, tedy to, jaká nota se v daném taktu počítá na 1 dobu. Na obrázku je označen takt čtyř čtvrťový (4), v uvedeném příkladu lze ale najít i takt dvou čtvrťový. Součástí skladby může být také text (5). Jde o slova, která se zpívají do rytmu písně.

Nota. Základní hudební jednotkou je nota (6), která je vizuálním zápisem tónu. Může mít různou délku – tu symbolizuje změna tvaru noty. Čím je nota vizuální složitější, tím je kratší. Pokud jsou noty poskládané vertikálně přímo nad sebou, znamená to, že mají být zahrané ve stejný čas. Tři a více takových not nad sebou se nazývá akord (7). V některých skladbách jsou akordy značeny také akordovou značkou (8), která akord popisuje slovně. Ve skladbách se také mohou objevit melodické ozdoby (9). Značí se malými notami a jejich délka je pouze přibližná.

Posuvky. Před notu můžeme také přidat posuvku. Tím měníme její výšku o půl tón, a to buď směrem nahoru (křížek #) , nebo dolů (béčko b) . Křížky a béčka také tvoří předznamenání (10). To se nachází na začátku notové osnovy hned za notovým klíčem. Předznamenání naznačuje, že jedna nebo více not budou v celé skladbě posunuté směrem nahoru nebo dolů. Díky předznamenání se už nemusí psát posuvka před notu pokaždé, když se ve skladbě objeví. Podle předznamenání se také určuje tónina.

Pomlky. Prázdná doba v taktu se označuje pomlkou (11). Podobně jako noty mají pomlky různé tvary, které určují jejich délku. Pomlka může trvat i celý takt. Také pomlce říkáme celotaktová; pokud trvá více taktů v kuse, jedná se o pomlku vícetaktovou (12). Vícetaktové pomlky se využívají hlavně v orchestrech, kdy je běžné, že se jeden nástroj ve skladbě na chvíli odmlčí a znovu se připojí až po několika taktech.

Repetice. Repetice (13) je značka označující úsek, který se má ve skladbě opakovat. Repetice se v hudbě objevuje ve dvou variantách – opakování celku a opakování úseku. V příkladu, který vidíme na obrázku, jde o první variantu. Repetice zde má podobu závěrečné taktové čáry, která má navíc dvě tečky. Značí, že se má skladba opakovat znovu od začátku.

Kapitola 3

Neuronové sítě

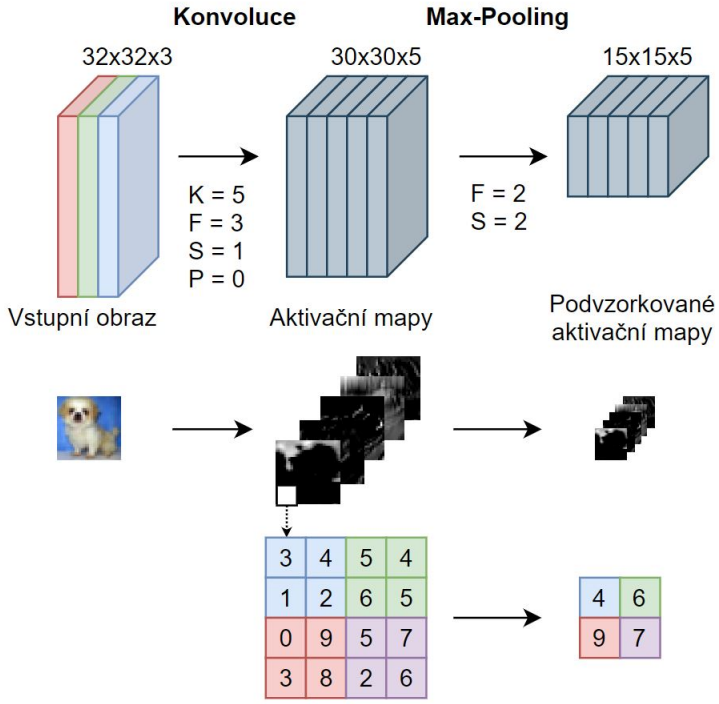
Tato kapitola představuje obecné pojmy, architektury a algoritmy, které jsou využity dále v návrhu řešení a experimentech. popisuje: konvoluční a rekurentní neuronové sítě a jejich kombinaci při vytvoření modelu CRNN a chybovou funkci CTC. Dále představuje architekturu Transformer a Vision-Transformer, vysvětluje princip multi-head attention a definuje hyperparametry pro experimenty.

3.1 Konvoluční a rekurentní neuronové sítě

Konvoluční neuronová síť. Konvoluční neuronová síť (Convolutional Neural Network – CNN) je architektura, která se využívá zejména ve zpracování obrazu díky svým vlastnostem. Skládá se z vrstev, které postupně zpracovávají vstup (obrázek) pomocí naučených konvolučních filtrů. Základem jsou konvoluční a pooling vrstvy. Konvoluční vrstvy zpracovávají data pomocí tzv. konvolučních filtrů. Tyto filtry se vytváří pomocí trénovatelných parametrů, takže se síť postupně učí rozeznávat grafická primitiva a v dalších vrstvách celé tvary, struktury a objekty bez nutnosti zadání člověkem. Pro agregaci této informace jsou však nutné tzv. pooling vrstvy, které se většinou nachází mezi konvolučními vrstvami a zmenšují objem dat předávaných do dalších vrstev pro lepší zpracování. V navrženém modelu se používá tzv. max-pooling, který ze sousedících hodnot vybere maximální. Dále existuje např. average-pooling, které ze sousedících hodnot vypočítává aritmetický průměr. Podrobněji vysvětluje Obr. 3.1

Speciálním případem konvoluční vrstvy je tzv. VGG vrstva (Visual Geometry Group), která obsahuje konvoluční jádra výhradně ve velikosti 3×3. Takové omezení má vliv na chování sítě a její pozitivní vlastnosti právě ve vztahu ke zpracování obrazu.

Rekurentní neuronová síť. Rekurentní neuronová síť (Reccurrent Neural Network – RNN) je další typ architektury. Narozdíl od konvolučních sítí produkujících pevnou délku výsledných dat, rekurentní sítě generují výstupní vektor a fungují pomocí předávání infor-



Obrázek 3.1: Vizualizace příkladu zpracování obrazu jedním blokem CNN. CNN na vstupu dostane obraz o rozměrech 32×32 pixelů se 3 kanály. Ten je zpracován konvoluční vrstvou, která obsahuje $K = 5$ filtrů o velikosti $F = 3$ s krokem $S = 1$ a šířkou okraje $P = 0$. Konvoluční vrstva na výstupu vyprodukuje objem o rozměrech 30×30 pixelů s hloubkou 5, která je rovná počtu filtrů. Objem dále zpracuje Max-pooling vrstva s prostorovým obsahem $F = 2$ a krokem $S = 2$, která objem $2 \times$ podvzorkuje. Obrázek i s popisem převzat z Buchal: *Využití neanotovaných dat pro trénování OCR* [2]

mací mezi tzv. rekurentními buňkami, které jsou propojené mezi sebou. Existují například následující typy rekurentních buněk:

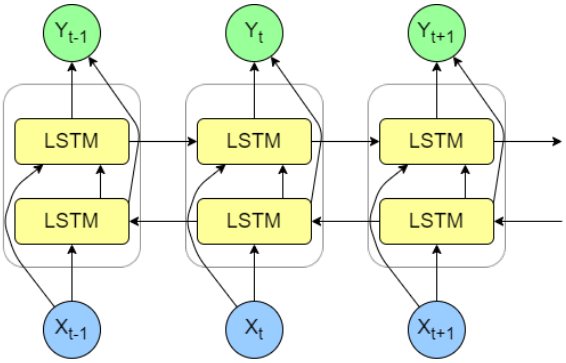
1. STM (Short-term memory) – pouze předává informaci z předešlých vstupů dál. Problémem zde je, že buňky rychle ztrácejí paměť o předešlých krocích, mají tak omezené možnosti.

2. LSTM (Long Short-term memory) – tyto buňky mají větší možnosti práce se vstupy a tak „vidí dále do minulosti“

3. BiLSTM (BiDirectional Long Short-term memory) – struktura složená z buněk typu BiLSTM je schopná předávat informace v obou směrech a umožňuje tak nové funkce samotné architektury. Vizualizace architektury je na obrázku 3.2

Tento odstavec byl inspirován článkem *Long short-term memory* [11]

Kombinace konvoluční a rekurentní sítě V OMR se často využívá architektura CRNN (Convolutional recurrent neural network) [20]. Jedná se o kombinaci konvoluční sítě, která je napojena na rekurentní síť. Zjednodušeně funguje tak, že vstup je nedříve



Obrázek 3.2: Rekurentní neuronová síť typu BiLSTM funguje na principu obousměrné výměny informací mezi buňkami typu LSTM (Long short-term memory). X_i je část vstupní sekvence. Y_i je část výstupní sekvence.

zpracován konvoluční sítí (složené z konvolučních vrstev, pooling vrstev a dalších) a rozdělen na sloupce, které se samostatně zpracovávají v konvoluční síti. Ze sloupců se tak stanou vzorky, které již v sobě mají zakódovanou informaci o obsahu (rozeznání grafických primitiv). Tyto vzorky se dále zpracovávají rekurentní vrstvou, která jim přiděluje význam a podle toho generuje výstup sítě. K trénování se většinou používá chybová funkce CTC [10].

CTC. Connectionist Temporal Classification (CTC) se překládá do češtiny jako "Spojovací Temporální Klasifikace". Je to chybová funkce a používá se např. v OCR [14] k trénování modelů, kde není jasně daná spojitost mezi vstupními a výstupními posloupnostmi. Umožňuje modelu naučit se spojitosti mezi vstupy a výstupy bez potřeby data detailně anotovat. Pomocí algoritmu SGD (Stochastic gradient decent) a zpětné propagace hodnot (back-propagation) tak dodává informace o úspěšnosti předchozích iterací sítí a dává jim informaci potřebnou ke změně koeficientů v síti. [10]

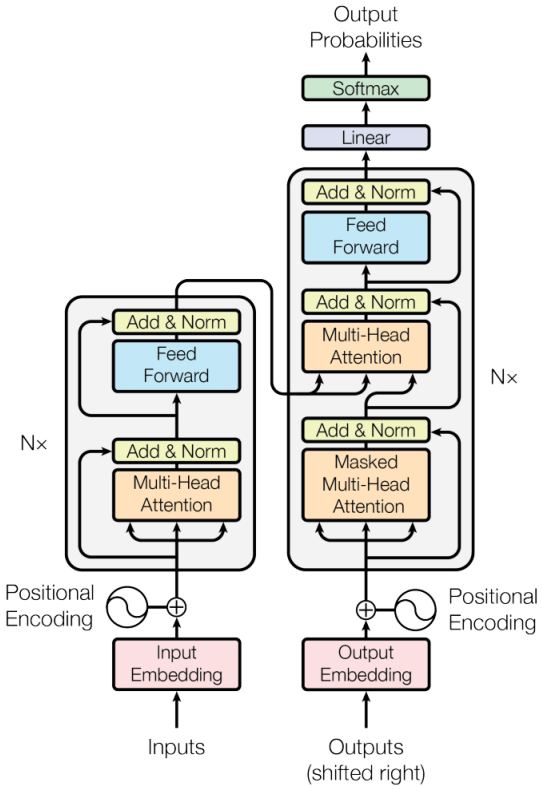
3.2 Transformer

Neuronová síť s architekturou transformer byla prvně představená v článku *Attention is all you need* [23]. Od jiných neuronových sítí se liší zejména využitím tzv. attention vrstev, které umožňují při výpočtu výstupní sekvence zohlednit libovolnou část vstupní i výstupní sekvence. Architektura dále využívá principu zřetězených enkodérů i dekodérů a pozičního kódování. Pomocí nových vlastnosti, které sítě typu transformer přináší, lze dosáhnout zcela nových výsledků v různých polích výzkumu.

Primární využití sítí typu transformer je v oborech zpracování textu. Proto původní architektura [23] pracuje na principu zpracování vstupní textové sekvence na tzv. tokeny. Vstupní abeceda (seznam všech možností vstupů) se skládá ze slov nebo jejich částí, které jsou následně pomocí enkodéru zakódovány do vektoru, tzv. „Input embedding“.

Attention. Převratnou novinkou v transformer architektuře je princip multi-head attention [23], který umožňuje při výpočtu výstupu zohlednit libovolné části vstupní i výstupní sekvence. Architektura využívá dva základní stavební kameny:

1. **Scaled dot-product attention** (škálovaný skalární součin) – Funguje na principu úpravy vstupního tokenu pomocí tří trénovatelných matic a vytváření skalárních sou-



Obrázek 3.3: Architektura základního modelu transformer. Převzato z *Attention is all you need* [23]

činů těchto upravených tokenů s ostatními tokeny sekvence. Mezi tokeny tedy vznikají vztahy, které nesou informaci, jak spolu dané dva tokeny souvisí.

2. **Multi-head attention** (více-hlavová pozornost) – spojuje více výsledků předchozího bloku: scaled dot-product. Každý tak může nést jinou informaci, která přispěje k lepší úspěšnosti sítě.

Detailní vysvětlení je k dispozici v článku *Attention is all you need* [23] v kapitole 3.2 Attention.

Vision-transformer. Vision-transformer je označení pro variantu původní architektury v podobě změny architektury enkodéru. Tato úprava má za důsledek využití obrázků jako vstup sítě místo textu.

Vstupní obrázek se rozdělí na několik částí, většinou pomocí konvolučního enkodéru [8] buď na sloupce, pokud má vstupní obrázek charakter řádku [14], nebo na čtvercové výřezy [22]. Tyto části původního obrázku jsou dále zakódovány do tzv. „Input embedding“, které mají stejnou formu jako zakódované tokeny při zpracování textu. Díky výměně textového enkodéru za konvoluční enkodér lze použít zbytek architektury modelu téměř beze změny. Tato obměna původní architektury se označuje jako Vision-Transformer [8].

Hyperparametry. V rámci experimentů s různým nastavením architektury pro danou úlohu je potřeba definovat vnitřní vlastnosti sítě. Pro experimenty v této práci to jsou následující hyperparametry:

- Velikost vnitřní reprezentace tokenu.
- Počet enkodérů zřetězených za sebou.
- Počet dekodérů zřetězených za sebou.
- Počet attention hlav.

Kapitola 4

Návrhy řešení

V této práci se věnuji konkrétní podúloze OMR, kterou je přepis tištěných notových zápisů na sekvenční textový zápis s end-to-end přístupem. V oblasti monofonního zápisu již existují jak datasety (viz kapitola 2.3), tak ozkoušené přístupy k jejímu vyřešení [5]. V oblasti polyfonního zápisu je ovšem situace jiná a je zde stále prostor pro zlepšení. Neexistuje zde stabilní dataset, avšak existují zdokumentované pokusy jak ho vytvořit ([9], [5]). Dále je otázka, zda architektura CRNN (viz kapitola 3.1) bude u polyfonního zápisu nejúspěšnější architekturou, nebo zde bude dominovat jiná možnost. Na tyto a další otázky se v této práci snažím nalézt odpověď pomocí následujících kroků:

1. Implementaci dvou neuronových sítí pro vlastní experimenty: CRNN a Transformer.
2. Vytvořením nového datasetu s polyfonní hudbou a zefektivnit známý přístup [9]. Viz kapitola 5.2
3. Provedením experimentů s různými hyperparametry s cílem zjisti, která konfigurace je nejlepší. Viz kapitola 6

Pro tuto práci jsem zvolil dva modely, každý s jinými vlastnostmi a s jiným využitím. Oba typy modelů již byly použity v jiném oboru, proto se zde zkouší vhodnost těchto modelů i pro úlohu OMR.

CRNN + CTC. První architektura vytvořená pro experimenty v této práci je tzv. CRNN model (Konvoluční-rekurentní neuronová síť). Skládá se ze dvou částí. První část je konvoluční neuronová síť s VGG vrstvami a aktivační funkcí ReLU. Druhá část je rekurentní síť typu BiLSTM. K trénování obou částí současně se využívá algoritmus CTC.

Tato architektura dosahuje „state-of-the-art“ výsledků v end-to-end OMR ([20], [5]) s monofonní hudbou díky kombinaci dvou komponent, které spolu vytváří dříve nepoznané možnosti. Konvoluční část se konkrétně skládá z vrstev typu VGG s pevným konvolučním jádrem.

Tento model byl zvolen jako baseline pro experimenty pro svou podobnost s předchozím článkem *Empirical evaluation...* [9].

Vision-Transformer Druhá architektura pro experimenty byla vytvořena jako Vision-transformer [8]. Tedy jedná se o spojení konvolučních enkodérů spojených pomocí multi-head attention vrstev a běžných transformerových dekodérů. Za zmínku stojí, že enkodér je tvořen stejnými vrstvami jako v předchozí architektuře RCNN. Na experimentech tedy

bude možno pozorovat, jak se projeví využití mutli-head attention a dalších nových prvků oproti původnímu architektuře s identickým konvolučním blokem.

Shrnutí a srovnání Oba výše zmíněné přístupy (CRNN a Vision-Transformer) pracují na stejném principu. Vstup je obrázek (řádek s notovou osnovou), který je rozdělen na oblasti (sloupce). Z těch se pak extrahují informace pomocí jednoho nebo více konvolučních bloků. Na výstupu je sekvence znaků, která značí přepis notové osnovy do požadovaného sekvenčního kódování. Oba modely využívají přístup OCR, takže generují jednotlivé znaky výstupní abecedy. Ač je to přístup ojedinělý, rozhodl jsem se ho vyzkoušet a zjisti, zda i tak modely dosáhnou požadovaných výsledků.

Podle předchozí literatury je předpokládaná úspěšnost obou modelů podobná v rámci monofonních datasetů. U polyfonních to však není předem jisté.

Kapitola 5

Implementace

Mým přínosem je sada samostatných skriptů v jazyce Python3, nejedná se o celistvý systém.

Lze je nalézt v přiložených souborech (složka src) a zároveň v mém GitHub repozitáři[1]

Většina skriptů funguje jako samostatně spustitelné části procesu. Jejich architektura je však připravená pro možnost budoucího vývoje a spojení samostatných skriptů do jednoho systému. Každý skript obsahuje hlavní třídu, která nese podobný název jako samotný skript. Ta provádí hlavní část programu a navíc definuje funkci `main` volanou v případě spuštění skriptu z příkazové řády. Tato funkce zpřístupňuje parametry třídy jako vstupní argumenty skriptu pomocí knihovny `argparse`.

5.1 Dotvoření systému PERO

K provedení experimentů jsem využil systému PERO [12], [13], [14] vyvinutého skupinou pod vedením Ing. Michala Hradiše, Ph.D. Systém pro trénování modelů se skládá ze dvou repozitářů, jeden z nich je volně dostupný pod názvem `pero-ocr`[2], druhý je skrytý a je dostupný pouze po domluvě. Repozitáře byly použity na definici architektury použitých neuronových sítí, jejich trénování a testování.

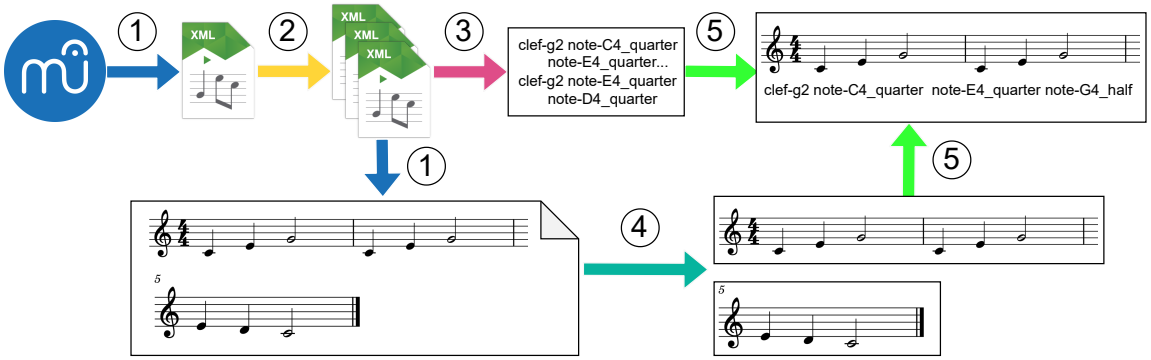
Do systému PERO jsem si však musel doplnit vlastní funkci hodnocení Symbol Error Rate. Pro tento účel jsem vytvořil skript `customwer.py` počítající různé errory za použití knihovny jiwer[3], který je také dostupný mezi odevzdanými skripty. V samotném systému PERO už jsem pak jen doplnil rozhraní komunikace se skriptem `customwer.py`.

5.2 Vytvoření datové sady BMPD

Jak již bylo zmíněno v sekci 2.3, nedostatek anotovaných dat vedl výzkumníky k adoptování kreativních přístupů k vytvoření vlastních anotací pomocí open-source projektů, bez nutnosti spotřebovávat mnoho prostředků na manuální anotaci. Tato kapitola popisuje nový proces vytvoření datasetu pro tuto práci.

Podobně jako v článku *Empirical evaluation...* [9] jsem pro tuto práci zvolil přístup s využitím programu MuseScore a modulu batch-convert, dále jsem pro tyto vytvořil několik

[1] github.com/vlachvojta/bachelor_thesis_OMR
[2] Pero-ocr: Knihovna v pip, GitHub repozitář
[3] pypi.org/project/jiwer/



Obrázek 5.1: Vytvoření datové sady BMPD (Brno MuseScore Polyphonic Dataset.
1: MuseScore Batch Convert, 2: part_splitter.py, 3: gen_labels.py,
4: img_to_staves.py 5: matchmaker.py
Soubory `mscz` jsou konvertovány do `MusicXML`, které jsou rozděleny na jednotlivé party a očištěny od přebytečných symbolů. Party jsou pak 1) převedeny pomocí MuseScore modulu Batch Convert na obrázky stránek, které jsou rozděleny na jednotlivé notové osnovy. 2) převedeny na sekvenční kódování vhodné pro výstup neuronové sítě. Tyto dvě komponenty jsou nakonec spárovány k sobě.

skriptů v jazyce python3 pro úpravu a zpracování souborů. Všechny skripty pro tuto část jsou dostupné na GitHubu[4].

Jsou to zejména následující skripty:

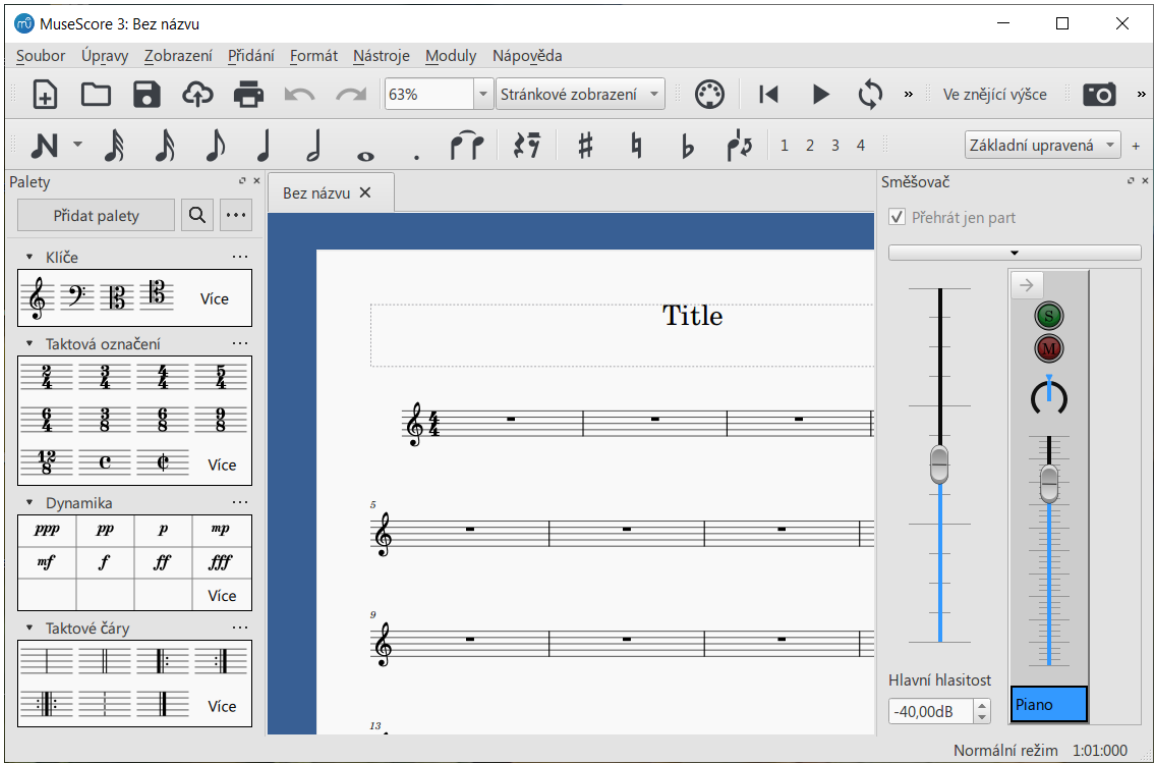
- `part_splitter.py`
- `img_to_staves.py`
- `gen_labels.py`
- `symbol_converter.py`.
- `matchmaker.py`

Jak naznačuje obrázek 5.1, dataset jsem vytvořil pomocí několika kroků. Původní vhodně vybrané soubory `mscz` jsem konvertoval do formátu `MusicXML`, rozdělil na jednotlivé party a očistil od přebytečných symbolů, které nejsou podstatou zkoumání. (Texty písně, dynamika, akordové značky a nákresy...). Party jsou pak 1) převedeny pomocí MuseScore modulu Batch Convert na obrázky stránek, které jsou rozděleny na jednotlivé notové osnovy. 2) převedeny na sekvenční kódování vhodné pro výstup neuronové sítě. Tyto dvě komponenty jsou nakonec spárovány k sobě.

Výsledek jsou obrázky s jednou notovou osnovou a přepisy ve speciálním kódování, které se velmi podobá sémantickému kódování datasetu PrIMuS [5](viz kapitola 2.3).

Následuje bližší představení jednotlivých kroků vytváření datasetu a popis skriptů k tomu využitých.

[4] github.com/vlachvojta/bachelor_thesis_OMR.git



Obrázek 5.2: Screenshot programu MuseScore využitého pro vytvoření konverzi souborů při vytváření dataset BMPD.

MuseScore. MuseScore[5] je zdarma dostupný open-source program pro tvorbu notových zápisů a umožňuje vytvořit velkou škálu běžně dostupných hudebních symbolů a struktur. Jeho hlavní funkce spočívá v možnosti přehrání vytvořeného notového zápisu pomocí virtuálních nástrojů. Patří mezi jedny z nejpoužívanějších programů na tvorbu hudby na světě. Pro výstup nabízí jak základní funkce ukládání obrázků do PNG či PDF, tak i pokročilé funkce jako exportování do zvukových souborů MIDI, MP3 nebo do standardizovaných XML formátů. Tím je například MusicXML[6], který se dá využít právě pro účely automatického zpracování. Díky obousměrnému převodu lze původní notový zápis převést do MusicXML, kde se jednoduše upraví, a poté převést zpět na soubor MuseScore.

Díky open-source podstatě samotného softwaru MuseScore lze do programu přidat moduly, které pomáhají automatizovat manuální práci v softwaru nebo přidávají novou funkcionalitu. Konkrétně lze využít modul Batch convert[7], který umožňuje automatické konvertování více souborů naráz mezi velkým množstvím formátů.

Fórum MuseScore[8] je webové fórum ke sdílení vlastních skladeb a aranží, které obsahuje nezměrné množství zápisů. K datu odevzdání této práce je na fóru 44 000 skladeb vytvořených autory fóra a dalších 1 570 000 skladeb vloženo uživateli. Fórum nabízí možnost přehrát skladby online, stáhnout notový zápis ve formátu mscz nebo exportovaný zápis jako PDF a dále možnosti filtrování pomocí několika kategorií (žánr, počet nástrojů, délka skladby, obtížnost, počet partů a rozlišení na veřejně dostupné zápisy a autorské podle licence).

[5] musescore.org
[6] musicxml.com
[7] musescore.org/en/project/batch-convert
[8] musescore.com

Stažení souborů MuseScore. Z MuseScore fóra jsem hromadně stáhl přes 100 000 souborů, dbal jsem při tom co nejvíce na stahování pouze souborů označených jako „veřejná doména“. Databáze notových zápisů od uživatelů je opravdu obsáhlá a pro další využití je zde k dispozici mnoho dalších souborů s různými vlastnostmi. Všechny stažené soubory jsem poté převedl do formátu `MusicXML` pomocí MuseScore modulu Batch Convert. Se soubory ve formátu XML už lze pracovat automaticky a postupně je konvertovat nebo analyzovat.

Zpracování souborů pomocí formátu MusicXML. Pro tuto část jsem vytvořil skript `part_splitter.py`, který prochází soubory MusicXML a analyzuje je pomocí knihovny lxml[9], která umožňuje snadnou navigaci mezi XML prvky a jejich úpravy. Skript plní následující funkce:

1. Rozdělení souboru na samostatné party
2. Očištění notového zápisu o nechtěné symboly – zakódování do sekvenčního zápisu využívá poměrně omezenou podmnožinu všech hudebních symbolů a struktur, proto jsou v tomto kroku odstraněny tyto symboly z XML reprezentace, aby výsledné obrázky obsahovaly co nejméně nepotřebných informací. Skript eliminuje např. texty písní, akordové značky, dynamiku, prstoklady, jména partů...
3. Analýza vlastností notových zápisů – Skript analyzuje zejména, zda separovaný part obsahuje polyfonní hudbu, a ukládá seznam takových partů do zvlášního souboru.
4. Nastavení počtu notových osnov na stránku – Této funkce se využívá dále při separaci obrázků u tzv. informované metody (viz kapitola 5.2).

Z původních 100 000+ souborů jsem vybral 7 336 partů obsahujících polyfonní hudbu. Party byly vybrány z 3 377 původních souborů, které jsem vybral zároveň z důvodu snahy o porovnatelnost výsledků s článkem *Empirical evaluation...* [9], který zveřejňuje seznam použitých souborů pro vytvoření vlastního datasetu MSPD. Další možnosti vytváření datasetu z původních 100 000+ souborů jsou tedy mnohem rozmanitější a tato skutečnost je ponechána pro budoucí vývoj

Seznamy těchto souborů jsou dostupné jak v příloze, tak na mém GitHubu[10] ve složce BMPD_stats

Vytvoření obrázků Tvoření obrázků probíhalo ve dvou krocích. 1) Vytvoření obrázků celých stránek pomocí modulu Batch Convert z upravených partů ve formátu MusicXML. 2) Ořezání stránek na samostatné notové osnovy pomocí skriptu `img_to_staves.py`, který využívá knihovnu open-cv[11] a kromě samotného ořezu notových osnov je dále převádí na černobílý obrázek a transformuje do stanovené výšky. Pro vytvoření datasetu BMPD byla použita výška 100 pixelů. Šířka obrázku se mění poměrně ke změně výšky za účelem zachování poměru stran.

V rámci ořezávání na notové osnovy jsem experimentoval s různým nastavením s cílem získat z vybraných souborů co nejvíce použitelných dat. Proto jsem implementoval dvě metody ořezávání obrázků. tzv. Naivní a informovanou.

[9] lxml.de
[10] github.com/vlachvojta/bachelor_thesis_OMR.git
[11] opencv.org, návod k instalaci: pypi.org/project/opencv-python

Naivní metoda ořezu využívá domněnku, že mezi každými dvěma notovými osnovami lze najít bílý prostor, jako jakýsi okraj okolo osnovy. Proto skript prochází postupně horizontální pruhy obrázku a hledá bílé pruhy. Na základě jejich pozic pak určí pozice jednotlivých notových osnov a separuje je do souborů.

Problém s touto metodou nastává ve dvou případech. 1) Pokud na vstupní stránce mezi osnovami není dostatečná mezera, skript tyto dvě osnovy vyhodnotí jako jednu. 2) Pokud je osnova sama vyšší než dvě prázdné osnovy, protože např. obsahuje jak nízké, tak vysoké tóny. V obou případech skript označí osnovu za podezřelou a obrázek nelze dál použít. Výsledné obrázky, které nejsou podezřelé, jsou následně oříznuté bez přebytečných symbolů jiných osnov.

Informovaná metoda vyžaduje informaci o počtu osnov na stránce, kterou lze určit pomocí předchozího dělení souboru na party. Informovaná metoda nejdříve shora i sdola ořízne bílý prostor a zbylou část rozdělí rovnoměrně na počet osnov. Mezi obrázky nechává přesah 25 % řádku nahoře i dole, aby se předešlo smazání okrajových symbolů. Výsledné obrázky často obsahují okrajové prvky sousedních osnov. Do trénování modelu se tak přidává další úloha, a to segmentace hlavní notové osnovy a ignorování okolních symbolů.

Vytvoření přepisů a jejich definice Pro generování samotných přepisů jsem poupravil původní skripty z článku *Empirical evaluation...* [9]. Takto upravené kódy jsou dostupné na mém osobním GitHubu[12] jako fork původního repozitáře, aby bylo jasné, kdo kterou část přidal.

Samotné ground-truths jsou vytvořeny pomocí skriptu `gen_labels.py` a aplikují tzv. „Advance position“ kódování stejné jako v článku *Empirical evaluation...* [9]. Toto kódování bylo vytvořeno na základě článku *Approaching end-to-end OMR for homophonic scores* [1]. Původně sice vychází ze Sémantického kódování představeném v datasetu PrIMuS [5]. Je zde však několik rozdílů, proto bez dodatečných úprav není možné datasety kombinovat. Hlavním rozdílem je právě způsob zapsání polyfonní hudby do sekvenčního kódování. V tomto způsobu je obrázek rozdělen na jednotlivé sloupce. Symboly v jednom sloupci jsou odděleny mezerou nebo bílým znakem. Sloupce mezi sebou jsou odděleny znakem „+“.

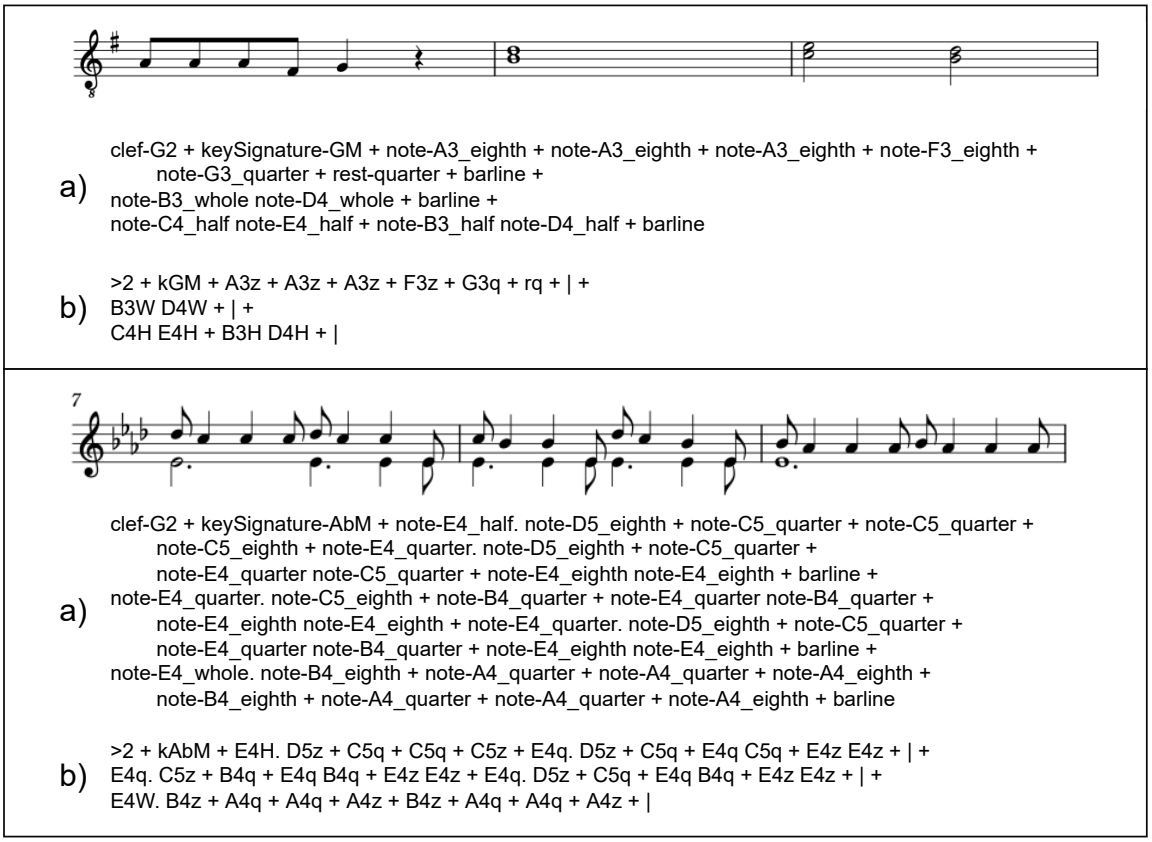
Výsledné kódování obsahuje následující typy znaků: noty (note), pomlky (rest), příznakové noty (gracenote), klíč (clef), předznamenání (keySignature), více násobné prázdné takty (multirest), časové předznamenání (timeSignature) a taktovou čáru (barline).

Zkrácení sekvenčního kódování Kvůli využití OCR přístupů trénovaný model nevnímá hudební symboly jako předem daný prvek ze slovníků výstupních symbolů, ale generuje znak po znaku. Kvůli této charakteristice jsem zkrátil přepisy na minimální počet znaků v rámci zjednodušení trénování.

Ke zkrácení byl použit skript `symbol_converter.py`, který pracuje na principu překládání symbolů pomocí předem daného „slovníku“ načteného ze souboru `JSON`. Příklady takových slovníků jsou vidět v příloze a na GitHubu[13]. Příklad lze vidět na obrázku 5.3

Spárování obrázků a přepisů Jelikož jsou kroky vytvoření obrázků a vytvoření přepisů samostatné procesy, bylo potřeba výsledné soubory správně spojit, aby vznikly správné dvojice obrázek-přepis. Tuto funkci zajišťuje skript `matchmaker.py`, který načte dva seznamy souborů (u přepisů je možnost načítat z jednoho souboru) a podle počtu vzorků pro každý

[12] polyphonic-omr-by-sachindae
[13] github.com/vlachvojta/bachelor_thesis_OMR.git



Obrázek 5.3: Příklady obrázků a jejich přepisů z nové datové sady BMPD. Nový řádek (odstavec) přepisu značí nový takt osnovy.
a) Původní kódování představené v článku *Empirical evaluation...* [9]. Obrázek rozdělen na jednotlivé sloupce. Symboly v jednom sloupci jsou odděleny mezerou nebo bílým znakem. Sloupce mezi sebou jsou odděleny znakem „+“.
b) Zkrácené kódování pro účely trénování modelu.

part buď spáruje dané dvojice, pokud jejich počet odpovídá, nebo označí za chybné. Nakonec tiskne statistiku, kolik souborů bylo zkpoírováno a jeká množství je chybné.

Dokončení datasetu a srovnání Výše uvedené principy a nástroje byly využiti k vytvoření celistvého datasetu. Pojmenoval jsem ho BMPD (Brno MuseScore Polyphonic Dataset). Obsahuje 116 000 párů (obrázků s přepisy) a byl rozdělen na tři podmnožiny. Trénovací sada se skládá ze 114 000 párů, testovací sada BMPD obsahuje 2 000 párů a do testovací sady BMPD-Hard bylo vybráno 400 párů podle kritérií na vlastnosti řádku.

V článku *Empirical evaluation...* [9] autor rozděluje dataset na tři sady: trénovací sadu, testovací sadu MSPD a testovací podmnožinu MSPD-Hard, kterou vybírá jako speciální příklad polyfonní hudby za účelem co nejpřesněji ohodnotit skutečné schopnosti nového modelu. Pro stanovení složitosti přepisu používá průměrnou hustotu not v taktu a vybírá 900 párů s hustotou větší nebo rovnou 41.

Podobně v mém datasetu jsem vytvořil testovací podmnožinu o velikosti 400 párů s hustotou větší nebo rovnou 40. Detailní srovnání všech čtyř testovacích sad je v tabulce 6.2.

Tabulka 5.1: Srovnání vlastností nového datasetu BMPD s variantami datasetu MSPD [9].

		MSPD	BMPD	MSPD-Hard	BMPD-Hard
Počet řádků		18 000	2 000	900	400
Počet symbolů	Min	3	5	41	40
	Průměr	70.59	61.78	79.2	117
	Max	819	285	679	332
Počet taktů	Min	1	1	1	1
	Průměr	4.15	3.96	2.11	1.53
	Max	55	20	8	5
Hustota	Min	3	4	41	40
	Průměr	20.3	19.75	52.8	84.16
	Max	165	137	165	172
Počet hlasů	Min	2	1	2	1
	Průměr	2.05	2.25	2.42	2.06
	Max	4	21	4	17

Přehled dalších možností V rámci vytváření datasetu jsem vyzkoušel několik dalších možností, jak docílit koherentního datasetu, většina z nich však skýtala různá úskalí. Tento seznam slouží jako přehled slepých uliček, které byly vyzkoušeny, ale nakonec jsem od nich upustil.

1. Kromě modulu Batch Convert lze využít rozhraní příkazové řádky, program nabízí mnoho příkazů[14]. Toho lze využít ke konverzi souborů mscz na základní typy souborů (png, mp3, musicxml), kdy se program spustí na pozadí, ale pouze provede konverzi bez spuštění grafického rozhraní. Bohužel zde nefunguje zpětná konverze. Na vstupu program očekává pouze soubor s příponou mscz. Další nevýhodou je fakt, že se program sám o sobě spouští kolem 1 minuty a to platí i pro omezené spuštění pomocí příkazové řádky.

2. Další možností je využít MuseScore ve verzi 4, což je novinka zveřejněná v lednu roku 2023. Tato nová verze nabízí kromě modernizovaného grafického rozhraní rychlejší spuštění samotné aplikace, průměrně se spouští kolem 15 sekund. Zároveň zde platí zpětná kompatibilita s rozhraním příkazové řádky, takže lze využít rychlejšího spouštění programu a konvertovat soubory rychleji.

3. Další možností pro syntetizaci vlastních obrázků je knihovna Lilypond[15], která umožňuje různé operace s hudebními zápisy včetně exportu jako obrázek.

[14] musescore.org/en/handbook/3/command-line-options
[15] lilypond.org

Kapitola 6

Experimenty a vyhodnocení

Tato kapitola představí experimenty, které byly pro účely této práce vykonány. Popíše jejich prostředí, cíle a průběh. Nakonec je vyhodnotí a prezentuje jejich výsledky.

6.1 Vyhodnocení přesnosti přepisů

Experimenty v této práci se soustředí pouze na samotný přepis obrázku na text. Výsledky jsou porovnávány s předchozí literaturou. Kvůli porovnání s článkem Calvo-Zaragoza et al. *End-to-end neural optical music recognition of monophonic scores* [5] použiji následující metriky:

- SeqER (Sequence Error Rate) (%): Poměr chybně rozpoznaných řádků k celkovému počtu řádků. (stačí jedna chyba, aby byl řádek považován za chybný)
- SER (Symbol Error Rate) (%): Poměr elementárních operací (vložení, změna, smazání) s výstupními symboly, které je potřeba provést, aby byl přepis uveden do správného stavu k celkovému počtu symbolů. Počty operací se agregují napříč jednotlivými řádky, a poté se vypočítá poměr všech chyb vůči všem symbolům.

Dále kvůli porovnání s článkem Edirisooriya et al. *An Empirical Evaluation of End-to-End Polyphonic Optical Music Recognition* stanovuji dvě další metriky pracující na podobném principu jako SER:

- PSER (Pitch Symbol Error Rate) (%): Poměr elementárních operací (vložení, změna, smazání) s výstupními symboly, které nesou informaci o výškách tónů k celkovému počtu symbolů.
- RSER (Rhythm Symbol Error Rate) (%): Poměr elementárních operací (vložení, změna, smazání) se zbytkem výstupních symbolů (symboly označující délku not atd.) k celkovému počtu symbolů.

Rozdělení těchto dvou metrik od původního SER má za účel zjistit více informací o průběhu samotného rozpoznávání a identifikaci konkrétních problémů.

Všechny tyto typy hodnocení ovšem mají limitace, co se týče hudebního kontextu. Pokud například systém chybně rozezná typ klíče na začátku osnovy a kvůli tomu všechny noty rozezná chybně, dalo by se argumentovat, že v kontextu hudební sémantiky se jedná o chybu méně závažnou, než samotné chybné rozeznání všech not. Při výzkumu end-to-end OMR

systémů se ovšem na tyto limitace běžně nehledí a používají se dále tyto metriky zejména pro svoji algoritmickou jednoduchost a nezávislost na konkrétním typu kódování.

Kvůli provádění experimentů v OCR prostředí je primární metrika pro chybu tzv. CER (Character error rate), který se počítá jako Levenštejnova editační vzdálenost mezi řetězci. Ta ovšem nemá návaznost na žádnou z předchozích literatur ani neudává přesně informaci o přesnosti kódování ve vztahu k hudebním symbolům, proto se v této práci neuvádí.

6.2 Prostředí Experimentů

Systém PERO, který jsem využil, je zaměřen na výzkum a aplikaci OCR technologí, proto i experimenty v této práci pracují na OCR přístupu (výstup sítě jsou znaky abecedy, ne přímo hudební symboly). Primární metrikou úspěšnosti systému se tedy stává CER (Character Error Rate), který porovnává jednotlivé znaky výstupní sekvence. Další chybové funkce na zpětné odhalení úspěšnosti (SER, SeqER, PitchSER, RhythmSER. Více viz 6.1) jsou odvozené pouze podle výstupních sekvencí a nemají na samotné trénování vliv. Tento přístup je v rámci OMR ojedinělý, ale podle výsledků lze pozorovat, že na úspěšnost samotné úlohy nemá zásadní vliv.

V rámci trénování nebyl použit žádný algoritmus na postupné snižování learning rate. Změny jsem prováděl manuálně podle potřeby, což má za následek například to, že jednotlivé experimenty mezi sebou nemají porovnatelný průběh, pouze výsledky. Protože každý mohl v danou etapu trénování pracovat s jinak nastaveným learning-rate. Hodnotu learning jsem postupně maunálně snižoval podle dosažených výsledků a průběhu trénování. Počáteční hodnota byla vždy 1×10^{-4} a postupně jsem ji skokově snižoval. Minimální vyzkoušená hodnota byla 1×10^{-7}.

Výpočetní zdroje byly poskytnuty projektem e-INFRA CZ (ID: 90140), podporovaným Ministerstvem školství, mládeže a tělovýchovy České republiky[1].

Datová augmentace V průběhu trénování byla na vstupní obrázky použita metoda rozšíření a upravení datové sady (augmentace dat) pomocí geometricky definovaných transformací vstupního obrázku. Takto trénovaný model by bylo v budoucnu možno použít na testování se skenovanými zápisy nebo fotkami. Touto aplikací se však práce nezabývá a je tak ponechána pro budoucí vývoj.

6.3 Experimenty s monofonními zápisy

Nejdříve jsem provedl 4 experimenty s monofonní hudbou a datasetem PrIMuS [5], kde jsem se snažil dosáhnout zlepšení oproti baseline přístupu [20].

Dataset, který obsahuje celkem 87 000 anotovaných řádků, jsem náhodně rozdělil na trénovací a validační sadu. Validační sada obsahovala 2 000 náhodně vybraných řádků a trénovací sada obsahovala zbytek řádků, tedy 85 000. Původní kódování přepisů (Agnostické i Sémantické) jsem zkrátil z důvodu využití OCR technologií pro trénování na kratší sekvence znaků. (viz 5.2) Takto zkrácená kódování jsou dále označována jako SAgnostic a SSemantic.

Primárním cílem experimentů s monofonními zápisy bylo ověřit funkčnost navržených modelů na jednodušší úloze, která již má zdokumentované vyhovující výsledky ([5], [20]), a porovnat je s úspěšností baseline přístupu [20].

[1] metavo.metacentrum.cz/

Sekundárním cílem bylo porovnat složitost dvou typů zápisů (sémantického a agnostického) ve vztahu k oběma architekturám.

Jako baseline pro své experimenty jsem zvolil výsledky popsané v článku Ríos-Vila, et al. *On the use of transformers ...* [20], jednak protože autoři provedli trénování a testování na stejném datasetu (PrIMuS [5]), a jednak protože používanou architekturu (jedná se o Konvoluční síť spojenou s rekurentní a trénováním pomocí CTC) lze přirovnat k mé první navržené architektuře CRNN-CTC (viz kapitola 4).

Aby mohl být experiment považován za úspěšný, měl by dosáhnout podobných nebo lepších výsledků než sama baseline. Požadovaná minimální chybovost je vidět v Tabulce 6.1 v prvním řádku.

Model se trénoval tak dlouho, dokud chybová funkce klesala, nebo alespoň oscilovala kolem stále snižujícího se průměru. Pokud dlouho nepřicházel pokrok, experiment jsem prohlásil za ukončený. Výsledná úspěšnost experimentu (zobrazená v tabulce, v grafu pomocí křížku) je pak aritmetický průměr z posledních pěti měření. Výsledky lze vidět v tabulce 6.1

Tabulka 6.1: Výsledky experimentů s monofonním datasetem PrIMuS [5] a dvěma navrženými modely. Řádek s nejlepším výsledkem je zvýrazněn **tučně**. Pro vysvětlení chyby viz kapitola 6.1

Model	Kódování	SER	SeqER	PSER	RSER
CRNN baseline	Agnost.	1.7	–	–	–
CRNN-CTC	Agnost.	3.65	22.85	4.46	3.6
	Sémant.	1.50	9.35	1.18	0.74
Transformer	Agnost.	0.58	5.35	0.40	0.34
	Sémant.	**0.30**	**2.80**	**0.31**	**0.10**

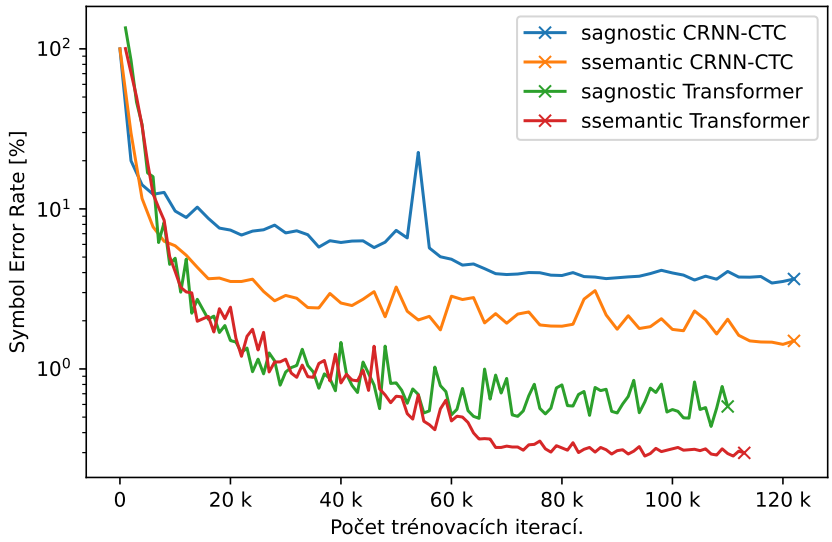
Výsledky experimentu. Dosažené výsledky jsou ke zobrazeny v grafu 6.1 a v tabulce 6.1. Lze usoudit, že oba modely jsou funkční a mohou se tak použít v dalších experimentech. U obou typů kódování dosahuje transformer vždy lepších výsledků, než první síť CRNN-CTC.

6.4 Experimenty s polyfonním datasetem BMPD

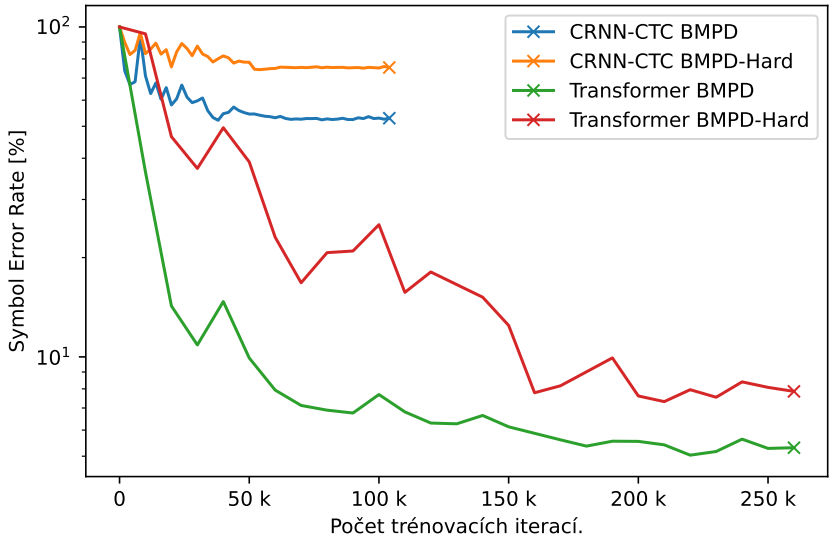
Tato kapitola popisuje dva experimenty s novou polyfonní sadou BMPD vytvořenou pro tuto práci a dvěma zmíněnými modely (CRNN-CTC + Vision-Transformer, viz 4). Výsledky jsou porovnány v kontextu předchozího díla [9].

Primárním cílem je porovnat úspěšnost obou modelů mezi sebou na dvou sadách BMPD a BMPD-Hard. Jejich výsledky lze porovnat pozorováním hodnot chybových metrik. Všechny použité metriky jsou popsány v kapitole 6.1

Sekundárním cílem je srovnat výsledky obou navržených modelů s předchozí literaturou [9]. Dosažené výsledky v tomto článku lze nalézt jako hodnoty PSER a RSER v řádcích tabulky 6.1 s modelem RNNDecoder. Přesné hodnoty je ale nutné brát s rezervou, protože obě testovací sady obou datasetů nemají stejné vlastnosti. Přesnější informaci o jejich rozdílech znázorňuje tabulka 6.2.



Obrázek 6.1: Symbol error rate přepisů při trénování modelů na datasetu PrIMuS [5]. SAgnostic = zkrácené agnostické kódování, SSemantic = zkrácené sémantické kódování. Poslední hodnota (označená křížkem) je aritmetický průměr z posledních 5 hodnot.



Obrázek 6.2: Symbol error rate v průběhu trénování modelů na polyfonním datasetu BMPD a BMPD-Hard. Jako příklad transformerové architektury byla vybrána architektura 512_8h_6e_6d

Výsledky. V tabulce 6.2 je vidět úspěšnost obou navrhovaným modelů. CTC model bohužel nedosahuje přijatelných výsledků, proto se potvrzuje, že v současné podobě není využitelný pro polyfonní hudbu. Podle vývoje má dokonce výrazný problém i s jednodušší sadou BMPD. Navržená architektura CTC nejspíš pro úlohu polyfonní OMR není vhodná.

Tabulka 6.2: Výsledky experimentů s polyfonním datasetem (BMPD a BMPD-Hard) a dvěma navrženými modely, tabulka uvádí trénovací epochu s nejmenší chybovostí pro každý model. Řádek s nejlepším výsledkem v rámci sady je zvýrazněn tučně.
(SER: Symbol Error rate, SeqER: Sequential error rate, PSER: Pitch SER, RSER: Rhythm SER (viz kapitola 6.1).

Sada	**Model**	**SER**	**SeqER**	**PSER**	**RSER**
MSPD	RNNDecoder	–	–	**5.64**	3.92
BMPD	CRNN-CTC	52.90	82.58	37.14	97.15
	Transformer	5.31	25.90	9.02	**3.15**
MSPD-Hard	RNNDecoder	–	–	8.57	5.82
BMPD-Hard	CRNN-CTC	75.34	100	61.84	90.75
	Transformer	7.86	55.69	**8.40**	**4.62**

Naopak transformerová síť prokazuje svou všestrannost a relativně bezpečně si poradí i polyfonní úlohou. Za zmínku stojí naměřený SeqSER, který nám prozrazuje, že i při výsledcích SER v jednotkách, pro daný model SeqSER může dosáhnout hodnot i přes 50 %.

Můžeme sledovat také trend rozdílu rozpoznání rytmických symbolu (RSER) a výškových symbolů (PSER). Tento trend jasně říká, že napříč modely je RSER vždy nižší a rozpoznání rytmu je tedy pro model jednodušší. Oproti tomu PSER dosahuje horších výsledků, což může být způsobeno například poměrem rytmických symbolů ku hudebním. Méně symbolů => Rychleji roste chyba v procentech.

Při celkovém pohledu na úspěšnosti modelů v tabulce 6.2 lze tedy pozorovat, že transformerové sítě fungují s podobným výsledkem jako předchozí architektury. Bohužel výsledky nelze porovnat přesně zejména kvůli dosud neexistujícímu datasetu pro polyfonní hudbu a end-to-end přístup.

6.5 Porovnání různých architektur transformeru

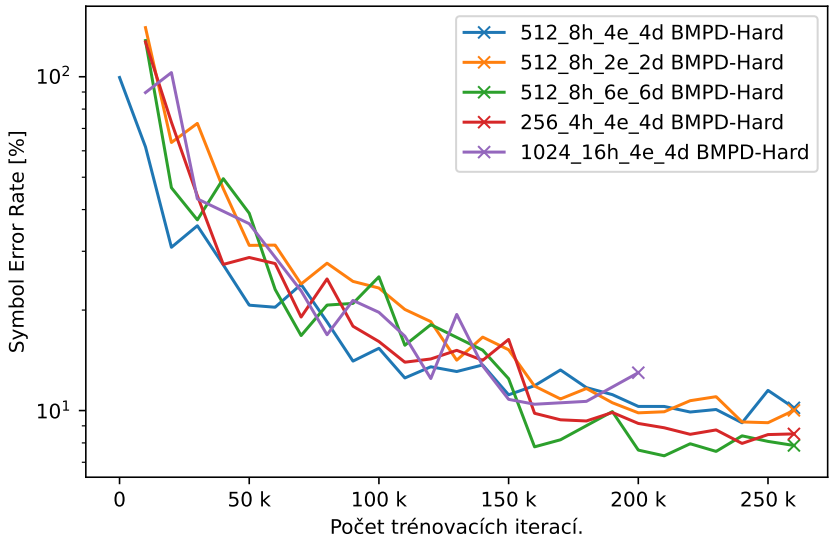
Mým posledním pokusem bylo vyzkoumat, jak bude na průběh trénování a samotný výsledek působit změna hyperparametrů v transformeru. Zkoumané jsou následující hyperparametry:

- Velikost vnitřní reprezentace tokenu.
- Počet attention hlav.
- Počet enkodérů zřetězených za sebou.
- Počet dekodérů zřetězených za sebou.

Vytvořil jsem 5 modelů s informacemi o parametrech přímo v názvu (viz Tabulka 6.3). Všechny experimenty byly puštěny s datasetem BMPD-Hard a graf 6.3 zobrazuje výsledky, kterých jednotlivé modely dosáhly v maximální době trénování nastavené na 250 000 trénovacích iterací.

Tabulka 6.3: Pět zkoumaných modelů, jejich hyperparametry a úspěšnost rozpoznání sady BMPD-Hard při trénování se sadou BMPD. Nejnižší chybovost je označena **tučně**

Název modelu	Velikost tokenu	Hlav	Enkodérů	Dekodérů	SER (%)
512_8h_4e_4d	512	8	4	4	10.19
512_8h_2e_2d	512	8	2	2	10.01
512_8h_6e_6d	512	8	6	6	**7.86**
256_4h_4e_4d	256	4	4	4	8.51
1024_16h_4e_4d	1024	4	4	4	12.99



Obrázek 6.3: Výsledky 5 experimentů s různým nastavením hyperparametrů transformerů. Podle názvu jsou to: Velikost tokenu, Počet hlav, Počet enkodérů, Počet dekodérů. Poslední prvek v daném experimentu (označen křížkem) je označen za výslednou přesnost modelu. Je vypočítán jako průměr z posledních pěti měření

Výsledky. Z grafu 6.3 a tabulky 6.3 lze vyčíst, že se zadanými podmínkami si nejlépe vedla architektura 512_8h_6e_6d

Je možné, že by se experimenty ještě vyvíjeli dál a výsledky se trochu obměnily, pokud by nebyla pevná hranice 250 000 iterací. To je patrné například při pohledu na model 1024_16h_4e_4d, který se nepovedlo v časové dotaci natrénovat tak daleko jako ostatní modely. Větší architektura nevyhnutelně zvedá nároky na výpočetní sílu a čas výpočtů.

6.6 Zhodnocení

Na experimenty s monofonním datasetem byly stanoveny dva cíle. 1) otestovat funkčnost navržených modelů na jednodušší úloze. Podle přesně měřitelných výsledků lze pozorovat, že modely fungují, a dokonce dosahují zlepšení oproti baseline z předchozí literatury. 2) porovnat dva typy zápisů (Agnostický a Sémantický) a zjistit, který se dá využít pro přesnější

výsledný systém. Tento experiment odhalil, že v případě neuronových sítí CTC i seq2seq se lépe pracuje se stylem informací, které nesou sémantický význam. viz 6.1

Na polyfonní experimenty byly cíle následující. 1) porovnat dva navržené modely ve výsledku při trénování. Transformery se jasně osvědčily s výsledky mnohem více vyhovujícími. 2) porovnat výsledky s baseline RNNDecoder. Podle výsledků v tabulce 6.2 lze vidět, že oba modely se postupně trénují, pouze Transformer dosahuje mnohem lepších výsledků.

V porovnání s baseline se může zdát, že experiment dosáhl zlepšení, nicméně oba systémy byly nezávisle testovány na dvou různým datasetech a proto nesdílí žádnou společnou hodnotu a čísla jsou zde uvedena spíš pro představu o kontextu předchozí literatury.

Poslední experiment měl za cíl zjistit, jaké nastavení hyperparametrů bude nejúspěšnější, nakonec se jim stal model 512_8h_6e_6d. Větší počet enkodérů a dekodérů zjevně působí pro lepší pochopení a rozeznání symbolů.

Kapitola 7

Závěr

Cílem práce bylo experimentovat s různými architekturami neuronových sítí k využití na automatické rozpoznání polyfonních hudebních zápisů a překonat předchozí výsledky. Podařilo se vytvořit dvě architektury (CRNN-CTC a Vision-Transformer) a jejích varianty a porovnat je mezi sebou i v kontextu předchozích článků. Pro účel trénování a testování byla vytvořena datová sada BMPD s polyfonní hudbou a těžkou podmnožinou BMPD-Hard pro konkrétnější hodnocení. Záměr práce byl splněn a dokazují to jak statistiky obsahu datové sady, tak úspěšnosti modelu transformeru na datové sadě BMPD-Hard. Původní model CRNN-CTC pro úlohu polyfonního rozpoznání nedosahoval vyhovujících výsledků, a proto se k řešení nehodí.

Z této práce si odnáším detailnější porozumění základních stavebních kamenů umělé inteligence a jejich využití v praxi, dále praktické zkušenosti trénování modelů na vzdáleném serveru a v neposlední řadě hlubší porozumění samotného oboru OMR.

V budoucnu bych rád tuto práci rozšířil co nejvíce lidem z oboru, aby jim mohla posloužit, a dále pracoval na praktickém využití zjištěných poznatků, aby práce alespoň zpětně dostala prakticky využitelný rozměr.

Literatura

[1] Alfaro Contreras, M., Calvo Zaragoza, J. a Iñesta, J. M. Approaching end-to-end optical music recognition for homophonic scores. In: Springer. *Pattern Recognition and Image Analysis: 9th Iberian Conference, IbPRIA 2019, Madrid, Spain, July 1–4, 2019, Proceedings, Part II 9*. 2019, s. 147–158.

[2] Buchal, P. *Využití neanotovaných dat pro trénování OCR*. Brno, CZ, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Dostupné z: https://www.fit.vut.cz/study/thesis/24175/.

[3] Byrd, D. a Simonsen, J. G. Towards a standard testbed for optical music recognition: Definitions, metrics, and page images. *Journal of New Music Research*. Taylor & Francis. 2015, sv. 44, č. 3, s. 169–195.

[4] Calvo Zaragoza, J., Jr, J. H. a Pacha, A. Understanding optical music recognition. *ACM Computing Surveys (CSUR)*. ACM New York, NY, USA. 2020, sv. 53, č. 4, s. 1–35.

[5] Calvo Zaragoza, J. a Rizo, D. End-to-End Neural Optical Music Recognition of Monophonic Scores. *Applied Sciences*. MDPI AG. Apr 2018, sv. 8, č. 4, s. 606. DOI: 10.3390/app8040606. ISSN 2076-3417. Dostupné z: http://dx.doi.org/10.3390/app8040606.

[6] Chiu, C.-C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P. et al. State-of-the-art speech recognition with sequence-to-sequence models. In: IEEE. *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. 2018, s. 4774–4778.

[7] Chowdhury, A. a Vig, L. An efficient end-to-end neural model for handwritten text recognition. *ArXiv preprint arXiv:1807.07965*. 2018.

[8] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv preprint arXiv:2010.11929*. 2020.

[9] Edirisooriya, S., Dong, H.-W., McAuley, J. a Berg Kirkpatrick, T. An Empirical Evaluation of End-to-End Polyphonic Optical Music Recognition. *ArXiv preprint arXiv:2108.01769*. 2021.

[10] Graves, A., Fernández, S., Gomez, F. a Schmidhuber, J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: *Proceedings of the 23rd international conference on Machine learning*. 2006, s. 369–376.

[11] Graves, A. a Graves, A. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*. Springer. 2012, s. 37–45.

[12] Kišš, M., Beneš, K. a Hradiš, M. AT-ST: self-training adaptation strategy for OCR in domains with limited transcriptions. In: Springer. *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part IV 16*. 2021, s. 463–477.

[13] Kodym, O. a Hradiš, M. Page layout analysis system for unconstrained historic documents. In: Springer. *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16*. 2021, s. 492–506.

[14] Kohút, J. a Hradiš, M. TS-Net: OCR trained to switch between text transcription styles. In: Springer. *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part IV 16*. 2021, s. 478–493.

[15] Lisá, D. *Hudební nauka pro malé i větší muzikanty*. Editio Bärenreiter Praha, 2003. ISBN 80-86385-20-5.

[16] MARC, S. *The everything reading music book: A step-by-step introduction to understanding music notation and theory*. United States of America: Library of Congress Cataloging-in-Publication Data, 2005. ISBN 978-1-59337-324-5.

[17] Mayer, J. a Pecina, P. Synthesizing Training Data for Handwritten Music Recognition. In: Springer. *Document Analysis and Recognition–ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part III 16*. 2021, s. 626–641.

[18] Pacha, A. *Optical Music Recognition Datasets: Collection of datasets used for Optical Music Recognition* [online]. Github.com, 2022 [cit. 2023-05-07]. Dostupné z: https://apacha.github.io/OMR-Datasets/.

[19] Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marcal, A. R., Guedes, C. et al. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*. Springer. 2012, sv. 1, s. 173–190.

[20] Ríos Vila, A., Iñesta, J. M. a Calvo Zaragoza, J. On the use of transformers for end-to-end optical music recognition. In: Springer. *Pattern Recognition and Image Analysis: 10th Iberian Conference, IbPRIA 2022, Aveiro, Portugal, May 4–6, 2022, Proceedings*. 2022, s. 470–481.

[21] Shatri, E. a Fazekas, G. Optical music recognition: State of the art and major challenges. *ArXiv preprint arXiv:2006.07885*. 2020.

[22] Shinde, O., Gawde, R. a Paradkar, A. Image Caption Generation Methodologies. 2021.

[23] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L. et al. Attention is all you need. *Advances in neural information processing systems*. 2017, sv. 30.

[24] Wel, E. van der a Ullrich, K. Optical music recognition with convolutional sequence-to-sequence models. *ArXiv preprint arXiv:1707.04877*. 2017.

[25] Zhang, J., Du, J., Zhang, S., Liu, D., Hu, Y. et al. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition*. Elsevier. 2017, sv. 71, s. 196–206.

Příloha A

Obsah přiloženého paměťového média

- `dataset/` – adresář se samotným datasetem BMPD
- `docs/` – adresář s technickou zprávou
- `src/` – adresář se zdrojovými kódy k vytváření datasetu BMPD a hodnocení experimentů.
- `video/` – adresář s videem prezentující moji práci