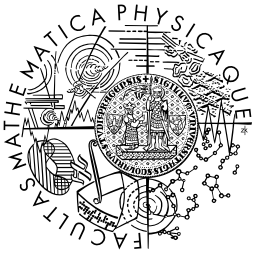


MATEMATICKO-FYZIKÁLNÍ
FAKULTA
Univerzita Karlova

DIPLOMOVÁ PRÁCE

David Vondrák

Rozpoznávání a klasifikace učebnic pomocí hlubokého učení

Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce:	doc. RNDr. Pavel Pecina, Ph.D.
Studijní program:	Informatika
Studijní obor:	ISS

Praha 2022

Prohlašuji, že jsem tuto diplomovou 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

Chtěl bych poděkovat především doc. RNDr. Pavlu Pecinovi, Ph.D. za trpělivost, ochotu a podporu při vedení mé práce i za všechny jeho cenné rady a připomínky. Děkuji také své sestře Barboře Vondrákové za její pomoc a za odvedenou práci na anotaci knih.

Název práce: Rozpoznávání a klasifikace učebnic pomocí hlubokého učení

Autor: David Vondrák

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: doc. RNDr. Pavel Pecina, Ph.D., Ústav formální a aplikované lingvistiky

Abstrakt: Cílem práce bylo použití hlubokého učení k rozpoznávání učebnic a jejich klasifikaci do vyučovacích předmětů a úrovní, a to na základě textových údajů, jako je název knihy, autor, nakladatel a stručný popis obsahu. Jako součást práce formulujeme vlastní definici učebnice, vytváříme dataset pomocí extrakce údajů ze zdrojů dostupných na internetu a ručně anotujeme trénovací a testovací množinu dat. Pro klasifikaci používáme jako baseline naivní bayesovský klasifikátor, z neuronových sítí pak konvoluční a rekurentní architekturu i jejich kombinace. Porovnáváme také různý způsob reprezentace dat vektory (tzv. word embedding) a dosažené výsledky podrobně analyzujeme. Výsledný nejlepší model dosahuje na testovacích datech u všech tří úloh vysokých úspěšností a nabízí se tak možnost jeho uplatnění v praxi.

Klíčová slova: hluboké učení, klasifikace knih, neuronové sítě, učebnice

Title: Recognition and classification of textbooks by deep learning

Author: David Vondrák

Institute: Institute of Formal and Applied Linguistics

Supervisor: doc. RNDr. Pavel Pecina, Ph.D., Institute of Formal and Applied Linguistics

Abstract: The aim of the thesis was to use deep learning methods for recognizing textbooks and classifying their subject and level, based on text parameters, like name of the book, author, publisher or brief content description. As part of the thesis, we formulate custom definition of textbook, create a dataset by extracting data from source available on the internet and manually label train and test dataset. We use naive bayes classifier as a baseline and then neural networks with convolutional, recurrent or combined architectures. We compare various methods of representing data with vectors (word embedding) and analyze the results in detail. Resulting best model reaches high accuracy in all three tasks which suggests the possibility of its application in practice.

Keywords: deep learning, book classification, neural networks, textbook

Obsah

Úvod

Kniha je bezpochyby jedním z nejdůležitějších vynálezů v historii lidstva a má svůj velký podíl na vzniku moderní vyspělé civilizace. Ačkoliv první knihy se datují již do starověku, svému radikálnímu rozšíření vděčí až vynálezu knihtisku, který učinil knihy výrazně levnějšími a dostupnějšími široké veřejnosti. S dalším rozvojem tiskařského lisu a za současného uvolňování přísné cenzury se postupně zvyšoval objem nově vydávaných knih. V polovině 20. století to bylo pouze v Evropě přibližně 200 000 nových knih ročně[1].

V současné době vychází každým rokem jen v České republice přibližně 15 000 knih. Aby bylo možné se v tomto množství nějakým způsobem orientovat, vzniká zde potřeba knihy třídit. To je možné mnoha různými způsoby – můžeme třídit knihy podle jazyka, ve kterém jsou psány, podle média (tištěná kniha, e-kniha), podle autora nebo podle obsahu. Na základě obsahu pak můžeme rozlišovat fikci a literaturu faktu, případně knihy dělit do různých žánrů – romány, poezie, kuchařky apod. V této práci se zaměříme na žánr, se kterým má zkušenost každý z nás ze svých školních let, a sice na učebnice.

Vzhledem ke zmíněnému objemu, ať už existujících či nově vydávaných knih, je vhodné, aby třídění knih mohlo probíhat automaticky. Toho se pokusíme dosáhnout pomocí technologie, která v posledních letech rychle stoupá na oblibě – pomocí hlubokého učení. Sestavíme model, který bude pracovat s několika textovými údaji o knihách – konkrétně jde o název, autora, nakladatele a anotaci/popis knihy – a dokáže na jejich základě rozhodnout, jestli je daná kniha učebnicí, či nikoliv. V druhé fázi pak navrhneme další modely, které vytříděné učebnice zařadí do kategorie podle předmětu, k jehož výuce slouží, a do úrovně, resp. do stupně vzdělávacího procesu – učebnice pro první stupeň ZŠ, učebnice pro SŠ apod.

Motivace

Nabízí se otázka, proč jsme si vybrali právě učebnice a jejich klasifikaci. Motivací byla především potřeba takového modelu v reálné situaci a zároveň to, že se nám nepodařilo najít existující práci, která by se konkrétně tímto problémem zabývala.

Díky širokému využití učebnic ve všech stupních vzdělávacího systému představuje trh s učebnicemi (ať už novými či použitými) velmi atraktivní obchodní příležitost. Z tohoto důvodu vznikají na internetu portály a e-shopy, které se na prodej učebnic přímo specializují. A právě k tomu, aby bylo možné automaticky, bez manuální práce, moderovat, co se na takových webech bude nabízet, je nutné mít algoritmus, který rozhodne, co učebnice je a co naopak do nabídky nepatří.

Když už máme sortiment omezený na učebnice, je jako další krok nutné umožnit zákazníkům učebnice filtrovat. A nejběžnější dva filtry jsou právě podle předmětu a úrovně – například chceme učebnice chemie pro střední školy, matematiku pro první stupeň základní školy a tak dále.

Náš algoritmus bude tedy použitelný na jakémkoliv e-shopu, který chce nabízet učebnice, případně na portálu, kde uživatelé mohou nabízet své použité

[1] https://cs.wikipedia.org/wiki/Kniha

učebnice. Algoritmus bude určovat, co se bude nabízet a nabízené produkty řadit do kategorií dle předmětu a úrovně.

Struktura práce

V první kapitole se pokusíme formalizovat cíl této práce. Uvedeme několik možných definic učebnice z různých zdrojů a z těchto definic následně vyjdeme při formulování vlastní definice učebnice, kterou budeme používat v této práci. Poté podrobněji popíšeme klasifikaci učebnic do předmětů a úrovní a na závěr specifikujeme údaje (metadata) o knihách, které budeme pro všechny klasifikační úlohy využívat.

Ve druhé kapitole jsou podrobně popsány veškeré metody použité v této práci. Zmíníme metody pro extrakci dat z webových stránek, předzpracování surových dat, propojení heterogenních dat z různých zdrojů a také jak připravit data pro zpracování neuronovými sítěmi. Následuje popis metod strojového učení, které budeme používat pro klasifikaci knih. Na závěr kapitoly se podíváme na existující práce zaměřené na klasifikaci knih, na to, jaké metody využívají a jakých výsledků jejich autoři dosáhli.

Třetí kapitola se věnuje získávání a přípravě dat, na cestě od veřejně dostupných webových stránek až po sady dat, které se dají přímo vložit do modelů hlubokých sítí. Popíšeme, jaké máme zdroje, jaké informace a jakým způsobem z nich budeme získávat, jak surová data vyčistíme a také se budeme zabývat sloučením dat z různých zdrojů do jedné homogenní sady dat. Jelikož se nám nepodařilo získat z žádného zdroje vhodně anotovaná data, obsahuje tato kapitola také informace o anotaci dat a příslušné instrukce pro anotátory.

Ve čtvrté kapitole se zaměříme na konkrétní modely strojového učení. Nejprve popíšeme způsoby, jakými budeme připravená data předzpracovávat tak, aby jim jednotlivé modely „rozuměly“, uvedeme jednoduchý, avšak poměrně efektivní model, který nám poslouží jako baseline a podrobně rozebereme používané neuronové sítě. Kromě popisu struktury sítí a nastavení jejich hyperparametrů zmíníme také metody, jak budeme modely mezi sebou porovnávat.

V páté kapitole shrneme výsledky modelů ze čtvrté kapitoly v jednotlivých úlohách. Porovnáme, jak si vedou neuronové sítě ve srovnání s jednodušším baseline algoritmem, jaký typ předzpracování dat dává nejlepší výsledky a otestujeme, zda jsou rozdíly mezi jednotlivými modely statisticky signifikantní. Vyzkoušíme také, jak si různé architektury neuronových sítí povedou na zcela nových testovacích datech. Nakonec tyto výsledky podrobně analyzujeme a rozebereme chyby, které modely dělají.

V závěru pak zhodnotíme přínosy práce a využitelnost výsledných modelů v praxi. Navrhneme také další možnosti, jak klasifikaci učebnic zpřesnit a jak výsledky této práce zobecnit.

1. Definice problému

V této kapitole se budeme věnovat formálnímu popisu úloh, které tato práce řeší, zavedeme všechny potřebné definice a vymezíme množinu údajů o knihách, kterou budeme používat.

1.1 Rozpoznávání učebnic

Úloha 1

Rozhodnout, jestli daná kniha je učebnicí, či nikoliv. (binární klasifikace)

Na první pohled se zdá úloha poměrně jednoduchá, dostaneme informace o knize a na jejich základě odpovíme ano/ne – jde o učebnici, nebo nikoliv. V případě mnoha knih se skutečně bude jednat o jednoduché rozhodnutí – například „Babička“ od Boženy Němcové učebnice určitě není, naopak „Matematika pro 5. ročník ZŠ“ učebnicí zřejmě je. Existuje však také velké množství knih, které jsou na pomyslné hranici toho, co bychom mohli označit za učebnici, a pokud bychom se zeptali na názor různých lidí, dostali bychom pravděpodobně různé odpovědi. Protože se však chceme vyhnout tomu, aby řešení naší rozhodovací úlohy záviselo na názoru konkrétního člověka, pokusíme se zmíněnou hranici co nejpřesněji vyznačit a definovat, co konkrétně budeme v naší práci považovat za učebnici.

1.1.1 Jak definovat učebnici

Přestože asi pro nikoho není problém říct příklad nějaké učebnice a všichni mají určitou představu o tom, co se pod pojmem učebnice rozumí, dát dohromady přesnou definici, podle níž by bylo možné knihy třídit na učebnice a ne-učebnice, se ukazuje jako poměrně složitý a komplexní problém.

Začněme nejprve tím, že se podíváme, jak učebnici definují různé zdroje dohledatelné na internetu. Následující větou začíná článek s názvem „Učebnice“ na Wikipedii[1]:

> *Učebnice je školní učební pomůcka (většinou kniha) pro žáky a studenty, určená k výuce.*

Slovník spisovného jazyka českého[2] říká o heslu „učebnice“ toto:

> *Kniha sloužící k učení, obsahující předepsanou vyučovací látku.*

V České terminologické databázi knihovnictví a informační vědy (TDKIV)[3] se lze dočíst následující obsáhlejší definici učebnice:

[1] https://cs.wikipedia.org/wiki/Učebnice
[2] https://ssjc.ujc.cas.cz/search.php?hledej=Hledat&heslo=učebnice
[3] https://aleph.nkp.cz/F/?func=direct&doc_number=000001062&local_base=KTD

> *Dokument sledující didaktické cíle a podávající výklad poznatků z určité teoretické nebo praktické oblasti. Většinou je určený vymezeným skupinám uživatelů. V praxi se uplatňují čtyři základní modely učebnice, a to tradiční, programová, problémová a kombinovaná učebnice.*

Přestože se výše zmíněné definice navzájem odlišují, lze z nich vyčíst minimálně tři základní poznatky o tom, co by měla učebnice splňovat:

1. Je určená k výuce, obsahuje vyučovanou látku.
2. Má vymezenou skupinu uživatelů – žáci a studenti.
3. Obvykle jde o knihu (my budeme uvažovat pouze knihy, viz dále).

Tyto body nám dávají již poměrně dobrou představu, podle čeho učebnice a ne-učebnice rozlišovat. Pro naše účely však potřebujeme jednoznačnější definici, a tedy stanovit jasné hranice. Toho se pokusíme dosáhnout pomocí několika omezení, resp. zjednodušení, která zavedeme v následující sekci.

1.1.2 Zjednodušení

Omezíme se pouze na učebnice pro základní školy, střední školy, odborná učiliště, autoškoly a jazykové kurzy/samostudium. Toto omezení má za cíl ohraničit obory, kterým se učebnice, jenž budou splňovat naši definici, mají věnovat. Vyučovaná látka na základních a středních školách je vymezena osnovami, na odborných učilištích je tomu podobně. V autoškolách se pak samozřejmě vyučují pravidla silničního provozu a v jazykových kurzech daný cizí jazyk.

Vysokoškolské učebnice (skripta) pro nás učebnicemi nebudou, protože zde už neexistují osnovy, často mají školy vlastní skripta a kromě toho studenti čerpají znalosti z monografií a dalších vědeckých publikací. To by znamenalo, že připuštěním VŠ učebnic by do této kategorie mohla spadat téměř veškerá odborná literatura, čehož bychom se chtěli vyvarovat.

Za učebnice nebudeme považovat ani knížky pro předškoláky či různé dětské knížky. Opět zde platí, že neexistují žádné osnovy a bylo by složité určit nějakou hranici, co ještě učebnice je a co už není. Navíc je otázka, zda lze různé knihy aktivit pro předškoláky považovat za učebnici k výuce, vzhledem k tomu, že se často jedná spíše o zábavné aktivity, jejichž „vedlejším účinkem“ je výuka nějakých znalostí a dovedností, než o výuku v pravém slova smyslu.

Třetí a poslední skupinou „učebnic“, které v naší práci za učebnice považovat nebudeme, jsou populárně naučné knihy. Tato skupina může zahrnovat knihy od populárně naučných o vědě (vesmír, lidské tělo apod.), přes počítačovou literaturu (učebnice programování, Photoshopu), až po různé hobby aktivity (zahrádkaření, pletení, pečení).

Omezíme se pouze na knihy, kterým bylo přiřazeno ISBN. Přestože si většina lidí pod pojmem učebnice představí nejspíše knihu, v širším slova smyslu by šlo pod tento pojem zahrnout i další učební pomůcky, jako např. audio CD,

tabulky, plakáty atd. V této práci však zůstaneme pouze u knih a cílem bude, aby se nic jiného než knihy v našich datech neobjevilo. Toho dosáhneme jednoduše tím, že budeme požadovat, aby vše mělo platný identifikátor ISBN.

Existují sice knihy, které ISBN nemají (např. některá vysokoškolská skripta), avšak vzhledem k tomu, na které cílové skupiny se zaměřujeme (viz předchozí omezení), nepředstavuje to pro nás takový problém – učebnice pro ZŠ/SŠ a jazykové učebnice v naprosté většině případů ISBN přiřazeno mají.

Požadavek na existenci ISBN pro nás bude mít ještě jednu výhodu – jde o prakticky jediný identifikátor, jak jednoznačně určit knihu – ostatní údaje, ať už název, autor nebo nakladatel, mohou být v různých zdrojích uvedeny mírně odlišně. ISBN má však standardizovaný formát a díky tomu budeme moci navzájem párovat záznamy z vícero zdrojů.

1.1.3 Definice učebnice pro tuto práci

Shrneme-li všechny poznatky zmíněné v této kapitole, můžeme říci, že učebnice je věc, která splňuje následující podmínky:

1. Jedná se o knihu, která má přiřazeno ISBN.
2. Sleduje didaktické cíle a podává výklad poznatků z určité teoretické nebo praktické oblasti, nebo slouží k procvičování a opakování těchto poznatků.
3. Její cílovou skupinu tvoří žáci a studenti základních a středních škol, odborných učilišť, autoškol a jazykových kurzů.
4. Je zřejmé, že je určena k použití při výuce na základních a středních školách, odborných učilištích, v autoškole, v jazykových kurzech či při samostudiu jazyka.
5. Je psána v českém jazyce; jedná-li se o jazykovou učebnici, může být psána ve vyučovaném jazyce.

Některé z bodů této definice ještě objasníme podrobněji:

Sleduje didaktické cíle a podává výklad poznatků z určité teoretické nebo praktické oblasti, nebo slouží k procvičování a opakování těchto poznatků. Učebnice musí být napsána primárně za účelem výuky nějakého tématu nebo procvičení jeho znalosti. Může se tak jednat i o cvičebnice, sbírky úloh nebo procvičování gramatiky cizího jazyka. Za učebnici však nebudeme považovat knihy, které znalost procvičují „nepřímo“, třeba zjednodušenou četbu, anglické křížovky apod.

Její cílovou skupinu tvoří žáci a studenti základních a středních škol, odborných učilišť, autoškol a jazykových kurzů. Jak jsme již rozebrali v předchozí sekci, touto podmínkou se vyřazují vysokoškolská skripta a knihy určené studentům VŠ, stejně jako populárně naučné knihy, určené pro širokou veřejnost. Rovněž nebudeme jako učebnice označovat žádné knihy pro předškoláky.

Lze o ní předpokládat, že bude běžně využívána při výuce na základních a středních školách, odborných učilištích, v autoškole, v jazykových kurzech či při samostudiu jazyka. Na základě názvu knihy, popisu knihy, případně dalších informací by mělo být jasné, že kniha může být využívána přímo během výuky nebo k opakování znalostí získaných při výuce.

Je psána v českém jazyce; jedná-li se o jazykovou učebnici, může být psána ve vyučovaném jazyce. Budeme se zabývat pouze českými učebnicemi, pro české studenty, případně o učebnice češtiny pro cizince. Jazykové učebnice mohou být „univerzální“, psané v jazyce, který vyučují. Pokud se tedy objeví v datech například slovenská učebnice matematiky pro střední školy, označíme ji jako ne-učebnici, přestože ostatní podmínky naší definice splňuje.

Na závěr uvedeme několik typů knih, které by se daly považovat za hraniční případy a ani podle naší definice by nemuselo být zřejmé, jestli jsou učebnicemi, či nikoliv.

Za učebnice budeme považovat:

- Cvičebnice a pracovní sešity (slouží k výuce a opakování dané látky formou praktického procvičování).
- Slovníky a přehledy gramatiky, u kterých lze předpokládat, že budou používány při výuce na ZŠ, SŠ či při studiu jazyka.
- Knihy pro předškoláky, které však mají přesah i do první třídy ZŠ.
- Počítačová literatura zjevně zaměřená na výuku na školách (Excel pro školy apod.).
- Atlasy, které jsou určeny k výuce ve školách či mohou sloužit jako její doplněk.
- Metodický průvodce a příručky pro učitele, které přímo patří k nějaké učebnici nebo řadě učebnic.

Za učebnice nebudeme považovat:

- Zjednodušená četba.
- Encyklopedie, populárně naučné atlasy (lidského těla, dějin, rostlin atd.).
- Slovníky pro určitý specializovaný obor, např. lékařský slovník, ekonomický slovník (slouží k výuce na VŠ, nikoliv obecně ke studiu daného jazyka).
- Knihy pro předškoláky, příprava na školu apod.
- Počítačová literatura, která není přímo zaměřená na školy.
- Jazykové konverzace, turistické průvodce.
- Metodické průvodce a příručky pro učitele, které existují samostatně a jsou nezávislé na konkrétní učebnici.

1.2 Klasifikace učebnic

Ve zbylých dvou úlohách budeme pracovat pouze s učebnicemi, tedy knihami, které splňují definici z předešlé sekce. Učebnice budeme klasifikovat jednak podle předmětu, do kterého spadá probírané téma, a také podle úrovně školského systému, pro nějž je učebnice určena.

Úloha 2

Zařadit učebnici do jedné z pevně daných tříd na základě školního předmětu, k jehož výuce je určena. (klasifikace do jedné z N tříd)

Budeme pracovat se seznamem předmětů, který je uveden níže. Vychází z předmětů běžně vyučovaných na školách s tím, že některé předměty, pro které by existovalo pouze malé množství učebnic, jsou sloučeny dohromady nebo přiřazeny k „běžnějšímu“ předmětu.

Co se týče jazyků, existují třídy jen pro ty nejběžnější; ostatní jazykové učebnice budou patřit do skupiny „Ostatní jazyky“.

Protože existují i velmi málo zastoupené předměty (např. cukrářství), které by se těžko slučovaly s jinými, zavedeme i speciální třídu „Ostatní“. Sem budou spadat také učebnice, které pokrývají vícero předmětů a nelze tedy rozhodnout, kam mají patřit. Typickým příkladem jsou přípravy k přijímacím zkouškám z českého jazyka a matematiky nebo cvičebnice znalostí z první třídy ZŠ, která obsahuje úlohy napříč předměty.

Seznam předmětů

Angličtina	Biologie a přírodopis	Český jazyk a literatura
Chemie	Dějepis	Ekonomie a účetnictví
Francouzština	Fyzika a geologie	Hudební výchova
Informatika	Italština	Matematika a geometrie
Medicína a ošetřovatelství	Němčina	Ostatní
Ostatní jazyky	Prvouka	Ruština
Španělština	Stavebnictví	Technika a strojírenství
Vlastivěda	Výtvarná výchova	Základy společenských věd
Zeměpis		

Úloha 3

Zařadit učebnici do jedné z pevně daných tříd na základě úrovně školského systému, pro kterou je primárně určena. (klasifikace do jedné z N tříd)

Úrovně budou opět pevně dané a jejich seznam je uveden níže. Základní školu budeme dělit na první a druhý stupeň, střední školy budou naopak zastoupeny jednou skupinou (ať už jde o gymnázia, odborné střední školy či učiliště). Speciálními skupinami pak budou učebnice pro autoškoly a jazykové učebnice, u kterých

nelze předpokládat, že by byly využívány ve školní výuce. Do posledně zmiňované skupiny budeme řadit také cizojazyčné slovníky a přehledy gramatik, které mohou používat žáci a studenti na kterékoliv úrovni.

U této úlohy nebudeme zavádět skupinu Ostatní, neboť dle naší definice učebnice by každá učebnice měla spadat do alespoň jedné z níže uvedených úrovní. V opačném případě by neplatilo, že se používá k výuce na základních školách, středních školách, učilištích, v autoškolách či v jazykových kurzech.

Seznam úrovní

První stupeň základní školy	Druhý stupeň základní školy
Střední školy a odborná učiliště	Jazykové školy, kurzy a samostudium
Autoškoly	

1.3 Data o knihách

Z veřejně dostupných zdrojů na internetu lze získat množství údajů, které různými způsoby charaketrizují danou knihu. Tyto údaje bychom mohli rozdělit do skupin:

- identifikační – název knihy, ISBN, číslo vydání
- o původu knihy – autor, nakladatel, rok vydání
- o obsahu knihy – anotace/popis, jazyk, kategorie, obsah (seznam kapitol)
- o fyzické podobě knihy – rozměry, hmotnost, počet stránek, vazba, obrázek obálky
- ...a mnoho dalších

Ne všechny údaje lze jednoduše dohledat pro všechny knihy a zároveň různé údaje mají různou informační hodnotu z pohledu úloh, které řešíme v této práci. Rozhodli jsme se proto omezit na několik málo nejdůležitějších údajů:

- ISBN/EAN
- Název knihy
- Popis knihy
- Autor
- Nakladatelství

1.3.1 ISBN/EAN

ISBN (z angl. International Standard Book Number) je jednoznačný číselný identifikátor knihy, který má pevně danou strukturu a lze z něj vyčíst i základní informace o původu knihy. Existují dva tvary ISBN, starší desetimístný (ISBN-10) a novější třináctimístný (ISBN-13), oba jsou na sebe ve většině případů navzájem jednoznačně převoditelné. My se budeme nadále zabývat pouze ISBN-13.

EAN je mezinárodní jednoznačný identifikátor obchodních položek, který se obvykle tiskne na výrobky v podobě čárového kódu. Pro nás je klíčové, že u knižních produktů odpovídá EAN číslu ISBN-13 a můžeme tedy tyto údaje používat zaměnitelně.

ISBN/EAN budeme využívat zejména k filtrování zdrojových dat stažených z internetu, kdy budeme chtít ponechat pouze záznamy, které mají přiřazeno ISBN. Skutečnost, zda daný EAN kód představuje ISBN, poznáme podle jeho prefixu – ISBN knihy totiž vždy začíná prefixem „978“. Uděláme však jednu výjimku, a sice pro prefix „979“, kterým začíná ISMN (z angl. Internation Standard Music Number), neboli mezinárodní standardní číslo hudebnin, používané pro tištěné hudebniny. Tímto číslem mohou být označeny učebnice hudební výchovy a tedy i produkty s ISMN lze považovat za knihy.

V neposlední řadě využijeme jednoznačnosti identifikátoru ISBN ke spojení záznamů o téže knize v různých zdrojích.

1.3.2 Název knihy

Název knihy bude klíčový údaj, který budeme ze zřejmých důvodů vyžadovat u všech záznamů – nepodaří-li se nám u nějakého produktu zjistit jeho název, nebudeme se jím vůbec zabývat.

V některých zdrojích se v názvu knihy objevuje také autor – tyto případy se budeme snažit eliminovat odstraněním autora z názvu. Ostatní možné údaje v názvu (pořadí vydání, forma knihy apod.) budeme ignorovat, neboť se jedná o menší počet případů a není pravděpodobné, že by tento šum v názvech ovlivnil výslednou úspěšnost klasifikačních úloh.

1.3.3 Popis knihy

Tento údaj bude informačně nejobsáhlejší a budeme za něj považovat jakýkoliv popis, anotaci či obsah knihy, který je dostupný z jednotlivých zdrojů dat. Popis nebudeme nijak filtrovat, pouze odstraníme případné formátovací (HTML) značky tak, abychom dostali čistý text.

1.3.4 Autor

Autor (případně autoři) knihy může být dobrým vodítkem při její klasifikaci, je k tomu ale nutná určitá znalost. Například skutečnost, že knihu napsal Jan Novák, není příliš užitečná, pokud takového spisovatele neznáme. Naopak u knihy, kterou napsal William Shakespeare, téměř každý správně odhadne, že se bude jednat pravděpodobně o divadelní hru, případně o poezii.

Klasifikační algoritmus musí uvedené znalosti získat z trénovacích dat. Vzhledem k tomu, že dat budeme mít omezené množství a autoři se příliš často nebudou

opakovat, neočekáváme, že by informace o autorovi byla natolik informativní, jako název či popis knihy. Přesto zkusíme v experimentech autora zahrnout a vyhodnotit jeho skutečný vliv.

1.3.5 Nakladatelství

Pro kompletnost budeme využívat jako jeden z parametrů knihy i nakladatele. Platí zde totéž, co je popsáno výše o autorovi, avšak v omezenější míře, protože nakladatelství mívají širší záběr vydávaných knih než jednotlivý autoři. Proto předpokládáme, že tento údaj bude mít ze všech nejmenší vliv na výslednou přesnost klasifikace.

1.3.6 Ostatní údaje

Ostatní údaje o knihách nebudeme uvažovat – většina z nich podle nás nemá pro klasifikaci žádnou informační hodnotu (např. počet stránek, vazba, rozměry knihy). Naopak užitečný by mohl být obrázek obalu knihy, a to navzdory známému přísloví „nesuď knihu podle obalu“. Existuje také několik prací, které se zabývají právě klasifikací knih na základě jejich obalu (viz část 2.5). V naší práci se ale zaměříme na textové údaje a klasifikace podle obrázku je tak nad její rámec a může být ideálním kandidátem na vylepšení úspěšnosti našich modelů.

2. Použité metody

V této kapitole popíšeme z teoretického pohledu všechny technologie a nástroje, které budeme používat. První část se věnuje metodám sběru dat, která budeme získávat z webových stránek e-shopů a online databází knih, následně spojování údajů o autorech a nakladatelích z různých zdrojů a také vytváření reprezentací slov číselnými vektory (tzv. word embedding). Další části popisují metody strojového učení, které budeme používat v klasifikačních úlohách. Jako baseline využijeme naivní bayesovský klasifikátor, následně popíšeme neuronové sítě a také prostředky pro vyhodnocování a porovnávání modelů strojového učení. V poslední části uvedeme existující práce zabývající se podobnými tématy jako tato práce, tedy automatickou klasifikací knih.

2.1 Sběr dat

2.1.1 Web scraping

Web scraping je technika získávání nestrukturovaných informací z webových stránek. Používá se tam, kde není možné data stáhnout ve strojově čitelném formátu a není k dispozici žádné API. Ačkoliv by se dalo za web scraping označit i ruční kopírování textů z webu prostřednictvím prohlížeče, většinou jde spíše o automatický proces, kdy jsou shromážděna určitá data z webu do lokální databáze, tabulky či textového souboru pro jejich pozdější analýzu.

Získávání probíhá obvykle ve dvou fázích:

- Stažení webové stránky ve formě HTML. K tomu, abychom věděli, jaké stránky stahovat, se používá technika web crawling, neboli procházení webu skrze hypertextové odkazy.
- Extrakce dat z HTML. Máme-li stránku staženou, můžeme z ní extrahovat data, např. pomocí vyhledávání nebo regulárních výrazů.

2.1.2 Deduplikace

Deduplikace (anglicky deduplication, record linkage, data matching a mnoho dalších označení) je proces, který má za cíl najít záznamy reprezentující stejný objekt v různých zdrojích dat, případně duplicitní záznamy pro tentýž objekt v rámci jednoho zdroje. Formální základy metody zavedli Fellegi a Sunter (1969) a od té doby vzniklo na toto téma mnoho prací.

Obecně lze celý proces rozdělit na následující kroky:

1. Získávání a čištění dat – deduplikace obvykle vyžaduje kvalitní data, aby bylo možné párovat jednotlivé atributy.
2. Vytvoření potenciálních odpovídajících si dvojic – aby nebylo nutné procházet celý kartézský součin všech možných dvojic záznamů, používají se různé heuristiky k omezení tohoto prostoru. Například blocking – rozdělení záznamů do skupin na základě hodnoty jednoho či více atributů a následné porovnávání dvojic pouze v rámci skupiny.

3. Porovnání záznamů v každém z potenciálních párů – pro každý pár se vytvoří množina charakteristik, na základě kterých se následně rozhoduje, zda oba záznamy z páru reprezentují tentýž objekt. Obvykle jde o metriku vzdálenosti či podobnosti řetězců (viz 2.1.3).

4. Podle charakteristik z minulého kroku se rozhoduje, zda jde o záznamy pro tentýž objekt či nikoliv. K tomu lze využít postup založený na pravidlech (rule-based), kdy jsou určené různé prahové hodnoty, nebo lze pro rozhodování natrénovat binární klasifikátor.

Speciálním typem deduplikace je deduplikace bibliografických dat. Tou se zabývají například Jiang a kol. (2014), kteří představují algoritmus založený na pravidlech, nebo Borges a kol. (2011), kde jsou porovnávány různé natrénované klasifikátory.

2.1.3 Porovnávání řetězců

V předešlém odstavci jsme popsali proces deduplikace dat, kde v jednom kroku je nezbytné porovnat dva záznamy. Jsou-li záznamy v textové podobě, chceme pro takovou dvojici najít metriku, která nám řekne, jak moc jsou si dva textové řetězce navzájem podobné. Podle toho se pak rozhodneme, jestli reprezentují stejné jméno, název firmy, adresu apod. Existuje množství různých přístupů a různých algoritmů pro řešení tohoto problému:

Editační vzdálenost řetězců Udává počet jednoduchých operací, které je potřeba provést pro jeden řetězec, abychom dostali řetězec druhý. Obvyklé operace jsou následující:

- Vložení znaku – „aut“ -> „auto“
- Smazání znaku – „auto“ -> „aut“
- Substituce – „auto“ -> „auta“
- Transpozice – „auto“ -> „auot“

Jednotlivé algoritmy pro editační vzdálenost se liší v operacích, které povolují:

- Hammingova vzdálenost – umožňuje pouze substituci, lze proto použít jen u řetězců stejné délky a určuje počet pozic, na kterých se řetězce liší.
- Levenštejnova vzdálenost – povoluje vložení, smazání a substituci. Využívá se často v algoritmech pro opravu překlepů.
- Damerau-Levenštejnova vzdálenost – oproti Levenštejnově vzdálenosti povoluje také transpozici dvou sousedních znaků.

Podobnost řetězců. Jde o algoritmy, které pro dvojici řetězců určí, jak moc jsou si podobné, obvykle jako reálné číslo mezi 0 (žádná podobnost) a 1 (shodné řetězce). Příkladem těchto algoritmů je Jaro-Winklerův algoritmus, Gestalt Pattern Matching nebo procentuální vyjádření nejdelšího společného podřetězce.

Nejdelší společný podřetězec Nejdelší společný podřetězec (budeme jej značit NSP) dvou řetězců s_1 a s_2 je nejdelší řetězec s, který je podřetězcem s_1 i s_2. V této práci budeme využívat metriku, která dává délku NSP do poměru s délkou kratšího z řetězců s_1 a s_2. Dva řetězce tak mohou být „podobné“ i v případě, že jeden z nich je výrazně delší, avšak nějaká jeho část se velmi podobá kratšímu řetězci. Výpočet metriky bude následující:

$$D_{NSP} = \frac{L_{NSP}(s_1,s_2)}{min(|s_1|, |s_2|)} \tag{2.1}$$

kde L_{NSP} je délka nejdelšího společného podřetězce a $|s_i|$ je délka řetězce s_i.

Jaro-Winklerův algoritmus Jedná se o vylepšenou verzi Jarova algoritmu. Jarův algoritmus přiřazuje dvěma řetězcům s_1 a s_2 podobnost podle následujícího výpočtu:

$$D_j = \begin{cases} 0, & \text{pro } m = 0, \\ \frac{1}{3}(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}) & \text{pro } m \neq 0, \end{cases} \tag{2.2}$$

kde $|s_i|$ je délka řetězce s_i, m je počet odpovídajících si znaků a t je počet transpozic.

Dva znaky jsou si odpovídající, pokud se jedná o stejný znak a jejich vzdálenost je nejvýše $\left\lfloor \frac{max(|s_1|,|s_2|)}{2} \right\rfloor - 1$.

Počet transpozic vypočítáme jako polovinu počtu znaků, které si odpovídají, ale nejsou ve stejném pořadí.

Rozšířením tohoto algoritmu je Jaro-Winklerův algoritmus, který navíc přidává „bonus“ za délku společného prefixu obou řetězců. Přesný vzorec je tento:

$$D_{jw} = D_j + pL(1 - D_j), \tag{2.3}$$

kde:

- D_j je Jarova metrika,
- p je konstantní koeficient (mezi 0 a 0,25, obvykle se volí 0,1),
- L je délka společného prefixu (může být nejvýše 4, i když je prefix delší).

Gestalt Pattern Matching Tento algoritmus počítá podobnost řetězců s_1 a s_2 na základě rekurzivního hledání nejdelších společných podřetězců. Konkrétně podobnost je dána vztahem

$$D_g = \frac{2K_m}{|s_1| + |s_2|} \tag{2.4}$$

kde $|s_i|$ je délka řetězce s_i a K_m se spočítá jako délka nejdelšího společného podřetězce (NSP) plus rekurzivně délka NSP pro levé a pravé „zbytky“ řetězců po odebrání NSP.

Fonetické algoritmy Kódují řetězce takovým způsobem, aby ty, které se vyslovují stejně, měly stejný kód, i když jsou zapsány mírně odlišně. Příkladem takových algoritmů jsou SoundEx nebo NYSIIS.

2.1.4 Word embedding

Algoritmy hlubokého učení se mimo jiné používají i na úlohy zpracování přirozeného jazyka, které mají na vstupu textová data. Surový text však není možné přímo vložit do hluboké sítě, neboť ta umí zpracovávat pouze numerická data. Je proto nutný proces nazývaný vektorizace textu. Text na numerická data lze převést několika způsoby, podle toho, jak text chápeme:

- Text jako sekvence znaků – každému znaku přiřadíme numerický vektor.
- Text jako sekvence slov – každé slovo pak má svůj vektor.
- Text jako sekvence n-gramů znaků či slov – vektor pro každý n-gram.

Nejčastěji se text chápe jako sekvence slov. Nejjednodušší metodou je pak one-hot kódování, kdy si vytvoříme slovník všech slov z textu očíslovaný 1 až N a slovu přiřadíme tzv. one-hot vektor, který obsahuje samé nuly, kromě pozice odpovídající číslu slova, kde bude jednička. Tento způsob má několik nevýhod, zejména zpravidla velmi dlouhé vektory slov a to, že nejsou žádným způsobem zachyceny sémantické vztahy mezi slovy.

Tyto problémy řeší tzv. word embedding. Jde o koncept, kdy je každé slovo reprezentováno hustým číselným vektorem s dimenzí obvykle v řádu nižších stovek. Zároveň by mělo platit, že vektory podobných slov jsou si také podobné. Pro word embeddingy se obvykle používají neuronové sítě trénované na velkých korpusech o celkovém počtu slov až v řádech miliard (např. texty z Google News, články z Wikipedie). Nejznámějšími algoritmy pro word embedding jsou word2vec (Mikolov a kol., 2013) či GloVe (Pennington a kol., 2014).

Pro algoritmus word2vec existují dvě základní architektury, jak embeddingy trénovat:

- CBOW (Continous Bag of Words)
- Skip-gram

CBOW Tato architektura je založena na principu, že máme v textu neznámé slovo, které se snažíme „uhodnout“ na základě okolních slov. Řekněme, že máme neuronovou síť (viz kapitola 2.3), která na vstupu dostane dvě slova před neznámým slovem a dvě slova za ním (jednotlivá slova jsou reprezentována jako one-hot vektor) a jejím výstupem bude neznámé slovo (opět reprezentováno jako one-hot vektor). Trénovacími daty této sítě budou všechny takovéto vzorky 5 slov z daného korpusu. Ve výsledku nás nebude příliš zajímat úspěšnost natrénované sítě, ale váhy její skryté vrstvy, které budou právě požadované word embeddingy.

Skip-gram U architektury Skip-gram je princip obrácený a naopak známe slovo v textu, ke kterému se snažíme odhadnout jeho kontext (okolní slova). Opět embeddingy získáme jako váhy skryté vrstvy neuronové sítě, v tomto případě však bude mít síť na vstupu jen jedno slovo a na výstupu slov několik.

V obou architekturách platí, že počet uvažovaných okolních slov je hyperparametrem.

2.2 Naivní Bayesovský klasifikátor

Naivní Bayesovský klasifikátor je pravděpodobnostní klasifikátor založený na Bayesově větě a na (naivním) předpokladu, že jednotlivé vstupní parametry jsou na sobě navzájem zcela nezávislé. Bayesova věta říká následující:

Věta 1. *Máme dva náhodné jevy A a B, které mají pravděpodobnost $P(A)$ a $P(B)$, kde $P(B) > 0$. Potom platí:*

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

kde $P(A|B)$ je pravděpodobnost jevu A za předpokladu, že nastal jev B a $P(B|A)$ je pravděpodobnost jevu B za předpokladu, že nastal jev A.

Předpoklad nezávislosti vstupních parametrů není většinou realistický, přesto takový klasifikátor obvykle dává dobré výsledky. V naší práci budeme používat naivní bayesovký klasifikátor mimo jiné pro klasifikaci textových vstupů (název a popis knihy).

Uvažujme například název knihy „Němčina pro samouky“, pro který chceme určit, zda se jedná o učebnici. Vypočítáme pravděpodobnost pro obě možnosti, tedy že kniha je učebnice (tento jev označíme U), resp. že není učebnice (tento jev označíme N). Jinými slovy, hledáme podmíněnou pravděpodobnost, že kniha je učebnice, pokud víme, že nese název „Němčina pro samouky“. Podle Bayesovi věty bude platit:

$$P(U|\text{Němčina pro samouky}) = \frac{P(\text{Němčina pro samouky}|U)P(U)}{P(\text{Němčina pro samouky})} \tag{2.5}$$

$$P(N|\text{Němčina pro samouky}) = \frac{P(\text{Němčina pro samouky}|N)P(N)}{P(\text{Němčina pro samouky})} \tag{2.6}$$

Jmenovatel obou zlomků je stejný, můžeme jej proto zanedbat a porovnávat pouze následující pravděpdobnosti:

$$P(\text{Němčina pro samouky}|U)P(U) \tag{2.7}$$

$$P(\text{Němčina pro samouky}|N)P(N) \tag{2.8}$$

$P(U)$ a $P(N)$ snadno určíme podle poměru učebnic a ne-učebnic v trénovacích datech. Podmíněná pravděpodobnost $P(\text{Němčina pro samouky}|U)$, tedy jaká je pravděpodobnost právě tohoto názvu, je-li kniha učebnicí, by teoreticky měla být rovna poměru počtu učebnic s tímto názvem ku celkovému počtu učebnic. To však ve většině případů bude nula, protože názvy knih se až na výjimky neopakují. Využijeme proto naivní předpoklad nezávislosti jednotlivých slov v názvu knihy a rozepíšeme pravděpodobnost následovně:

$$P(\text{Němčina pro samouky}|U) = P(\text{Němčina}|U) \times P(\text{pro}|U) \times P(\text{samouky}|U) \tag{2.9}$$

kde $P(\text{Němčina}|U)$ bude vyjadřovat pravděpodobnost, že se v názvu učebnice objeví slovo Němčina. Jinak řečeno, jde o poměr výskytu slova Němčina ku celkovému počtu všech slov v názvech knih.

S tímto výpočtem však nastane problém v případě, kdy se v názvu knihy objeví slovo, které se v trénovacích datech nevyskytuje v žádném názvu učebnice. Dílčí pravděpodobnost pak vyjde 0 a nulová tak bude i celková pravděpodobnost $P(\text{Němčina pro samouky}|U)$. Řešením je metoda vyhlazování, která se nazývá additive smoothing nebo také Laplace smoothing. Máme-li slovo X a třídu klasifikace C (např. v našem případě U nebo N), pak bychom pravděpodobnost měli počítat následovně:

$$P(X|C) = \frac{N_{X,C}}{N_C} \tag{2.10}$$

kde $N_{X,C}$ je počet výskytů slova X ve třídě C a N_C je celkový počet slov ve třídě C. Použijeme-li vyhlazování, výpočet se změní na

$$P(X|C) = \frac{N_{X,C} + A}{N_C + Ad} \tag{2.11}$$

kde A je parametr vyhlazování (pro Laplace smooting $A = 1$) a d je počet slov v názvu knihy.

Nakonec porovnáme podmíněné pravděpodobnosti pro učebnici a ne-učebnici a knihu klasifikujeme do třídy s vyšší pravděpodobností.

2.3 Dopředné neuronové sítě

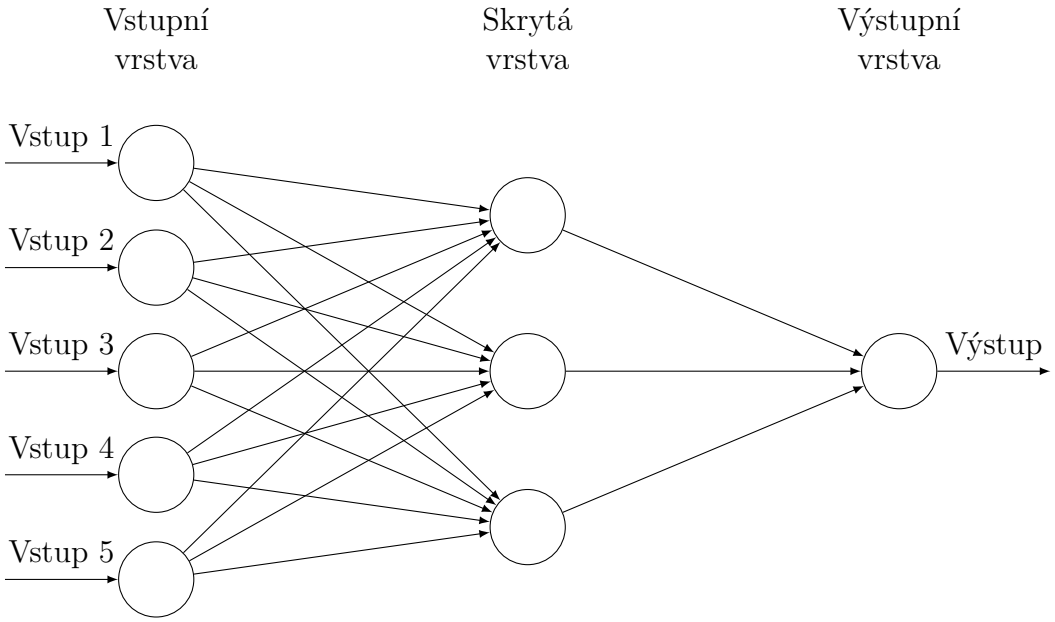
Dopředné neuronové sítě (FNN, z angl. Feedforward Neural Network) jsou jedním z modelů strojového učení, který se skládá z několika vrstev (vstupní, výstupní a případných skrytých vrstev), přičemž výstup jedné vrstvy se použije jako vstup vrstvy následující. Na počátku do sítě vstupují data, která postupně prochází vrstvami a transformují se na výsledek (predikci), který je výstupem sítě. Schéma dopředné neuronové sítě je na obrázku 2.1.

Základním stavebním prvkem FNN je neuron znázorněný na obr. 2.2. Může mít libovolný počet N vstupů a jeden výstup, který je dán aktivační funkcí (viz kapitola 2.3.3), souborem N vah pro jednotlivé vstupy a prahovou hodnotou (tzv. biasem). Matematicky je výstup neuronu určen následujícím vztahem:

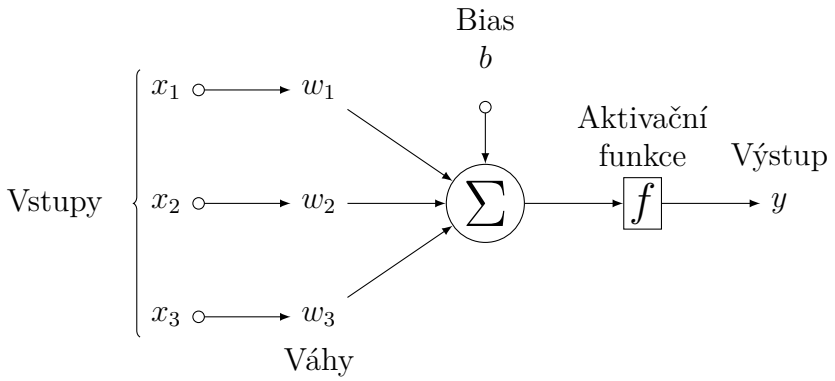
$$Y = f(\sum_{i=1}^{N}(w_i x_i) + b) \tag{2.12}$$

kde:

- x_i jsou vstupy neuronu,
- w_i jsou váhy neuronu,
- b je prahová hodnota (bias),
- $f(x)$ je aktivační funkce,
- Y je výstup neuronu.



Obrázek 2.1: Dopředná neuronová síť



Obrázek 2.2: Schéma neuronu

2.3.1 Učení

Cílem učení neuronové sítě je nastavit váhy v síti tak, aby síť dávala co nejlepší výsledky. Existují dva základní typy učení, a sice s učitelem a bez učitele. Při učení bez učitele se síť učí na základě vstupních dat klasifikovat vzorky do několika tříd. V této práci se budeme dále zabývat pouze učením s učitelem, kdy učení probíhá na základě správných výstupů, které jsou síti při procesu učení předkládány.

V dopředných neuronových sítí probíhá učení zpravidla metodou zpětného šíření (backpropagation) – na počátku se inicializují váhy sítě náhodně, následně se vezmou trénovací data a vypočítá se výsledek. Ten se porovná se správnými (očekávanými) výsledky a chyba se propaguje zpět sítí, přičemž se upravují váhy sítě v jednotlivých vrstvách. Tento postup se opakuje, dokud se síť „nenaučí“ vhodné váhy tak, aby chyba byla co nejmenší. K výpočtu nových vah se obvykle používá tzv. stochastický gradientní sestup (SGD, z angl. Stochastic Gradient

Descent), který se pomocí výpočtu gradientu snaží hledat minimum funkce, která počítá chybu (ztrátová funkce, viz kapitola 2.3.3).

2.3.2 Typy vrstev

V dopředných neuronových sítích se můžeme setkat s velkým množstvím různých vrstev (ty, které využíváme v této práci, popisujeme níže). Podle toho, které vrstvy tvoří strukturu sítě, se pak dopředné neuronové sítě dělí do několika kategorií, například hustě propojené sítě, konvoluční sítě či rekurentní sítě.

Hustě propojená vrstva (Dense) Vrstva Dense sestává z N neuronů, přičemž N je hyperparametrem. Každý neuron je na vstupu propojen s každým z výstupů předchozí vrstvy a má jeden výstup.

Pomocí maticového zápisu lze vyjádřit výstup hustě propojené vrstvy takto:

$$Y = f(W^T X + b) \tag{2.13}$$

kde:

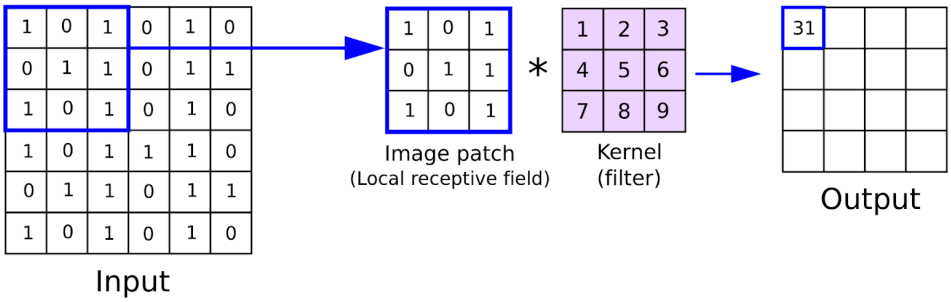
- f je aktivační funkce,
- W je matice vah vrstvy,
- X je vstup vrstvy,
- Y značí výstup,
- b je vektor biasů.

Embedding vrstva Jde o vrstvu z frameworku Keras, která dostane na vstupu sekvenci N celých čísel (reprezentujících např. slova) a na výstupu vrátí matici o velikosti $N \times D$, reprezentující vstupní sekvenci zakódovanou do embedding vektorů (D je dimenze těchto vektorů). Vektory pro jednotlivá slova mohou být předem dána a během tréninku se neměnit, nebo naopak mohou být trénovány společně se zbytkem sítě.

Regularizační Dropout vrstva Dropout vrstva má za úkol zabránit přeučení sítě. Během tréninku proto náhodně vynulovává některé ze vstupů, přičemž procento vstupů, které vynuluje, je dáno hyperparametrem r (r je číslo mezi 0 a 1). Zbylé, nevynulované vstupy pak vynásobí číslem $\frac{1}{1-r}$, aby součet všech vstupů zůstal nezměněný. Při testování natrénovaného modelu již Dropout vrstva nemá žádný efekt.

Konvoluční vrstva Konvoluční vrstva je základním stavebním prvkem konvolučních neuronových sítí. Používá se většinou pro extrakci příznaků z obrázků, tedy dvojrozměrných dat, ale lze ji aplikovat i na jednorozměrná data (text) nebo trojrozměrná data (video). Základem konvoluční vrstvy je N filtrů o velikosti K, přičemž N i K jsou hyperparametry a K je menší než velikost vstupu. Tyto filtry

jsou postupně aplikovány na každou možnou pozici na vstupu, je vypočítán skalární součin filtru a odpovídající části vstupních hodnot a všechny tyto výsledky pak tvoří tzv. mapu příznaků. Výpočet této mapy je znázorněn na obrázku 2.3. Na mapu příznaků může být ještě aplikována aktivační funkce a přičten bias vektor. Konečný výsledek je vrácen jako výstup konvoluční vrstvy, přičemž obvykle je filtrů více ($N > 1$) a výstup má tak o jednu dimenzi více než mapa příznaků.



Obrázek 2.3: Princip fungování konvoluční vrstvy
Zdroj: https://anhreynolds.com/blogs/cnn.html

Sdružovací vrstva (Pooling) Tato vrstva slouží k redukci map příznaků, které jsou výstupem konvolučních vrstev. Jejími hyperparametry jsou velikost okna K a velikost posuvu P. Princip vrstvy je, že postupně posouvá okno velikosti K po vstupu a na výstup zapíše maximální (MaxPooling) nebo průměrnou (AvgPooling) hodnotu z části vstupu pod oknem. Čím větší je posuv P, tím více se redukuje rozměr vstupních map příznaků. Běžně se používá i $P = K$, kdy jsou výstupem maxima/průměry navzájem se nepřekrývajících částí vstupu.

Rekurentní vrstva Rekurentní vrstva, která je základním prvkem rekurentních neuronových sítí (RNN), byla navržena pro zpracovávání sekvencí a narozdíl od jiných vrstev nevyžaduje všechny vstupy stejné délky. Navíc si uchovává skrytý stav a díky tomu výstup buňky nezávisí pouze na aktuálním vstupu, ale také na vstupech předchozích.

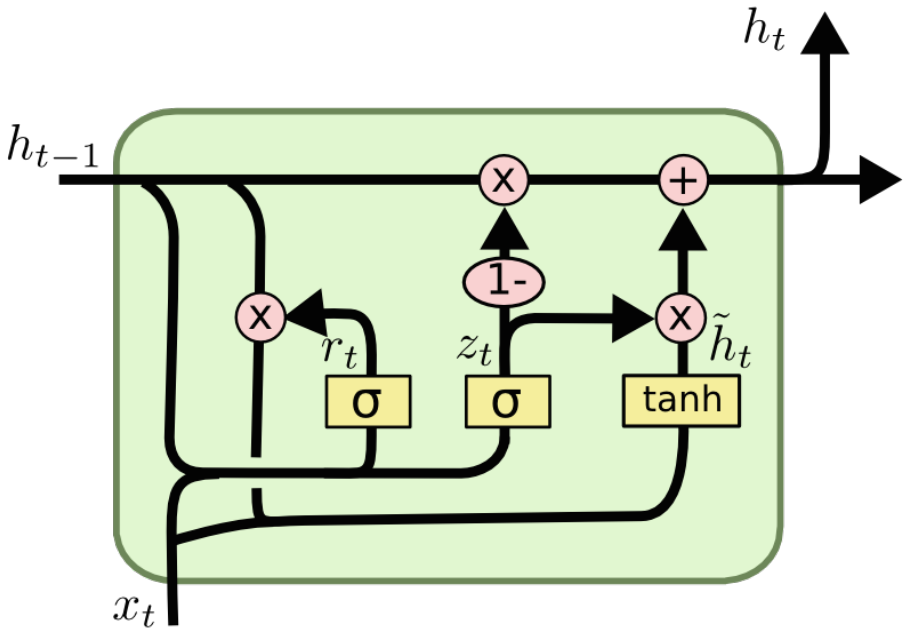
Základní verze RNN trpí tzn. problémem mizejícího gradientu, což je efekt, kdy síť není schopná naučit se časově vzdálené závislosti. Tento problém studovali Bengio a kol. (1994). Řešení přineslo rozšíření konceptu rekurentní vrstvy zvané LSTM (z angl. Long Short-Term Memory, Hochreiter a Schmidhuber (1997)), které zavádí vedle hlavního zpracování sekvence i „přenosovou cestu“, která pomáhá přenášet informace přes delší časové úseky. Později byla navržena alternativa k LSTM zvaná GRU (z angl. Gated Recurrent Unit, Chung a kol. (2014)), která funguje na stejném principu jako LSTM, ale je výpočetně jednodušší při zachování téměř stejné síly.

Schéma GRU buňky je na obrázku 2.4. Matematicky ji lze charakterizovat těmito vzorci:

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{2.14}$$
$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{2.15}$$
$$\tilde{h_t} = tanh(W x_t + U(r_t \cdot h_{t-1})) \tag{2.16}$$
$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h_t} \tag{2.17}$$

kde:

- r_t, z_t a $\tilde{h_t}$ jsou interní stavy buňky,
- U, U_r, U_z, W, W_r, W_z jsou matice vah vrstvy,
- x_t je vstup buňky,
- h_t je výstup buňky,
- σ a $tanh$ jsou aktivační funkce (viz 2.3.3), které jsou hyperparametry rekurentní vrstvy.



Obrázek 2.4: Schéma GRU buňky
Zdroj: Riaz a kol. (2020)

Vrstvy Concatenate a Flatten Tyto dvě vrstvy z frameworku Keras slouží k transformaci hodnot uvnitř modelu a nemají žádné trénovatelné parametry. Vrstva Concatenate vezme výstup z několika vrstev a zřetězí jej do jediného výstupu, vrstva Flatten pak vezme vícerozměrný vstupní vektor a převede jej na jednorozměrný vektor.

2.3.3 Hyperparametry

Hyperparametry jsou parametry sítě nebo trénovacího procesu, které se během učení nemění a musí být předem nastaveny uživatelem. Mají zásadní vliv na výslednou efektivitu i rychlost učení a dalo by se pod ně zahrnout vše od struktury sítě (počet, velikost, typ vrstev, jejich propojení atd.) až po parametry trénovacího procesu (počet epoch, optimalizátor).

Neexistuje žádné univerzální optimální nastavení hyperparametrů, neboť pro každou úlohu může být odlišné. Optimální hyperparametry lze hledat buď ručně (trénováním modelů s různým nastavením), nebo některou z automatizovaných technik. Mezi ty nejpoužívanější patří GridSearch, kdy se určí pro každý hyperparametr množina hodnot a natrénuje se model pro každou možnou kombinaci hyperparametrů, případně RandomSearch, kdy se trénují a porovnávají modely s náhodně zvoleným nastavením hyperparametrů.

Základní hyperparametry, společné nejrůznějším architekturám neuronových sítí, zahrnují následující:

Inicializátor a aktivační funkce Aktivační funkce určuje, jakým způsobem bude transformován výstup dané vrstvy. Mezi nejčastěji používané aktivační funkce, jejichž grafy znázorňuje obrázek 2.5, patří:

- Identita

$$f(x) = x \tag{2.18}$$

- ReLU (Rectified Linear Activation)

$$f(x) = \begin{cases} 0, & \text{pro } x <= 0 \\ x & \text{jinak} \end{cases} \tag{2.19}$$

- Sigmoidní funkce

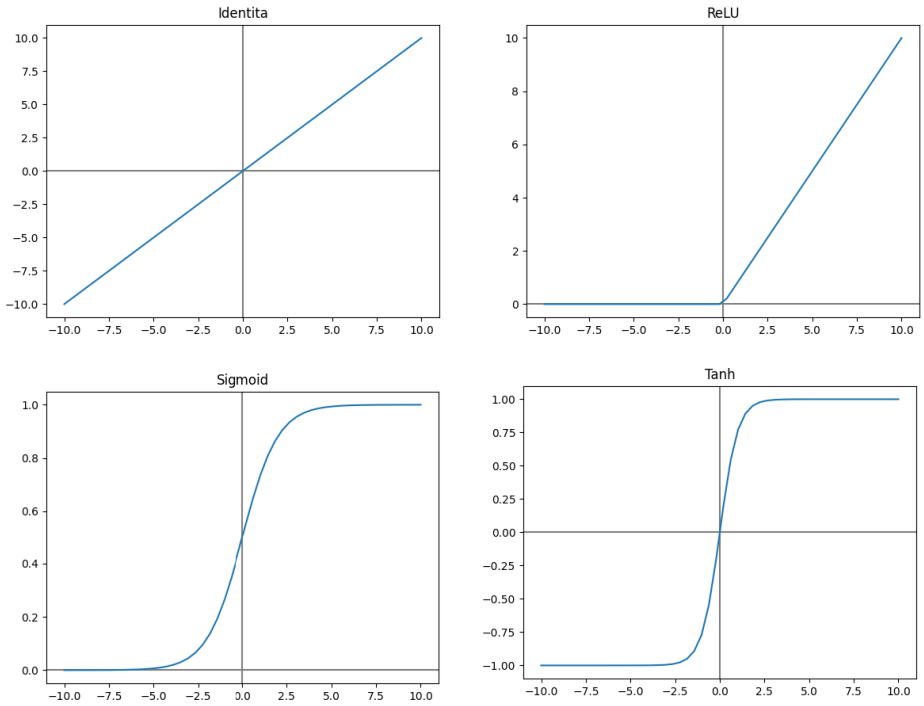
$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.20}$$

- Hyperbolický tangens

$$tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{2.21}$$

Inicializátor definuje počáteční váhy vrstvy. Dříve se na počátku učení vždy váhy nastavovaly na malá náhodná čísla, ale v poslední době bylo vyvinuto několik heuristik, které mají za cíl nastavení optimálnějších počátečních vah, obvykle v závislosti na typu aktivační funkce či velikosti vstupu. Nejvíce používané inicializátory jsou:

- Xavier (někdy též označován Glorot) – tento inicializátor byl popsán v práci Glorota a Bengia (Glorot a Bengio, 2010) a používá se v kombinaci se sigmoidní aktivační funkcí nebo s funkcí tanh. Počáteční váhy mají rovnoměrné rozdělení a jsou z intervalu $[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}]$, kde n je počet vstupů neuronu.
- He – inicializátor, který se používá především s aktivační funkcí ReLU, popsali ve své práci He a kol. (2015). Počáteční váhy mají normální (Gaussovo) rozdělení se střední hodnotou 0 a směrodatnou odchylkou $\sqrt{\frac{2}{n}}$.



Obrázek 2.5: Grafy aktivačních funkcí

Ztrátová funkce Proces trénování neuronové sítě se snaží najít takové váhy modelu, aby model mapoval vstupní hodnoty co nejpřesněji na očekávané výstupní hodnoty. K tomu využívá tzv. ztrátovou funkci, která vyjadřuje jedním číslem chybu modelu, neboli jak moc se hodnoty predikované modelem liší od očekávaných hodnot. Cílem trénování je tedy minimalizovat tuto funkci. Výběr ztrátové funkce závisí na typu úlohy, nejpoužívanější funkce jsou následující:

- Střední kvadratická chyba (Mean Squared Error, MSE) – pro úlohy logistické regrese, kde výstupem modelu je reálné číslo. Spočítá se jako průměr druhých mocnin rozdílů mezi predikcí a očekávaným výstupem:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.22}$$

- Binární křížová entropie – pro úlohy binární klasifikace, výstupem modelu je číslo mezi 0 a 1, udávající pravděpodobnost, že vzorek má danou vlastnost. Křížová entropie se spočítá podle vzorce 2.23, kde n je počet vzorků, y_i očekávaný výstup a $\hat{y}_i$ predikce vrácená modelem.

$$E = -\frac{1}{n}\sum_{i=1}^{n}[y_i log\hat{y}_i + (1 - y_i)log(1 - \hat{y}_i)] \tag{2.23}$$

- Kategorická křížová entropie – pro úlohy klasifikace do jedné z M tříd, výstupem modelu jsou pravděpodobnosti, že vzorek patří do té které třídy (tedy M čísel, jejichž součet je 1). Křížová entropie se pak spočítá podle vzorce 2.24, kde n je počet vzorků, m je počet tříd, $y_{i,j}$ je očekávaný výstup $i$tého vzorku pro třídu j a $\hat{y_{i,j}}$ je predikce vrácená modelem značící

pravděpodobnost, že $i$tý vzorek patří do třídy j.

$$E = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}[y_{i,j}log\hat{y_{i,j}} + (1 - y_{i,j})log(1 - \hat{y_{i,j}})] \tag{2.24}$$

Optimalizátor Určuje konkrétní implementaci gradientního sestupu pro aktualizaci vah sítě. Existuje množství různých algoritmů, které řeší některé z problémů základního stochastického gradientního sestupu, např. AdaGrad, AdaDelta či RMSProp. Zlatým standardem mezi optimalizátory je ale Adam, který kombinuje výhody vícera uvedených optimalizátorů a navíc je poměrně nenáročný co se časových a paměťových nároků týče.

Velikost dávky Udává počet vzorků, které trénovací algoritmus použije, než aktualizuje (zpětnou propagací) váhy v modelu. Podle hodnoty tohoto parametru se rozlišují tři varianty gradientního sestupu (SGD):

- Stochastický (on-line) SGD – velikost dávky = 1 – jde o „skutečný“ stochastický gradientní sestup, kdy se váhy aktualizují po každém trénovacím vzorku.
- Dávkový (batch) SGD – velikost dávky = velikost trénovací množiny – aktualizace vah na základě všech trénovacích vzorků je přesnější, ale zpětné šíření pak bývá výpočetně příliš náročné.
- Minidávkový (mini-batch) SGD – velikost dávky větší než 1 a menší než velikost trénovací množiny – kompromis mezi dvěma extrémními přístupy.

Počet epoch Jedna epocha znamená jeden průchod trénovacího algoritmu všemi vzorky trénovací množiny. Optimální počet epoch je silně závislý na úloze – může to být jedna epocha, ale i několik set nebo tisíc epoch. Pokud je počet epoch příliš malý, váhy modelu se dostatečně nenaučí vzory v trénovacích datech a dochází k podučení (underfitting). Naopak pokud nastavíme moc vysoký počet epoch, váhy se přizpůsobí až příliš trénovacím datům a na nových, testovacích datech má pak model úspěšnost daleko nižší. Tomuto jevu se říká přeučení (overfitting) a takový model má pak vysokou tzv. generalizační chybu (generalization error).

2.4 Vyhodnocování modelů

2.4.1 Křížová validace

Pomocí křížové validace budeme odhadovat, jak se bude daný model strojového učení chovat na neznámých (testovacích) datech. Principem je rozdělení trénovací množiny dat na několik podmnožin, přičemž jedna podmnožina se určí jako validační a zbylé podmnožiny jako trénovací. Model je natrénován na trénovací části dat a jeho úspěšnost ověřena na validační podmnožině.

Často využívaným typem křížové validace je $k$násobná křížová validace (k-fold cross-validation), kdy se trénovací množina dat rozdělí na k stejně velkých

podmnožin a natrénuje se k různých modelů, přičemž každá podmnožina slouží právě jednou jako validační.

Variantou $k$násobné křížové validace, jenž budeme využívat v této práci, je stratifikovaná křížová validace, kdy se trénovací množina rozdělí na podmnožiny tak, aby v každé podmnožině byl (přibližně) stejný počet zástupců jednotlivých tříd. To zaručí, že jednotlivé validační podmnožiny budou více reprezentativní.

2.4.2 Párový t-test

Pro vzájemné porovnávání různých modelů a zjišťování, jestli jsou rozdíly mezi nimi statisticky signifikantní, budeme používat párový t-test. Jde o metodu matematické statistiky, která umožňuje ověřit hypotézu, že rozdíl středních hodnot veličiny X a veličiny Y je roven určitému číslu (zpravidla nule). Tato hypotéza se označuje jako nulová.

Mějme dva náhodné výběry hodnot $(x_1, x_2,...,x_n)$ a $(y_1,y_2,...,y_n)$, které mohou reprezentovat např. úspěšnost dvou modelů na testovacích množinách dat 1 až n. Označme rozdíly v jejich úspěšnostech $(d_1, d_2,...,d_n)$. Budeme testovat nulovou hypotézu, že tyto rozdíly pocházejí z normálního rozdělení se střední hodnotou 0, což by znamenalo, že rozdíl mezi modely není statisticky signifikantní. Pokud nulovou hypotézu zamítneme, můžeme naopak učinit závěr, že jeden model je signifikantně lepší než druhý.

2.5 Existující práce

Není nám známo, že by existovala práce zabývající se přímo rozpoznáváním či klasifikací učebnic. Existuje však množství prací zaměřených na klasifikaci knih do různých žánrů, za použití krátkých textových dat či obrázků obalů knih. Sobkowicz a kol. (2018) klasifikují pomocí strojového učení krátké popisy získané z portálu GoodReads a zařazují je do jedné či více z 506 kategorií. Kundu a Zheng (2020) se zabývají použitím hlubokých sítí pro klasifikaci knih do žánrů, na základě obalu knihy. Předem však z obalu extrahují textové informace, které obvykle reprezentují název a autora knihy. Obrázkové a textové informace trénují zvlášť a následně oba klasifikátory spojují. Dosahují přesnosti 56,1 % (nejpravděpodobnější žánr správně), resp. 77,7 % (správný žánr mezi 3 nejpravděpodobnějšími). Zhu a kol. (2015) pak používá strojové učení – SVM a konvoluční neuronové sítě – k rozpoznávání konkrétních knih na fotografiích.

Vzhledem k tomu, že název a popis knihy jsou textové údaje a provádíme klasifikaci do N tříd, můžeme obecněji naše úlohy považovat za klasifikaci textů. V principu se tak neliší od úloh klasifikace dokumentů či zpráv do kategorií, rozpoznávání kladných a záporných recenzí apod. Přehled modelů pro různé úlohy klasifikace textů založených na hlubokém učení lze nalézt v práci Minaee a kol. (2021).

3. Data

Tato kapitola popisuje kompletní proces získávání a přípravy dat používaných ve všech úlohách. Data budou pocházet z veřejně přístupných webových stránek, jako jsou online databáze knih či detaily produktů internetových knihkupectví.

3.1 Získávání dat

Prvním krokem bude výběr vhodných zdrojů dat, sběr dat a jejich extrakce do strukturované podoby. Na konci tohoto kroku budeme mít data ze všech zdrojů uložená ve vhodné reprezentaci, abychom s nimi mohli dále pracovat. Data zatím nebudou téměř nijak očištěna, záznamy o stejných knihách z různých zdrojů nebudou nijak pospojované atd. Bude se tedy jednat o strukturovaná surová data.

3.1.1 Zdroje

Výběr správných zdrojů dat je důležitou součástí celého procesu a může mít velký vliv na konečnou úspěšnost klasifikačních úloh. Naším cílem při výběru zdrojů jsou dva základní požadavky na data:

1. Data musí být dostatečně velká. Pro neuronové sítě a speciálně pro hluboké učení je velikost dat klíčová. Přestože nakonec z důvodu ruční anotace využijeme pro samotnou klasifikaci jen část ze získaných dat, bude se nám co největší dataset hodit zejména pro dílčí úlohu vytváření vektorové reprezentace slov (word embedding).

2. Data musí být reprezentativní. V trénovacích datech budeme chtít mít zastoupeno co největší množství různých žánrů a druhů knih, ať už se jedná o učebnice, nebo ne-učebnice.

S ohledem na tyto požadavky jsme zvolili jako zdroje dat kombinaci internetových knihkupectví a online databází knih. Jak e-shopy, tak online databáze nabízí obvykle dostatečně velké množství dat (v případě našich zdrojů jsou to desítky až stovky tisíc knih) a zároveň poskytují reprezentativní vzorek, neboť jde o knihy všech možných žánrů.

Nebudeme na tomto místě zmiňovat konkrétní názvy e-shopů, ze kterých jsme čerpali, ani nebudeme jmenovat konkrétní online databáze knih. Budeme je označovat pouze písmenem a číslem. Zde je kompletní výčet všech zdrojů:

- 8 internetových knihkupectví (označme je K1 až K8), z toho dvě zaměřená výhradně na učebnice (K7 a K8)
- 2 online databáze knih (označme je D1 a D2)

3.1.2 Stažení dat

Tento krok budeme implementovat v jazyce Python pomocí knihovny requests[1]. Stažené webové stránky si budeme ukládat a následně je parsovat. Postup stažení

[1] https://pypi.org/project/requests/

je přibližně následující a je vždy přizpůsoben konkrétnímu zdroji, zejména formát URL adres:

1. Identifikovat URL stránky s kategoriemi.
2. Stáhnout stránky kategorií ve formátu HTML.
3. Pomocí reguálních výrazů vytáhnout z HTML odkazy na jednotlivé detaily produktů/knih.
4. Stáhnout detaily ve formátu HTML.

Pro některé zdroje jsme měli tento postup usnadněný tím, že jsme získali přístup k neveřejným souborům, XML feedům s produkty, ze kterých šlo seznam URL adres detailů produktů získat bez nutnosti stahování jednotlivých kategorií. Výše uvedeným postupem by však bylo možné získat stejný seznam URL adres i bez přístupu k těmto neveřejným souborům.

Počet stažených záznamů (HTML souborů) pro jednotlivé zdroje shrnuje tabulka 3.1.

Zdroj	**Počet**	**Zdroj**	**Počet**
K1	33 136	K6	276 421
K2	152 162	K7	6 251
K3	128 089	K8	5 982
K4	93 039	D1	175 167
K5	137 397	D2	86 806

Tabulka 3.1: Počet HTML souborů pro jednotlivé zdroje

3.1.3 Parsování dat

V tomto kroku extrahujeme data ze stažených HTML souborů a uložíme strukturovaná data do souborů na disku. Opět k tomu zvolíme vhodnou knihovnu pro Python, v tomto případě lxml[2], která umožňuje parsování mimo jiné i HTML souborů. Výstupem parseru je třída reprezentující strom HTML elementů, nad kterým pak můžeme provádět XPath dotazy. Prostřednictvím vhodných dotazů vyextrahujeme data a uložíme je (s využitím knihovny pandas[3]) do souboru. Jako formát souboru budeme používat klasické CSV (textová reprezentace, hodnoty odděleny čárkami či středníkem). Dotazy budeme konstruovat zvlášť pro každý zdroj dat, neboť se struktura HTML kódu samozřejmě liší zdroj od zdroje. Pro zjištění struktury a umístění příslušných dat uvnitř HTML budeme analyzovat několik náhodně vybraných stažených souborů. Níže uvádíme příklady několika XPath dotazů, které slouží k extrakci ISBN z různých zdrojů:

```
//*[@itemprop="identifier"]/text()
//td[text()="ISBN:"]/following-sibling::th[1]/text()
//dt[text()="ISBN"]/following-sibling::dd[1]/text()
```

[2]https://lxml.de/
[3]https://pandas.pydata.org/

Už v této fázi budeme vyřazovat produkty, pro které se nám nepodaří zjistit jejich ISBN/EAN, a produkty, jejichž EAN kód neodpovídá formátu pro knihy (nemá prefix 978 nebo 979). Dále přeskočíme záznamy, u nichž se nám z jakéhokoliv důvodu nepovede extrahovat název, což může být v důsledku toho, že produkt nebyl nalezen na stažené URL adrese nebo došlo k jiné chybě, která způsobila, že nebyl správně vrácen detail produktu. Počty extrahovaných záznamů z jednotlivých zdrojů, které mají ISBN, zachycuje tabulka 3.2.

Zdroj	**Počet**	**Zdroj**	**Počet**
K1	24 774	K6	120 968
K2	116 162	K7	6 251
K3	95 104	K8	5 294
K4	63 822	D1	121 670
K5	95 769	D2	86 757

Tabulka 3.2: Počet knih v jednotlivých zdrojích

Přestože se počty výrazně snížily (v jednom případě i na méně než polovinu), stále máme celkem 736 571 záznamů o knihách.

3.2 Předzpracování dat

Dalším krokem bude předzpracování dat, kdy stažená surová data vyčistíme a pokusíme se také vytvořit rejstřík všech autorů a nakladatelů, kde každý autor či nakladatel bude mít přiřazeno vždy jedno ID, přestože může být v datech reprezentován odlišným textovým řetězcem (zkrácení jména autora na první písmeno, (ne)uvedení typu společnosti u nakladatelství atd.). Tím budeme mít připravena data pro převod do reprezentace, kterou následně použijeme v neuronových sítích – název a popis jako vektory slov a autor a nakladatel jako entita, resp. jejich ID z rejstříku.

3.2.1 Čištění dat

Surová data obsahují spoustu „šumu“, který se budeme snažit do určité míry eliminovat. Konkrétně budeme provádět následující kroky:

- Z popisu odstraníme HTML tagy. Budeme se však snažit zachovat strukturu textu, tedy pokud někde byl tag
, mělo by zde zůstat odřádkování, kde byly HTML seznamy, budou jednotlivé položky na samostatných řádcích apod. Výsledkem bude čistý text bez jakýchkoliv formátovacích značek.
- Odstraníme přebytečné mezery a odřádkování. Jde o mezery na začátku a konci každého textového údaje, více mezer či tabulátorů za sebou, které nahradíme jednou mezerou, a také odstraníme prázdné řádky.
- Odstraníme autory, nakladatele a popisy, které oznamují, že daný údaj chybí. Například nakladatel „neuveden“, nebo autor „nemá autora“. Za tímto účelem si sestavíme seznam takových hodnot, který získáme ruční

analýzou dat. Zde využijeme především toho, že nechtěné hodnoty se zpravidla budou vyskytovat opakovaně a budou se vyskytovat mezi nejfrekventovanějšími hodnotami příslušného parametru.

3.2.2 Párování nakladatelů

V dosavadních datech máme nakladatele knihy reprezentovaného jako textový řetězec. S touto reprezentací však přichází problém, kdy stejné nakladatelství může být v různých zdrojích dat uváděno odlišně. K tomu dochází z různých důvodů:

- Vynechání některých částí delšího oficiálního názvu – např. „Svojtka“ vs „Svojtka & Co.“, „Fraus“ vs „Nakladatelství Fraus, s.r.o.“.
- Použití zkratky názvu – „OUP“ vs „Oxford University Press“.

Skutečnost, že dvě různé textové hodnoty reprezentují stejného nakladatele, obvykle není pro člověka těžké odhalit. Automatické rozhodování už ale tak triviální není, proto využijeme skutečnosti, že můžeme spojit záznamy o stejné knize z různých zdrojů. Pokud není v jednom ze zdrojů chyba, můžeme předpokládat, že oba textové řetězce v údaji o nakladateli reprezentují totéž nakladatelství.

Matematicky by se dalo říci, že relace, ve které dva textové řetězce reprezentují stejného nakladatele, je ekvivalencí. Sestavení rejstříku nakladatelů tak není nic jiného, než rozklad množiny všech možných řetězců s nakladateli na třídy ekvivalence. Toho můžeme dosáhnout jednoduše tak, že jednotlivé řetězce budou reprezentovány vrcholy neorientovaného grafu a mezi dvěma vrcholy povede hrana právě tehdy, když odpovídající řetězce představují téhož nakladatele. Každá ze souvislých komponent výsledného grafu pak zastupuje jednoho „skutečného nakladatele“, tedy jedno ID z rejstříku.

Pokud bychom měli o nakladatelích spolehlivá data, měla by výše uvedená metoda fungovat. V praxi se však stane to, že ve stejné komponentě grafu, resp. pod stejnou entitou, skončí často mnoho textových řetězců, které na první pohled reprezentují různé vydavatele.

Příčinou jsou zřejmě situace, kdy je v jednom ze zdrojů uveden chybný nakladatel, nebo když má kniha výjimečně dva nakladatele. Ačkoliv jsou obě situace spíše vzácné, vedou v důsledku ke spojení dvou velkých komponent, které ve skutečnosti reprezentují dva různé nakladatele. Dalším problémem jsou velká nakladatelství, která používají různé značky – např. nakladatelství Grada vydává knihy pod značkami Bambook, Cosmopolis a další, zejména v závislosti na žánru knihy. Přestože by nebylo chybou považovat všechny tyto značky za jedno nakladatelství, větší informační hodnotu pro nás určitě bude mít, když budeme mezi jednotlivými značkami rozlišovat.

Zavedeme proto ještě některé metriky pro každou dvojici textových řetězců. Empiricky určíme podmínky, za kterých budeme danou hranu uvažovat (tedy kdy budeme považovat dva řetězce za reprezentace stejného nakladatele) a „graf nakladatelů“ zredukujeme pouze na hrany splňující tyto podmínky. Nastavení podmínek nám umožní zvolit kompromis mezi většími skupinami s více chybně spojenými nakladateli a menšími skupinami s méně chybami, ale s rizikem, že jeden nakladatel bude reprezentován více skupinami. Budeme využívat následující metriky:

- Jaro-Winklerovu metriku (JW) pro podobnost řetězců (viz 2.1.3)
- Nejdelší společný podřetězec (NSP) (viz 2.1.3)

Obě metriky souvisí s tím, že jde-li o jednoho nakladatele, měly by být řetězce podobné. Jako metriku bychom mohli vzít také počet dvojic záznamů se stejným ISBN, které mají jako nakladatele uvažovanou dvojici. Taková metrika by nám ale nepomohla eliminovat problém značek nakladatelství, protože podobné dvojice by se mohli objevovat velmi často.

Výsledné podmínky jsme nastavili empiricky, zkoušením různých hodnot. Následující nastavení se ukázalo být vhodným pro naše potřeby:

- $JW > 0{,}5$
- a zároveň $NSP > 0{,}9$

3.2.3 Párování autorů

Pro údaj o autorech provedeme obdobnou věc jako u nakladatelů – pokusíme se převést textové reprezentace autorů na „entity“ a sestavit si rejstřík všech autorů, v ideálním případě bez duplicit. Oproti nakladatelům jsou u autorů dvě významné odlišnosti:

- Autorů bývá často u jedné knihy více, a někdy tak bude potřeba rozdělit správně vstupní řetězec na jednotlivé autory. Speciálním případem „autora“ je kolektiv autorů, který bývá zapsán v řetězci několika různými formami.
- Jména osob mají relativně přesně dané složky (jméno, příjmení, tituly), které však mohou být zapsány mnoha odlišnými styly. Například všechny následující řetězce mohou reprezentovat stejného autora: „Jan Novák“, „J. Novák“, „Novák Jan“, „Novák, J.“, „Ing. Jan Novák“.

Z tohoto důvodu budeme postupovat jinak než u nakladatelů – nebudeme porovnávat podobnost řetězců, ale zaměříme se jednak na rozdělení řetězce na jednotlivé autory a u každého autora pak na rozložení jména na jednotlivé složky. Klíčové myšlenky algoritmu budou následující:

- Textová reprezentace autora může obsahovat na konci různé údaje o překladatelích, ilustrátorech apod. Tyto údaje budeme chtít odstranit. Sestavíme si za tím účelem seznam slov, která tyto údaje uvozují a celou část řetězce od tohoto slova dále smažeme.
- Odstraníme z textu vše v závorkách a také tituly, pro které vytvoříme seznam všech možných titulů. Důvod pro odstranění titulů je, že není příliš pravděpodobné, aby se dva autoři lišili pouze tituly, navíc titul se může i změnit („Bc. Jan Novák“ a „Ing. Jan Novák“ může být stejná osoba).
- Sestavíme si seznam oddělovačů autorů, podle kterých textovou reprezentaci autorů rozdělíme na jednotlivé autory. Speciálním případem bude čárka a středník, které mohou oddělovat i jméno a příjmení zapsané inverzně („Novák, J.“). Pro tyto oddělovače použijeme určité heuristiky, podle kterých se rozhodneme, jestli oddělují jednotlivé autory, nebo jenom části jména jednoho autora.

- Na konci algoritmu budeme mít vstupní textový řetězec převedený na pole autorů, kde každý autor bude reprezentován polem složek jeho jména (pouze jméno a příjmení, bez titulů).

Máme-li autory ve strukturované reprezentaci, je potřeba umět rozhodnout, jestli dvě hodnoty reprezentují stejnou osobu. Nestačí přitom porovnat pouze jednotlivé složky jména mezi sebou, a to z několika důvodů:

- Křestní jména mohou být zkrácena na první písmeno, nebo mohou zcela chybět.
- Je potřeba dávat pozor na situace, kdy sice odpovídají iniciály, ale jde o různá příjmení. Např. Stanislav Dvořák a David Svoboda, zapsaní jako „Dvořák S.“ a „D. Svoboda“. Musí vždy platit, že alespoň jedno nezkrácené jméno odpovídá u obou osob.

Samotné párování autorů provedeme tím způsobem, že si rozdělíme autory do „přihrádek“ podle jejich iniciál seřazených abecedně. Následně porovnáme autory stylem každý s každým v rámci jednotlivých přihrádek. Pokud nám vyjde, že dva záznamy reprezentují stejného autora, přidáme jej do rejstříku pouze jednou, a to s co nejvíce částmi jména nezkrácenými.

3.3 Výsledný dataset

Než začneme popisovat výsledný dataset, upřesníme názvosloví, které v souvislosti s daty budeme používat.

- **Záznam**, **vzorek** – těmito slovy budeme vždy označovat údaje o jedné knize získané z jednoho zdroje. Pokud bychom o stejné knize získali údaje například z 5 zdrojů, budeme mít v datasetu 5 záznamů, které se budou týkat stejné knihy.
- **Kniha** – je v našich datech jednoznačně identifikovaná svým ISBN. Pokud mají dva záznamy stejné ISBN, pak jsou to záznamy o stejné knize. Když tedy mluvíme o počtu knih, máme na mysli počet unikátních ISBN a tak dále.

Může se stát, že budeme mít v datech dva záznamy, které budou mít všechny údaje shodné, avšak budou z různých zdrojů. Tyto duplicity nebudeme nijak odstraňovat, právě proto, že pochází z jiných zdrojů a odráží tak „reálný stav světa“.

3.3.1 Oddělení trénovacích a testovacích dat

Data, která jsme získali z našich zdrojů, obsahují ISBN knihy, její název a případně popis, autora a nakladatele. Zatím však o nich nevíme, zda to jsou učebnice a do které z námi definovaných předmětů a úrovní patří. Tyto anotace budeme pro trénování neuronových sítí potřebovat a musíme je proto ručně doplnit. Protože je však anotace dat časově náročná, rozhodli jsme se neanotovat

všechna data, ale vybrat pouze jejich podmnožinu. Tato podmnožina bude mít 15 000 knih a způsob, jakým ji vybereme, je popsán níže. Těchto 15 000 knih dále rozdělíme do dvou skupin:

- Trénovací množina (10 000 knih) – tato data využijeme k trénování modelů, můžeme se na ně dívat během čištění dat a na základě těchto knih sestavíme rejstříky autorů a nakladatelů.
- Testovací množina (5 000 knih) – tuto množinu si dáme stranou ještě ve formě surových dat a nebudeme se na ni dívat. V závěru ji využijeme k finálnímu otestování modelů na neznámých datech.

Využijeme znalosti ISBN a toho, že pomocí něj můžeme slučovat záznamy o stejné knize. Každé ISBN nám tak stačí anotovat pouze jednou a tuto anotaci přenést na všechny záznamy o příslušné knize. Uvedené počty knih v množinách (10 000 a 5 000) tedy v souladu s názvoslovím zavedeným výše značí počet jedinečných ISBN, nikoliv počet záznamů, kterých může být více.

Při výběru ISBN k anotaci bychom správně měli postupovat zcela náhodně a vybrat tak reprezentativní vzorek dat z množiny, jenž máme k dispozici. Takový vzorek by však trpěl nedostatkem, že by obsahoval málo učebnic, které jsou obecně mezi knihami zastoupeny méně než ne-učebnice. Tuto situaci vyřešíme pomocí znalosti, že dva ze zdrojů (K7 a K8) obsahují převážně učebnice a budeme postupovat následovně:

- Polovinu z celkového počtu 15 000 ISBN (tedy 7 500) vybereme náhodně z takových ISBN, která se objevují v datech z alespoň jednoho ze zdrojů K7 a K8.
- Druhou polovinu naopak vybereme náhodně z těch ISBN, která se nevyskytují v datech zdroje K7 ani zdroje K8.
- Do trénovací množiny dáme 5 000 náhodně vybraných ISBN z každé z polovin, ostatní ISBN dáme do testovací množiny.

3.3.2 Příprava testovacích dat

Množinu testovacích dat jsme oddělili od zbytku dat ještě před čištěním dat a párováním autorů a nakladatelů. Před otestováním natrénovaných modelů proto musíme tyto operace na testovacích datech provést.

Čištění dat bude shodné s čištěním zbytku dat, tak jak bylo popsáno v části 3.2.1. Vzhledem k tomu, že jsme při sestavování metody čištění testovací data nepoužívali, může i po čištění v testovacích datech zůstat určitý šum (text oznamující, že autor knihy není uveden apod.). Budeme předpokládat, že tento šum nijak výrazně neovlivní výslednou úspěšnost algoritmu.

Dále je nutné převést v testovacích datech nakladatele na jejich ID z rejstříku. Jednoduše budeme hledat v rejstříku nakladatelů přesnou shodu s řetězcem v testovacích datech. Pokud přesnou shodu nenalezneme, budeme nakladatele ignorovat. V opačném případě si do testovacích dat zapíšeme nalezené ID. Budeme zde předpokládat, že v rejstříku máme většinu možných reprezentací pro všechny nakladatele, a to s ohledem na skutečnost, že byl rejstřík sestaven za použití dat pocházejících ze stejné množiny, z jaké byla oddělena i testovací podmnožina.

Nakonec zbývá dohledat v rejstříku autory. Zde využijeme stejný proces parsování řetězce s autory na pole složek jejich jmen, jako v kapitole 3.2.3. Následně budeme porovnávat parsované autory s rejstříkem, opět stejným postupem jako při vytváření rejstříku, a zapíšeme si do testovacích dat autory reprezentované jejich ID. Pokud některého autora nenalezneme, budeme jej ignorovat.

3.3.3 Statistiky datasetů

Analyzovali jsme výsledné množiny dat, na kterých budeme testovat a trénovat množiny. Charakteristiky jednotlivých údajů zachycuje tabulka 3.3.

	Trénovací data	**Testovací data**
Počet unikátních ISBN	10 000	5 000
Celkem záznamů	28 714	14 196
Počet záznamů s autorem	24 186	7 121
Počet záznamů s nakladatelem	27 690	13 666
Počet záznamů s popisem	23 184	11 530
Průměrná délka popisu	112 slov	111 slov

Tabulka 3.3: Charakteristiky trénovací a testovací množiny dat

Dále jsme zkoumali jazyky, ve kterých jsou napsané popisy a názvy knih. Data totiž obsahují i cizojazyčné knihy, ať už se jedná o učebnice pro výuku daného jazyka, nebo i o fikci. Výsledky – počty záznamů s názvem/popisem v daném jazyce – jsou v tabulce 3.4. Detekci jazyka jsme prováděli pomocí modelů FastText[4], jako jazyk jsme vybrali vždy ten, který byl podle modelu nejpravděpodobnější. V datech se objevují také případy, kdy popis knihy je ve dvou jazycích (například část popisu učebnice angličtiny v češtině, část v angličtině). Tyto případy jsme zanedbávali a podobné záznamy se tak objevují v tabulce vždy jen jednou, a to u jazyka, který byl pro model nejpravděpodobnější – zřejmě se tedy bude jednat o jazyk v daném textu převažující.

Uvedenou tabulku je potřeba brát s rezervou, zvláště údaje o názvech knih – jde o velmi krátké řetězce, proto zde očekáváme vysokou chybovost. Nejsme si vědomi toho, že některé uvedené jazyky by se v názvech vůbec vyskytovaly (slovinština, portugalština).

3.4 Anotace

Protože budeme v této práci využívat strojové učení s učitelem, je potřeba mít dostatečně velkou podmnožinu našich dat anotovanou. Anotace budeme provádět ručně na základě stažených dat o knihách, ale taky (pokud bude třeba) na základě externích informací, např. obrázku knihy z internetu.

3.4.1 Výběr dat k anotaci

Anotovat budeme všechna data v trénovací a testovací množině. Výběr dat pro tyto množiny jsme popsali v části 3.3.1. Máme celkem 15 000 knih (unikátních

[4]https://fasttext.cc/

Jazyk názvu	Trénovací data	Testovací data
Čeština	21 751 (75,75 %)	10 579 (74,52 %)
Angličtina	3 244 (11,30 %)	1 641 (11,56 %)
Slovenština	1 260 (4,39 %)	697 (4,91 %)
Němčina	714 (2,49 %)	367 (2,59 %)
Polština	384 (1,34 %)	200 (1,41 %)
Francouzština	249 (0,87 %)	148 (1,04 %)
Španělština	181 (0,63 %)	110 (0,77 %)
Slovinština	162 (0,56 %)	77 (0,54 %)
Portugalština	118 (0,41 %)	–
Italština	117 (0,41 %)	61 (0,43 %)
Ostatní (< 0,4 %)	534 (1,86 %)	316 (2,23 %)

Jazyk popisu	Trénovací data	Testovací data
Čeština	20 204 (87,15 %)	9 941 (86,22 %)
Angličtina	1 446 (6,24 %)	747 (6,48 %)
Slovenština	1 012 (4,37 %)	532 (4,61 %)
Polština	269 (1,16 %)	118 (1,02 %)
Němčina	122 (0,53 %)	72 (0,62 %)
Francouzština	–	72 (0,62 %)
Ostatní (< 0,4 %)	131 (0,57 %)	48 (0,42 %)

Tabulka 3.4: Detekované jazyky názvů a popisů knih

ISBN), pro které rozhodneme, jestli jsou učebnicí, nebo nikoliv. Pro knihy, které zařadíme mezi učebnice, pak ještě určíme předmět a úroveň.

Protože pro jedno ISBN můžeme mít více záznamů (z různých zdrojů), provedeme jejich sloučení, abychom získali CSV soubor s jedním ISBN na řádek. Použijeme heuristiku, kdy do sloučeného záznamu jednoduše vybereme pro každý z parametrů knihy (název, autor, nakladatel, popis) tu hodnotu, která má nejvyšší počet znaků (tedy nejdelší název, popis atd.). Předpokládáme, že tím dostane anotátor o knize dostatek informací a případné další potřebné informace si vyhledá na internetu (viz pokyny k anotaci v následující části).

3.4.2 Pokyny k anotaci

Před samotnou anotací se anotátor musí seznámit s definicí učebnice v části 1.1.3 a seznamem předmětů a úrovní v části 1.2. Anotátorovi bude následně předána tabulka knih (CSV soubor) s údaji, které o knihách sbíráme (ISBN, název, autor, nakladatel, popis) a se třemi prázdnými sloupečky, kam je třeba doplnit anotace. Pro každou knihu musí anotátor:

1. Rozhodnout, zda daná kniha je učebnicí dle definice.
2. Pokud jde o učebnici, rozhodnout, do kterého patří předmětu a pro jakou úroveň je určena.

V případě, že nelze na základě poskytnutých údajů učinit jednoznačné rozhodnutí, může si anotátor vyhledat podle ISBN o knize více informací na internetu. Využít může i obrázek obalu knihy, zařazení knihy do kategorie na e-shopu atd.

Jestliže ani poté nelze rozhodnout, uplatníme následující pravidla:

- Je-li nejasné, zda je kniha učebnicí, nebo nikoliv, přikloníme se k variantě učebnice. Toto pravidlo vychází z předpokladu, že v praxi, pokud budeme chtít vybírat učebnice z nějaké množiny knih, raději takovou neurčitou knihu přijmeme – např. do sortimentu e-shopu, do nabídky bazaru učebnic atd.
- Pokud není jasný předmět, zařadíme knihu do kategorie Ostatní.
- Pokud je sporná úroveň, zařadíme knihu do vyšší ze dvou úrovní, mezi kterými je těžké se rozhodnout. Hierarchie úrovní (od nejnižší po nejvyšší) je následující: 1. stupeň ZŠ, 2. stupeň ZŠ, Střední škola, Jazykové učebnice, Autoškola. V těchto situacích, např. je-li učebnice určena pro 2. stupeň ZŠ a střední školy, není důvod preferovat ani jednu variantu, avšak uvedené pravidlo zavedeme kvůli konzistenci rozhodování (jak jednoho anotátora, tak mezi anotátory).

3.4.3 Výsledky anotace

Anotace byla provedena dvěma anotátory, z nichž jedním byl autor této práce (budeme jej označovat jako anotátora A). Druhý anotátor (označme jej jako anotátora B) dostal jako zadání pouze definice a pokyny uvedené v této práci a na jejich základě anotoval přidělenou množinu dat. V následující tabulce 3.5 je uvedena mezianotátorská shoda pro každý typ anotace – učebnice/ne-učebnice, předmět a úroveň. Abychom mohli porovnávat mezianotátorskou shodu pro úlohy klasifikace učebnic do předmětů a úrovní, prováděli oba anotátoři tyto anotace na množině knih, které označil za učebnici anotátor A.

Úloha	**Shoda**
Rozpoznávání učebnic	95,72 %
Předmět učebnice	96,99 %
Úroveň učebnice	91,40 %

Tabulka 3.5: Mezianotátorská shoda pro jednotlivé úlohy

Rozdíly v anotacích úlohy rozpoznávání učebnic byly způsobeny především nedokonalostmi naší definice učebnice a obvykle šlo o knihy, které v jednom či více bodech definice byly „na hraně“. Několik příkladů:

- Zda knihy k přípravě na přijímací zkoušky na VŠ lze použít ve výuce na SŠ, nebo už patří do VŠ učebnic.
- Jestli se některé jazykové učebnice pro děti využívají i v první třídě ZŠ, nebo jen pro předškolní výuku.
- Zda jsou některé učebnice (různých předmětů) určeny pro SŠ nebo až pro VŠ.

- Jestli určité cvičebnice jsou ještě učebnicemi, nebo spíše knihou zábavných aktivit, při kterých si žáci opakují učivo.

U úlohy klasifikace předmětů byla mezianotátorská shoda nejvyšší. Část rozdílů zde navíc tvořily chyby z nepozornosti jednoho či druhého anotátora. Zbytek byly většinou učebnice na rozhraní dvou předmětů, šlo například o tyto případy:

- Různé učebnice pro učiliště, pro předměty, u kterých je těžké říct, zda patří k některému z „podobných“ předmětů, nebo spíše do kategorie Ostatní.
- Matematické a fyzikální tabulky – zda patří do matematiky (obsahují vzorce, konstanty a další matematické věci), nebo do kategorie Ostatní (vztahují se k více předmětům – matematice i fyzice).
- Výchova k finanční gramotnosti – předmět Ekonomie, nebo Základy společenských věd?

V poslední z úloh, klasifikaci úrovní, se kromě pár chyb z nepozornosti vyskytovala většina chyb u učebnic cizích jazyků. Toto pozorování jsme si potvrdili následujícím měřením:

- Za učebnice cizích jazyků budeme považovat takové učebnice, které anotátor A zařadil do jednoho z následujících předmětů: Angličtina, Francouzština, Italština, Němčina, Ruština, Španělština, Ostatní jazyky.
- V anotovaných datech je pak 2 360 učebnic cizích jazyků a 4 048 ostatních učebnic (celkem 6 408 knih, které označil anotátor A za učebnice).
- Mezianotátorská shoda ohledně úrovně učebnice byla pro podmnožinu učebnic cizích jazyků pouze 80 %. Naproti tomu shoda u ostatních učebnic činila téměř 98 %.

Vysvětlením výše uvedeného jevu je, že jazykové učebnice mají svůj rámec úrovní (začátečníci, pokročilí atd., případně A1-C2) a často je proto velmi obtížné odhadnout, zda jde o učebnici pro 2. stupeň ZŠ, pro střední školy, případně zda bychom měli učebnici řadit mezi učebnice pro jazykové školy/kurzy a samostudium.

3.4.4 Rozdělení dat

V předchozí sekci jsme zmínili, že anotace prováděli dva anotátoři, které jsme označili jako anotátor A a anotátor B. Měli jsme tak k dispozici dvě sady anotací a rozhodli jsme, že pro účely trénování a testování modelů budeme používat anotace od anotátora A a anotátora B použijeme pouze pro stanovení mezianotátorské shody (tabulka 3.5) a pro určení podobnosti „chyb“ anotátora B s chybami natrénovaného modelu (tabulka 5.18).

Níže uvádíme rozdělení dat v jednotlivých třídách pro každou z úloh. V tabulkách 3.6, 3.7 a 3.8 uvádíme počty knih (tedy unikátních ISBN, nikoliv záznamů).

	Počet
Není učebnice	8 592
Je učebnice	6 408

Tabulka 3.6: Rozdělení dat pro úlohu rozpoznávání učebnic

Předmět	**Zkratka**	**Počet**
Angličtina	AJ	1 470
Český jazyk a literatura	CJ	1 175
Matematika a geometrie	MA	911
Němčina	NJ	453
Ostatní	OS	222
Základy společenských věd	ZS	204
Francouzština	FJ	200
Biologie a přírodopis	BI	194
Zeměpis	ZE	185
Fyzika a geologie	FY	185
Dějepis	DE	178
Ekonomie a účetnictví	EK	150
Chemie	CH	115
Technika a strojírenství	TE	111
Prvouka	PR	95
Ruština	RJ	89
Medicína a ošetřovatelství	ME	86
Vlastivěda	VL	81
Španělština	SJ	81
Informatika	IT	56
Stavebnictví	ST	40
Hudební výchova	HU	40
Ostatní jazyky	OJ	38
Italština	IJ	29
Výtvarná výchova	VY	20

Tabulka 3.7: Rozdělení dat pro úlohu klasifikace předmětů

Úroveň	**Počet**
Střední školy a učiliště	2 458
1. stupeň ZŠ	1 473
2. stupeň ZŠ	1 396
Jazykové školy a samostudium	1 049
Autoškoly	32

Tabulka 3.8: Rozdělení dat pro úlohu klasifikace úrovní

4. Modely

Na začátek této kapitoly uvedeme, jak budeme upravovat reprezentaci vyčištěných dat z minulé kapitoly za účelem jejich použití v klasifikačních algoritmech. Dále zde popíšeme jednotlivé modely, které použijeme k rozpoznávání učebnic a ke klasifikaci učebnic do předmětů a úrovní – nejprve naivní bayesovský klasifikátor, který nám poslouží jako baseline, a následně i modely hlubokého učení.

4.1 Reprezentace dat

Než se pustíme do klasifikace, je nezbytné převést data do vhodné reprezentace, kterou budou naše modely umět zpracovávat. Pro název a popis knihy jsme se rozhodli využít převod slov na vektory pomocí techniky word embedding (viz část 2.1.4), a to jak s předchozí lemmatizací slov, tak bez ní. Pro nakladatele a autory využijeme jednodušší převod jejich ID na one-hot vektory.

4.1.1 Word embedding

Pro klasifikátory založené na hlubokém učení budeme používat pro textové parametry (název a popis) techniku vnoření slov (word embedding). Za tímto účelem využijeme jednak předpřipravené embeddingy, stažené z internetu, které byly natrénované na rozsáhlých korpusech (zde konkrétně Wikipedie), ale zkusíme také natrénovat vlastní vektory slov, a to přímo na datech o knihách, která máme k dispozici. Vyzkoušíme natrénovat i modely, kde se na začátku použijí pro slova náhodné vektory, které se následně trénují společně se zbytkem sítě.

Vlastní embeddingy Vlastní embeddingy budeme trénovat pomocí knihovny Gensim[1] (Řehůřek a Sojka, 2010), která za tímto účelem nabízí mnoho různých modelů a algoritmů. Na základě práce Hořeňovské (Hořeňovská, 2019) jsme se rozhodli použít známý model Word2Vec a zvolit algoritmus CBOW (Continous Bag-of-Words), který dle závěrů zmíněné práce poskytuje pro český jazyk lepší výsledky než algoritmus Skip-gram. Jako korpus použijeme názvy a popisy všech knih, které máme k dispozici, mimo ty, jenž jsme si dali stranou jako trénovací a testovací množinu. Půjde tedy o všechny záznamy, které nemáme anotované. Co se týče vícejazyčnosti (viz část 3.3.3), necháme natrénovat slova z různých jazyků dohromady, do jednoho prostoru, přestože pro cizojazyčná slova nemusíme získat vzhledem k menšímu objemu dat ideální výsledky.

Stažené embeddingy Předtrénované embeddingy si stáhneme zvlášť pro každý vybraný jazyk. Jazyky budeme vybírat jednak na základě nejfrekventovanějších jazyků v trénovací a testovací sadě dat (tabulka **??**) a také podle toho, jaké cizí jazyky máme mezi předměty, do kterých budeme klasifikovat učebnice (viz 1.2). Konkrétně jsme zvolili češtinu, slovenštinu, angličtinu, němčinu, francouzštinu, polštinu, ruštinu, italštinu a španělštinu. Dále budeme požadovat, aby vektory pro různé jazyky byly zarovnány v jednom prostoru a aby si tak vektory

[1] https://radimrehurek.com/gensim/

slov z různých jazyků mající stejný význam byly navzájem podobné. Z tohoto důvodu využijeme embeddingy z knihovny MUSE[2] (Conneau a kol., 2017). Jednotlivé soubory s vektory po stažení sloučíme do jednoho. Pokud se některé slovo vyskytuje ve více jazycích, dostane se do výsledných embeddingů vektor z dříve zpracovávaného jazyka, přičemž jazyky se zpracovávají v pořadí, v jakém jsou vypsané výše. Nakonec omezíme embeddingy pouze na ta slova, která budeme v experimentech potřebovat – konkrétně tedy všechna slova, která se vyskytují v názvu nebo popisu u alespoň jednoho záznamu v trénovací nebo testovací množině.

Dimenze embeddingů Co se týče dimenze embeddingů, tedy velikosti vektorů pro jednotlivá slova, zvolili jsme pro embeddingy trénované společně se sítí a pro vlastní předtrénované embeddingy dimenzi 100. Pro naše úlohy považujeme tuto velikost za dostačující, vzhledem k omezené doméně textů. Větší dimenze by podle nás nebyla vhodná z důvodu menší velikosti trénovacích dat. U vlastních embeddingů jsme měli pro jejich trénink k dispozici data s několika miliony slov, což je v kontextu trénování embeddingů relativně málo. U embeddingů trénovaných zároveň se sítí jsme zase vzali v úvahu velikost trénovací množiny, která činí přibližně 28 tisíc záznamů pro úlohu rozpoznávání učebnic a asi polovinu z toho pro ostatní úlohy (neboť v nich využíváme již pouze záznamy anotované jako učebnice). Stažené předtrénované embeddingy, o kterých pojednává předchozí odstavec, však mají dimenzi 300. Rozhodli jsme se je převzít tak, jak jsou, nijak do nich nezasahovat a ponechat jim původní dimenzi. Přestože jsou tyto embeddingy trénované na obecnější doméně než ostatní námi používané embeddingy, jsme si vědomi toho, že rozdíl v dimenzích může přinášet staženým embeddingům výhodu a pozitivně ovlivnit úspěšnost modelů, které je využívají.

4.1.2 Lemmatizace

Lemmatizaci, neboli převod slov na základní (slovníkový) tvar (lemma), budeme využívat pro názvy a popisy knih, přičemž k tomu použijeme knihovnu MorphoDiTa (Straková a kol., 2014). Pro každý model vyzkoušíme natrénovat jak variantu s předchozí lemmatizací textů, tak variantu bez lemmatizace, a budeme porovnávat úspěšnost obou verzí. Naším předpokladem je, že vzhledem k omezené množině trénovacích záznamů by lemmatizace (a z ní vyplývající redukce množiny různých slov) mohla vést k přesnějším modelům, přestože se tím ztrácí část informace.

Největší efekt očekáváme u naivního bayesovského klasifikátoru, kde se nepoužívají vektory slov, a tím pádem různé tvary téhož slova jsou považovány za dvě rozdílná, nezávislá slova. V hlubokých sítích může být výhoda lemmatizace omezena tím, že převádíme slova na vektory a vektory různých tvarů jednoho slova si tak mohou být navzájem podobné.

U lemmatizace se též budeme muset vypořádat s vícejazyčností textových údajů. Kromě českých modelů pro MorphoDiTu proto využijeme i všechny ostatní dostupné modely, a sice anglické (Straka a Straková, 2014), slovenské (Straka, 2017) a polské (Piasecki a Walentynowicz, 2017). Pokud text není ani v jednom

[2]https://github.com/facebookresearch/MUSE

z těchto jazyků, použijeme jako výchozí český model. Budeme předpokládat, že v případě cizojazyčného slova vrátí MorphoDiTa jako lemma původní tvar slova.

4.1.3 Autor a nakladatel

V kapitolách 3.2.2 a 3.2.3 jsme popsali převod textových reprezentací nakladatelů a autorů na entity a vytvoření jejich rejstříků. Aktuálně data díky tomu obsahují nakladatele a autory ve formě jejich číselného ID, resp. seznamu ID. Pro strojové učení je však reprezentace pomocí celého čísla nevhodná, a proto budeme jak autory, tak nakladatele převádět na vektory. K jejich kódování využijeme tzv. one-hot vektory, tedy vektor o velikosti celkového počtu autorů/nakladatelů v rejstříku, který má pro entitu s ID i všude nuly, kromě pozice i, kde je jednička. Pro autory může být pozic s jedničkou více, má-li kniha více autorů. Pokud u knihy autor či nakladatel uveden není, případně pokud se v testovacích datech objeví autor/nakladatel, který se v trénovacích datech nevyskytuje, bude vektor obsahovat samé nuly.

4.2 Naivní Bayes

Abychom dokázali posoudit výkonnost složitějších klasifikačních algoritmů, jako jsou hluboké sítě, vyzkoušíme si nejdřív sestavit jednodušší model, se kterým pak budeme algoritmy porovnávat (tzv. baseline). Jeden z nejtriviálnějších modelů by dával pro všechny vzorky vždy stejný výsledek, a sice třídu, která by se v trénovacích datech vyskytovala nejčastěji. Takový model by tedy dosahoval úspěšnosti rovnající se procentuálnímu zastoupení této třídy v testovacích datech. V praxi by ale neměl velké uplatnění, kromě toho, že by nám poskytl představu o očekávaných výsledcích – například pokud by měl uvedený model úspěšnost 90 %, víme, že klasifikátory dosahující na dané úloze úspěšnosti kolem 90 % nemůžeme považovat za příliš dobré.

Dalším jednoduchým modelem by mohl být algoritmus založený na pravidlech („pokud název knihy obsahuje slovo X, nejde o učebnici“, „pokud název obsahuje slovo Y, jde o učebnici češtiny“ apod.). My si však zvolíme algoritmus, který pracuje s jistým zobecněním tohoto principu a využívá pro jednotlivé třídy klasifikace pravděpodobnostní model. Jde o tzv. naivní bayesovský klasifikátor (viz část 2.2), který nám podle slov vyskytujících se v názvu a popisu knihy a podle jejího autora a nakladatele určí pravděpodobnost, že kniha patří do té které třídy.

Natrénujeme jednak hlavní model využívající všechny parametry knih (název knihy, popis, autor, nakladatel), ale také samostatné modely, pro každý parametr zvlášť, čímž určíme informační hodnotu každého z parametrů. Tento postup provedeme pro každou ze tří úloh.

4.3 Hluboké učení

Pro úlohy hlubokého učení, které pracují na vstupu s textem, se nejčastěji používají konvoluční neuronové sítě (CNN) nebo rekurentní neuronové sítě (RNN). Nelze obecně říci, která z nich je pro zpracování textu vhodnější, to obvykle záleží na konkrétní úloze, ale většinou platí následující:

- CNN se dokáží naučit lokální vzory, v textu malé skupiny slov, tedy fráze.
- RNN jsou naopak lepší v učení se vztahů mezi slovy, které jsou v textu dále od sebe.

V této práci jsme se rozhodli používat především konvoluční sítě, na kterých budeme porovnávat úspěšnosti modelů s různými způsoby reprezentace slov a určovat, jestli je mezi nimi statisticky signifikantní rozdíl. Vyzkoušíme ale také odlišné architektury sítí, a sice rekurentní síť a dvě architektury kombinující CNN i RNN, přičemž tyto tři modely porovnáme na testovacích datech s nejlepší CNN sítí.

4.3.1 Struktura sítí

Všechny sítě budou mít společnou obecnou strukturu, některé části se pak budou lišit na základě architektury (CNN, RNN atd.). Obecná struktura bude vypadat následovně – na začátku jsou 4 vstupní „větve“ (jedna pro každý vstupní parametr – název knihy, popis, autor, nakladatel), které pomocí vrstvy Concat sloučíme do jediného vstupu pro závěrečnou část sítě, jenž bude končit jedním výstupem (dle úlohy buď binární klasifikace, nebo klasifikace do N tříd).

První (vstupní) vrstva sítě tedy bude sestávat z těchto vstupů:

- Název knihy – vektor celých čísel, kde $n$tý prvek určuje index $n$tého slova názvu ve slovníku slov.
- Popis knihy – vektor celých čísel, kde $n$tý prvek určuje index $n$tého slova popisu ve slovníku slov.
- Autor – one-hot vektor nul a jedniček, jednička na $n$tém místě určuje, že jedním z autorů knihy je $n$tý autor z rejstříku autorů.
- Nakladatel – one-hot vektor obsahující samé nuly a jedničku na $n$tém místě, pokud nakladatelem knihy je $n$tý nakladatel z rejstříku.

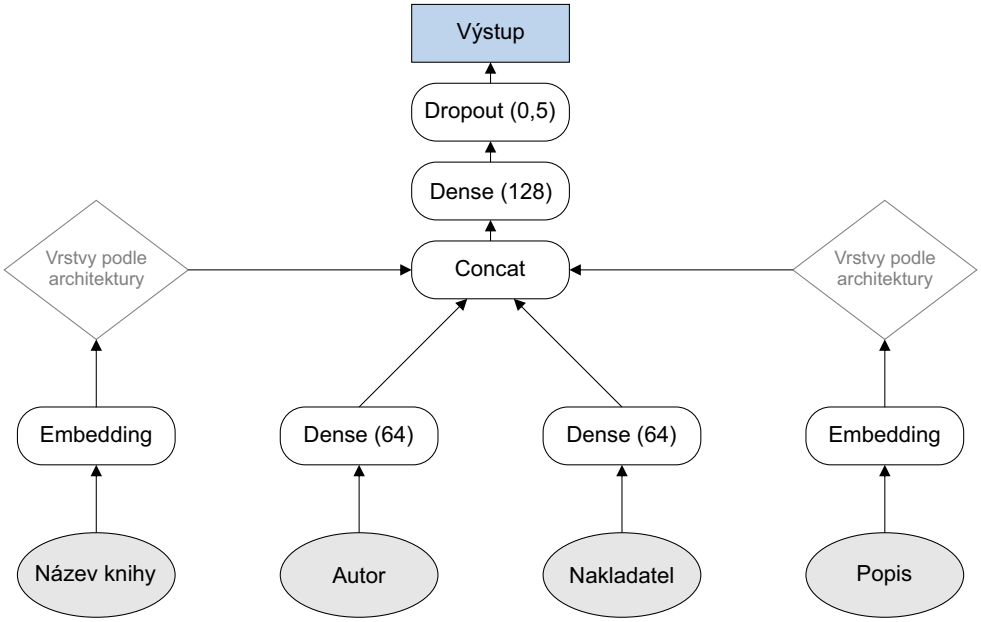
Pro větve názvu a popisu knihy bude vždy následovat vrstva Embedding, ve které použijeme buď vlastní předtrénované embeddingy, obecné předtrénované embeddingy, nebo náhodně inicializované embeddingy (viz část 4.1.1). Pro autora a nakladatele knihy bude naopak první vrstvou ve všech modelech hustě propojená vrstva (Dense).

U názvu a popisu knihy bude po vrstvě Embedding následovat část sítě závislá na architektuře – buď konvoluční a sdružovací vrstva (CNN), rekuretní vrstva (RNN), nebo kombinace těchto vrstev. V jedné architektuře se budou tyto podsítě větvit, vždy ale skončí jedním výstupem.

Nyní všechny čtyři větve sloučíme do jedné pomocí vrstvy Concat a na ni umístíme ještě jednu Dense vrstvu. Následuje regularizační Dropout vrstva, která má za úkol zabránit rychlému přeučení sítě. Funguje na jednoduchém principu, kdy náhodně vybranou část vstupů „vynuluje“, viz část 2.3.2.

Nakonec umístíme výstupní vrstvu, která nám poskytne pravděpodobnost, že daná kniha je učebnicí, nebo pravděpodobnosti pro jednotlivé třídy (předměty a úrovně).

Obecné schéma sítě je znázorněno na obr. 4.1.



Obrázek 4.1: Schéma společné struktury neuronových sítí.

4.3.2 Hyperparametry

Správné nastavení hyperparametrů je důležitou součástí tvorby co nejlepšího modelu pro danou úlohu. V našem případě se však nebudeme pouštět (s výjimkou počtu epoch) do žádného prohledávání prostoru hyperparametrů a do jejich optimalizace, ale budeme se držet „standardních“, doporučovaných hodnot, případně se inspirujeme sítěmi pro klasifikaci textů z existujících prací. Následuje přehled různých hyperparametrů a jejich hodnoty, které budeme používat.

Optimalizátor Přestože má výběr optimalizátoru vliv na konečnou úspěšnost tréninku sítě, budeme využívat pouze jeden, a sice optimalizátor Adam (Kingma a Ba, 2014). Jde o moderní a v současné době asi nejpoužívanější optimalizátor, který se ukázal být ideální volbou pro širokou škálu úloh. Navíc je poměrně rychlý a má nízké nároky na paměť.

Velikost dávky Velikost dávky nastavíme na 32, což bývá obvykle optimální hodnota (Bengio, 2012).

Počet epoch Tento hyperparametr je výjimkou a jeho hodnotu budeme pro jednotlivé experimenty ladit zvlášť. Nejprve zvolíme vyšší počet trénovacích epoch a následně vyhodnotíme křivku trénovací a validační ztráty a úspěšnosti. Na jejím základě odhadneme, kde se síť začíná přeučovat a zopakujeme trénink s upraveným počtem epoch.

Velikost Dense vrstev Ve společné struktuře sítě máme několik Dense vrstev – po jedné ve vstupních větvích pro autora a nakladatele a jednu v závěrečné části sítě, po spojení všech vstupních větví dohromady. Velikost vrstvy nesmí být především příliš malá, aby nepůsobila jako úzké hrdlo sítě, což by mělo za následek podučení (underfitting). Vybereme proto následující hodnoty:

- Vrstva u autora a nakladatele – 64 jednotek
- Vrstva v závěrečné části sítě – 128 jednotek

Konvoluční a sdružovací vrstva Počet filtrů nastavíme na 128, velikost jádra na 3. Stejně tak velikost sdružovací vrstvy bude 3.

Rekurentní vrstva Jako rekurentní vrstvu využijeme obousměrnou GRU vrstvu (Bi-GRU), s velikostí 32 jednotek pro název knihy a 64 jednotek pro popis.

Regularizace Abychom zabránili rychlému přeučení sítě, použijeme regularizační Dropout vrstvu, která vynulovává náhodně vybrané vstupy. Má jediný parametr, a sice procento vstupů, které budou vynulovány. Tuto hodnotu nastavíme na 0,5, což bývá optimální pro většinu sítí a úloh (Srivastava a kol., 2014).

Aktivační funkce, inicializátor Aktivační funkce bude záviset na konkrétním typu vrstvy a na jejím umístění v síti. Inicializátor zvolíme podle aktivační funkce – inicializátor He pro aktivační funkci ReLU a inicializátor Xavier pro ostatní aktivační funkce.

- Dense vrstva uvnitř sítě – ReLU
- Výstupní vrstva u binární klasifikace – Sigmoid
- Výstupní vrstva u klasifikace do N tříd – Softmax
- Konvoluční vrstva – ReLU
- Rekurentní (GRU) vrstva – Tanh, Sigmoid (rekurentní aktivační funkce)

Ztrátová funkce Jako ztrátovou funkci, tedy funkci, kterou se proces trénování snaží minimalizovat (viz část 2.3.3), zvolíme křížovou entropii. Pro binární klasifikaci, kde je výstupem algoritmu pravděpodobnost, že daný vzorek má určitou vlastnost (v našem případě, že kniha je učebnice), jde o binární křížovou entropii, u klasifikace do N tříd pak o kategorickou křížovou entropii.

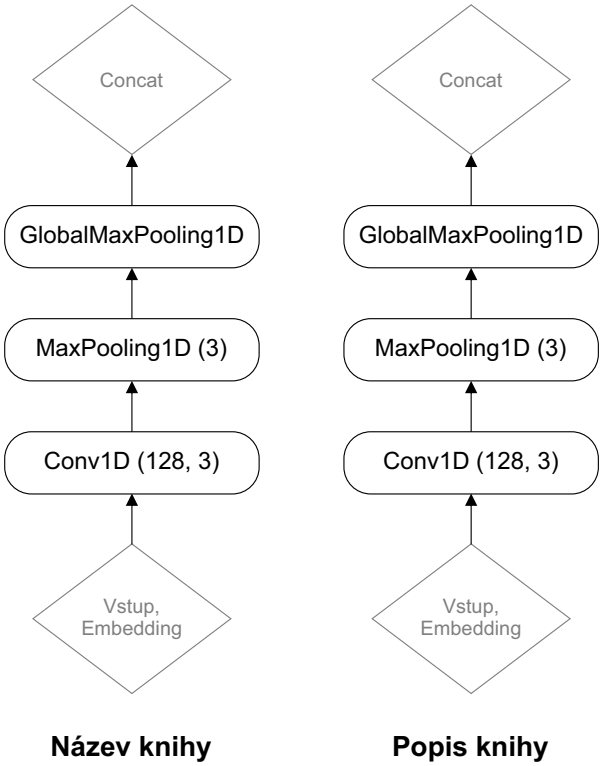
Embeddingy Způsob reprezentace slov se dá také považovat za hyperparametr a v tomto případě budeme zkoušet trénovat modely pro různé možnosti, o kterých pojednává část 4.1.1. Velikost embeddingů bude 100, kromě stažených embeddingů, které mají velikost 300.

- Embeddingy inicializované náhodně a natrénované společně se sítí.
- Embeddingy natrénované na původních tvarech všech slov v názvech a popisech knih v neanotovaných datech.
- Embeddingy natrénované na lemmatech všech slov v názvech a popisech knih v neanotovaných datech.
- Stažené vícejazyčné embeddingy, které byly trénovány na článcích z Wikipedie.

4.3.3 Architektury sítí

Kromě hlavní konvoluční sítě jsme vybrali další 3 architektury, které budeme na testovacích datech porovnávat s CNN. Jde o rekurentní síť a dvě varianty kombinované sítě, kde se objevuje jak konvoluční, tak rekurentní vrstva. Jak bylo uvedeno výše, část sítě je společná všem architekturám a ty se od sebe liší pouze ve způsobu, jakým se zpracovává vstupní text názvu knihy a popisu knihy.

Konvoluční (CNN) síť Tato síť má pro textové parametry podsíť složenou z jedné konvoluční vrstvy (Conv1D), následované sdružovací vrstvou (MaxPooling1D) a zakončenou globální sdružovací vrstvou (GlobalMaxPooling1D). Schéma podsítě je na obrázku 4.2.

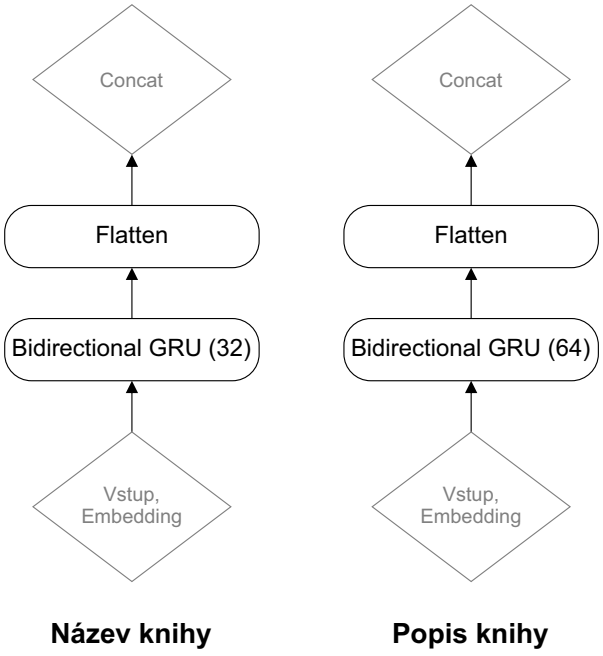


Obrázek 4.2: Konvoluční síť

Rekurentní (RNN) síť RNN síť bude mít jednu obousměrnou GRU vrstvu a vrstvu Flatten, která výstup převede na jednorozměrný vektor. Schéma podsítě zachycuje obrázek 4.3.

Hluboká konvoluční a rekurentní (Deep-CNN-RNN) síť První z kombinovaných sítí bude mít konvoluční a rekurentní vrstvu naskládané na sebe. Myšlenka je taková, že rekurentní vrstva bude zpracovávat sekvenci příznaků, které ze vstupních dat vyextrahovala konvoluční vrstva. Schéma podsítě je znázorněno na obrázku 4.4.

Duální konvoluční a rekurentní (Dual-CNN-RNN) síť Tato architektura bude rovněž kombinovat CNN a RNN síť, ale v tomto případě budou stát vedle



Obrázek 4.3: Rekurentní síť

sebe a jejich výsledky budou zřetězeny do jednoho výstupu. Cílem této architektury je využít potenciál jak konvoluční, tak rekurentní vrstvy a vybrat to nejlepší z jejich výstupů. Schéma podsítě je na obrázku 4.5.

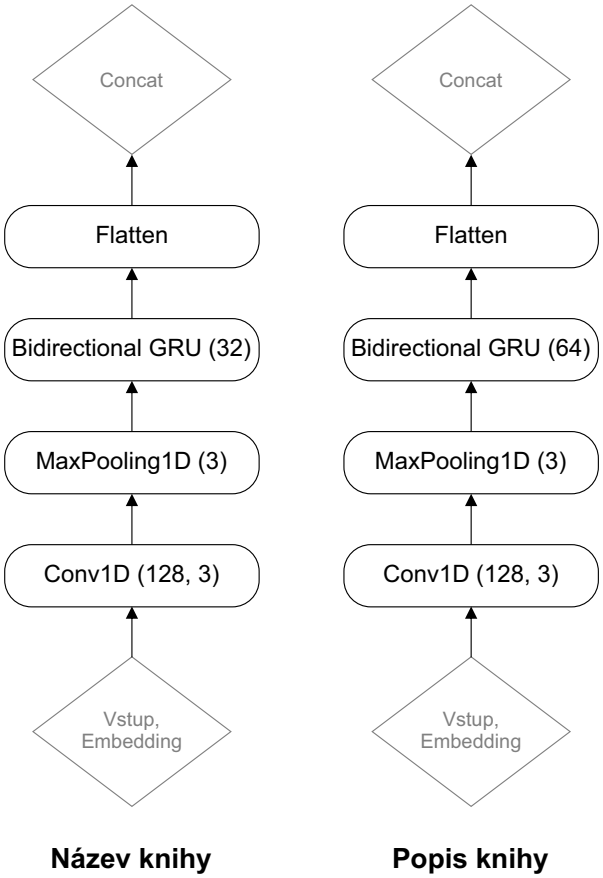
4.3.4 Vyhodnocování modelů

Pro vyhodnocování a porovnávání modelů použijeme techniku zvanou $k$násobná krosvalidace (k-fold cross-validation), kdy rozdělíme trénovací množinu na k stejně velkých částí a natrénujeme stejný model $k$krát, přičemž vždy z trénovacích dat oddělíme jednu z částí jako validační data. Zvolíme $k = 5$, abychom měli dostatečný počet trénovacích i validačních dat (v tomto případě 8 000/2 000 unikátní ISBN) a zároveň aby nebylo potřeba trénovat až příliš vysoký počet modelů. Poznamenejme, že rozdělení na části provádíme na úrovni ISBN, nikoliv na úrovni jednotlivých záznamů. Nechceme totiž, abychom na jednom záznamu model naučili a na druhém záznamu o téže knize jej validovali. Skutečnost, že pro různé knihy máme různé počty záznamů, vede k tomu, že jednotlivé části budou stejně velké co do počtu unikátních ISBN, ale můžou se lišit v počtu záznamů.

Do tréninku vstupuje hned na několika místech náhoda – inicializace vah v síti, promíchání dat apod. Za účelem eliminace náhodně dobrých či špatných výsledků zopakujeme celou krosvalidaci vždy 5krát. To nám dá mimo jiné dostatek dat na to, abychom mohli posoudit statistickou signifikanci rozdílu mezi různými modely pomocí t-testu. Výjimkou bude naivní bayesovský klasifikátor, který je deterministický, a není proto důvod jej trénovat na každé množině vzorků vícekrát.

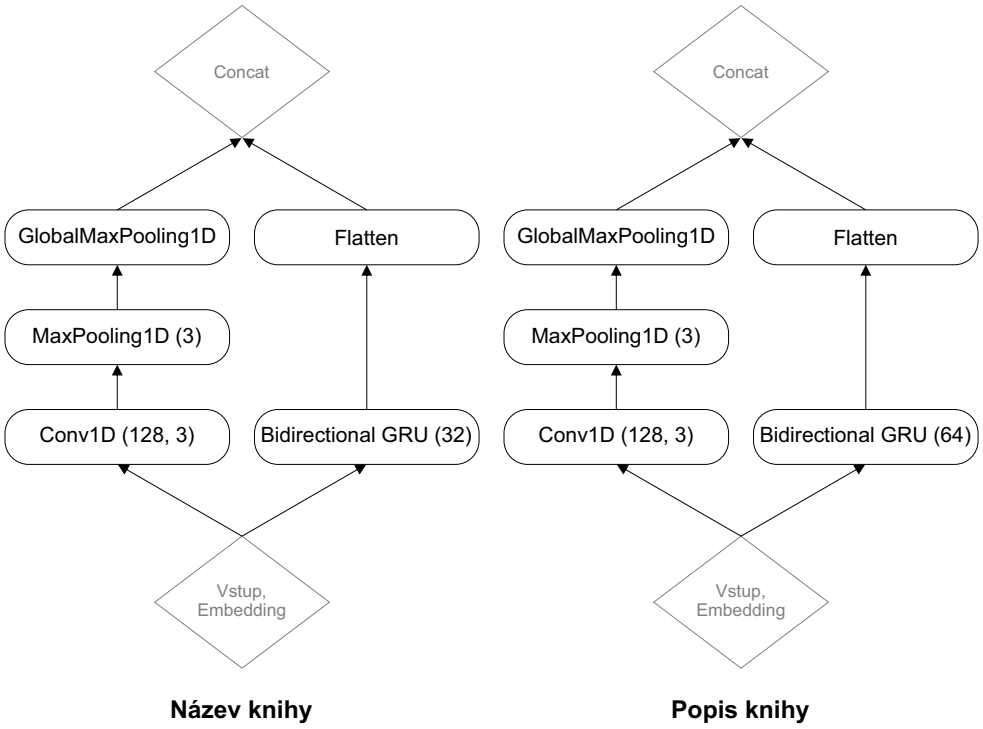
Co se týče počtu různých modelů, budeme mít celkem 18 hlavních konvolučních modelů a půjde o všechny možné kombinace těchto tří parametrů:

- Úloha (rozpoznání učebnic, určení předmětu učebnice a určení úrovně učebnice)



Obrázek 4.4: Hluboká konvoluční a rekurentní síť

- Typ slov (tvary nebo lemma)
- Embeddingy (trénované se sítí, natrénované na neanotovaných datech a předtrénované)



Obrázek 4.5: Duální konvoluční a rekurentní síť

5. Výsledky

V této kapitole uvedeme výsledné úspěšnosti všech modelů, pro které jsme prováděli krosvalidaci a podíváme se, jak zvolené embeddingy a typ slov (tvary či lemma) ovlivňují výkonnost modelu. Vybereme nejlepší modely a otestujeme je na testovací sadě dat, přičemž vyzkoušíme kromě konvolučních sítí i další alternativní architektury. Nakonec rozebereme fungování a chyby nejlepšího modelu pro každou z úloh.

5.1 Naivní Bayes

Jako první jsme natrénovali naivní bayesovský klasifikátor. Kromě hlavního modelu zahrnujícího všechny dostupné parametry (název, autor, nakladatel, popis), jsme trénovali také modely, které měly jako vstup vždy jen jeden z těchto parametrů. Dále jsme vyzkoušeli pro všechny modely jak variantu používající tvary slov, tak variantu používající lemmata. V odstavcích a tabulkách níže uvádíme výsledné úspěšnosti pro jednotlivé úlohy. Používali jsme 5násobnou kros-validaci, proto je v tabulce vždy průměr z pěti úspěšností a směrodatná odchylka. Naivní Bayes je deterministický model a nedává tak smysl opakovat trénování stejného modelu několikrát, jako jsme to prováděli u neuronových sítí (viz níže). Máme tedy pro každý model jen 5 výsledků (z krosvalidace), což není dostačující pro provedení t-testu a odhad statistické signifikance rozdílu mezi dvěma modely. Z tohoto důvodu u naivního bayesovského klasifikátoru test signifikance dělat nebudeme.

Rozpoznávání učebnic V této úloze, jak lze vidět v tabulce 5.1, měl z jednotlivých parametrů nejvyšší úspěšnost popis knihy, těsně následovaný názvem. O moc horší nebyla ani úspěšnost klasifikace pouze podle autora. Naopak model využívající jen nakladatele nebyl prakticky lepší než model, který by klasifikoval vše jako učebnici. Hlavní model trénovaný se všemi parametry pak předčil ostatní modely a o něco lépe dopadla jeho varianta využívající lemmata.

Parametr	**Typ slov**	**Úspěšnost**	**Odchylka**
Všechny	tvary	91,99 %	0,95 %
	lemma	92,10 %	1,10 %
Název knihy	tvary	90,38 %	0,49 %
	lemma	91,11 %	0,74 %
Autor	-	86,10 %	0,25 %
Nakladatel	-	51,95 %	0,74 %
Popis knihy	tvary	91,27 %	1,31 %
	lemma	91,25 %	1,22 %

Tabulka 5.1: Výsledky naivního bayese pro rozpoznávání učebnic

Klasifikace předmětu V úloze klasifikace předmětu si modely příliš dobře nevedly, když nedosáhly úspěšnosti ani 70 % (viz tabulka 5.2). Nejlepší výsledek měl model využívající pouze název knihy. Modely se všemi parametry a pouze s autorem však dosáhly podobných úspěšností. Stejně jako v předchozí úloze se ani zde neosvědčil model využívající pouze nakladatele.

Parametr	Typ slov	Úspěšnost	Odchylka
Všechny	tvary	67,37 %	1,73 %
	lemma	68,89 %	1,44 %
Název knihy	tvary	68,65 %	1,37 %
	lemma	68,95 %	1,68 %
Autor	-	67,66 %	1,59 %
Nakladatel	-	20,30 %	0,67 %
Popis knihy	tvary	64,07 %	2,71 %
	lemma	65,73 %	1,92 %

Tabulka 5.2: Výsledky naivního bayese pro klasifikaci předmětu

Klasifikace úrovně I v této úloze měly výsledky podobný charakter jako v předchozích dvou, jak ukazuje tabulka 5.3. Za zmínku stojí, že u modelu klasifikujícího podle popisu vyšla lepší úspěšnost u varianty s tvary slov než u varianty s lemmaty. To zřejmě ovlivnilo hlavní model, kde se vyskytuje tento jev také. Rozdíly jsou však malé a jednotlivé úspěšnosti mají velký rozptyl, může se tedy jednat jen o statistickou chybu.

Parametr	Typ slov	Úspěšnost	Odchylka
Všechny	tvary	83,88 %	0,71 %
	lemma	83,42 %	1,17 %
Název knihy	tvary	85,36 %	0,34 %
	lemma	85,52 %	0,93 %
Autor	-	73,17 %	1,40 %
Nakladatel	-	37,97 %	0,52 %
Popis knihy	tvary	79,49 %	1,34 %
	lemma	78,81 %	1,76 %

Tabulka 5.3: Výsledky naivního bayese pro klasifikaci úrovně

Shrnutí Nejlepšího výsledku dosáhl naivní bayesovský klasifikátor u úlohy rozpoznávání učebnic, a to přes 92 %. U ostatních úloh by zřejmě nebyl kvůli vysoké chybovosti v praxi využitelný. Z úspěšností dosažených modely využívajícími pouze jeden vstupní parametr jsme zjistili, že největší informační hodnotu mají (alespoň pro pravděpodobnostní model) textové parametry název a popis knihy a že modely fungují lépe, když jsou tyto parametry lemmatizovány. Téměř stejnou informační hodnotu má však i autor knihy, naopak pouze podle nakladatele model nedokázal klasifikovat vzorky prakticky vůbec, a to ani v jedné z úloh.

5.2 Hluboké sítě

V této sekci uvedeme validační úspěšnost modelů pro jednotlivé úlohy. Jak bylo zmíněno v kapitole 4.3.4, používali jsme pro tyto modely konvoluční architekturu sítě a všechny modely jsme trénovali 25krát (5krát 5násobná krosvalidace). V následujících tabulkách proto jako validační úspěšnost uvádíme průměrnou úspěšnost na validační části dat z těchto 25 tréninků a její směrodatnou odchylku.

Abychom zjistili, zda rozdíly mezi modely nejsou způsobené jen statistickou chybou a jestli je skutečně jeden model lepší než druhý, budeme zkoumat statistickou signifikanci těchto rozdílů. V každé úloze vybereme nejlepší model podle průměrné úspěšnosti a provedeme pro něj párový t-test (viz kapitola 2.4.2) s každým z ostatních modelů. Jako nulovou hypotézu stanovíme, že oba modely mají stejnou střední hodnotu úspěšnosti, tedy že jeden není významně lepší než druhý. Testovat budeme na hladině významnosti $\alpha = 0{,}05$ a nulovou hypotézu tak zamítneme, pokud nám v testu vyjde p-hodnota (signifikance) menší než 0,05. V takovém případě můžeme říct, že model s nižší úspěšností je skutečně statisticky významně horší než nejlepší model. V tabulkách označujeme úspěšnost modelů, které nejsou statisticky signifikantně horší než model s nejvyšší úspěšností, dvěma hvězdičkami.

Rozpoznávání učebnic Jednalo se o binární klasifikaci, jako trénovací data jsme použili celou trénovací množinu. Počet trénovacích epoch pro jednotlivé modely je uveden v tabulce 5.4. Tabulka 5.5 potom shrnuje úspěšnost modelů.

Použité embeddingy	**Počet epoch**
Žádné	2
Vlastní	3
Stažené	1

Tabulka 5.4: Počet epoch modelů pro rozpoznávání učebnic

Použité embeddingy	**Typ slov**	**Úspěšnost**	**Odchylka**
Žádné	Tvary	**94,90 %**[1]	0,87 %
	Lemma	94,90 %**	0,54 %
Vlastní	Tvary	94,04 %	0,70 %
	Lemma	94,53 %	0,74 %
Stažené	Tvary	94,74 %**	1,00 %
	Lemma	94,82 %**	0,81 %

Pozn: ** Modely, které nebyly statisticky signifikantně horší než nejlepší model.

Pozn: [1] Před zaokrouhlením byl tento model nepatrně lepší než model s lemmaty.

Tabulka 5.5: Validační úspěšnost rozpoznávání učebnic

U této úlohy dosáhly všechny modely podobné úspěšnosti, avšak modely s vlastními embeddingy jsou přesto statisticky signifikantně horší. Mezi modely

bez embeddingů a modely se staženými embeddingy naopak není významný rozdíl, ať už používají tvary slov či lemmata.

Klasifikace předmětu učebnice Jednalo se o klasifikaci do jedné z N tříd, jako trénovací/validační data jsme použili z trénovací množiny pouze vzorky anotované jako učebnice. Počet trénovacích epoch pro jednotlivé modely je uveden v tabulce 5.6. Tabulka 5.7 shrnuje výsledky všech modelů.

Použité embeddingy	**Počet epoch**
Žádné	6
Vlastní	10
Stažené	7

Tabulka 5.6: Počet epoch modelů pro klasifikaci předmětu učebnice

Použité embeddingy	**Typ slov**	**Úspěšnost**	**Odchylka**
Žádné	Tvary	94,35 %	0,96 %
	Lemma	**95,55 %**	0,69 %
Vlastní	Tvary	89,19 %	0,96 %
	Lemma	92,20 %	1,10 %
Stažené	Tvary	95,18 %	0,73 %
	Lemma	95,22 %[1]	0,78 %

Pozn: [1] *p*hodnota vyšla v tomto případě 0,036.

Tabulka 5.7: Validační úspěšnost klasifikace předmětu učebnice

Pro klasifikaci učebnic do předmětů se ukázal být nejlepším model bez předtrénovaných embeddingů používající lemmata. Ostatní modely byly statisticky významně horší. Model se staženými embeddingy a lemmaty však nebyl od nejlepšího modelu daleko – nulovou hypotézu, že mezi nimi není významný rozdíl, jsme sice na námi zvolené hladině významnosti zamítli, ale *p*hodnota 0,036 značí, že s pravděpodobností 3,6 % jsme ji zamítli chybně.

Klasifikace úrovně učebnice Jednalo se o klasifikaci do jedné z N tříd, jako trénovací/validační data jsme použili z trénovací množiny pouze vzorky anotované jako učebnice. Počet trénovacích epoch pro jednotlivé modely je uveden v tabulce 5.8. Tabulka 5.9 shrnuje výsledky všech modelů.

Použité embeddingy	**Počet epoch**
Žádné	4
Vlastní	5
Stažené	5

Tabulka 5.8: Počet epoch modelů pro klasifikaci úrovně učebnice

Použité embeddingy	Typ slov	Úspěšnost	Odchylka
Žádné	Tvary	93,46 %	0,83 %
	Lemma	**93,76 %**	0,70 %
Vlastní	Tvary	89,58 %	0,88 %
	Lemma	90,21 %	0,90 %
Stažené	Tvary	89,53 %	0,78 %
	Lemma	89,33 %	0,60 %

Tabulka 5.9: Validační úspěšnost klasifikace úrovně učebnice

Stejně jako v předchozí úloze byl nejúspěšnějším modelem ten bez předtrénovaných embeddingů využívající lemmata. Všechny ostatní modely byly statisticky signifikantně horší. Zajímavostí jsou výsledky modelů se staženými embeddingy, které zde narozdíl od ostatních úloh za nejlepším modelem zaostávaly, a dokonce měly horší úspěšnost než modely s vlastními embeddingy. Tento výsledek by mohl souviset s naším zjištěním, že stažené embeddingy neobsahují vektory pro čísla. Ta však mohou být u klasifikace úrovní klíčová („Matematika pro 1. ročník ZŠ“ vs. „Matematika pro 9. ročník ZŠ“).

Shrnutí Z výše uvedeného můžeme vyvodit, že je výhodnější používat lemmata než tvary slov. Vlastní předtrénované embeddingy se nám příliš neosvědčily a nejlépe vyšla možnost, kdy embeddingy natrénovala sama síť. Mnohdy však rozdíl oproti staženým předtrénovaným embeddingům nebyl statisticky signifikantní.

5.3 Úspěšnost klasifikátorů na testovacích datech

Pro porovnání modelů na testovacích datech jsme na základě výsledků z předchozí sekce vybrali variantu používající lemma, a to ve dvou verzích – bez předtrénovaných embeddingů a se staženými embeddingy. Tyto dvě verze natrénujeme pro každou ze čtyř architektur (viz kapitola 4.3.3) a společně s baseline klasifikátory (naivním bayesem pro tvary slov a pro lemmata) budeme u každé z úloh testovat právě 10 modelů. Trénovat modely nyní budeme na celé trénovací množině, nebudeme již používat žádnou její část jako validační data. Každý z modelů natrénujeme pouze jednou, nebudeme proto už provádět test statistické signifikance a modely tak budeme vyhodnocovat na základě jejich úspěšnosti na testovacích datech.

V tabulce 5.10 lze vidět souhrn úspěšností pro všechny testované modely a všechny úlohy. Testovat budeme tyto modely:

- Naivní bayesovský klasifikátor, oba typy slov
- Všechny architektury neuronových sítí, bez předtrénovaných embeddingů, typ slov lemma
- Všechny architektury neuronových sítí, se staženými vícejazyčnými embeddingy, typ slov lemma

Model	Učebnice	Předmět	Úroveň
Bayes (tvary)	91,77 %	69,47 %	84,85 %
Bayes (lemma)	91,86 %	71,20 %	85,07 %
CNN (bez emb.)	95,11 %	96,02 %	93,06 %
CNN (s emb.)	94,61 %	95,76 %	90,37 %
RNN (bez emb.)	95,24 %	94,13 %	91,56 %
RNN (s emb.)	91,91 %	92,70 %	85,21 %
Deep CNN-RNN (bez emb.)	95,30 %	94,28 %	91,96 %
Deep CNN-RNN (s emb.)	93,65 %	94,70 %	88,22 %
Dual CNN-RNN (bez emb.)	**95,57 %**	**96,14 %**	**93,30 %**
Dual CNN-RNN (s emb.)	94,49 %	95,78 %	89,70 %

Tabulka 5.10: Úspěšnost modelů na testovacích datech

Z tabulky vyplývá, že konvoluční sítě se ukázaly být poměrně dobrou volbou pro všechny úlohy – přestože duální kombinovaná síť je překonává ve všech úlohách, rozdíly jsou pouze v řádech desetin procent. Kombinovaná síť je také složitější, protože obsahuje vše co konvoluční síť a navíc ještě rekurentní větev u názvu a popisu knihy. Zejména na úlohách klasifikace předmětů a úrovní je pak ze srovnání úspěšnosti CNN a RNN vidět, že duální síť těží spíše ze své konvoluční části. Tento úspěch konvolučních sítí může být dán charakterem úloh, zejména tím, že nejdůležitější parametr – název knihy – obsahuje málo slov na to, aby se zde využily přednosti rekurentních sítí.

Dále se potvrdilo, že embeddingy trénované společně se sítí si vedou lépe, než ty předtrénované, přestože jsme měli k dispozici vícejazyčné embeddingy, předtrénované na velké množině dat.

Obecně lze říci, že úspěšnost odhadnutá pomocí $k$násobné validace na trénovacích datech se potvrdila i na „neznámých“ testovacích datech. Velkou roli v tom ale může hrát skutečnost, že trénovací i testovací data jsme vybírali ze stejné množiny dat. Mírné zlepšení úspěšnosti oproti validaci je pak dáno větším počtem trénovacích vzorků – pro trénování jsme použili všechna trénovací data, namísto oddělení jedné části (cca jedné pětiny) jako validační množiny.

V následujících sekcích rozebereme detailně výsledky a chyby nejlepšího modelu pro danou úlohu, porovnáme chyby tohoto modelu s ostatními architekturami neuronových sítí a také s „chybami“ druhého anotátora.

5.3.1 Typy chyb

V této části se podrobněji podíváme na to, jaké chyby dělá na testovacích datech model s nejvyšší úspěšností – tedy ve všech úlohách duální CNN-RNN síť bez předtrénovaných embeddingů, používající lemmata slov. Pro každou třídu dané úlohy uvedeme precision (kolik procent záznamů klasifikovaných modelem jako daná třída patřilo skutečně do dané třídy), recall (kolik procent záznamů patřící do dané třídy model správně klasifikoval) a F_1 skóre, které se vypočítá následovně:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{5.1}$$

Dále pro každou úlohu uvedeme tzv. matici záměn (angl. confusion matrix), která vizualizuje chyby modelu. Konkrétně řádky matice reprezentují skutečné třídy a sloupce matice pak predikce modelu. Číslo v buňce pak udává počet záznamů, které ve skutečnosti patří do třídy daného řádku, ale model je zařadil do třídy příslušného sloupce.

Rozpoznávání učebnic U této úlohy byly obě třídy poměrně vyvážené a stejně tak predikce algoritmu vykazovaly chyby v podobném množství na obě strany, jak lze vidět v tabulce 5.11. Matice záměn je pak v tabulce 5.12.

	Počet	**Precision**	**Recall**	**F_1 skóre**
Je učebnice	7 361	96,39 %	95,01 %	95,70 %
Není učebnice	6 835	94,71 %	96,17 %	95,43 %

Tabulka 5.11: Precision, recall a F_1 skóre pro úlohu rozpoznávání učebnic

	Není učebnice	**Je učebnice**
Není učebnice	6 573	262
Je učebnice	367	6 694

Tabulka 5.12: Matice záměn pro rozpoznávání učebnic

Klasifikace předmětů Tabulka 5.13 shrnuje metriky pro všechny klasifikované předměty. Nejlépe dopadl algoritmus u hudební výchovy, kde zařadil všechny učebnice správně. Vzhledem k malé velikosti této třídy to však není statisticky významná informace. Zajímavější situace je u angličtiny, která je druhou největší třídou a algoritmus zde zaznamenal recall 100 % při velmi solidní přesnosti 99,35 %. Jak vyplývá z matice záměn (tabulka 5.14), algoritmus klasifikoval jako angličtinu pouze 9 učebnic jiných jazyků, včetně 3 učebnic češtiny.

Problémy měl algoritmus především s kategoriemi Ostatní a Ostatní jazyky. Zde lze nízkou úspěšnost vysvětlit tím, že jsou tyto třídy těžko rozpoznatelné, neboť tvoří jakési doplňky ke zbytku tříd. Úplně nejhorší úspěšnost měl předmět Stavitelství, ale opět jde o malou třídu a navíc většina záměn byla s „příbuzným“ předmětem Technika a strojírenství a s třídou Ostatní.

Klasifikace úrovní U této úlohy byly výsledky poměrně vyrovnané, až na úroveň Jazyky (učebnice pro jazykové školy a samostudium), která dopadla jasně nejhůře (viz tabulka 5.15). Je pravděpodobné, že se zde projevil stejný problém s určováním úrovně jazykových učebnic jako při anotaci (viz část 3.4.3). Učebnice, které mají jako anotaci předmětu některý cizí jazyk, nebo kategorii „Ostatní jazyky“ (celkem 2 410 v testovacích datech), měly úspěšnost klasifikace 88,30 %,

Předmět	Počet	Precision	Recall	F_1 skóre
Angličtina	1 373	99,35 %	100,00 %	99,67 %
Biologie	224	93,53 %	96,88 %	95,18 %
Chemie	135	98,48 %	96,30 %	97,38 %
Český jazyk	1 586	98,47 %	97,67 %	98,07 %
Dějepis	289	99,30 %	98,27 %	98,78 %
Ekonomie a účetnictví	89	98,85 %	96,63 %	97,72 %
Francouzština	196	95,10 %	98,98 %	97,00 %
Fyzika	273	98,83 %	93,04 %	95,85 %
Hudební výchova	39	100,00 %	100,00 %	100,00 %
Italština	29	86,21 %	86,21 %	86,21 %
Informatika	49	95,92 %	95,92 %	95,92 %
Matematika	1 116	97,18 %	98,66 %	97,91 %
Medicína	109	91,92 %	83,49 %	87,50 %
Němčina	504	99,20 %	98,61 %	98,91 %
Ostatní jazyky	61	63,22 %	90,16 %	74,32 %
Ostatní	244	82,64 %	81,97 %	82,30 %
Prvouka	95	95,74 %	94,74 %	95,24 %
Ruština	145	100,00 %	94,48 %	97,16 %
Španělština	102	90,82 %	87,25 %	89,00 %
Stavitelství	36	57,45 %	75,00 %	65,06 %
Technika a strojírenství	139	96,77 %	86,33 %	91,25 %
Vlastivěda	96	97,53 %	82,29 %	89,27 %
Výtvarná výchova	7	100,00 %	45,86 %	60,00 %
Zeměpis	166	90,96 %	90,96 %	90,96 %
Základy spol. věd	259	85,66 %	92,28 %	88,85 %

Tabulka 5.13: Precision, recall a F_1 skóre pro úlohu klasifikace předmětů

naproti tomu u ostatních učebnic (celkem 4 951) dosáhl algoritmus úspěšnosti 95,74 %.

Z matice záměn vyplývá, že chyby byly mezi všemi úrovněmi navzájem (až na autoškolu, která však byla velmi malou třídou). Přesto lze zaznamenat větší množství záměn mezi „sousedními“ úrovněmi – 1. stupeň ZŠ vs. 2. stupeň ZŠ, 2. stupeň ZŠ vs. střední škola, ale i střední škola vs. jazykové školy (zde pochopitelně také vlivem jazykových učebnic, viz předchozí odstavec).

5.3.2 Rozbor vzorku chyb

Z predikcí nejlepších modelů pro každou úlohu jsme vybrali náhodný vzorek 100 chyb, který jsme ručně prošli a zhodnotili, jakých chyb se klasifikátory dopouští.

Rozpoznávání učebnic Ze vzorku 100 chyb byla ve 45 případech ne-učebnice chybně predikována jako učebnice. Dalo by se říct, že všechny knihy, až na jednu výjimku, se týkaly vzdělávání, kromě jednoho případu se tedy nevyskytla žádná zjevná chyba ve smyslu románu klasifikovaného jako učebnice apod. Konkrétněji,

	AJ	BI	CJ	CH	DE	EK	FJ	FY	HU	IT	IJ	MA	ME	NJ	OS	OJ	PR	RJ	SJ	ST	TE	VL	VY	ZS	ZE
AJ	1 373	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BI	0	217	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	4	0
CJ	3	9	1 549	0	0	0	1	0	0	2	0	2	1	2	4	7	0	0	0	0	0	0	0	3	3
CH	0	0	0	130	0	0	0	0	0	0	0	2	0	0	0	2	0	0	0	0	0	0	0	1	0
DE	0	0	0	0	284	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	4
EK	0	0	0	0	0	86	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0
FJ	0	0	0	0	0	0	194	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
FY	0	0	0	0	0	1	5	254	0	0	0	10	0	0	1	2	0	0	0	0	0	0	0	0	0
HU	0	0	0	0	0	0	0	0	39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IT	0	0	0	0	0	0	2	0	0	47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IJ	1	0	1	0	0	0	0	0	0	0	25	0	0	0	0	0	0	0	2	0	0	0	0	0	0
MA	0	0	4	0	0	0	0	0	0	0	0	1 101	0	0	1	1	0	0	1	0	0	0	0	8	0
ME	0	0	1	0	0	0	0	0	0	0	0	0	91	0	12	5	0	0	0	0	0	0	0	0	0
NJ	1	0	2	0	0	0	0	0	0	0	0	0	0	497	1	2	0	0	1	0	0	0	0	0	0
OS	0	3	1	0	0	0	0	3	0	0	0	9	1	0	200	4	0	0	1	10	0	0	0	12	0
OJ	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	55	0	0	1	0	0	0	0	0	0
PR	0	1	0	0	0	0	0	0	0	0	0	4	0	0	0	0	90	0	0	0	0	0	0	0	0
RJ	1	0	3	0	0	0	0	0	0	0	0	0	0	1	0	1	0	137	2	0	0	0	0	0	0
SJ	3	0	1	0	0	0	2	0	0	0	4	0	0	0	0	3	0	0	89	0	0	0	0	0	0
ST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	27	4	0	0	0	0
TE	0	0	0	2	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	7	120	0	0	0	0
VL	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	79	0	4	8
VY	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	3	0	0
ZS	0	1	5	0	0	0	0	0	0	0	0	0	6	0	3	2	0	0	0	3	0	0	0	239	0
ZE	0	1	0	0	2	0	0	0	0	0	0	4	0	0	0	1	0	0	0	0	0	0	0	7	151

Tabulka 5.14: Matice záměn pro klasifikaci předmětů

Úroveň	**Počet**	**Precision**	**Recall**	**F_1 skóre**
1. stupeň ZŠ	1861	96,03 %	96,29 %	96,16 %
2. stupeň ZS	1782	91,46 %	93,21 %	92,33 %
Střední škola	2759	93,35 %	94,67 %	94,01 %
Jazyky	920	90,89 %	83,48 %	87,03 %
Autoškola	39	97,22 %	89,74 %	93,33 %

Tabulka 5.15: Precision, recall a F_1 skóre pro úlohu klasifikace úrovní

téměř polovina z chyb byly učebnice pro vysoké školy nebo naopak předškoláky, zbytek tvořily knihy populárně naučné, odborné, zjednodušená četba, metodické pokyny pro učitele (nepatřící k žádné konkrétní učebnici), knihy pohádek a říkanek a také 3 záznamy, které při zpětném pohledu byly chybně anotovány. Z 55 učebnic, které algoritmus vyhodnotil jako ne-učebnice, pak bylo „zřejmých“ jen cca pětina učebnic, zbytek byly učebnice, které jsou na hraně definice. Na základě vyhodnoceného vzorku tak můžeme říct, že model funguje spolehlivě a problémy mu dělají zejména hraniční případy mezi učebnicí a ne-učebnicí.

Klasifikace předmětu U předmětů byla chybná anotace v 8 případech. Necelé dvě třetiny chyb klasifikátoru jsou zřejmé, kdy je předmět určen jednoznačně špatně. Zbytek tvoří učebnice na hranici dvou předmětů, učebnice, kde by se dal predikovaný předmět obhájit, případně kde i chybný předmět dává určitý smysl. Domníváme se, že v mnoha případech je chyba způsobena určitými matoucími slovy, zejména v názvu knihy. Například:

- Communicative Czech – Intermediate Czech (angličtina místo češtiny – protože jsou v názvu i popisu knihy anglická slova)

	1. st. ZŠ	2. st. ZŠ	SŠ	Jazyky	Auto
1. st. ZŠ	1 792	49	19	1	0
2. st. ZŠ	32	1 661	67	21	1
SŠ	19	73	2 612	55	0
Jazyky	22	32	98	768	0
Auto	1	1	2	0	35

Tabulka 5.16: Matice záměn pro klasifikaci úrovní

- Stroje a zařízení v **chemickém** průmyslu (chemie místo techniky a strojírenství)
- Poznáváme svět **v číslech** (matematika místo zeměpisu)
- Velký psychologický **slovník** (ostatní jazyky místo základů spol. věd)

Klasifikce úrovně Velkou část chyb u klasifikace úrovně tvořily jazykové učebnice. Z chyb u ostatních předmětů šlo jednak o chybné anotace, ale také o učebnice, u kterých na základě informací dostupných klasifikačnímu algoritmu nelze úroveň jednoznačně určit (např. učebnice dějepisu či matematiky).

Ve vzorku se nevyskytla ani jedna chyba, kde by v názvu učebnice byl uveden ročník, pro který je určena (např. „Český jazyk pro 5. ročník“ apod.). Z toho můžeme soudit, že model se naučil spolehlivě určovat úroveň na základě čísel v názvech knih.

5.3.3 Podobnost chyb modelů různých architektur

Pokusili jsme se porovnat podobnost chyb modelů neuronových sítí, které se liší jen architekturou sítě (viz část 4.3.3). Pro každou architekturu jsme vybrali model, který používá lemmata a embeddingy trénuje zároveň se sítí, protože tato kombinace měla obecně nejvyšší úspěšnost. Výsledky pro jednotlivé úlohy shrnuje tabulka 5.17.

	Učebnice	Předmět	Úroveň
Všechny chybně	2,51 %	2,05 %	4,20 %
Všechny chybně a stejně	2,51 %	0,88 %	3,30 %
Všechny modely správně	92,76 %	91,55 %	88,41 %
Úspěšnost nejlepšího modelu	95,57 %	96,14 %	93,30 %
Úspěšnost nejhoršího modelu	95,11 %	94,13 %	91,56 %

Tabulka 5.17: Podobnost chyb modelů různých architektur

První řádek tabulky udává procento záznamů, které všechny vybrané modely predikovaly chybně. Může tedy jít o „těžké“ záznamy, chyby v anotaci, případně záznamy s nedostatkem dat pro spolehlivou predikci.

Druhý řádek pak říká, kolik záznamů predikovaly všechny vybrané modely nejen chybně, ale kde se také modely shodly na stejné predikci. Jinými slovy,

kde udělaly všechny vybrané modely stejnou chybu. Jedná se tedy o podmnožinu záznamů z prvního řádku tabulky. V tomto případě půjde pravděpodobně o záznamy s „matoucí“ informací či s chybnou anotací.

Třetí řádek naopak udává procento záznamů, které všechny vybrané modely vyhodnotily správně.

Pro referenci jsme do tabulky přidali také úspěšnost nejlepšího a nejhoršího z vybraných modelů v rámci každé z úloh. Lze z toho vyvodit například to, že nejlepší model klasifikoval v úloze rozpoznávání učebnic chybně 4,43 % záznamů (doplněk do sta k úspěšnosti 95,57 %), avšak u necelé poloviny z toho (4,43 - 2,51 = 1,92 % záznamů) predikoval některý model s jinou architekturou správný výsledek. Nabízí se tak možnost, že úspěšnost nejlepšího modelu by mohla jít ještě o tato necelá dvě procenta vylepšit. Podobně je tomu u klasifikace předmětů (100 - 96,14 - 2,05 = 1,81 %) i u klasifikace úrovní (100 - 93,30 - 4,20 = 2,50 %).

Dále stojí za zmínku 3,30 % záznamů, u kterých se všechny architektury spletly a přitom zařadily učebnici do stejné úrovně. Jde o poměrně vysoké číslo, které by mohlo značit systémovou chybu v tom, jak se algoritmy naučily reprezentaci dat.

5.3.4 Podobnost chyb klasifikátoru a druhého anotátora

Při pohledu na tabulku mezianotátorské shody 3.5 a tabulku výsledků klasifikátorů testovaných na testovacích datech 5.10 lze pozorovat jistou podobnost hodnot shody anotátorů a úspěšnosti klasifikátorů. Rozhodli jsme se proto zjistit, zda druhý anotátor (anotátor B) nedělá podobné „chyby“ jako klasifikační algoritmus. V tabulce 5.18 je pro každou úlohu uvedeno, u kolika procent záznamů chybně klasifikovaných algoritmem se navzájem rozcházely i anotace obou anotátorů pro danou knihu. Pro toto měření byl zvolen model, který dosahoval nejvyšší úspěšnosti, tedy u všech úloh duální CNN-RNN síť bez předtrénovaných embeddingů, využívající lemmata slov.

	Učebnice	**Předmět**	**Úroveň**
Procent stejných chyb	23,05 %	20,77 %	23,73 %

Tabulka 5.18: Podobnost chyb klasifikátoru a druhého anotátora

Jak můžeme vidět, procenta jsou poměrně nízká, mezi 20 % a 24 %, což znamená, že druhý anotátor a algoritmus dělali spíše odlišné chyby. Nejspíš tedy platí, že člověk a neuronová síť se na data dívají jinak a anotace jednoduchá pro člověka může být pro počítač obtížná a naopak.

Závěr

V této práci jsme se zabývali problémem automatického rozpoznávání učebnic a jejich klasifikací podle předmětu a úrovně. Nejprve jsme sestavili vlastní definici učebnice a definovali jednotlivé úlohy, přičemž jsme stanovili, do kterých předmětů a úrovní budeme učebnice dělit. Rozhodli jsme také, že jako vstupní parametry v našich úlohách budeme využívat název učebnice, její popis, autora a nakladatele.

Jako zdroj dat jsme zvolili veřejně dostupné zdroje informací o knihách, a sice internetové databáze knih a detaily produktů na internetových knihkupectvích. Potřebné údaje jsme extrahovali z HTML souborů a vyčistili, pro autory a nakladatele jsme vytvořili algoritmy k převodu jejich textové reprezentace na ID z rejstříku všech autorů/nakladatelů nacházejících se v datech. Podmnožinu dat jsme ručně anotovali dvěma anotátory.

Nakonec jsme natrénovali pro jednotlivé úlohy několik modelů - naivní bayesovský klasifikátor jako baseline a pak různé varianty neuronových sítí. Nejlepší modely jsme otestovali na testovacích datech a analyzovali chyby, které dělají.

Diskuze výsledků

Ukázalo se, že stanovit přesnou definici učebnice, nebo určit hranice mezi jednotlivými předměty a úrovněmi, abychom mohli učebnice jednoznačně zařadit, je poměrně obtížný úkol. Na druhou stranu pokud vezmeme konzistentní anotace jednoho anotátora a natrénujeme konvoluční neuronové sítě, můžeme získat relativně dobrou úspěšnost pohybující se (dle konkrétní úlohy) kolem 93-96 %. Takové výsledky umožňují i využití algoritmů v praxi, například pro třídění produktů v e-shopech, což bylo i motivací této práce.

Možná zlepšení

Přestože jsou výsledky našich sítí velmi dobré, přichází v úvahu několik možných dalších vylepšení:

Více dat Výkonnost neuronových sítí, obzvláště těch hlubokých, je obvykle silně závislá na množství trénovacích dat. Náš anotovaný vzorek dat nebyl z největších, a proto si myslíme, že více dat by mohlo přispět k ještě lepší úspěšnosti modelů.

Využití obrázků Říká se, že obrázek často řekne více než tisíc slov. Bylo by proto zajímavé vyzkoušet zahrnout mezi údaje o knihách i obrázek jejího obalu a využít jej jako další vstup pro konvoluční sítě.

Optimalizace hyperparametrů Drobné zlepšení úspěšnosti by mohlo přinést pečlivé vyladění hyperparametrů sítí.

Seznam použité literatury

Bengio, Y., Simard, P. a Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, **5**(2), 157–166.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. *CoRR*, **abs/1206.5533**.

Borges, E. N., Becker, K., Heuser, C. A. a Galante, R. (2011). An automatic approach for duplicate bibliographic metadata identification using classification. In *2011 30th International Conference of the Chilean Computer Science Society*, pages 47–53.

Chung, J., Gülçehre, Ç., Cho, K. a Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, **abs/1412.3555**.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L. a Jégou, H. (2017). Word translation without parallel data. *CoRR*, **abs/1710.04087**.

Fellegi, I. P. a Sunter, A. B. (1969). A theory for record linkage. *Journal of the American Statistical Association*, **64**(328), 1183–1210.

Glorot, X. a Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W. a Titterington, M., editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

He, K., Zhang, X., Ren, S. a Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, **abs/1502.01852**.

Hochreiter, S. a Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, **9**(8), 1735–1780. ISSN 0899-7667.

Hořeňovská, K. (2019). An evaluation of Czech word embeddings. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 65–75, Turku, Finland, September–October 2019. Linköping University Electronic Press.

Jiang, Y., Lin, C., Meng, W., Yu, C., Cohen, A. M. a Smalheiser, N. R. (2014). Rule-based deduplication of article records from bibliographic databases. *Database : the journal of biological databases and curation*, **2014**, bat086.

Kingma, D. a Ba, J. (2014). Adam: A method for stochastic optimization. *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Kundu, C. a Zheng, L. (2020). Deep multi-modal networks for book genre classification based on its cover. *CoRR*, **abs/2011.07658**.

Mikolov, T., Chen, K., Corrado, G. a Dean, J. (2013). Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*, **2013**.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M. a Gao, J. (2021). Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv.*, **54**(3). ISSN 0360-0300.

Pennington, J., Socher, R. a Manning, C. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

Piasecki, M. a Walentynowicz, W. (2017). Morphodita-based tagger adapted to the polish language technology. In *8th Language and Technology Conference, LTC 2017, Poznań, Poland, November 17–19, 2017, Revised Selected Papers*.

Řehůřek, R. a Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

Riaz, A., Nabeel, M., Khan, M. a Jamil, H. (2020). Sbag: A hybrid deep learning model for large scale traffic speed prediction. *International Journal of Advanced Computer Science and Applications*, **11**, 287–291.

Sobkowicz, A., Kozłowski, M. a Buczkowski, P. (2018). Reading book by the cover—book genre detection using short descriptions. pages 439–448. ISBN 978-3-319-67791-0.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. a Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, **15**(56), 1929–1958.

Straka, M. (2017). Slovak MorphoDiTa models 170914. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Straka, M. a Straková, J. (2014). English models (morphium + WSJ) for MorphoDiTa. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Straková, J., Straka, M. a Hajič, J. (2014). Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 13–18, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Zhu, B., Yang, L., Wu, X. a Guo, T. (2015). Automatic recognition of books based on machine learning. In *2015 3rd International Symposium on Computational and Business Intelligence (ISCBI)*, pages 74–78.

Seznam obrázků

Seznam tabulek