VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ HISTORICKÝCH TEXTŮ POMOCÍ HLUBOKÝCH NEURONOVÝCH SÍTÍ

CONVOLUTIONAL NETWORKS FOR HISTORIC TEXT RECOGNITION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **NELA MACUROVÁ**
AUTHOR

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

BRNO 2018

Zadání bakalářské práce/20899/2017/xmacur07

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2017/2018

Zadání bakalářské práce

Řešitel: **Macurová Nela**
Obor: Informační technologie
Téma: **Rozpoznávání historických textů pomocí hlubokých neuronových sítí**
Convolutional Networks for Historic Text Recognition
Kategorie: Zpracování obrazu

Pokyny:

1. Prostudujte základy konvolučních sítí a rozpoznávání textu.
2. Vytvořte si přehled o současných metodách rozpoznávání textu pomocí konvolučních sítí.
3. Vyberte nebo navrhněte metodu aplikovatelnou na rozpoznávání historických textů.
4. Obstarejte si databázi vhodnou pro experimenty se zaměřením na gotická písma.
5. Implementujte navrženou metodu a proveďte experimenty nad datovou sadou.
6. Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje.
7. Vytvořte stručné video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Krizhevsky, A., Sutskever, I. and Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012
- http://www.image-net.org/challenges/LSVRC/2013/results.php

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Hradiš Michal, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 16. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
L.S. 612 00 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato práce se zabývá rozpoznáváním historických textů pomocí hlubokých neuronových sítí, konkrétně rozpoznáváním jednotlivých slov v gotickém písmu v českém jazyce. Je zde vytvořen obecný přehled o konvolučních sítích a metodách rozpoznávání textu. Byl vytvořen dataset, který se skládá z reálných i generovaných dat. Síť byla trénovná na generovaných datech a testována na reálných obrázcích slov. Tato navrhovaná metoda klasifikace slov, nebyla moc úspěšná, kvůli rozdílným testovacím a trénovacím datům.

Abstract

This thesis deals with the recognition of historical texts using deep neural networks, specifically the recognition of individual words in Gothic script in Czech. Here is a general overview of convolutional networks and text recognition methods. A dataset was created with real and generated data. The network was trained on generated data and testing on real images of words. This proposed word classification method was not very successful due to different test and training data.

Klíčová slova

Rozpoznávání slov, OCR, hluboká neuronová síť

Keywords

Word recognition, OCR, deep neural network

Citace

MACUROVÁ, Nela. *Rozpoznávání historických textů pomocí hlubokých neuronových sítí.* Brno, 2018. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Michal Hradiš, Ph.D.

Rozpoznávání historických textů pomocí hlubokých neuronových sítí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana Ing. Michala Hradiše, Ph.D. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Nela Macurová
17. května 2018

Poděkování

Tímto bych chtěla poděkovat mému vedoucímu práce Ing. Michalovi Hradišovi, PhD., za odborné vedení, ochotu, podporu, cenné rady a čas, který věnoval seminářům a mi při řešení této práce. Poděkování také patří Bc. Martinovi Kiššovi za poskytnutí generátoru historických textů v gotickém písmu. Dále bych chtěla poděkovat výpočetnímu centru MetaCentrum za poskytnutí přístupu k výpočetním a úložným zařízením. Tato práce vznikla za podpory projektů CERIT Scientific Cloud (LM2015085) a CESNET (LM2015042) financovaných z programu MŠMT Projekty velkých infrastruktur pro VaVaI.

Obsah

Kapitola 1

Úvod

Historické texty je dobré uchovat i jiným způsobem než jen uschováním v archívu. Protože tak stále může být možnost, že bude zničen, papír totiž není trvalá forma uchování textu. Hlavně také nastává problém, že se k němu nikdo nedostane, nebo jen omezené množství lidí. Proto je dobré uchovat dokumnety i v digitální podobě, která je dostupná odkudkoliv. Z toho důvodu jsou historické texty digitalizovány. Skenované stránky dokumentu však nejsou nejlepším formátem. Nelze v textu vyhledávat, kopírovat a editovat ho a také některé historické fonty jsou těžce čitelné, protože obsahují neobvyklé, netypické znaky. Pokud by si tedy chtěl někdo tento text číst, musí si vyhledat a dekódovat znaky, které jsou v dokumentu použity. Je tedy vhodné tento text přepsat do podoby, ve které leze editovat, kopírovat, vyhledávat a dále zpracovávat. Digitalizace historických textů je už dlouhou dobu otevřené téma a stále je aktuální.
Tato práce se zabývá rozpoznáváním tištěného historického textu v gotickém písmu, zaměřuje se na rozpoznávní izolovaných slov z vymezeného slovníku metodou hlubokých neuronových sítí. První kapitola se zabývá obecně rozpoznáváním textu a také současnými metodami rozpoznávání textu pomocí neuronových sítí. Jsou zde zmíněny různé metody, které se využívají k rozpoznávání. Tato kapitola se také zaměřuje na historické texty, na které se práce orientuje a probírá jejich specifikaci. Druhá kapitola popisuje obecně neuronové sítě a zmiňuje se také o konvolučních a rekurentních sítích. Které sice nejsou cílem práce, ale mohly by být cílem dalšího vývoje. Je zde vytvořen přehled o jednotlivých vrstvách a principech, které jsou sítěmi využívány. Tato kapitola tvoří obecný vhled do neuronových sítí. Ve třetí kapitole je popsáno moje řešení rozpoznávání slov a aplikování některých principů zmíněných v předešlých kapitolách. V další kapitole jsou popsány dosažené výsledky experimentu a možnosti budoucího vývoje.

Kapitola 2

Rozpoznávání textu

Optical Character Recognition (OCR) [32] neboli optické rozpoznávání znaků patří mezi úlohy spadající pod počítačové vidění. Počítačové vidění (Computer vision) se zabývá vývojem zařízení získávajících informace ze zachyceného obrazu. OCR je tedy technologie, která převádí obrázky obsahující tištěný nebo psaný text do formátů, ve kterých lze dále text upravovat. Všechna písmena vždy nemusí být rozpoznána správně, ať už je to kvůli špatné kvalitě nebo nestandardnímu formátu, je proto nutná korekce. Existují různé metody rozpoznávání jako je fuzzy logika a strojové učení, pod které spadají i neuronové sítě.

2.1 Proces rozpoznávání

Proces rozpoznání textu v obrázku se dělí do několika kroků: získání obrázku, předzpracování, rozpoznávání znaků a následné zpracování.

Získání obrázku

Je potřeba získat určitý formát obrázku. U metod OCR se jedná převážně o naskenovaný dokument, který má určité parametry, třeba maximální a minimální velikost písma, rozměry obrázku, font, formát a kvalita. Při získání obrazu může také dojít k určitému zkreslení, natoční textu, různé velikosti a kvalitě, někdy je potřeba obrázek oříznout, ale to už záleží na dané metodě. Proto je nutné tento vstupní obraz upravit a zpracovat do podoby, kdy je možné s ním nějak pracovat. Některé systémy jsou schopné vytvářet formátovaný výstup, který se téměř blíží originální stránce, včetně sloupců, tabulek a jiných netextových komponent.

Předzpracování

Předzpracování neboli preprosessing, je fáze v níž dochází ke zpracování a úpravě naskenované stránky či fotky s textem. Předzpracování zlepší kvalitu vstupu a tím i šanci na úspěšné rozpoznání. Obraz může být zkosený, nebo může mít hodně šumu, takže se používají různé algoritmy ke zlepšení kvality obrazu, odstranění šumu a tak podobně. Pokud obraz nebyl při skenování správně zarovnán, je nutné naklonit obrázek tak, aby byly řádky textu vodorovně. Dále může dojít k odstranění pozitivních a negativních skvrn na a vyhlazení okrajů. Důležitý krok je binarizace, protože nesprávná binarizace způsobí spoustu problémů. Tento proces převede obraz z barvy nebo stupně šedi na černobílou, to pomůže lépe rozeznat text od pozadí a tím i ovlivní kvalitu fáze rozpoznávání znaků. Také detekce a odstraňování

částí, je nutným krokem ke zlepšení analýzy rozložení stránky, k dosažení lepší kvality rozpoznávání podtrženého textu, ke zjišťování tabulek atd. Linky můžou být odstraněny pro následné rozpoznávání. Analýza rozložení stránky se také nazývá zónování, dochází k označení sloupce, odstavce a dalších částí jako samostatných bloků. Dalším krokem je detekce řádků a slov, kdy se stanoví výchozí hodnota pro tvary slov a znaků. Například neproporcionální písmo má pevně danou šířku a tak se segmentace provádí jednoduše srovnáním. Ale u znaků zase záleží na fontech a typu písma, což může být problémovým faktorem kvůli různým velikostem v jednom bloku a malým mezerám mezi slovy. Také některé znaky se mohou navzájem dotýkat nebo naopak jedno písmeno může být rozděleno do několika částí. Je proto potřeba detekovat takové případy a najít správnou pozici každého znaku. Může tedy docházet i k segmentaci nebo izolaci znaků. Předzpracování obrazu může zahrnovat operace jako otočení, měřítko, vyhlazení, binarizace, normalizace, zkosení textu a může brát v potaz aspekty jako pozadí, rozlišení, barvu, šum a další deformace textu.

Rozpoznávání znaků

Pro tuto fázi existují dva způsoby rozpoznávání. Jednou z nich je přizpůsobení vzoru (pattern matching), také známá pod názvem rozpoznávání vzoru (pattern recognition) spadající do strojového učení. Už podle názvu je zřejmé, že tato metoda tkví v porovnávání obrazu se vzorem, který je uložen ve stejném měřítku a je téměř totožný se vstupním glyfem. Vstuní znak musí být správně izolován od zbytku obrazu. Druhou metodou je extrakce příznaků, která tyto znaky rozkládá do funkcí jako jsou přímky, smyčky, směry a průsečíky. Tato metoda je výpočetně efektivnější, protože extrakce snižujií rozměry zobrazení, tyto funkce jsou porovnány s abstraktní vektorovou reprezentací znaku, což sníží tuto reprezentaci na jeden nebo více prototypů znaku. Pro srovnání vlastností obrazu s uloženými znaky glyfů se používají klasifikátory nejbližšího souseda, třeba algoritmus k-nearest neighbors, a vybere se nejbližší shoda. Často se v OCR využívá dvoustuňový průchod k rozpoznávání, kde druhý průchod používá tvar písmen s vysokou „důvěrou“ při prvním průchodu a tak v druhém průchodu lépe rozpozná zbývající písmena. Tomuto přístupu se říká adaptivní rozpoznávání a hodí se hlavně pro nekvalitní zkreslené dokumenty nebo pro neobvyklé fonty. Problémem rozpoznávání jednotlivých znaků může být nejednoznačný tvar daného znaku. Některé znaky můžou vypadat podobně, nebo stejně, takže určité jednotlivé znaky nelze s jistotou určit, pokud není znám kontext. Například rozpoznání obrazu znaků I, |, 1, l, konečný znak může být vybrán později při zpracování.

Následné zpracování

Následné zpracování neboli postprocessing je proces, ve kterém se zpracovávají již rozpoznané znaky a dochází ke zvýšování přesnosti. V téhle fázi dochází k úpravě a korekci textu. Kvalita rozpoznávání se může zlepšit pokud je výstup omezen předem danou slovní zásobu. Některé znaky jako 1 a I, C a G, O a 0 mohou vypadat velmi podobně a tak slovník může pomoci při rozhodování. Problém však může nastat pokud zpracovávaný dokument obsahuje slovo, které ve slovníku není. Také znalost gramatiky skenovaného jazyka může přispět ke zvýšení přesnosti. Pomocí gramatiky lze předpovědět, zda je další slovo sloveso nebo podstatné jméno a to pomůže vybrat správně slovo. Každý krok OCR je velmi důležitý. Celý proces rozpoznání selže, pokud některý krok nemůže daný obrázek správně zpracovat. Výstupní formát textu může být buď prostý text, nebo formátovaný, to záleží na systému. K dosažení lepších výsledků a kvality rozpoznání je také někdy možné nastavit systém pro

dané parametry, pokud je třeba zpracovat pouze jeden druh obrázků. Někdy je to jediný způsob, jak zlepšit kvalitu rozpoznávání.

Inteligentní rozpoznávání slov

Intelligent word recognition (IWR) [30] neboli inteligentní rozpoznávání slov je metoda rozpoznávání ručně psaných slov. IWR rozpoznává celá ručně psaná nebo tištěná slova nebo fráze místo rozpoznávání po znacích jako u OCR. Slova přiřadí odpovídajícímu slovníku, který je definovaný uživatelem a tím významně sníží chyby jednotlivých znaků. IWR se snaží zautomatizovat ručně psané dokumenty, které mohly být v minulosti zadávány pouze manuálně. Pro každé analyzované slovo systém rozdělí slova na řadu grafémů nebo dílčích písmen. Tyto různé křivky, tvary a čáry vytvářejí písmena a IWR používá tyto tvary a seskupení k výpočtu hodnoty ve spojení s daným slovem. Tato metoda se nasnaží nahradit metody ICR a OCR, které dobře fungují na tištěné dokumenty, ale je ideální pro zpracování dokumentů v reálném světě, které obsahují většinou těžko rozpoznatelné údaje.

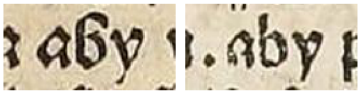
2.2 Historické texty

Tato kapitola zachycuje obecné informace o historicých textech, na které je tato práce zaměřena, ale i specifické informace týkající se zpracovávaných dokumentů. Historické texty se liší od normálních textů stářím, to se projevuje v mnoha aspektech. Jedním z nich je poškození materiálu, na kterém je text vytištěn, a také textu. Poškození papíru mohou být například fleky, zežloutnutí, natrhnutí, zmačkání a prosvítání nebo obtisk stran. Písmo může být vybledlé, špatně vytištěné, znaky se můžou slévat, mohou být různé mezery mezi písmeny, slovy a řádky. Dalším aspektem jsou odlišnosti fontů, typografie a pravidel uspořádání textu, to činí text těžko čitelným. Tato práce se zaměřuje hlavně na fonty gotického typu, které byly vydávány v 15.-19. století v češtině, jedná se převážně o ručně tištěné texty. Obrázek 2.2 slouží pro ilustraci různorodosti písem textů. Obecně historické texty obsahují velké množství šumu.

Písmo

Gotické písmo [31] je etapa ve vývoji latinky. Tato etapa se prolíná se středověkou gotikou, která je uměleckým slohem vrcholného středověku. Začala se projevovat na konci 12. století, v českých zemích nastupuje o něco později, ve střední Evropě a Německu trvala až do počátku 16. století. Gotické písmo je ostře lomené a postupně se nahradilo lomeným novogotickým písmem, to je označení skupiny novověkých latinských písem mezi něž patří fraktura, kanzlei a kurent. Na obrázku 2.1a jsou ukázky gotických fontů. Prosazení gotického písma souvisí s rozvojem hnutí katolické církve, jehož snahou bylo upevnění autority církve. Přechodným prvkem se stává tzv. romanogotika a navazující gotická minuskula, která má tři typy: základní, dekorativní (textura) a zběžnější. Vzniká také kurzívní gotické písmo, které lépe vyhovuje potřebám rychlého psaní. Typy kurzívy: bastardní písmo neboli švabach (prosazující se od 14. století), kaligrafická polokurzíva a běžná polokurzíva. Písmo je hodně nakloněné a využívá různé ligatury, označení pro spojení dvou znaků. Církevní a také právnické texty byly až do 16. století tištěny právě novogotickým švabachem a později frakturou, v českých zemích do roku 1848 a v Německu až do druhé světové války. Naopak pro latinské texty, už od začátku knihtisku, je typycký tisk renesanční antikvou.
U tohoto písma je typickým prvkem psaní velkých velmi zdobných prvních písmen na za-

(a) Ukázka fontů gotického typu. Obrázek ze zdroje [29].



(b) Různý znak pro jedno písmeno.

(c) Znak dlouhé s (long s).

Obrázek 2.1: Obrázky související s gotickým písmem.

čátku textu. Písmo obsahuje i některé neobvyklé znaky jako je například dlouhé s. Dříve se tato varianta používala pro zobrazení hlásky „s“ na začátku a uprostřed slabiky, zatímco na konci se používalo tzv. koncové, neboli okrouhlé s, ukázka na obrázku 2.1b. Některé dokumenty obsahují i různé znaky pro jedno písmeno v jednom souvislém textu. Většinou to bývá u písmen b a l, jak je vidět na obrázcích ??.

Jazyk

Staré texty obsahují nezvyklá slova, která pocházejí z doby, kdy se používala střední stará čeština a čeština doby husitské. Převážně se jedná o texty s církevními tématy. Využívá se spřežkový pravopis, kde různá písmena nahrazují dvojice písmen či jiné písmo – ou místo ů, ale i ů se zde vyskytuje, zdvojené dlouhé s značí š, ale používá se i š na konci slova, jak již bylo uvedeno, místo normálního s se psalo dlouhé s, ale na konci slova a jako spojka se psalo s. Dále g místo j, w místo v a v místo u na začátu slova, j místo dlouhého í a cž jako č, ale i č se používalo, rz nebo rž místo velkého ř. V některých textech se vyskytují zdvojené samohlásky místo dlouhé samohlásky nebo se také zaměňuje tvrdé y a měkké i. Tyto pravidla nebyly pevně dána, proto se může vyskytovat více variant jednoho slova v různých textech ale i v jednom dokumentu.

2.3 Existující řešení

V této sekci je stručné shrnutí existujích nástrojů pro detekci textu, rozpoznávání textů a znaků. Obecně pro optické rozpoznávání znaků existuje mnoho nástrojů. Mezi nejznámější

opensource nástroje patří Tesseract [1] a Ocropy[2], které využívají síť s dlouhou krátkodobou pamětí (LSTM). Dalším nástrojem je ABBYY [3], jedná se však o komerční produkt.

ABBYY

ABBYY je softwarová společnost poskytující nástroje optického rozpoznávání znaků, zachycení dokumentů a jazykových softwarů pro počítačová a mobilní zařízení. Nejznámějším produktem je ABBY FineReader, který převádí papírové dokumenty do digitálních. Produkt ABBYY historic OCR[4] je zaměřen na OCR historických textů s gotickým písmem.

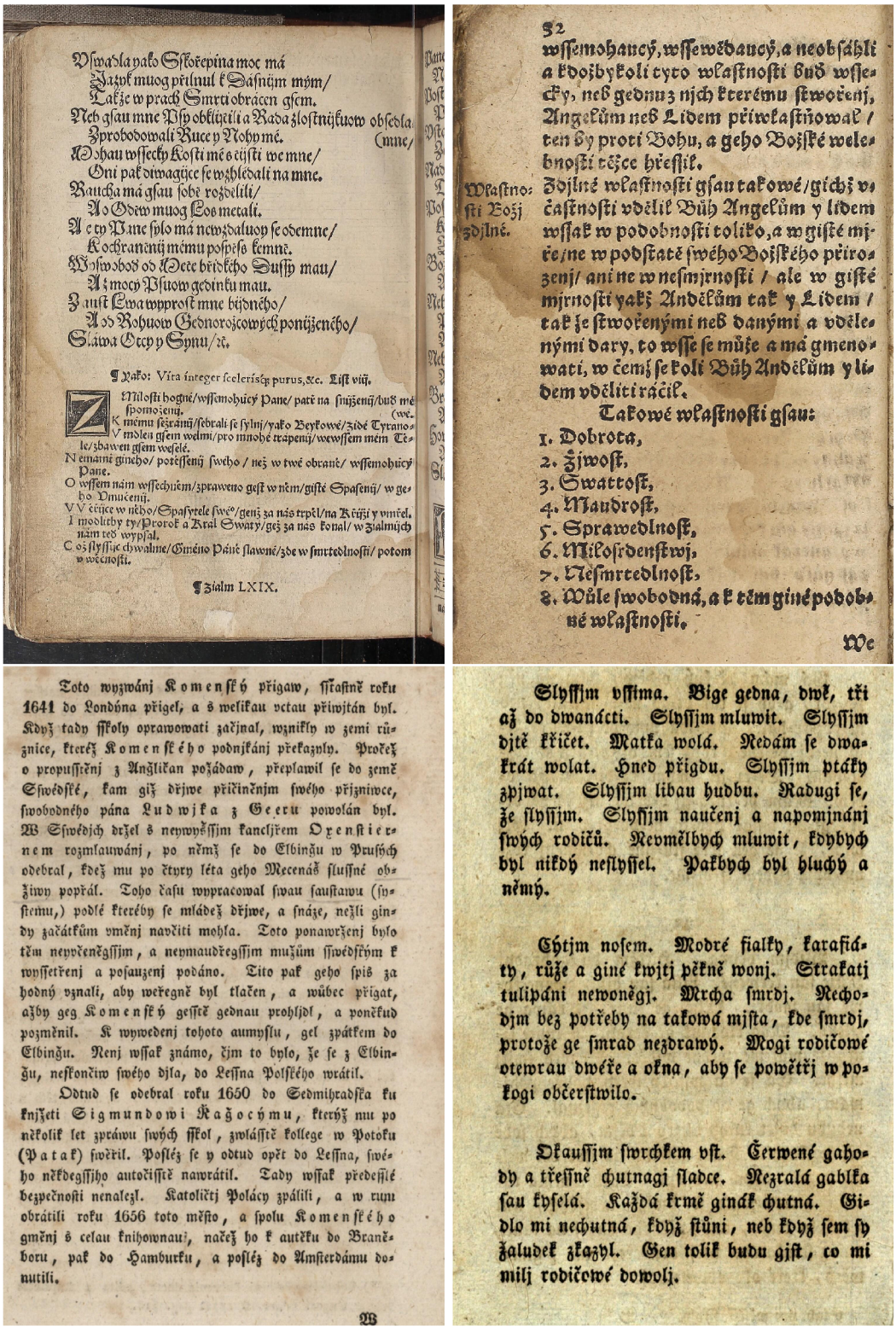
OCRopus

OCRopus je volně dostupný systém pro analýzu a optické rozpoznávání znaků, který je vydán pod licencí Apache. Tento systém byl speciálně navržen pro rozsáhlou digitalizaci knih jako jsou třeba Knihy Google. OCRopus řeší i analýzu rozvržení stránky. Rozpoznání textu je založeno na rekurentní síti LSTM, nevyžaduje tedy jazykový model, což dělá model jazykově nezávislým. Vhodným trénováním sítě, se může systém naučit i nové znaky a může být tedy aplikován na speciální dokumenty.

Tesseract

Tesseract je volně dostupný software, který je vydán pod licencí Apache a řadí se mezi nejpřesnější nástroje pro rozpoznávání znaků. Je k dispozici pro většinu operačních systémů. Dokáže rozpoznat více než 100 jazyků a podporuje velké množství typů písma, mezi které patří právě i gotické fonty – Fraktur. Tesseract může být natrénován i na jiné jazyky. Základem verze 4.0 je neuronová sít LSTM (s dlouhou krátkodobou pamětí), pro podporu více jazyků a jejich rozpoznávání. Tesseract používá metodu adaptivní klasifikace [27]. Rozpoznávání probíhá dvoustupňovým průchodem, v prvním průchodu se pokusí o rozpoznání každého slova, každé slovo, které je uznáno jako uspokojivé, je předáno adaptivnímu klasifikátoru, který pak v druhé fázi rozpoznává text s vyšší přesností. Klasifikátor se tedy v prvním průchodu časem naučí něco, co pak využije v druhém průchodu. Tesseract může být také využit ke trénování na historických textech, například v článku [22], je řešen problém chybějícího ground – truth (přepisu) textu u historických textů. To je řešeno automatickým generováním výcvikových dat s označenými obrázky a digitálním písmem, na kterém je pak trénován model Tesseract.

[1] https://github.com/tesseract-ocr
[2] https://github.com/tmbdev/ocropy
[3] https://www.abbyy.com
[4] https://www.frakturschrift.com/



Obrázek 2.2: Historické texty. Již na první pohled se texty liší velikostí písma, typem fontu a barvou papíru. První stránka je naskenovaná trochu křivě a obsahuje více bloků textu s různými velikostmi písma. Papír je u třetí stránky hodně znečištěný. Tisk je nedokonalý, téměř každé písmeno má jinou tloušťku. Také řádky jsou různě široké. U prvního a třetího textu jsou řádky hodně nahuštěné. Mezery mezy slovy jsou také u každého textu jiné. Jak je vidět u třetího textu, tak občas nejde rozeznat jednotlivá slova, jestli se jedná o dvě či jedno spojené slovo.

Kapitola 3

Neuronová síť

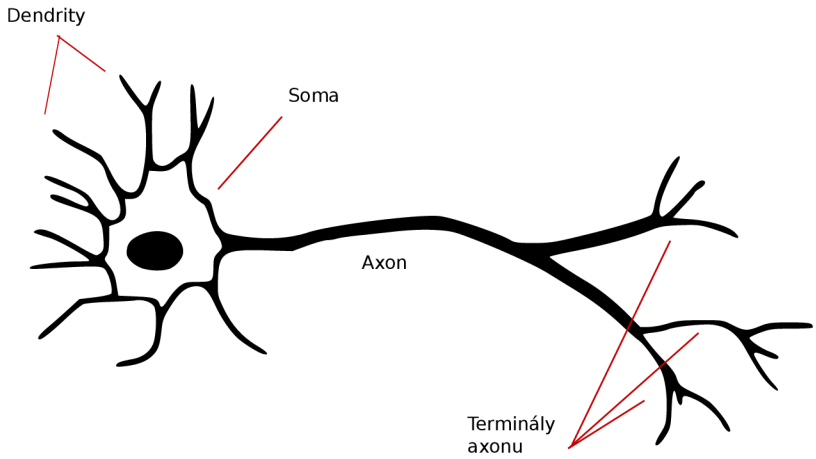
Tato kapitola postupně definuje základy neuronových sítích a stučně popisuje funkce které síť využívá. Neuronová síť (Neural network) je jedním z výpočetních modelů umělé inteligence, které jsou inspirovány chováním biologických struktur – lidské nervové soustavy. Neuronová síť je struktura určená pro distribuované paralelní zpracování dat a používá se v oblastech jako je klasifikace, aproximace a predikce. Také řeší úlohy počítačového vidění, rozpoznávání řeči, zpracování přirozeného jazyka, rozpoznávání zvuku, filtrování sociálních sítí, strojového překladu a bioinformatiky. První neuronovou síť – Percepton vytvořil Frank Rosenblatt [24] v roce 1958. Původním cílem umělých neuronových sítí bylo řešit problémy stejným způsobem jako je řeší lidský mozek. Neuronoá síť je složena z mnoha malých jednotek, které se nazývají neurony.

3.1 Neuron

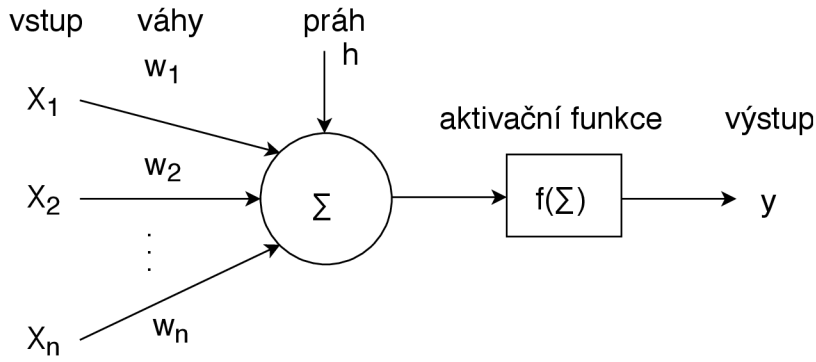
Umělý neuron je základní jednotka umělé neuronové sítě a vychází z biologické nervové buňky, jejíž model je na obrázku 3.1. Tělo neuronu neboli soma zprostředkovává základní funkci buňky. Pro komunikaci s ostatními neurony slouží výběžky neuronu – dendrity a axon. Dendrity jsou dostředivé výběžky a s jejich pomocí neuron přijímá vzruchy od ostatních neuronů. Množství dendritů se liší podle typu neuronu. Dlouhý výstupní výběžek neuronu se nazývá axon. Zprostředkovává výstup informací z nervové buňky a synapsemi se napojuje na dendrity dalších neuronů. Neuron má vždy jeden axon, na konci rozdělený do množství tzv. terminálů. Chemické rozhraní, které zprostředkovává výměnu informací, vzruchů, mezi neurony se nazývá synapse. Synapse můžeme rozdělit na dva základní druhy excitační a inhibiční. Pokud vzruch projde excitační synapsí, tak je zesílen, pokud přes inhibiční synapsi, je utlumen. Předpokládá se, že lidská paměť je tvořena právě jedinečným nastavením synapsí v mozku každého člověka. Tento fakt lze brát jako analogii pro nastavování vah pro jednotlivé neurony při učení umělých neuronových sítí. Nejpoužívanější matemitický model neuronu navrhli v roce 1943 Warren S. McCulloch a Walter Pitts [20] a je vyjádřen rovnicí

$$y = f(\sum_{i=0}^{N}(x_i w_i) + h), \tag{3.1}$$

kde x_i je hodnota i-tého vstupu neuronu, w_i je váha na i-tém vstupu, h nebo také bias je prahová hodnota aktivace neuronu, f je aktivační (přenosová) funkce a y je výstupní hodnota neuronu. Model je znázorněn na obrázku 3.2.



Obrázek 3.1: Model biologického neuronu. Obrázek je převzatý z [?].



Obrázek 3.2: Model matematického neuronu. Vstupem neuronu je vektor $x = (x_1, x_2...x_n)$, každý vstup x_i je ohodnocen váhou w_i z vektoru $w = (w_1, w_2...w_n)$, kde i je z intervalu od 0 do N. Dalším vstupem je prahová hodnota h. Vstupy se sečtou a vstupují do aktivační přenosové funkce, vznikne tak výstup neuronu y.

3.2 Aktivační funkce

Aktivační přenosová funkce je obecně nelineární funkcí transformující hodnotu vnitřního potenciálu neuronu. U neuronových sítí se nejčastěji používají tyto druhy přenosové funkce: skoková, usměrněná lineární (ReLU), sigmoida a hyperbolický tangens. Vstupní hodnota aktivační funkce je označena proměnnou x.

Skoková funkce

Skoková funkce vrací pro vstup menší než daná mez nulu, pro větší vrací jedna. Graf funkce je znázorněn na obrázku 3.3b. Lze ji vyjádřit rovnicí

$$f(x) = \begin{cases} 0 & \text{pro } x < 0 \\ 1 & \text{pro } x \geq 0. \end{cases} \tag{3.2}$$

Usměrněná lineární funkce

Usměrněná lineární funkce neboli ReLU (rectified linear unit) patří mezi nejpoužívanější ze všech přenosových funkcí. Rectifier neboli usměrňovač v oblasti neuronových sítí jako aktivační funkci poprvé představil Richard H. R. Hahnloser [12] v roce 2000 a Vinod Nair v roce 2010 ReLU [21]. Výhodou funkce je malá výpočetní náročnost. Graf funkce je na obrázku 3.3a. Je charakterizována rovnicí

$$f(x) = \max(x, 0). \tag{3.3}$$

Sigmoida

Sigmoida je jedna z logistických funkcí a její výhodou je, že výstup je v omezeném intervalu $(-1, 1)$. Graf funkce je znázorněn na obrázku 3.3d. Je popsána rovnicí

$$f(x) = \frac{1}{1 + e^{-kx}}. \tag{3.4}$$

Hyperbolický tangens

Hyperbolický tangens, kde výstup je v omezeném intervalu $(-1, 1)$, je definovaný rovnicí

$$f(x) = \tanh x = \frac{2}{1 + e^{-kx}} - 1, \tag{3.5}$$

Graf je znázorněn na obrázku 3.3c.

Softmax

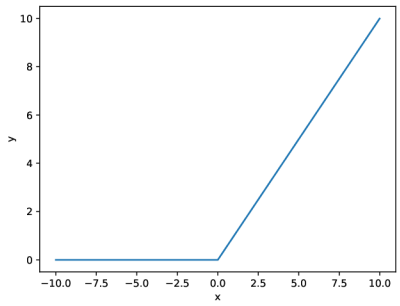
Funkce softmax rozdělí výstupy každé jednotky na hodnotu mezi 0 a 1 a rozdělí každý výstup tak, že celkový součet výstupů je rovný 1. Výstup funkce softmax je ekvivalentní kategorickému rozdělení pravděpodobnosti. Tato funkce se většinou používá v poslední vrstvě sítě, kde vstupní vektor převede na vektor pravděpodobností, kde každá hodnota reprezentuje pravděpodobnost příslušnosti do určité třídy. Matematicky je funkce softmax vyjádřena

$$f(x)_j = \frac{e^{x_j}}{\sum_{k=1}^{K} e^{x_k}}, \tag{3.6}$$

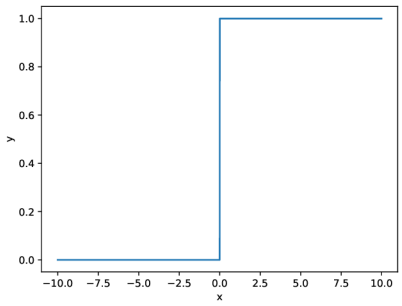
kde x je vektor vstupů do výstupní vrstvy, j indexuje výstupy a K je počet tříd. Softmax může být použita pro libovolný počet tříd.

3.3 Hluboká neuronová síť

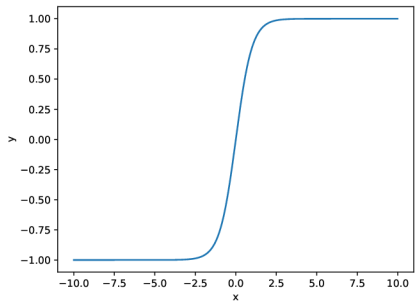
Umělé neuronové sítě jsou tvořeny z vrstev neuronů. Výstup z neuronu v jedné vrstvě je připojen ke vstupu neuronu v další vrstvě, toto platí pro dopředné neuronové sítě. Obsahují vstupní vrstvu, jednu nebo více skrytých vrstev a výstupní vrstvu, jak je znázorněno na obrázku 3.4a. Když zvyšujeme počet skrytých vrstev, rozšiřujeme tím síť. Hluboká neuronová síť je tedy umělá neuronová síť s více skrytými vrstvami mezi vstupní a výstupní vrstvou. U uměné neuronové sítě proces učení moderuje pomocí nastavení vah vstupů popř. změnou aktivační funkce. Jak se bude v neuronové síti šířit vzruch určuje postavení spojů mezi neurony, jedním způsobem postupu vzruchu je tzv. dopředné šíření (acyklická topologie sítě), které přenáší vzruch pouze jedním směrem, taková síť je na obrázku 3.4a. Rekurentní šíření



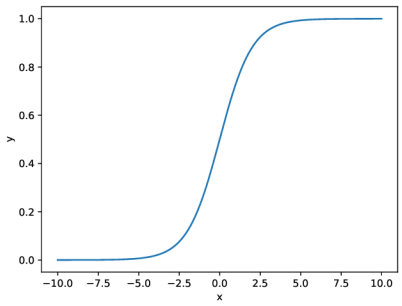
(a) Usměrněná linární



(b) Skoková



(c) Hyperbolický tangens



(d) Sigmoida

Obrázek 3.3: Graf aktivační funkce

(cyklická topologie sítě) pak navrací výsledky vyšších vrstev do vrstev nižších. Existuje také varianta cyklického šíření ve všech směrech, síť s tímto propojením všech neuronů je nazývaná Hopfieldova síť [16].

3.4 Konvoluce

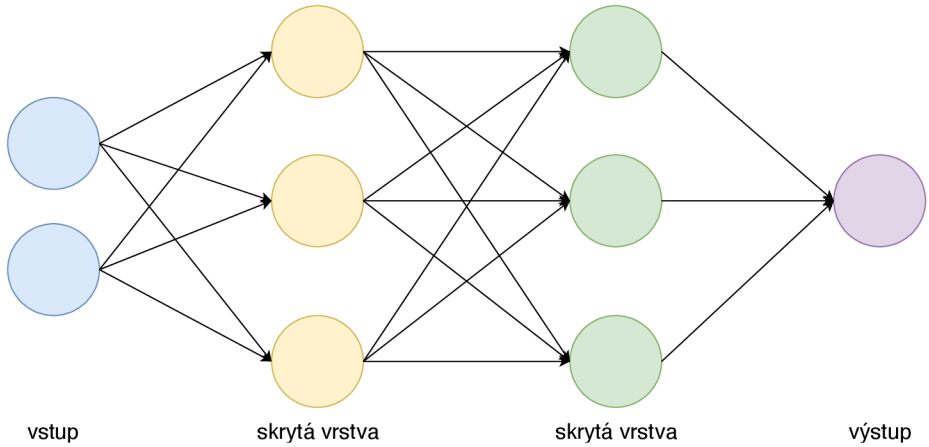
Použití konvolučního integrálu se nejdříve objevilo v D'Alembertově odvození Taylorovy věty. Konvoluce je matematiký operátor značící se $*$, který zpracovává dvě funkce. U konvoluce při zpracování obrazu je většinou jednou z funkcí $f(x)$ obrázek a druhou je nějaký filtr $g(x)$. Filtru $g(x)$ se říká konvoluční jádro. Hodnota konvoluce f s jádrem g v bodě x je integrál ze součinu funkce f s otočenou funkcí konvolučního jádra posunutou do bodu x. Spojitá konvoluce jednorozměrných funkcí $f(x)$ a $g(x)$ je definována rovnicí:

$$(f * g)(x) = \int_{-\infty}^{\infty} f(\alpha)g(x - \alpha)\mathrm{d}\alpha, \tag{3.7}$$

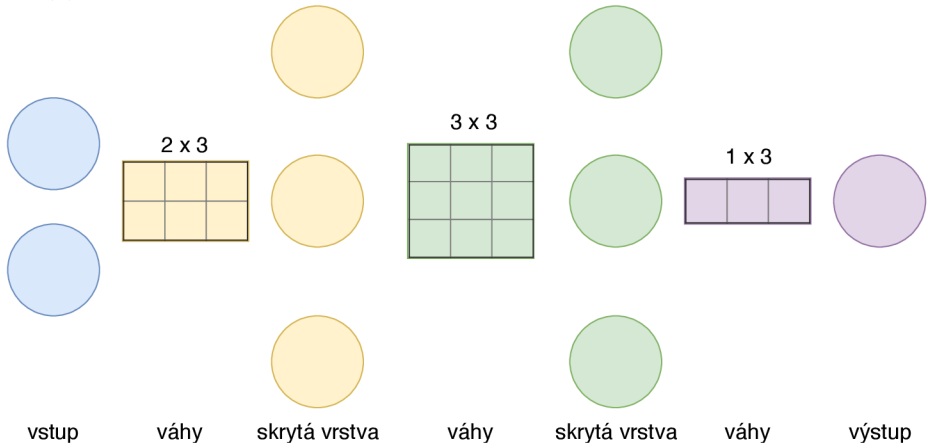
kde integrační proměnná α značí posun do bodu x. Diskrétní kovoluce je vyjádřena vztahem:

$$(f * g)(x) = \sum_{i=-\infty}^{\infty} f(i) \cdot g(x - i). \tag{3.8}$$

Výsledná řada je o jeden prvek kratší než je součet délek konvoluovaných řad. Operace konvoluce hlavně snižuje počet volných parametrů, což umožňuje, aby byla síť hlubší s méně



(a) Průchod vstupů přes vrstvy plně propojené dopředné neuronové sítě.



(b) Ukázka neuronové sítě s maticemi vah pro vstup do skrytých vrstev, každá vrstva je nelineární transformace vstupu z jednoho vektorového prostoru do druhého.

Obrázek 3.4: Umělá neuronová síť s více vrstvami, průchodem vstupů neuronové sítě přes skryté vrstvy vznikne výstup. Inspirace k obrázkům převzatá z [7].

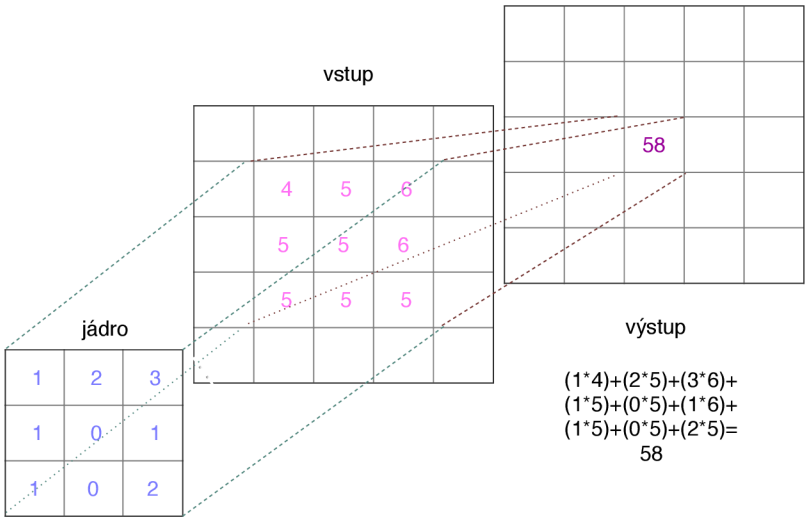
parametry než v plně propojené síti. Konvoluce se často používá při algoritmech zpracování dvourozměrného diskrétního obrazu v počítačové grafice. Jádro si lze představit jako masku, kterou položíme na obraz. Každý pixel překrytý tabulkou vynásobíme koeficientem v příslušné buňce a provedeme součet všech těchto hodnot, jak znázorňuje obrázek 3.5. Tím dostaneme jeden nový pixel. Výsledný obraz je rozostřen. Vzorec diskrétní konvoluce má potom tvar

$$(f * g)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x - i, y - j) \cdot g(i, j). \tag{3.9}$$

Čím větší konvoluční maska, tím bude obraz více rozostřen. Pomocí hodnot masky můžeme detekovat hrany.

3.5 Rekurentní neuronová síť

Rekurentní neuronová síť (Recurrent neural network – RNN) je třída umělé neuronové sítě obsahující smyčky, které jim umožňují uchovávat informace. Takže na rozdíl od dopředných



Obrázek 3.5: 2D konvoluce.

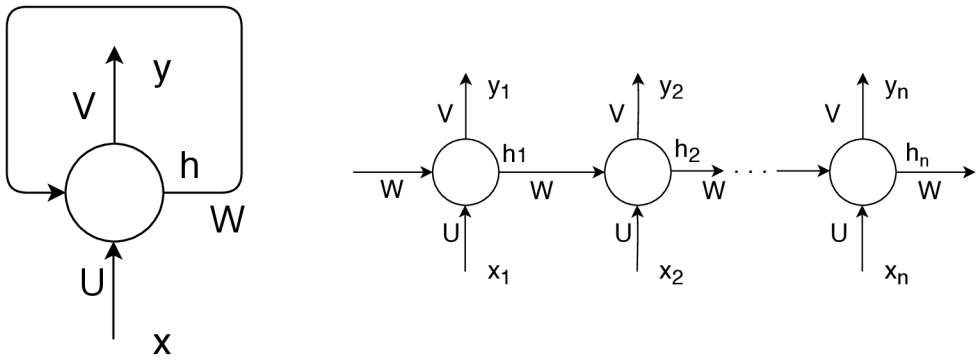
neuronových sítí, kde jsou výstupy navzájem nezávislé mohou rekurentní sítě používat svůj interní stav (paměť) pro zpracování sekvencí vstupů. Předpověď dalšího stavu je založena na předchozích předpovědích. Typickým využitím této sítě je například generování a rozpoznávání řeči, modelování jazyků, překladu nebo třeba rozpoznávání ručně psaných textů. Na obrázku 3.6 je model rekurentní sítě. Výpočet skrytého stavu neuronu h po daný časový krok n se vstupem x a výstupem y lze vyjádřit rovnicí

$$h_n = f(Ux_n + Wh_{n-1}), \tag{3.10}$$

kde funkce f je aktivační funkce neuronu, tedy nelineární funkce (většinou ReLU nebo hyperbolický tangens). Stav h_0, který je potřebný pro výpočet prvního skrytého stavu, je obvykle inicializovaný na nuly. Tento skrytý stav h_n je paměť sítě a zachycuje infomace o předchozích krocích. Výstup kroku y_n se vypočítá pouze na základě paměti v čase t, výstup může být například vypočten $y_n = \text{softmax}(Vh_n)$, výsledkem je vektor pravděpodobností z dané množiny a vyjadřuje předpověď následujícího prvku. Na rozdíl od tradiční hluboké neuronové sítě, která používá různé parametry v každé vrstvě, sdílí RNN stejné parametry (U, V, W) ve všech krocích, což snižuje celkový počet parametrů. Pro trénink sítě se také používá zpětnovazební algoritmus, ale protože jsou parametry sdílené všemi časovými kroky a aktuální výpočty jsou závislé na předchozích krocích, tak je potřeba sečíst jednotlivé gradienty. K tomu se využívá algoritmus Backpropagation Through Time (BPTT). RNN trénované s BPTT mají potíže s naučením dlouhodobých závislostí, tedy závislostí vzdálených kroků. To je způsobeno násobením stále stejnými sdílenými parametry, které se provádí při zpětném šíření. Čím více se budeme pohybovat dozadu, tím větší nebo menší se chybový signál stává. To znamená, že síť má potíže s memorováním slov daleko v sekvenci a dělá předpovědi založené jen na těch nejnovějších. Tento jev se nazývá problém mizejícího či explodujícího gradientu a řeší ho různé modely jako dlouhá krátkodobá paměť (LSTM) či uzavřená rekurentní jednotka (GRU).

Long short – term memory (LSTM) – Dlouhá krátkodobá paměť

LSTM je model pro krátkodobou paměť, která může trvat dlouhou dobu. Tuto síť požívá Tesseract i OCRopus. Sítě dlouhé krátkodobé paměti (LSTM) byly vynalezeny pány



Obrázek 3.6: Model rekurentní neuronové sítě a časové rozvinutí při výpočtu. Rozvinutí znamená zapsání celé sítě pro celou sekvenci. Proměnné $x_1 ... x_n$ reprezentují vstupní slova v kroku n, $y_1 ... y_n$ reprezentují předpokládané další slova a $h_0 ... h_n$ je skrytý stav v kroku n, který nese informaci o předešlém vstupním slově, vypočítá se na základě předchozího skrytého stavu a vstupu v aktuálním kroku: $h_n = f(Ux_n + Wh_{n-1})$. Obrázek přejat ze zdroje [1]

Hochreiterem a Schmidhuberem v roce 1997 [15]. Bloky dlouhé krátkodobé paměti jsou stavební jednotkou vrstvy rekurentní neuronové sítě, které jsou schopné zachytit dlouhodobé závislosti. Tato jednotka je složena z buňky, vstupní, výstupní a zapomínací brány. Mezi buňkou a bránami existuje propojení. Buňka je zodpovědná za uchování hodnot v libovolných časových intervalech. Každá ze tří bran může být myšlena jako umělý neuron vícevrstvé neuronové sítě. Tyto brány regulují tok hodnot, které procházejí spojeními LSTM, podrobněji popsáno v článku [23, 2].

Gated Recurrent Unit (GRU) – Uzavřená rekurentní jednotka

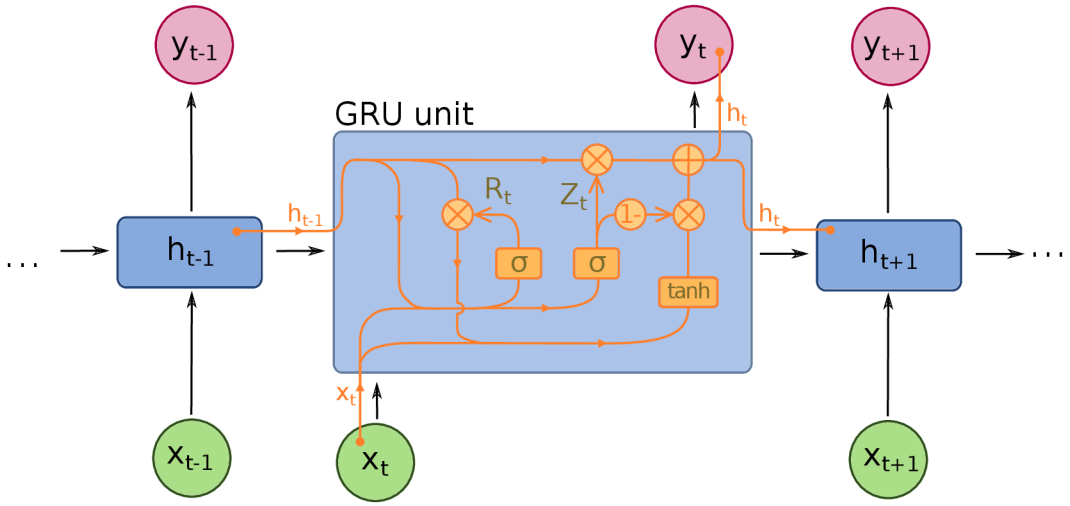
GRU představil Kyunghyun Cho [3] v roce 2014. Tato jednotka se může považovat za variantu LSTM, protože je jí velmi podobná. Také řešení problém snižujícího se gradientu u rekurentních sítí. Ale oproti LSTM má méně parametrů, protože neobsahuje výstupní bránu. Má aktualizační (update gate) a resetovací bránu (reset gate), které rozhodují o tom jaké informace by měly být předány dál na výstup. Zvláštností je, že tyto jednotky mohou být vyškoleny k tomu, aby uchovávaly informace dlouhou dobu, aniž by je v průběhu času odstranili i když jsou pro předpověď bezvýznamné [18].

Update gate – Aktualizační brána Pomocí této brány jednotka zjistí, kolik z předchozích časových kroků $t-1$ bude předáno dále. Vstup x_t je vynásoben váhou W a také vstup, který obsahuje informace o předchozích jednotkách h_{t-1} je vynásoben váhou U. Oba jsou sečteny a vstupují do aktivační funkce sigmoid. Výsledek z této funkce se pohybuje mezi 0 a 1. Výpočet brány pro aktualizaci z_t je vyjádřen vztahem

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1}). \tag{3.11}$$

Tato rovnice je zobrazena v obrázku 3.7 a výstup je značen Z_t.

Reset gate – Resetovací brána Pomocí této brány jednotka zjistí, kolik z minulých informací zapomene. Pro výpočet je použita rovnice, která je stejná jako pro předchozí



Obrázek 3.7: Model GRU. Označní operací: + (plus), x (násobení matic), σ (funkce sigmoid) a tanh (hyperbolické tangens). Šipky znázorňují vstup nebo výstup proměnných do operací. Vstupy do jednotky představují proměnné x, y znázorňuje výstup jednotky a h představuje obsah paměti v čase t. R_t značí výstup resetovací brány a Z_t znační výstup aktualizační brány. Obrázek je převzat ze zdroje [6].

bránu

$$r_t = \sigma(W^{(r)}x_t + U^{(r)}h_{t-1}). \tag{3.12}$$

Výstup má však jiný účel. Tato rovnice je zobrazena v obrázku 3.7 a výstup je značen R_t.

Obsah paměti Obě brány ovlivňují konečný výstup. Jako první se použije resetovací brána. Vynásobením matice mezi resetovací bránou r_t a informací z předchozího jednotky vynásobené svou váhou Uh_{t-1} se určí, co se odstraní z předchozích časových kroků t. Čím víc se r_t blíží k 0, tím víc dochází k zapomínání. To se pak sečte se vstupem x_t vynásobeným váhou a vloží se do nelineární aktivační funkce tanh. Tento výpočet je vyjádřen vztahem

$$h'_t = \tanh(Wx_t + r_t \times Uh_{t-1}). \tag{3.13}$$

Jako poslední krok se vypočte vektor h_t, který uchovává informace o aktuální jednotce a předává je dál do sítě. Pro tento účel je použita aktualizační brána z_t, která určuje, co se má shromáždit z aktuálního obsahu paměti h'_t a co z předchozích kroků h_{t-1}. Pokud se vektor z_t blíží 1, zachová většinu předchozích informací, ale ztrácí většinu aktuálních, protože $1 - z_t$ se blíží 0. Rovnice

$$h_t = z_t \times h_{t-1} + (1 - z_t) \times h'_t \tag{3.14}$$

vyjadřuje konečný stav buňky v kroce t. V obrázku 3.7 je to zakreslono jako h_t.

Výsledný model je jednodušší než standardní modely LSTM a stává se víc populárnější.

Connectionist temporal classification (CTC)

V roce 2006 představil Alex Graves [10, 11] metodu CTC, která je vhodná pro trénování sítě. Tato metoda se zabývá klasifikací a označováním sekvenčních dat, znázorněno

na obrázku 3.8. Byla navržena pro značení sekvenčních dat s RNN a odstraňuje potřebu předběžné segmentace trénovacích dat a následné zpracování výstupů. Například u ručně psaného písma nebo zvukového záznamu, kde není přesné zarovnání mezi vstupem a výstupem. Výstupy jsou dány výběrem nejpravděpodobnějšího označení pro danou vstupní sekvenci. Cílem je tedy mapovat vstupní sekvenci $X = x_1, x_2...x_T$ jako je obrázek nebo zvukový záznam slova ke korespondující výstupní sekvenci $Y = y_1, y_2...y_U$, což je přepis. CTC algoritmus nám dává pro dané X výstupní distribuci nad všemi možnými Y. Výstup Y je matice, kde sloupce odpovídají časům a každý řádek odpovídá pravděpodobnosti písmene (softmax) z definované abecedy $L \cup blank$, kde jednotka blank značí „prázdné“ označení. To znázorňuje obrázek 3.8. Společně tyto výstupy definují pravděpodobnosti všech možných způsobů zarovnání všech možných sekvencí štítků se vstupní sekvencí. CTC zarovnání dává přirozený způsob, jak přejít z pravděpodobností v každém časovém kroku k pravděpodobnosti výstupní sekvence. Symbolickým propojením nejvyšších pravděpodobností štítků v čase T se nazývá cesta a značí se π. Pravděpodobnost určité cesty je produktem všech výstupů y aktivační funkce (softmaxu) v čase T. Vyjádřeno rovnicí

$$p(\pi|x) = \prod_{t=1}^{T} y_{\pi_t}^t. \tag{3.15}$$

Pro nalezení optimálního sekvence l je potřeba nejprve vypočítat pravděpodobnosti všech možných cest a shrnout pravděpodobnosti cest, které dávají stejné označení, a nakonec vybrat sekvenci s nejvyšší pravděpodobností. Cesty π jsou mapovány operátorem $\mathbb{B}$ na štítky $l \in L^{<=T}$. Operátor $\mathbb{B}$ odstraní opakované štítky a pak také prázdné označení, znázorněno na obrázku reffig:ctc. Součet pravděpodobností všech odpovídajících cest dává pravděpodobnost výskytu štítku l vyjadřeno rovnicí

$$p(l|x) = \sum_{\pi \in \mathbb{B}^{-1}(l)} p(\pi|x). \tag{3.16}$$

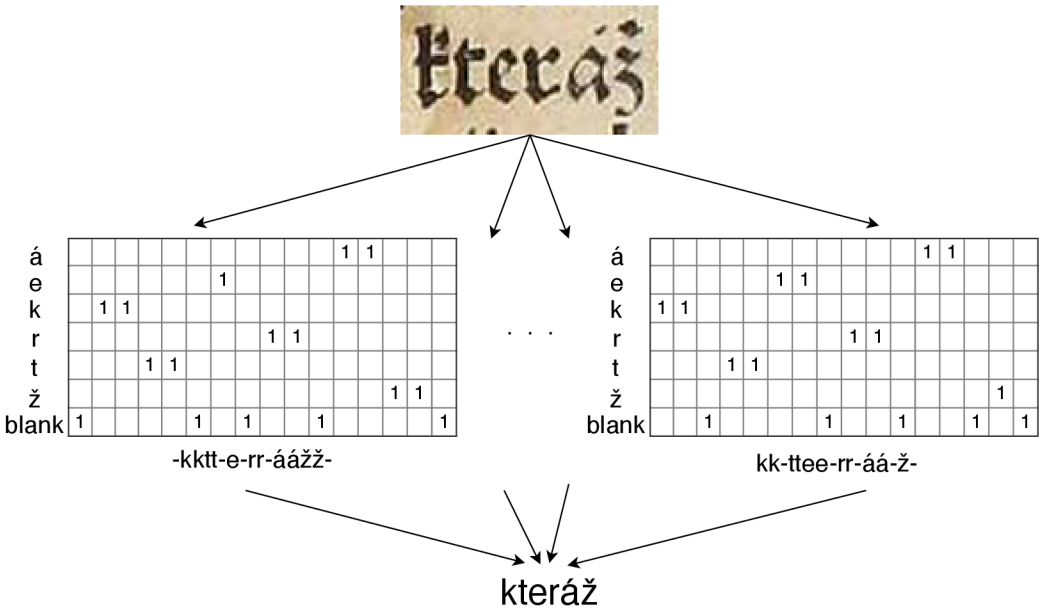
Ale protože existuje velké množství cest, tak součet všech možností by tedy byl výpočetně velmi náročný, proto se tento výpočet nepoužívá. Místo toho se používá algoritmus dopředného zpětného šíření (forward-backward algorithm), obrázek ??. Pro modifikované sekvence l tedy sekvence, které obsahují prázdné znaky se používá označené l'. Prázdé znaky jsou vloženy na začátek, na konec a mezi každý štítek. Délka sekvence je $|l'| = 2|l| + 1$. Pravděpodobnost sekvence štítků je dána součtem dopředné a zpětné proměnné,

$$p(l|x) = \sum_{s=1}^{|l'|} \alpha_t(s)\beta_t(s). \tag{3.17}$$

Obě proměnné dopředné i zpětné se počítají rekurzivně a jen se „správnými“ štítky, tedy jen se štítky s povolenými přechody, jak je znázorněno na obrázku 3.8. Dopřednou proměnnou $\alpha(s)$ definuje součet pravděpodobností všech počátečních cest dosahujících indexu s ze sekvence l' v čase t

$$\alpha_t(s) = P(\pi_{1:t} : \mathbb{B}(\pi_{1:t}) = l_{1:s/2}, \pi_t = l'_s|x) = \prod_{\pi:\mathbb{B}(\pi_{1:t})=l_{1:s/2}} y^{t'}_{\pi'_t}, \tag{3.18}$$

kde, pro nějakou sekvenci $s, s_{a:b}$ je podsekvence $(s_a, s_{a+1}, ..., s_{b-1}, s_b)$. V tohoto případě využijeme tedy polovinu původní sekvence l pro indexování sekvence l'. Zpětná proměnná



Obrázek 3.8: Princip CTC. Vstupem je obrázek slova, získáme matice s rozdělením pravděpodobnostmi nad abecedou štítků v čase t. V tabulce jsou zobrazeny pravděpodobnosti písmene (neboli štítku) z dané abecedy na dané pozici v čase t, blank je „-“ prázdný znak. Pro názornost je nejvyšší pravděpodobnost předpovídané kategorie označena 1. Pokud propojíme v tabulce nejvyšší pravděpodobnosti, získáme cestu. To znázorňuje sekvence štítků pod tabulkou. Cesty se dekodují odstraněním opakováných štítků a prázdného znaku. Výsledná sekvence nám vznikce sumou pravděpodobností cest.

$\beta(s)$ je definována jako součet pravděpodobnosti všech zakončení cesty, které by dokončily označení l, pokud začátek cesty dosáhl indexu l' v čase t

$$\beta_t(s) = P(\pi_{t+1:T} : \mathbb{B}(\pi_{t:T}) = l_{s/2:|l|}, \pi_t = l'_s | x) = \prod_{\pi:\mathbb{B}(\pi_{t:T})=l_{s/2:|l|}} y^{t'}_{\pi'_t}. \tag{3.19}$$

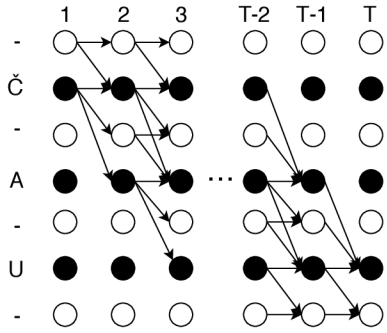
Cílem výcviku neuronové sítě je dosáhnout největší pravděpodobnosti, tedy je nutné současně maximalizovat logické pravděpodobnosti všech správných klasifikací v trénovací množině. Nechť S je trénovací množina obsahující dvojici vstupních a cílových sekvencí (x, z), kde délka posloupnosti z je menší nebo rovna délce vstupní sekvence x. Jeden nebo více vstupních prvků lze zarovnat s jedním výstupním prvkem, ne však naopak. Cílem je minimalizovat následující objektivní funkci

$$O^{ML}(S, N) = -\sum_{(x,z)\in S} \ln(p(z|x)), \tag{3.20}$$

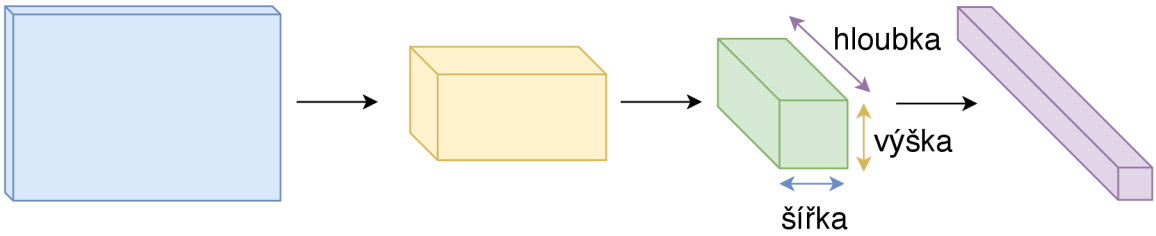
kde N značí síť. Tedy vztah vstupu a výstupu $y = N(x)$. Více informací lze nalézt v článcích [10, 11, 13] a [26, 19].

3.6 Vrstvy sítě

Zde jsou popsány jednotlivé vrstvy sítě. Ne všechny jsou využity v praktické části.



Obrázek 3.9: Algoritmus dopředného zpětného šíření na označení slova „ČAU“. Bílé kruhy znázorňují prázdný znak (blank) a černé kruhy značí štítky. Každý sloupec znázorňuje čas. Šipky znamenají povolené přechody. Směr šipky značí směr šíření aktualizace dopředné proměnné, zpětné proměnné se aktualizují v opačném směru. Obrázek přejatý z [10]

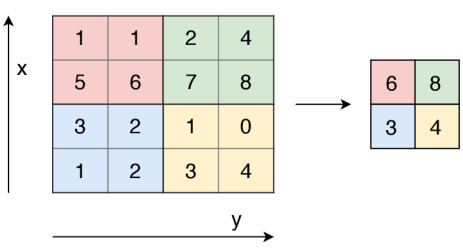


Obrázek 3.10: Konvoluční síť s neurony ve třech dimenzích – hloubka, výška a šířka. Každá vrstva transformuje 3D vstup do 3D výstupu. Vstupem je obrázek určité výšky, šířky a hloubky, kterou tvoří 3 barevné kanály RGB

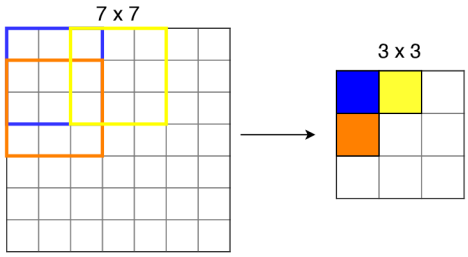
Konvoluční vrstva

Na vývoji konvolučních neuronových sítích se podílel Yann le Cun [4, 5], síť použil k rozpoznávání ručně psaných číslic. Konvoluční neuronové sítě, pojmenované podle toho, že obsahují mimo jiné minimálně jednu konvoluční vrstvu (také pooling, plně propojenou a normalizační vrstvu), využívají skutečnosti, že vstup je tvořen obrazy. Vrstvy mají neurony uspořádané ve třech rozměrech: šířka, výška a hloubka, zobrazeno na obrázku 3.10 [1]. V této vrstvě se nastavují tři hyperparametry, které ovládají velikost výstupu: hloubka – počet a velikost filtrů (konvolučních jader), krok – posunutí filtru a nulování neboli padding – zachování prostorové velikosti vstupního a výstupního objemu. Počet konvolučních jader určuje počet kanálů výstupných dat a velikost kroku posunutí jádra určuje velikost výstupních dat, čím je krok větší tím je výstup menší. Jedna vrstva neuronů tedy provede konvoluci mezi vstupem konvoluční vrstvy a množinou vah jejích neuronů, model na obázku 3.4b. Výstup konvoluční vrstvy tvoří výstupy všech rovin neuronů, které se označují jako mapy příznaků. Neurony konvoluční vrstvy slouží k detekci příznaků jako jsou hrany, rohy, změna barvy atp. Následující vrstvy kombinují jednotlivé příznaky z několika map na nové příznaky o vyšší úrovni.

[1] Obrázek přejat z http://cs231n.github.io/convolutional-networks/



(a) Maxpooling s filtrem 2x2 a krokem 2.



(b) 7x7 vstup a 3x3 filtr s krokem 2 - výstup 3x3

Obrázek 3.11: Velikost výstupu vrstvy je ovlivněna mnoha faktory. Obrázky přejaté z [8]

Zploštění – Flatten

Tato vrstva přemění všechny vstupní dvourozměrné vektory do jednoho jednorozměrného vektoru.

Plně propojená vrstva – Dense

Plně připojené vrstvy (Fully connected layer) spojují každý neuron v jedné vrstvě s každým neuronem v jiné vrstvě, toto propojení je zakresleno v modelu na obrázku 3.4a. Počet výstupů je stejný jako počet neuronů ve vrstvě, případně počet výstupních tříd reprezentované výstupním vektorem.

Pooling – Podvzorkování

Tato vrstva provádí převzorkování prostorových rozměrů jako je šířka a výška. To přináší výhody jako je snížení množství parametrů nebo vah, čímž se sníží výpočetní náklady. Konvoluční sítě mohou obsahovat lokální nebo globální vrstvy vzorkování, které kombinují výstupy skupin neuronů na jedné vrstvě do jednoho neuronu v další vrstvě. Podvzorkování je určené velikostí vzorkovací matice a kroku, což je vidět na obrázku **??**, používají se různé druhy. Například podle maxima (maxpooling) využívá maximální hodnotu z každé skupiny neuronů na předchozí vrstvě, znázorněno na obrázku 3.11a. Dalším příkladem je podvzorkování podle průměru (avgpooling), které využívá průměrnou hodnotu z každé skupiny neuronů na předchozí vrstvě.

Dropout – Vyřazení

Dropout je regularizační technika pro snížení přetrénování (overfitting) v neuronových sítích. Brání přílišnému přizpůsobení na trénovací data. V této vrstvě dojde k vynechání jednotek v neuronové síti, toho se docílí nastavením aktivace některých náhodných neuronů na nulu. Tato technika snižuje vzájemné působení uzlů, což vede k tomu, že se učí na více robustních funkcí, které se lépe zobecňují na nová data. Tato metoda také výrazně zlepšuje rychlost výcviku. Více o dropoutu si lze přečíst v článcích [14, 28]

BatchNormalization – Dávková normalizace

Normalizace dávek řeší problémy změny rozložení aktivací neuronů a umožňuje použít vyšší koeficient učení, který zrychlí trénování. Můžeme použít vyšší míru učení, protože dávková

normalizace zajišťuje, že nedojde k žádné aktivaci, která by byla skutečně vysoká nebo skutečně nízká. Podobně jako při dopoutu přidává do aktivace každé skryté vrstvy nějaký šum. Aby se zvýšila stabilita neuronové sítě, normalizace dávek normalizuje výstup předchozí aktivační vrstvy odečtením průměru dávky a dělením standardní odchylkou dávky. Normalizace se provádí při trénování pomocí statistických dat jedné dávky. Tato vrstva bývá mezi plně propojenou vrstou a aktivační vrstvou. Batch normalizace je popsána v článku [17].

3.7 Chybové funkce a optimalizace

Trénování neuronové sítě zahrnuje vyhodnocování chyby výpočtu. Ztrátová vrstva je obvykle konečnou vrstvou sítě a určuje odchylku při tréninku mezi výslednou předpovídanou hodnotou a požadovanou pravdivou hodnotou. Tuto hodnotu lze získat pomocí chybových funkcí. Hodnota chybové funkce udává odchylku výstupu od očekávaného výstupu. Cílem trénování je stanovení takových vah neuronů, aby chyba byla minimalizována. Pro snižování chyby se využívá optimalizace. K dispozici mohou být různé funkce ztrát, které jsou vhodné pro různé úkoly. Ztráta softmaxu se používá pro předpovídání jedné třídy vzájemně se vylučujících tříd. Pro klasifikaci je vhodná křížová entropie. Euklidovská ztráta se používá pro regrese na skutečně hodnocené štítky výpočtem euklidovské vzdálenosti. Mezi nejpoužívanější chybové funkce však patří: Cross entropy Loss a Mean Square Error.

Cross Entropy Loss

Křížová entropie je funkce, která se používá k předpovědi nezávislých pravděpodobností a proto se hodí pro klasifikaci tříd. Křížová entropie mezi skutečným rozložením pravdivosti tříd p a odhadovanou distribucí g definuje nad vstupy x rovnice

$$H(p, q) = -\sum_{x} p(x) \log q(x). \tag{3.21}$$

Cílem funkce je co nejvíce zvětšit pravděpodobnost správné třídy.

Mean Square Error

Střední kvadratická chyba měří průměr čtverců odchylek mezi požadovaným vzorem Y' a skutečným výstupem Y přes vstupní vektor o velikosti N, vyjádřeno matematicky

$$MSE = \frac{1}{N} \sum_{i=0}^{N} (Y_i - Y_i')^2. \tag{3.22}$$

Cílem funkce je minimalizovat rozdíl mezi Y a Y'.

Backpropagation – Zpětné šíření chyby

Zpětné šíření chyby je základní algoritmus pro výpočet gradientu, který je potřebný pro výpočet vah používaných v síti pro učení vícevrstné neuronové sítě. Principem této metody je, že chyba je vypočtena na výstupu a distribuována zpět přes síťové vrstvy.

Stochastic gradient descent (SGD) – Stochastický gradientní sestup

Tento algoritmus je jedním z nejpoužívanějších optimalizačních algoritmů pro trénovaní hlubokých neuronových sítí. Jde o iterativní aproximační gradientní sestup. Základním principem výpočtu gradientu ∇ je součet parciálních derivací chybové funkce Q vůči jednotlivým parametrům sítě w. Podle gradientu ∇, který udává směr chyby, se pak upravují váhy w. Rychlost trénování sítě ovlivňuje krok α tzv. learning rate, který udává velikost změny vah. Vyšší hodnota zrychlí učení, protože změna bude větší, ale menší hodnota způsobí lepší aproximaci k ideální hodnotě kvůli menší změně. Vzorec pro úpravu váhy dávky je vyjádřen rovnicí

$$w = w - \alpha \nabla Q(w), \tag{3.23}$$

kde $Q(w)$ je průměrná hodnota gradientu v dávce (batch). Existují i jiné varianty optimalizačních algoritmů, více informací je v [25, 9].

Kapitola 4

Specifikace zvolené metody

Tato část obsahuje popis přípravy datasetu a zvolené architektury sítě. Rozpoznávání slov jsem pojala jako klasifikaci obrázku pomocí vícevrstvé konvoluční sítě, prototože vstupní obrázky obsahují velké množství šumu a také protože je to i jednoduše představitelné. Vzhledem k tomu že obrázky musí mít stejnou velikost, byl řešen problém upravení velikosti. Proto byly vytvořeny dvě sady trénovacích i testovacích, ve kterých to bylo řešeno jiným způsobem. Tato práce řeší rozpoznávání jednotlivých „izolovaných“ slov na obrázku, používá k tomu hlubokou konvoluční neuronovou síť. Síť slouží jako klasifikátor, který určí slovo s největší pravděpodobností. Pro každé slovo je jedna třída. Cílem je získat z obrázku slova dné slovo. Vstup je tedy obrázek určité velikosti a výstupem je pravděpodobnost daného slova z omezeného slovníku. Počet tříd je tedy roven počtu slov daného slovníku. Zvolila jsem metodu klasifikace obrazu pomocí hluboké konvoluční sítě, kde výstupem je pravděpodobnost třídy daného obrázku, tedy pravděpodobnost daného slova. Tuto metodu jsem zvolila, protože historické texty obsahují velké množství šumu, což by bylo pro rozpoznávání a různých nedokonalostí. Slovo je vstupní popis (label) obrázku, toto slovo pak značí index do vektoru obsahující všechny slova. Je tu tedy vstupní slovník. Cílem je zjistit pro jaký obrazový vstup bude model lépe fungovat. Zvolila jsem metodu konvolučních neuronových sítí, které se skládají z vrstev konvoluční, maxpooling, a dropoutem ve spojení s plně propojenými vrstvami, které umožnují jednoduchou klasifikaci.

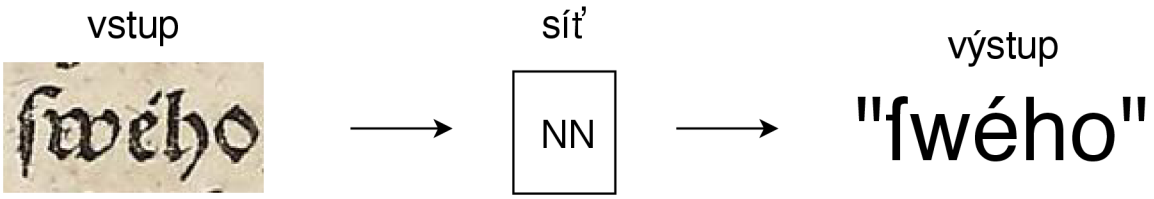
4.1 Příprava dat

Protože nebyl nalezen žádný vhodný dataset, který by obsahoval sadu českých historických textů s anotacemi, pro české texty je dost možné, že ani neexistuje, tak bylo nutné si vytvořit svou datovou sadu. V datasetu se data dělí na trénovací a testovací. Testovací data obsahují obrázky slov vyextrahované z reálných obrázků historických textů. Trénovací data, byla vytvořena uměle generátorem a měla by se co nejvíc podobat testovacím. Proto byla data zpracována podobným způsobem.

4.2 Generátor

Pro generování datasetu byl využit generátor, jehož autorem je Bc. Martin Kišš [1]. Generátor je vytvořen tak, že umí generovat buď strany, řádky nebo slova. Aby byly data co nejpodobnější, byla použita část generátoru vytvářející celé stránky. A následně podle anotací

[1] https://github.com/xkissm00/gothic_dataset_generator



Obrázek 4.1: Cíl řešení. Vstupem sítě je obrázek izlovaníého slova, výstupem je samotné slovo.

Obrázek 4.2: Na horním řádku je ukázka testovacích dat, na spodním řádku je ukázka generovaných dat.

vystřiženy slova stejným způsobem jako u zpracovávání dat anotovaných ručně. Generátor gotického textu se snaží co nejvěrněji napodobit reálné dokumenty. Proto jsou aplikovány různé metody poškození textu či písma, použité pozadí napodobující stránky a podobně.

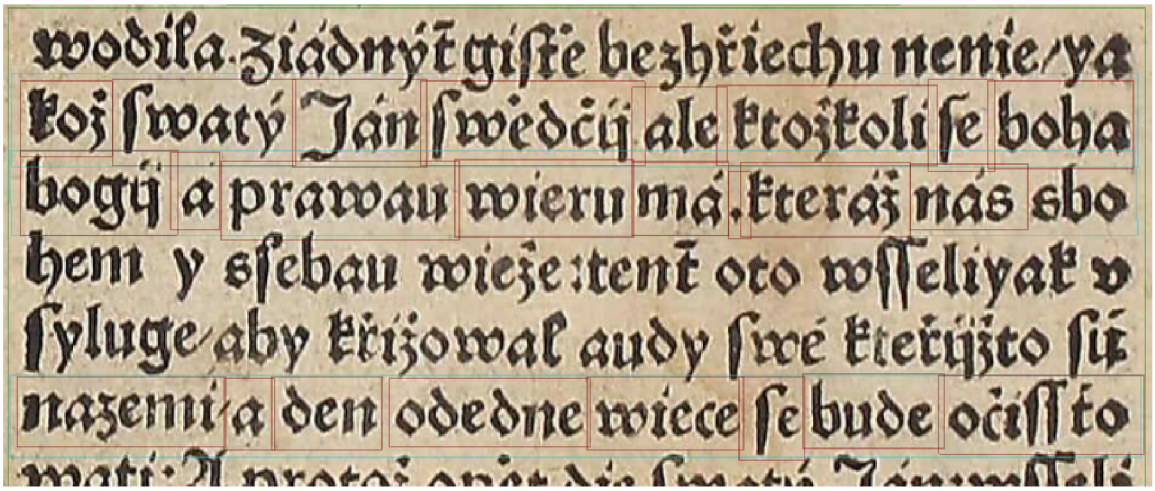
Nastavení

V generátoru lze konfigurovat různé parametry jako je nastavení vstupního souboru s textem pro generovnání, vstupní složky obsahující aplikované fonty a pozadí. Také je tu možnost tištění anotací přímo do textu. U textu lze nastavit možnost interpunkce za slovy a nastavení velkých prvních písmen, či převedení vstupního textu pouze na malé písmena. Dále jdou nastavit parametry pro výslednou stránku, například výška a okraje stránky, počet a délka řádků textu a mezery mezi slovy. Také lze nastavit pravděpodobnost obtištěného/ prosvítajícího textu. U nastavení nedokonalostí tisku lze určit maximální a minimální frekvenci, maximální sytost barvy, maximální počet a minimální a maximální barvu skvrn. Další modifikace je rozmazání, u té lze nastavit parametr pro gausovo rozmazání, z nastaveného rozmezí maximální a minimální sigma. Také lze stanovit maximální a minimální frekvence rozmazání a maximální barva. Poslední je nastavení transformací jako je překlad a rotace.

Pro genenování českého textu bylo nutné vybrat pouze české gotického fonty kvůli slovům s diakritikou, pro slova bez diakritiky byly použity obecné fonty, aby byly generované texty rozmanitější. U některých fontů byl problém se specifickým znakem „dlouhé s“. Proto se musely vytřídit a používat pouze pro slova, která tento znak neobsahují.

4.3 Anotace dat

Anotace neboli označení udávají informace o obrázku, konkrétně slovo obsažené v obrázku. Anotování slov reálných dat bylo tedy důležitou součástí datasetu. Byl využit volně do-



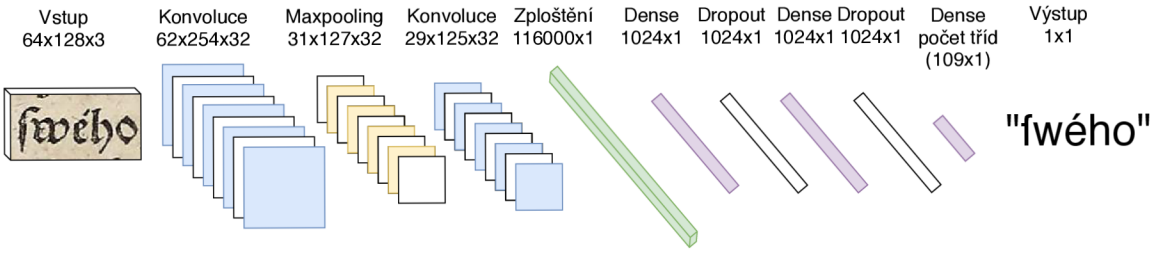
Obrázek 4.3: Manuální anotace v Transkribusu, slova jsou anotována hnědým rámečkem, řádky modrým a blok textu zeleným. Lze vidět, že anotace nejsou úplně těsně kolem slova, ale obsahuje kousíček okolí případně i útržek jiného znaku, ale vždy obsahuje jen jedno slovo celé.

stupný nástoj Transkribus [2], ve kterém byly manuálně označeny oblasti textu, řádky a slova, to je znázorněno na obrázku 4.3. Dohromady bylo zpracováno zhruba 104 obrázků stránek historických textů, které jsou zhruba z 30 různých dokumentů stažených z volně dostupné online databáze knih [3]. Z každého dokumentu jsou použité asi 3 strany a na každé je oanotováno 3 – 5 řádků. To tvořilo dohromady 3340 anotovných slov. Výstup z tohoto nástroje je dokument ve formátu .xml, ve kterém jsou zapsány souřadnice jednotlivých označených textových oblastí. Problémem zpracování reálných textů je, že musí mít při vstupu do sítě všechna data stejnou velikost. Proto mohla být buď slova zreslena nebo výstřižek obsahoval více jak jedno slovo, nebo pozůstatky slov.

4.4 Architektura sítě

Použitá síť obsahuje dvě konvoluční vrstvy, jednu vrstvu maxpooling, tři plně propojené vrstvy, jednu vrstvu pro zploštění a dva dropouty. Síť je propojena následujícím způsobem: Do sítě vstupuje obrázek velikosti 64 x 128 x 3, jako první je obrázek zpracován konvoluční vrstvou s konvolučním jádrem 3 x 3 a aktivací ReLU, to způsobí změnšení obrazu a také zvýšení počtu vrstev. Následuje vrstva maxpooling s filtrem 2x2 a další konvoluční vrstva s jádrem 3 a aktivací ReLU. Poté je vstup serializován vrstvou zploštění na jednu dimenzi a propojen s plně propojenou vrstvou. Následuje vrstva dropout, která nastaví náhodné aktivace na nulu. Pak je další plně propojená vrstva s aktivací ReLU a opět dropout. Jako poslední je plně projená vrstva s aktivací softmax, která je využita jako klasifikátor a udává pravděpodobnost dané třídy. Tedy pravděpodobnost daného slova. Architektura je znázorněna na obrázku 4.4. Síť je vhodná jako klasifikátor díky funkci softmax, která vstupní vektor přemění na vekter pravděpodobnostní, kde součet všech hodnot dává 1.

[2] https://transkribus.eu/Transkribus/
[3] http://www.digitalniknihovna.cz/



Obrázek 4.4: Architekrura konvoluční neuronové sítě, síť obsahuje funkce konvoluce, max-pooling, dropout, ReLU, softmax a plně propojené vrstvy.

Kapitola 5

Experimenty a výsledky

V této kapitole jsou popsány jednotlivé experimenty a demonstrovány výsledky. Dále jsou uvedeny nástroje, které byly zvoleny pro implementaci experimentů.

5.1 Implementace a testování

Implementace experimentu je v jazyce Python s využitím knihovny Keras[1]a TensorFlow[2], které obsahují speciální výpočetní metody. Výpočty se prováděly na výpočetním centru Metacentrum[3].

Experminenty

[t] iter	class	train/test	ztráta	přesnost	test. ztáta	test. přesnost	verze
140	109	10476/1131	0.00 3	0.99	9.330	0.36	v1
50	109	10476/1131	0.005	0.998	6.30	0.37	v2
20	100	10476/1131	0.045	0.986	9.76	0.19	v2
60	100	24896/6490	0.50	0.89	2.44	0.71	v1, gen
78	100	24896/6490	0.48	0.891	3.28	0.69	v2, gen
140	100	10480/1131	0.006	0.998	9.45	0.34	v1
50	102	10480/1131	0.014	0.996	7.72	0.336	v3

5.2 Výsledky a zhodnocení

Jak jde vidět v tabulce 5.1, (je zde zmíněn počet iterací, počet trénovacích a testovacích dat, ztráty a přesnost), tak vyšší přesnosti jak 40 % na reálných datech nebylo dosaženo. To může být způsobeno mnoha aspekty. Například malá frekvence učitého slova. Bude to hlavně odlištými daty při trénování a testování. Může to také být i předzpracováním slov tak, aby je bylo možné vložit do modelu a testovat. Musely se tedy všechny slova upravit na danou velikost. To bylo řešeno třemi různými způsoby vystřižení. Tento model je nevhodný pro velkou slovní zásobu, ale pouze omezenou. Pro zajímavost jsem trénovala síť i na hodně španých datech s vysokým šumem a navíc jsem přimíchala do testovaných dat i uměle vytvořená data, na těch byla očekávaně dosažena vyšší přenost klasifikace. Testování jsem prováděla na dvou a někdy i třech různých variantách zpracování dat. Jedna varianta

[1] https://keras.io/
[2] https://www.tensorflow.org/
[3] https://metavo.metacentrum.cz/

vstupních dat byla vystřižena z reálných a generovaných dat, tak aby byla v poměru 2:1. Další data jsem pouze vystřihla kolem anotace a vložla na průhledné pozdí a poslední variatou bylo, vystřižení podle řádku s určitou velikostí.

5.3 Možnosti budoucího vývoje

Tento model může být vhodný jako část řešení rozpoznávání textu například po předzpracování. Jinak pro rozsáhlé dokumenty by vyžadoval velké množství dat pro trénování, kvůli množství tříd, aby obsahoval minimální slovník. To vždy nemusí být vhodné, ale mohl by být řešením pro specifické jednotlivé dokumenty s jednotným písmem. Také je nutné slova předem zpracovat, což je poměrně náročné. Vhodným řešením a budoucím vývojem by mohlo být aplikovat část tohoto modelu na vyhledání slova na řádku či písmena ve slově a poté použít rekurentní síť, která je teď velmi rozšířená a používá se právě na rozpoznávání slov v určitém kontextu, a protože se sama učí může být jazykově nezávislá. Těmto modelům jsem se taky dost věnovala v teoretické části, protože by mohly být aplikovány právě na tento problém.

Kapitola 6

Závěr

Tato práce obsahuje obecný přehled o neuronových sítích, hlavně se zabývá konvolučními sítěmi a vrstvami sítě. Cílem práce bylo navrhnout systém na pro zpoznávání historických textů. Zaměřila jsem se hlavně na rozpoznávání slov. To je řešeno klasifikací slov ze slovníku. Byl také vytvořen dataset umělých a reálných dat s českými slovy pro testování sítě na historických textech v gotickém písmu. Byl imlementován navrhovný model hluboké konvoluční sítě a tento model byl trénován na vytvořených datech. Trénování neproběhlo moc úspěšně. Nejspíš to bylo rozdílnými daty, které se mi nepovedlo přizpůsobit tak, aby je byl model schopen úspěšně rozpoznat. Přesnost se pohybovala kolem 30 až 40 %. Na generované sadě byla přesnost vyšší. V budoucím vývoji by se model mohl propojit s rekurentní sítí, což by pomohlo při rozpoznávání. A také by bylo vhodné využít jiný formát textu než jen amotná slova, ale třeba celé řádky.

Literatura

[1] Britz, D.: *Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs*. Září 2015, [Online; navštíveno 09.05.2018].
URL http://www.wildml.com/wp-content/uploads/2015/09/rnn.jpg

[2] Britz, D.: *Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano*. Říjen 2015, [Online; navštíveno 09.05.2018].
URL http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano/

[3] Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; aj.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR*, ročník abs/1406.1078, 2014.

[4] le Cun, Y.: *Generalization and network design strategies*. 1989, technical Report CRG-TR-89-4.

[5] le Cun, Y.; Bottou, L.; Bengio, Y.; aj.: *Gradient-based learning applied to document recognition*. 1998, ISSN 2278–2324, doi:10.1109/5.726791.

[6] Deloche, F.: *Gated Recurrent Unit*. Červen 2017, [Online; navštíveno 09.05.2018].
URL https://en.wikipedia.org/wiki/File:Gated_Recurrent_Unit.svg

[7] Dertat, A.: *Applied Deep Learning - Part 1: Artificial Neural Networks*. Srpen 2017, [Online; navštíveno 28.04.2018].
URL https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

[8] Deshpande, A.: *A Beginner's Guide To Understanding Convolutional Neural Networks Part 2*. Červenec 2016, [Online; navštíveno 28.04.2018].
URL https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/

[9] Goodfellow, I.; Bengio, Y.; Courville, A.: *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[10] Graves, A.; Fernández, S.; Gomez, F.: Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *In Proceedings of the International Conference on Machine Learning, ICML 2006*, 2006, s. 369–376.

[11] Graves, A.; Fernández, S.; Liwicki, M.; aj.: Unconstrained On-line Handwriting Recognition with Recurrent Neural Networks. 01 2007.

[12] Hahnloser, R. H. R.; Sarpeshkar, R.; Mahowald, M. A.; aj.: *Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit.* Nature, *ročník 405, 2000: s. 947–951.*

[13] Hannun, A.: Sequence Modeling with CTC. Distill, *2017, doi:10.23915/distill.00008, https://distill.pub/2017/ctc.*

[14] Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; aj.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR, *ročník abs/1207.0580, 2012.*

[15] Hochreiter, S.; Schmidhuber, J.: Long Short-Term Memory. Neural Comput., *ročník 9, č. 8, Listopad 1997: s. 1735–1780, ISSN 0899-7667, doi:10.1162/neco.1997.9.8.1735.*
URL http://dx.doi.org/10.1162/neco.1997.9.8.1735

[16] Hopfield, J. J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences, *ročník 79, č. 8, 1982: s. 2554–2558, ISSN 0027-8424, doi:10.1073/pnas.79.8.2554.*

[17] Ioffe, S.; Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. CoRR, *ročník abs/1502.03167, 2015.*

[18] Kostadinov, S.: Understanding GRU networks. Prosinec 2017, [Online; navštíveno 10.05.2018].
URL https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

[19] Liwicki, M.; Graves, A.; Bunke, H.; aj.: A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In In Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007, *2007.*

[20] McCulloch, W. S.; Pitts, W.: A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics, *ročník 5, č. 4, 1943: s. 115–133, ISSN 0007-4985.*

[21] Nair, V.; Hinton, G. E.: Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on International Conference on Machine Learning, *ICML'10, USA: Omnipress, 2010, ISBN 978-1-60558-907-7, s. 807–814.*
URL http://dl.acm.org/citation.cfm?id=3104322.3104425

[22] Nunamaker, B.; Bukhari, S. S.; Borth, D.; aj.: A Tesseract-based OCR framework for historical documents lacking ground-truth text. In 2016 IEEE International Conference on Image Processing (ICIP), *Sept 2016, s. 3269–3273, doi:10.1109/ICIP.2016.7532964.*

[23] Olah, C.: Understanding LSTM Networks. Srpen 2015, [Online; navštíveno 09.05.2018].
URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[24] Rosenblatt, F.: The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. Psychological Review, *ročník 65, č. 6, 1958: s. 386–408.*

[25] *Ruder, S.: An overview of gradient descent optimization algorithms.* CoRR, *ročník abs/1609.04747, 2016, 1609.04747.*
URL http://arxiv.org/abs/1609.04747

[26] *Shi, B.; Bai, X.; Yao, C.: An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition.* CoRR, *ročník abs/1507.05717, 2015, 1507.05717.*
URL http://arxiv.org/abs/1507.05717

[27] *Smith, R.: An Overview of the Tesseract OCR Engine. In* Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), *ročník 2, Sept 2007, ISSN 1520-5363, s. 629–633, doi:10.1109/ICDAR.2007.4376991.*

[28] *Srivastava, N.; Hinton, G.; Krizhevsky, A.; aj.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting.* Journal of Machine Learning Research, *ročník 15, 2014: s. 1929–1958.*

[29] *Wikipedia contributors: black-letter typefaces. Březen 2005, [Online; navštíveno 13.05.2018].*
URL https://en.wikipedia.org/wiki/Blackletter#/media/File:Gebrochene_Schriften.png

[30] *Wikipedia contributors: Intelligent word recognition. Prosinec 2014, [Online; navštíveno 13.05.2018].*
URL https://en.wikipedia.org/wiki/Intelligent_word_recognition

[31] *Wikipedia contributors: Gotické písmo. Říjen 2017, [Online; navštíveno 13.05.2018].*
URL https://cs.wikipedia.org/wiki/Gotick%C3%A9_p%C3%ADsmo

[32] *Wikipedia contributors: Optical character recognition. Květen 2018, [Online; navštíveno 13.05.2018].*
URL https://en.wikipedia.org/wiki/Optical_character_recognition