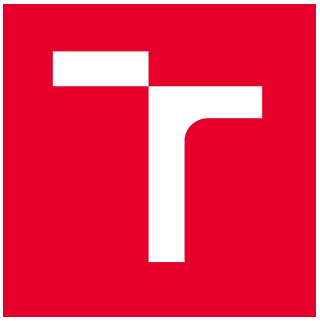


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

FORENZNÍ ANALÝZA RUČNĚ PSANÉHO PÍSMA PRO ČESKÉ PROSTŘEDÍ S POUŽITÍM UMĚLÉ INTELIGENCE

FORENSIC ANALYSIS OF HANDWRITING FOR THE CZECH ENVIRONMENT USING ARTIFICIAL INTELLIGENCE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Jan Stejskal**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Radim Burget, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Bc. Jan Stejskal ***ID:*** 211272

Ročník: 2 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Forenzní analýza ručně psaného písma pro české prostředí s použitím umělé inteligence

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s problematikou analýzy ručně psaného písma s použitím umělé inteligence a zpracujte přehlednou rešerši současného stavu vědy a techniky. Na základě poskytnuté databáze (poskytne vedoucí) vytvořte trénovací a testovací databázi krátkých úryvků textu. Navrhněte experiment, s pomocí kterého bude možné měřit podobnost dvou vzorků textu. Měření proveďte s použitím různých technik. Naměřené výsledky zaneste do tabulky a výsledky vhodně komentujte.

DOPORUČENÁ LITERATURA:

podle pokynů vedoucího práce

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 19.5.2023

Vedoucí práce: doc. Ing. Radim Burget, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

ABSTRAKT

Analýza ručně psaného písma je důležitou oblastí výzkumu moderní vědy. Jedná se však o velice složitý proces, jelikož ručně psaný text může nabývat různých podob. Využití umělé inteligence k analýze a identifikaci textu pocházejícího od různých autorů není ve světě nic nového. Avšak výzkum v této oblasti pro české prostředí mírně zaostává. Z tohoto důvodu bylo v rámci této práce navrženo a porovnáno několik architektur konvolučních sítí, ve snaze nalézt nejvhodnější strukturu pro řešení tohoto problému. Ze všech natrénovaných a otestovaných modelů dosáhl nejvyšší přesnosti model založený na struktuře ResNet18, který měl úspěšnost 92,2 % na vlastní databázi tvořené 1328 ukázkami s rozlišením 750×256. Tento výsledek naznačuje, že s dostatečně velkou a kvalitní databází je daný problém řešitelný i v českém prostředí s jeho komplikovanější znakovou sadou.

KLÍČOVÁ SLOVA

Identifikace autora, Ručně psaný text, Neuronová síť, Konvoluční neuronová síť, Siamská konvoluční síť, VGG, ResNet, GoogLeNet, Attention mapy, Python, Tensorflow

ABSTRACT

The analysis of handwriting is an important area of research in modern science. However, it is a very complex process because handwritten text can take on various forms. The use of artificial intelligence for analyzing and identifying text from different authors is nothing new in the world. Research in this area is, however, slightly lagging behind in the Czech environment. For this reason, several convolutional network architectures were proposed and compared in this work in an effort to find the most suitable structure for solving this problem. Of all the trained and tested models, the model based on the ResNet18 architecture achieved the highest accuracy, with a success rate of 92.2 % on a self-made database of 1328 samples with a resolution of 750×256. This result suggests that with a sufficiently large and high-quality database, the problem can be solved even in the Czech environment with its more complicated character set.

KEYWORDS

Author identification, Handwritten text, Neural network, Convolutional neural network, Siamese convolutional network, VGG, ResNet, GoogLeNet, Attention maps, Python, Tensorflow

STEJSKAL, Jan. *Forenzní analýza ručně psaného písma pro české prostředí s použitím umělé inteligence*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023, 56 s. Diplomová práce. Vedoucí práce: doc. Ing. Radim Burget, Ph.D

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Jan Stejskal
VUT ID autora:	211272
Typ práce:	Diplomová práce
Akademický rok:	2022/23
Téma závěrečné práce:	Forenzní analýza ručně psaného písma pro české prostředí s použitím umělé inteligence

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Doc. Ing. Radimu Burgetovi, Ph.D. za odborné vedení, pravidelné konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Analýza ručně psaného písma je jednou z velice důležitých oblastí výzkumu moderní vědy. Své uplatnění nachází v oborech, které se zabývají biometrikou – například zkoumání validity podpisů nebo forenzním zkoumáním dokumentů. V posledních letech se ukázal zájem o převod ručně psaných dokumentů do digitální formy pro jejich bezpečnější uchování a ochranu před degradací materiálu. A v neposlední řadě se také objevil zvýšený zájem o tuto oblast výzkumu v kriminalistice a zabezpečovacích systémech.

Jedná se však o velice složitý proces, jelikož ručně psaný text může nabývat různých podob. Ukázka se od ukázky může lišit použitým jazykem a jeho příslušnou znakovou sadou – abecedou. Tyto ukázky můžou být psány hůlkovým písmem, kurzívou, tiskacím nebo psacím písmem. Často se jedná o kombinaci vícero zmíněných stylů. Slovanské jazyky také využívají velké množství diakritických znamének, které při snaze o porovnání dvou ukázek mohou hrát velkou roli. Výzkum v této oblasti pro české prostředí mírně zaostává.

Proto se tato práce snaží přijít s funkčním experimentem, pomocí kterého by bylo možné porovnat páry ukázek českého písma a na základě podobnosti rozhodnout, zda tyto ukázky pochází od stejného autora, či nikoliv. Ze všech dostupných technik byla pro tento experiment využita siamská konvoluční síť. Ta umožňuje zpracovat vstupní pár obrázků konvoluční sítí a z jednotlivých ukázek získat informace o ručně psaném textu. Tyto informace lze následně s použitím různých technik porovnat a vyvodit z nich závěr.

Hlavním přínosem této práce jsou potom navržené a implementované modely. Ty využívají jak vlastní, tak různé ověřené struktury konvolučních sítí, jako jsou VGG nebo ResNet a jejich zakončení je tvořeno buďto plně propojenou sítí neuronů nebo je pro výstupní vektory počítána euklidovská metrika. Všechny tyto modely byly natrénovány na vlastní databázi a jejich výsledky porovnány pro různá rozlišení vstupních ukázek.

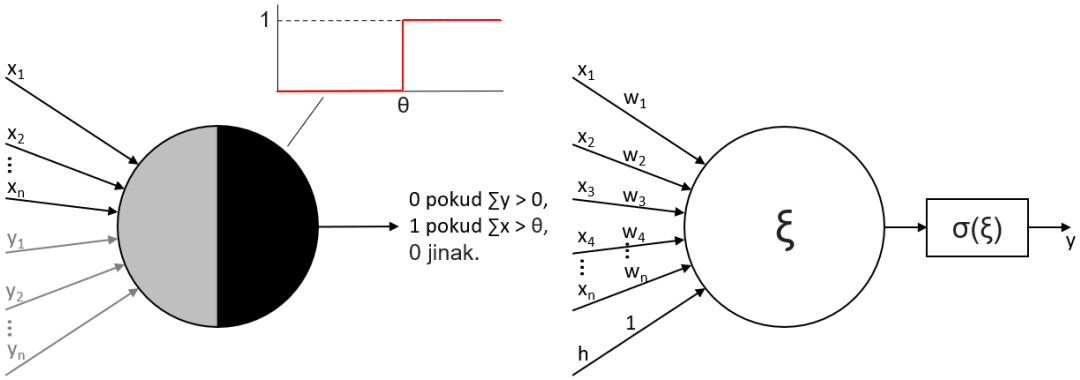
Práce se dělí na čtyři hlavní kapitoly. První kapitola je čistě teoretická a seznámí čtenáře s základními informacemi o neuronových sítích. Druhá kapitola představí některé zajímavé práce, které se zabývají touto tématikou a byly sepsány v posledních pěti letech. Třetí kapitola popisuje praktickou část této práce. To znamená využívané knihovny, zpracované databáze, datové sady a jejich tvorbu, navržené modely a také tvorbu attention map. Čtvrtá a poslední kapitola představí výsledky experimentu formou tabulek, grafů a attention map a pokusí se tyto výsledky objasnit a vyvodit z nich závěr.

1 Neuronové sítě

Tato kapitola popisuje základní poznatky o neuronových sítích. Zaměřuje se na základní stavební blok neuronové sítě – neuron, popisuje topologii, způsoby učení, predikce a představí čtenáři základní druhy neuronových sítí.

1.1 Neuron

Základem každé neuronové sítě je neuron. Prv ní, velice jednoduchý matematický model byl poprvé oficiálně představen již v roce 1943. Měl několik binárních vstupů a jeden výstup. Vstupy se dělily na excitační a inhibiční a nebyly nijak váhovány. Výstupní signál se potom měnil na základě poměru aktivních vstupů, pokud bylo aktivních více excitačních vstupů, byl excitován i binární výstup neuronu. A naopak v případě, že převažovaly inhibiční vstupy, výstup excitován nebyl.[1]



(a) První návrh neuronu (b) Moderní představa neuronu

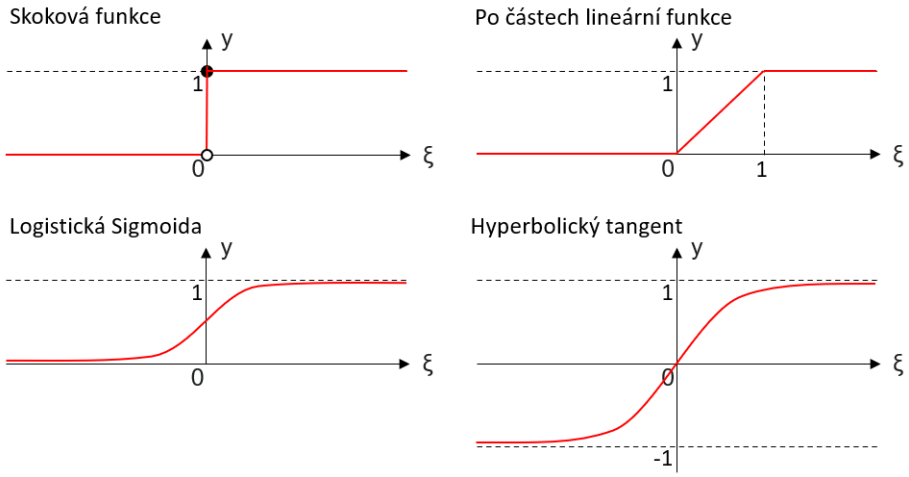
Obr. 1.1: Porovnání neuronu

V dnešní době je neuron komplexnější. Stejně jako jeho první návrh má určitý počet vstupů **x** – vstupních synapsí. Tyto vstupy se však od sebe nijak neliší a každá synapse je ohodnocena vahou **w**. Ta rozhoduje o excitační nebo inhibiční funkci vstupu svým znaménkem. Společně se synapsemi se na neuron napojuje i konstanta **h**, které říkáme prahová hodnota nebo také anglicky – bias. Tuto konstantu je možné do neuronu přivést zvlášť nebo ji začlenit jako jednu z vah vstupních synapsí. Vstupní hodnota této váhy potom bude vždy rovna jedné.[1][3] Součástí těla je i blok spravující aktivační přenosovou funkci neuronu σ a výstup **y** z tohoto bloku je přímým výstupem neuronu.[1]

Aktivační přenosová funkce se rozhoduje na základě vážené sumy vstupních hodnot, tu můžeme popsat řeckým písmenem ξ a lze vypočítat podle rovnice:

$$\xi = \sum_{i=1}^{n} w_i \, x_i + h = w_1 \, x_1 + w_2 \, x_2 + \, \ldots \, + w_n \, x_n + h. \tag{1.1}$$

Tuto hodnotu nazýváme také jako vnitřní potenciál neuronu.[1][2][3] Hodnota na výstupu se potom bude řídit podle zvolené přenosové funkce neuronu. Těchto přenosových funkcí existuje mnoho. Může se jednat například o skokovou funkci, po částech lineární funkci, logistickou sigmoidu nebo třeba hyperbolický tangent.[1]



Obr. 1.2: Aktivační funkce

V případě skokové funkce lze výstup y matematicky popsat jako:[2]

$$y = \sigma(\xi) = \begin{cases} 1 \text{ pokud } \xi \geq 0 \\ 0 \text{ pokud } \xi < 0 \end{cases}, \tag{1.2}$$

kde ξ je vnitřní potenciál neuronu a σ je daná aktivační přenosová funkce. Nebo v případě sigmoidy, by se jednalo o předpis:[1]

$$y = \sigma(\xi) = \frac{1}{1 + e^{-\xi}}. \tag{1.3}$$

Díky svému odlišnému charakteru každá z funkcí obvykle nalézá využití v jiných implementacích neuronových sítí. Není to však podmínkou. V dnešní době se velice běžně využívají různé přenosové funkce v rámci různých vrstev jednoho modelu. Z teoretického hlediska je dokonce možné použít více funkcí v rámci jedné vrstvy, této možnosti se ale v praxi téměř nevyužívá.[4]

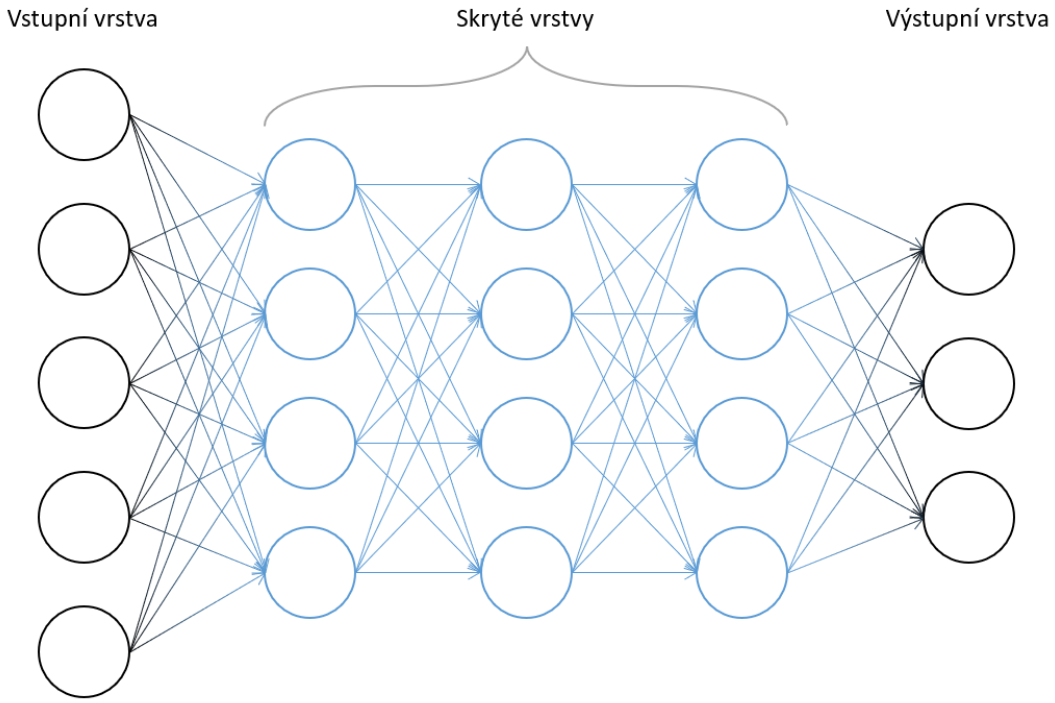
1.2 Neuronová síť a Hluboké učení

Roku 1944 byla poprvé představena myšlenka, že by se pro rozvoj strojového učení mohla využít takzvaná Neuronová síť. Tato síť, která má napodobovat lidský nervový systém, by se učila provádět různé úlohy, jako je detekce objektů v obraze nebo převod mluvené řeči do písemné podoby, zkoumáním a analýzou stovek i tisíců ukázek.[5]

Ukázalo se však, že neuronové sítě je možné použít k řešení mnohonásobně většího množství problémů. Může se jednat například o odhad spotřeby úsporných bioklimatických budov[22], předpověď teploty změny skupenství skla[23] nebo odhad polní sklizně[24].

1.3 Topologie

Jak již z názvu sítě vyplývá, je tvořena neurony (1.1). Jejich počet se obvykle pohybuje v rámci desítek až tisíců. Neurony se v rámci modelu dělí do vrstev, kdy každý neuron je ovlivněn výstupními hodnotami několika neuronů předchozí vrstvy a ovlivňuje svou výstupní hodnotou několik neuronů ve vrstvě následující.[5]



Obr. 1.3: Topologie neuronové sítě

Sítě, které mají jednu nebo více skrytých vrstev, to jsou všechny vrstvy, které

se nachází mezi vstupní a výstupní vrstvou, nazýváme jako vícevrstvé neuronové sítě. Takové sítě jsou lepší v řešení nelineárních problémů, jsou však složitější na trénování.[3]

Při návrhu neexistuje žádný pevně daný postup, jak dosáhnout ideálního modelu pro požadovaný úkol. Existuje několik pravidel a doporučení, jak při návrhu postupovat, která prvotní návrh usnadní. Avšak určování parametrů, jako je počet neuronů v každé vrstvě, rychlost učení modelu nebo způsob inicializace vrstev se provádí experimentálně.[3]

1.4 Způsoby učení

Způsobů, jak neuronovou síť učit, existuje mnoho. S úplně prvním a nejstarším způsobem přišel Donald Hebb. Po něm pojmenované Hebbovo učení popisuje způsob, jakým je možné trénovat neuron s binárními vstupy a výstupy.[1] Základní myšlenka, na které je toto učení založeno, zní následovně:

- hodnoty vah, které se nachází mezi dvěma aktivními neurony se budou zvyšovat,
- hodnoty vah, které se nachází mezi dvěma neurony, které aktivní nejsou, se budou snižovat.

Pokud by jsme měli nějaký vstupní vektor binárních hodnot $\mathbf{x}$, potom lze hodnoty vah $\mathbf{w}$ upravit na nové hodnoty $\mathbf{w}_n$ podle rovnice:

$$\mathbf{w}_n = \mathbf{w} + \mathbf{xy}, \tag{1.4}$$

kde $\mathbf{y}$ je vektor výstupních hodnot.[7]

Dále se pro učení využívá například Delta pravidlo, které je vhodné k trénování neuronů s lineární aktivační funkcí, nebo učení podle Widrowa, kdy je představa taková, že se výstupní prostor neuronu s binárním výstupem dělí na dva podprostory a vstupní data se klasifikují vždy do jednoho z těchto podprostorů.[1]

Všechny tyto způsoby se však dělí do dvou základních skupin adaptačních algoritmů a to učení bez učitele a učení s učitelem.

1.4.1 Učení bez učitele

Typickým znakem těchto adaptačních algoritmů je, že při učení nemají k dispozici žádné kritérium, podle kterého by bylo možné určovat správnost interpretace vstupních dat.[1]

Základním principem je shlukování, dělení vstupních dat do skupin na základě určité podobnosti vstupních elementů. Síť nemá možnost kontroly správnosti rozdělení, může však mít předem určený počet tříd, do kterých bude vstupní data dělit.

Samotné učení je založeno zcela a pouze na informacích, které obsahují vstupní data.[1]

Hlavní skupina neuronových sítí, která tento způsob učení využívá, se nazývá jako samoorganizující se mapy.[1]

1.4.2 Učení s učitelem

Zásadní rozdíl mezi učením bez učitele a učením s učitelem je, že při učení s učitelem má adaptační algoritmus možnost ověřit si správnost interpretace vstupních dat. Nejtypičtějším způsobem kontroly je porovnání predikované výstupní třídy s třídou, které byla každému elementu vstupních dat přiřazena ve fázi předzpracování dat.[1]

1.5 Predikce

Jak již bylo zmíněno, každý neuron má své vstupní synapse ohodnocené určitou vahou **w**. Této vlastnosti neuronů se využívá při jejich trénování. Do vstupní vrstvy jsou předána trénovací data, model se pokusí vyhodnotit z těchto dat závěr a v závislosti na tom, zda-li byla vstupní data vyhodnocena správně či špatně, dochází k úpravě hodnot jednotlivých vah.[5]

Tento popsaný algoritmus je jedním z nejpoužívanějších při učení neuronových sítí. Říká se mu algoritmus zpětné propagace a dělí se na tři dílčí části:

- dopředné šíření vstupního signálu,
- zpětné šíření chyb,
- adaptace hodnot vah synapsí neuronu.

Při trénování sítě se tyto tři kroky opakují, dokud není dosaženo uspokojivých výsledků – dostatečně malé chybovosti, nastaveného limitu opakování nebo jiného kritéria.[7]

Chybovostí je myšlena odchylka predikované hodnoty na výstupu neuronové sítě a skutečné hodnoty. Pro výpočet této odchylky je však možné využít různých ztrátových funkcí. Tyto funkce se dělí do několika kategorií, mezi hlavní dvě patří regresivní ztrátové funkce a pravděpodobnostní ztrátové funkce. V praxi se potom nejčastěji využívá **mse** – mean squared error, česky střední kvadratická chyba.[21] Tu lze vypočítat podle rovnice:

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2, \tag{1.5}$$

kde m je počet vzorků, y_i je i-tá pravá hodnota a $\hat{y}_i$ je i-tá predikovaná hodnota. Další často využívanou ztrátovou funkcí je Binary Crossentrophy, neboli křížová

entropie, kterou je vhodné použít v případě, že vstupní data jsou dělena pouze na dvě rozdílné třídy.[21] Tuto funkci lze zapsat jako:

$$BCE = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})), \tag{1.6}$$

kde y je pravá hodnota nabývající 0 a 1 a $\hat{y}$ je predikovaná pravděpodobnost, že objekt spadá do skupiny 1.

Pokud je tříd více než dvě, je třeba využít Cathegorical Crossentropy[21], kdy rovnicí je zobecněný předpis křížové entropie:

$$CCE = -\sum_{i=0}^{M} (y_{o,i} \log(\hat{y}_{o,i}), \qquad o \in M, \tag{1.7}$$

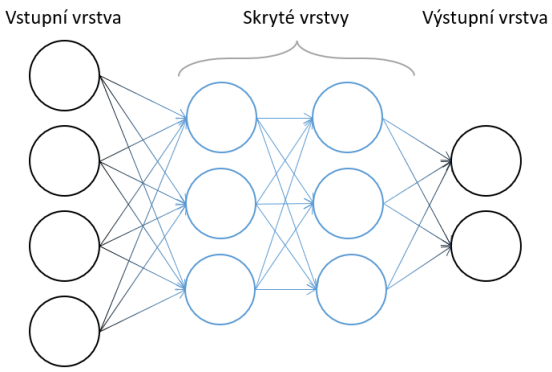
kde M je počet tříd, y_o je pravá hodnota třídy o a $\hat{y}_o$ je pravděpodobnost, že objekt spadá do dané třídy.

1.6 Základní druhy neuronových sítí

Moderní neuronové sítě je možné rozdělit na 3 základní druhy: ANN – Umělé neuronové sítě, CNN – Konvoluční neuronové sítě a RNN – Rekurzivní neuronové sítě.[17] Každá z těchto sítí má určité výhody a nevýhody, které budou probrány v následujících podkapitolách.

1.6.1 ANN – Umělé neuronové sítě

Jedná se o síť tvořenou vrstvami perceptronů či neuronů. Říká se jim také dopředné neuronové sítě, jelikož data neuronovou sítí putují ze vstupu, skrz několik skrytých vrstev, rovnou na výstup.[17][1]



Obr. 1.4: Topologie Dopředné neuronové sítě

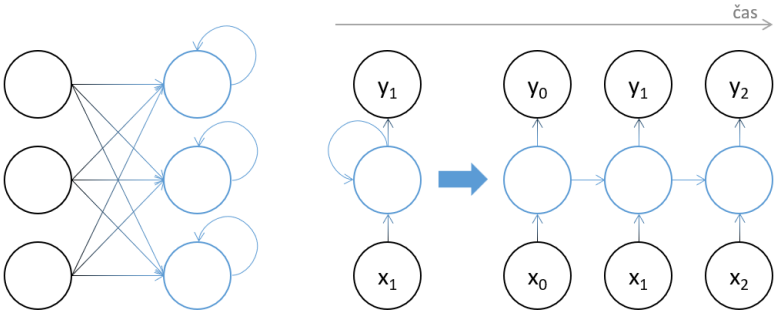
Jednotlivé vrstvy těchto sítí mohou být plně propojené, každý neuron jedné vrstvy je propojený se všemi neurony vrstvy následující, nebo neúplně propojené.[18]

Jejich využití je velice obecné, obvykle se využívají pro řešení problémů spojených s textovými nebo číselnými daty. Velkou výhodou je jejich schopnost aproximovat libovolnou nelineární funkci. Lze je využít i pro zpracovávání obrazových dat, mají však tu nevýhodu, že při procesu zpracování se ztrácí informace o rozmístění jednotlivých pixelů.[17]

Dopředné neuronové sítě využívají k úpravě vah neuronů algoritmus zpětné propagace, který byl popsán v kapitole 1.5. Z tohoto důvodu tyto sítě ztrácí schopnost využít sekvenční informaci k jejich učení.[17]

1.6.2 RNN – Rekurzivní neuronové sítě

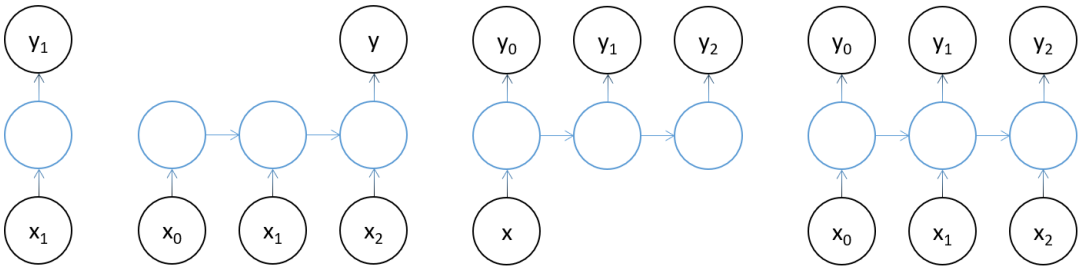
Stejně jako u ANN jsou vrstvy tvořeny určitým počtem neuronů. Hlavní rozdíl mezi těmito dvěma druhy sítí je, že jednotlivé neurony obsahují rekurzivní smyčku viz. obrázek 1.5, která těmto sítím umožňuje zachytit a zpracovat informaci o pořadí předávaných dat. Z tohoto důvodu jsou tyto sítě vhodné pro zpracování audia, mluvené řeči, textu a obecně všech dat, která se dají vyjádřit jako nějaká posloupnost hodnot.[17]



Obr. 1.5: Funkce Rekurzivní neuronové sítě

Na rozdíl od dopředných neuronových sítí, které mají vždy pouze jednu vstupní a jednu výstupní vrstvu, rekurzivní sítě mohou tvořit propojení jeden vstup ku více výstupům, více vstupů ku jednomu výstupu a i více vstupů ku více výstupům. Tyto propojení je možné vidět vyobrazené na obrázku 1.6.[16]

Další výhodou je, že výstupní data nejsou vázána pevně stanovenou délkou a jejich typ nemusí být vždy pouze hodnota reprezentující určitou třídu, do kterých jsou vstupní data dělena. Lze je využít ke klasifikaci, může se jednat například o určování sentimentu vět. Lze je však také využívat k překladu textu z jednoho jazyka do druhého nebo i k dynamickému vytváření popisů fotografií.[19]

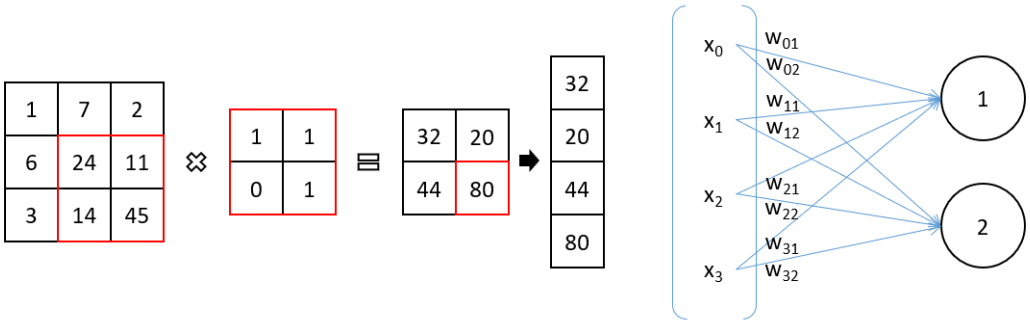


Obr. 1.6: Typy propojení neuronů Rekurzivní neuronové sítě

Tato architektura neuronových sítí má několik variant, které se v dnešní době využívají. Jsou to BRNN – Obousměrné rekurzivní neuronové sítě, které dokáží vylepšit svou přesnost zkoumáním vstupních dat v obou směrech. Následně využívají získané informace o začátku a konci sekvence k přesnějšímu určení hodnot nacházejících se mezi nimi. Dále se může jednat o LSTM – Long Short-Term Memory, ty obsahují ve svých skrytých vrstvách buňky, které dokáží uchovat informace o datech, jež se ve vstupní sekvenci vyskytovala ojediněle. Tato vlastnost LSTM sítím umožňuje lépe udržet kontext a spojitosti posloupnosti dat. Za zmínku také stojí GRUs – Gated recurrent units, které se ve své funkcionalitě podobají LSTM sítím, jen místo buněk využívají skrytých stavů.[16]

1.6.3 CNN – Konvoluční neuronové sítě

Na rozdíl od předchozích dvou typů neuronových sítí, tyto sítě nejsou tvořeny čistě vrstvami neuronů. K plně propojeným vrstvám je v tomto případě přidáno několik konvolučních vrstev.



Obr. 1.7: Konvoluce obrazu filtrem

Konvoluční vrstvu si lze představit jako filtr, který prochází vstupními daty. Jeho úkolem může být například potlačovat šum, zvýraznit rychlé přechody hodnot,

které se v daných datech nachází, a nebo ze vstupní sekvence vytřídit data, která nesou velké množství informace. Už z popisu může být zřejmé, že je tento druh neuronových sítí vhodný pro zpracování obrazových dat.[17] Mezi příklady využití lze potom uvést rozpoznávání obličejů, chytré kamerové systémy na křižovatkách nebo třeba analýza dokumentů a ručně psaného textu.[20]

Při trénování sítě se jednotlivé konvoluční vrstvy učí, jaký filtr by měly na vstupní data aplikovat, aby dosáhly správného výsledku. Dokáží tak zvýraznit a zachytit shluky dat, které jsou pro jednotlivé třídy charakteristické a provádět rozhodování na základě těchto vlastností.[17]

Obr. 1.8: Obraz zpracovaný konvolučním filtrem [1]

Jedním z druhů konvolučních sítí jsou Siamské neuronové sítě, které byly poprvé představeny kolem roku 1990 za účelem verifikace podpisů. Tyto sítě se skládají ze dvou paralelních konvolučních větví, které mají sdílené parametry. To zajišťuje konzistentní rozhodování a znamená to také, že je síť symetrická, tedy že obě větve vytvoří stejný proud výstupních dat pro daný vstupní obraz. Do každé z těchto větví vstupuje jeden z páru zkoumaných obrazů, výsledky po konvoluci se následně porovnají a na základě porovnání se rozhoduje o podobnosti vstupních dat.[8]

[1] Obrázek převzat ze článku [25].

2 Metody identifikace autora ručně psaného textu a jejich srovnání

Rozpoznávání autora ručně psaného textu je jedním z témat, které je v dnešní době značně populární. Jedná se o velmi složitý úkol, který se převážně uplatňuje v oborech zabývajících se biometrikou a forenzním zkoumáním dokumentů.[12] Velmi často se pro tuto problematiku využívá neuronových sítí díky jejich schopnosti naučit se a aproximovat téměř jakýkoliv algoritmus.

V roce 2017 byla představena práce *SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification*[12]. Smyslem této práce bylo navrhnout siamskou konvoluční síť, která by byla schopná rozeznat malé nesrovnalosti mezi originálním podpisem a padělkem. Autoři tuto síť nazvali SigNet. Tvořily ji čtyři konvoluční vrstvy s hloubkami 96, 256, 384 a 256 a velikostmi jádra 11×11, 5×5 a poslední dvě vrstvy 3×3. Mezi jednotlivými konvolučními vrstvami se nacházely kombinace vrstev provádějící normalizaci dat, MaxPooling, ReLU aktivační funkci a Dropout. V rámci předzpracování byl u textů invertovaný barevný prostor, hodnoty pixelů byly normalizovány podělením jejich hodnoty střední odchylkou maximální hodnoty pixelu a velikost byla zmenšena na 155×220 pixelů. Výsledky této práce byly do určité míry dobré. Na veřejné databázi CEDAR měla síť úspěšnost 100 %. V případě ostatních testovaných databází však schopnost správně rozpoznat originální a falšovaný podpis značně klesla viz. tabulka 2.2.

Další z prací, které byly představeny v témže roce je *Siamese Convolutional Neural Networks for Authorship Verification*[10]. Autoři této práce se pokoušeli navrhnout model, který by dokázal vyhodnotit jestli dvě představené ukázky pochází od stejného autora nebo ne. Zmínili se také ve své práci, že tato problematika nebyla doposud velmi studována. Jedním z důvodů může být, že se do určité míry jedná o složitější úlohu než klasifikace rukopisu. Při klasifikaci je neuronová síť trénována na rukopisu, o kterém se následně snaží určit, zdali je pravý či falešný. Jedná se tedy o model zaměřený na specifickou skupinu osob. Tato práce se však snaží nalézt obecný model, který by byl schopný porovnat texty od autorů, na kterých trénovaný nebyl. V rámci této práce byla představena siamská konvoluční síť, kterou autoři nazvali TinyResNet. Ta je založena na již existující síti ResNet s parametrem n, který v rámci ResNet sítě určuje její hloubku, nastaveným na hodnotu 1. Síť je tedy tvořena 8 vrstvami. Tato volba byla zdůvodněna tak, že na základě testování vyšší hloubka nepřinášela téměř žádné benefity. Navržený model byl testován na databázi IAM, kdy jednotlivá slova byla upravena na velikost 500x75 a věty na 1000×150 pixelů. Při testování autoři porovnali tento model s dalšími populárními modely jako je VGG13, GoogLeNet, Resnet3 a referenční Baseline model.

Výsledky testování je možné nalézt v tabulce 2.1.

Roku 2019 byla na mezinárodní konferenci IMITEC v Jihoafrické republice představena práce nazvaná *Author Identification from Handwritten Characters using Siamese CNN*[11]. Jejím cílem bylo navrhnout siamskou konvoluční síť, která by dokázala rozeznávat, zda ukázky pochází od stejného autora či nikoliv. Tato konvoluční síť byla navržena autory a nevyužívala žádné již existující ověřené struktury. Tvořily ji tři konvoluční vrstvy, dvě vrstvy MaxPooling a tři plně propojené vrstvy. Síť také využívala Euklidovskou metriku pro výpočet podobnosti výstupních tensorů a tedy finální vyhodnocení předložených ukázek. Pro tento experiment byl využit dataset NIST-SD19. Porovnávána byla písmena anglické abecedy a číslice 0-9 napsaná 100 autory, kdy každý z autorů zopakoval jednotlivé znaky 13 krát. Výsledná zpracovaná databáze byla tvořena 46 tisíci ukázkami textu, jejichž velikost byla 28×28 pixelů. Jednotlivé vytvořené datové sady, na kterých byl model následně trénován, byly nevyvážené. Uvedené výsledky jsou zaneseny do tabulky 2.1. Průměrná přesnost klasifikace výsledného modelu se pohybovala kolem 80 % a nejvyšší úspěšnost síť prokazovala u písmene A, kdy výsledná přesnost byla 88 %.

Nová end-to-end učící metoda byla představena v rámci práce nazvané *Writer Verification using CNN Feature Extraction*[13], která byla vydána v roce 2018. Tato metoda je založena na extrahování statistických informací ze sady ukázek. Obykle se při trénování využívá sample-to-sample přístup, kdy jsou neuronové síti předkládány jednotlivé ukázky. V rámci této práce autoři přednatrénovali části neuronové sítě touto metodou a následně všechny části spojili, aby je bylo možné přetrénovali je jejich navrženou End-to-End metodou za účelem zvýšit přesnost rozhodování. Byla využita vlastní datová sada, která byla tvořena slovy „and“ od 1555 autorů s velikostí 32×60 pixelů a „th“ od 499 autorů o velikosti 32x32 pixelů. Výsledky práce byly velice překvapivé. Úspěšnost konvoluční sítě při přechodu ze sample-to-sample na end-to-end metodu vzrostla z 89,58 % na 98,93 % pro databázi tvořenou slovy „and“ a z 77,91 % na 95,71 % u daleko menší databáze tvořené slovy „th“.

V posledních letech se také značně rozšířilo množství prací, ve kterých se autor či autoři snaží určit osobnost člověka na základě jejich rukopisu. Jednou takovou prací je *Handwritten Texts for Personality Identification Using Convolutional Neural Networks*[15] představenou v roce 2018. Autoři představili vlastní model, který byl založený na již existujících modelech AlexNet a DeepWriter. Jako datovou sadu využili HWxPI, která byla vytvořena přímo pro tento způsob využití. Velikost vstupních dat byla omezena na 200x200 pixelů a jednalo se o slova, někdy pouze části slov. Autoři využili pro hodnocení AUC, tedy plochu pod křivkou. Výsledná hodnota AUC byla 0,5023. Autoři v závěru práce zmiňují, že se nejedná o velice výkonný klasifikátor a vysvětlují, že problémem byla malá datová sada.

Ze zmiňovaných prací je možné vidět, že konvoluční neuronové sítě jsou pro tuto

problematiku velmi často používány. Existuje však i několik jiných způsobů, jak problém rozpoznávání autora ručně psaného textu řešit.

Jednou takovou prací co přišla s řešením, které nevyužívá konvolučních sítí je *A graph-based solution for writer identification from handwritten text*[14]. Byla představena v roce 2022 a jedná se o řešení identifikace autorů založené na metodě grafů, kdy se z písmene, číslice či slova získá „kostra“ aproximací jeho podoby sadou bodů. Následně se z tohoto grafu bodů extrahují detailní informace o rukopisu, na základě kterých se dále provádí klasifikace. Pro testování navrženého modelu byly využity datové sady jako CERUG-EN, CVL, Firemaker, IAM a vlastní datová sada. Tento přístup a model se ukázal být úspěšný, přesnost odhadu se u jednotlivých sad pohybovala v rozmezí 92,8-94,6 %.

Výhodou těchto algoritmických přístupů je, že se dají velice dobře odladit a specifikovat na jednu určitou činnost. Jejich zobecnění však může být složitější, než se na první pohled zdá. Často se také objevují metody, které využívají tvaru slov nebo třeba takzvané Zernikeovy polynomy.

Z prozkoumaných prací se žádná nepokoušela o porovnávání odstavců textu. Nejvíce se tomuto experimentu přiblížili autoři práce [10], kteří dosáhli úspěšnosti 92,08 % při porovnávání vět.

Tab. 2.1: Tabulka srovnání výsledků prací

Použitá technologie	Rozměry vstupních dat	Databáze	Výsledná přesnost
Práce zabývající se porovnáváním autorů			
CNN model, VGG13, GoogLeNet, ResNet3, TinyResNet [10]	věty: 1000 × 150 px slova: 500 × 75 px	IAM	CNN model – 72,4 % VGG13 – 60,25 % GoogLeNet – 88,18 % ResNet3 – 89,30 % TinyResNet – 92,08 %
CNN model [11]	anglická písmena A-Z a číslice 0-9 28 × 28 px	NIST-SD19	A – 88 % B – 80 % C – 75 % 6 – 85 % 9 – 75 % 5 – 70 %

Tab. 2.2: Pokračování tabulky srovnání výsledků prací

Použitá technologie	Rozměry vstupních dat	Databáze	Výsledná přesnost
Práce zabývající se rozpoznávání autorů			
SigNet [12]	podpisy: 155 × 220 px	CEDAR, GPDS300, GPDS Synthetic Signature, BHSig260	CEDAR – 100 % GPDS300 – 76,83 % GPDS-SS – 77,76 % Bengálština – 86,11 % Hindština – 85,64 %
CNN [13], inovativní end-to-end učení	slova „and“: 32 × 60 px, slova „th“: 32 × 32 px	Vlastní	sample-to-sample: „and“ – 89,58 % „th“ – 77,91 % end-to-end: „and“ – 98,93 % „th“ – 95,71 %
Grafy [14]	slova číslice písmena	CERUG-EN, CVL, Firemaker, IAM, Vlastní	CERUG-EN – 92,85 % CVL – 93,12 % Firemaker – 93,44 % IAM – 94,61 % Vlastní – 93,01 %
Práce zkoumající osobnost z rukopisu autora			
CNN [15]	slova: 200 × 200 px	HWxPI	AUC – 0,5023

3 Popis praktické části

Tato práce byla vypracovávána v jazyce Python 3.10 a pro tvorbu a trénování modelů byly využity knihovny Tensorflow a Keras.

Trénované modely využívají technologie Siamských konvolučních neuronových sítí. U jednotlivých modelů byl zvolen přístup porovnávání párů fotografií s využitím kontrastivní ztrátové funkce namísto trojícové ztrátové funkce. Všechny konvoluční sítě byly implementovány vlastnoručně a nebylo využito předtrénovaných modelů.

3.1 Knihovny

Při vývoji byly využívány zejména tyto knihovny:

- Tensorflow a Keras pro tvorbu samotných modelů,
- OpenCV – pro načítání a úpravu fotografií,
- Numpy – pro matematické operace a tvorbu struktury fotografií v paměti,
- Scikit-learn – pro zamíchaní pořadí a dělení jednotlivých párů fotografií,
- Csv – pro ukládání rozdělených datových sad,
- Multiprocessing – pro dynamické načítání fotografií během trénování,
- Os a Glob – pro tvorbu složek a načítání jejich obsahu,
- Argparse – pro zpracování argumentů předávaných při spouštění trénování, či testování.

3.2 Databáze

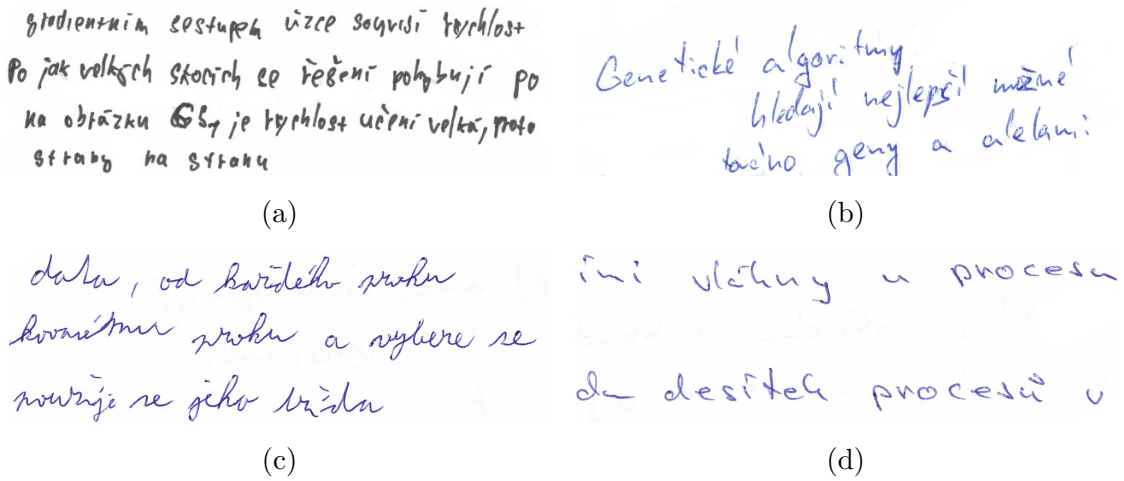
V rámci této práce byly vytvořeny a zpracovány dvě databáze tvořené českými a anglickými ukázkami textu.

3.2.1 Původní databáze

Materiály pro tuto databázi musely být zpracovány ručně, jelikož nebyly v takové podobě, aby se daly zpracovat programově. Celkově je databáze tvořena ze dvou částí.

Česká část

Tato část se skládá ze vzorků textu pocházejících od 26 různých osob s unikátním rukopisem a pro každou z těchto osob existuje 12-20 ukázek. Průměrně se jedná o 17 ukázek na osobu. Tato část databáze byla vytvořena z části starých testů studentů VUT a dohromady je tvořena 453 fotografiemi ručně psaného českého textu.

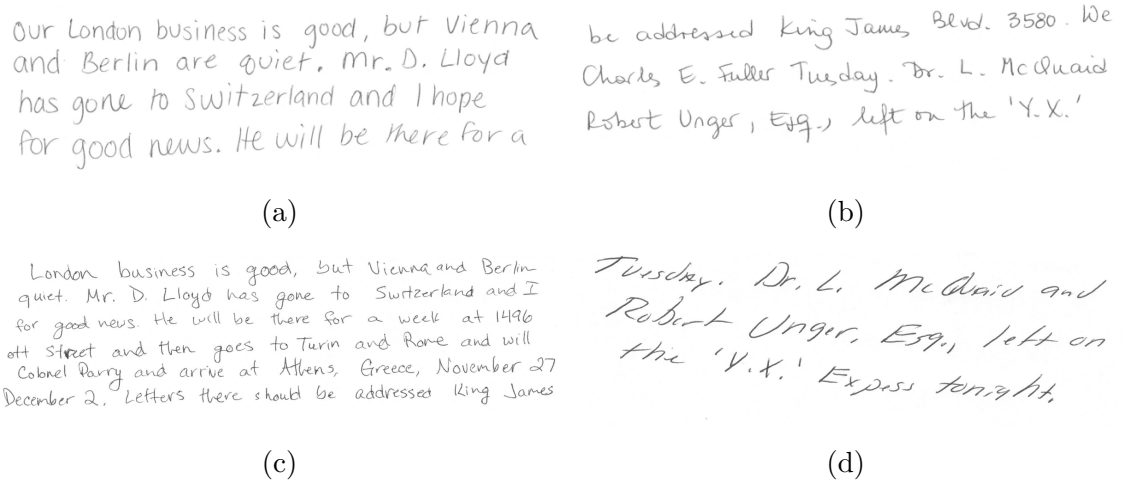


(a) (b) (c) (d)

Obr. 3.1: Ukázky textů české databáze

Anglická část

Anglická část byla vytvořena z veřejně dostupné databáze ručně psaných textů. Tu zprostředkovává společnost *CSAFE – Center for Statistics and Applications in Forensic Evidence* [9]. Účastníci byli při tvorbě této databáze požádání o přepsání tří různých odstavců textu. Každý z účastníků je opisoval vždy třikrát a v celkem třech sezeních. Odstup mezi jednotlivými sezeními byl stanoven na minimálně tři týdny.



(a) (b) (c) (d)

Obr. 3.2: Ukázky textů anglické databáze

Po zpracování je tato část tvořena 93 různými osobami s unikátním rukopisem. Pro každou z osob existuje 9-18 ukázek ručně psaného textu, průměrně se tedy jedná

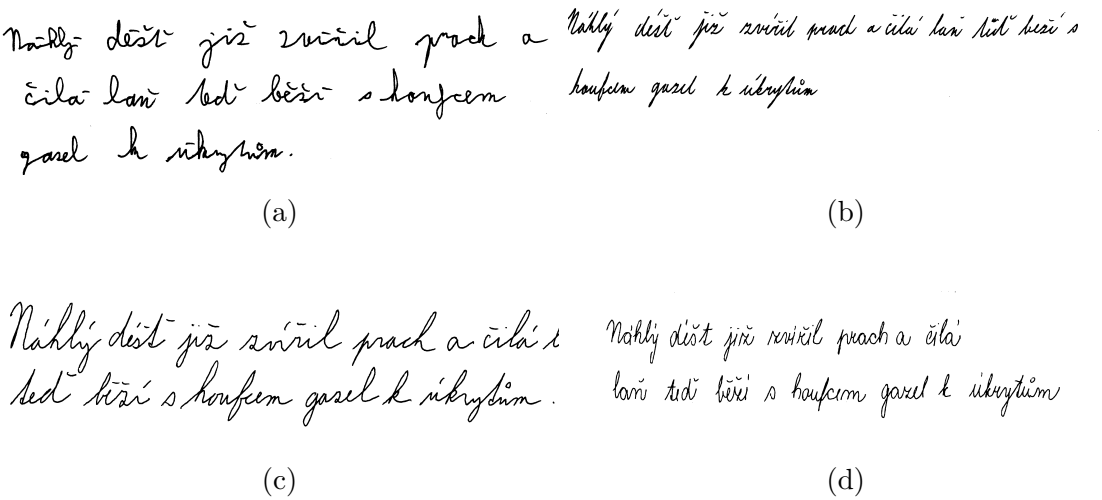
o 15 ukázek na osobu. Dohromady tuto část tvoří 1405 fotografií ručně psaného anglického textu.

Obě části databáze tvoří stromové struktury a všechny fotografie jsou uloženy v RGB formátu s rozlišením 2060 × 644 px.

Ukázalo se však, že použité materiály nebyly pro řešení tohoto problému vhodné. Jednotlivé ukázky byly příliš různorodé na to, aby se neuronová síť dokázala naučit rozeznávat detailní rozdíly ve stylu rukopisu. Z tohoto důvodu byla vytvořena nová databáze.

3.2.2 Nová databáze

Tato databáze se skládá z celkem 1328 ukázek od 332 různých autorů. Každý z autorů byl požádán, aby čtyřikrát opsal větu „Náhlý déšť již zvířil prach a čilá laň teď běží s houfcem gazel k úkrytům“. Jedná se o větu, která byla speciálně navržena tak, aby obsahovala všechny diakritická znaménka českého pravopisu.



(a) (b)

(c) (d)

Obr. 3.3: Ukázky textů druhé české databáze

To, že je tato databáze tvořena ukázkami, které obsahují stejný text, umožňuje neuronové síti, aby se zaměřila na detaily jednotlivých slov a písmen. Tedy problém, který se tato práce snaží řešit.

Všechny ukázky textu jsou uloženy ve složce v černobílém formátu s rozlišením 3660 × 1250 px a následně logicky děleny při tvorbě datových sad.

3.3 Zpracování a augmentace dat

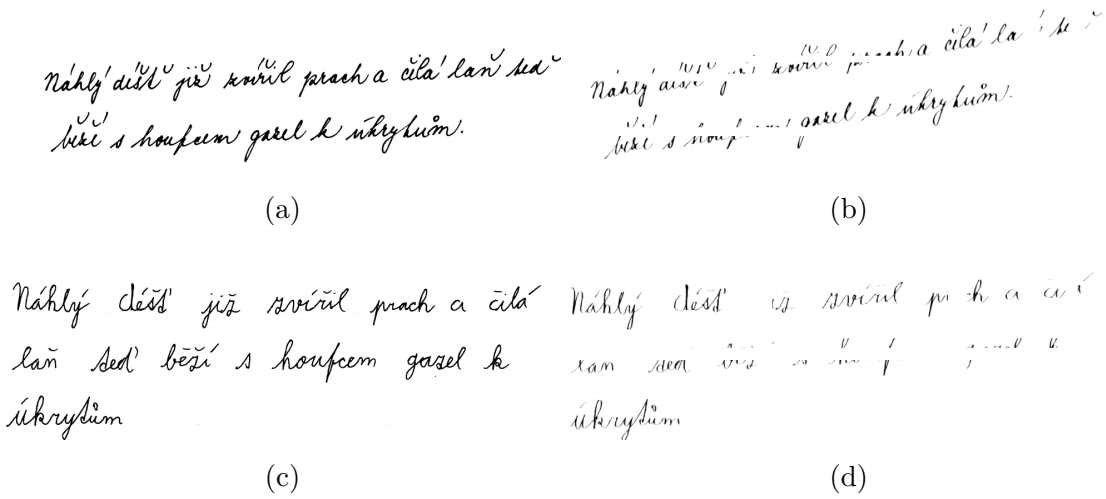
Fotografie s rozlišením 3660×1250 px jsou zbytečně velké a detailní. Z tohoto důvodu jsou před jejich využitím zmenšeny.

V rámci této práce byly všechny navržené modely natrénované na třech rozlišeních a to 750 × 256 px, 1000 × 342 px a 1250 × 428 px, aby bylo možné porovnat, jaký vliv bude mít rozlišení vstupních fotografií na výsledky modelů.

Během trénování se ukázalo, že celkové množství párů trénovací datové sady není dostačující. Sítě dosahovaly 100% přesnosti odhadu na trénovací datové sadě v prvních deseti epochách. Byly proto vytvořeny kopie všech fotografií a ty následně augmentovány. Celková velikost trénovací datové sady se tak zvětšila dvojnásobně.

Při procesu augmentace je využíváno několik metod, jak data upravit a zdeformovat tak, aby stále zachovávala stejnou podobu jako originál, ale nebyla jeho přesnou kopií.

Na obrázcích 3.4 jsou uvedené ukázky dvou textů před a po augmentaci algoritmem, který je popsaný níže.



(a) (b)

(c) (d)

Obr. 3.4: Ukázky textů před augmentací a po augmentaci

V prvním kroku jsou data za pomocí metody warpAffine z knihovny OpenCV pootočena o náhodný úhel ϕ z rozsahu $\phi = \langle -6°, 6° \rangle$. Následně dojde k jejich posunutí v osách x a y o hodnoty x_1 a y_1 z rozsahů $x_1 = \langle -6, 6 \rangle$ px a $y_1 = \langle -5, 5 \rangle$ px. V dalším kroku dochází k degradaci kvality textu. K tomuto procesu byly využity metody erode a dilate, které pochází rovněž z knihovny OpenCV. Velikost jádra pro tuto operaci je 5x5. Počet iterací závisí na rozlišení vstupní fotografie a je uveden v tabulce 3.1. V posledním kroku augmentace dochází k umazání několika částí obrázku ve formě pruhů po celé vertikální a nebo horizontální délce fotografie. Šířky

pruhů i pozice středu jsou voleny náhodně a jejich hodnoty jsou závislé na rozlišení fotografie. Počet umazávaných pruhů je uveden v tabulce 3.1 a uvedený počet opakování je proveden jak horizontálně, tak i vertikálně.

Tab. 3.1: Tabulka předzpracování obrazu

Rozlišení	Počet iterací degradace	Počet odmazávaných pruhů	Šířka odmazávaného pruhu
750×256 px	2	2	$\langle 15, 37 \rangle$ px
1000×342 px	3	3	$\langle 20, 50 \rangle$ px
1250×428 px	5	4	$\langle 25, 62 \rangle$ px

3.4 Tvorba datových sad

Proces tvorby datových sad lze rozdělit do několika základních kroků:

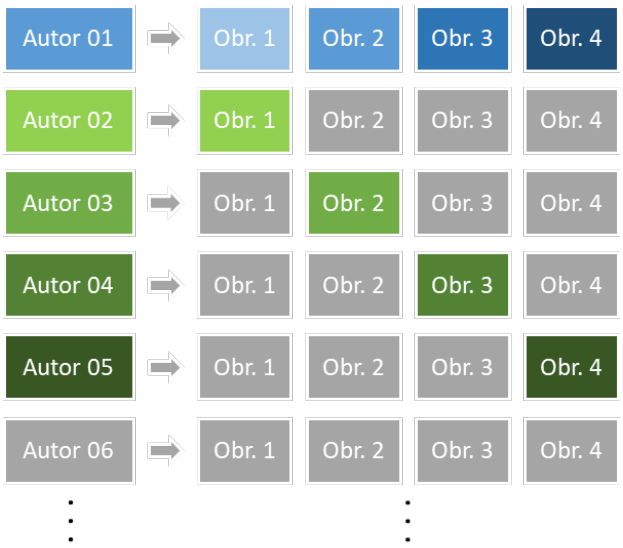
1. Načtení názvů fotografií a k nim příslušných cest.
2. Tvorba párů fotografií pocházejících od stejného autora.
3. Tvorba párů pocházejících od různých autorů.
4. Rozdělení vytvořených párů do trénovací, validační a testovací datové sady.
5. Rozšíření trénovací datové sady o páry tvořené augmentovanými fotografiemi.
6. Uložení cest vytvořených párů do příslušných *.csv* souborů.

Tvorba párů

Páry od stejného autora jsou tvořeny velice jednoduše. Proces je takový, že se provedou kombinace každé fotografie s každou fotografií. Výsledný list stejných párů je tedy tvořen $4^2 \times 332$ páry fotografií.

Páry od různých autorů je možné tvořit různými způsoby. Nejjednodušším z nich by bylo vytvořit kombinace všech fotografií každého autora s každým dalším autorem. V takovém případě by se jednalo o celkem $4^2 \times 332^2$ párů. S tímto přístupem by však výsledná datová sada byla velice nevyvážená, jelikož by celkové zastoupení rozdílných párů bylo 332-krát vyšší než stejných párů. Proto byl zvolen takový přístup, při kterém jsou vytvářeny páry kombinací všech čtyř fotografií jednoho autora se čtyřmi fotografiemi, které pochází od čtyř následujících autorů. Na obrázku 3.5 je vyobrazena jedna iterace tohoto procesu.

S využitím tohoto přístupu bude výsledný list rozdílných párů obsahovat $4^2 \times 332$ párů. To znamená, že výsledné datové sady budou nejen vyvážené, ale také zastoupení jednotlivých fotografií bude pro všechny autory stejné.



Obr. 3.5: Tvorba párů od rozdílných autorů

Dělení párů

Takto vytvořené listy jsou následně děleny na jednotlivé datové sady. K dělení je využita funkce train_test_split z knihovny Scikit-learn, která umožňuje list rozdělit ve stanoveném poměru a potenciálně i zamíchat pořadí jednotlivých párů. Tato funkcionalita však není využívána, jelikož je důležité, aby validační a testovací sady neobsahovaly ukázky od autorů, kteří jsou obsaženi v sadě trénovací.

List je v prvním kroku rozdělen na testovací a zbytkovou sadu, kdy testovací sada obsahuje ukázky od přesně 60 autorů. V následujícím kroku je zbytková sada rozdělena na trénovací a validační sadu, kdy validační obsahuje ukázky od 30 autorů. Poměr rozdělení je počítán podle rovnice

$$velikost\ sady = \frac{požadovaný\ počet\ autorů \times 16}{celkový\ počet\ párů}.$$

Rozšíření a uložení

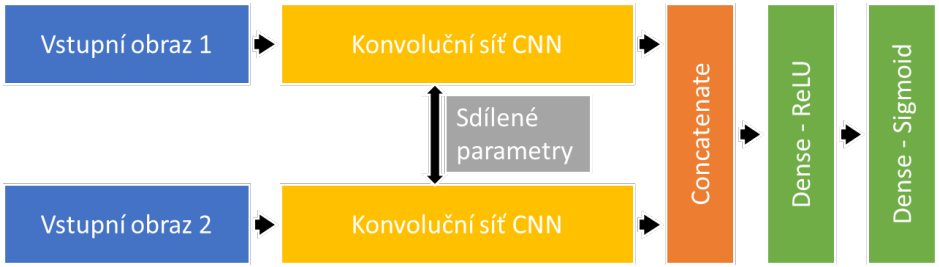
Totožným způsobem jsou zpracovány a rozděleny páry tvořené originálními a augmentovanými fotografiemi. Avšak, aby nedocházelo k ovlivňování výsledků testování a validace datové sady, je využita pouze část augmentovaných fotografií určených k trénování modelů. Tato augmentovaná sada je následně připojena k trénovací sadě originálních ukázek.

Po rozšíření jsou jednotlivé sady, které jsou tvořeny cestami a informacemi o tom, zdali se jedná o stejný nebo rozdílný pár, uloženy do *.svg* souborů. Tyto soubory jsou nadále využívány pro dynamické načítání fotografií při trénování, validaci či testování modelů.

3.5 Popis navržených modelů

3.5.1 Prvotní modely

Jedná se o tři navržené modely které, jak již bylo zmíněno, spadají pod skupinu siamský konvolučních sítí. Tyto modely mají podobnou strukturu, liší se však počtem konvolučních vrstev. Dále jejich hloubkou, velikostí konvolučních jader, využívaným optimizátorem, rychlostí učení a dalšími parametry. Základní podoba modelů je vyobrazena na obrázku 3.6.



Obr. 3.6: Základní struktura modelů

Pár obrazů vstupuje do konvoluční části sítě, která má sdílené parametry a jsou zpracovány nezávisle na sobě. Jejich výstupní vektory se spojí do jednoho výsledného vektoru. Ten je předán dopředné neuronové síti tvořené až třemi vrstvami neuronů. Poslední z těchto tří vrstev obsahuje pouze jeden neuron, který na výstupu sítě udává hodnotu s jakou pravděpodobností mají oba vstupní obrazy stejného autora.

Model 1

První model je velice jednoduchý, jeho konvoluční síť je tvořena čtyřmi konvolučními vrstvami s hloubkou filtrů 32, 64, 128 a 256. Velikost konvolučního jádra je 5×5 pro první vrstvu a 3×3 pro následující tři vrstvy. Po každé provedené konvoluci se výsledky normalizují a na normalizované výsledky je aplikována aktivační funkce ReLU. Model nevyužívá žádné vrstvy pro filtraci výsledků, těmi můžou být například vrstvy MaxPooling nebo Dropout. Po provedení těchto čtyř konvolucí je na výsledný vektor dat aplikován GlobalAveragePooling, který vypočte průměrnou hodnotu tensoru pro každý z 256 výstupních filtrů poslední konvoluční vrstvy. Data jsou předána vrstvě neuronů, která využívavá jako svou aktivační funkci ReLU. Po aktivaci jsou výstupní vektory obou zpracovaných obrázků spojeny a zpracovány plně propojenou sítí neuronů. Struktura modelu je vyobrazena na obrázku 3.7.

Obr. 3.7: Struktura konvoluční části modelu 1

Model 2

Druhý model má konvoluční síť tvořenou třemi konvolučními vrstvami s hloubkou 64, 128 a 256 filtrů. Jádra všech tří konvolucí mají svoji velikost nastavenou na 5×5. Hodnoty jádra jsou inicializovány metodou RandomNormal a při učení dochází k jejich regulaci funkcí L2 s předávaným parametrem 0,0001. Po každé z konvolucí se provádí normalizace, je aplikována ReLU aktivační funkce a následně se provádí MaxPooling s velikostí jádra 2×2 a krokem 2×2. Na výsledný vektor je aplikován GlobalAveragePooling. Průměrné hodnoty po zpracování jednotlivými filtry jsou předány vrstvě neuronů, která má váhy inicializované metodou HE_Normal a jako aktivační funkci využívají ReLU. Podoba konvoluční části tohoto modelu se nachází na obrázku 3.8.

Obr. 3.8: Struktura konvoluční části modelu 2

Model 3

Třetí navržený model je tvořen konvoluční sítí se čtyřmi konvolučními vrstvami. Ty mají hloubky nastavené na 48, 96, 192 a 192 filtrů. Jádro první konvoluční vrstvy má velikost 5×5 a jádra následujících tří vrstev mají velikost 3×3. Hodnoty konvolučních vrstev jsou inicializované, stejně jako u druhého modelu, metodou

RandomNormal a regulovány funkcí L2 s předaným parametrem 0,001. Po každé z konvolucí se provádí normalizace, je aplikována aktivační funkce ReLU a následně je provedený Dropout s parametrem 0,3, kdy dochází k zahození náhodných 30 % dat. Na výstupní vektor čtvrté konvoluce je aplikován GlobalAveragePooling a data se dále předávají výstupní vrstvě konvoluční sítě. Ta je tvořena vrstvou neuronů, které mají váhy inicializované metodou HE_Normal a jako aktivační funkce je využívané ReLU.

Obr. 3.9: Struktura konvoluční části modelu 3

3.5.2 Finální modely

Tyto modely využívají rozličné, ve většině případů již ověřené a známé konvoluční sítě jako základ pro konvoluční část siamského modelu. Avšak některé vlastnosti jednotlivých struktur byly poupravené tak, aby se dokázaly lépe zaměřit na řešený problém.

Základní model

Jedná se o vlastní model, který byl navržen na základě tří prvotních modelů a nevyužívá žádné již existující nebo otestované struktury.

Stejně jako u modelu 1 (3.5.1) konvoluční část siamské sítě obsahuje 4 bloky. Ty jsou však tvořené konvoluční vrstvou, normalizací, ReLU aktivací a MaxPooling vrstvou. Hloubky, tedy počty filtrů konvolučních vrstev jednotlivých bloků, jsou 64, 128, 256 a 512. Velikost jádra konvoluční vrstvy pro první blok je 7×7. Druhý blok má velikost jádra 5×5 a třetí a čtvrtý 3×3. Krok, o kolik se bude filtr posouvat s každou iterací, je pro první tři bloky nastaven na hodnotu 1×1. Čtvrtý a poslední blok má krok nastaven na hodnotu 2×2. Co se MaxPooling vrstev týče, tak všechny vybírají z 2×2 pole prvků s krokem nastaveným na 2×2. Jedinou výjimkou je první blok, který vybírá z pole o velikosti 3×3.

Obr. 3.10: Struktura konvoluční vrstvy základního modelu

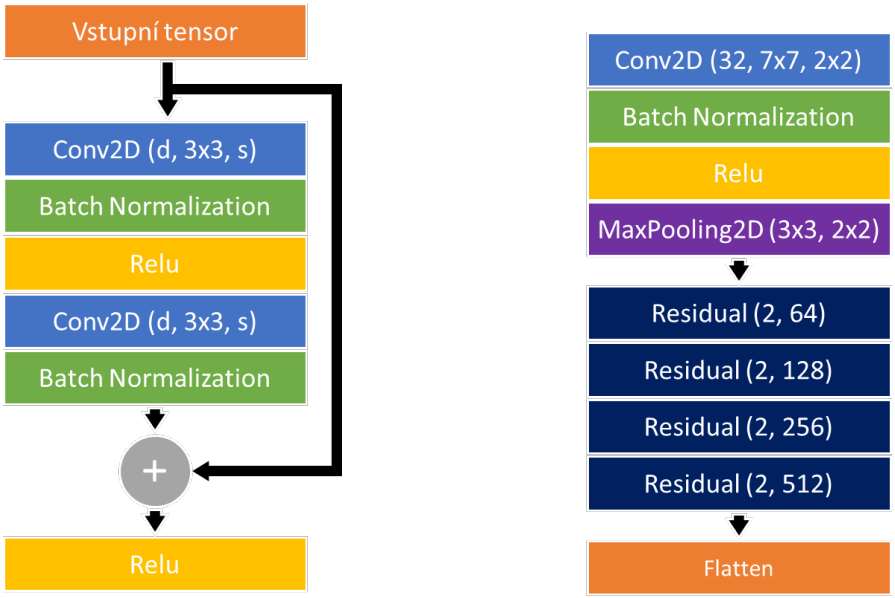
V čem se tento model liší je zakončení konvoluční vrstvy. Prvotní modely byly zakončené vrstvou GlobalAveragePooling. Ta však z výstupních dat odstraňovala velké množství důležitých informací, jelikož finální hodnoty průměrovala. Proto byla pro tento model použita vrstva Flatten. Ta z výstupního tensoru nepočítá průměrnou hodnotu pro každý výstup filtru, ale pouze převede tento tensor na jednorozměrné pole.

Model založený na síti ResNet18

Tento model využívá jako svou konvoluční část mírně poupravenou síť ResNet18, která je jednou z populárních a osvědčených sítí využívaných pro zpracování obrazu. Sítě Resnet využívají jako základní stavební kámen takzvaný Residuální blok, který se následně spojuje do sekvencí.

Jeden Residuální blok je tvořen dvěma konvolučními vrstvami s velikostí jádra 3×3 a hloubkou d. Ta je pro všechny Residuální bloky v sekvenci stejná a je specifikovaná samotnou strukturou modelu. Krok s obvykle bývá nastaven na hodnotu 1×1 s výjimkou prvního bloku v každé sekvenci, kdy je krok nastaven na hodnotu 2×2. Specifickou vlastností je sčítání vstupního a výstupního tensoru na výstupu bloku. Tato vlastnost upravuje chování této struktury tak, že nedochází k hledání aproximační funkce $f(x)$, jak by tomu bylo běžně. Dochází však k hledání aproximační funkce $f(x) + x$, která je obvykle snazší na vyřešení. To dělá sítě ResNet tak účinné při řešení podobných problémů. Struktura jednoho bloku je vyobrazena na obrázku 3.11a.

Samotný model se skládá z jedné konvoluční vrstvy s hloubkou 32, velikostí jádra 7×7 a krokem 2×2. Výstupní tensor je normalizován, provádí se aktivace ReLU a MaxPooling s jádrem 3×3 a krokem 2×2. Takto je obraz předzpracován a následují čtyři sekvence Residuálních bloků. Jak lze vidět na obrázku 3.11b, každá ze sekvencí je tvořena dvěma Residuálními bloky a hloubka vnitřích konvolučních vrstev je nastavena na hodnoty 64, 128, 256 a 512 filtrů. Na výstupní tensor



(a) Residuální blok

(b) Upravený model Resnet18

Obr. 3.11: Struktura Residuálního bloku a upraveného modelu Resnet18

poslední sekvence se obvykle aplikuje GlobalAveragePooling. Ten však byl v této implementaci nahrazen vrstvou Flatten pro zachování většího množství informací.

Model založený na síti ResNet34

Síť ResNet34, která byla využita jako konvoluční část pro tento model, je velice podobná síti ResNet18. Obě sítě využívají stejný Residuální blok, který je uvedený na obrázku 3.11a. Co se týče finální struktury konvoluční části modelu, tak jsou taktéž velice podobné. Jediným rozdílem je počet Residuálních bloků v každé ze sekvencí. Zatím co ResNet18 má 2 bloky v každé ze sekvencí, ResNet34 jich má více. Sekvence s hloubkou 64 filtrů je tvořena třemi bloky, sekvence s hloubkou 128 filtrů čtyřmi bloky, šest Residuálních bloků tvoří sekvenci s hloubkou 256 filtrů a tři bloky sekvenci s hloubkou 512 filtrů.

Model založený na síti ResNet50

Zatím co menší sítě ResNet18 a ResNet34 mají Residuální blok založený na dvou konvolučních vrstvách, v sítích ResNet50 a obecně větších sítích ResNet, se využívají Residuální bloky založené na třech konvolučních vrstvách.

Počet vrstev však není jediným rozdílem mezi jednotlivými strukturami. Jak je vidět na obrázku 3.12, velikost jádra první a třetí konvoluce je pouze 1×1 na rozdíl od Residuálního bloku menších ResNet sítí, který má jádro vždy velikost 3×3.

Dalším rozdílem je, že hloubka nové – třetí konvoluční vrstvy je čtyřikrát vyšší než hloubka ostatních konvolučních vrstev bloku. To umožňuje síti se lépe zaměřit na větší objekty v obraze. Co naopak zůstává stejné je krok s, který se řídí stejnými pravidly jako u menších sítí.

Obr. 3.12: Struktura Residuálního bloku větších modelů ResNet

Struktura samotného modelu je identická s modelem založeným na síti ResNet34 a je vyobrazena na obrázku 3.13. Tedy počet Residuálních bloků v jednotlivých sekvencích má velikost tři pro 64 filtrů, čtyři pro 128 filtrů, šest pro hloubku 256 filtrů a tři pro 512 filtrů. Hlavním rozdílem je tedy struktura samotného Residuálního bloku, který daná síť využívá.

Obr. 3.13: Struktura upraveného modelu ResNet50

Model založený na síti VGG16

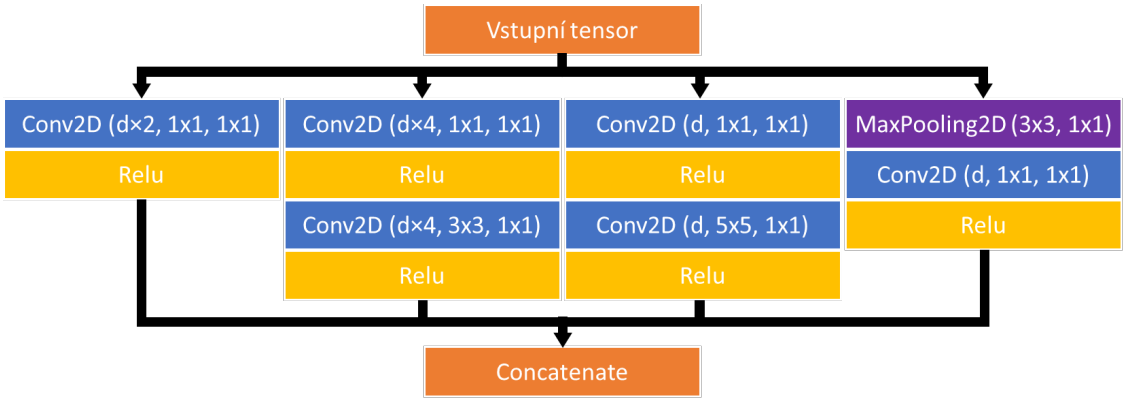
VGG je jedna z nejznámějších struktur konvolučních neuronových sítí. Jedná se o relativně jednoduchou lineární strukturu, která však podává velmi dobré výsledky při konvolučním zpracování obrazu. Tento model využívá konvoluční část šestnácti-vrstvé sítě VGG16 a je zakončen jedním ze dvou zakončení popsaných v sekci 3.5.3.

Síť je tvořena z pěti logických bloků, kdy každý z těchto bloků je zakončený MaxPooling vrstvou s jádrem a krokem o velikost 2×2. První a druhý blok je každý tvořen dvěma konvolučními vrstvami. Hloubky těchto vrstev jsou 64 filtrů pro

první a 128 filtrů pro druhý blok. Třetí, čtvrtý a pátý blok je každý tvořen třemi konvolučními vrstvami. Hloubky pro všechny tři konvoluční vrstvy v rámci třetího bloku jsou 256, v rámci čtvrtého a pátého bloku 512 filtrů. Všechny konvoluční vrstvy v modelu mají velikost jádra 5×5, jejich krok nastavený na hodnotu 1×1 a jejich výstupní tensor je vždy zpracován aktivací ReLU.

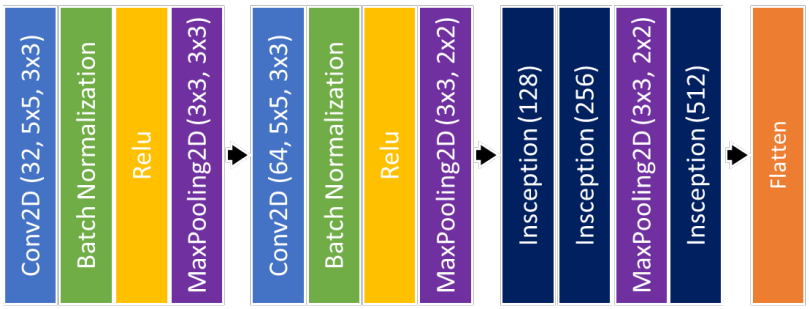
Model založený na síti GoogLeNet

Síť GoogLeNet, vyvinutá společností Google, je založená na takzvaném Insception bloku. Ten je vyobrazen na obrázku 3.14. První věc, které si lze všimnou je, že se tato struktura výrazně liší od například již uvedeného Residuálního bloku využívaného sítěmi ResNet. Vstupní tensor je zpracováván v několika paralelních větvích namísto jedné sekvenční. Účelem každé z těchto větví je se zaměřit na odlišné detaily ve vstupním tensoru dat. Jednotlivé větve se liší počtem konvolučních vrstev a také množstvím filtrů, které tyto vrstvy vytváří. Výsledky z těchto větví jsou po zpracování sloučeny do jednoho výstupního tensoru.



Obr. 3.14: Struktura Insception bloku

Samotný použitý model 3.15 je inspirovaný částí komplexní struktury modelu GoogLeNet, jedná se však o velice modifikovanou verzi. Pro zmenšení dimenzí a předzpracování dat jsou využity dvě konvoluční vrstvy s hloubkami 32 a 64 filtrů. Velikost jader těchto vrstev je 5×5 a krok je 3×3. Obě konvoluce jsou následovány normalizací dat, aktivací ReLU a MaxPooling vrstvou, která vybírá nejvyšší hodnotu z jádra o velikosti 3×3. Krok je pro první MaxPooling vrstvu nastaven na 3×3 a pro druhou 2×2. Následují první dvě Insception vrstvy. Ty mají nastavenou základní hloubku konvolučních vrstev na 128 a 256 filtrů. Dále jsou data zpracována třetí MaxPooling vrstvou, která má stejné parametry jako vrstva druhá. Po snížení množství informace v tensoru dat dochází k jeho předání poslední Insception vrstvě. Ta má hloubkou 512 filtrů a po zpracování dat je výstupní tensor převeden z n-rozměrného pole na jednorozměrné vrstvou Flatten.



Obr. 3.15: Struktura navrženého modelu GoogLeNet

3.5.3 Zakončení sítí

Prvotní modely měly zakončení tvořené vrstvou neuronů. Všechny využívaly binární křížovou korelaci jako ztrátovou funkci. Až na Model 1 (3.5.1), který využíval optimizér Adam s rychlostí učení 0,0001, využívaly modely optimizér SGD s rychlostí učení 0,01 a clipvalue nastavenou na hodnotu 0,5.

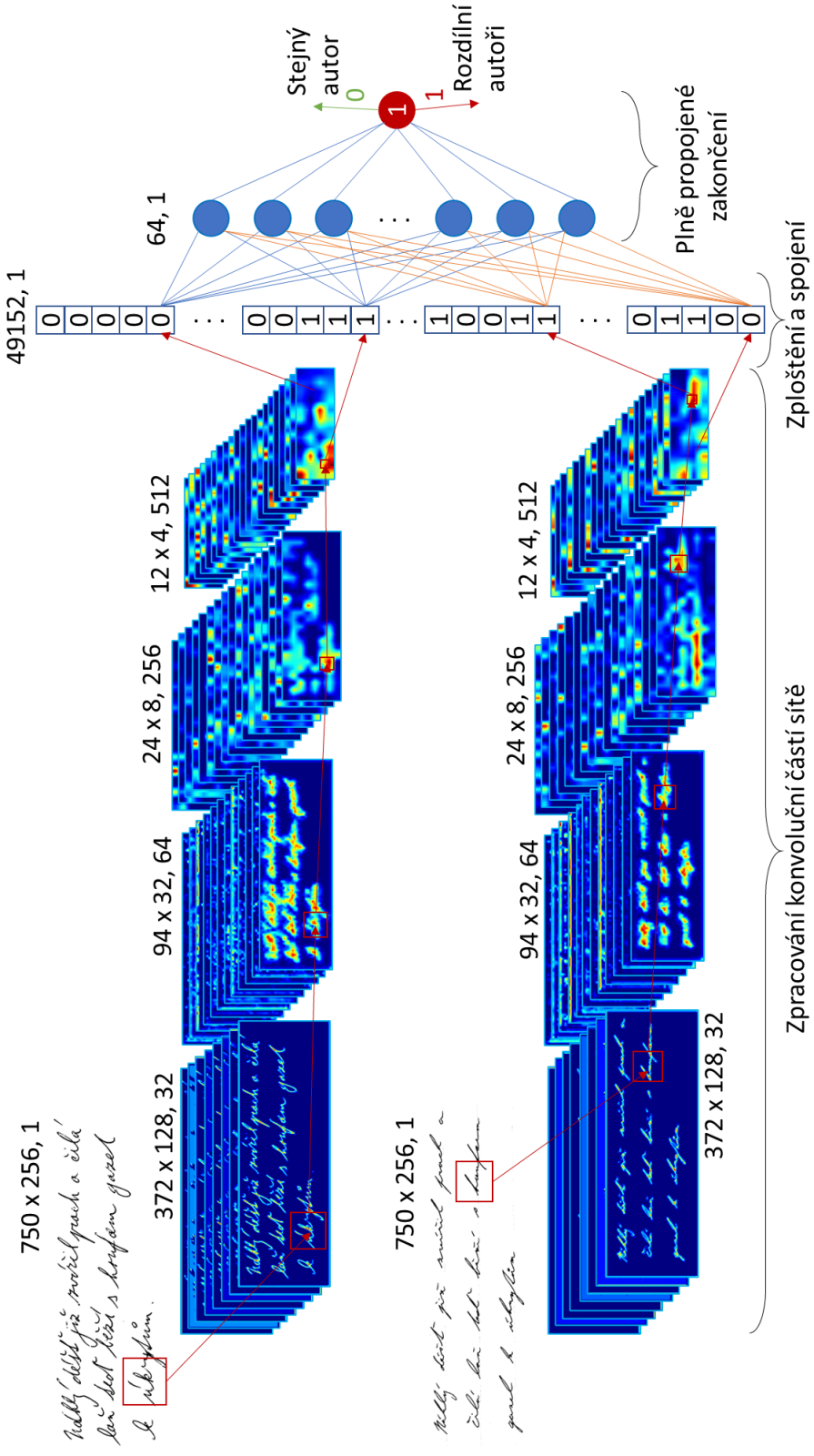
V případě finálních modelů byly provedeny změny. Každá ze sítí byla natrénována s dvěma různými zakončeními, aby bylo možné vidět, jak velké rozdíly toto zakončení vytváří.

V prvním případě, bylo zakončení tvořené plně propojenou sítí, která obsahovala vrstvu 64 neuronů. Namísto binární křížové korelace, kterou využívaly prvotní modely, byla využita kontrastivní ztrátová funkce a jako optimizér se osvědčilo SGD s rychlostí učení 0,001 a clipvalue nastavenou na hodnotu 0,5. Pro adresování tohoto stylu zakončení se pro zjednodušení bude používat zkratka FC z anglického výrazu *Fully Connected*.

V případě druhém, se z výstupních tensorů vypočítá euklidovská metrika, podle které se podobnost následně posuzuje. I pro tyto modely je využívána kontrastivní ztrátová funkce. Rozdílem je využitý optimizér, kdy v tomto případě je využit Adam s rychlostí učení 0,0001 a clipvalue nastavenou na hodnotu 0,5. Pro označování tohoto způsobu zakončení bude používána zkratka ED z anglického názvu *Euclidian Disctance*.

3.6 Schéma experimentu

Na obrázku 3.16 je uvedené schéma experimentu pro model ResNet18 s plně propojeným zakončením, ve které jsou porovnány dvě odlišné ukázky a model je vyhodnocuje jako rozdílné.



Obr. 3.16: Schéma experimentu porovnání dvou ukázek

3.7 Trénování

Prvotní sítě byly trénovány na původní anglické a české databázi. Jejich výsledky však nebyly dobré, a proto se u finálních modelů přešlo k trénování na databázi nové.

Knihovna tensorflow umožňuje trénovat modely jak na CPU, tak na GPU. Jelikož je během trénování na vstupních tensorech prováděno velké množství relativně jednoduchých operací jako je násobení a dělení, je trénování neuronových sítí na GPU podstatně rychlejší než na CPU.

Jedna iterace trénování se běžně označuje jako epocha. V rámci této epochy jsou neuronové síti postupně předloženy všechny ukázky určené k trénování. Pro menší modely a databáze lze všechny ukázky načíst do paměti a udržovat je připravené na moment, kdy je jich třeba. Avšak v momentě, kdy je model nebo databáze větší může nastat situace, kdy systém není schopný udrže jak model, tak ukázky načtené v paměti.

Z tohoto důvodu byly napsané třídy pro dynamické načítání dat za běhu trénováni. Obě implementované třídy dědí ze třídy Sequence v knihovně Keras a přepisují některé její dunder metody. První třída slouží k načítání dat při trénování s parametrem batch_size nastaveným na hodnotu 1. Tento parametr udává, kolik je zároveň zpracováno ukázek v jednom kroku epochy. V tomto případě, třída jednoduše načítá požadované obrázky na základě uvedených cest. Načítání je provedeno v hlavním vlákně programu a to kdykoliv, kdy je o ně zažádáno. Druhá třída slouží k načítání dat při trénování s parametrem batch_size nastaveným na hodnotu vyšší jak 1. V tomto případě by sekvenční načítání x ukázek zpomalilo celkovou dobu trénování sítě přibližně x-krát. Proto bylo využito knihovny multiprocessing a objektu pool, který umožňuje jednoduchou tvorbu a správu vláken procesu. S jejich využitím jsou jednotlivé páry ukázek načítány paralelně. Při vytvoření objektu této třídy dochází k vytvoření stejného počtu vláken jako je nastavená hodnota batch_size. To znamená, že při každém kroku epochy načte každé vlákno přesně jeden pár ukázek pro trénování.

Celková rychlost trénování je s využitím těchto tříd stejná, jako kdyby všechny ukázky byly již načtené v paměti. Celkové využití paměti RAM je při využívání paralelního načítání mnohonásobně nižší. Jelikož je ale spuštěno více vláken současně, tak je využití CPU vyšší.

3.8 Tvorba Attention map

Attention mapy jsou obrázky, které vyznačují ve vstupním obraze oblasti, na které se konvoluční síť zaměřuje. Jsou tvořeny vstupním obrazem a výstupním tensorem

některé z vrstev konvolučního modelu.

U běžných konvolučních sítí je tvorba těchto attention map jednoduchá. Avšak u siamských konvolučních sítí se to ukázalo být komplikovanější. Proto bylo nutné přijít se způsobem, jak z modelu získat potřebná data pro jejich tvorbu.

Jak již bylo zmíněno, siamské konvoluční sítě zpracovávají dva obrázky jednou sítí. Tímto způsobem dochází ke sdílení parametrů a síť se je schopná učit rozeznávat podobnost, nebo rozdílnost vstupních obrázků. Implementace velice jednoduché siamské konvoluční sítě může vypadat nějak takto 3.1:

```
from keras.layers import Conv2D, Input, ReLU
from keras.models import Sequential, Model

obrázek_a = Input(shape=(výška, šířka))
obrázek_b = Input(shape=(výška, šířka))

konvoluční_část = Sequential([
    Conv2D(64, name="vrstva1"), ReLU(),
    Conv2D(128, name="vrstva2"), ReLU()
], name="konvoluční_část")

tensor_obrázku_a = konvoluční_část(obrázek_a)
tensor_obrázku_b = konvoluční_část(obrázek_b)

# Porovnání výstupních tensorů...
output = ...

model = Model(
    inputs=[obrazek_a, obrazek_b], outputs=output
)
```

Výpis 3.1: Ukázka jednoduchého siamského modelu

Tato síť má definované dva vstupy nazvané `obrazek_a` a `obrazek_b`, u kterých je třeba určit velikost jednotlivých vstupů. Dále `konvoluční_část`, kterou tvoří dvě konvoluční vrstvy a dvě aktivace ReLU.

Na první pohled by se mohlo zdát, že stačí pouze sáhnout do konvoluční části modelu a uložit si výstup požadované vrstvy 3.2.

```
konvoluční_část = model.get_layer("konvoluční_část")
tensor1 = konvoluční_část.get_layer("vrstva1").output
```

Výpis 3.2: Nefunkční způsob získání výstupu vrstvy modelu

Tento přístup však nebude fungovat, jelikož je model spojován formou grafu a definice vstupního tensoru (`obrazek_a, obrazek_b`) pro `konvoluční_část` probíhá až po definici samotné konvoluční sítě 3.1. To znamená, že v momentě, kdy je získáván `tensor1` 3.2, konvoluční část sítě nemá určený její vstup, a i když je možné model zkompilovat, není na něm možné spustit testování.

Řešením tohoto problému je extrahovat vrstvy natrénovaného modelu a využít je k vytvoření nové konvoluční sítě. Ta bude mít pouze jeden vstup a bude definována funkcionálně. Tento způsob řešení umožní přístup k jednotlivým výstupním tensorů, které následně stačí uvést jako výstup sítě. Implementace by v případě uvedené sítě vypadala následovně 3.3:

```
obrazek_a = model.get_layer(index=0).input
konvoluční_část = model.get_layer("konvoluční_část")

tensor1 = konvoluční_část.layers[0](obrazek_a)
tensor2 = konvoluční_část.layers[1](tensor1)
tensor3 = konvoluční_část.layers[2](tensor2)
tensor4 = konvoluční_část.layers[3](tensor3)

nový_model = Model(
    inputs=obrazek_a, outputs=[tensor1, tensor3, tensor4]
)
```

Výpis 3.3: Funkční způsob získání výstupu vrstvy modelu

Po vytvoření tohoto modelu jej stačí pouze otestovat a výstupem sítě budou všechny tensory uvedené v parametru `outputs` nového modelu. Výsledné tensory budou uložené ve čtyřrozměrném poli. První dimenze adresuje jednotlivé výstupy, druhá a třetí adresuje jednotlivé hodnoty fotografie a čtvrtá z dimenzí adresuje výstupy z jednotlivých filtrů, které jsou definovány hloubkou konvolučních sítě. Takto získaná data lze následně lehce přetransformovat na attention mapy. Na ty je následně vhodné aplikovat nějaký barevný gradient pro lepší čitelnost. Obvykle se využívají tepelné mapy, kdy červené odstíny označují vyšší množství zaměření a modré nižší.

4 Výsledky práce a diskuze

V rámci této práce bylo natrénováno několik různých modelů siamských konvolučních sítí na dvou rozdílných databázích a pro několik rozdílných rozlišení vstupních fotografií. Cílem bylo porovnat, který model bude podávat nejlepší výsledky při pokusu o rozeznávání ručně psaného písma a jaký vliv na tyto výsledky má rozlišení vstupních obrázků.

Prvotní modely

Prvotní modely, zmíněné v sekci 3.5.1, byly natrénovány na původní databázi tvořené českými a anglickými ukázkami textu 3.2.1. Trénovací sada obsahovala 19900 párů, validační sada 4840 párů a testovací sada 4990 párů textu. Jednalo se o první pokusy, které nebyly velice úspěšné, jak je možné vidět v tabulce 4.1.

Tab. 4.1: Tabulka výsledků prvotních modelů

Rozlišení	Model	Zakončení	Ztrátovost	Přesnost odhadu
600 × 200	Model 1	plně propojené	3,26	66,1 %
	Model 2		2,37	**76,2** %
	Model 3		1,40	66,3 %

Nejlepší výsledek podával Model 2 s přesností odhadu 76,2 %. Je pravděpodobné, že se model nezaměřoval pouze na text samotný. Zaměřoval se pravděpodobněji na jeho pozici v obraze, vzdálenost od okrajů a jiné metriky, které lze ze dvou ukázek textu vyčíst. Z experimentování bylo zjištěno, že problém byl na vícero místech.

Původní databáze nebyla pro prvotní trénování vhodná. Kdyby modely byly již předtrénovány na rozsáhlé databázi ukázek, tak by se dalo předpokládat, že výsledky těchto modelů budou po natrénování na původní databázi podstatně lepší. Tyto navržené modely však byly trénovány od nuly, jednotlivé filtry a neurony měly hodnoty inicializované pouze náhodnými hodnotami.

Dalším z problémů se ukázala být použitá ztrátová funkce. Binární křížová korelace, byť užitečná při snaze klasifikovat data jako jednu ze dvou možných kombinací, se ukázala být nevhodnou ztrátovou funkcí pro klasifikaci takto komplexního problému. Z tohoto důvodu byla u finálních modelů použita kontrastivní ztátová funkce, která s použitými strukturami modelů podávala daleko lepší výsledky.

Posledním problémem se v některých případech ukázal být použitý optimizér. Ukázalo se, že optimizér Adam funguje velice špatně s jednoduchou vrstvou neuronů,

ale naopak podává dobré výsledky při trénování konvolučních sítí, které tyto vrstvy neobsahují. Optimizér SGD, neboli stochastický gradientní sestup, naopak funguje dobře s jednoduchými vrstvami neuronů, ale v případě čistě konvolučních sítí podává velmi špatné výsledky.

Finální modely

Finální modely, které jsou popsané v sekci 3.5.2 byly natrénovány na nově vytvořené databázi. Celkově tuto databázi tvořilo 18368 párů fotografií. Databáze byla rozdělena na 960 párů pro validační sadu, 1920 párů pro testovací sadu a 15488 párů pro trénovací sadu. Pouze trénovací sada obsahovala jak originální, tak augmentované páry fotografií. Rozdělení stejných a rozdílných ukázek ve všech sadách je přesně 50:50. Výsledky natrénovaných modelů pro rozlišení 750 × 256 pixelů je možné nalézt uvedené v tabulce 4.2.

Z tabulky je vidět, že nejlepšího výsledku pro toto rozlišení, dosáhl model založený na síti ResNet18, který dosáhl celkově nejvyšší přesnosti 92,2 % a využíval ED zakončení sítě. Další z modelů, které podávaly podobně úspěšné výsledky je ResNet50 s FC zakončením. Ten dosáhl výsledné přesnosti 90,1 %. Je nutné po-

Tab. 4.2: Tabulka výsledků finálních modelů pro rozlišení 750 × 256

Rozlišení	Model	Zakončení	Ztrátovost	Přesnost odhadu
750 × 256	Základní model	euklidovská metrika	0,530	50,6 %
		plně propojené	0,529	87,0 %
	VGG16	euklidovská metrika	0,109	89,9 %
		plně propojené	0,482	89,5 %
	ResNet18	euklidovská metrika	0,111	**92,2** %
		plně propojené	2,178	86,1 %
	ResNet34	euklidovská metrika	0,148	83,3 %
		plně propojené	3,981	87,8 %
	ResNet50	euklidovská metrika	0,296	87,1 %
		plně propojené	9,314	90,1 %
	GoogLeNet	euklidovská metrika	0,250	50,0 %
		plně propojené	6,030	74,4 %

dotknout, že tento model preferoval odhadovat, že páry ukázek jsou rozdílné. To pro praktické využití není nejvhodnější. Je však lepší, když model o stejných ukázkách řekne, že jsou rozdílné, než když o různých ukázkách řekne, že jsou stejné.

Za zmínku také stojí model využívající síť VGG16. Oba modely dosáhly výsledku pouze o pár desetin procent horšího než výsledek sítě ResNet50 s FC zakončením.

Nejhorší výsledky naopak podal základní model a také model, který využíval strukturu GoogLeNet jako svou konvoluční část. Obě struktury využívaly ED zakončení. Přesný důvod, proč se sítě nebyly schopné naučit rozeznávat alespoň část ukázek, lze těžko určit.

Do tabulky 4.3 byly zaneseny výsledky sítí natrénovaných na ukázkách s rozlišením 1000×342 pixelů.

Nejlepšího výsledku pro toto rozlišení znovu dosáhl model založený na síti ResNet18. Jeho přesnost dosahovala 88,6 %. Byť je to nejlepší výsledek pro dané rozlišení, je o téměř 4 % horší, než když byla tato síť natrénována na rozlišení 750×256. Dalších sítí, které dosáhly podobného výsledku jako nejlepší model v dané kategorii, je v porovnání s tabulkou 4.2 více. Patří mezi ně modely založené na sítích VGG16, ResNet18 a GoogLeNet s FC zakončením a také ResNet50 s ED zakončením.

Tab. 4.3: Tabulka výsledků finálních modelů pro rozlišení 1000×342

Rozlišení	Model	Zakončení	Ztrátovost	Přesnost odhadu
1000×342	Základní model	euklidovská metrika	0,532	62,5 %
		plně propojené	0,557	83,1 %
	VGG16	euklidovská metrika	0,115	85,7 %
		plně propojené	0,496	88,4 %
	ResNet18	euklidovská metrika	0,162	**88,6** %
		plně propojené	1,831	88,4 %
	ResNet34	euklidovská metrika	0,148	87,7 %
		plně propojené	3,978	84,1 %
	ResNet50	euklidovská metrika	0,175	88,5 %
		plně propojené	9,457	50,0 %
	GoogLeNet	euklidovská metrika	0,165	79,3 %
		plně propojené	5,990	88,2 %

Jediným modelem, který se nebyl schopný ani částečně naučit rozeznávat jednotlivé páry, byl ResNet50 s FC zakončením. Tento výsledek je velice zajímavý, jelikož pro rozlišení 750×256 se jednalo o druhý nejlepší model. Dalším z modelů, které nepodávaly dobré výsledky je základní model využívající ED.

V tabulce 4.4 lze nalézt výsledky modelů natrénovaných na databázi ukázek s rozlišením 1250×427 pixelů.

Pro toto rozlišení nejlepšího výsledku překvapivě nedosáhl model založený na struktuře ResNet18. Dosáhl jej model založený na síti GoogLeNet s přesností odhadu 89,0 %. Druhý v pořadí byl model ResNet18 využívající ED jako zakončení sítě.

Tab. 4.4: Tabulka výsledků finálních modelů pro rozlišení 1250 × 427

Rozlišení	Model	Zakončení	Ztrátovost	Přesnost odhadu
1250 × 427	Základní model	euklidovská metrika	0,822	50,0 %
		plně propojené	0,539	84,3 %
	VGG16	euklidovská metrika	0,250	50,0 %
		plně propojené	0,484	88,5 %
	ResNet18	euklidovská metrika	0,139	88,2 %
		plně propojené	2,169	84,4 %
	ResNet34	euklidovská metrika	0,134	87,4 %
		plně propojené	3,964	86,7 %
	ResNet50	euklidovská metrika	9,458	63,6 %
		plně propojené	0,250	50,0 %
	GoogLeNet	euklidovská metrika	0,266	68,4 %
		plně propojené	5,849	**89,0** %

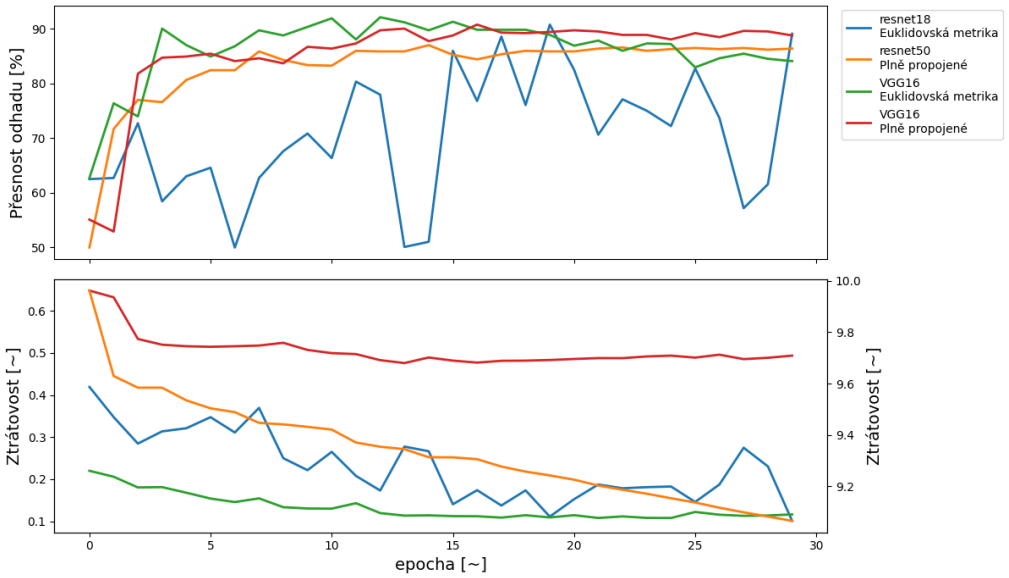
Modelů, které se pro toto rozlišení nedokázaly naučit rozeznávat ukázky, je několik. Patří mezi ně základní model a model založený na struktuře VGG16, které využívají ED a také model založený na síti ResNet50 s FC zakončením. Mezi modely, které nepodávaly dobré výsledky patří GoogLeNet a ResNet50 využívající ED jako jejich zakončení.

Ze všech natrénovaných modelů dosáhl nejlepšího výsledku model ResNet18 s ED natrénovaný na datové sadě s rozlišením 750×256. Dosáhl přesnosti 92,2 % a podával celkově dobré výsledky i pro ostatní rozlišení. Avšak z grafů trénování 4.1,4.3 je vidět, že tyto výsledky nebyly stabilní. Naopak nejhorších výsledků dosáhl základní model, který taktéž využíval ED. Je vidět, že výstupní tensory konvoluční části tohoto modelu nelze jednoduše porovnat. V momentě, kdy bylo zakončení vyměněno za FC, model začal dosahovat daleko obstojnějších výsledků.

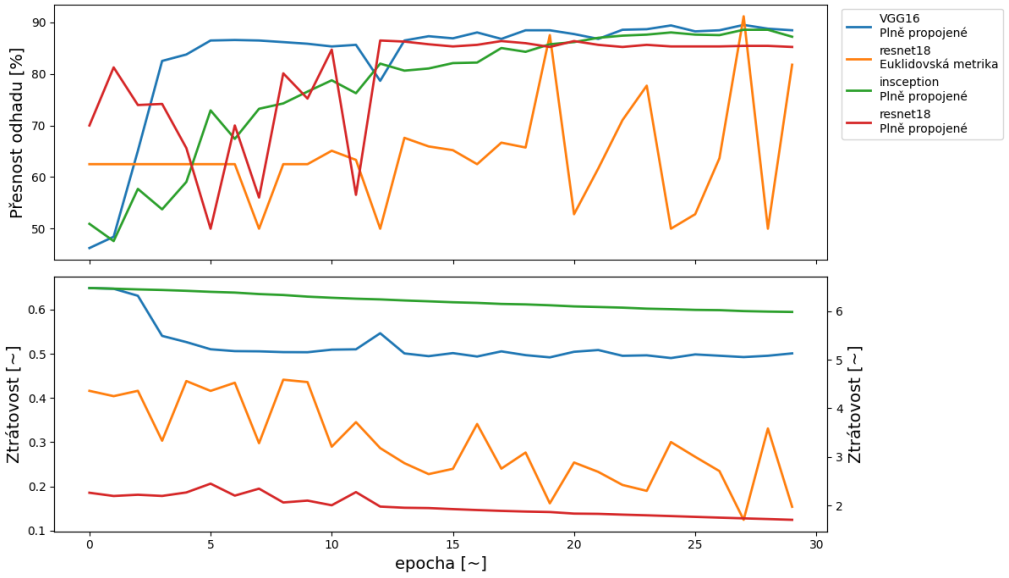
Zajímavostí, kterou lze z tabulek vyčíst je, že ne všechny modely se řídily stejnými principy. Modely jako je ResNet18, VGG16 a ResNet50 podávaly daleko lepší výsledky při nižším rozlišení vstupních fotografií. Kdežto model GoogLeNet naopak podával lepší výsledky při vyšším rozlišení a model založený na ResNet34 dosahoval přibližně stejných výsledků pro všechna rozlišení.

Ukázalo se, že hodnota ztrátovosti u modelů s FC zakončením neudává skutečnou chybovost sítě. Lze si také všimnout, že použití FC zakončení mělo u většiny modelů, které podávaly špatné výsledky s ED zakončením, pozitivní vliv. Jedinou výjimkou je síť využívající ResNet50, na kterou toto zakončení mělo u vyšších rozlišení vliv špatný.

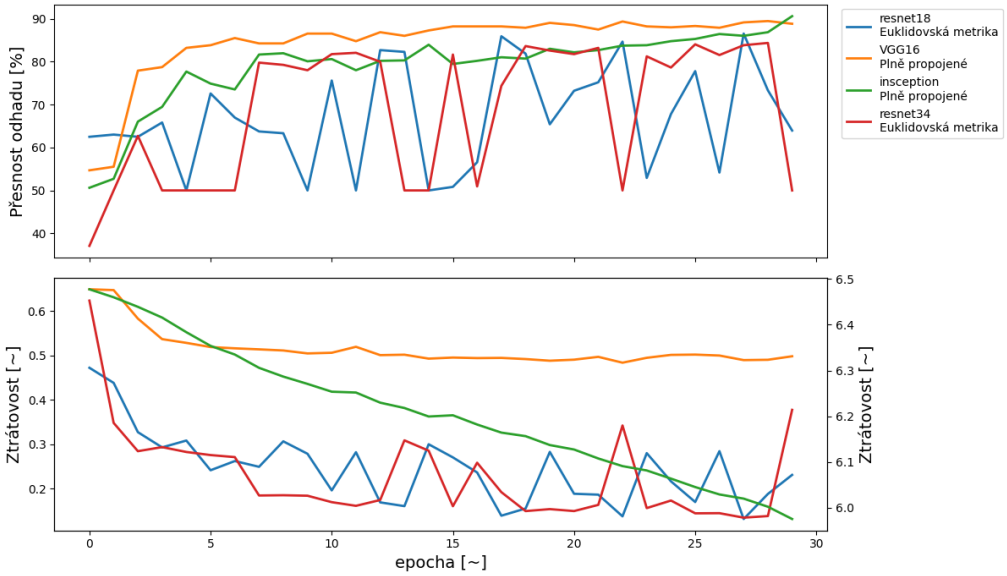
Na následujících grafech 4.1, 4.2 a 4.3 jsou zobrazeny výsledky trénování nejlepších modelů pro jednotlivá rozlišení.



Obr. 4.1: Výsledky trénování nejlepších modelů pro rozlišení 750 × 256



Obr. 4.2: Výsledky trénování nejlepších modelů pro rozlišení 1000 × 342



Obr. 4.3: Výsledky trénování nejlepších modelů pro rozlišení 1250×427

Lze si všimnout, že modely ResNet, které využívají zakončení ED, jsou velmi nestabilní oproti jejich verzím s FC zakončením. Přesný důvod, proč se tento problém u daných sítí vyskytuje, není jistý. Síť byla natrénována a otestována pro všechny dostupné optimizéry a různé rychlosti učení. Tento problém však přetrvával. Je možné, že pouhé porovnání výstupních tensorů je u této sítě nedostatečné a problematické.

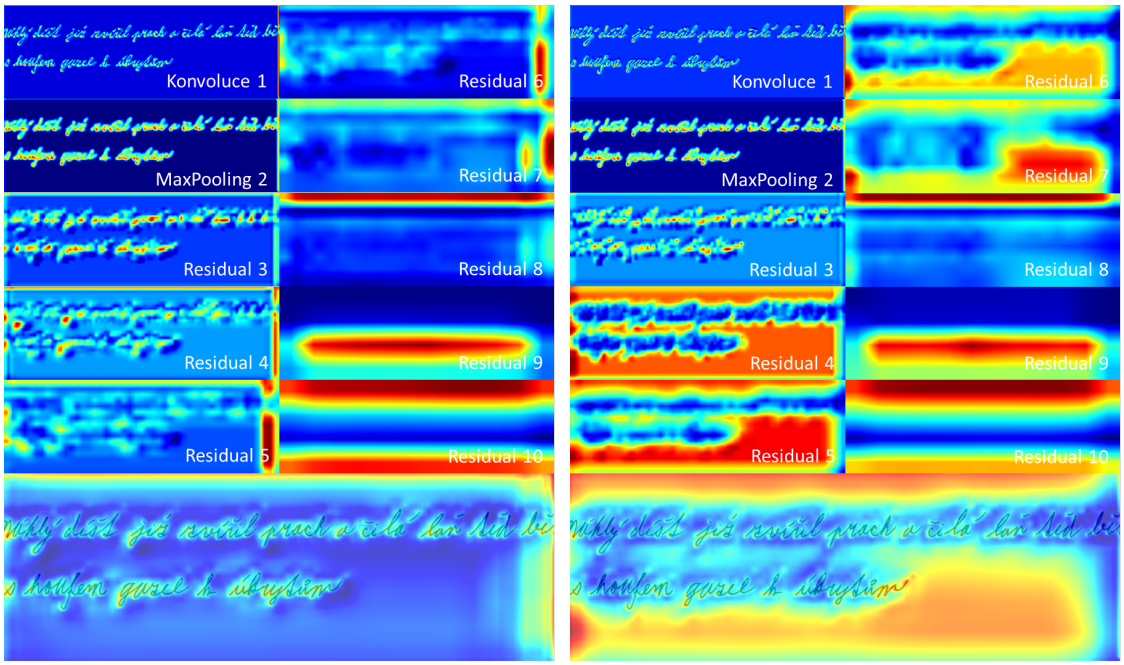
Attention mapy

Ve snaze porozumět nestabilnímu chování sítí byly vytvořeny attention mapy, které umožní porovnat na co se síť zaměřuje a co by mohlo být důvodem této nestability.

Na obrázcích 4.4 lze vidět attention mapy nejlepšího modelu ResNet18, který využívá ED zakončení a toho stejného modelu po dvou extra epochách trénování. Největších rozdílů si lze všimnout u čtvrté, páté, šesté a sedmé vrstvy modelu. Tyto Residuální bloky ve dvou epochách kompletně změnily svoje zaměření a soustředí svoji pozornost na množství prázdné bílé plochy v obraze.

Pro porovnání jsou na obrázku 4.5 vyobrazeny attention mapy modelu ResNet18 s FC zakončením. Průběh učení tohoto modelu byl stabilní, a i když nedosahoval tak dobrých výsledků jako zmíněný ResNet18 s ED, tak je z map vidět, že se model zaměřuje čistě na text ukázky.

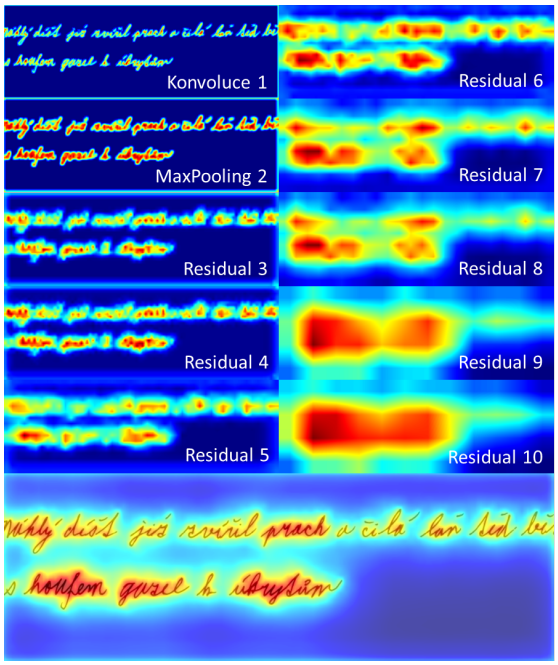
Z těchto výsledků by se dal vyvodit závěr, že ED zakončení není pro řešení tohoto problému nejvhodnější. I když jsou modely, které ho využívají, schopny dosáhnout lepších výsledků na této limitované databázi, způsob jakým těchto výsledků dosáhnou není správný a žádaný.



(a) Attention mapy pro nejlepší epochu modelu

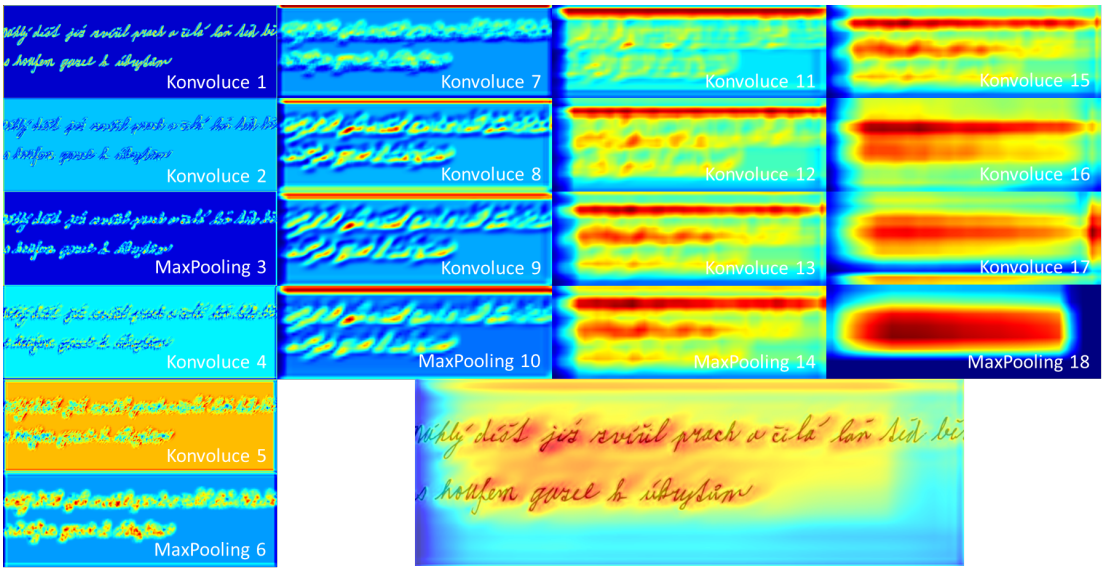
(b) Attention mapy po dvou následujících epochách

Obr. 4.4: Attention mapy vrstev modelu ResNet18 se zakončením ED

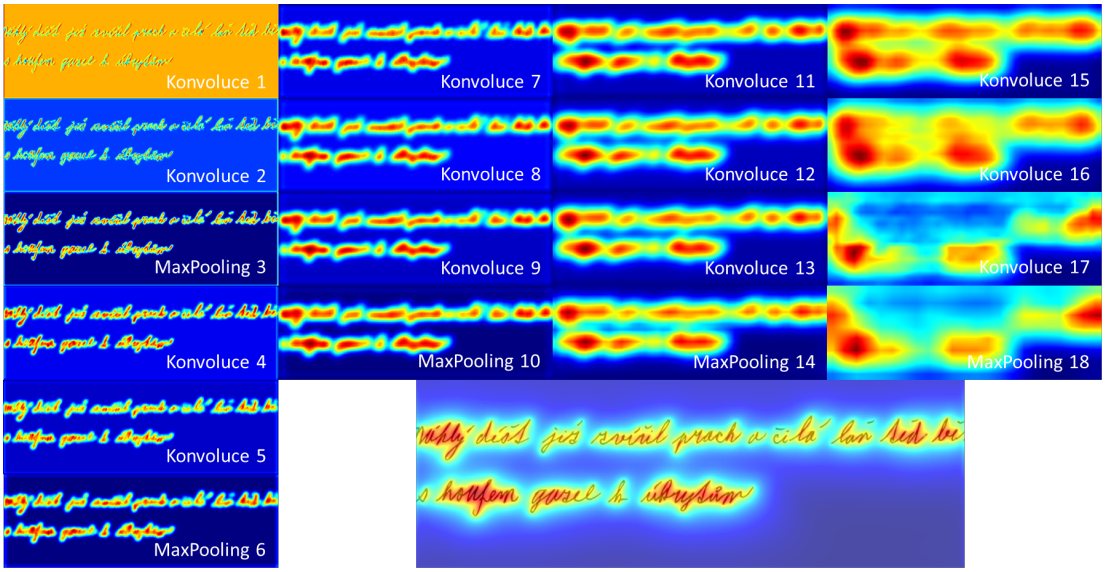


Obr. 4.5: Attention mapy vrstev modelu ResNet18 se zakončením FC

Avšak výsledky jednoho modelu nejsou pro vyvození takového závěru dostačující. Pro kontrolu, že se nejedná pouze o náhodu a vlastnost struktury ResNet, byly



Obr. 4.6: Attention mapy vrstev modelu VGG16 se zakončením ED



Obr. 4.7: Attention mapy vrstev modelu VGG16 se zakončením FC

vytvořeny attention mapy i pro model VGG16. Ten využívá kompletně odlišnou strukturu konvoluční sítě a pro rozlišení 750×256 dosahoval u obou zakončení velice dobrých výsledků. Výsledné mapy pro model využívající ED jsou zobrazeny na obrázku 4.6 a pro model využívající FC zakončení na obrázku 4.7.

Jak je z attention map jednotlivých modelů vidět, VGG16 s FC zakončením se zaměřuje čistě na text ukázky. VGG16 s ED zakončením se naopak více zaměřuje na délky vět – vrstvy třináct až patnáct, a velikost plochy pokryté textem.

Lze si také všimnout nechtěného pruhu, který vznikl na vrchní hraně obrázku.

Tento pruh se objevil v sedmé vrstvě modelu, kdy byl pravděpodobně vytvořen ostrým přechodem mezi hodnotami tensoru a nulovými hodnotami doplněnými na přesahující pozice jádra konvoluce. Místo potlačení se model na tento pruh zaměřil a kladl na něj velký důraz.

Tyto výsledky podporují již stanovený závěr, že euklidovská metrika není pro řešení tohoto problému vhodná. Tedy alespoň ve stylu použití, v jakém byla využita v této práci u daných modelů.

Závěr

Tato práce se zabývá problematikou analýzy ručně psaného českého písma s využitím umělé inteligence. Tedy navržením funkčního experimentu, pomocí kterého by bylo možné porovnat páry ukázek tvořené českými texty a na základě jejich podobnosti rozhodnout, zda tyto ukázky pochází od stejného autora, či nikoliv.

V rámci této diplomové práce byly zpracovány dohromady dvě databáze. První je tvořena českými a anglickými ukázkami textu. Ty pochází ze starých školních testů a veřejné databáze CSAFE [9]. Druhá databáze je tvořena ukázkami věty, která byla speciálně navržené tak, aby obsahovala všechna diakritická znaménka používaná v českém jazyce. V rámci experimentu bylo navrženo několik siamských konvolučních sítí, které se lišily v používaném zakončení a struktuře konvoluční části sítě. Experiment byl implementován v jazyce Python verze 3.10 a v rámci implementace byly využívány knihovny jako Tensorflow, Keras, OpenCV, Numpy, Scikit-learn a další. Jednotlivé sítě byly natrénovány na několika odlišných rozlišeních a jejich výsledky byly zaneseny do tabulek a grafů. Pro vizualizaci informací, na které se sítě v textu zaměřují, byly pro dvě rozdílné konvoluční struktury a zakončení extrahovány a vytvořeny attention mapy, které byly posléze porovnány.

Z tabulek všech porovnaných modelů lze vidět, že nejlepšího výsledku dosáhla síť založená na struktuře ResNet18. Model využíval zakončení tvořené euklidovskou metrikou a jeho úspěšnost je 92,2 % na vlastní databázi tvořené ukázkami s rozlišením 750×256. Při porovnání výsledků modelů pro různá rozlišení vstupních dat, si lze všimnout určitých zajímavostí. Struktury jako jsou ResNet a VGG podávaly lepší výsledky s nižším rozlišením vstupních dat, zatím co GoogLeNet dosahoval lepších výsledků s vyšším vstupním rozlišením. Z grafů trénování vyšlo najevo, že modely využívající euklidovskou metriku byly velmi nestabilní v porovnání s modely, které využívaly plně propojené zakončení. Vytvořené attention mapy dále ukázaly, že plně propojené zakončení má na siamské konvoluční sítě velmi pozitivní vliv. Modely se zaměřují na jednotlivá slova, jejich tvary a pozici v textu. Samotná euklidovská metrika se naopak ukázala nebýt pro zakončení sítí nejlepší volbou. I když jsou modely, které ji využívají, schopny dosáhnout výborných výsledků, mají tendenci se zaměřovat na informace, které se netýkají rukopisu autora.

Na tuto práci by bylo možné navázat rozšířením datových sad, otestováním dalších rozlišení pro již existující navržené sítě, vyzkoušením jiných způsobů zakončení sítí a také navržením nových modelů využívajících vlastní, či již otestované struktury konvolučních sítí.

Literatura

[1] BLAHA, Milan. MUNI. *Neuronove-site-jednotlivy-neuron* [online]. 17. 2. 2014 [cit. 2022-10-02]. Dostupné z: <https://portal.matematickabiologie.cz/res/f/neuronove-site-jednotlivy-neuron.pdf>

[2] IBM CLOUD EDUCATION. *Neural Networks: Artificial intelligence.* IBM [online]. 17. 8. 2020 [cit. 2022-11-05]. Dostupné z: <https://www.ibm.com/cloud/learn/neural-networks>

[3] KAČER, Petr. *Forexový automatický obchodní systém založený na neuronových sítích.* Brno, 2015. DIPLOMOVÁ PRÁCE. Vysoké Učení Technické v Brně. Vedoucí práce Doc. Ing. VÁCLAV JIRSÍK, CSc.

[4] FAUSETT, Laurene. *Fundamentals of Neural Networks: Architectures, Algorithms And Applications.* Prentice-Hall, 1994. ISB N 978-0-13-334186-7.

[5] HARDESTY, Larry. *Explained: Neural networks: Ballyhooed artificial-intelligence technique known as "deep learning" revives 70-year-old idea.* Massachusetts Institute of Technology: MIT News [online]. Massachusetts: MIT News Office, 2017, April 14, 2017 [cit. 2022-11-07]. Dostupné z: <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>

[6] WU, Yu-chen a Jun-wen FENG. *Development and Application of Artificial Neural Network.* Wireless Personal Communications [online]. 2018, 30. 12. 2017, 102(2), 1645-1656 [cit. 2022-11-08]. ISSN 0929-6212. DOI: 10.1007/s11277-017-5224-x. Dostupné z: <http://link.springer.com/10.1007/s11277-017-5224-x>

[7] VOLNÁ, Eva. *Neuronové sítě 1* [online]. Druhé. Ostrava: Ostravská univerzita v Ostravě, 2008 [cit. 2022-11-17]. Dostupné z: <https://web.osu.cz/~Volna/Neuronove_site_skripta.pdf>

[8] KOCH, Gregory. *Siamese Neural Networks for One-Shot Image Recognition [online].* Toronto, 2015 [cit. 2022-12-01]. Dostupné z: <http://www.cs.toronto.edu/~gkoch/files/msc-thesis.pdf>. Diplomová práce. University of Toronto.

[9] CRAWFORD, Amy, Anyesha RAY, Alicia CARRIQUIRY, James KRUSE a Marc PETERSON. *CSAFE Handwriting Database* [online]. Iowa State University, 2019 [cit. 2022-12-02]. Dostupné z: <https://doi.org/10.25380/iastate.10062203.v1>

[10] DU, William, Michael FANG a Margaret SHEN. *Siamese Convolutional Neural Networks for Authorship Verification* [online]. 450 Serra Mall, Stanford, CA 94305, 2017 [cit. 2022-12-09]. Dostupné z: <http://cs231n.stanford.edu/reports/2017/pdfs/801.pdf>. Stanford University.

[11] DLAMINI, Nkosikhona a Terence L VAN ZYL. *Author Identification from Handwritten Characters using Siamese CNN.* 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC) [online]. IEEE, 2019, 2019, 1-6 [cit. 2023-05-05]. ISBN 978-1-7281-0040-1. DOI: 10.1109/IMITEC45504.2019.9015897 Dostupné z: <https://ieeexplore.ieee.org/abstract/document/9015897>

[12] DEY, Sounak, Anjan DUTTA, J. Ignacio TOLEDO, Suman K. GHOSH, Josep LLADOS a Umapada PAL. *SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification.* Elsevier [online]. 2017, 30. 9. 2017, 1-7 [cit. 2022-12-09]. DOI: 10.48550/arXiv.1707.02131 Dostupné z: <https://arxiv.org/pdf/1707.02131.pdf>

[13] CHU, Jun, Mohammad Abuzar SHAIKH, Mihir CHAUHAN, Lu MENG a Sargur SRIHARI. *Writer Verification using CNN Feature Extraction.* 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR) [online]. IEEE, 2018, 2018, 181-186 [cit. 2022-12-09]. ISBN 978-1-5386-5875-8. DOI: 10.1109/ICFHR-2018.2018.00040 Dostupné z: <https://ieeexplore.ieee.org/document/8563247>

[14] RAHMAN, Atta Ur a Zahid HALIM. *A graph-based solution for writer identification from handwritten text.* Knowledge and Information Systems [online]. 2022, 64(6), 1501-1523 [cit. 2022-12-09]. ISSN 0219-1377. DOI: 10.1007/s10115-022-01676-7 Dostupné z: <https://link.springer.com/10.1007/s10115-022-01676-7>

[15] VALDEZ-RODRÍGUEZ, José E., Hiram CALVO a Edgardo M. FELIPE-RIVERÓN. *Handwritten Texts for Personality Identification Using Convolutional Neural Networks.* Pattern Recognition and Information Forensics [online]. Cham: Springer International Publishing, 2019, 2019-12-19, 140-145 [cit. 2022-12-09]. Lecture Notes in Computer Science. ISBN 978-3-030-05791-6. DOI: 10.1007/978-3-030-05792-3_13 Dostupné z: <http://link.springer.com/10.1007/978-3-030-05792-3_13>

[16] IBM CLOUD EDUCATION. *Recurrent Neural Networks.* IBM [online]. 14. 9. 2020 [cit. 2022-11-29]. Dostupné z: <https://www.ibm.com/cloud/learn/recurrent-neural-networks>

[17] ARAVINDPAI, Pai. *CNN vs. RNN vs. ANN – Analyzing 3 Types of Neural Networks in Deep Learning.* Analytics Vidhya [online]. 2020, 17. 2. 2020 [cit. 2022-11-26]. Dostupné z: <https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/#h2_4>

[18] SAUNIL, Ray. *Understanding and coding Neural Networks From Scratch in Python and R.* Analytics Vidhya [online]. 24. 7. 2020 [cit. 2022-11-28]. Dostupné z: <https://www.analyticsvidhya.com/blog/2020/07/neural-networks-from-scratch-in-python-and-r>

[19] DISHASHREE, Gupta. *Fundamentals of Deep Learning – Introduction to Recurrent Neural Networks.* Analytics Vidhya [online]. 28. 11. 2020 [cit. 2022-11-28]. Dostupné z: <https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks>

[20] AISHWARYA, Singh. *Demystifying the Mathematics Behind Convolutional Neural Networks (CNNs).* Analytics Vidhya [online]. 8. 5. 2020 [cit. 2022-11-29]. Dostupné z: <https://www.analyticsvidhya.com/blog/2020/02/mathematics-behind-convolutional-neural-network

[21] *Keras: Losses* [online]. [cit. 2022-11-17]. Chollet, Francois and others, 2015 Dostupné z: <https://keras.io/api/losses/>

[22] MENA, R, F RODRÍGUEZ, M CASTILLA a M.R ARAHAL. *A prediction model based on neural networks for the energy consumption of a bioclimatic building.* Energy and Buildings [online]. Elsevier, 2014, Říjen 2014, 82, 142-155 [cit. 2022-11-25]. ISSN 03787788. DOI: 10.1016/j.enbuild.2014.06.052. Dostupné z: <https://www.sciencedirect.com/science/article/pii/S0378778814005349>

[23] CASSAR, Daniel R., André C.P.L.F. DE CARVALHO a Edgar D. ZANOTTO. *Predicting glass transition temperatures using neural networks.* Acta Materialia [online]. 2018, 159, 249-256 [cit. 2022-11-25]. ISSN 13596454. DOI: 10.1016/j.actamat.2018.08.022. Dostupné z: <https://www.sciencedirect.com/science/article/pii/S1359645418306542>

[24] KHAKI, Saeed a Lizhi WANG. *Crop Yield Prediction Using Deep Neural Networks.* Frontiers in Plant Science [online]. 2019, 10 [cit. 2022-11-25]. ISSN 1664-462X. DOI: 10.3389/fpls.2019.00621. Dostupné z: <https://www.frontiersin.org/article/10.3389/fpls.2019.00621/full>

[25] HARLEY, Leonard. *Environmental Sustainability Vs. Climate Change.* In: Medium [online]. 2017, 25. 9. 2017 [cit. 2022-12-09]. Dostupné z: <https://medium.com/earthtokens/environmental-sustainability-vs-climate-change-2cdc93390dfe>

A Obsah elektronické přílohy

V přiloženém souboru lze nalézt:

- `/` ..... kořenový adresář přiloženého archivu
  - `Attention_maps` ..... složka s vygenerovanými attention mapami
  - `Handwriting_scripts` ..... skripty napsané v jazyce Python
    - `attention_maps.py`
    - `augment_photos.py` ..... augmentace a zmenšení obrázků
    - `config.py` ..... konfigurace parametrů sdílených mezi skripty
    - `create_dataset.py` ..... dělení obrázků a tvorba datových sad
    - `dataset_loader.py` ..... funkce pro načítání ukázek
    - `data_generators.py` ... třídy pro dynamické načítání ukázek během trénování
    - `layer_functions.py`
    - `siamese_GoogLeNet.py`
    - `siamese_model.py`
    - `siamese_resnet.py`
    - `siamese_VGG.py`
    - `test_model.py`
    - `train_model.py`
  - `Photos` ..... obrázky a jejich rozdělení do datových sad
    - `paragraphs` ..... složka obsahující všechny obrázky
      - `original` ..... složka obsahující složky originálních obrázků
        - `0` ..... složka s originálními obrázky
      - `all` ..... složka obsahující zmenšené a augmentované obrázky
    - `pairs_train.csv`
    - `pairs_test.csv`
    - `pairs_valid.csv`
    - `pairs_cross.csv`
    - `pairs_all.csv`
  - `Models` ..... složka pro nově vytvořené modely
  - `README.md` ..... návod
  - `Results` ..... složka pro výsledky trénování a vygenerované attention mapy
  - `./run_training.sh` ..... skript pro trénování všech modelů ve složce Models
  - `requirements.txt` ..... potřebné knihovny jazyka Python

Kód byl otestován na operačním systému Linux s verzí jazyka Python 3.10.7. Při vývoji bylo používáno IDE Visual Studio Code.