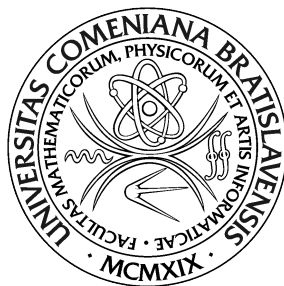


UNIVERZITA KOMENSKÉHO V BRATISLAVE  
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



Možnosti využitia metód hlbokého učenia  
v predpovedi počasia

Diplomová práca

UNIVERZITA KOMENSKÉHO V BRATISLAVE  
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



Možnosti využitia metód hlbokého učenia  
v predpovedi počasia

Diplomová práca

Študijný program: Aplikovaná informatika  
Študijný odbor: 2511 Aplikovaná informatika  
Školiace pracovisko: Katedra aplikovanej informatiky  
Školiteľ: RNDr. Andrej Lúčny, PhD.



Univerzita Komenského v Bratislave  
Fakulta matematiky, fyziky a informatiky

## ZADANIE ZÁVEREČNEJ PRÁCE

- Meno a priezvisko študenta:** Bc. Juraj Mašlej  
**Študijný program:** aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)  
**Študijný odbor:** aplikovaná informatika  
**Typ záverečnej práce:** diplomová  
**Jazyk záverečnej práce:** slovenský  
**Sekundárny jazyk:** anglický
- Názov:** Možnosti využitia metód hlbokého učenia v predpovedi počasia  
*Application of Deep Learning Methods for Weather Forecast*
- Anotácia:** Kompilačná a čiastočne implementačná práca zo strojového učenia s fyzikálnym rozmerom
- Cieľ:** Hlboké učenie využíva konvolučné neurónové siete, kde sa z dát učí tzv. kernel, ktorý sa paralelne aplikuje na všetky miesta na dátach a spracuje lokálne okolie tohto miesta na zložku nadradenej dátovej úrovne. Táto metóda sa s úspechom používa na spracovanie obrazu. Cieľom tejto práce je preskúmať možnosti jej aplikácie v inej doméne a to pri spracovaní meteorologických údajov. Tieto údaje majú tiež 2D charakter ako obraz, avšak rôzne zložky v rôznych jednotkách: jednotlivé meteorologické veličiny (teplota, vlhkosť, tlak, rýchlosť vetra) a geografické dáta (nadmorská výška, zemepisná dĺžka, ...). Údaje s potrebnou anotáciou budú poskytnuté. Počítačový jazyk a knižnica pre hlboké učenie si diplomant vyberie sám, avšak použije existujúce riešenie, odporúčané prostredie je tensorflow. Hardwarová platforma na rozsiahle výpočty potrebné pre spracovanie dát, bude poskytnuté.
- Literatúra:** Ian Goodfellow and Yoshua Bengio and Aaron Courville: Deep Learning, MIT Press Book, <http://www.deeplearningbook.org/>  
Články o deep learning  
Dokumentácia k nástrojom ako Tensorflow
- Poznámka:** Práca má komerčné využitie a je možné pre ňu získať komerčnú podporu.
- Kľúčové slová:** hlboké učenie, deep learning, meteorológia, predpoveď počasia
- Vedúci:** RNDr. Andrej Lúčny, PhD.  
**Katedra:** FMFI.KAI - Katedra aplikovanej informatiky  
**Vedúci katedry:** prof. Ing. Dr. Igor Farkaš  
**Dátum zadania:** 25.09.2017
- Dátum schválenia:** 09.10.2017
- prof. RNDr. Roman Ďurikovič, PhD.  
garant študijného programu

Čestne prehlasujem, že túto diplomovú prácu som vypracoval samostatne len s použitím uvedenej literatúry a za pomoci konzultácií u môjho školiteľa.

Bratislava, 2018

.....

Bc. Juraj Mašlej

# Pod'akovanie

Todo : pod'akovanie

# Abstrakt

Problematika tejto diplomovej práce vychádza z meteorológie, prezentujeme problém určenia miery oblačnosti zo snímok meteorologických staníc. V súčasnosti sa oblačnosť klasifikuje rozdelením na osminy, kde 8/8 predstavuje plne zamračenú oblohu. Ako nástroj na riešenie tohto problému chceme preskúmať metódy hlbokého učenia, najmä konvolučné siete. Pre získanie základného riešenia chceme využiť niektorú z známych metód strojového učenia. Boli nám poskytnuté dáta, a to snímky oblohy a synopy, z ktorých vieme selektovať hodnotenie oblačnosti. Vďaka tomuto môžeme použiť vyššie spomínané metódy učenia s učiteľom. Pri štúdiu literatúry sme narazili na fakt, že hlavným problémom bol dostatočný objem kvalitných dát. Očakávame tento problém aj pri našej práci.

Výsledkom našej práce má byť model neurónovej siete schopný na základe snímky oblohy určiť podiel oblačnosti na oblohe v osminách. Nevylučujeme, že niektorá z jednoduchších metód strojového učenia môže priniesť podobnú, alebo aj vyššiu presnosť klasifikácie. Na základe preštudovanej literatúry si ale myslíme, že to je nepravdepodobné.

Kľúčové slová: hlboké učenie, predpoveď počasia, meteorológia

# Abstract

Todo : abstrakt en

Keywords: deep learning, weather forecast, meteorology

# Obsah

<b>1</b>	<b>Prehľad problematiky</b>	<b>1</b>
1.1	Neurónové siete . . . . .	1
1.1.1	Výpočet v neurónovej sieti . . . . .	1
1.2	Konvolučné neurónové siete [GBC16] . . . . .	3
1.2.1	Štruktúra konvolučných sietí . . . . .	3
1.2.2	Zlučovanie . . . . .	4
1.3	Reziduálne neurónové siete (ResNet) [dS16] . . . . .	6
1.3.1	Motivácia . . . . .	6
1.3.2	Koncept reziduálnych sietí . . . . .	6
1.4	Prehľad technológií . . . . .	8
1.4.1	Keras . . . . .	8
1.4.2	OpenCV, Cuda . . . . .	12
1.5	Doterajšie riešenia . . . . .	13
1.5.1	Deep Convoltional Neural Network for Cloud Coverage Estima- tion from Snapshot Camera Images“, Ryo Onishi [OS17] . . . . .	13
1.5.2	Deep Learning for Cloud Detection [LGTW <sup>+</sup> 17] . . . . .	16



# Kapitola 1

## Prehľad problematiky

### 1.1 Neurónové siete

Vývoj neurónových sietí bol inšpirovaný ľudským mozgom. Jeho štruktúra prepojenia neurónov synapsami a posilňovanie aktívnych spojení sa stala ideou pre sieť prepojení a aktivačných funkcií neurónovej siete.

Základný koncept neurónovej siete pozostáva z neurónov realizujúcich logickú funkciu. Podobne ako biologické neuróny majú tieto jednotky niekoľko logických vstupov, ktoré môžu zvyšovať, alebo naopak, znižovať výstupnú hodnotu. Hodnota tohto výstupu závisí od vstupných hodnôt a aktivačnej funkcie vo vnútru neurónu. Neurón s vhodnou aktivačnou funkciou a vhodne nastavenými parametrami vstupov dokáže simulovať základné logické operácie ako AND, OR alebo NOT. Sieť takýchto neurónov môže byť použitá ako univerzálny model pre výpočty pravdivostných funkcií. Medzi najpoužívannejšie aktivačné funkcie patria sigmoid, softmax, hyperbolický tangens alebo ReLu - teda výber najvyššej hodnoty. Funkciu softmax sa venujeme bližšie v samostatnej kapitole.

#### 1.1.1 Výpočet v neurónovej sieti

**Dopredný výpočet – forward pass**

Vstup neurónu – vstup siete vypočítame ako vstupný vektor vynásobený váhami, plus pridávame bias.

$$input\_layer_1 = w_1 * i_1 + \dots + w_n * i_n + b_1 * 1 \quad (1.1)$$

Na tento vstup aplikujeme aktivačnú funkciu zvolenú pre danú sieť. Výsledok tejto funkcie je výstup daného neurónu.

**Výpočet chyby**

Pre výpočet chyby môžeme zvoliť „mean squared error“ funkciu, teda strednú kvadratickú chybu.

$$E_{total} = \frac{1}{2} * \sum (target - output)^2 \quad (1.2)$$

**Spätná propagácia chyby**

Metóda prvýkrát navrhnutá Werbosom(1974) [Wer74] na úpravu váh v neurónovej sieti. Kombinuje metódu klesajúceho gradientu s algoritmom na iteráciu vrstvami siete na výpočet gradientu chýb v sieti, ktorý je potrebný pre jeho znižovanie.

Cieľom tejto metódy je úprava váh tak, aby sa nový výpočet siete priblížil požadovanému výstupu. Tým sa zároveň zníži celková chyba výstupu siete.

Pri výpočte postupujeme od výstupnej, poslednej, vrstvy. Potrebujeme zistiť nakoľko zmena jednotlivých váh vchádzajúcich do výstupnej vrstvy ovplyvňuje celkovú chybu siete. Pre tento výpočet aplikujeme pravidlo zreťazenia(chain rule). Jeho použitím dostaneme

$$\frac{\partial E_{total}}{\partial \omega_x} = \frac{\partial E_{total}}{\partial out_{o_z}} * \frac{\partial out_{o_z}}{\partial net_{o_z}} * \frac{\partial net_{o_z}}{\partial \omega_x} \quad (1.3)$$

Kde  $w_x$  označuje  $x$ -tú vchádzajúcu váhu do neurónu  $o_z$  na výstupnej vrstve.  $net$  predstavuje vstupnú hodnotu neurónu,  $out$  hodnotu po aplikovaní aktivačnej funkcie. Pre celkový opis výpočtu spätnej propagácie chyby chceme odkázať na literatúru [Maz15].

## 1.2 Konvolučné neurónové siete [GBC16]

### 1.2.1 Štruktúra konvolučných sietí

Konvolučné siete sú špecializovaným druhom neurónových sietí zameraných na spracovanie dát s mrežovou, alebo jej podobnou štruktúrou. Príkladmi sú časové série, ktoré sa dajú interpretovať ako 1 dimenzionálna mriežka v pravidelných časových intervaloch, alebo obrazové dáta kde pixely tvoria 2 dimenzionálnu mrežovitú štruktúru.

Sieť vykonáva matematickú operáciu označovanú ako konvolúcia. Za konvolúciou používa väčšina týchto sietí operáciu nazývanú „pooling“ (zlučovanie), teda znižovanie rozmeru dát. Pre modely sietí kde vrstevme konvolučné vrstvy bolo ukázané, že vrchné vrstvy sa učia jednoduché príznaky ako čiara, oblúk, a pod. Zatiaľ čo hlbšie vrstvy tieto príznaky spájajú a umožňujú tým klasifikáciu komplikovanejších tvarov.

#### Konvolúcia v neurónových sieťach

Vo všeobecnosti je konvolúcia operácia hodnôt dvoch funkcií na parametri reálnej hodnoty. Príkladom môže byť sledovanie pohybu objektu v čase. Náš senzor poskytuje v čase  $t$  výstup  $x$ , ktorý označuje polohu objektu, teda  $x(t)$ . Obidve premenné nadobúdajú reálne hodnoty, v priebehu času sa menia. Predpokladajme, že označenie polohy senzorom je nepresné. Na získanie presného určenia polohy chceme priemerovať merania. Časovo nedávnym meraniam chcem zvýšiť váhu na výstup oproti starším meraniam. Uvažujme funkciu  $w(a)$ , kde „ $a$ “ je doba od uplynutia merania, teda „vek“ merania. Ak túto funkciu aplikujeme na každé diskkrétne meranie v čase  $t_x$ , dostaneme novú funkciu  $s$  poskytujúcu lepšiu odhad polohy objektu.

$$s(t) = \int_w x(a)w(t-a)da \quad (1.4)$$

Pre tento príklad ako funkciu  $w$  potrebujeme funkciu distribúcie pravdepodobností. Pre konvolučné siete sa funkcia  $x(t)$  označuje ako vstup, zatiaľ čo funkcia  $w$  je označovaná ako „kernel“ teda jadro siete. Túto operáciu nazývame konvolúcia.

V našej práci ale pracujeme s diskkrétnymi konvolučnými sieťami, preto pre definíciu

kernelu môžeme použiť zápis pomocou súm.

$$f(x * g) = \sum_{m=-M}^M \sum_{n=-N}^N f(x - n, y - m)g(n, m) \quad (1.5)$$

[Poc] Pričom  $g$  označujeme ako kernelovú funkciu

### Motivácia pre konvolučné siete

Tento druh sietí využíva najmä zdieľanie parametrov a riedke interakcie medzi váhami. Všetky neuróny v jednej konvolučnej vrstve majú rovnakú váhu a bias. Príčinu a dôsledky tohto javu rozoberáme v nasledujúcich odsekoch. Ďalšou výhodou je schopnosť siete pracovať so vstupom rôznej veľkosti, a teda škálovateľnosť vstupu.

#### Zdieľanie parametrov

V konvolučnej sieti je vektor váh každého kernelu použitý na všetkých dátach vstupu, teda na každej pozícii obrazu pre obrazové dáta. To znamená, že na rozdiel od učenia sa parametrov diskkrétne pre každú polohu na vstupe, sa konvolučná vrstva učí set parametrov, ktorý aplikuje na celý vstup.

#### Riedke interakcie

Klasická neurónová sieť využíva maticové násobenie váh a vstupu, kde sa použije samostatný parameter pre spojenie konkrétneho vstupu s konkrétnym neurónom. Na rozdiel od tohto prístupu má konvolučná sieť nízku prepojenosť, teda použijeme kernel menší ako vstup, čo zapríčiní, že váhy budú zdieľané medzi neurónmi a vplyv jednej váhy bude obmedzený na istú skupinu neurónov. Každý neurón nemá svoj jedinečný vektor váh. Príkladom je spracovanie obrazu, kde vstup do siete môže obsahovať tisíce pixelov, no kernel pre detekciu jednoduchých objektov, alebo napríklad hrán, môže mať rádovo menšie rozmery, to v desiatkách alebo stovkách pixelov. Tento princíp zdieľania parametrov znižuje výpočtovú aj pamäťovú náročnosť celého algoritmu.

### 1.2.2 Zlučovanie

Vrstva v konvolučnej sieti používaná na vyhodnotenie konvolúcie, teda reakcie vstupu na kernel. Ide o aplikovanie danej štatistickej funkcie na výstup z konvolučnej vrstvy.

Príklad pre obrazové dáta. Určíme si rozmer dát pre ktorý chceme vykonať zlučovanie, nech ten je 3 na 3. Pre túto oblasť použijeme zlučovanie s výberom najvyššieho člena, teda vyberieme najvyššiu odpoveď siete na tomto okolí.

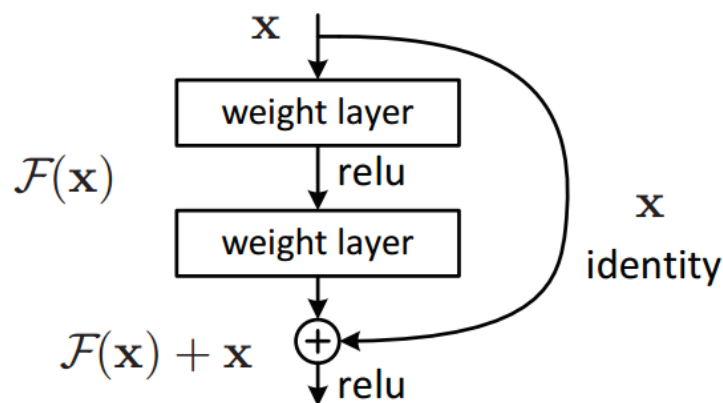
## 1.3 Reziduálne neurónové siete (ResNet) [dS16]

### 1.3.1 Motivácia

Vývoj v oblasti konvolučných sietí ukázal, že viac vrstiev siete dokáže zlepšovať výsledky klasifikácie, ale neplatí univerzálny fakt, čím viac vrstiev, tým lepšie výsledky. Pridávanie vrstiev sa preto pri neurónových sieťach nezaobíde bez problémov. Jedným z problémov je miznutie, alebo, naopak, príliš rýchly nárast gradientu čo zabrzdí konvergenciu siete. Tento problém sa dá riešiť normalizovanou inicializáciou a použitím normalizačných vrstiev. Tie zabezpečia konvergenciu siete pri použití algoritmu spätnej propagácie chyby. Ďalším problémom hlbokých sietí je degradácia presnosti siete. Pre riešenie tohto problému je využívané reziduálne učenie.

### 1.3.2 Koncept reziduálnych sietí

Ideou je pridanie spojenia medzi vstupom pre vrstvu  $n$  a vstupom pre vrstvu  $n + x$ , pričom  $x$  je nenulové. Týmto chceme pridať do hlbšej vrstvy stav siete naučený vo vyššej vrstve. Výstup hlbšej vrstvy do ktorej takéto prepojenie vchádza, nepozostáva iba z konceptu naučeného o 1 vrstvu vyššie, ale je mu pridaná aj referencia o tom ako vznikol stav siete ktorý do danej vrstvy vchádza.



Obr. 1.1: Reziduálna sieť

[dS16]

**Možné využitie reziduálnych sietí v tejto práci**

V práci chceme postaviť model viac vrstvovej neurónovej siete. Preto môžeme naraziť na problém miznúceho, alebo naopak neúmerne rastúceho gradientu (vanishing/exploding gradient). Pomocou zakomponovania reziduálnej zložky môžeme s týmto možným problémom bojovať. Takisto táto metóda môže pomôcť presnosti klasifikácie.

## 1.4 Prehľad technológií

V tejto kapitole sa budeme zaoberať technológiami vybranými pre túto prácu. Chceme čitateľovi priblížiť technológie Keras, OpenCV a Cuda.

### 1.4.1 Keras

#### O Kerase

Keras je vysoko-úrovňová knižnica pre prácu s neurónovými sieťami vyvinutá v jazyku python. Je nadstavbou technológií TensorFlow, CNTK a Theano. Cieľom pri vývoji bolo umožniť čo najrýchlejší vývoj modelu a tým pádom umožniť užívateľom vyskúšať viacero modelov. Keras sa uvádza ako vhodná knižnica ak je potreba pre rýchle prototypovanie, podporu rôznych typov neurónových sietí vrámci jednej knižnice a predpokladá sa potreba spúšťať výpočty aj na procesore, aj na grafickej karte.

#### Konvolučné siete a vývoj modelu - zamerané na Keras

Keras ponúka viacero typov neurónových sietí, rozhodli sme sa bližšie predstaviť konvolučnú sieť, pretože ju používame v našej práci.

#### Výber datasetu [YSAM]

Konvolučná sieť je nelineárny model, preto si vyžaduje väčšiu veľkosť datasetu ako niektoré jednoduchšie algoritmy. Koľko dát je dosť nie je možné povedať všeobecne. Prof. Yaser Abu-Mostafa vo svojom kurze uvádza, že dát potrebujeme mať približne 10 krát viac ako je stupňov voľnosti nášho modelu [YSAM]. Teda sieť s 3 váhami, by mala mať aspoň 30 vstupných dát. Existujú ale prípady, kedy vieme úspešne natrénovať model, ktorý má viac parametrov (stupňov voľnosti) ako vstupných dát. Jake Vanderplas vo svojom článku [Van] ponúka viacero modelov, ktoré môžeme úspešne trénovať napriek malému počtu dát. V závere tvrdí, že zo situácie keď máme viac stupňov voľnosti ako dát vieme vyťažiť najmä v prípade ak sú dáta skreslené, teda majú bias, sú veľmi rôznorodé, alebo sú zašumené. Potrebnú veľkosť datasetu takisto ovplyvňuje náročnosť klasifikácie ktorú od siete budeme požadovať, a prípadne schopnosť predvýpočtu príznakov zľahčujúcich klasifikáciu. V závere teda nevieme s určitosťou predpovedať



potrebnú veľkosť datasetu. Ryo Onishi v práci [OS17] uvádza použitie vyše 1700 fotografií.

## Predpríprava datasetu

Je potrebné aby obrazové dáta mali rovnaké rozmery, takisto je vhodné vyskúšať viacero farebných schém. Predvýpočtom môžeme zistiť v ktorých farebných schémach dokážeme nájsť kanály s najvyššiou variabilitou potrebnou pre našu klasifikáciu. Obmedzením farebných kanálov, a použitím iba pre klasifikáciu relevantných, vieme zrýchliť výpočet neurónovej siete.

V tomto bode predpokladáme, že máme dataset s obrazovými dátami s rovnakými rozmermi, prevedenými do rovnakého farebného spektra. Pre zakódovanie želaného výsledku klasifikácie môžeme použiť "one-hot-encode", teda kódovanie binárnym vektorom, kde na príslušnom indexe pre danú kategóriu budeme mať hodnotu 1, zvyšné hodnoty budú nulové. Dospeli sme k pripravenému datasetu spolu s kódovaním výsledných kategórií.

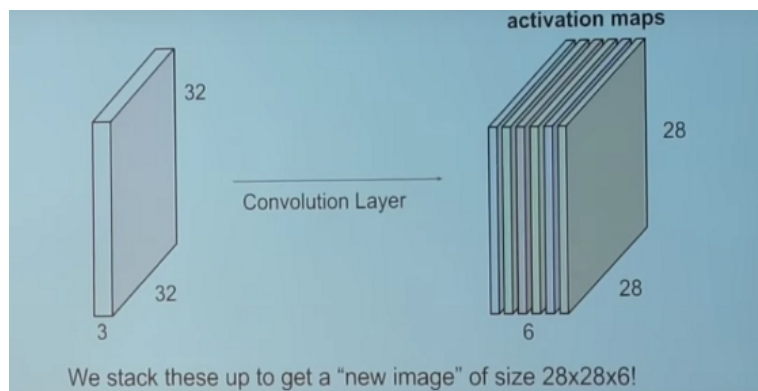
## Vytvorenie modelu siete

O teórii konvolučných sietí sme hovorili v kapitole Konvolučné siete. Pre potreby tejto kapitoly iba zopakujeme, že prvotné vrstvy konvolučnej siete sa učia jednoduché príznaky ako priamka, zatiaľ čo hlbšie vrstvy tieto jednoduché príznaky spájajú do zložitejších príznakov umožňujúcich lepšie rozpoznanie objektov. Pre potreby opisu práce s knižnicou Keras uvažujeme použitie nasledujúcich vrstiev. Takisto použijeme sekvenčný typ modelu, ktorý nám umožňuje postupne pridávať vrstvy.

### Konvolučné vrstvy [YSAM]

Vo vytváranom modeli môžeme použiť niekoľko konvolučných vrstiev. Potrebujeme špecifikovať počet filtrov, teda veľkosť výstupu, aktivačnú funkciu a veľkosť kernelu. Počet filtrov špecifikuje koľko kernelov chceme použiť na danú vrstvu. Filtre, teda kernely, sú navzájom nezávislé.[Pok]

Veľkosť kernelu špecifikuje ako veľký príznak chceme zachytiť. Pri nižších vrstvách



Obr. 1.2: CNN, 6 kernelov, [Pok]

sa väčšinou usilujeme o rozpoznanie základných príznakov, nemá preto zmysel používať zbytočne veľký kernel. Pri vyšších vrstvách sa ale na rovnakom rozmere zobrazuje vďaka zlučovacej vrstve už spracovaná väčšia plocha vstupu, vďaka tomu môžeme kernel držať rádovo menší ako vstupné dáta. Ako aktivačná funkcia je v súčasnosti pri konvolučných sieťach najčastejšie používaná ReLU funkcia, kde  $f(x) = \max(0, x)$ .

### Aktivačná funkcia softmax [Yan]

Zadefinujme funkciu softmax ako:

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} \quad (1.6)$$

Kde  $z$  je vektor vstupov a  $K$  je počet výstupných jednotiek, resp. kategórií pre prípad použitia v plne prepojenej vrstve. Funkcia *softmax* zabezpečí, že ak zosumujeme výstup pre všetky kategórie, dostaneme sa na číslo 1. Výsledky tejto funkcie môžeme interpretovať ako rozdelenie pravdepodobností. Túto funkciu budeme využívať ako aktivačnú funkciu pre plne prepojenú vrstvu.

### Plne prepojená a vyrovnávacia vrstva

Rozbaľovacia (flatten) vrstva vytvorí z  $n$  rozmerného vektoru 1 rozmerný vektor. Rozbaľovaciu vrstvu použijeme ako spojenie medzi poslednou konvolučnou (alebo zlučovaciu) a plne prepojenou vrstvou. Pre našu aplikáciu uvažujeme vyrovnávaciu vrstvu ktorá jednoducho rozbalí výstup poslednej konvolučnej vrstvy na 1 rozmerný vektor. Ten sa stane vstupom pre plne prepojenú vrstvu. Počet neurónov plne prepojenej vrstvy bude počtom výstupných tried pre ktoré chceme vykonať klasifikáciu. Ako aktivačnú funkciu môžeme použiť *softmax* funkciu. Teda výstup tejto vrstvy vieme interpretovať

ako pravdepodobnosť zaradenia do jednotlivých kategórií.

### Parametre modelu

Potrebuje sa špecifikovať ešte rýchlosť učenia, chybovú funkciu a metódu vyhodnocovania úspešnosti klasifikácia. Knižnica Keras ponúka nástroje pre odhad rýchlosti učenia. Jedným z nich je napríklad využitie algoritmu *adam* [Bro]. Často používanou chybovou funkciou je napríklad stredná kvadratická chyba. V konvolučných sieťach je ale častejšie využívaná *crossentropy* funkcia, pre jej definíciu sa odkážeme na literatúru [cro]. Pre zjednodušenie, ak predpovedáme pravdepodobnosť medzi 1 a 0 a výsledkom malo byť hodnotenie 1, predpoveď 0.1 trestáme udelením rádovo vyššej chyby ak predpoveď 0.7. Pre klasifikáciu viac ako dvoch kategórií použijeme *multicategory – crossentropy* funkciu [cro]. Ako metriku vyhodnocovania úspešnosti siete môžeme jednoducho merať je presnosť (accuracy metric) alebo použiť napríklad maticu zámeny (confusion matrix).

### Trénovanie modelu

Predpokladáme, že dataset máme rozdelený na trénovacie, validačné a testovacie zložky. Potrebuje sa nastaviť počet epoch, teda koľkokrát ma výpočet cez neurónovú sieť prebehnúť na trénovacom datasete. Existuje viacero metód ako nastaviť počet epoch, pre jednoduchosť uvedme, že sieť trénujeme pokiaľ sa testovacia chyba nedostane pod nami stanovenú úroveň a zároveň zmena chyby medzi epochami je dostatočne veľká. V tomto bode môžeme spustiť trénovanie modelu na časti datasetu vyhradenej na trénovanie. Keras ponúka funkcionality, kde volaním jednej funkcie model natrénujeme na trénovacích dátach a zároveň dostaneme výsledky z behu predikcie na validačných dátach. Po dobehnutí algoritmu máme natrénovanú sieť s nastavenými váhami, ktorú môžeme použiť na vytváranie predikcií pre dáta, ktoré ešte modelom neprešli.

### Vytváranie predpovedí

Ak sme úspešne splnili všetky predošlé kroky, sieť môžeme použiť k vytváraniu predpovedí. Knižnica Keras vytváranie predpovedí zjednodušuje na volanie jednej funkcie, ktorej parametrom sú vstupné dáta. Vďaka použitiu aktivačnej funkcie softmax na plne prepojenej vrstve dostávame ako výsledok pre každý vstup vektor s pravdepodobnosťami kategorizácie vstupu do jednotlivých tried.

### 1.4.2 OpenCV, Cuda

OpenCV je voľne šíriteľná knižnica na spracovanie obrazových dát. Je podporovaná na všetkých bežných operačných systémoch. Je použiteľná pre programovacie jazyky ako C++, Python a Java. OpenCV takisto ponúka model viacvrstvého perceptronu, no kvôli využitiu konvolučných sietí sme sa v našej práci rozhodli použiť OpenCV len pre spracovanie a predprípravu datasetu. Plánujeme využívať funkcionality ako načítavanie .jpg a .png súborov, zmenu ich veľkosti, zmenu farebného spektra. Knižnica načítava obrazové dáta ako 3 rozmerné pole, kde prvé 2 rozmery sú výška a šírka, tretí obsahuje informáciu o farbe pixelu v závislosti od farebného spektra pre ktoré chceme dáta načítať. OpenCv prostredníctvom rozšírenia Cuda ponúka možnosť akcelerovania výpočtu pomocou presunutia výpočtu na grafickú kartu. To znamená masívne zrýchlenie výpočtov. S nainštalovaným Cuda modelom nie je potrebné pre vývojara sa zaoberať tým, či sa kód vykonáva na procesore alebo grafickej karte, nakoľko Cuda to vykonáva zaňho.

## 1.5 Doterajšie riešenia

### 1.5.1 Deep Convolutional Neural Network for Cloud Coverage Estimation from Snapshot Camera Images“, Ryo Onishi [OS17]

V článku sa zaoberajú využitím konvolučnej neurónovej siete na identifikáciu pokrytia oblohy oblačnosťou zo snímok zo zeme. Tento fakt odlišuje túto prácu od väčšiny prác, ktoré sa zaoberajú identifikáciou oblačnosti z radarových snímok. Tento rozdiel je veľmi podstatný nakoľko mení identifikáciu z problému oblačnosť vs. zemský povrch na problém oblačnosť vs. jasná obloha. Druhý menovaný problém je podľa nás ťažší, no bez radarových snímok je nutné ho riešiť. V práci takisto porovnávajú výsledky svm-algoritmu s neurónovou sieťou. Prvotný rozdiel medzi touto a našou prácou je určovanie oblačnosti na desatiny, oproti osminám v našom prípade. Prvotný prístup bolo použitie metódy podporných vektorov.

#### Klasifikácia pixelov oblohy

Na začiatok klasifikovali pixely na 2 triedy, a to obloha a ne-obloha, teda napr. pohoria, stĺpy, budovy. Pre túto klasifikáciu zozbierali 120 fotografií obsahujúcich výlučne oblohu, resp. oblačnosť. Pomocou tohto datasetu určili hodnoty na HSV farebnom spektre ktorým v klasifikácii prideliť označenie obloha. Pixely vo zvyšnom farebnom spektre dostali označenie ne-obloha.

#### Klasifikácia jasná obloha – oblačnosť

Bola použitá metóda podporných vektorov. Ako parametre použili RGB hodnoty, nasýtenosť v cylindrickom súradnicovom systéme, nasýtenosť v kónyckom súradnicovom systéme, hodnotu jasú a rozdiel modrej a červenej zložky rgb. Tento rozdiel definovali ako:

$$br = (b \setminus r) / (b + r) \quad (1.7)$$

kde  $b$  predstavuje modrú zložku,  $r$  červenú zložku. Parametre boli vybrané na základe najvyššej variancie medzi fotografiami vybranými ako zábery oblohy.

### Klasifikácia jasná obloha – oblačnosť

Navrhovaná architektúra predpokladá vstupné dáta upravené na rozmery  $128 \times 128 \times 3$ . Tretí rozmer predstavujú 3 zložky farebného spektra pre každý pixel. Ako prvú používa konvolučnú vrstvu, za ňou nasleduje zlučovacia vrstva s faktorom 2. Opakovaním týchto dvoch vrstiev, so zmeňšujúcim sa vstupom na každú ďalšiu konvolučnú vrstvu, sa dostanem až k piatej zlučovacej vrstve, ktorej výstup má rozmery  $4 \times 4 \times 256$ . Nasleduje plne prepojená vrstva, ktorej výstupom je vektor o dĺžke 512. Posledná vstupná vrstva má ako výstup číselnú hodnotu v rozmedzí 1 až 10 označujúcu oblačnosť.

Konvolučné vrstvy majú za úlohu extrahovať učacími sa filrami, kernelmi, vlastnosti a príznaky na vstupných dátach. S každým filtrom je použitá aktivačná funkcia na vytvorenie mapy aktivácií jednotlivých filtrov. Pooling je používaný na zmenšovanie rozmeru. V práci sa uvádza, že intuitívne je presná poloha daného príznaku menej podstatná ako jeho približná poloha vzhľadom na ostatné príznaky. Vďaka tejto intuícii teda pri použití zlučovacej vrstvy strata informácie o presnej absolútnej polohe príznaku nie je problémom.

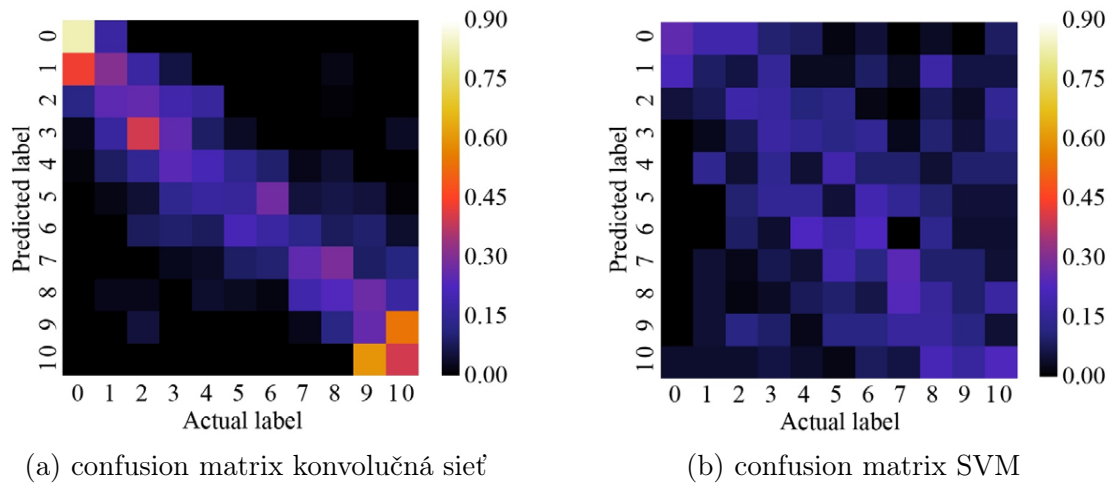
Špecifikácia kernelu pre konvolučnú vrstvu Nech má vrstva vstup o rozmeroch  $H \times N \times N$ , tento vstup je v práci mapovaný na výstup s rozmermi  $H \times H \times N$  za použitia konvolučného filtra s rozmermi  $3 \times 3 \times N$ . Ako aktivačnú funkciu použili ReLU, teda výber maximálneho komponentu.

$$f(x) = \max(x, 0) \quad (1.8)$$

### Výsledky

Po natrénovaní konvolučnej siete sa tejto práci podarilo dosiahnuť priemernú kvadratickú chybu vo výške 3. Tento výsledok považujú za úspešný nakoľko sa priblížili chybe vrámci použitého datasetu, ktorá bola 2,15. Pri porovnaní metódy podporných vektorov a konvolučnej siete má sieť jednoznačne lepšie výsledky. Konvolučná sieť

### Záver práce - prepojenie na našu prácu



Obr. 1.3: Porovnanie matíc zámeny tried

V práci úspešne vyvinuli neurónovú sieť pre klasifikáciu oblačnosti na zašumených fotkách oblohy. Chyba natrénovanej siete nebola masívne vyššia ako chyba v označeniach oblačnosti v datasete. Neurónová sieť bola porovnávaná s metódou podporných vektorov. Problémom metódy podporných vektorov bola častá klasifikácia vodných plôch ako oblohy. Riešenie tohto problému konvolyčnými sieťami bola podľa autorov schopnosť siete naučiť sa, že obloha sa zväčša nachádza vo vrchnej časti obrazu, zatiaľ čo voda v spodnej. Metóda podporných vektorov by fungovala dostatočne dobre na datasete kde sú fixne oddelené vodné plochy od oblohy.

Táto práca nás motivovala skúsiť najprv vyvinúť jednoduchšie, "proof of concept" riešenie bez použitia neurónovej siete. Ďalej pred nás predostrela spôsob ako úspešne vyvinúť model konvolyčnej siete a aplikovať ho. Dostali sme lepši odhad o potrebnej veľkosti datasetu, výpočtovej náročnosti a očakávateľných výsledkoch. Podstatný bol koncept striedania konvolyčnej a zlučovacej vrstvy.

### 1.5.2 Deep Learning for Cloud Detection [LGTW<sup>+</sup>17]

Daná práca sa zaoberá klasifikáciou oblačnosti na radarových snímkach z družice. Napriek tejto výraznej odlišnosti od našej práce sme sa rozhodli ju zaradiť do prehľadu problematiky nakoľko využíva metódy hlbokého učenia. V práci boli využité SPOT datasety radarových snímkov.

#### Výber príznakov pre hlboké učenie

Jedným z cieľov práce bolo porovnať výkon použitých metód strojového učenia na týchto príznakoch: - RGBI hodnota pre pixely na snímkach - koreláciu medzi 2 vybranými RGB kanálmi - Koeficienty Gabor-transformácie - koeficienty diskkrétnej kosínusovej transformácie

Autori zdôrazňujú dôležitosť výberu príznakov. Klasifikácia iba na základe neupravených hodnôt jednotlivých pixelov je podľa nich príliš náchylná na šum. Naopak tvorba príznakov napríklad na základe pomeru jednotlivých kanálov RGB spektra výrazne vylepšuje klasifikáciu oblačnosti. Použitie Gábor transformácie zdôvodňujú jej využitím pre kódovanie štrukturálnych vlastností snímaného povrchu.

#### Superpixels

Pre redukciu šumu na jednotlivých pixeloch autori využívajú superpixely. Skupiny podobných pixelov sa zgrupujú a vytvárajú „regióny“. Toto zgrupovanie pixelov neprebieha len na základe pozície pixelov na snímke ale aj na základe farebnosti, prípadne ďalších príznakov. Hlavnou výhodou tohto prístupu je zozbieranie štatistík pre podobné pixely, kde sú výsledky ovplyvnené šumom výrazne menej ako v prípade disktrétnych pixelov. Na základe týchto štatistík môžeme potom vytvoriť spoľahlivejšiu klasifikáciu. Autori na tento účel využili k-means algoritmus a pomocou metódy krížovej validácie rozdelili každý snímok na 250 regiónov.

#### Využitie neurónovej siete

Vstupom do siete v tejto práci nebola samotná snímka, ale množina predpočítaných príznakov. Ako chybová funkcia bola použitá „cross entropy“ funkcia

$$L(y_i, f(x_i)) = y_i * \log(f(x_i)) + (1 - y_i) * \log(1 - f(x_i)) \quad (1.9)$$



### Konvolučná vrstva

Vstupom do konvolučnej vrstvy je matica rozmermi  $M \times N \times B$  a kernel  $K$ . Výstupom je

$$F = I * K \quad (1.10)$$

kde  $*$  označuje konvolúciu

Operácia konvolúcie je definovaná ako

$$F(i, j, b) = \sum_k \sum_m \sum_n I(m, n, k) * K_b(i - m, j - n, k) \quad (1.11)$$

Autori ako spomínajú ako jednu z výhod konvolučnej siete zdieľanie parametrov naučených kernelom.

### Aktivačná vrstva

Využitá je nelineárna aktivačná funkcia, výber člena s najvyššou hodnotou.

$$h = \max(0, s) \quad (1.12)$$

### Ďalšie vrstvy modelu

Autori využili ešte zlučovaciu a plne prepojenú vrstvu, o týchto vrstvách sme ale už hovorili v predošlých kapitolách

### Výsledky práce

Pri porovnávaní presnosti klasifikácie boli dosiahnuté najlepšie výsledky pri použití príznakov ako pomer farebných spektier. Pre neupravované hodnoty pixelov a Gabor transformácie viedli naopak, k najmenej presnej klasifikácii. Využitie superpixelov zabezpečilo elimináciu šumu, jeho nevýhodou bola ale neidentifikácia menších oblakov, čo autori pripisujú zvolenej veľkosti regiónov pre superpixely.

**Záver práce - prepojenie na našu prácu**

Podstatným výsledkom tejto práce bol záver, kde autori tvrdia, že konvolučná sieť prekonala v presnosti klasifikácie autormi vybrané príznaky. Preto sme sa aj my v našej práci rozhodli zamerať na využitie neurónovej siete s konvolyčnými vrstvami. Zaujala nás možnosť využitia superpixelov a ich vplyv na redukciu šumu. Hlavným rozdielom tejto práce oproti našej bolo použitie satelitných snímok, oproti nášmu datasetu, ktorý má snímky zo zeme.

# Literatúra

- [Bro] Jason Brownlee. Gentle introduction to the adam optimization algorithm for deep learning. <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>.
- [cro] Loss functions. [https://ml-cheatsheet.readthedocs.io/en/latest/loss\\_functions.html](https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html).
- [Des] Mohit Deshpande. <https://pythonmachinelearning.pro/perceptrons-the-first-neural-networks/>.
- [dS16] Leonardo Araujo dos Santos. 2016. [https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/residual\\_net.html](https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/residual_net.html).
- [GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [LGTW<sup>+</sup>17] Matthieu Le Goff, J.-Y Tournieret, Herwig Wendt, M Ortner, and M Spigai. Deep learning for cloud detection. pages 10 (6 .)–10 (6 .), 01 2017.
- [Maz15] Matt Mazur. A step by step backpropagation example. march 2015. <https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>.
- [OS17] Ryo Onishi and Daisuke Sugiyama. Deep convolutional neural network for cloud coverage estimation from snapshot camera images. 13:235–239, 01 2017. [https://www.researchgate.net/publication/321984089\\_Deep\\_Convolutional\\_Neural\\_Network\\_for\\_Cloud\\_Coverage\\_Estimation\\_from\\_Snapshot\\_Camera\\_Images?enrichId=](https://www.researchgate.net/publication/321984089_Deep_Convolutional_Neural_Network_for_Cloud_Coverage_Estimation_from_Snapshot_Camera_Images?enrichId=)

rgreq-e33ee10867e6c80318fa13144a125516-XXX&enrichSource=Y292ZXJQYWdlOzMzMk4NDA4OTtBUzo1NzY4ODYwNTQ1MDIOMDBAMTUxNDU1MTc2MTkxMG3D%3D&el=1\_x\_2&\_esc=publicationCoverPdf.

- [Poc] Barnabas Póczos. Convolutional neural networks. [http://www.cs.cmu.edu/~10701/slides/cnn\\_notes.pdf](http://www.cs.cmu.edu/~10701/slides/cnn_notes.pdf).
- [Pok] Harsh Pokharna. The best explanation of convolutional neural networks on the internet! <https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet>
- [Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.
- [Van] Jake Vanderplas. <https://jakevdp.github.io/blog/2015/07/06/model-complexity-myth/>.
- [Wer74] P. Werbos. “beyond regression: New tools for prediction and analysis in the behavioral sciences. *Ph.D. Dissertation, Harvard University*, 1974.
- [Yan] Ji Yang. Relu and softmax activation functions! <https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions>.
- [YSAM] Zygmunt Z. Yaser S. Abu-Mostafa. <https://www.edx.org/course/learning-data-caltechx-cs1156x-0>.
- [ZW08] Bahman Zafarifar and Hans Weda. Horizon detection based on sky-color and edge features. 6822, 01 2008.