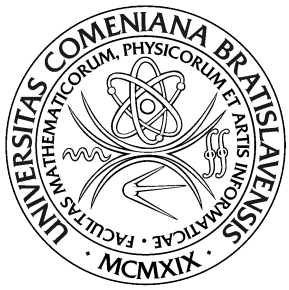
UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



Možnosti využitia metód hlbokého učenia v predpovedi počasia

Diplomová práca

2014 Bc. Juraj Mašlej

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.

Bratislava, 2018 Bc. Juraj Mašlej

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 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, odporučané 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

Poďakovanie

Todo : poďakovanie

Abstrakt

Problematika tejto diplomovej práce vychádza z meteorológie, prezentujeme problém určenia miery oblačnosti zo snímiek 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 hlobkého učenia, najmä konvolučné siete. Pre získanie základneho 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 môžme 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

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 stalo ideou pre sieť prepojení a aktivačných funkcií neurónovej siete.

Základný koncept neurónovej siete pozostáva z neurónov realizujucich logickú funkciu. Podobne ako biologické neuróny majú tieto jednotky niekoľko logických vstupov, ktoré môže zvyšovať, alebo naopak, znižovať výstupnú hodnotu. Podobnosť pokračuje aj pri výstupoch, podobne ako biologické neuróny, majú aj tie umelé jednoduchý logický výstup. Hodnota tohto výstupu závisí od vstupných hodnôt a logickej funkcie vo vnútru neurónu.

Neurón s vhodnou logickou 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.

1.1.1 Výpočet v neurónovej sieti

Dopredný výpočet – forward pass

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

$$input\_layer_1 = w_1 * i_1 + ... + w_n * i_n + b_1 * 1 \tag{1.1}$$

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 priemernú kvadratickú mocninu chyby.

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

Backpropagation – metóda úpravy váh

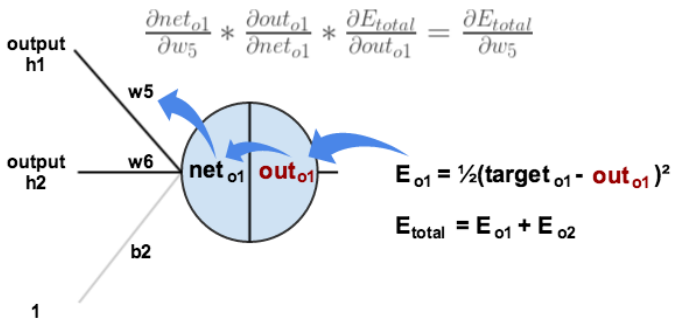
Metóda prvykrát navrhnutá Werbosom(1974) [Wer74] na úpravu váh v neurónovej sieti. Kombinuje gradient descent metódu s algoritmom na iteráciu vrstvami siete na výpočet gradientu chýb v sieti, ktorý je potrebný pre jeho znižovanie.

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. Uvažujme váhu vchádzajúcu do neurónu, označmu ju w5. Použitím reťazového pravidla dostávame:

$$\frac{\partial E_{total}}{\partial \omega_5} = \frac{\partial E_{total}}{\partial out_{o1}} \frac{\partial out_{o1}}{\partial net_{o1}} \frac{\partial net_{o1}}{\partial \omega_5} \tag{1.3}$$

Vizualizácia výpočtu na sieti: [Maz15]



Obr. 1.1: backpropagation

Pre výpočet tejto rovnice potrebujeme postupne vypočítať jednotilvé výrazy. Začneme výpočtom zmeny chyby k zmene celkového výstupu, teda out_{o1}. Ten vypočítame ako -($target_{o1} - out_{o1}$). Kedže počítame parciálnu deriváciu vzhľadom k out_{o1}, zmena chyby vzhľadom na out_{o2} je nulová. Ďalej postupujeme k výpočtu zmeny out_{o1} vzhľadom k vstupu do neurónu (net_1). Musíme poznať parciálnu deriváciu aktivačnej funkcie v neuróne Ako aktivačnú funkciu sme použili logistickú funkciu, ktorej derivácia je $\frac{1}{(1+e**-net_{o1})}$. Teda vieme vypočítať parciálnu deriváciu out_{o1} podľa net_{o1}. Posledným členom je zmena net_{o1} vzhľadom k váhe w_5. Postupujeme rovnako ako v predchádzajúcom člene. Následne všetky 3 členy vynásobíme, tým dostaneme zmenu celkovej chyby vzhľadom k váhe w_5. Pre znižovanie chyby odčítame vypočítanú hodnotu od váhy w_5, pre postupnú zmenu hodnotu ešte pred odčítaním upravíe vynásobením rýchlosťou učenia α. Tento postup opakujeme pre všetky váhy. Obdobne postupujeme pre ďalšie vrstvy siete.

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. Napríklad časové série, ktoré sa dajú interpretovať ako 1 dimenzionálna mriežka v pravidelných časových intervaloch, alebo napríklad obrazové dáta kde pixely tvoria 2 dimenzionálnu mrežovitú štruktúru. Už meno týchto sietí naznačuje fakt, že siete vykonávajú matematickú operáciu označovanú ako konvolúcia. Za konvolúciou používa väčšina týchto sieti operáciu nazývanú „pooling"(zlučovanie), teda zmenšovanie rozmeru dát. Pre modely sietí kde vrstvíme konvolučné siete 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 diskrétne meranie v čase t_x, dostaneme novú funkciu s poskytujúcu lepší odhad polohy objektu.

$$s(t) = \int_W x(a)w(t-a)da \tag{1.4}$$

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 diskrétnymi konvolučnými sieťami, preto pre definíciu kernelu môžeme použiť zápis pomocou súm.

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

Pričom g označujeme ako kernelovú funkciu

Motivácia pre konvolučné siete

Tento druh sietí využíva tri hlavné princípy, ktoré pomáhajú presnosti jeho výsledkov. A to zdieľanie parametrov, ekvivariantné zobrazenie a zriedkavé interakcie. Tieto vlastnosti zabezpečujú,že 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 parameterov

Opakované používanie jednej matice váh. V konvolučnej sieti je set váh každého kernelu použitý na všetkých dátach vstupu, teda na každej pozícií obrazu pre obrazové dáta. To znamená, že na rozdiel od učenia sa parametrov diskrétne pre každú polohu na vstupe, sa sieť učí set parametrov ktorý aplikuje na celý vstup.

Malá prepojenosť

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 budu zdieľané medzi neurónmi. Teda 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.

Ekvivariancia

Funkcia f je ekvivariantná k funkcii T ak platí:

$$f(T(x)) = T(f(x)) \tag{1.6}$$

Inak povedané, vykonanie transformácie (funkcie T) na x, je ekvivalentné k výsledku ak aplikujeme transformáciu na $f(x)$.

Zdieľanie parametrov zabezpečuje konvolučnej sieti vlastnosť nazývanú ekvivariancia k translácii. Ekvivariantná funkcia znamená, že ak sa vstup do funkcie zmení jedným smerom, zvyší alebo zníži, výstup sa upraví rovnakým smerom. Pre ekvivariantné funkcie platí $f(g(x)) = g(f(x))$. Pre príklad ekvivariancie v konvolučnej sieti uvažujme funkciu $C()$. Nech $C()$ určuje hodnotu modrej na RGB škále pre pixel na pozícií x,y. Ďalej uvažujme funkciu $g()$, takú že $g(C) = C'$, pričom $C'(x, y) = C(x - 1, y)$. Táto funkcia posúva každý pixel o jeden pixel doprava. Ak použijeme túto transformáciu na vstup, a potom použijeme konvolúciu, teda funkciu C(x,y). Výsledok bude rovnaký ako v prípade ak najprv vypočítame funkciu C a potom tranformáciu. V prípade časového radu ako vstupu to znamená, že konvolúcia vyprodukuje časový rad, ktorý ukazuje kedy sa jednotlivé javy alebo vlastnosti zachytené kernelom vyskytli. Pre obrazové dáta ako vstup konvolúcia vytvorí 2 dimenzionálnu mapu výskytu javov zachytených kernelom. Ak objekt vo vstupných dátach zmení polohu, výskyt na ňom zachytených javov sa vo výslednej mape taktiež posunie.

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.

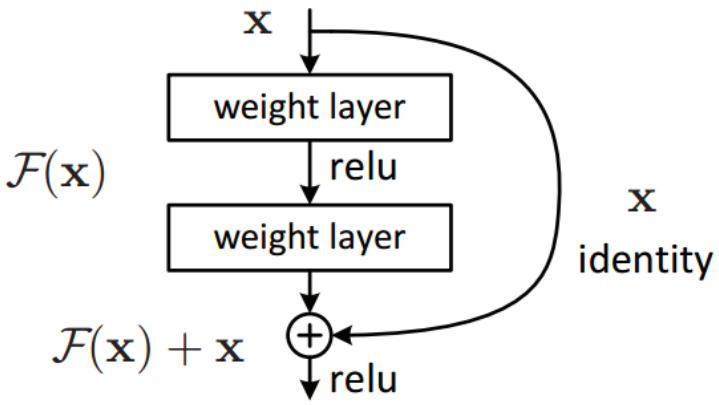
1.3 Reziduálne neurónové siete (ResNet) [dS16]

1.3.1 Motivácia

Vývoj v oblastí konvolučných sietí ukázal, že viac vrstiev siete dokáže zlepšovať výsledky klasifikácie. Pridávanie vrstiev sa ale 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 je ale 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“. Týmto chceme pridať do hlbšej vrstvy stav siete naučený vo vyššej vrstve. Výstup hlbšej vrstvy, do ktorej takéto prepojenie vchádza teda 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. 1.2: Reziduálna sieť

[dS16]

1.4 Prehľad technológií

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

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 sieti, no vybrali sme si konvolučnú sieť, pretože ju používame v našej práci.

Výber datasetu

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 slobody nášho modelu. Teda sieť s 3 váhami, by mala mať aspoň 30 vstupných dát (https://www.edx.org/course/learning-data-caltechx-cs1156x-0, http://fastml.com/how-much-data-is-enough/). Existujú ale prípady, kedy vieme úspešne natrénovať model, ktorý má viac parametrov (stupňov slobody) ako vstupných dát. Jake Vanderplas vo svojom článku (https://jakevdp.github.io/blog/2015/07/06/model-complexity-myth/) ponúka viacero modelov, ktoré môžme úspešne trénovať napriek malému počtu dát. V závere tvrdí, že zo situácie keď máme viac stupňov slobody 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 povedať potrebnú veľkosť datasetu. Ryo Onishi v práci (odkaz na lit.) uvádza použitie vyše 1700 fotografií.

Predpríprava datasetu

Je potrebné aby obrazové dáta mali rovnaké rozmery, takisto je vhodné vyskúšať viacero farebných schém. Predvýpočtom môžme 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 môžme použiť öne-hot-encode", teda kódovanie binárnym vektorom, kde na príslušnom indexe pre danú kategóriu budeme mať hodnotu 1, zvyšné budú nulové. Dospeli sme k pripravenému datasetu spolu s kódovaním výsledných kategórií.

Vytvorenie modelu siete

O teórii konvolučných sietí sme hovorili v kapitole (odkaz na kap. Konvolučné neurónové siete). Pre potreby tejto kapitoly iba zopakujme, 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 Keras-om uvažujme použitie nasledujúcich vrstiev. Takisto použijeme sekvenčný typ modelu, ktorý nám umožňuje postupne pridávať vrstvy.

Konvolučné vrstvy

Vo vytváranom modely použijeme 2 konvolučné vrstvy. Potrebujeme špecifikovať počet filtrov, teda veľkosť výstupu, aktivačnú funkciu a veľkosť kernelu. Pri špecifikovaní počtu filtrov uvažujeme koľkokrát chceme daný kernel použiť na vstupný obraz.(dobre vysvetlené : https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8). Veľkosť kernelu špecifikuje ako veľký príznak chceme zachytiť. Preto pri vyšších vrstvách nemá zmysel vytvá-

rať zbytočné veľký kernel. Pri vyšších vrstvách sa ale na rovnakom rozmere zobrazuje už spracovaná väčšia plocha vstupu, vďaka tomu môžme kernel držať rádovo menší ako vstupné dáta. Ako aktivančná funkcia je v súčasnosti pri konvolučných sieťach najčastejšie používaná ReLU funkcia (možem niečo odcitovať).

Plne prepojená a "flatten"vrstva

Flatten (rozbaľovacia?) vrstvu použijeme ako spojenie medzi konvolučnými a plne prepojenou vrstvou. Pre našu aplikáciu uvažujme flatten vrstvu ktorá jednoducho rozbalí výstup poslednej konvolučnej vrstvy na 1 rozmerný vektor. Ten sa stane vstupom pre plne prepojenú vrstvu. Počet neurónov plne prepojenej vrsvtvy budem počtom výstupných tried pre ktoré chceme vykonať klasifikáciu. Ako aktivačnú funkciu môžeme použiť "softmax"funkciu, ktorá zabezpečí, že ak zosumujeme výstup pre všetky kategórie, dostaneme sa na číslo 1. Teda výstup tejto vrstvy vieme interpretovať ako pravdepodobnosť zaradenia do jednotlivých kategórií.

Parametre modelu

Potrebujeme špecifikovať ešte rýchlosť učenia, chybovú funkciu a metódu vyhodnocovanie úspešnosti klasifikácia. Knižnica Keras ponúka nástroje pre odhad rýchlosti učenia. Jedným z nich je napríklad využitie algoritmu ädam"(https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/). Často používanou chybovou funkciou je napríklad stredná kvadratická chyba. V konvolučných sieťach je ale častejšie využívaná "cross entropy"funkcia (link, pridať vzorec). Pre klasifikáciu viac ako dvoch kategórií použijeme "multicategory - cross entropy"funkciu. Ako metriku vyhodnocovania úspešnosti siete môžme 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. Potrebujeme nastaviť počet epoch, teda koľkokrát ma výpočet cez neurónovú sieť prebehnúť na trénovacom datasete. Existuje viacero metód ako nastaviť počet epoch, pre jednoduchosť uveďme, že sieť trénujeme pokiaľ sa testovacia chyba nedostane pod nami stanovenú úroveň a zároveň zmena chyby medzi epochami je dostatočne veľká. V tomto bode môžme spustiť trénovanie modelu na časti datasetu vyhradenej na tré-

novanie. Keras ponúka funkcionalitu, 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 môžme použiť na vytváranie predikcií pre dáta, ktoré ešte modelom neprešli.

Vytváranie predpovedí

Ak sme úspešne splnili všetky predošlé kroky, sieť môžme 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 viacvrstvového perceptronu, no kvôli využitiu konvolučných sietí sme sa v našej práci rozhodli použiť OpenCV len pre spracovanie a predprípravu datasetu. Plánujeme využívať funkcionalitu ako načítavanie .jpg a .png súborov, zmenu ich veľkosti, zmenu farebného spektra. Knižnica načítavá 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 výpočtou 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 Convoltional 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 pridelili 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 jasu a rozdiel modrej a červenej zložky rgb. Tento rozdiel definovali ako:

$$br = (b \breve{} r)/(b + r) \tag{1.7}$$

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 ma ako výstup číselnú hodnotu v rozmedzí 1 až 10 označujúcu oblačnosť.

Konvolučné vrstvy majú za úlohu extrahovať učacimi sa filtrami, 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

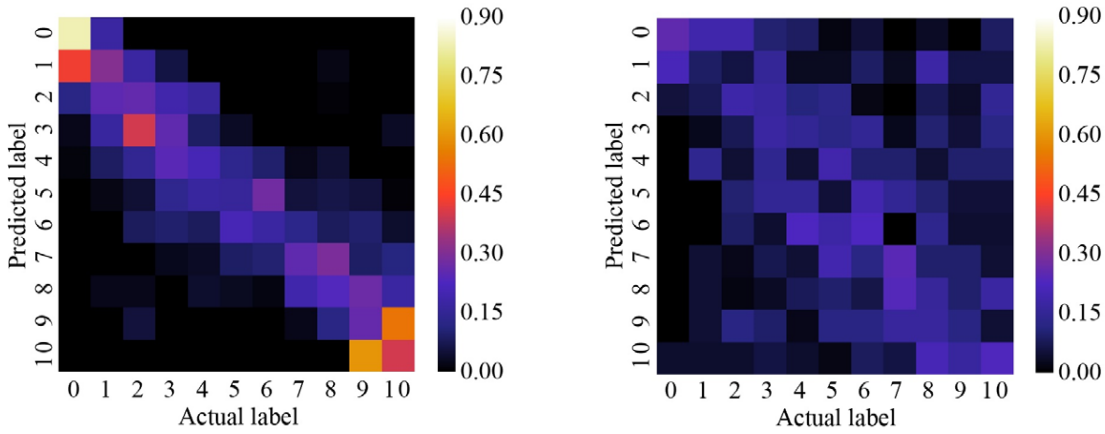
podstatná ako jeho približná poloha vzhľadom na ostatné príznaky. Vďaka tejto intuícii teda pri použití zlučovacej vrstvy strata informácie o presnej absolútnej polohe príznaku nie je problémom.

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.

$$f(x) = max(x, 0) \tag{1.8}$$

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



(a) confusion matrix konvolučná sieť (b) confusion matrix SVM

Obr. 1.3: Porovnanie matíc zámeny tried

Záver práce

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 konvolučnými 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.

1.5.2 Deep Learning for Cloud Detection [LGTW+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 diskrétnej 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.

Superpixely

Pre redukciu šumu na jednotlivých pixeloch autori využivajú 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

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

Konvolučná vrstva

Vstupom do konvolučnej vrstvy je matica rozmermi M x N x B a kernel K. Výstupom je

$$F = I * K \tag{1.10}$$

kde * označuje konvolúciu

Operácia konvolúcie je definovaná ako

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

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.

$$h = max(0, s) \tag{1.12}$$

Ďalšie vrstvy modelu

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

Výsledky

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

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 konvolučnými vrstvami.

Literatúra

[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.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[LGTW+17] Matthieu Le Goff, J.-Y Tourneret, 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=rgreq-e33ee10867e6c80318fa13144a125516-XXX&enrichSource=Y292ZXJQYWdlOzMyMTk4NDA4OTtBUzo1NzY4ODYwNTQ1MDI0MDBAMTUxNDU1MTc2MTkxMQ
3D%3D&el=1_x_2&_esc=publicationCoverPdf.

[Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[Wer74] P. Werbos. “beyond regression: New tools for prediction and analysis in the behavioral sciences. *Ph.D. Dissertation, Harvard University*, 1974.

[ZW08] Bahman Zafarifar and Hans Weda. Horizon detection based on sky-color and edge features. 6822, 01 2008.