**ŽILINSKÁ UNIVERZITA V ŽILINE**

FAKULTA RIADENIA A INFORMATIKY

DIPLOMOVÁ

PRÁCA

BRANISLAV TOMKA

**Implementácia konvolučnej hlbokej neurónovej siete pre diagnostiku rakoviny kože**

Vedúci práce: Ing. Lukáš Falát, PhD.

Registračné číslo: 988/2019

Žilina, 2020

**ŽILINSKÁ UNIVERZITA V ŽILINE**

FAKULTA RIADENIA A INFORMATIKY

BAKALÁRSKA

PRÁCA

ŠTUDIJNÝ ODBOR:

Inteligentné Informačné Systémy

BRANISLAV TOMKA

**Implementácia konvolučnej hlbokej neurónovej siete pre diagnostiku rakoviny kože**

Žilinská univerzita v Žiline

Fakulta Riadenia a Informatiky

Školiace pracovisko KMME

Žilina, 2020

#### [Čestné vyhlásenie](Pokyny_pre_vypracovanie_ZP.docx#Čestné_vyhlásenie)

Čestne prehlasujem, že som prácu vypracoval samostatne s využitím dostupnej literatúry a vlastných vedomostí. Všetky zdroje použité v diplomovej práci som uviedol v súlade s predpismi.

Súhlasím so zverejnením práce a jej výsledkov.

...........................................

V Žiline, dňa ............ Branislav Tomka

#### [Poďakovanie](Pokyny_pre_vypracovanie_ZP.docx#Poďakovanie)

„*Moje poďakovanie patrí školiteľovi diplomovej práce Ing. Lukáš Falát, PhD. za odbornú pomoc, pripomienky a usmerňovanie pri tvorbe práce.*“

[ABSTRAKT V ŠTÁTNOM JAZYKU](Pokyny_pre_vypracovanie_ZP.docx" \l "Abstrakt)

TOMKA, Branislav: ***Implementácia konvolučnej hlbokej neurónovej siete pre diagnostiku rakoviny kože***. [Diplomová práca]. – Žilinská univerzita. Fakulta Riadenia a Informatiky; Katedra makro a mikroekonomiky. – Školiteľ/Vedúci: Ing. Lukáš Falát, PhD. – Stupeň odbornej kvalifikácie: inžinier. – Žilina: FRI UNIZA, 2020. Počet strán .

[**Kľúčové slová**](Pokyny_pre_vypracovanie_ZP.docx#Kľúčové_slová)**:**

[ABSTRAKT V CUDZOM JAZYKU](Pokyny_pre_vypracovanie_ZP.docx" \l "Abstrakt_CJ)

**Key words**

[Obsah](Pokyny_pre_vypracovanie_ZP.docx#Obsah)

Zoznam obrázkov 9

Zoznam tabuliek 10

Zoznam skratiek 11

Úvod 12

1 Rakovina kože 13

1.1 Štatistiky 13

1.2 Prevencia 14

1.2.1 Metóda ABCDE 14

2 Strojové učenie 16

2.1 Typy algoritmov 16

2.1.1 Supervizované učenie 16

2.1.2 Nesupervizované učenie 17

2.1.3 Posilňujúce učenie 17

2.2 Modely 17

2.2.1 Regresná analýza 17

2.2.2 Rozhodovacie stromy 17

2.2.3 Genetický algoritmus 18

2.2.4 Umelé neurónové siete 18

3 Konvolučné neurónové siete 19

3.1 CNN a NS 19

3.2 Čo je CNN? 20

3.2.1 Konvolúcia a filter 20

3.2.2 Stride 22

3.2.3 Padding 22

3.2.4 Pooling 22

3.2.5 Aktivačné vrstvy 22

3.2.6 Plne prepojená vrstva 23

3.2.7 Softmax 23

3.3 Architektúry CNN 23

3.3.1 LeNet-5 24

3.3.2 AlexNet 25

3.3.3 VGG-16 25

4 Implementácia CNN 26

4.1 Návrh 26

4.2 Jadro 26

4.3 Vrstvy 27

4.3.1 Rozhranie ILayer 27

4.3.2 Konvolučná vrstva 28

4.3.3 Plne prepojená vrstva 29

4.3.4 Softmax 31

4.3.5 Pooling vrstva 32

Zoznam použitej literatúry 34

Zoznam príloh 35

Prílohy 36

# [Zoznam obrázkov](Pokyny_pre_vypracovanie_ZP.docx" \l "Zoznam_obrázkov)

# [Zoznam tabuliek](Pokyny_pre_vypracovanie_ZP.docx" \l "Zoznam_obrázkov)

# [Zoznam skratiek](Pokyny_pre_vypracovanie_ZP.docx" \l "Zoznam_skratiek)

# [Úvod](Pokyny_pre_vypracovanie_ZP.docx#Úvod)

# Rakovina kože

Rakovina kože sa môže vyskytnúť na hociktorom mieste na tele ale najčastejšie na koži, ktorá je často vystavená slnečnému svetlu ako napríklad tvár, krk a ruky. Hlavné typy rakoviny kože sú: skvamocelulárny karcinóm, bazocelulárny karcinóm a melanóm. Melanóm je menej častý typ rakoviny ako zvyšné dva (non-melanómové), ale má väčšiu pravdepodobnosť, že napadne okolité tkanivo a že sa rozšíri do ostatných častí tela. Najviac smrtí je spôsobených melanómom. Melanóm je choroba, pri ktorej sa v kožných bunkách vytvoria zhubné bunky.

## Štatistiky

Rakovina kože je jedna z najbežnejších typov rakoviny. Melanóm tvorí len 4% všetkých nádorov kože, avšak spôsobuje až 73% všetkých úmrtí na rakovinu kože. Približne 2,3 % mužov a žien majú diagnostikovaný melanóm kože počas života. Za rok 2019 v Spojených Štátoch Amerických pribudlo 96 480 nových prípadov s melanómom kože a 7 230 ľudí zomrelo.

|  | **Typy rakoviny** | **Odhad nových prípadov 2019** | **Odhad úmrtí 2019** |
| --- | --- | --- | --- |
| 1. | Rakovina prsníka (Ženy) | 268,600 | 41,760 |
| 2. | Rakovina pľúc a priedušiek | 228,150 | 142,670 |
| 3. | Rakovina prostaty | 174,650 | 31,620 |
| 4. | Rakovina hrubého čreva a konečníka | 145,600 | 51,020 |
| **5.** | **Melanóm kože** | **96,480** | **7,230** |
| 6. | Rakovina močového mechúra | 80,470 | 17,670 |
| 7. | Rakovina lymfatických uzlín | 74,200 | 19,970 |
| 8. | Rakovina obličiek | 73,820 | 14,770 |
| 9. | Rakovina krčka maternice | 61,880 | 12,160 |
| 10. | Leukémia | 61,780 | 22,840 |

Ročne pribudne 22,2 nových prípadov melanómu na 100 000 ľudí podľa štatistík z rokov 2012-2016. Melanóm kože sa viac vyskytuje u mužov ako u žien. U mužov je to 28.8 a u žien 17.5 nových prípadov ročne na 100 000 ľudí. Prípadov melanómu kože sa vyskytuje pri bielej rase viac ako pri ktorejkoľvek inej rase. Najčastejšie sa melanóm kože diagnostikuje osobám v rokoch 65-74 (24,1%). Percento úmrtnosti je najväčšie pri ľuďoch v rokoch 75-84 (23,9% z všetkých úmrtí).

## Prevencia

Pravdepodobnosť výskytu rakoviny kože je možné znížiť používaním ochrany proti slnečnému žiareniu a vyhýbaniu sa ultrafialovému žiareniu. Ultrafialové žiarenie je na základe aktuálnych vedeckých poznatkov klasifikované Medzinárodnou agentúrou pre výskum rakoviny (IARC) ako dokázaný karcinogén. Závery epidemiologických štúdií IARC potvrdili, že vystavenie sa UV žiareniu v mladosti najviac prispieva k poškodeniu kože a riziku rozvoja melanómu v ďalšej časti života. V súčasnosti je vo svete zaznamenávaný významný nárast melanómu kože najmä medzi mladými ženami do 30 rokov. Z tohto dôvodu bola v roku 2019 schválená novela zákona o ochrane, podpore a rozvoji verejného zdravia, ktorá sa zaoberala obmedzením osobám mladším ako 18 rokov navštevovať solária s umelým zdrojom UV žiarenia.

Ďalší spôsob ako predísť zlým následkom rakoviny kože je pravidelná kontrola buď u dermatológa, alebo samokontrolou využitím metódy ABCDE.

### Metóda ABCDE

* **A** – Asymetry (Asymetria) - Dermatológa by sme mali navštíviť v prípade, že je naše znamienko asymetrické. To spoznáme tak, že si ho pomyselne rozdelíme na dve polovice a zameriame sa na jeho tvar. Pokiaľ sú obe polovice súmerné, tak nie je treba nič riešiť. Pokiaľ je ale každá polovica iná, mali by sme zbystriť pozornosť.
* **B** – Borders (Okraje) - Okrem plasticity by sme sa mali u znamienok sústrediť aj na ich okraje. Ohraničenie melanómu totiž býva často nepravidelné či rozpité, pričom zdravé znamienko má okraje pravidelné a ostré.
* **C** – Colour (Farba) - O mnohom môže vypovedať aj farba znamienka. Pokiaľ pozorujeme, že niektorá z jeho častí zrazu zmenila farbu, mali by sme si ho nechať skontrolovať, nepravidelne škvrnité alebo čierne znamienka môžu byť nebezpečné.
* **D** – Diameter (Priemer) - Všímajte si aj veľké pehy a znamienka. Melanómy bývajú zvyčajne väčšie než zdravé znamienka a väčšinou sa ich veľkosť pohybuje nad pol centimetra v priemere.
* **E** – Evolution (Vývoj) - Môžeme povedať, že je potrebné venovať väčšiu pozornosť znamienku vždy, keď  náhle zmení tvar, veľkosť, farbu či štruktúru. V takom prípade je na mieste vyhľadať pomoc kožného lekára, môže sa totiž jednať o nebezpečný melanóm. Dôležitá je pri samovyšetrení aj pravidelnosť. Keď budeme kontrolovať znamienka pravidelne, všimneme si aj drobné odchýlky, ktoré sa na ňom mohli vytvoriť priebehom času.

# Strojové učenie

Strojové učenie je vedecký odbor algoritmov a štatistických modelov, ktorý využívajú počítačové systémy na vykonanie špecifických úloh bez použitia explicitne definovaného postupu a namiesto neho sa spolieha na vzorce a vyhodnotenia. Považuje sa za podmnožinu umelej inteligencie. Algoritmy strojového učenia tvoria matematický model na základe nejakej vzory dát, ktorú označujeme ako „trénovacie dáta“, za účelom robiť predikcie alebo rozhodnutia bez explicitného naprogramovania vykonať danú úlohu.  Mitchell, T. (1997). Machine Learning. McGraw Hill. p. 2. [*ISBN*](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [*978-0-07-042807-2*](https://en.wikipedia.org/wiki/Special:BookSources/978-0-07-042807-2). Mitchell (1997) definuje algoritmy strojového učenia “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.” Teda strojové učenie nám umožňuje vyriešiť úlohy, ktoré sú príliš náročné na vyriešenie fixnými programami napísanými a navrhnutými ľuďmi. Z vedeckého a filozofického pohľadu je strojové učenie zaujímavé pretože rozvoj nášho chápania strojového učenia si vyžaduje rozvíjanie nášho chápania zásad, ktoré sú základom inteligencie.

## Typy algoritmov

Typy algoritmov strojového učenia sa líšia v ich prístupoch, typoch dát na vstupe a výstupe a type úloh alebo problému, ktoré majú riešiť.

### Supervizované učenie

Supervizované učenie začína s cieľom predpovedať známy výstup alebo cieľ. V tomto type algoritmov sa často porovnávajú výkonnosti rôznych modelov na základe výsledkov na rovnakých datasetoch. Medzi typické problémy supervizovaného učenia patrí rozpoznávanie ručne písaného textu, klasifikovanie obrázkov rôznych objektov (napr. je to mačka alebo pes?, je to lietadlo alebo auto?) alebo klasifikovanie dokumentov (napr. je to klinické skúšanie srdcového zlyhania alebo finančná správa?). Ide predovšetkým o úlohy, ktoré môže školený človek urobiť dobre, a preto sa počítač často snaží priblížiť ľudskému výkonu. Supervizované učenie sa zameriava na klasifikáciu, ktorá zahŕňa výber z podskupín, ktorá najlepšie opisuje novú skupinu údajov, alebo predikciu, teda odhad neznámeho parametra (napr. teplota v Žiline zajtra poobede).

### Nesupervizované učenie

Na rozdiel, pri nesupervizovanom učení neexistujú žiadne výstupy, ktoré by sme mali predpovedať. Namiesto toho sa snaží nájsť prirodzene sa vyskytujúce vzorce alebo zoskupenia v rámci daného datasetu. Toto hodnotiť je už zo svojej podstaty náročnejšia úloha a často sa takéto vzorce alebo zoskupenia, ktoré sa získali prostredníctvom nesupervizovaného učenia hodnotia podľa výkonnosti v následných úlohách supervizovaného učenia (napr. sú tieto nové vzorce nejakým spôsobom užitočné?).

### Posilňujúce učenie

Posilňujúce učenie je oblasť strojového učenia zaoberajúca sa tým, ako by softvéroví agenti mali konať v prostredí s cieľom maximalizovať nejaký prínos alebo odmenu. Tento typ učenia sa odlišuje od supervizovaného učenia tým, že nepotrebuje vopred označené vstupy prislúchajúcim výstupom. Namiesto toho sa zameriava na nájdenie rovnováhy medzi skúmaním (nepreskúmaného územia) a využívaním (súčasných poznatkov).

## Modely

Vykonanie strojového učenia zahŕňa vytvorenie modelu, ktorý je natrénovaný na nejakých trénovacích dátach a potom spracúva ďalšie dáta aby robil predikcie. Existuje niekoľko rôznych typov modelov, ktoré sa používajú v strojovom učení.

### Regresná analýza

Regresná analýza zahŕňa veľké množstvo štatistických metód na odhad vzťahu medzi vstupnými premennými a ich vlastnosťami. Najbežnejšia forma regresnej analýzy je lineárna regresia, ktorá vytvára funkciu, ktorá najlepšie vyhovuje daným dátam na základe matematických kritérií, ako napríklad suma najmenších štvorcových chýb (SSE). Toto kritérium sa často rozširuje pomocou regularizačných (matematických) metód na zmiernenie pretrénovania. Pri riešení nelineárnych problémoch sa používa napríklad logistická regresia alebo polynomiálna regresia (používa sa v MS Excel na určenie trendu).

### Rozhodovacie stromy

Učenie rozhodovacieho stromu používa rozhodovací strom ako prediktívny model na prechod z pozorovaní o nejakej položke (v strome predstavuje vetvu) k záverom o cieľovej hodnote položky (v strome predstavuje list). Stromy, v ktorých cieľová premenná môže mať diskrétnu množinu hodnôt, sa nazývajú klasifikačné stromy. V týchto stromových štruktúrach listy reprezentujú označenia triedy a vetvy predstavujú spojenia prvkov, ktoré vedú k týmto označeniam triedy. Stromy, v ktorých cieľová premenná môže mať spojité hodnoty (zvyčajne reálne čísla), sa nazývajú regresné stromy. Pri rozhodovaní sa môže použiť rozhodovací strom na vizualizáciu postupnosti rozhodnutí vedúcich k danému výsledku.

### Genetický algoritmus

Genetický algoritmus je heuristika, ktorá napodobňuje proces prirodzenej selekcie pomocou metód ako mutácia a kríženie, aby sa vytvorili nové genotypy v nádeji, že sa nájdu dobré riešenia daného problému. V 80. a 90. rokoch 20. storočia sa používali genetické algoritmy v strojovom učení. Dnes sa však často kombinujú genetické algoritmy s inými modelmi strojového učenia na dosiahnutie najlepšieho výsledku.

### Umelé neurónové siete

Umelé neurónové siete boli pôvodne vyvinuté na modelovanie činnosti nervového systému mozgu. Dnes už sa však môžu použiť na riešenie veľkého množstva úloh. Umelá neurónová sieť sa rovnako ako mozgová neurónová sieť skladá z neurónov a axónov, teda prepojení týchto neurónov. Každý neurón môže prijať signál a ten následne spracovať a preposlať do ďalších neurónov. Neurónovú sieť teda tvorí veľké množstvo navzájom prepojených neurónov, ktoré si medzi sebou posielajú signál od vstupu až po výstup. Spracovanie signálu v neuróne je reprezentované nelineárnou funkciou súčtu všetkých vstupov neurónu a výsledok je jeden výstup. Učenie umelej neurónovej siete prebieha v nastavovaní váh každého neurónu pre konkrétne vstupy. V dnešnej dobe sa umelé neurónové siete používajú na veľké množstvo úloh ako napríklad rozpoznanie jazyka, prekladanie, filtrovanie na sociálnych sieťach, pri videohrách a taktiež ich veľké využitie je v medicínskej diagnostike.

# Konvolučné neurónové siete

Konvolučná neurónová sieť (CNN) je špeciálny typ neurónovej siete, ktorej hlavné využitie je klasifikovanie obrázkov. Napríklad rozhodovanie či je na obrázku mačka alebo pes, alebo rozpoznanie ručne písaného textu.

## CNN a NS

Konvolučná neurónová sieť je veľmi podobná klasickej neurónovej sieti. Pozostávajú z neurónov, ktoré majú váhy, ktoré sa sieť učí. Každý neurón dostane vstup, vykoná súčet vynásobení váh so vstupmi a prípadne sa aplikuje aktivačná funkcia. Celá sieť stále vyjadruje jednu diferencovateľnú funkciu. Z obrázku na vstupe vyprodukuje skóre jednotlivých tried na výstupe sieti. Avšak konvolučná neurónová sieť predpokladá že má na vstupe obrázok.

Hlavným dôvodom prečo sa na obrázky používajú konvolučné neurónové siete je, že obrázky sú veľké. Väčšina obrázkov, ktoré sa v dnešnej dobe používajú pri rozpoznávaní majú veľkosť 224x224 alebo väčšie. Farebné obrázky navyše bývajú reprezentované 3 farebnými kanálmi (RGB) a teda takýto obrázok má 224 \* 224 \* 3 = 150 528 vstupných bodov/pixelov. Typická skrytá vrstva v takejto sieti má 1024 uzlov, takže by sa neurónová sieť musela učiť 150 528 \* 1024 = 154 140 672 váh. Viac ako 150 miliónov váh len na jednu vrstvu. Takáto sieť by bola obrovská a bolo by takmer nemožné učiť takúto sieť. Navyše pri obrázkoch ani toľko váh nepotrebujeme, pretože ako vieme tak pixely jedného obrázka sú najviac popisujúce v rámci kontextu ich susedov. Všetky objekty obrázkov sú tvorené malými, presne umiestnenými vlastnosťami, ako napríklad kruhová dúhovka oka, alebo štvorcový hranatý roh papiera. Potom učiť sa váhy pre každý jeden pixel môže byť veľmi zbytočné.

Ďalším veľmi podstatným dôvodom nepoužívania klasickej neurónovej siete je fakt, že pozícia objektu sa môže meniť. Ak by bola sieť naučená rozpoznávať psa na obrázku, tak v prípade, že sa pes na obrázku posunie, sieť ho vôbec nemusí nájsť, pretože by sa neaktivovali potrebné neuróny a takáto sieť by bola zbytočná.

Konvolučná neurónová sieť rieši oba tieto problémy .

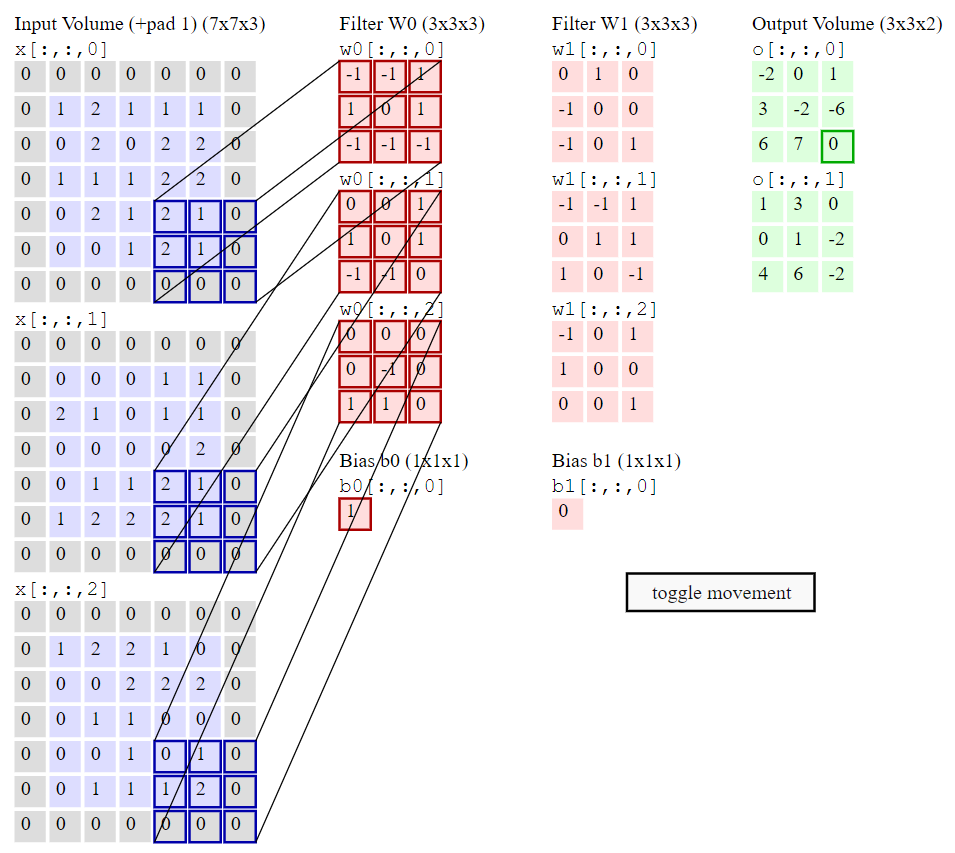
## Čo je CNN?

Konvolučné neurónové siete sú v podstate neurónové siete, ktoré používajú konvolučné vrstvy, ktoré sú založené na matematickej operácií, ktorá sa nazýva konvolúcia.

### Konvolúcia a filter

Hlavným rozdielom konvolučnej vrstvy oproti normálnej vrstve je zdieľanie váh. Každý neurón konvolučnej vrstvy sa pozerá na štvorcový výrez obrázka. Neuróny zoberú na vstup každý pixel z tohto výrezu a zdieľa váhy bez ohľadu na to, kde sa tento výrez nachádza na obrázku, takže sú rovnocenné. Ďalšia interpretácia konvolučnej vrstvy namiesto množiny rovnocenných neurónov, je jeden neurón, ktorý nazývame filter. Tento filter je štvorcová matica, ktorá obsahuje váhy a konvolúciou prechádza celý obrázok výrez po výreze. Konvolúcia pozostáva zo 4 hlavných bodov:

1. Prekrytie filtra cez jeden výrez obrázka.
2. Vykonanie násobenia po prvkoch medzi hodnotami vo filtri a ich zodpovedajúcimi hodnotami na obrázku.
3. Sčítanie všetkých výsledkov násobení po prvkoch a prípadné prevedenie cez aktivačnú funkciu ako pri klasickej vrstve. Výsledné číslo bude jeden pixel vo výstupnom obrázku.
4. Kroky sa opakujú cez všetky výrezy celého obrázka.



Na obrázku je vidieť ako postupuje konvolúcia. Na vstupe je obrázok 7x7x3 s paddingom 1 a je prechádzaný dvoma filtrami o veľkosti 3x3x3 a stride je 2, teda výstup má rozmer 3x3x2.

Filter teda môžeme interpretovať aj ako nejaký detektor vlastností, ktorý vo výsledku vráti veľké číslo, ak detekuje danú vlastnosť.



Na obrázku je 96 filtrov o veľkosti [11x11x3], ktoré sú naučené Krizhevskym. Na niektorých filtroch je jasne vidieť, ktoré vlastnosti obrázka aktivujú (rôzne okraje, tvary, ...).

### Stride

Stride je parameter konvolučnej vrstvy, ktorý určuje veľkosť kroku medzi jednotlivými výrezmi, cez ktoré prechádza filter pri konvolúcií. Takmer vždy sa tento parameter nastavuje na 1, tak aby ďalší výrez bol hneď nasledujúci. Výhoda vyššej hodnoty tohto parametra je, že zníži výšku a šírku výstupu. Avšak v praxi stride 2 alebo viac môže byť považované za príliš agresívny prístup a teda zníženie veľkosti výstupu je za cenu straty niektorých informácií.

### Padding

Pri konvolúcií pixely, ktoré sú viac pri strede sú prejdené filtrom viac krát, ako tie pri kraji obrázka, takže tam môže dôjsť k strate nejakej informácie. Aby sa predišlo takejto strate informácií, je možné nastaviť parameter pooling, ktorý pridá 0 na okraje obrázka aby nebola pridaná žiadna informácia navyše, keďže násobením 0 sa výsledná hodnota nezvýši.

### Pooling

Susedné pixely v obrázkoch zvyčajne majú podobné hodnoty, takže konvolučné vrstvy takisto dajú rovnaké hodnoty pre susedné pixely vo výstupe. Vo výsledku väčšina informácií, ktoré sa nachádzajú vo výstupe konvolučnej vrstve sú redundantné. Napríklad ak použijeme filtre na detekciu hrán a na niektorom mieste nájdeme hranu, je veľmi pravdepodobné, že hranu nájdeme aj o 1 pixel vedľa a teda je to stále tá istá hrana, takže sme nenašli nič nové.

Pooling vrstva rieši tento problém. Táto vrstva redukuje veľkosť vstupu združovaním hodnôt. Obvykle sa na pooling využívajú jednoduché matematické operácie ako min, max alebo priemer. Zvyčajne sa na združujú pixely po výrezoch veľkých 2x2 pixely. Teda šírka a výška výstupného obrázka bude o polovicu menšia ako vstupného obrázka. Napríklad pri max-pooling vyberieme z výrezu 2x2 len jedno najväčšie číslo.

### Aktivačné vrstvy

Pri konvolučných neurónových sieťach sa veľmi často hneď za konvolučnú vrstvu pridáva aktivačná vrstva, ktorá aktivuje výstup konvolučnej vrstvy niektorou z aktivačných funkcií. Zvyčajne sa používa ReLU alebo Tanh. Často pri uvádzaní konvolučnej vrstve v architektúre sa automaticky predpokladá, že súčasťou tejto konvolučnej vrstvy je aj aktivačná vrstva (najmä ReLU).

### Plne prepojená vrstva

Plne prepojená vrstva sa pridáva za konvolučné vrstvy, kde sú všetky neuróny prepojené so všetkými výstupmi predošlej vrstvy. Výstup z poslednej konvolučnej vrstvy sa roztiahne do jedného vektora, ktorý sa použije ako vstup pre túto vrstvu. Týchto plne prepojených vrstiev sa môže v CNN nachádzať aj viacej. Tieto vrstvy pracujú ako klasická neurónová sieť. Prenásobia vstupy s váhami, sčítajú ich a aplikujú na tento súčet aktivačnú funkciu.

### Softmax

Ako posledná, výstupná, vrstva CNN sa takmer vždy používa Softmax vrstva. Táto vrstva je plne prepojená vrstva s softmax aktivačnou funkciou. Táto vrstva obsahuje toľko neurónov, koľko rôznych tried je potrebné klasifikovať.

Softmax aktivačná funkcia je zovšeobecnenie sigmoidnej aktivačnej funkcie pretože tiež prideľuje pravdepodobnosť každému z neurónov (jeden neurón pre každú triedu). Pre neurón i a počet tried j, sa softmax funkcia vypočíta nasledovne:

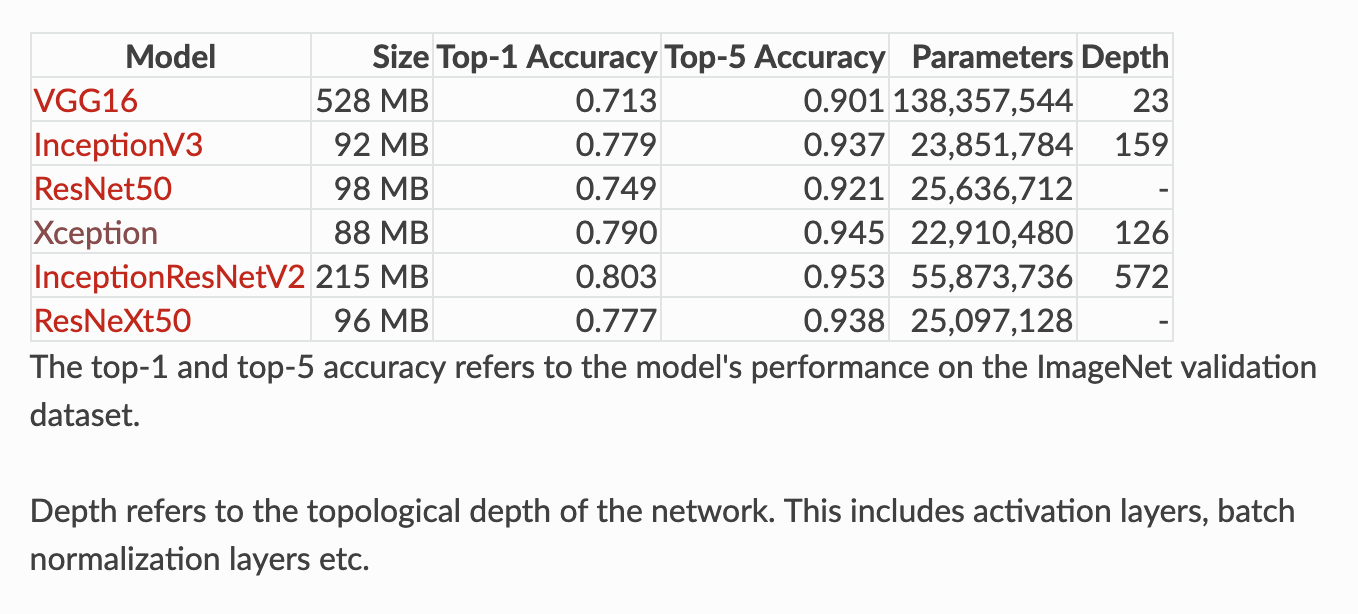
Najskôr sa vypočíta exponent váhami prenásobené vstupy neurónu i a následne sa vydelí súčtom exponentov všetkých váhami prenásobených vstupov všetkých j neurónov, aby sa normalizoval výstup a teda aby pravdepodobnosti pre všetky triedy po sčítaní dali 1.

## Architektúry CNN

Ako aj normálne neurónové siete, aj pri konvolučné sa skladajú z rôznych vrstiev. Avšak pri CNN existuje viacero rôznych typov vrstiev. Vo všeobecnosti sa konvolučné neurónové siete skladajú z troch hlavných typov vrstiev: konvolučné (vrátane aktivačnej vrstvy), pool vrstvy (najčastejšie max) a plne prepojených vrstiev (často zakončených softmax vrstvou). Najbežnejší spôsob tvorby konvolučných neurónových sietí je nakopiť niekoľko konvolučných vrstiev nasledovaných pool vrstvami a toto vrstvenie opakovať až dokým sa obrázok postupne nezmenší na veľmi malý (niekedy až na veľkosť 1 pixela). Následne sa aplikuje niekoľko plne prepojených vrstiev zakončených softmax vrstvou. Inak povedané, najbežnejší vzorec pre tvorbu architektúry CNN je nasledovný:

Zvyčajne 0 < N <= 3, M >= 0, 0 < K < 3. Samozrejme počet jednotlivých vrstiev a ich usporiadanie nie je jediná vec, ktorá ovplyvňuje výkonnosť konvolučnej neurónovej siete. Záleží aj na nastavení rôznych hyperparametrov, ako je veľkosť a počet filtrov v jednotlivých konvolučných vrstvách, stride, padding. Veľkosť výrezu v pool vrstvách, počet neurónov v plne prepojených vrstvách a mnoho ďalšieho.

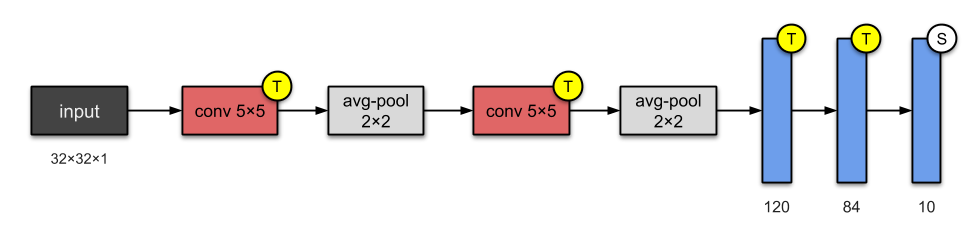
V posledných rokoch sa ukazuje vo svete veľké množstvo rôznych architektúr CNN. Mnohé z nich sú tak veľké a komplikované, že je veľmi náročné si predstaviť a zobraziť ako vyzerajú. V oblasti rozpoznávania obrazu bol zaznamenaný obrovský pokrok, avšak ako poznamenal výskumník z Googlu Christian Szegedy „most of this progress is not just the result of more powerful hardware, larger datasets and bigger models, but mainly a consequence of new ideas, algorithms and improved network architectures“. Avšak z toho veľkého množstva rôznych architektúr, existuje niekoľko veľmi často používaných sa dobrými výsledkami a je vidno že aj keď nie sú veľmi veľké, tak ich výsledky sú skvelé.



V tabuľke na obrázku je porovnaných 6 architektúr. Ich veľkosti, hĺbka a počet parametrov sa môžu veľmi líšiť a pritom výsledky sú veľmi podobné.

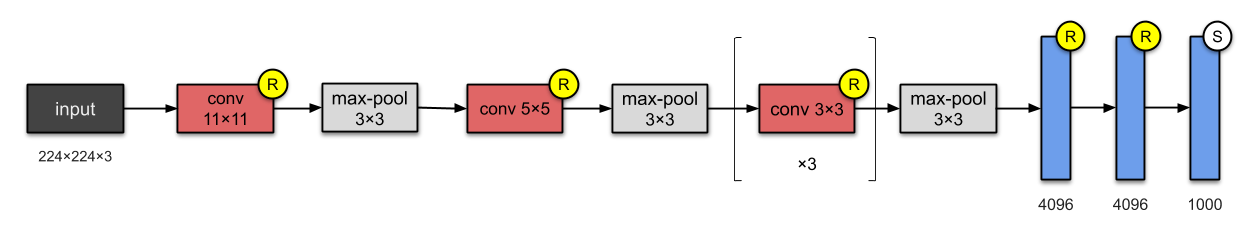
### LeNet-5

Jedna z najjednoduchších architektúr, ktorá má len 2 konvolučné a 3 plne prepojené vrstvy a len okolo 60 000 parametrov je LeNet-5 architektúra z roku 1998.



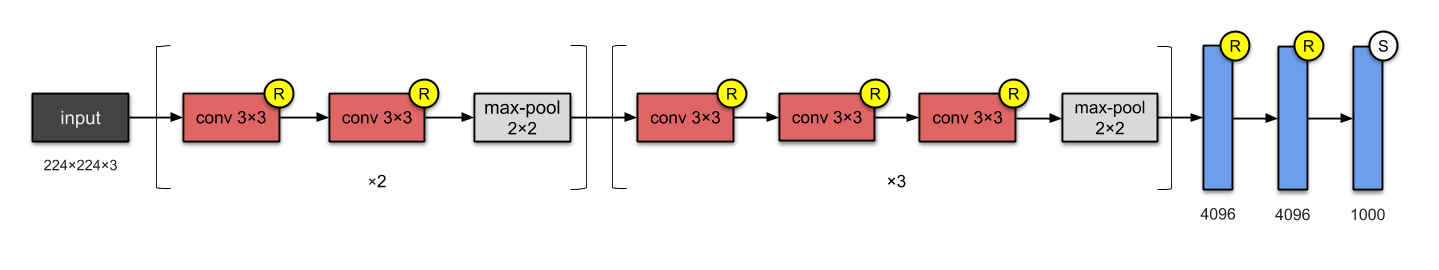
### AlexNet

Novšia architektúra z roku 2012 AlexNet ako prvá prišla z myšlienkou vymeniť Tanh aktivačnú funkciu za ReLU. Má 8 vrstiev a okolo 60 miliónov parametrov. V tej dobe to bola najväčšia konvolučná neurónová sieť použitá na ImageNet dataset.



### VGG-16

Je vidieť, že v poslednej dobe sa začali konvolučné neurónové siete zväčšovať, pretože najpriamejší a najjednoduchší spôsob ako zlepšiť CNN je zväčšiť jej veľkosť. Vo Visual Geometry Group (VGG) v roku 2014 vytvorili sieť zvanú VGG-16, ktorá má 13 konvolučných a 3 plne prepojené vrstvy. Táto architektúra pridala niekoľko vrstiev na AlexNet architektúru a použila menšie filtre (2x2 a 3x3). Pozostáva z približne 138 miliónov parametrov a zaberá až 528MB miesta. Okrem VGG-16 vytvorili aj väčšiu VGG-19.



# Implementácia CNN

Vlastná konvolučná neurónová sieť bola implementovaná v jazyku Python s využitím vývojového prostredia PyCharm IDE. V práci bol použitý Python 3.6 interpreter.

## Návrh

Sieť bola navrhnutá tak, aby bolo možné si jednoducho vytvoriť vlastnú architektúru s použitím štyroch hlavných typov vrstiev konvolučných neurónových sietí:

* Konvolučná vrstva
* Pool vrstva
* Dense vrstva
* Softmax vrstva

Všetky vrstvy implementujú spoločný interface ILayer, ktorý má tri hlavné metódy forward, backward a updateWeights. Vďaka tomu je možné jednoducho pridávať rôzne vrstvy do jedného modelu, ktorý reprezentuje trieda CNN.

## Jadro

Jadro celej aplikácie predstavuje trieda CNN, ktorá vytvára model konvolučnej neurónovej siete. Teda do tohto modelu je možné pridávať vrstvy, trénovať model a získať predikcie. Trieda taktiež vypočíta hodnotu danej loss funkcie a presnosť modelu.

Trieda má 2 hlavné atribúty:

* **layers** – je to atribút typu *List*, ktorý v sebe uchováva všetky vrstvy modelu, ktoré implementujú interface *ILayer*
* **num\_outputs** – tento atribút typu *Integer* určuje počet výstupných neurónov, tento atribút je potrebný pre niektoré loss funkcie.

Ďalej trieda obsahuje 4 metódy:

* **\_\_init\_\_** - inicializačná void metóda triedy
* **add** – void metóda, ktorá má ako parameter vrstvu a pridá túto vrstvu do zoznamu vrstiev *layers*
* **forwardBatch** – metóda, ktorá má ako parameter obrázky a triedy obrázkov. Táto metóda slúži na predikciu triedy obrázkov, všetky obrázky preženie postupne po vrstvách až k poslednej vrstve, ktorá určí pravdepodobnosti jednotlivých tried pre všetky obrázky. Následne vypočíta a vráti gradienty, priemernú presnosť a hodnotu danej loss funkcie pre dané obrázky a vopred špecifikovaný model.
* **train** – metóda slúži na trénovanie modelu. Ako parameter dostane obrázky, triedy obrázkov a *learn rate* (veľkosť kroku učenia). Najskôr metóda zavolá metódu *forwardBatch* na určenie tried a výpočet gradientu pre všetky obrázky. Výsledný gradient sa použije ako vstup do *backward* metódy poslednej vrstvy modelu, ktorá vypočíta gradient pre ďalšiu vrstvu. Následne sa zavolá metóda *updateWeights*, s danou hodnotou *learn rate* a nastavia sa váhy vo vrstve. Postupne sa takto upravia váhy pre všetky vrstvy modelu. Návratová hodnota metódy je priemerná loss funkcia a presnosť pre všetky dané obrázky.

## Vrstvy

Druhou hlavnou časťou aplikácie sú jednotlivé vrstvy CNN. Všetky vrstvy implementujú rozhranie **ILayer**, aby bolo možné ich spolu nakopiť v jednom modely.

### Rozhranie ILayer

Toto rozhranie obsahuje 3 hlavné metódy, potrebné na predikciu tried obrázkov a učenie modelu:

* **forward** – táto metóda slúži na posunutie obrázku cez danú vrstvu. Má jeden parameter *prev\_layer*, ktorý predstavuje vstup vrstvy, resp. výstup predchádzajúcej vrstvy. Metóda vracia už upravený obrázok alebo vstup danou vrstvou.
* **backward** – táto metóda slúži na výpočet gradientu, podľa ktorého sa následne upravujú váhy a ktorý sa ďalej pošle predchádzajúcej vrstve. Parameter metódy *prev\_layer* predstavuje výstup *backward* metódy ďalšej vrstvy, resp. podľa gradientu upravený výstup danej vrstvy.
* **updateWeights** – metóda upravuje váhy danej vrstvy podľa gradientu vypočítaného v metóde *backward* a parametru *learn rate*, ktorý je parametrom metódy.

### Konvolučná vrstva

Táto vrstva má na rozdiel od klasickej NN vrstvy váhy určené filtrami, ktoré sa učia hľadať určité vlastnosti obrázku, ktoré by mohli byť významné pre rozlíšenie jednotlivých tried. Avšak rovnako ako aj klasické NN vrstvy, často aj konvolučná vrstva používa aktivačné funkcie na výstup konvolúcie. V aplikácií je konvolučná vrstva definovaná triedou **Convolutional**, ktorá má nasledujúce metódy:

* **\_\_init\_\_** - inicializačná metóda vrstvy. Inicializuje hlavné parametre triedy, ktoré sú aj parametrami tejto metódy:
  + *prev\_layer\_size* – veľkosť predchádzajúcej vrstvy, resp. hĺbka vstupného obrázka
  + *num\_filters* – počet filtrov
  + *filter\_size* – veľkosť filtrov. Číslo je typu *Integer*, pretože všetky filtre sú štvorcové, teda pri *filter\_size = 3*, budú mať všetky filtre rozmer 3x3
  + *stride* – určuje veľkosť kroku pre operáciu konvolúcie, po koľkých pixeloch sa má posúvať filter cez obrázok
  + *padding* – určuje koľko nulových pixelov okolo obrázka je treba pridať. Používa sa, ak sa významné časti obrázka nachádzajú pri kraji, alebo ak chceme zachovať veľkosť obrázka aj po vykonaní operácie konvolúcie
  + *initializer* – *String* typ inicializácie váh filtrov
  + *activation* – *String* typ aktivačnej funkcie, ktorá sa aplikuje na výstup operácie konvolúcie

V metóde sa taktiež inicializujú váhy a bias

* **initializeWeights** – metóda inicializuje váhy filtrov konvolučnej vrstve podľa zadaného typu inicializácie, ktorý je ako parameter metódy.
* **convolve** – metóda vykonáva operáciu konvolúcie nad obrázkami, ktoré sú zadané ako parameter metódy. Vstupný parameter metódy sú obrázky, následne sa vykoná konvolúcia so všetkými filtrami pre všetky obrázky a výsledný obrázok sa vracia ako návratová hodnota.
* **backprop** – metóda má ako parameter gradientom upravený výstup, ktorý sa upraví na požadovanú veľkosť, aby po vykonaní konvolúcie nad týmto výstupom mal obrázok veľkosť vstupu danej vrstvy. Takýto upravený vstup vráti metóda ako návratovú hodnotu pre ďalší krok backpropagation.
* **filterSet** – podľa zadaného upraveného výstupu metóda vypočíta priemerný gradient pre všetky obrázky, ktorý sa následne vynásobí veľkosťou kroku učenia alebo sa upraví nejakým optimalizačným algoritmom (napr. Adam).
* **activate** – táto metóda sa zavolá po vykonaní operácie konvolúcie nad obrázkom a „aktivuje“ tento výstup zavolaním konkrétnej aktivačnej funkcie. Názov aktivačnej funkcie je parameter tejto metódy a jej výstup je výstupom danej aktivačnej funkcie.
* **derivate** – ak bola pri procese backpropagation na výstup aplikovaná aktivačná funkcia je potrebné výstup vynásobiť deriváciou tejto aktivačnej funkcie. Na to slúži táto metóda, ktorá má ako parameter názov aktivačnej funkcie a výstupom je výstup derivácie aktivačnej funkcie.

### Plne prepojená vrstva

V konvolučnej neurónovej sieti predstavuje plne prepojená vrstva klasickú vrstvu neurónovej siete, kde všetky neuróny sú prepojené so všetkými vstupmi a každému prepojeniu je priradená váha. Každý pixel obrázka vstupujúceho do plne prepojenej vrstvy predstavuje jeden výstupný neurón vrstvy. Keďže výstup má na rozdiel od konvolučných vrstiev len jeden rozmer (počet neurónov siete) po plne prepojenej vrstve už nenasleduje konvolučná vrstva ale len ďalšie plne prepojené vrstvy.

Hlavnou úlohou filtrov konvolučnej vrstvy je naučiť sa a nájsť rôzne vlastnosti obrázkov (*feature extraction*) a hlavnou úlohou plne prepojenej vrstvy je určiť významnosť jednotlivým vlastnostiam a naučiť sa kedy sú, ktoré vlastnosti významné (*feature selection*).

V aplikácií je plne prepojená vrstva definovaná triedou **dense** a špeciálny prípad plne prepojenej vrstvy používajúci sa ako výstupná kategorizačná vrstva je trieda **softmax**. Medzi hlavné metódy triedy dense patria:

* **\_\_init\_\_** - inicializačná metóda triedy. Inicializuje hlavné parametre triedy, ktoré sú aj parametrami tejto metódy:
  + *layer\_size* – počet neurónov vrstvy
  + *prev\_layer\_size* – počet neurónov predchádzajúcej vrstvy. V prípade že je predchádzajúca vrstva konvolučná, tento parameter udáva veľkosť hĺbky výstupného obrázka.
  + *initializer* – typ inicializácie váh filtrov určený v podobe reťazca. Základná hodnota tohto parametra je *xavier\_uniform,* čo predstavuje inicializáciu váh rovnomerným rozdelením upraveným podľa Xavierovej inicializácie.
  + *activation* – typ aktivačnej funkcie určený v podobe reťazca. Základná hodnota tohto parametra je *relu,* ktorá je najpoužívanejšia aktivačná funkcia.

Metóda taktiež inicializuje váhy prepojení a ich biasy.

* **initializeWeights** – metóda inicializuje váhy všetkých prepojení podľa zadaného typu inicializácie, ktorý je ako parameter metódy.
* **activate** – táto metóda sa zavolá nad vstupom pri predikovaní triedy obrázka. Metóda najskôr zavolá metódu *\_\_calculate\_potential\_\_*, ktorá vynásobí vstup váhami a pripočíta biasy. Následne sa tento výstup „aktivuje“ zavolaním konkrétnej aktivačnej funkcie. Názov aktivačnej funkcie je parameter tejto metódy a jej výstup je výstupom danej aktivačnej funkcie.
* **derivate** – pri procese backpropagation sa na výstup aktivačnej funkcie aplikuje derivácia danej aktivačnej funkcie zavolaním príslušnej metódy. Názov aktivačnej funkcie je parametrom metódy a výstupom je výstup derivácie aktivačnej funkcie.
* **\_\_calculate\_potential\_\_** - metóda je volaná len z metódy *activate* a ako parameter je vstup danej vrstvy, ktorý sa v tejto metóde vynásobí s príslušnými váhami jednotlivých spojení a ku každému neurónu sa ešte pripočíta hodnota biasov. Výsledok metóda vráti ako návratovú hodnotu.
* **forward** – metóda prejde postupne cez všetky vstupy, vyrovná ich do jednorozmerného poľa a na každý vstup zavolá metódu *activate*. Návratová hodnota je pole „aktivovaných“ všetkých vstupov.
* **backprop** – parametrom metódy je upravený výstup z predchádzajúcej vrstvy pri procese backpropagation. Metóda postupne prechádza všetky takto upravené výstupy, vynásobí ich deriváciou aktivačnej funkcie a váhami a takto upravené vstupy následne vráti ako pole v návratovej hodnote.

### Softmax

Špeciálnym prípadom plen prepojenej vrstvy je softmax vrstva, ktorá sa v sieti nachádza vždy najviac raz a používa sa ako výstupná vrstva, pretože jej úlohou je určiť podľa daných vstupov pravdepodobnosť každej triede. Na základe týchto pravdepodobností sa následne robí predikcia (trieda s najväčšou pravdepodobnosťou = predikovaná trieda).

V aplikácií je táto vrstva definovaná triedou **softmax** a medzi jej hlavné metódy patria:

* **\_\_init\_\_** - inicializačná metóda triedy. Inicializuje hlavné parametre triedy, ktoré sú aj parametrami tejto metódy:
  + *layer\_size* – počet neurónov vrstvy
  + *prev\_layer\_size* – počet neurónov predchádzajúcej vrstvy. V prípade že je predchádzajúca vrstva konvolučná, tento parameter udáva veľkosť hĺbky výstupného obrázka.
  + *initializer* – typ inicializácie váh filtrov určený v podobe reťazca. Základná hodnota tohto parametra je *xavier\_uniform,* čo predstavuje inicializáciu váh rovnomerným rozdelením upraveným podľa Xavierovej inicializácie.

Metóda taktiež inicializuje váhy prepojení a ich biasy.

* **initializeWeights** – metóda inicializuje váhy všetkých prepojení podľa zadaného typu inicializácie, ktorý je ako parameter metódy.
* **forward** – metóda vynásobí vstupy príslušnými váhami, pripočíta biasy a následne určí podľa výsledku pravdepodobnosť všetkým triedam. Tieto pravdepodobnosti vracia ako návratovú hodnotu.
* **backprop** – parametrom metódy sú upravené pravdepodobnosti podľa loss funkcie. Metóda z týchto upravených pravdepodobností vypočíta upravený vstup, ktorý vráti ako návratovú hodnotu pre ďalšie vrstvy.

### Pooling vrstva

Posledná vrstva, ktorá bola implementovaná je Pooling vrstva. Jej hlavnou úlohou je zmenšiť obrázok, tak aby všetky dôležité informácie v obrázku boli zachované. Používa sa vždy za konvolučnou vrstvou. Zvyčajne sa používajú dva hlavné typy poolingu max a average, preto v aplikácií boli implementované len tieto dva, ktoré sú implementované v triede **pool** s metódami:

* **\_\_init\_\_** - inicializačná metóda triedy. Inicializuje hlavné parametre triedy, ktoré sú aj parametrami tejto metódy:
  + *stride* – veľkosť kroku, ktorý určuje o koľko sa má posúvať filter, ktorý zmenšuje obrázok
  + *filter\_size* – veľkosť filtra, ktorý prechádza obrázkom a zmenšuje jeho veľkosť. Rovnako ako pri konvolučnej vrstve je tento parameter hodnoty *Integer*, pretože sú filtre vždy štvrocové a teda rovnako pri hodnote *filter\_size = 2* bude mať filter rozmer 2x2.
  + *pool\_type –* typ poolingu typu *String* (max/avg)
* **maxPooling** – metóda filtrom postupne prejde cez celý vstupný obrázok a vždy vyberie z filtra jedno najväčšie číslo a takto sa zmenší veľkosť a zachovajú sa najvýznamnejšie pixely
* **avgPooling** *–* metóda podobne ako *maxPooling* prejde filtrom cez celý obrázok ale namiesto vybratia najväčšieho čísla vypočíta priemer zo všetkých čísel.

Síce trieda nemá žiadne váhy, ktoré sa učí, pri procese backpropagation musí upraviť výstup tak aby mohol pokračovať v procese. Na úpravu (zväčšenie) výstupu používa trieda dve metódy:

* **maxBackprop** – v metóde *maxPooling* si metóda ukladala indexy vybratých prvkov. Táto metóda postupne prechádza výstupom a vkladá prvky na zväčšený obrázok podľa indexov uložených v metóde *maxPooling* a teda všetky pixely, mimo tých, ktoré obsahovali vybraný pixel majú hodnotu 0.
* **avgBackprop** *–* každý pixel výstupu vynásobí 0.25 a rozšíri ho na rozmer filtra a tým sa obrázok rozšíri do požadovanej veľkosti.

# [Zoznam použitej literatúry](Pokyny_pre_vypracovanie_ZP.docx#Zoznam_použitej_literatúry)

# [Zoznam príloh](Pokyny_pre_vypracovanie_ZP.docx" \l "Prílohy)

# 

# Prílohy