SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 533

PRIMJENA KONVOLUCIJSKIH MODELA NA PROBLEM KLASIFIKACIJE SLIKA

Renato Jurišić

Zagreb, lipanj 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 533

PRIMJENA KONVOLUCIJSKIH MODELA NA PROBLEM KLASIFIKACIJE SLIKA

Renato Jurišić

Zagreb, lipanj 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 11. ožujka 2022.

ZAVRŠNI ZADATAK br. 533

Pristupnik:	**Renato Jurišić (0036521711)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	izv. prof. dr. sc. Zoran Kalafatić

Zadatak: **Primjena konvolucijskih modela na problem klasifikacije slika**

Opis zadatka:

Duboki konvolucijski modeli pokazali su se vrlo uspješnima na raznim zadacima računalnog vida, posebno u klasifikaciji slika. U okviru završnog rada treba proučiti duboke konvolucijske modele opisane u literaturi, s posebnim naglaskom na model EfficientNet. Ispitati nekoliko popularnih konvolucijskih modela na javno dostupnim skupovima podataka za klasifikaciju slika. Analizirati dobivene rezultate u pogledu točnosti klasifikacije te računske zahtjevnosti. Radu priložiti izvorni i izvršni kôd razvijenih postupaka, ispitne slike i rezultate, uz potrebna objašnjenja i dokumentaciju.

Rok za predaju rada: 10. lipnja 2022.

Zahvaljujem mentoru izv. prof. dr. sc. Zoranu Kalafatiću na usmjeravanju i stručnim savjetima tijekom Projekta R i tijekom pisanja ovog rada.

Zahvaljujem svim nastavnicima, asistentima i ostalom osoblju fakulteta na posljednje tri godina studija u kojima sam puno naučio.

Zahvaljujem obitelji i prijateljima što mi život ispunjavaju srećom i ljubavlju.

Sadržaj

Uvod

Računalo radi točno ono što mu je rečeno u programskom jeziku i ništa više. Ako je potrebno riješiti računsko zahtjevan problem poput provjere da li je neku broj prost ili nije, onda će se poznati algoritam napisati u proizvoljnom jeziku i dati računalu na izvršavanje. Međutim, ako nam algoritam dolaska do rješenja nije poznat onda ne znamo eksplicitno reći računalu kako da riješi problem. Za neke probleme znamo da algoritam postoji jer ih ljudi rješavaju svakodnevno, ali ne znamo kako. Prepoznavanje objekata, snalaženje u prostoru, praćenje pokreta i slični problemi spadaju u kategoriju problema računalnog vida. Čovjek te problema naizgled rješava bez imalo truda, no te iste probleme je teško riješiti računalom. Kada bi računala mogla riješiti probleme koje sada bolje rješavaju ljudi rekli bismo da su ona inteligentna. Govorimo o simuliranoj ili emuliranoj inteligenciji tj. umjetnoj inteligenciji. Umjetna inteligencija može se definirati kao sposobnost računala da rješava probleme za koje je tipično potrebna ljudska inteligencija.

Prije nego što je čovječanstvo poletjelo avionima, promatralo je ptice. Prije nego što je Isaac Newton otkrio zakon gravitacije i izumio matematičku analizu, promatrao je planete. Također, prvotni pokušaji rješavanja problema računalnog vida pratila su istu analogiju. Ideja je sljedeća: simulirati ili emulirati oko i dio mozga koji procesira sliku. Kamere su dostupne od 1816. godine, tako da je problem oka riješen. Ono što je malo teže za računalo je interpretacija piksela sa slike.

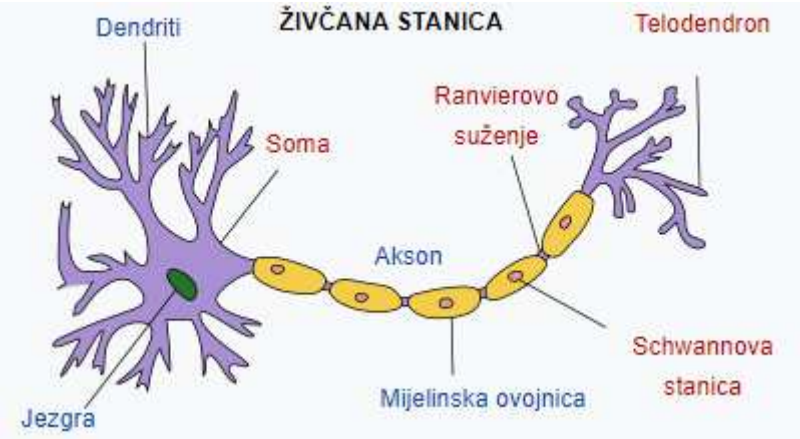
Cilj rada je objasniti današnje metode rješavanja problema klasifikacije slike tj. objasniti kako računalo može prepoznati objekt koji se nalazi na slici. Rad je podijeljen u tri dijela. Prvi dio rada opisuje neuronske mreže, formulira problem klasifikacije, daje uvid u metodu treniranja neuronskih mreža, objašnjava metodu prijenosa učenja te augmentaciju podataka. Drugi dio proučava moderne konvolucijske mreže s posebnim naglaskom na model EfficientNet. U trećem dijelu su opisani rezultati treniranja modela na skupovima podataka za klasifikaciju. Korišteni su oxford_iit_pet, oxford_flowers102, i uc_merced skupovi podataka.

1. Strojno učenje

Strojno učenje je podgrana umjetne inteligencije koja se bavi izradom inteligentnih sustava koji su sposobni iz dostupnih podataka razaznati uroke i prepoznati ih u budućim podacima. Taj proces učenja uzoraka naziva se strojno učenje i može se definirati kao sposobnost računala da se poboljša u nekom zadatku nakon što vidi više primjera točnih rješenja. Inicijalno se postupak pronalaženja uzoraka temeljio statistici što je zahtijevalo puno ručnog rada, dok danas posebnu pažnju dobivaju neuronske mreže koje su inspirirane ljudskim mozgom. One su sposobne u podacima same razaznati uzorke i njihovu važnost.

1.1. Biološki neuron

Osnovna građevna jedinica u mozgu je živčana stanica (neuron) (Slika 1). Sadrži dendrite koji sa susjednih neurona primaju podražaje. Ako su ti podražaji dovoljno jaki šalje se električni impuls preko aksona sve do telodendronova koji na svojim krajevima imaju sinapse. Ako je impuls dovoljno jak on će uzbuditi sinapse koje će otpustiti signalne molekule zvane neurotransmiteri, a oni će pak uzbuditi dendrite na susjednim neuronima [3].



Slika 1: Biološki neuron.[1]

Funkcija neurona sama po sebi je jednostavna, ali snaga je u brojkama. U ljudskom mozgu svaki neuron povezan je s tisućama ostalih, a ima ih na desetke milijardi. Zajedno obrađuju ogromnu količinu informacija koje dolaze s osjetila i na temelju njih zaključuju. Sposobni su pisati pjesme, maštati, dokazivati matematičke teoreme i dr. Mozak je

[1] Preuzeto iz [3]

najkompleksniji organ u ljudskom tijelu, a njegova struktura i funkcionalnost se još uvijek slabo razumije. Međutim, neki dijelovi mozga su mapirani i čini se da su neuroni posloženi u slojevitu strukturu.

1.2. Neuronske mreže

Osnovna građevna jedinica neuronskih mreža je neuron. Neuron se može promatrati kao matematička funkcija koja obavlja skalarni produkt svojih ulaza i nad tim primijeni neku aktivacijsku funkciju f.

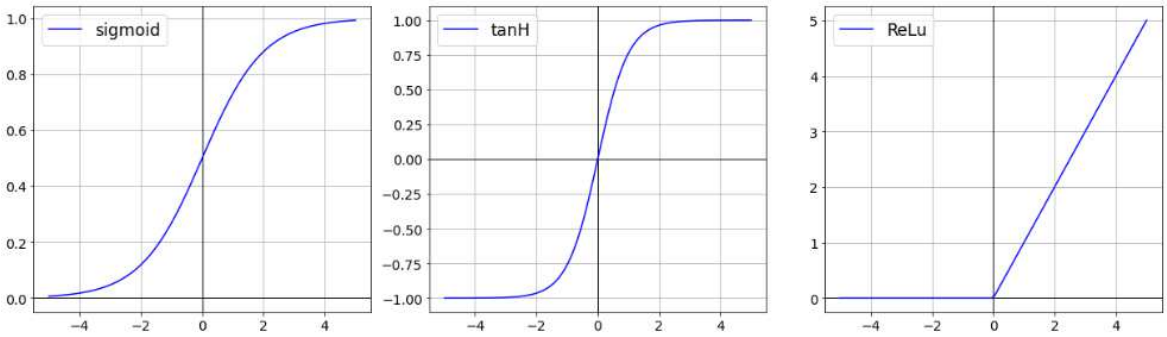
$$y = f(\vec{x} \cdot \vec{w} + b) \tag{1.1}$$

Glavna uloga aktivacijske funkcije je da razbije linearnost mreže, jer bez nje mreža može aproksimirati samo linearne funkcije. Konkretna funkcija ovisi o problemu i izlazu kojeg želimo, a u praksi su se najbolje pokazale sigmoid (1.2), tanH (1.3) i ReLu (1.4) (Slika 2). ReLu je najzastupljenija aktivacijska funkcija jer se može vrlo efikasno izračunati i jer su joj gradijenti različiti od nule za pozitivne vrijednosti. Međutim, nije ni ona savršena jer pati od problema zvanog umirući neuroni gdje se tijekom treniranja može dogoditi da neuron na izlaz ne daje ništa osim nule. Zbog toga postoje razne varijacije Relu funkcije od kojih su poznatije cureći Relu i ELU.

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{1.2}$$

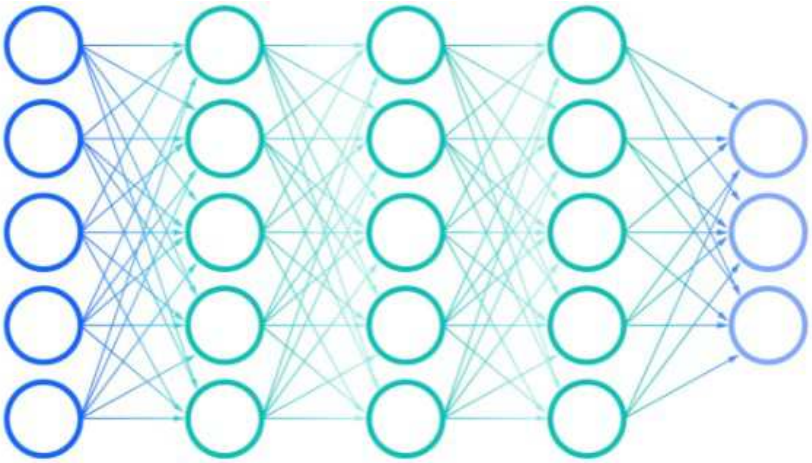
$$\tanh(x) = 2sigmoid(2x) - 1 \tag{1.3}$$

$$ReLu(x) = \max(0, x) \tag{1.4}$$



Slika 2: Sigmoid funkcija (lijevo), tanH funkcija (sredina), ReLu funkcija (desno)

Također, kao i kod bioloških neurona, do zanimljivijeg ponašanja dolazi se tako da se više neurona grupira u slojevitu strukturu koju onda zovemo neuronska mreža (Slika 3).



Slika 3: Neuronska mreža.[2]

Prvi sloj u neuronskoj mreži poprima vrijednosti koje dovedemo na ulaz. Zato se zove ulazni sloj. Neuroni u svakom sljedećem sloju uzimaju vrijednosti onih neurona iz prijašnjeg sloja s kojima su povezani i pomoću tih vrijednosti računaju svoj izlaz. Taj postupak se nastavlja sve do zadnjeg sloja u mreži kojeg zovemo izlazni sloj.

Slojevi između ulaza i izlaza nazivaju se skriveni slojevi. Njihove vrijednosti su neka kompleksna funkcija ulaza i potrebni su zadnjem sloju kako bi došli do rješenja, ali se ne zna što oni predstavljaju. Stoga se neuronske mreže često uspoređuju s crnim kutijama jer se ne zna kako mreža dolazi do rješenja. Na ulaz kutije se dovedu podaci i na izlaz se dobije rješenje.

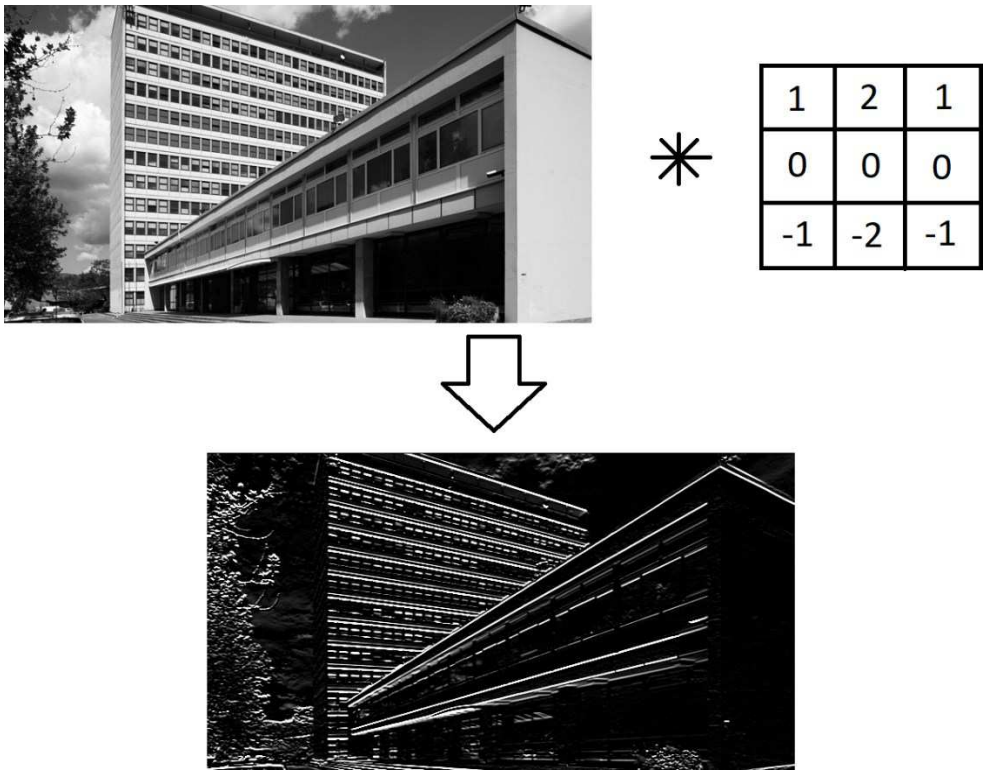
1.3. Slojevi u neuronskoj mreži

1.3.1. Konvolucijski sloj

Najvažniji sloj u konvolucijskim neuronskim mrežama je konvolucijski sloj, po kojem su i dobile ime. Njegova zadaća je iz slike izvući uzorke i karakteristike koje će se dublje u mreži koristiti pri zaključivanju, a to ostvaruje tako da na vrijednosti koje dobije na ulazu primijeni određene filtere. Filter je matrica određene veličine, a primjenjivanje filtera je samo množenje ulazna s tom matricom (Slika 4). Ako je filter manji od slike, a obično jest, onda translatiramo filter s lijeva na desno, pa kada dođemo do kraja prelazimo u novi red. U svakom trenutku računamo umnožak filtera s vrijednostima koje prekriva [5].

[2] Preuzeto s https://www.ibm.com/cloud/learn/neural-networks

Na izlaznoj slici će biti istaknute značajke ovisne o primijenjenom filteru, npr. na sljedećoj slici (Slika 4) primijenjen je filter koji ističe horizontalne linije. Ručno izrađivanje filtera mukotrpan je posao, a često i ne znamo koji filteri su najbolji za dani problem. Zato odabir filtera potpuno prepuštamo mreži koja će naučiti dobar filter tijekom treniranja.

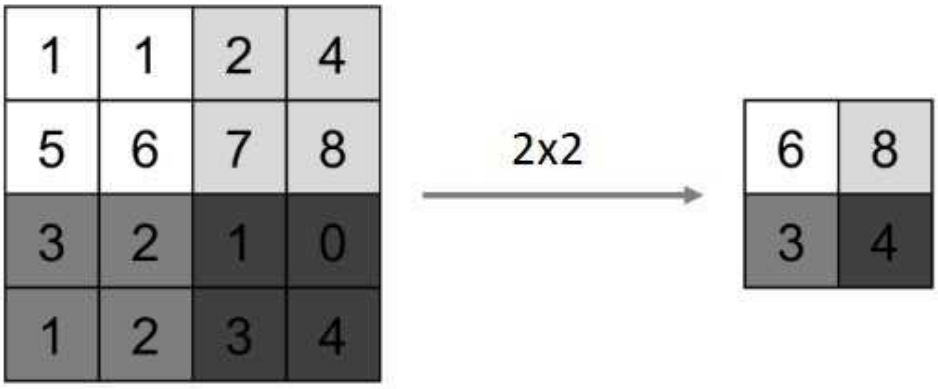


Slika 4: Primjena filtera koji ističe horizontalne linije.

Najniži konvolucijski slojevi u mreži raspoznavati će jednostavne karakteristike poput linija i rubova. Sljedeći će onda koristiti te zaključke kako bi prepoznali npr. geometrijske oblike. Dakle, svaki konvolucijski sloj kombinira karakteristike prethodnog sloja kako bi on prepoznao neke kompleksnije karakteristike.

1.3.2. Sloj sažimanja

Sloj sažimanja jednostavan je sloj koji na neki način agregira vrijednosti s ulaza i time smanjuje dimenzije slike. Preko ulazne slike postavimo rešetku i iz svake se ćelije uzima predstavnik po nekom kriteriju npr. maksimalna (Slika 5) ili prosječna vrijednost. Sloj služi kako bi smanjio računsku složenost te kao sredstvo regularizacije.



Slika 5: Primjena maksimalne agregacije (engl. *max pool*).[3]

1.4. Klasifikacija slika

Klasifikacija slike je zadatak pridjeljivanja jedne ili više klasa slici, tj. za dani ulaz pokušava se prepoznati objekt na slici i svrstati ga u neku od prije definiranih klasa.

Neuronska mreža koja rješava ovaj problem u zadnjem sloju ima onoliko neurona koliko želimo prepoznati klasa. Također, zadnji sloj ima „softmax“ aktivaciju (1.5) koja ima svojstvo da zbroj vrijednosti neurona iznosi jedan. Zbog toga se vrijednost svakog neurona može interpretirati kao vjerojatnost da je na slici ona klasa za koju je taj neuron određen.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{1.5}$$

Postoje različite metrike koje ocjenjuje performanse našeg sustava na problemu klasifikacije. Najčešće korišteni su točnost, preciznost i odaziv. Točnost je omjer primjera koji su dobro klasificirani naspram ukupnog broja primjera. Preciznost i odaziv definirani su posebno za svaku klasu. Preciznost je omjer broja primjera kojima je sustav točno odredio klasu naspram ukupan broj primjera koje je svrstao u tu klasu. Odaziv je omjer broja primjera kojima je sustav točno odredio klasu naspram ukupnog broja primjera za tu klasu.

$$točnost = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.6}$$

$$preciznost = \frac{TP}{TP + FP} \tag{1.7}$$

$$odaziv = \frac{TP}{TP + FN} \tag{1.8}$$

[3] Preuzeto s https://matlab1.com/max-pooling-in-convolutional-neural-network/

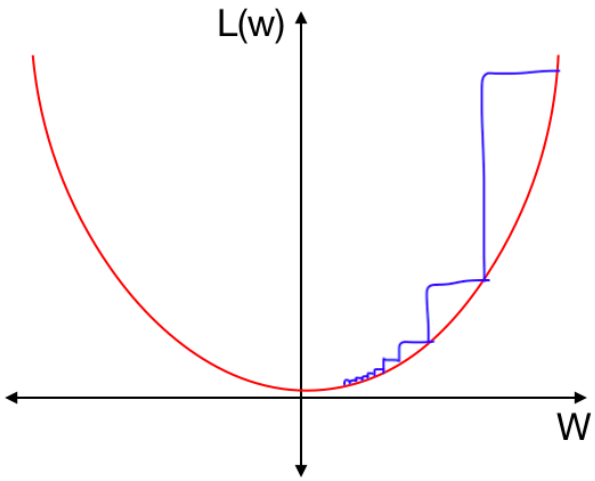
- TP – broj primjera gdje je sustav točno prepoznao pozitivnu klasu
- TN – broj primjera gdje je sustav točno prepoznao negativnu klasu
- FP – broj primjera gdje je sustav krivo prepoznao negativnu klasu
- FN – broj primjera gdje je sustav krivo prepoznao pozitivnu klasu

Razlikujemo nadgledanu klasifikaciju i nenadgledanu klasifikaciju. Razlika između ta dva problema je u tome da li skup podatka na kojem se trenira naš sustav sadrži tražene izlaze. Ovaj rad obrađuje nadgledanu klasifikaciju gdje su dostupni traženi izlazi.

1.5. Treniranje neuronskih mreža

Tek inicijalizirana neuronska mreža nema nikakvu prediktivnu moć. U suštini radi se množenje i zbrajanje parametara i ulaza dok ne dobijemo izlaz. Cilj je treniranja ažurirati parametre tako da to množenje i zbrajanje daje željeni izlaz. U svrhu procjene performansi definira se funkcija troška koja govori koliko izlazi neuronske mreže odstupaju od željenih izlaza za dani ulaz. Onda je samo potrebno tu funkciju minimizirati gradijentnim spustom.

Spust gradijentom iterativan je proces koji u svakoj iteraciji računa funkciju troška i nastoji ju smanjiti tako da ažurira parametre suprotno od smjera gradijenta (Slika 6).



Slika 6: Gradijenti spust.[4]

Za računanje gradijenata koristi se algoritam zvan propagacija u nazad. On se može podijeliti u dva koraka: prolaz u naprijed i prolaz u nazad. Tijekom prolaza u naprijed na ulaz se dovede jedna hrpa podataka. Sada sloj nakon ulaza izračuna svoje vrijednosti i

[4] Preuzeto s https://towardsdatascience.com/https-medium-com-reina-wang-tw-stochastic-gradient-descent-with-restarts-5f511975163

proslijedi ih sljedećem sloju. Tako svaki sloj izračuna svoje vrijednosti i zapamti ih jer će biti potrebni tijekom prolaza u nazad. Jednom kada je dostupna vrijednost izlaznog sloja računa se funkcija troška. Za klasifikaciju funkcija troška izgleda ovako:

$$L(w) = -\sum_{i=1}^{N} y_i \, log(\hat{y}_i) \tag{1.9}$$

Prolaz u nazad kreće jednom kada je izračunata vrijednost funkcije troška. Pravilom lanca računa se koliko je svaki od parametara pridonio ukupnoj grešci. Prvo se računa doprinos zadnjeg sloja, pa pomoću njih predzadnjeg i tako sve do početka. Doprinos svakog parametra ukupnoj grešci upravo su traženi gradijenti.

1.6. Prijenos učenja

Treniranje dubokih neuronskih mreža veoma je računski intenzivno. Neizbježno je korištenje grafičkih kartica koje mogu paralelno obaviti veliki broj matematičkih operacija po sekundi. Srećom treniranje se može još i dodatno paralelizirati, pa treniranje možemo podijeliti i na više grafičkih kartica. Međutim, veći modeli i dalje se treniraju se na razini magnitude danima. Zbog toga su se razvile mnoge metode koje taj proces dodatno ubrzavaju. Jedna od takvih metoda je i prijenos učenja.

Prijenos učenja je metoda u strojnom učenju s kojom znanje naučeno na jednom problemu primjenjujemo na neki drugi, ali sličan problem. Kod modela računalnog vida donji slojevi u mreži prepoznaju jednostavne značajke sa slika kao što su linije, rubovi, geometrijskih oblici. Kako novi model ne bi opet morao učiti iste značajke, ideja je kopirati već istrenirane slojeve u implementaciji novog modela. Npr. donji slojevi u modelu za prepoznavanja modela automobila mogu se iskoristiti za model koji prepoznaje modele kamiona.

Što su problemi sličniji to je moguće kopirati više slojeva. Onda se na te slojeve, prema karakteristikama problema, dodaje još slojeva i glava modela koja odgovara problemu. Poželjno je tako dobiven model trenirati u dva navrata. Prvi puta na način da su preneseni slojevi zamrznuti tj. njihovi parametri se ne ažuriraju nego se treniraju samo novo dodani slojevi. Razlog tomu je što je pogreška modela velika zbog novo dodanih slojeva. Pretpostavka je da preneseni slojevi dobro rade svoj posao, pa se pogrešku ne želi propagirati

kroz njih. Tijekom drugog treniranja odmrzava se dio prenesenih slojeva i treniranje se provodi s vrlo malom stopom učenja kako bi se i ti slojevi malo prilagodili za novi problem.

1.7. Augmentacija podataka

Dobri podaci su ključni za dobre performanse sustava. To znači da skup podataka za treniranje treba imati dovoljno različitih primjera da neuronska mreža nauči generalizirati podatke. Npr. kada bi trenirali sustav za prepoznavanje ruke htjeli bi smo skup slika gdje ima puno varijacije u položaju ruke, ali također i puno varijacije u pozadini, osvjetljenju, veličini itd. Međutim, u praski se često nalazimo u situacijama gdje imamo manji skup podataka nego što nam treba ili primjeri u nekom skupu podataka međusobno nisu dovoljno različiti. Taj problem ublažava metoda koja se zove augmentacija podataka.

Augmentacija podataka je metoda generiranja novih podataka iz starih podataka. U statistici takve su metode poznate kao bootstrap metode. Ime su dobile po tome da generiranje novih podataka iz starih zapravo ne stvara nikakvu novu informaciju. Kao da sami sebe povučemo za čizme (bootstrap je naziv za remen koji je zašiven sa strane čizme kao pomoć prilikom oblačenja). Međutim, prijašnji eksperimenti pokazuju da bootstrap tehnike značajno poboljšavaju performanse sustava. U ovom radu korištene su četiri vrste augmentacije podataka.

Prva je nasumično preokretanje slike (engl. *random flip*). Ona prilikom treniranja nasumično zrcali sliku horizontalno ili vertikalno. Druga je nasumična rotacija (engl. *random rotation)* koja nasumično rotira sliku. Zatim, nasumična translacija (engl. *random translation*) koja translatira sliku. Konačno, korišten je nasumičan kontrast (engl. *random contrast*) koji prilikom treniranja nasumice mijenja kontrast slike.

2. Moderne arhitekture

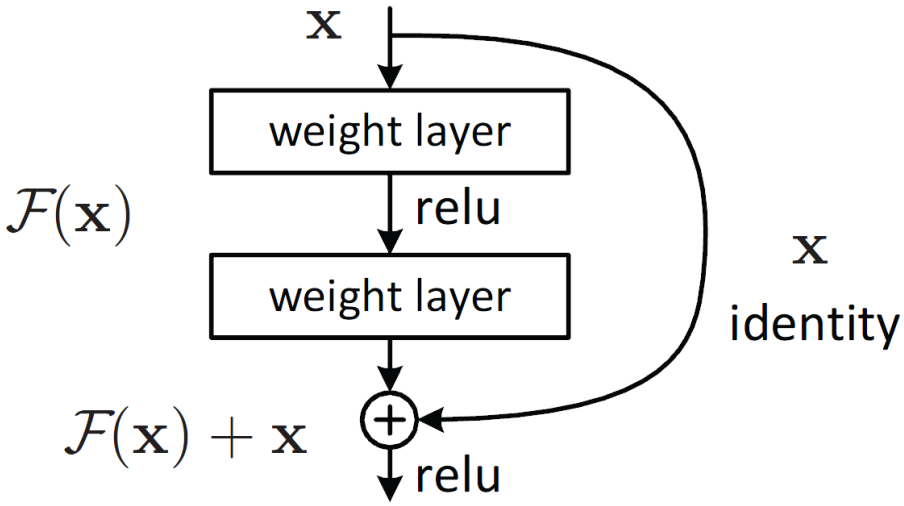
Tipična konvolucijska neuronska mreža prvo sadrži konvolucijski sloj kojeg prati sloj sažimanja. Nakon toga opet jedan konvlucijski sloj kojeg prati sljedeći sloj sažimanja. Ta dva sloja nižemo jedan za drugim sve dok mreža ne bude odgovarajuće dubine za problem. Tako slike postaju sve manje i manje kako napreduju kroz mrežu, ali uobičajeno su sve šire i šire tj. imaju sve više i više kanala. Nakon konvolucijskih slojeva i slojeva sažimanja dolazi nekoliko potpuno povezanih sloja, a na kraju se nalazi „softmax“ sloj.

Kroz godine razvile su se brojne varijacije na ovu temeljnu arhitekturu. Glavni trend je da najbolje arhitekture imaju sve manji broj parametara, ali su sve dublje. Najbolji pokazatelj napretka je natjecanje na velikom skupu podataka zvanog „ImageNet“. Stopa pogrešaka u najviših pet predikcija (engl. *top 5 error rate*) je 2011. godine bila 26%, godine 2016 4%, a trenutno u 2022. godini je manja od 1% [1]. Stopa pogrešaka u najviših pet predikcija je omjer broja primjera gdje model nije predvidio stvarnu istinu u svojih najboljih 5 odgovora naspram ukupnog broja primjera. U nastavku su opisane danas popularne arhitekture.

2.1. ResNet

Kako su mreže postajale sve dublje pojavio se problem umirućih ili eksplodirajućih gradijenata. Naime, algoritam propagacije u nazad računa gradijente sloj po sloj, od kraja pa do početka mreže. Na tom putu može se dogoditi da gradijenti polako nestaju ili suprotno da eksplodiraju.

Kao djelomično rješenje tog problema osmišljen je rezidualni blok koji ima preskočne veze (Slika 7). Rezidualni blok je niz konvolucijskih slojeva, ali s jednom dodatnom vezom koja povezuje ulaz i izlaz iz bloka. Vrijednosti ulaza u blok se zbrajaju s vrijednostima izlaza iz bloka. Zbog toga greška može bolje propagirati kroz model. Dodatna je prednost preskočne veze da blok, a time i sama mreža, jednostavno može naučiti funkciju identiteta [7].

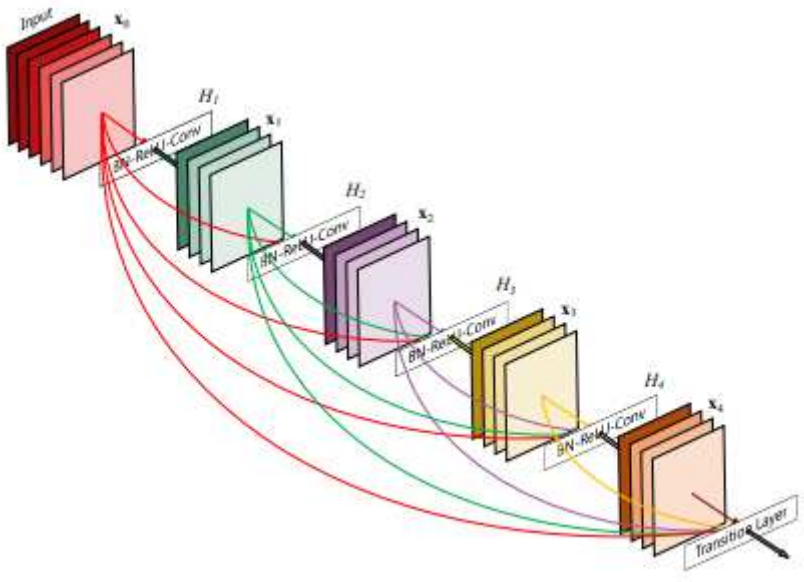


Slika 7: Rezidualni blok.[5]

Preskočne veze također mogu imati konvolucijske slojeve, a to je i neizbježno ako se ulaz i izlaz ne mogu direktno zbrojiti. U tim slučajevima će biti jedan konvolucijski sloj u rezidualnoj vezi koji će napraviti potrebnu korekciju dimenzija.

2.2. DenseNet

Blokovi koji imaju više paralelnih rezidualnih veza nazivaju se gusti blokovi (Slika 8). U takvom bloku svaki sloj kao ulaz prima izlaze svih slojeva prije njega [4].



Slika 8: Gusti blok.[6]

[5] Preuzeto s https://paperswithcode.com/method/residual-block
[6] Preuzeto iz [4]

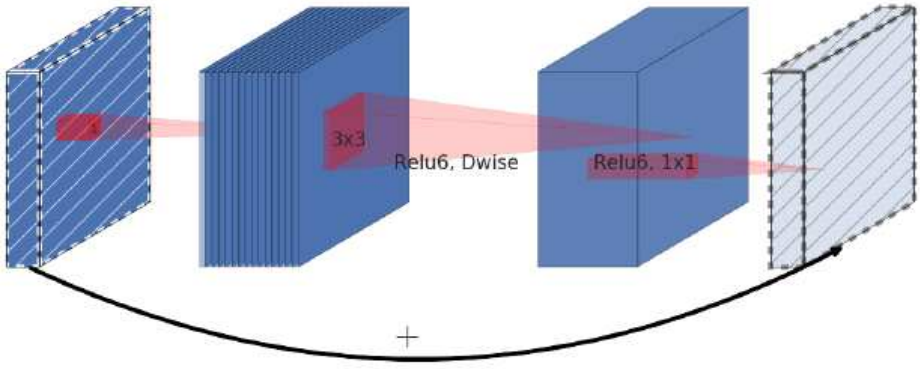
Osim što dodatne veze olakšavaju probleme s gradijentima, one omogućuju da se značajke bolje prenose kroz mrežu. Rekli smo kako svaki sljedeći konvolucijski sloj koristi značajke iz prethodnog sloja kako bi on prepoznao neke kompleksnije značajke. U gustom bloku svaki sloj ima dostupne izlaze iz svih prijašnjih slojeva, pa može kombinirati značajke različite kompleksnosti (npr. auto se može gledati kako suma karoserije i kotača).

2.3. EfficientNet

Nova arhitektura koja pruža najbolje rezultate zove se „EfficientNet“. To je zapravo familija modela koji su dobiveni iz baznog modela novo predloženom metodom skaliranja mreže. Svaki sljedeći model u nizu napravljen je tako da koristi duplo više računalnih resursa nego prethodni. U ovom su radu opisana dva koraka: izvod i opis baznog modela te metoda skaliranja mreže [2].

2.3.1. Bazni model

Osnovni gradivni blok modela je sloj zvan „mobilno invertirano usko grlo“ (engl. *mobile inverted bottleneck MBConv*) (Slika 9) na koji je dodana stisni-i-uzbudi optimizacija (engl. *squeeze-and-excitation optimization*) (Slika 11) [6]. To je konvolucijski blok koji ima malo parametara i izvrsne performanse, pa je cijela mreža po tome dobila ime.

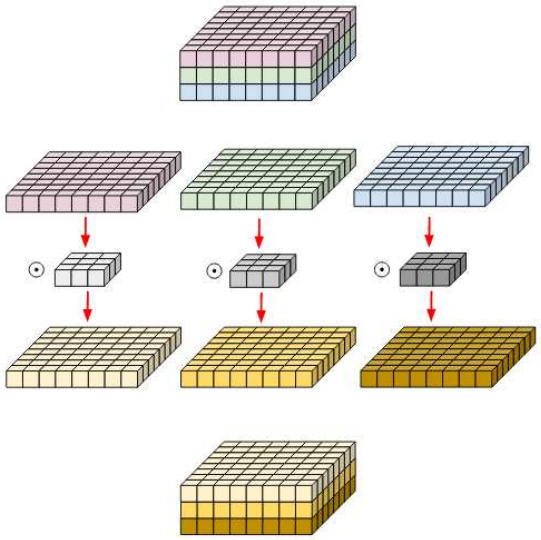


Slika 9: Mobilno invertirano usko grlo (engl. *mobile inverted bottleneck MBConv*).[7]

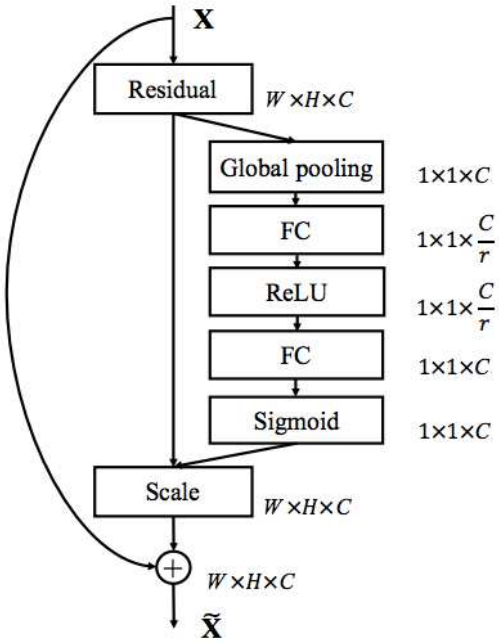
Prvi sloj u tom bloku je normalni konvolucijski blok s filterom veličine 1x1 i proizvoljnim brojem kanala. Ideja je da taj sloj napravi veliki broj mapa, tj. da proširi broj kanala. Nakon njega dolazi konvolucija u dubinu. Za razliku od normalne konvolucije koja za jedan izlazni kanal razmatra sve ulazne kanale, konvolucija u dubinu za jedan izlazni

[7] Preuzeto iz [6]

kanal razmatra samo jedan ulazni kanal. Stoga ima manje parametara i jednak broj izlaznih kanala. (Slika 10)



Slika 10: Dubinska konvlucija.[8]



Slika 11: SE-ResNet modul.[9]

Nakon toga slijedi SE blok koji kanalima pridjeljuje težine (Slika 11). Ideja je dati bloku mogućnost da odabere koji kanali su mu najbitniji umjesto da ih sve tretira jednako. Zatim slijedi normalni konvolucijski sloj s filterom veličine 1x1 koji smanjuje broj kanala. Na kraju se na rezultat dodaju ulazne vrijednosti (preskočna veza).

Napravljena je pretraga mrežne arhitekture (engl. *NAS- neural architecture search*) koja je kao cilj imala optimizirati (2.1):

$$ACC(m)\left(\frac{FLOPS(m)}{T}\right)^{w} \tag{2.1}$$

gdje ACC(m) predstavlja točnost modela m, FLOPS(m) predstavlja broj operacija modela m, T= 400M predstavlja željeni broj operacija, w= -0.07 predstavlja odnos točnosti i broja operacija.

Tako se dobio bazni model EfficientNetB0 (Slika 12).

[8] Preuzeto s https://medium.com/@zurister/depth-wise-convolution-and-depth-wise-separable-convolution-37346565d4ec
[9] Preuzeto iz [6]

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

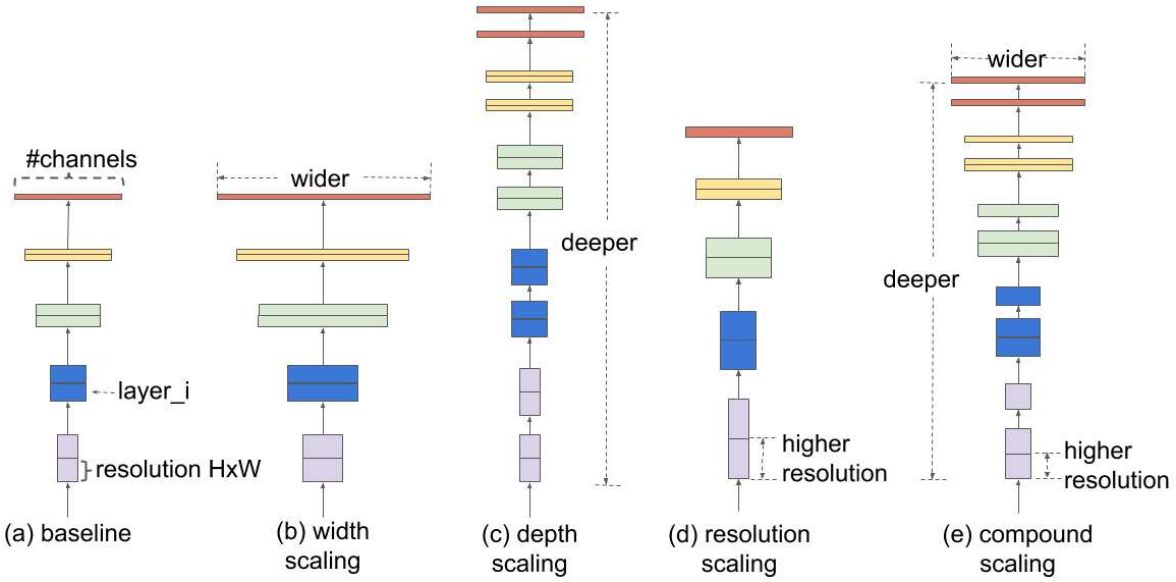
Slika 12: EfficientNetB0 arhitektura.[10]

2.3.2. Metoda skaliranja mreže

Konvolucijske neuronske mreže često se razvijaju za fiksni računalni trošak, pa se zatim skaliraju kada je dostupno više računalnih resursa. Intuitivno se može shvatiti da će modeli s više parametara imati veću točnost, ali u nekom trenutku naša točnost konvergira tj. dodatni računalni troškovi ne daju velike napretke u točnosti. U jednom trenutku doći ćemo do granice onoga što je moguće s trenutnim računalnim resursima. Zato je potrebna što bolja efikasnog parametra i metoda koja će na pametan način skalirati mrežu tako da dodatni resursi rezultiraju najvećim mogućim povećanjem točnosti.

Do sada su se modeli skalirali tako da se povećava dubina ili širina modela. Moguće je skalirati model tako da povećavamo dubinu i širinu u isto vrijeme, ali takav je postupak arbitraran i zahtjeva puno ljudskih i računalnih resursa. Autori ističu [2] da i povećavanje rezolucije slike također jako dobro povećava točnost. Time definiraju 3 dimenzije konvolucijskih neuronskih mreža (Slika 13).

[10] Preuzeto iz [2]



Slika 13: Povećavanje dimenzija mreže. Bazni model (a), povećanje širine (b), povećanje dubine (c), povećanje rezolucije (d), povećanje zajedno (e).[11]

Prva dimenzija je širina modela. Širina je zapravo broj filtera koje ima konvolucijski sloj, a njihovim povećavanjem se povećava broj kanala koje slika ima. Druga dimenzija je dubina modela. Kako bi povećali dubinu potrebno je dodati još slojeva u model i onda je on u mogućnosti naučiti kompleksnije karakteristike. Treća, nova dimenzija je rezolucija slike. Naime, povećavanjem dubine i širine model je sposoban uočiti finije detalje iz slike s većom rezolucijom.

Prije nego što iskažemo metodu skaliranja matematički ćemo definirati problem. Već ranije je rečeno da se neuronske mreže mogu razmatrati kao množenje matrica. Tom analogijom definiramo konvolucijski sloj kao funkciju.

$$Y_i = F_i(X_i) \tag{2.2}$$

Y_i je matrica izlaznih vrijednosti, F_i operator te X_i matrica ulaznih vrijednosti dimenzija $(H_i,\ W_i,\ C_i)$. Ovdje je H_i visina matrica, W_i širina matrica i C_i dubina matrice. Nadalje, kako je neuronska mreža samo niz takvih operacija možemo mrežu definirati kao:

$$N = \odot_{i=1}^{s} F_i^{L_i}\left(X_{(H_i,W_i,C_i)}\right) \tag{2.3}$$

Ovdje je potrebno istaknuti vrijednost L_i koja označava koliko se pojedini sloj puta ponavlja. Takva situacija je uobičajena u konvolucijskim mrežama. Najčešće su podijeljene u nekoliko faza, a u svakoj fazi nalazi se niz istih konvolucijskih slojeva.

[11] Preuzeto iz [2]

Sada metoda skaliranja pokušava pronaći vrijednosti L_i, H_i, W_i, C_i za svaki sloj. Taj prostor pretraživanja je poprilično velik, pa ćemo raditi pod uvjetom da svi slojevi moraju biti skalirani uniformno s konstantnim faktorom, što se može formulirati kao optimizacijski problem:

$$\max_{(\mathrm{d,w,r})} \mathrm{Accuracy}\big(\mathrm{N(d,w,r)}\big) \tag{2.4}$$

pod uvjetima da:

$$N = \bigodot_{i=1}^{s} F_i^{d \cdot L_i}\left(X_{(r \cdot H_i, r \cdot W_i, w \cdot C_i)}\right) \tag{2.5}$$

$$\text{Memorija(N)} \leq \text{ciljana memorija}$$

$$\text{FLOPS(N)} \leq \text{ciljani FLOPS}$$

$F_i,\ L_i,\ H_i,\ W_i,\ C_i$ definirani su u baznom modelu. Vrijednosti w, d, r su koeficijenti za skaliranje širine, dubine odnosno rezolucije mreže. Kao što je već spomenuto, prijašnje metode za skaliranje razmatrale su samo jednu od tih vrijednosti ili arbitrarno dvije od njih. Sljedeća metoda daje jasan postupak kako da mrežu skaliramo u sve tri dimenzije.

Ideja je definirani parametar ϕ koji određuje koliko je računalnih resursa dostupno. Onda su w, d i r definirani na sljedeći način:

$$\text{dubina: } d = \alpha^{\phi}$$

$$\text{širina: } w = \beta^{\phi}$$

$$\text{rezlucija: } r = \gamma^{\phi} \tag{2.6}$$

pod uvjetima da:

$$\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \tag{2.7}$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

gdje su α, β, γ konstante. Uvjet $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$ služi tome da svaka sljedeće mreža u familiji koristi duplo više računalnih resursa. Naime broj računskih operacija (engl. *FLOPs*) regularne konvolucijske operacije proporcionalne su $d \cdot w^2 \cdot r^2$, pa će skaliranje povećati broj operacija za faktor $(\alpha \cdot \beta^2 \cdot \gamma^2)^{\phi} = (2)^{\phi}$.

Familija mreža onda se generira u dva koraka. U prvom koraku fiksira se ϕ = 1 i onda se oko baznog modela napravi pretraga prostora za α, β, γ koje maksimiziraju točnost modela (2.4). U primjeru familije EfficientNet vrijednosti za konstante su α = 1.2, β = 1.1

i $\gamma = 1.15$. Nakon toga se te vrijednosti fiksiraju i onda pomoću različitih vrijednosti ϕ generiramo nove modele prema (2.6).

3. Eksperimenti

Ispitani su popularnih konvolucijski modeli na javno dostupnim skupovima podataka za klasifikaciju slika. Razmatrana je obitelj modela EfficientNet, konkretno EfficientNetB0, EfficientNetB1, EfficientNetB2, EfficientNetB3. Obitelj DenseNet predstavljaju DenseNet121 te DenseNet169. Iz obitelji ResNetV2 uzeti su modeli ResNet101v2 i ResNet152v2. Također treniran je i jedan model iz obitelji MobileNet, a to je MobileNetV2.

Od svakog modela uzeta je kralježnica koja je trenirana na skupu podataka „ImageNet“ i na nju je stavljana glava za klasifikaciju koja ima jedan hiperparametar stopa ispadanja (engl. *dropout rate*). Za taj parametar napravljena je jednostavna pretraga prostora (engl. *grid search*).

Razmatrane su stope 0.2, 0.3, 0.45, 0.55, 0.7. Za svaku stopu model je treniran pet iteracija i odabrana je model koji ima najbolju točnost na validacijskom skupu podataka. Ako su u nekom slučaju točnosti za neke stope bile bliske, polazilo se heuristikom da veći i jači modeli trebaju više regularizacije, a manji modeli manje. Dakle, u slučaju bliskih točnosti davala se prednost većoj stopi ispadanja ako je model velik, a manjoj ako je model mali.

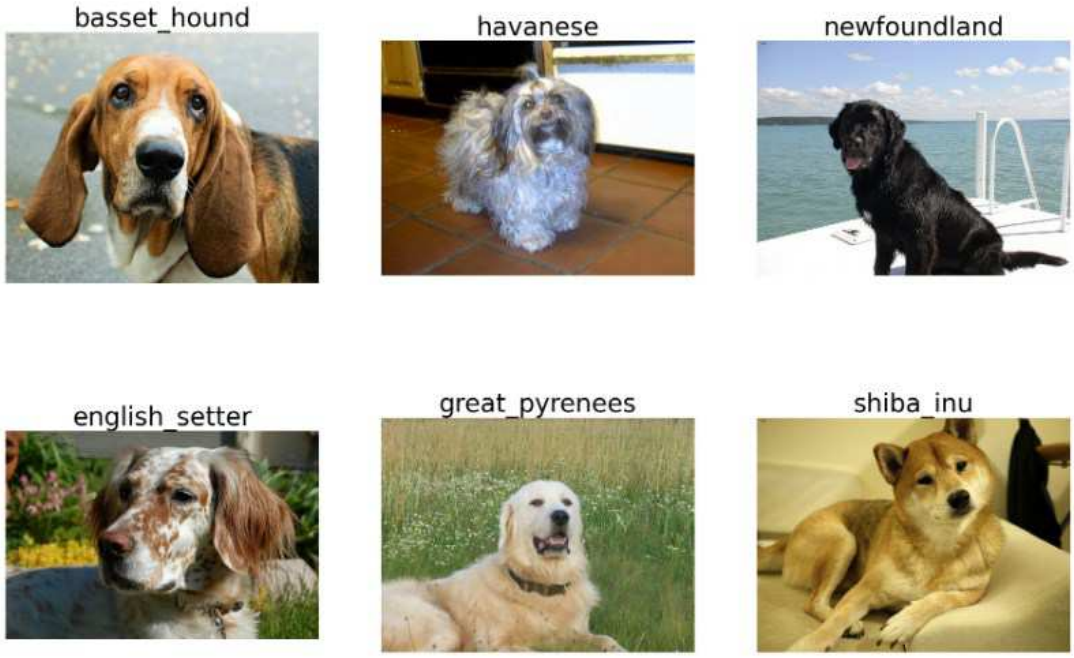
Nakon pretrage hiperprostora modeli su trenirani optimizatorom Adam s već opisanim postupkom prijenosa učenja na augmentiranim skupovima podataka. Tijekom prve iteracije učenja, dok su parametri kralježnice zamrznuti, stopa učenja bila je postavljena na 0.001. Koristilo se rano stajanje (engl. *early stopping*) koje prekida treniranje kada model konvergira tj. u ovom slučaja kada model ne napreduje pet epoha u nizu na validacijskom skupu podataka. U drugoj iteraciji treniranja odmrznuto je zadnjih 20 slojeva kralježnice i stopa učenja je postavljena za magnitudu niže na 0.0001 kako bi se model dodatno naštimao. Treniranje je opet zaustavljeno ranim stajanjem kada je model konvergirao.

Na kraju su svi modeli evaluirani na skupu podataka za testiranje i to točno jednom. Iako je bilo neočekivanih rezultata (npr. slabije performanse ResNetV2 mreža), modeli se nisu opet trenirali kako se ne bi unijela pristranost (engl. *bias*) u analizu. Ne bi imalo smisla da se modeli opet treniraju, sve dok ne dobijemo rezultate koje očekujemo na skupu podataka za testiranje jer onda unosimo pristranost u analizu i nećemo dobiti rezultate koji dobro generaliziraju.

3.1. Skupovi podataka

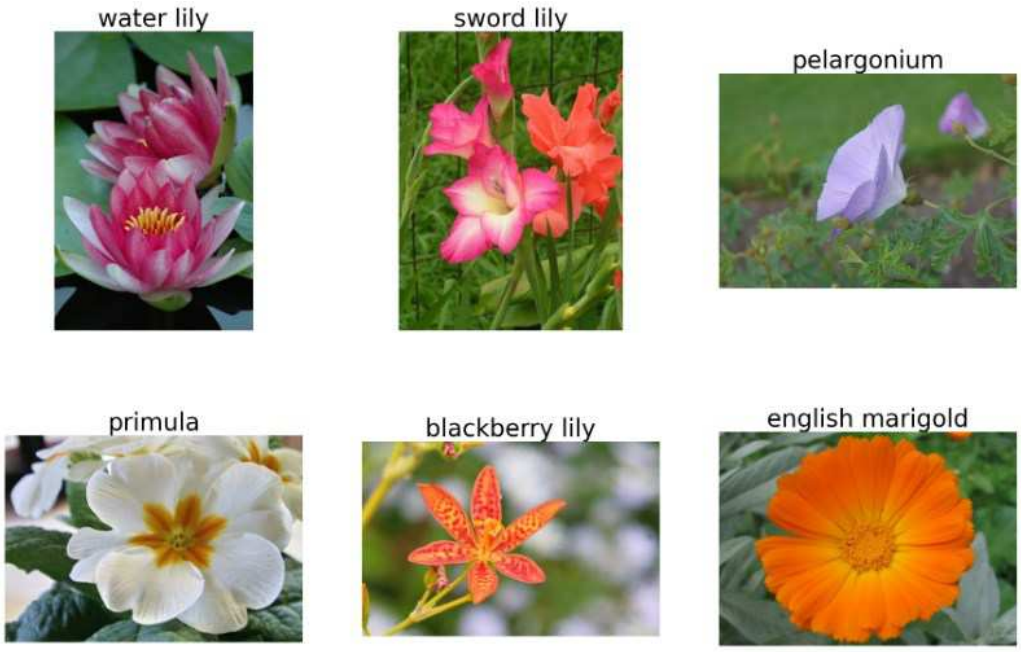
Rad analizira sposobnost prijenosa učenja popularnih konvolucijskih modela sa skupa podataka „ImageNet“ na druge popularne skupove podataka za klasifikaciju slika. Stoga su razmatrani podaci koji pokrivaju što veću domenu objekata tj. skupovi podataka birani su na način da se što više razlikuju. Izabrana su ukupno tri skupa podataka.

„Oxford iiit pets“ skup podataka sadrži slike ljubimaca. Podijeljen je u 37 klasa od kojih svaka ima otprilike 200 slika, pa ukupno sadrži 7349 slika. Jedna polovica slika uzeta je za treniranje, a druga za testiranje s time da je skup za testiranje dodatno podijeljen u omjeru 90:10 kako bi odvojili dio slika za validaciju. Slike imaju veliku varijaciju u osvjetljenu, pozi te veličini objekta na slici (Slika 14).



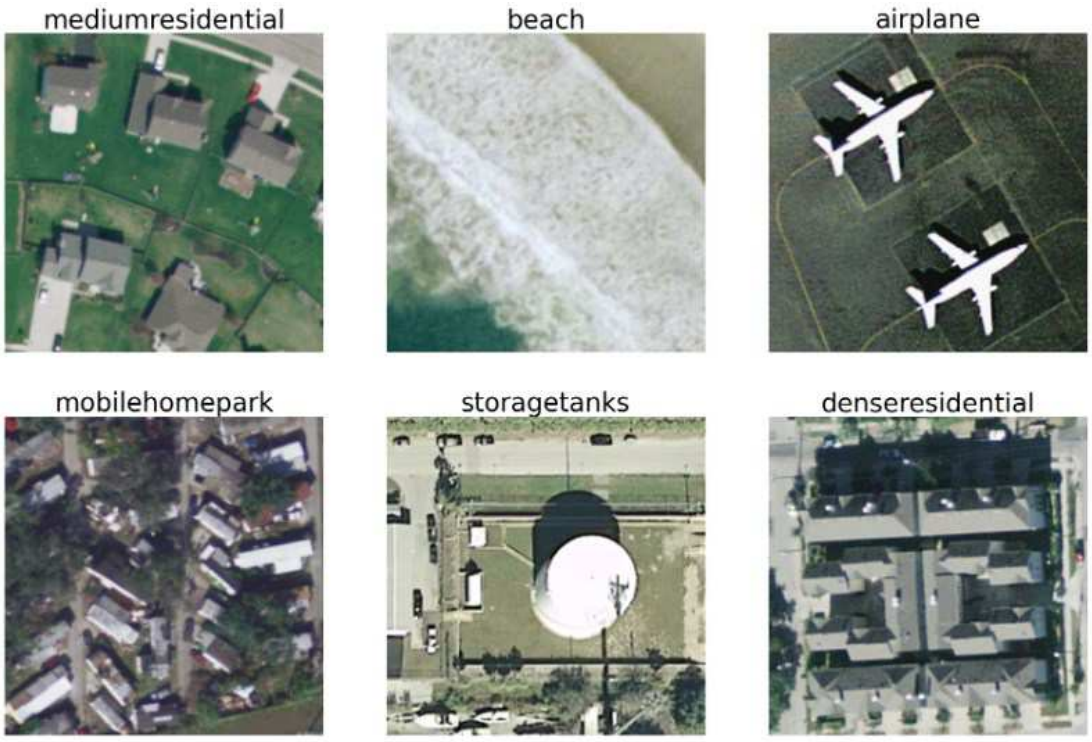
Slika 14: Primjeri slika iz "Oxford iit pets" skupa podataka.

„Flowers102“ skup podataka sadrži slike cvijeća. Podijeljen je u 102 klase od kojih svaka ima minimalno 20 primjeraka. Ukupno sadrži 8189 slika, od kojih je 1020 uzeto za testiranje, 1020 za validaciju i ostatak za testiranje. Slike imaju veliku varijaciju u osvjetljenu, pozi te veličini objekta na slici (Slika 15).



Slika 15: Primjeri slika iz "Flowers102" skupa podataka.

„Uc_merced“ skup podatka sadrži satelitske snimke. Podijeljen je u 21 klasu od kojih svaka ima točno 100 primjeraka, pa ukupno sadrži 2100 slika. Skup je podijeljen u omjeru 70:15:15 za treniranje, validaciju odnosno testiranje. Slike su ručno izabrane iz USGS-ove (Geološka istraživanja sjedinjenih država) baze nacionalnih slika urbanih područja. Veličina im je 256x256 piksela, gdje jedan piksel predstavlja 0.3 metra u prirodi (Slika 16).



Slika 16: Primjeri slika iz "Uc merced" skupa podataka.

3.2. Rezultati

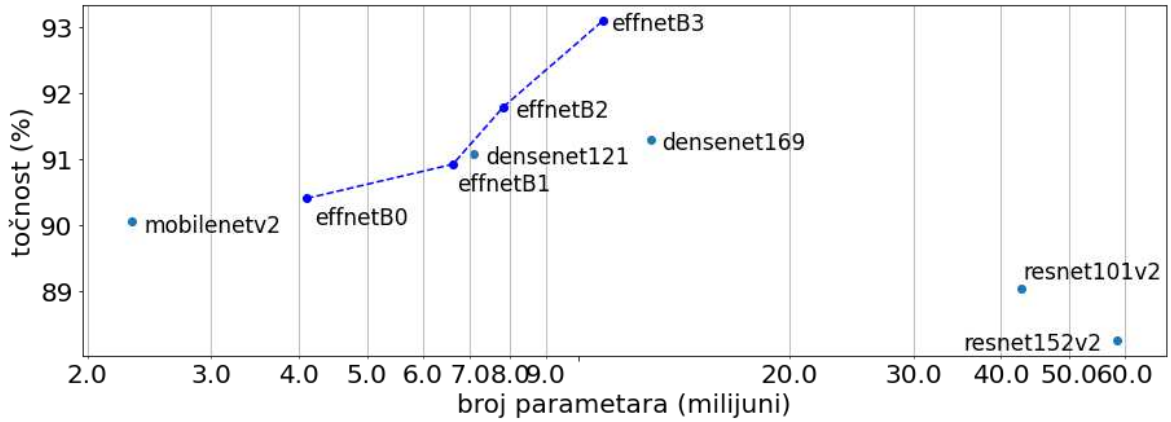
U sljedećoj tablici (Tablica 1) prikazane su točnosti na skupu podataka za testiranje koje su postigli pojedini modeli na pojedinim skupovima podataka.

Tablica 1: Točnosti klasifikacije pojedinih modela na skupovima podataka.

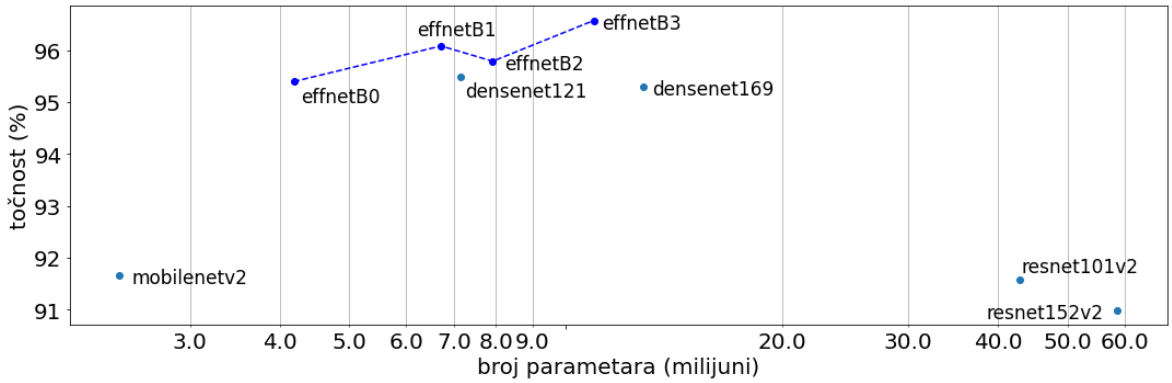
Ime modela	Oxford iiit pets	Flowers102	Uc merced
EfficientNetB0	90.41	95.39	95.56
EfficientNetB1	90.92	96.08	96.19
EfficientNetB2	91.80	95.78	97.46
EfficientNetB3	93.10	96.57	97.78
DenseNet121	91.09	95.49	96.19
DenseNet169	91.31	95.29	96.19
ResNet101v2	89.04	91.57	96.51
ResNet152v2	88.25	90.98	95.56
MobileNetV2	90.05	91.67	93.97

Očekivano, svaki sljedeći model u familiji EfficientNet ima bolju točnost, a konačni model EfficientNetB3 ima najbolju točnost na svim skupovima podataka. Nakon njih po uspješnosti dolaze DenseNet-ovi, a zatim ResNetV2-ovi. ResNetV2-ovi pomalo neočekivano imaju relativno lošu točnost čemu je najvjerojatniji razlog prenaučenost (engl. *overfit*) jer su to modeli s velikim brojem parametara. ResNetV2-ovi bi se mogli ponovno trenirati s boljom regularizacijom i boljim hiperparametrima, ali to nije učinjeno kako se ne bi unijela pristranost (engl. *bias*) u analizu. S druge strane, u praksi bi htjeli modele koji dobro generaliziraju bez prevelike pretrage hiperparametara, pa se u tom smislu ResNetV2-ovi ovdje nisu pokazali toliko dobri.

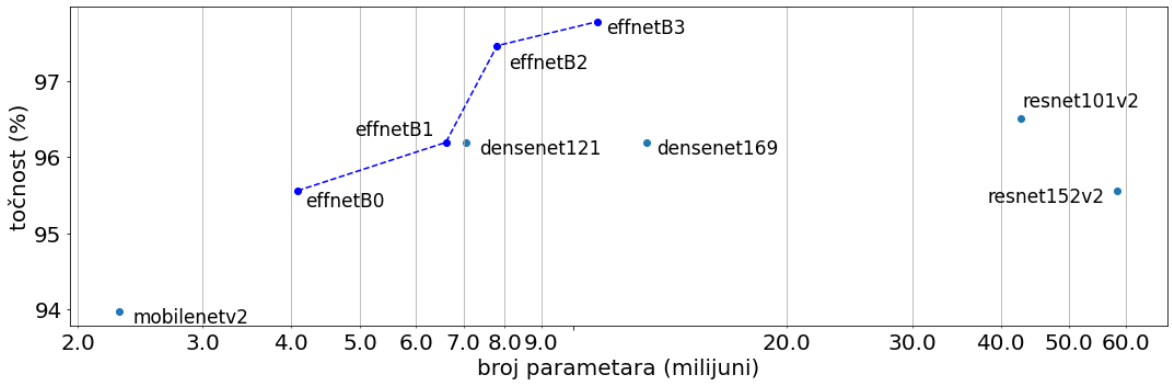
EfficientNet-ovi osim što imaju veliku točnost, to čine vrlo efikasno s relativno malim brojem parametara (Slika 17, 18, 19).



Slika 17: Odnos točnosti i broja parametara na "oxford iit pets" skupu podataka.

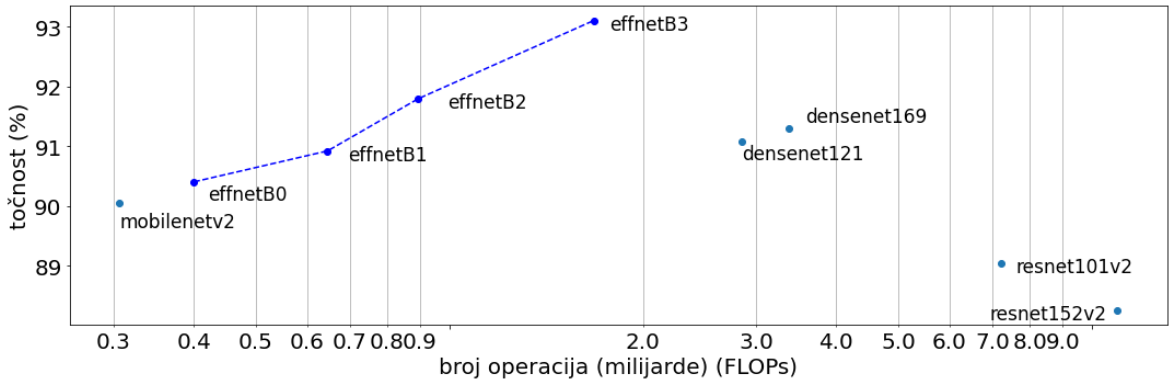


Slika 18: Odnos točnosti i broja parametara na "Flowers102" skupu podataka.

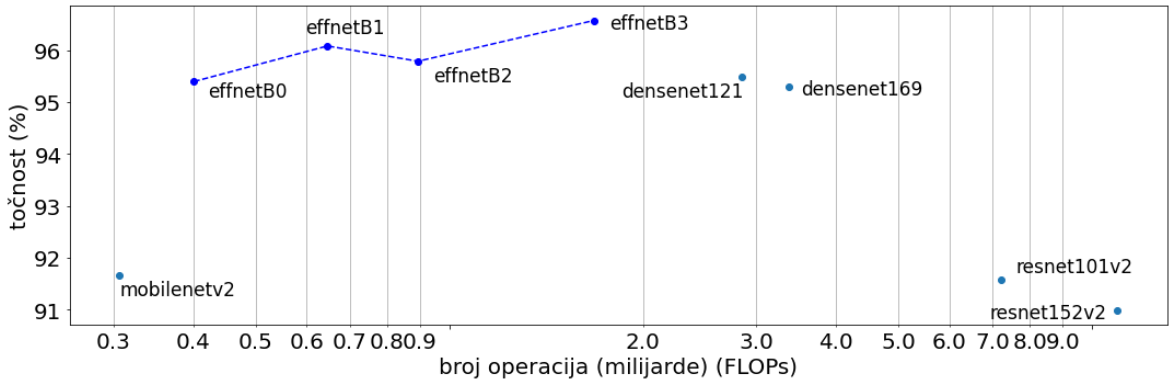


Slika 19: Odnos točnosti i broja parametara na "Uc merced" skupu podataka.

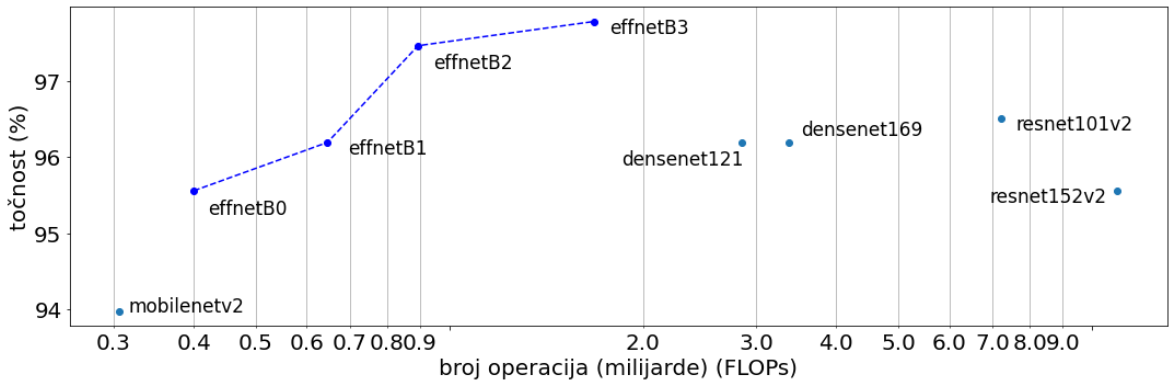
Rezultati još više idu u korist EfficientNet-ova kao pobjednika kada se pogleda graf odnosa točnosti i broja računskih operacija (engl. *FLOPs*). Zahvaljujući vrlo efikasnom bloku s kojim su modeli izgrađeni, oni imaju relativno nizak broj računskih operacija (Slika 20, 21, 22).



Slika 20: Odnos točnosti i broja operacija na "oxford iit pets" skupu podataka.



Slika 21: Odnos točnosti i broja operacija na "Flowers102" skupu podataka.



Slika 22: Odnos točnosti i broja operacija na "Uc merced" skupu podataka.

Konačno možemo redom poredati familije po uspješnosti. EfficientNet-ovi su najbolji, zatim DenseNet-ovi i na kraju ResNetV2-ovi. Dodani su i rezultati modela MobileNetV2-a jer je izgrađen od istih blokova kao i EfficientNet-ovi, ali nema SE blokove. Vidimo da model EfficientNetB0 koji je po veličini sličan modelu MobileNetV2-u ima bolju točnost.

4. Zaključak

Duboki konvolucijski modeli pokazali su se vrlo uspješnima na raznim zadacima računalnog vida, posebno u klasifikaciji slika. Prvotna konvolucijska mreža sastajala se od naizmjenice naslaganih konvolucijskih slojeva i slojeva sažimanja. Nastale su mnoge inačice na tu arhitekturu koji postižu bolje rezultate, a glavi trend je da modeli postaju dublji i da imaju manje parametara.

Možemo odrediti tri dimenzije konvolucijskih modela, a to su dubina, širina i rezolucija. Kako bi dobili snažniji model, kao najbolja metoda pokazala se skaliranje te tri dimenzije istovremeno. Tako se iz baznog modela dobije familija modela EfficientNet. Od ispitanih konvolucijskih modela, ti modeli postigli su najbolje rezultate u smislu točnosti klasifikacije na standardnim skupovima podataka za prijenos učenja, a to su postigli s manje parametara i manje računskih operacija.

Duboke konvolucijske mreže trenutno su vrlo aktivno područje istraživanja. Stoga, ovaj rad će se u budućnosti trebati revidirati. Konstanto se pojavljuju novi konvolucijski modeli koje postižu bolje rezultate od prijašnjih modela. Također, moguće je ispitati modele na nekim drugim skupovima podataka ili nekim drugim problemima računalnog vida.

Literatura

[1] Géron, A.. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.* 2. izdanje. Sebastopol: O'Reilly Media, 2019.

[2] Tan M., Le Q. V. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. ICML, 2019.

[3] *Nueron,* Wikipedia, (31, prosinac). Poveznica: https://hr.wikipedia.org/wiki/Neuron; pristupljeno 24. svi. 22.

[4] Huang, G., Liu, Z., Maaten, L., Weinberger K. Q. *Densely Connected Convolutional Networks.* CVPR, 2017.

[5] Reynolds, A. H., *Convolutional Neural Networks (CNNs),* (2019), Poveznica: https://anhreynolds.com/blogs/cnn.html; pristupljeno 24. svi. 22.

[6] Borad, A. *Image Classification with EfficientNet: Better performance with computational efficiency,* Medium, (2019, prosinac). Poveznica: https://datamonje.medium.com/image-classification-with-efficientnet-better-performance-with-computational-efficiency-f480fdb00ac6; pristupljeno 24. svi. 22.

[7] He, K., Zhang, X., Ren, S., Sun, J. *Deep Residual Learning for Image Recognition.* Tech report, 2015.

Primjena konvolucijskih modela na problem klasifikacije slika

Sažetak

U ovom radu opisane su duboke konvolucijske mreže i njihova primjena na problem klasifikacije slika. Poseban naglasak stavljen je na familiju modela EfficientNet i metodu skaliranja konvolucijskih mreža koja je s njima uvedena. Ispitani su modeli EfficientNet, DenseNet, ResNetV2 i MobileNetV2 na standardnim skupovima podataka za prijenos učenja. Dobiveni rezultati su analizirani u pogledu točnosti klasifikacije te računske zahtjevnosti.

Ključne riječi: računalni vid, duboki konvolucijski modeli, klasifikacija slika, EfficientNet, prijenos učenja

Using convolutional models for image classification

Abstract

This paper describes deep convolutional networks and their application to the image classification problem. Special emphasis was placed on the EfficientNet family of models and the method of scaling convolutional networks introduced with them. EfficientNet, DenseNet, ResNetV2 and MobileNetV2 models were tested on standard transfer learning datasets. The obtained results were analyzed in terms of classification accuracy and computational complexity.

Keywords: computer vision, deep convolutional models, image classification, EfficientNet, transfer learning