

**SVEUČILIŠTE U SPLITU
FAKULTET ELEKTROTEHNIKE, STROJARSTVA I
BRODOGRADNJE**

DIPLOMSKI RAD

**PREGLED AKTIVACIJSKIH FUNKCIJA U
KONVOLUCIJSKIM NEURALNIM
MREŽAMA**

Veronika Ozretić

Split, srpanj 2021.



SVEUČILIŠTE U SPLITU
FAKULTET ELEKTROTEHNIKE, STROJARSTVA I
BRODOGRADNJE



Diplomski studij: **Računarstvo**

Oznaka programa: 250

Akadska godina: 2020./2021.

Ime i prezime: **VERONIKA OZRETIĆ**

Broj indeksa: 661-2018

ZADATAK DIPLOMSKOG RADA

Naslov: Pregled aktivacijskih funkcija u konvolucijskim neuralnim mrežama

Zadatak: Proučiti različite aktivacijske funkcije koje se mogu koristiti u konvolucijskim neuralnim mrežama. Konstruirati i implementirati jednostavnu konvolucijsku neuralnu mrežu za semantičku klasifikaciju digitalnih slika prirodnog krajolika, te analizirati dobivene rezultate segmentacije za različite aktivacijske funkcije.

Prijava rada: 28.09.2020.

Rok za predaju rada: 17.09.2021.

Rad predan:

Predsjednik
Odbora za diplomski rad:

prof. dr. sc. Sven Gotovac

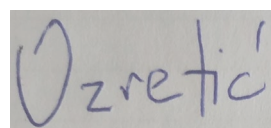
Mentor:

doc. dr. sc. Maja Braović

IZJAVA

Ovom izjavom potvrđujem da sam diplomski rad s naslovom Pregled aktivacijskih funkcija u konvolucijskim neuralnim mrežama pod mentorstvom doc. dr. sc. Maje Braović pisala samostalno, primijenivši znanja i vještine stečene tijekom studiranja na Fakultetu elektrotehnike, strojarstva i brodogradnje, kao i metodologiju znanstveno-istraživačkog rada, te uz korištenje literature koja je navedena u radu. Spoznaje, stavove, zaključke, teorije i zakonitosti drugih autora koje sam izravno ili parafrazirajući navela u diplomskom radu citirala sam i povezala s korištenim bibliografskim jedinicama.

Studentica



Veronika Ozretić

Sadržaj

UVOD.....	1
1. UVOD U KONVOLUCIJSKE NEURONSKE MREŽE.....	3
1.1. ŠTO JE DUBOKO UČENJE.....	3
1.2. UVOD U UMJETNE NEURONSKE MREŽE.....	3
1.2.1. SLOJEVI U NEURONSKOJ MREŽI	4
1.2.2. VEZE IZMEĐU SLOJEVA.....	7
1.2.3. PROSLJEĐIVANJE PREMA NAPRIJED	9
1.3. KONVOLUCIJSKE NEURONSKE MREŽE	10
2. AKTIVACIJSKE FUNKCIJE U UMJETNOJ NEURONSKOJ MREŽI.....	14
2.1. SIGMOIDALNA AKTIVACIJSKA FUNKCIJA	15
2.1.1. KORIŠTENJE SIGMOIDALNE AKTIVACIJSKE FUNKCIJE	16
2.1.2. USPOREDBA SIGMOIDALNE I ReLU FUNKCIJE	17
2.2. HIPERBOLIČKO-TANGENTNA AKTIVACIJSKA FUNKCIJA	17
2.3. ReLU AKTIVACIJSKA FUNKCIJA	18
2.3.1. DOKAZ DA JE ReLU NE-LINEARNA FUNKCIJA.....	20
2.3.2. UMIRUĆI ReLU.....	20
2.4. SWISH AKTIVACIJSKA FUNKCIJA	21
2.4.1. SVOJSTVA SWISH FUNKCIJE.....	22
2.5. SOFTMAX AKTIVACIJSKA FUNKCIJA.....	23
3. SEMANTIČKA SEGMENTACIJA.....	25
3.1. SEMANTIČKA SEGMENTACIJA TEMELJENA NA POTPUNO KONVOLUCIJSKIM MREŽAMA 26	
3.2. U-NET ARHITEKTURA POTPUNO KONVOLUCIJSKE MREŽE.....	26
3.3. PREDPROCESIRANJE I AUGMENTACIJA PODATAKA ZA SEMANTIČKU SEGMENTACIJU....	28
4. KERAS.....	31
4.1. SLOJEVI I MODELI U KERASU	31
4.1.1. SeparableConv2D sloj.....	32
4.1.2. BatchNormalization sloj	33
4.1.3. Activation sloj.....	33
4.1.4. MaxPooling2D sloj.....	33
4.1.5. Conv2DTranspose sloj	34
4.1.6. Dropout sloj.....	35
4.2. KOMPILIRANJE I UČENJE MODELA.....	35
5. USPOREDBA AKTIVACIJSKIH FUNKCIJA	37
5.1. ReLU AKTIVACIJSKA FUNKCIJA	37
5.2. SIGMOIDALNA AKTIVACIJSKA FUNKCIJA	39
5.3. TANGENTNO-HIPERBOLIČNA AKTIVACIJSKA FUNKCIJA.....	41
5.4. SOFTMAX AKTIVACIJSKA FUNKCIJA.....	43
5.5. SWISH AKTIVACIJSKA FUNKCIJA	45
5.6. PARALELNE USPOREDBE AKTIVACIJSKIH FUNKCIJA	46
5.6.1. USPOREDBA GUBITAKA.....	46

5.6.2.	USPOREDBA TOČNOSTI	50
5.6.3.	USPOREDBA VREMENA	54
5.6.4.	VIZUALIZACIJA PREDVIĐANJA	61
6.	ZAKLJUČAK.....	66
7.	LITERATURA I REFERENCE.....	68
8.	POPIS OZNAKA I KRATICE	73
	SAŽETAK	74
	SUMMARY	76

UVOD

Ideja iza pregleda aktivacijskih funkcija u konvolucijskim neuronskim mrežama jest prvo proučiti različite aktivacijske funkcije koje se mogu koristiti u konvolucijskim neuronskim mrežama. Nakon toga je ideja konstruirati i implementirati jednostavnu konvolucijsku neuronsku mrežu za semantičku klasifikaciju digitalnih slika prirodnog krajolika kako bi se moglo analizirati dobivene rezultate segmentacije za različite aktivacijske funkcije.

Kako bi došlo do samog analiziranja rezultata, prvo će se dati kratki uvod u konvolucijske neuronske mreže, njihovu strukturu i način rada kako bi se stekao uvid u to gdje zapravo u konvolucijskim mrežama stoje aktivacijske funkcije i koja je njihova uloga.

Nakon toga će se dati kratki matematički pregled svake pojedine aktivacijske funkcije koja se koristila prilikom izrade ovog rada. Razlog zašto je dan ovaj pregled jest da bi se stekao bolji uvid u sam način rada aktivacijskih funkcija i njihove važnosti tijekom učenja i poslije samog korištenja neuronske mreže. Uz to su ukratko opisane njihove prednosti i nedostaci u odnosu na druge aktivacijske funkcije, gdje se najčešće koriste te u kakvom su mogućem odnosu s ostalim aktivacijskim funkcijama koje su se koristile tijekom izrade ovog rada.

Poslije kratkog pregleda aktivacijskih funkcija, dat će se kratki pregled semantičke segmentacije kao prirodnog koraka iz grubog u fino zaključivanje. U sklopu pregleda semantičke segmentacije uključen je pregled U-net arhitekture kojom je ostvarena implementacija mreže koja se koristi u ovom radu. Na kraju upoznavanja sa semantičkom segmentacijom opisat će se podaci kao jedan od najvažnijih dijelova strojnog učenja te će se nabrojati skupovi podataka koji se najčešće koriste za potrebe treniranja modela semantičke segmentacije.

S obzirom da se ovaj rad sastoji i od praktičnog dijela gdje su rezultati aktivacijskih funkcija analizirani preko konstruiranja, implementacije i samog pokretanja jednostavne konvolucijske neuronske mreže, umetnut je i kratki pregled Keras biblioteke te njenih funkcija i parametara funkcija koji su se koristili u ovom radu.

Konačno, analizirat će se rezultati izvršavanja konvolucijske neuronske mreže s različitim aktivacijskim funkcijama. Aktivacijske funkcije će biti analizirane na temelju nekoliko parametara: vrijednosti gubitka, vrijednosti validacijskog gubitka, točnosti, validacijske točnosti te vremena izvršavanja i parametra sekundi/koraku. Radi bolje preglednosti, prvo će biti analizirane aktivacijske funkcije svaka pojedinačno te će na kraju biti paralelno uspoređene. Uz svaku analizu funkcije po pojedinom parametru, bit će prikazano nekoliko grafova kako bi se mogla bolje predočiti razlika između aktivacijskih funkcija.

1. UVOD U KONVOLUCIJSKE NEURONSKE MREŽE

U ovom poglavlju će se dati općeniti uvid u pojmove i strukture na kojima se temelji ovaj rad. Prvo će se objasniti što je to duboko učenje, što je motivacija dubokog učenja i gdje se ono danas primjenjuje. Nakon toga će se opisati što su to tradicionalne ili standardne umjetne neuronske mreže kako bi se stvorila podloga za upoznavanje s konvolucijskim neuronskim mrežama. Potom će se dati kratki, općeniti pregled rada i strukture umjetnih neuronskih mreža kako bi se opisalo kako točno podaci prolaze kroz umjetnu neuronsku mrežu.

1.1. ŠTO JE DUBOKO UČENJE

Duboko učenje (eng. *deep learning*) je područje strojnog učenja koje 'uči' računala ono što ljudi rade po svojoj prirodi: uče na primjerima [1].

Duboko učenje trenira računalni model kako ispravno klasificirati podatke iz vizualnih, tekstualnih ili audio podataka te u nekim situacijama modeli dubokog učenja pokazuju bolje rezultate od ljudi. Ti modeli su trenirani korištenjem velikog skupa označenih podataka što je značajka nadziranog učenja (eng. *supervised learning*) [1].

Duboko učenje se prvi put spominje tijekom 1980-ih, ali samo u teoretskom obliku. Dva su razloga zašto je tako:

1. Duboko učenje zahtijeva velike količine označenih podataka. Na primjer, razvoj autonomnog vozila zahtijeva milijune slika i tisuće sati videa.
2. Duboko učenje zahtijeva znatnu računalnu moć. Grafičke kartice visokih performansi s mogućnošću paralelnog izvođenja pokazale su se efikasnim u dubokom učenju. Kada se kombiniraju s računarstvom u oblaku (eng. *cloud computing*), znatno se smanjuje vrijeme potrebno za treniranje dubokih neuronskih mreža. Govori se o prvotnom vremenu od nekoliko tjedana do konačnog vremena od nekoliko sati ili manje [2].

1.2. UVOD U UMJETNE NEURONSKE MREŽE

Modeli dubokog učenja nazivaju se 'umjetne neuronske mreže' (eng. *artificial neural networks*). Umjetna neuronska mreža je računalni model inspiriran strukturom i načinom rada biološke

neuronske mreže (kao što je ljudski mozak). Umjetne neuronske mreže se sastoje od velikog broja međusobno povezanih jedinica koje se zovu neuroni, čvorovi ili perceptroni (eng. *neurons, nodes, perceptrons*) [2, 3].

Svaki neuron može donijeti jednostavnu odluku i proslijediti ju neuronima s kojima je povezan preko veze (eng. *connection*). Neuroni su organizirani u slojeve (eng. *layers*). Više međusobno povezanih slojeva tvori umjetnu neuronsku mrežu. Samo neuroni, koji pripadaju različitim susjednim slojevima, mogu biti međusobno povezani [3, 4].

Umjetna neuronska mreža tako može oponašati skoro bilo koju moždanu funkciju i na taj način praktički odgovoriti na bilo koje pitanje nakon što se istrenira na dovoljno velikom broju primjera za treniranje [4].

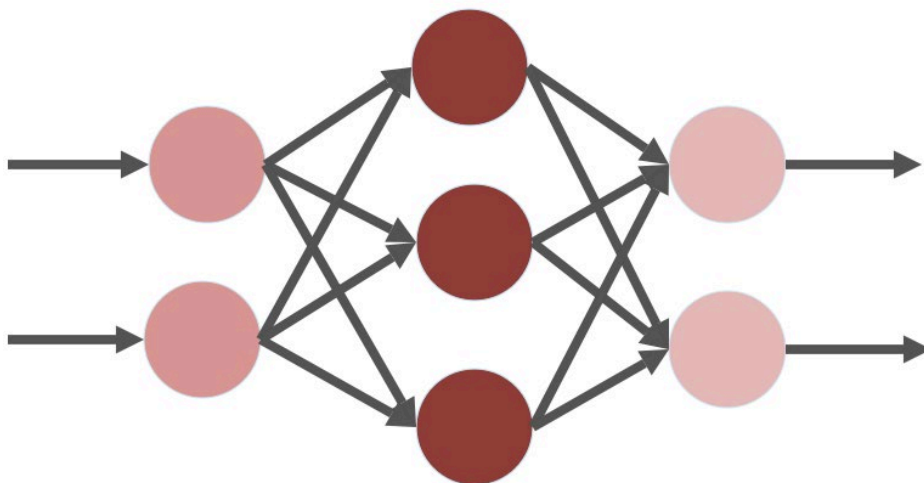
Umjetna neuronska mreža se u literaturi može naći i pod nazivima:

1. mreža (eng. *net*),
2. neuronska mreža (eng. *neural net*),
3. model.

1.2.1. SLOJEVI U NEURONSKOJ MREŽI

'Plitka' umjetna neuronska mreža, kakva je prikazana na slici (Slika 1.1), sadrži samo tri sloja:

1. ulazni sloj (eng. *input layer*) koji prima neovisne varijable ili ulaze u model;
2. jedan skriveni sloj (eng. *hidden layer*) koji se nalazi između ulaznog i izlaznog sloja;
3. izlazni sloj (eng. *output layer*) koji prikazuje predviđanja modela [4].



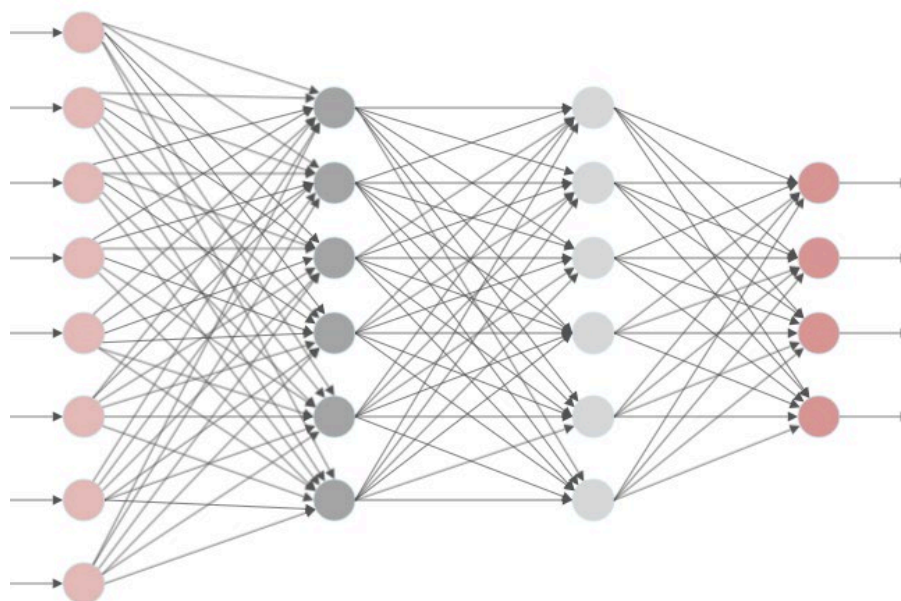
Slika 1.1: Prikaz plitke, potpuno povezane umjetne neuronske mreže.

Različiti slojevi izvode različite vrste transformacija na svojim ulazima koristeći aktivacijske funkcije (eng. *activation function*). Primjeri (podaci) se proslijeđuju mreži preko ulaznog sloja, obično u obliku nekog višedimenzionalnog vektora. Broj čvorova u ulaznom sloju jednak je broju dimenzija ulaznog primjera. Kada bi se kao primjer uzela gore prikazana slika (Slika 1.1) broj dimenzija ulaznog primjera bio bi dva. Ulazni sloj te podatke, takvima kakvi jesu, proslijeđuje skrivenom sloju [1, 3].

Skriveni slojevi donose određene odluke na temelju podataka koje su dobili na svojim ulazima od prethodnog sloja i na temelju aktivacijske funkcije koju koriste. Te odluke proslijeđuju sljedećem sloju kao vlastite izlaze. Proces se ponavlja sve dok se ne dosegne izlazni sloj. Broj čvorova u skrivenom sloju je proizvoljan u smislu da nije uvjetovan strukturom ulaznih primjera [1].

Izlazni sloj izvodi konačnu operaciju kako bi na ljudima razumljiv način prikazao predviđanja modela. Broj čvorova u izlaznom sloju jednak je broju klasa predviđanja (eng. *prediction classes*). Ako se za primjer uzme gore prikazana slika (Slika 1.1), broj klasa predviđanja, odnosno izlaznih čvorova, bi također bio broj dva [1].

Duboka neuronska mreža posjeduje sličnu strukturu. Razlika je u tome što duboka neuronska mreža sadrži dva ili više skrivenih slojeva koji obrađuju ulazne podatke [4]. Primjer takve mreže prikazan je na slici (Slika 1.2).



Slika 1.2: Prikaz strukture potpuno povezane duboke umjetne neuronske mreže.

Iako se i plitke neuronske mreže mogu uspješno nositi sa složenim problemima, duboke neuronske mreže daju ispravnija predviđanja i ta ispravnost raste kako im se pridodaje više skrivenih slojeva. Kod tradicionalnih dubokih neuronskih mreža najčešće je optimalno imati do devet ili deset skrivenih slojeva. Pokazalo se da, dodavanjem više skrivenih slojeva od toga broja, preciznost tradicionalne neuronske mreže počinje opadati. Danas, duboke neuronske mreže posjeduju najčešće između tri i deset skrivenih slojeva [4].

Obje slike (Slika 1.1 i Slika 1.2) prikazuju 'potpuno povezanu neuronsku mrežu' (eng. *fully-connected neural network*) s takozvanim 'gustim slojevima' (eng. *dense layers*). Taj naziv se odnosi na to da je neuron u određenom sloju povezan sa svakim neuronom u susjednom sloju.

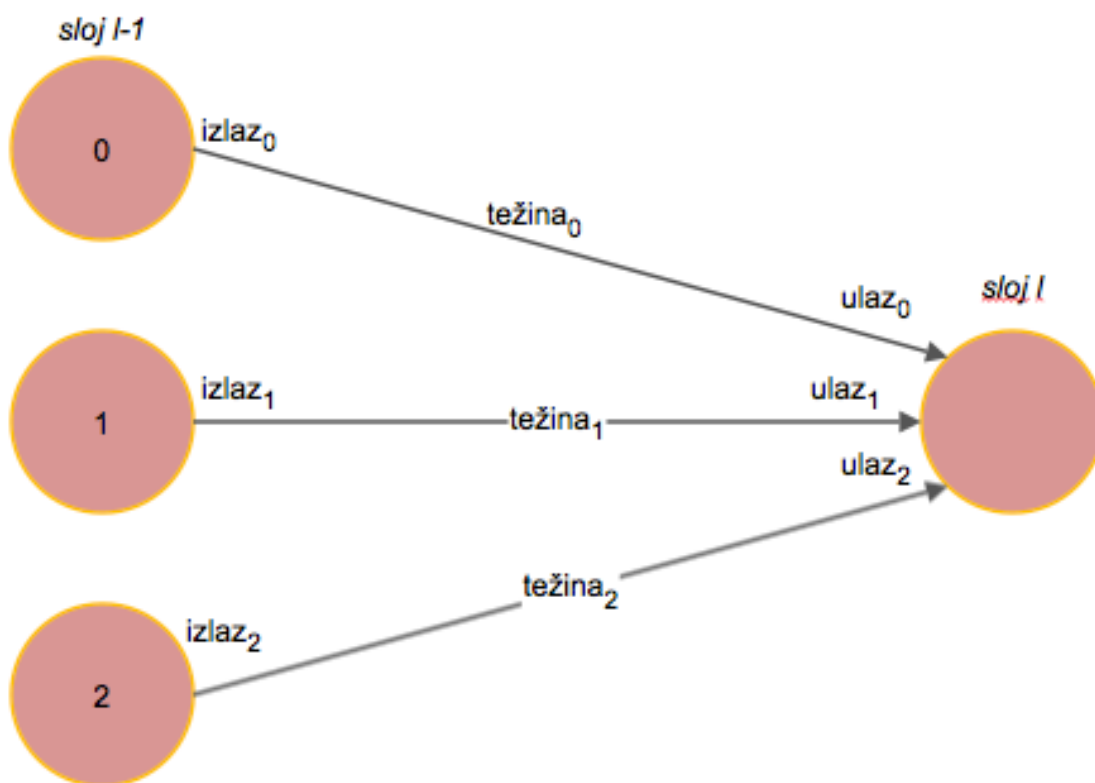
Postoje različite vrste slojeva u umjetnim neuronskim mrežama. Neke od njih su:

1. gusti (ili potpuno povezani) slojevi,
2. konvolucijski slojevi (eng. *convolutional layers*),
3. slojevi sažimanja (eng. *pooling layers*),
4. povratni slojevi (eng. *reccurent layers*),
5. slojevi za normalizaciju (eng. *normalization layers*).

Razlog zbog kojeg postoji više vrsta skrivenih slojeva je taj što različiti slojevi izvode različite transformacije na svojim ulaznim podacima te su stoga određene vrste slojeva prikladnije za pojedini zadatak u odnosu na druge slojeve [3].

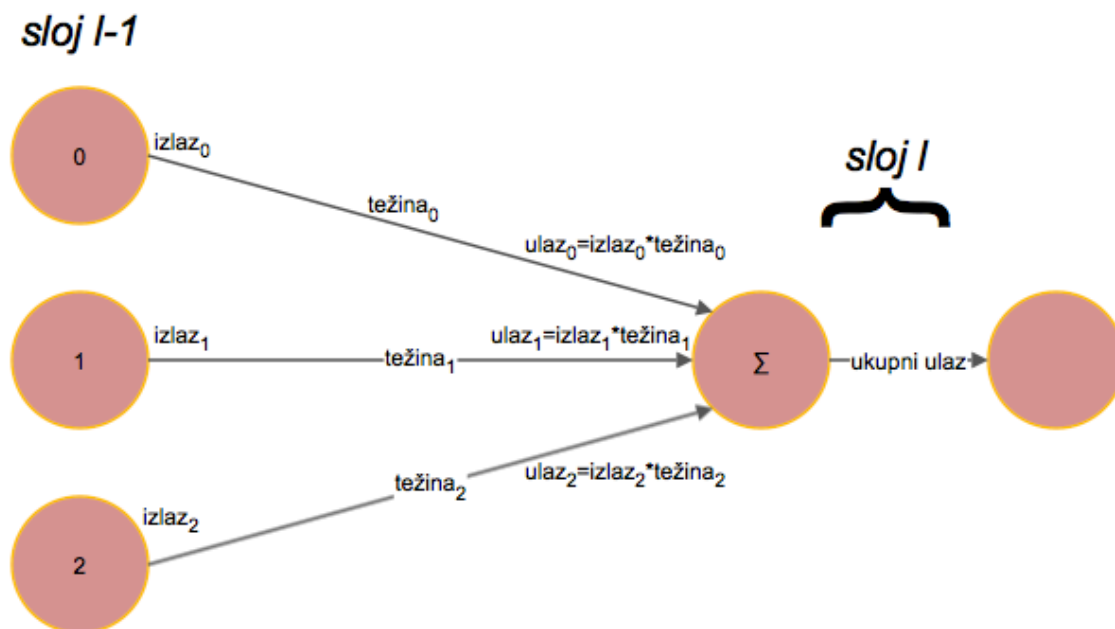
1.2.2. VEZE IZMEĐU SLOJEVA

Svakoj se vezi između dva neurona pridružuje određena težina (eng. *weight*) u brojanom obliku. Težina predstavlja snagu veze između dva određena čvora. Kada mreža na ulazu dobije neku ulaznu vrijednost (eng. *input value*), ona se pridružuje odgovarajućem čvoru u ulaznom sloju. Nadalje, kada se ta ulazna vrijednost prosljeđuje određenim neuronima u sljedećem, ovdje skrivenom, sloju, ona će biti pomnožena s težinom koja je pridružena odgovarajućoj vezi. Ta nova vrijednost, koja se prosljeđuje neuronu u sljedećem sloju, čini ulaznu vrijednost u neuron koji se nalazi u sljedećem sloju (eng. *output value*) [3].



Slika 1.3: Prikaz izlaza i ulaza u neuron

Čvor u skrivenom sloju, na svoj ulaz prima više ulaznih vrijednosti iz više čvorova koji se nalaze u prethodnom sloju. Konačnu će ulaznu vrijednost u određeni čvor činiti ponderirana suma (eng. *weighted sum*) ulaza u dani čvor (Slika 1.4).

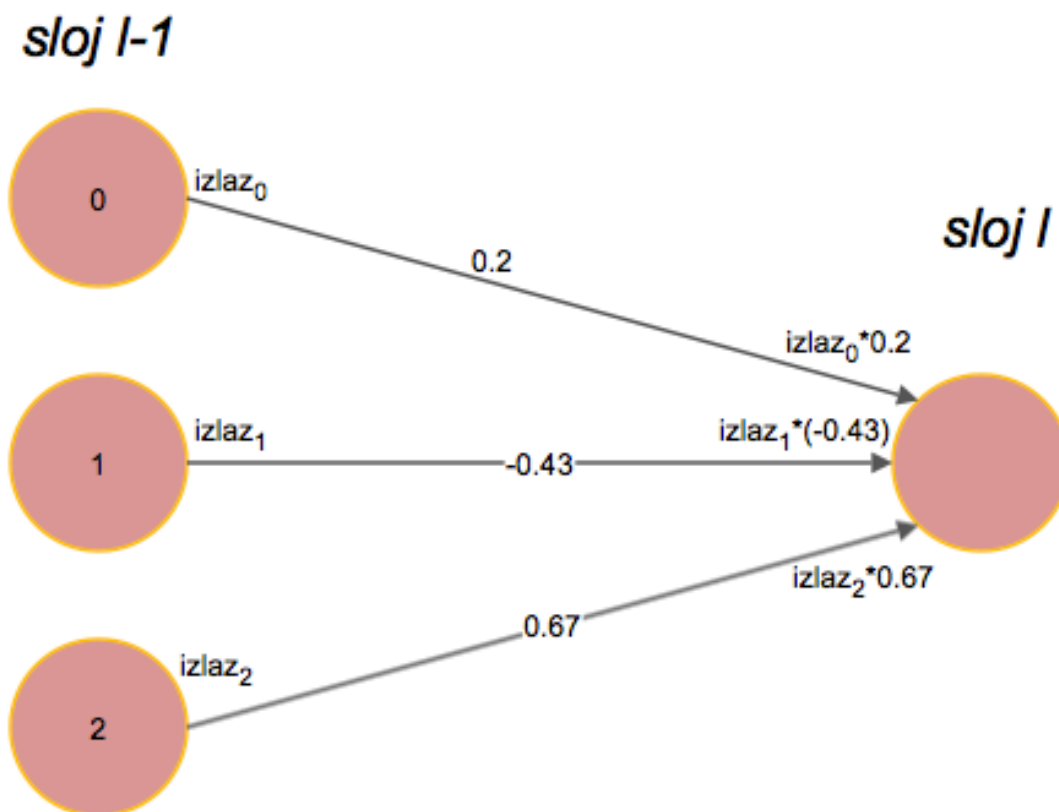


Slika 1.4: Konačni ulaz u određeni čvor.

Prema tome, izlazna vrijednost iz danog čvora može biti izražena na sljedeći način:

$$izlaz_{izčvora} = aktivacijska funkcija(ponderirana suma ulaza) \quad (1.1)$$

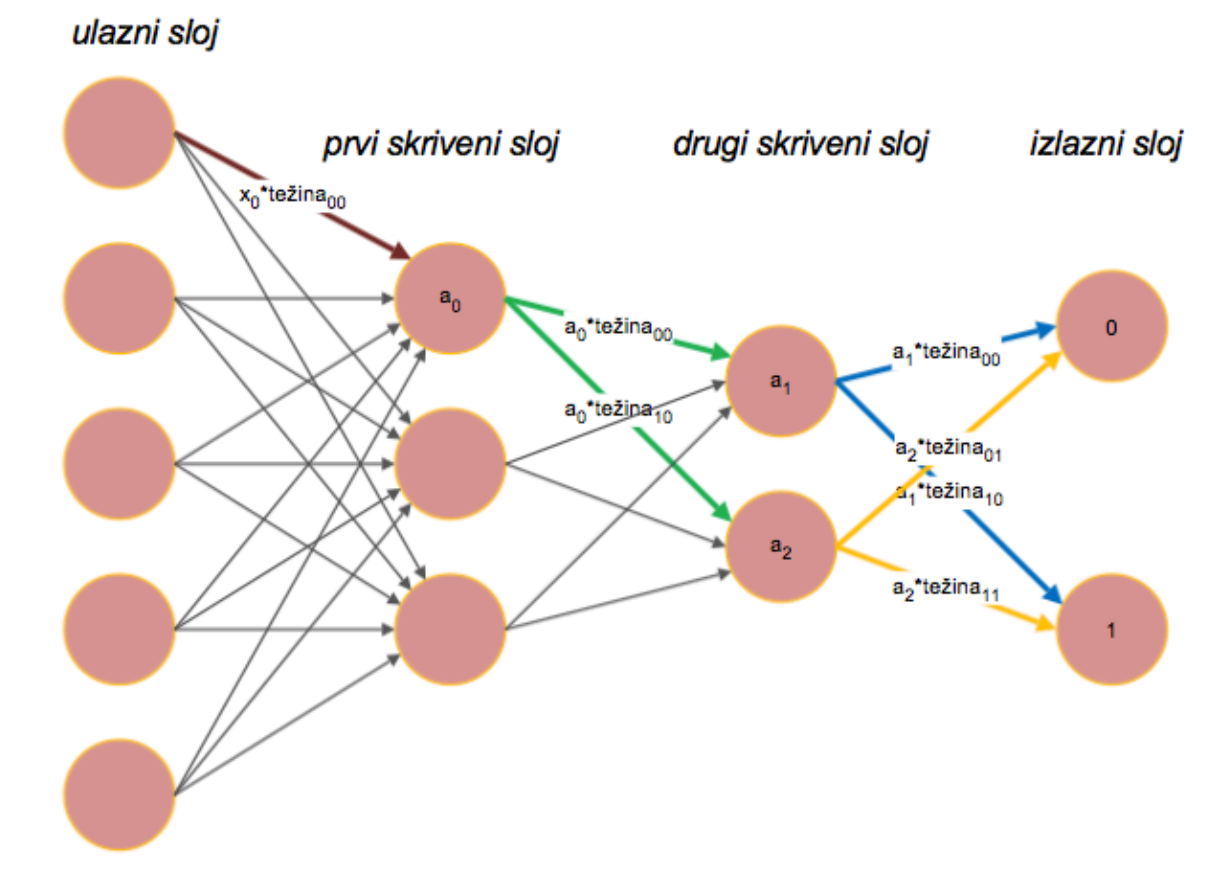
Težine veza su najčešće inicijalizirane ili nulama ili nasumičnim brojevnim vrijednostima i jedan su od glavnih parametara čije optimalne vrijednosti mreža treba naučiti tijekom procesa učenja kako bi što ispravnije preslikavala dani ulazni primjer u ispravnu klasu predviđanja. U primjeru danom na slici (Slika 1.5) veze između dva sloja su inicijalizirane nasumičnim vrijednostima. Tijekom procesa učenja težine veza će se mijenjati kako bi mreža dala što točnija predviđanja.



Slika 1.5: Primjer jednog od načina inicijalizacije težine veza u mreži gdje su one inicijalizirane nasumičnim brojčanim vrijednostima.

1.2.3. PROSLJEĐIVANJE PREMA NAPRIJED

Nastavljajući primjer iz prethodnog poglavlja, jednom kada se dobije izlazna vrijednost za dani čvor, taj dobiveni izlaz se pojedinačno množi sa svakom težinom veze koja povezuje dani čvor sa čvorovima koji se nalaze u sljedećem sloju. Te nove vrijednosti se prenose kao ulazne vrijednosti odgovarajućim čvorovima. Primjer dan na slici (Slika 1.6) pojednostavljeno prikazuje prosljeđivanje prema naprijed. Ovdje je bitno napomenuti da težina veza ne ostaje ista kroz sve slojeve u mreži već je svaka težina veze između dva neurona zaseban parametar koji se mijenja prema rezultatima algoritma za optimizaciju modela. Na dolje prikazanoj slici (Slika 1.6) težine su označene kao $težina_{00}$ ili $težina_{01}$ ili $težina_{10}$ ili $težina_{11}$. To ne znači da, iako neke težine na slici dijele isti naziv, imaju iste težine veza. Ovaj način imenovanja je odabran kako bi slika bila urednija i čitljivija te pravilo da težine veza ne ostaju iste kroz sve slojeve u mreži i dalje vrijedi.

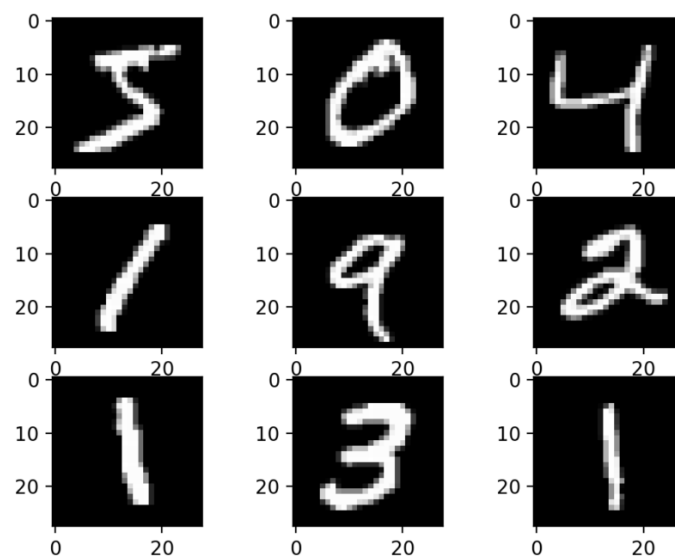


Slika 1.6: Pojednostavljeni prikaz prosljeđivanja prema naprijed.

Taj se opisani proces ponavlja sve dok se ne dosegne izlazni sloj s klasama predviđanja. Ako se za primjer uzme jedna od prije prikazanih slika, može se primijetiti da se opisani proces ponavlja od ulaznog do izlaznog sloja, odnosno s lijeva na desno. Tako se cijeli proces prolaska podataka kroz umjetnu neuronsku mrežu naziva 'prosljeđivanje prema naprijed' (eng. *forward pass*).

1.3. KONVOLUCIJSKE NEURONSKE MREŽE

Jedno od najvećih ograničenja tradicionalnih mreža jest da se počinju 'mučiti' pod računalnom složenošću koju zahtijeva obrada slika. Uobičajeni skupovi podataka za strojno učenje, kao što je MNIST skup podataka [5] (Slika 1.7), koji sadrži slike ručno pisanih znamenki dimenzija 28x28, prikladni su za većinu tipova umjetnih mreža. Neuron bi u prvom skrivenom sloju tradicionalne mreže trebao sadržavati prihvatljiv broj od 784 veze ($28 \times 28 \times 1$) [1].



Slika 1.7: Primjer iz MNIST skupa podataka [6].

Međutim, kada bi tradicionalna neuronska mreža trebala obraditi RGB (eng. *Red Green Blue*) sliku dimenzija 64x64, neuron u prvom skrivenom sloju bi trebao imati 12,288 veza. Tradicionalna mreža u tom slučaju ne bi raspolagala dovoljnom računalnom snagom i bila bi sklona prenaučivosti (eng. *overfitting*) [1].

Drugim riječima, struktura potpuno povezane neuronske mreže pokazala se neefikasnom ondje gdje je potrebno analizirati slike (ili podatke) velikih dimenzija.

Konvolucijska neuronska mreža (eng. *convolutional neural network*) koristi strukture kod kojih neuroni u jednom sloju ne komuniciraju nužno sa svim neuronima u sljedećem sloju. Umjesto toga, svaki skup neurona analizira malo područje ili značajku slike [7].

Pristup korištenja konvolucijskih mreža temelji se na pretpostavci da model može ispravno funkcionirati na temelju lokalnog razumijevanja slike. Konvolucijska mreža koristi manje parametara u odnosu na potpuno povezane mreže tako što ponovno iskorištava parametre i po nekoliko puta (eng. *parameter sharing*). Dok tradicionalna potpuno povezana neuronska mreža generira težinu veze za svaki piksel na slici, konvolucijska neuronska mreža generira tek toliko težina kako bi mogla skenirati malo područje na slici u danom trenutku [7].

Ovaj je pristup koristan tijekom procesa učenja – što mreža posjeduje manje parametara, to će pokazivati bolju izvedbu i brže će učiti [7].

Konvolucijske neuronske mreže primarno su bile korištene kako bi se riješili problemi raspoznavanja uzoraka ili struktura na slikama. One uče kako detektirati različite strukture (eng. *features*) na ulaznim primjerima korištenjem desetaka ili čak stotina skrivenih slojeva. Svaki sloj u konvolucijskoj mreži povećava kompleksnost strukture koju model uči raspoznavati. Na primjer, skriveni sloj na početku konvolucijske mreže uči detektirati rubove i svjetlinu, dok posljednji sloj uči detektirati strukture koje su specifične za objekt kojeg model treba moći znati prepoznati (kao na primjer raspoznavanje lica) [1, 2, 4].

Konvolucijske mreže su po strukturi analogne standardnim umjetnim neuronskim mrežama na način da se sastoje od neurona, veza i slojeva. I dalje će neuron primiti neki ulaz, izvesti određenu operaciju i dobivenu vrijednost proslijediti dalje kroz mrežu [1].

Ulazi u konvolucijski sloj se još mogu nazivati ulaznim kanalima (eng. *input channels*), a izlazi se mogu nazivati izlaznim kanalima (eng. *output channels*) [3].

Međutim, skriveni slojevi konvolucijske neuronske mreže izvode operacije koje transformiraju ulazne podatke s namjerom da model nauči strukture koje su specifične za pojedine ulazne primjere. Tri najčešće korištena sloja su:

1. Konvolucijski sloj: obrađuje ulaznu sliku koristeći skup konvolucijskih filtera. Svaki od filtera aktiviran je od strane određene strukture (ili značajke) na slici.
2. Aktivacijski sloj: aktivira značajke koje se prosljeđuju sljedećem sloju u mreži.
3. Sloj sažimanja (eng. *pooling layer*): pojednostavljuje izlaz tako što izvodi ne-linearno uzorkovanje i tako reducira broj parametara koje mreža treba naučiti, ali u isto vrijeme čuva najbitnije informacije.

Ove operacije se ponavljaju kroz desetke ili stotine slojeva. Svaki sloj uči detektirati različite strukture (ili značajke) [4].

Konvolucijske neuronske mreže su popularne iz tri glavna razloga:

1. eliminiraju potrebu ručnog izvlačenja struktura i značajki – značajke se uče direktno od strane konvolucijske mreže,

2. daju visoku kvalitetu raspoznavanja objekata,
3. mogu se ponovno istrenirati i tako omogućuju razvoj nove mreže na temelju već postojeće [4].

Konvolucijske mreže su osobito korisne kod prepoznavanja objekata, lica i krajolika. Također su izrazito efikasne i kod klasificiranja ne-vizualnih podataka kao što su audio podaci, podaci s vremenskim serijama ili signalima [4, 7].

2. AKTIVACIJSKE FUNKCIJE U UMJETNOJ NEURONSKOJ MREŽI

Aktivnost u biološkoj neuronskoj mreži, gdje su različiti neuroni aktivirani različitim podražajima, bila je inspiracija za korištenje aktivacijskih funkcija u umjetnim neuronskim mrežama. U biološkoj neuronskoj mreži, neki neuroni su u danom trenutku ili aktivirani ili ne. Primjenom aktivacijske funkcije u umjetnoj neuronskoj mreži, aktivnost neurona može biti prikazana bilo kojim brojem između nula (ili minus jedan) i jedan (ili proizvoljno odabrane gornje granice). Što je izlazna vrijednost neurona bliža nuli (ili što je više negativna), to je neuron manje aktiviran. S druge strane, što je izlazna vrijednost neurona bliža jedinici (ili pozitivnija), to je neuron više aktiviran [3, 4, 8, 9].

Aktivacijska funkcija u umjetnoj neuronskoj mreži jest funkcija koja preslikava ulaznu vrijednost, ponderiranu sumu svih ulaza u određeni čvor, u odgovarajuću izlaznu vrijednost.

$$izlaz iz čvora = aktivacijska funkcija(ponderirana suma ulaza) \quad (2.2)$$

Aktivacijska funkcija na unaprijed određeni način transformira ulaznu vrijednost u broj koji se nalazi između unaprijed određene donje i gornje granice. Obično, umjetne neuronske mreže koriste ne-linearne aktivacijske funkcije jer su vrste preslikavanja kod dubokih neuronskih mreža složenije od običnih linearnih preslikavanja i zbog toga što ne-linearne aktivacijske funkcije omogućuju računanje proizvoljno složenih funkcija. Još jedna prednost aktivacijskih funkcija jest ta što aktivacijske funkcije mogu ostvariti propagaciju prema natrag koja se koristi tijekom procesa učenja [8, 9, 10].

Aktivacijske funkcije, koje se koriste u skrivenim slojevima, iste su za sve skrivene slojeve u danoj neuronskoj mreži. Nije uobičajeno vidjeti ReLU aktivacijsku funkciju u jednom skrivenom sloju i sigmoidalnu aktivacijsku funkciju u drugom skrivenom sloju. Koristi se ili samo ReLU aktivacijska funkcija ili samo sigmoidalna aktivacijska funkcija u svim skrivenim slojevima dane neuronske mreže.

Kako bi se razumjelo zašto se koriste ne-linearne aktivacijske funkcije, prvo je potrebno razumjeti što linearne funkcije čini linearnima.

Neka je f funkcija na skupu X .

Neka su a i b elementi skupa X .

Neka je x neki realan broj.

Funkcija f je linearna funkcija ako i samo ako vrijedi:

$$f(a + b) = f(a) + f(b) \quad (2.3)$$

i ako vrijedi:

$$f(xa) = xf(a) \quad (2.4)$$

Važna značajka linearnih funkcija jest da kompozicija dviju linearnih funkcija čini novu linearnu funkciju. To znači da će, čak i u dubokim umjetnim neuronskim mrežama, ako se primjenjuju samo linearne funkcije nad ulaznim vrijednostima kod prosljeđivanja prema naprijed (eng. *forward propagation*), preslikavanje ulaza na izlaz uvijek biti linearno. Nadalje, kada bi svaki sloj u neuronskoj mreži koristio samo pragove (eng. *bias*) i težine (eng. *weights*) bez aktivacijskih funkcija, cijela mreža bi bila jednaka linearnoj kombinaciji težina i pragova. Drugim riječima, formula neuronske mreže bi se mogla faktorizirati i svesti na jednostavan linearni regresijski model. Takav model bi mogao rješavati jednostavne linearne ovisnosti, ali ne bi mogao obavljati konkretne zadatke neuronskih mreža kao što su obrada slika i zvukova [3].

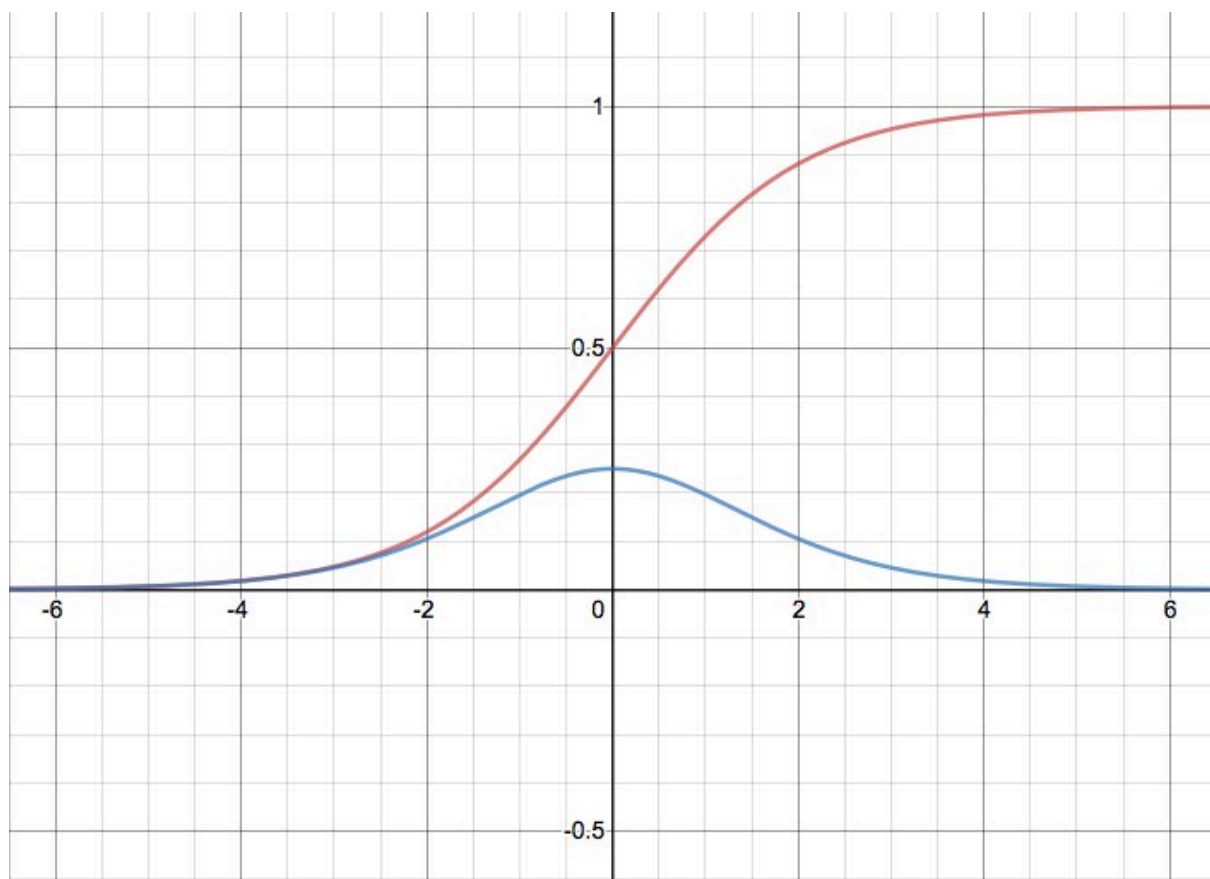
2.1. SIGMOIDALNA AKTIVACIJSKA FUNKCIJA

Sve sigmoidalne funkcije imaju sposobnost preslikavanja ulaznog niza brojeva u izlazni niz brojeva koji se nalaze u malom opsegu između nula i jedan ili minus jedan i jedan. Tako se sigmoidalna funkcija može koristiti za pretvorbu realnog broja u vrijednost koja se potom tumači kao vjerojatnost. Zbog toga se sigmoidalna funkcija [11], osim u skrivenim slojevima, može koristiti i u izlaznom sloju gdje pretvara izlazni rezultat cjelokupne mreže u prikaze vjerojatnosti s kojima se onda može lakše raditi i koje se mogu lakše protumačiti. Zbog svog 'S' oblika na grafu nosi naziv sigmoidalna funkcija. Taj naziv se često odnosi na naziv 'logistička sigmoidalna funkcija' [12, 13].

Matematička formula sigmoidalne funkcije glasi:

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \quad (2.5)$$

Grafički prikaz sigmoidalne funkcije i njene derivacije prikazani su na slici (Slika 2.1).



Slika 2.1: Sigmoidna aktivacijska funkcija (crveno) i njena prva derivacija (plavo).

Sigmoidalna funkcija je ne-linearna, kontinuirano diferencirana, monotona funkcija. Glavna prednost joj je to što je jednostavna i dovoljno dobra kao funkcija za klasifikaciju. S druge strane, veliki nedostatak joj je što uzrokuje problem 'nestajućeg gradijenta' (eng. *vanishing gradient problem*) jer joj vrijednost nije centrirana oko nule. Zbog toga nove vrijednosti težina veza mogu ići predaleko u različitim smjerovima. Nadalje, korištenjem sigmoidne funkcije, teže je provesti optimizaciju te računanje u skrivenim slojevima oduzima puno vremena [12, 13].

2.1.1. KORIŠTENJE SIGMOIDALNE AKTIVACIJSKE FUNKCIJE

Glavno područje strojnog učenja, gdje je bitno korištenje sigmoidalne funkcije, je područje korištenja logističkog regresijskog modela. Model logističke regresije se koristi kako bi se procijenila vjerojatnost na binarnom nivou, kao što je, na primjer, 'živo' ili 'neživo', 'lažno' ili 'istinito' i slično. Model vraća vrijednost koja se nalazi između 0 i 1 [13].

Razlog zašto se baš (logistička) sigmoidalna funkcija koristi u logističkog regresiji je zbog činjenice da funkcija uvijek vraća vrijednost između 0 i 1, a logistička regresija je izvedena iz pretpostavke da su podaci iz obje klase normalno distribuirani [13].

2.1.2. USPOREDBA SIGMOIDALNE I ReLU FUNKCIJE

U modernim umjetnim neuronskim mrežama običaj je vidjeti, umjesto sigmoidalne funkcije, ReLU aktivacijsku funkciju [14].

ReLU aktivacijska funkcija ima nekoliko bitnih prednosti u odnosu na sigmoidalnu funkciju. Glavna prednost jest da se ReLU funkcija puno brže izračunava. Nadalje, aktivacijski potencijal se u biološkim neuronskim mrežama ne mijenja za negativne ulaze. Zbog toga se smatra da ReLU funkcija bolje oponaša biološke neuronske funkcije [13].

Još jedna u nizu prednosti se odnosi na pozitivne vrijednosti x . ReLU funkcija posjeduje konstantni gradijent 1, dok sigmoidalna funkcija posjeduje gradijent koji vrlo brzo konvergira prema 0. Ovo svojstvo neuronsku mrežu, koja sadrži sigmoidalnu aktivacijsku funkciju, čini sporom za učenje [13].

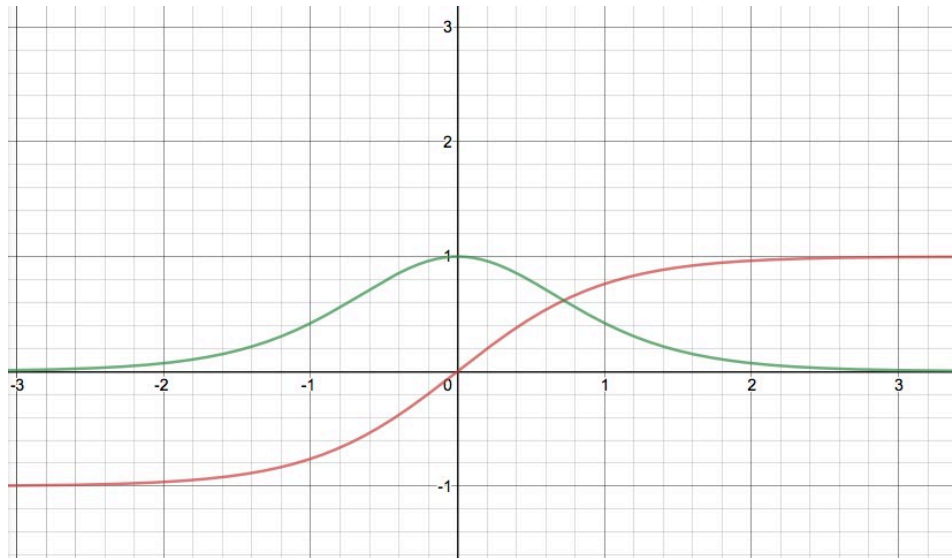
2.2. HIPERBOLIČKO-TANGENTNA AKTIVACIJSKA FUNKCIJA

Hiperboličko-tangentna funkcija (\tanh) [15] je poboljšana sigmoidalna funkcija. Opseg \tanh funkcije je od minus jedan do jedan te, kao i sigmoidalna funkcija, ima 'S' oblik [9, 12].

Matematička formula \tanh funkcije glasi:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2.6)$$

Grafički prikaz \tanh funkcije i njene prve derivacije prikazan je na donjoj slici (Slika 2.2):



Slika 2.2: Grafički prikaz tanh funkcije (crveno) i njene prve derivacije (zeleno).

Prednost tanh nad sigmoidalnom funkcijom je što će negativne vrijednosti biti preslikane u izrazito negativne vrijednosti, a vrijednosti, koje imaju približnu vrijednost nula, bit će preslikane u vrijednosti koje su blizu nule i tako rješava problem sigmoidalne funkcije. Nadalje, pokazalo se da daje bolje rezultate kod više-slojnih neuronskih mreža. Međutim, kao i sigmoidalna funkcija, uzrokuje problem nestajućeg gradijenta (vrijednost gradijenta funkcije gubitka se približava nuli zbog čega mreža jako sporo uči), zbog čega se danas preferira korištenje ReLU funkcije [9, 16].

Ova funkcija se najčešće koristi za klasifikaciju podataka u dvije kategorije.

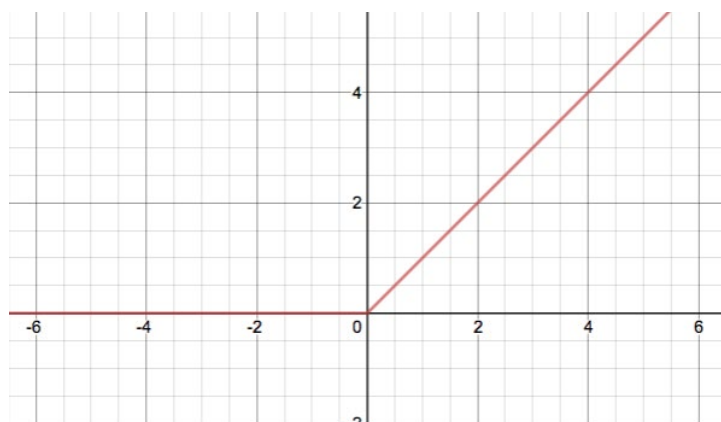
2.3. ReLU AKTIVACIJSKA FUNKCIJA

ReLU aktivacijska funkcija je kratica od *Rectified Linear Unit* i izražava se matematičkom formulom:

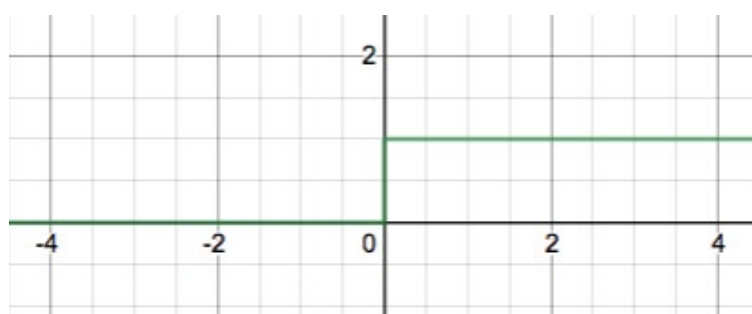
$$f(x) = \begin{cases} x, & \text{ako } x > 0 \\ 0, & \text{ako } x \leq 0 \end{cases} \quad (2.7)$$

To je jedna od najboljih i najčešće korištenih aktivacijskih funkcija koje se danas koriste (pogotovo u konvolucijskim neuronskim mrežama) u odnosu na sigmoidalnu i tanh funkciju jer se uspješno suočava s problemom nestajućeg gradijenta [17].

Grafički prikaz ReLU aktivacijske funkcije i njene derivacije nalaze se na slikama (Slika 2.3 i Slika 2.4)



Slika 2.3: ReLU aktivacijska funkcija.



Slika 2.4: Prva derivacija ReLU aktivacijske funkcije.

Kao što se može vidjeti na slici (Slika 2.3), ReLU aktivacijska funkcija ponaša se kao linearna funkcija za sve pozitivne vrijednosti i vraća nulu za sve negativne vrijednosti. To znači nekoliko stvari:

1. izlazna vrijednost ReLU funkcije se jednostavno izračunava. Zato će modelu trebati manje vremena za učenje i kasnije za donošenje odluka.
2. Brže konvergira. Linearnost ReLU funkcije znači da njen nagib nema 'visoravan' kada ulaz x poprimi veliku vrijednost.
3. Slabo je aktivirana. S obzirom da ReLU poprima vrijednost nula za sve negativne ulazne vrijednosti, postoji vjerojatnost da se neki neuroni u mreži nikada neće aktivirati, što je u većini slučajeva dobro [17].

U biološkim neuronskim mrežama, gdje postoji na milijarde neurona, nisu svi neuroni aktivni u isto vrijeme jer imaju različite uloge i aktiviraju ih različiti podražaji. Slaba aktiviranost u modelima rezultira bržim i boljim predviđanjima i manjoj prenaučenošću [17].

2.3.1. DOKAZ DA JE ReLU NE-LINEARNA FUNKCIJA

Kako bi se dokazalo da je ReLU aktivacijska funkcija ne-linearna funkcija, dokazat će se da ReLU ne uspijeva biti linearna funkcija. Kao što je već spomenuto, aktivacijske funkcije su ne-linearne funkcije.

Za svaki realan broj x , funkcija f je definirana tako da je:

$$f(x) = \text{relu}(x) \quad (2.8)$$

Neka se pretpostavi da je a realan broj za koji vrijedi $a < 0$.

Koristeći pretpostavku da vrijedi $a < 0$, može se vidjeti da je:

$$f(-1a) = \max(0, -1a) > 0 \quad (2.9)$$

i da je:

$$(-1)f(a) = (-1)\max(0, a) = 0 \quad (2.10)$$

Radi ovoga dolazimo do zaključka:

$$f(-1a) \neq (-1)f(a) \quad (2.11)$$

Tako je dokazano da funkcija f , odnosno ReLU funkcija, ne uspijeva biti linearna funkcija iako se za pozitivne vrijednosti ponaša kao linearna funkcija [3].

2.3.2. UMIRUĆI ReLU

Kao što je već spomenuto, slaba aktiviranost ReLU funkcije je u većini slučajeva poželjna jer to rezultira bržim i boljim rezultatima te manjim rizikom pojave prenaučivosti. Međutim, negativna strana toga što ReLU poprima vrijednost nula za sve negativne vrijednosti je ta što se može pojaviti problem koji se naziva 'umirući ReLU' (eng. *dying ReLU*) [12, 17].

ReLU neuron je 'mrtav' ako je ostao zaglavljen na negativnoj strani vrijednosti zbog čega uvijek ima vrijednost nula. Kako je nagib ReLU funkcije za negativne vrijednosti nula, jednom kada neuron poprimi negativnu vrijednost, malo je vjerojatno da će se oporaviti od te vrijednosti. Takvi neuroni ne pridonose preslikavanju ulaza na izlaz te postaju praktički beskorisni. Tijekom vremena se može dogoditi da veliki dio mreže ne radi ništa (u nekim slučajevima čak oko 40%) [12, 17, 18].

Zbog toga se, osim ReLU funkcije, znaju koristiti njene izvedenice koje nastoje zaobići taj problem. Neke od njih će biti opisane u zasebnim poglavljima. Neke od izvedenica su:

1. *Leaky ReLU* [19],
2. *Parametric ReLU* (PReLU) [20],
3. *Thresholded ReLU* [21],
4. *Concatenated ReLU* (CReLU) [22].

2.4. SWISH AKTIVACIJSKA FUNKCIJA

Aktivacijske funkcije imaju dugu povijest. Prvo se koristila sigmoidalna funkcija zbog svoje jednostavne derivacije, opsega između nula i jedan te zbog svog probabilističkog oblika. Tanh funkcija se smatrala zamjenom za 'običnu' sigmoidalnu funkciju jer je ulazne vrijednosti preslikavala u vrijednosti koje se nalaze između minus jedan i jedan. Međutim, te su dvije funkcije u većini slučajeva zamijenjene ReLU aktivacijskom funkcijom [23].

Kao i ReLU, Swish [24], aktivacijska funkcija, razvijena od strane Google razvojnog tima, omeđena je s donje strane. To znači da, kako se ulaz x približava negativnim vrijednostima, y se približava nekoj konstantnoj vrijednosti. Isto tako je, kao i ReLU, neograničena s gornje strane (kako se x približava pozitivnoj vrijednosti, y se približava nekoj beskonačnoj vrijednosti). Ali, za razliku od ReLU funkcije, Swish je 'glatka', odnosno nema iznenadne promjene gibanja [23].

Matematička formula Swish aktivacijske funkcije glasi:

$$f(x) = x \times \frac{1}{1+e^{-x}} \quad (2.12)$$

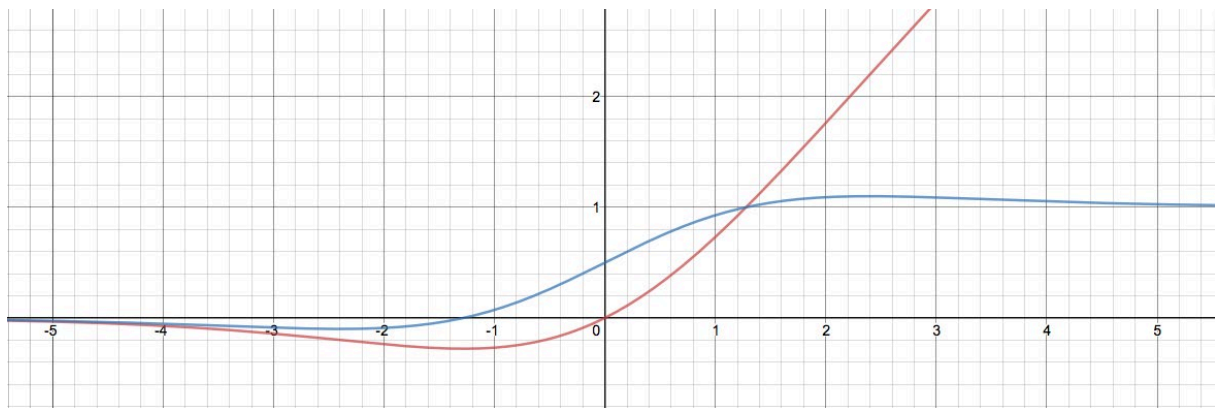
odnosno:

$$f(x) = x \times \text{sigmoidalna funkcija}(x) \quad (2.13)$$

Derivacija Swish funkcije je onda:

$$f'(x) = f(x) + \sigma(x)(1 - f(x)) \quad (2.14)$$

Grafički prikaz Swish funkcije kao i grafički prikaz njene derivacije nalaze se na slici (Slika 2.5).

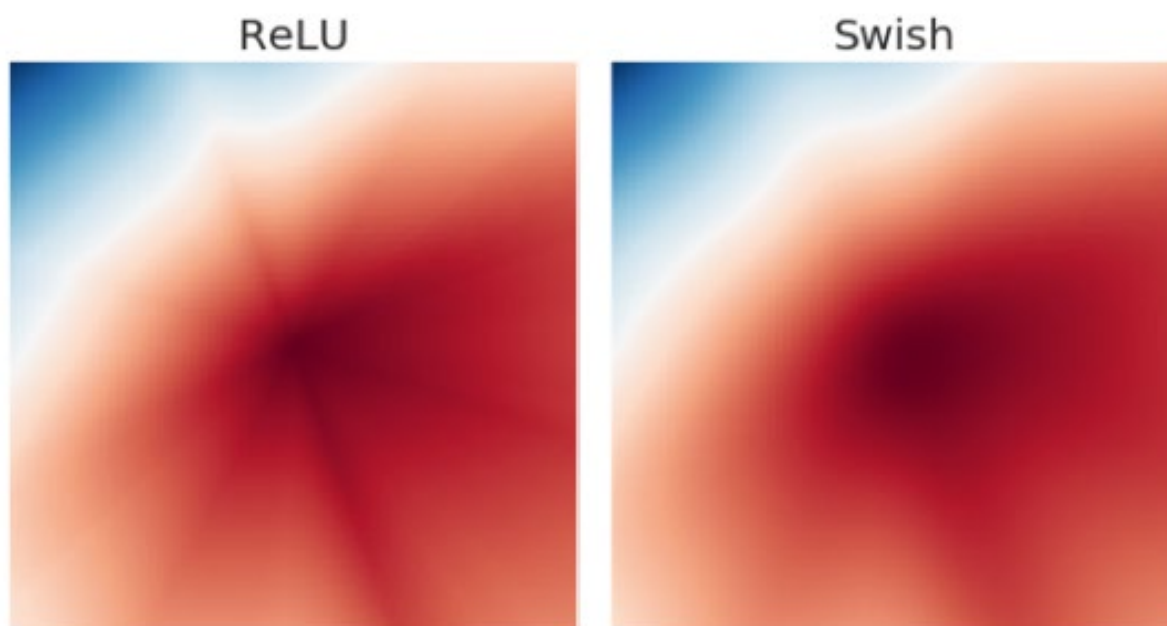


Slika 2.5: Swish aktivacijska funkcija (crveno) i njena prva derivacija (plavo).

2.4.1. SVOJSTVA SWISH FUNKCIJE

Neomeđenost funkcije je poželjna kod svake aktivacijske funkcije jer se time izbjegava dugo vrijeme učenja s vrijednostima gradijenata koje su blizu nule, kao što su sigmoidalna [11] i tanh [15] funkcije koje su ograničene s gornje i donje strane i zbog čega se mreže moraju oprezno inicijalizirati [23].

Imati funkciju koja je omeđena s donje strane, kao što je ReLU funkcija, daje prednost tijekom procesa učenja jer se odbacuju velike negativne ulazne vrijednosti. Nadalje, uglađenost funkcije pomaže kod optimiziranja i generalizacije neuronske mreže. Na dolje prikazanoj slici (Slika 2.6) uspoređene su ReLU i Swish funkcije gdje se vidi da Swish funkcija ima bolju uglađenost [23].



Slika 2.6: Usporedba ugladenosti ReLU i Swish funkcije [25].

Pejzažna glatkoća direktno korelira s pejzažem pogreški, što je on glađi, to je jednostavnije pronaći minimum [23].

Eksperimenti, koje je proveo Googleov razvojni tim, pokazali su da Swish funkcija daje bolje rezultate od ReLU funkcije kod dubokih neuronskih mreža. Na primjer, samo zamjenom ReLU funkcije Swish funkcijom, poboljšana je ispravnost klasifikacije ImageNet baze podataka za 0.9% i NASANetA za 0.6%. Nadalje, Swish je pokazala bolje izvođenje od ReLU kod dubokih mreža koje posjeduju između 40 i 50 slojeva, kada optimizacija postaje teška. Što se tiče veličine grupa (eng. *batch*), obje funkcije bilježe pad u kvaliteti izvođenja, ali i tada Swish ipak ima bolju izvedbu od ReLU funkcije [18].

2.5. SOFTMAX AKTIVACIJSKA FUNKCIJA

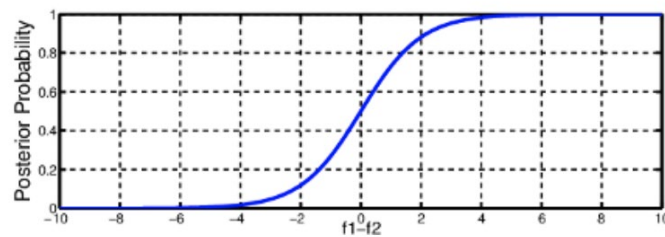
Još poznata i kao 'softargmax' funkcija ili više-klasna logistička regresija, softmax funkcija [26] je funkcija koja prima vektor K realnih vrijednosti i vraća vektor K realnih vrijednosti čiji zbroj daje jedan. Ulazne vrijednosti mogu biti pozitivne, negativne, nula, veće od jedan, manje od jedan, ali će ih softmax funkcija uvijek transformirati u vrijednosti koje se nalaze između nula i jedan tako da one mogu biti protumačene kao vjerojatnosti [27].

Matematička formula softmax funkcije slična je onoj sigmoidalne funkcije i glasi:

$$f(\vec{x})_i = \frac{e^{x_i}}{\sum_{j=1}^K e^{x_j}} \quad (2.15)$$

Sličnost je u tome što softmax funkcija prima vektor, dok sigmoidalna funkcija prima skalar. Dapače, sigmoidalna funkcija je posebni slučaj softmax funkcije kada je ulazne vrijednosti potrebno klasificirati u dvije izlazne klase [12].

Grafički prikaz sigmoidalne prikazan je na slici (Slika 2.7):



Slika 2.7: Grafički prikaz softmax funkcije [28].

Softmax funkcija se najčešće koristi u izlaznom sloju neuronske mreže gdje je potrebno izlazne rezultate prikazati kao vjerojatnosti da ulazni primjer odgovara određenom izlazu [12].

3. SEMANTIČKA SEGMENTACIJA

Semantička segmentacija jedan je od ključnih zadataka u području računalnog vida. Ona utire put prema računalnom razumijevanju okoline (eng. *scene understanding*). Razumijevanje okoline kao jedno od glavnih problema računalnog vida istaknuto je činjenicom da sve veći broj aplikacija ima koristi od razumijevanja slike. Neke od tih aplikacija su autonomna vozila, aplikacije koje koriste interakciju čovjeka i računala, računalna fotografija, preglednici za slike i izmijenjena stvarnost [29].

Semantička segmentacija nije izolirano polje proučavanja, već je ona prirodni korak u napredovanju iz grubog prema finom zaključivanju (optimizacija hiperparametara neuronske mreže tijekom koje se isprobavaju različite kombinacije hiperparametara s ciljem da se dobiju što bolji rezultati predviđanja). Evolucija raspoznavanja objekata (eng. *object recognition*) ili razumijevanja okoline kreće se od grubog prema finom zaključivanju. To su klasifikacija, detekcija (ili lokalizacija), semantička segmentacija i segmentacija instanci. Podrijetlo svega jest klasifikacija koja daje konačnu pretpostavku kojoj klasi pripada cijeli ulaz. Lokalizacija ili detekcija ne samo da klasificira cijeli ulaz, već daje i informacije o prostornoj lokalizaciji tih klasa. Cilj semantičke segmentacije je dati 'guste' pretpostavke tako da svaki piksel na slici označi određenom klasom. Tako je svaki piksel klasificiran klasom objekta ili područja koje ga okružuje. Drugim riječima, cilj je da se, kada se segmentacijskom algoritmu proslijedi slika, dobije izlaz koji pikseli na slici semantički pripadaju zajedno. Trenutno nije poznato na koji način ljudski mozak pronalazi ispravnu segmentaciju. Segmentiranje slike uključuje duboko semantičko razumijevanje svijeta te razumijevanje koje stvari čine cjelinu [29, 30, 31, 32].

Razlika između semantičke segmentacije s jedne strane i klasifikacije i raspoznavanja objekata s druge strane jest ta što za semantičku segmentaciju nije potrebno unaprijed znati koji se vizualni koncepti ili objekti nalaze na slici. Idealni algoritam za segmentaciju slike će uspješno segmentirati nepoznate objekte, odnosno objekte koji su novi ili još neidentificirani [29].

Metode semantičke segmentacije dijelimo na tri kategorije: na semantičku segmentaciju koja se temelji na regiji, semantičku segmentaciju koja se temelji na potpuno konvolucijskim mrežama (eng. *fully convolutional networks*, FCN) i konačno na metodu slabo nadzirane

segmentacije. S obzirom da je ovaj rad napravljen korištenjem metode potpuno konvolucijskih mreža, samo će se ona opisati [29].

3.1. SEMANTIČKA SEGMENTACIJA TEMELJENA NA POTPUNO KONVOLUCIJSKIM MREŽAMA

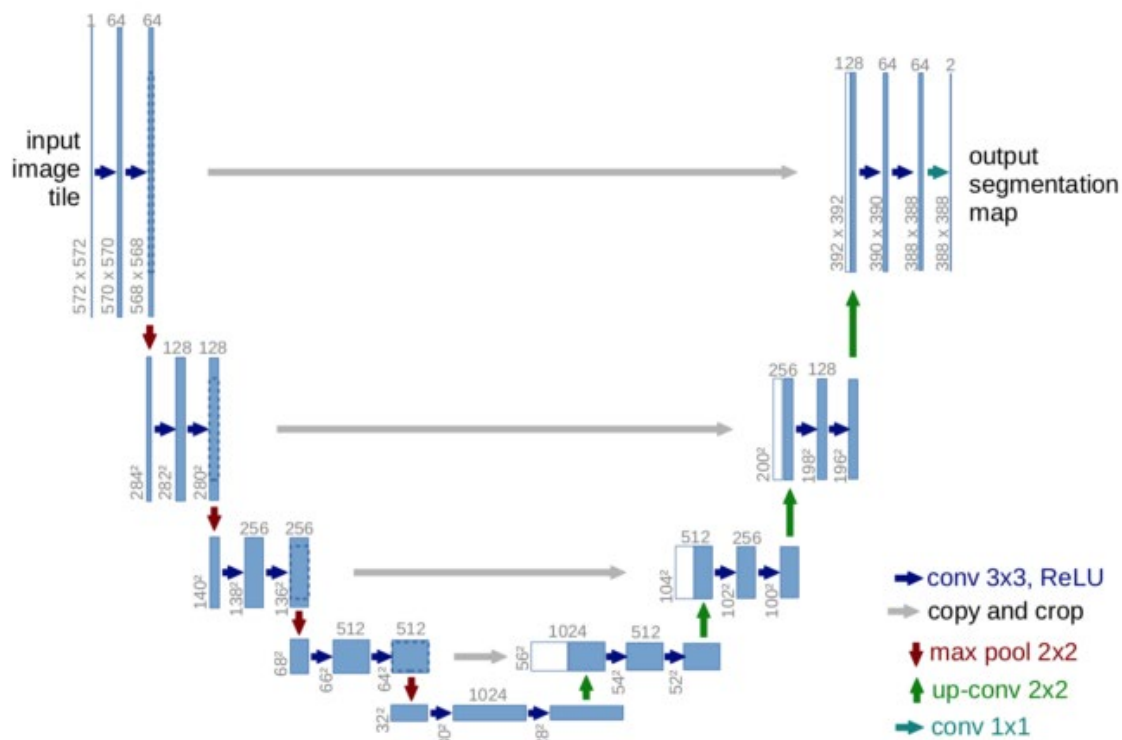
Potpuno konvolucijska mreža proširenje je klasične konvolucijske mreže. Ideja se prvi put pojavila od strane Matan et al. [33] da se konvolucijske mreže prošire tako da mogu primiti ulazne podatke proizvoljnih veličina. To je prvi put učinjeno proširenjem LeNet mreže [28] s ciljem prepoznavanja znamenki i brojeva. Glavna ideja je bila uzeti klasičnu konvolucijsku mrežu i učiniti da ona na svoj ulaz prima slike proizvoljnih dimenzija. Ograničenje klasičnih konvolucijskih mreža da mogu prihvatiti samo ulaze određenih veličina dolazi od potpuno povezanog sloja koji je, po definiciji, fiksiran. S druge strane, potpuno konvolucijske mreže sadrže samo konvolucijske slojeve i slojeve sažimanja (ako se aktivacijski sloj u tom slučaju smatra dijelom konvolucijskog sloja) zbog čega te mreže mogu dati predviđanja i za ulaze proizvoljnih veličina. U ovom slučaju veličina izlaza ovisi o veličini ulaza u mrežu. Potpuno konvolucijske mreže obično koriste softmax funkciju za mjerenje pogreške [29, 31].

Jedan od problema s kojim se susreću potpuno konvolucijske mreže je taj da, propagacijom kroz više konvolucijskih slojeva i slojeva sažimanja, dolazi do smanjene rezolucije matrice značajki. Zbog toga direktna predviđanja potpuno konvolucijskih mreža imaju manju rezoluciju od ulaza što rezultira relativno nejasnim granicama objekata [29].

3.2. U-NET ARHITEKTURA POTPUNO KONVOLUCIJSKE MREŽE

U-Net arhitektura [34] razvijena je od potpuno konvolucijske neuronske mreže 2015. kako bi procesirala biomedicinske slike. U biomedicinskim slučajevima cilj je ne samo dijagnosticirati postoji li anomalija, već joj znati i lokaciju ako neka anomalija postoji.

Arhitektura U-Net mreže prikazana je na slici (Slika 3.1):



Slika 3.1: Arhitektura U-Net mreže [35].

Ronneberg u svom [34] u kojem predstavlja U-Net mrežu piše kako predstavlja mrežu i strategiju treniranja mreže koji ovise o jakoj upotrebi izmijenjenih podataka. Arhitektura se sastoji od puta sažimanja (eng. *contracting path*) kojem je cilj razumjeti kontekst slike i od puta razvijanja (eng. *expanding path*) kojem je cilj moći precizno locirati objekte. Po Ronneberg-u [24], ova mreža se može uspješno trenirati od početka do kraja korištenjem malog skupa slika te ona svojim svojstvima nadigrava dotadašnje najbolje metode.

Put sažimanja (lijeva strana U-Net arhitekture koja je prikazana na slici (Slika 3.1)) se sastoji od ponavljane primjene 3×3 nepopunjenih konvolucija (eng. *unpadded convolutions*). Nakon svake primjene konvolucije slijede primjena ReLU aktivacijske funkcije i 2×2 sažimanje po maksimalnoj vrijednosti s korakom 2 za smanjenje uzorkovanja. Tijekom svakog koraka sažimanja, udvostručuje se broj kanala značajki (eng. *feature channels*). Svaki korak na putu razvijanja se sastoji od preuzorkovanja (eng. *upsampling*) matrice značajki nakon čega slijedi primjena 2×2 'transponirane' konvolucije (eng. *up-convolution*) koja raspolavlja broj kanala značajki, to je spajanje odgovarajuće obrezane matrice značajki iz puta sažimanja i dvije 3×3 konvolucije gdje nakon svake konvolucije slijedi primjena ReLU aktivacijske funkcije. U

zadnjem sloju U-Net mreže se primjenjuje 1x1 konvolucija koja služi za pridruživanje svakog piksela određenoj klasi [34].

3.3. PREDPROCESIRANJE I AUGMENTACIJA PODATAKA ZA SEMANTIČKU SEGMENTACIJU

Kao što je spomenuto u prethodnom poglavlju, U-Net mreža se može uspješno trenirati korištenjem malog skupa slika. Također je spomenuto kako ova arhitektura ovisi o jakoj upotrebi izmijenjenih podataka. Augmentacija podataka je danas uobičajena tehnika koja pomaže kod treniranja modela strojnog učenja te dubokih arhitektura na način da ili ubrzava konvergenciju ili provodi regularizaciju modela čime se izbjegava prenaučenosti i povećava se sposobnost uopćavanja [30, 34].

Augmentacija podataka se obično sastoji od primjene određenog broja transformacija nad podacima te na taj način generira nove podatke. Postoji više vrsta transformacija koje se mogu primijeniti na slikama kao što su translacija, rotacija, obrazivanje i slično. Ona je posebno korisna kod malih skupa podataka [30].

Podaci su jedan od najvažnijih dijelova strojnog učenja. Ta činjenica postala je uočljivija razvojem dubokih mreža. Skupljanje podataka i sastavljanje prikladnog skupa podataka koji mora biti dovoljno velik i dovoljno reprezentativan zahtijeva mnogo truda, vremena, znanja i infrastrukture. Zbog toga mnogo stručnjaka odlučuje koristiti već postojeći i standardizirani skup podataka koji je dovoljno reprezentativan za njihov problem. Tako su najpoznatije baze slika koje se koriste tijekom procesa učenja semantičke segmentacije sljedeće:

1. PASCAL Visual Object Classes (VOC) [36]: ovaj skup slika sadrži ukupno 21 klasu. Sadrži *ground-truth* označen skup slika. Ovaj skup slika podijeljen je na dva podskupa: na skup za treniranje od 1464 slike i na skup za validaciju od 1449 slike.
2. PASCAL Context [37]: ovaj skup je proširenje PASCAL VOC skupa. Sastoji se od ukupno 10103 slike. Sadrži 540 klasa uključujući originalnu 21 klasu. Unatoč velikom broju klasa, njih 59 koje se najčešće ponavljaju su vrijedne pažnje.
3. PASCAL Part [38]: ovaj skup je također proširenje PASCAL VOC skupa. Skup sadrži originalne klase s tim da su uvedene pod-klase. Tako je klasa bicikla raščlanjena na klase stražnjeg kola, lanca za kolo, prednjeg kola, volana, svjetla i sjedala. Sadrži sve

slike za treniranje i validaciju kao i originalni skup podataka uz dodatnih 9637 slika za testiranje.

4. Semantic Boundaries Dataset (SBD) [39]: ovaj skup je također proširenje PASCAL VOC skupa. Sadrži vlastitih 8498 slika za treniranje i 2857 slika za validaciju. Često se koristi kao zamjena za PASCAL VOC za duboko učenje.
5. Microsoft Common Objects in Context (COCO) [40]: sadrži više od 80 klasa. Sastoji se od 82783 slike za treniranje, 40504 slike za validaciju te više od 80000 slika za testiranje. Sam skup slika za testiranje je podijeljen na 4 podskupa.
6. SYNTHetic Collection of Imagery and Annotations (SYNTHIA) [41]: je skup fotorealističnih prikaza virtualnog semantički segmentiranog grada, čija je svrha razumijevanje okoline u kontekstu vožnje. Sadrži 11 klasa i 13407 slika za treniranje.
7. Cityscapes [42]: skup slika koji se fokusira na semantičko razumijevanje gradskih ulica. Pruža semantičke, pojedinačne i guste oznake za piksele s 30 klasa podijeljenih u 8 kategorija. Skup sadrži oko 5000 fino označenih slika i 20000 grubo označenih slika (fino označavanje slika se odnosi na označavanje slike oznakom koja ju detaljno opisuje dok se grubo označavanje odnosi na oznaku koja na općenitiji način opisuje sliku).
8. CamVid [43]: skup slika za razumijevanje okoline tijekom vožnje. Sadrži slike označene s 32 klase. Skup je podijeljen na 367 slika za treniranje, 100 slika za validaciju i 233 slike za testiranje.
9. KITTI [44]: jedan od najpopularnijih skupova slika za mobilnu robotiku i autonomnu vožnju. Ne sadrži *ground-truth* oznake za semantičku segmentaciju, već su istraživači pozvani manualno anotirati dijelove skupa za vlastite potrebe.
10. Youtube-Objects [45]: skup videa sakupljenih s YouTube-a koji sadrže PASCAL VOC klase. Sadrži sveukupno 10167 slika.
11. Materials in Context (MNIC) [46]: skup slika koji sadrži ukupno 23 klase. Sadrži 7061 označen materijal za treniranje segmentacije, 5000 za testiranje i 2500 za validaciju. Glavni izvor za ove slike je OpenSurfaces skup podataka.
12. Densely-Annotated Video Segmentation (DAVIS) [47]: skup razvijen za potrebe video segmentacije objekata. Sastoji se od 59 sekvenci visoke rezolucije koje zajedno čine 4219 okvira za treniranje i 2023 okvira za validaciju. Skup je dizajniran tako da ne sadrži veći broj objekata koji se značajno kreću.
13. Stanford background [48]: skup slika sa slikama vanjskog prostora koji se sastoji od slika iz različitih skupova: LabelMe, MSRC, PASCAL VOC i Geometric Context. Sadrži 715 slika s najmanje jednim objektom u prednjem dijelu slike.

14. SiftFlow [49]: sastoji se od ukupno 2688 u potpunosti označenih slika koje čine podskup LabelMe [50] skupa. Sadrži 33 klase. Neoznačeni pikseli ili pikseli koji su označeni različitom od 33 klase tretiraju se kao neoznačeni [30].

4. KERAS

Keras [51] je API za duboko učenje napisan u Python programskom jeziku. Izgrađen je na Tensorflow 2.0 *end-to-end open-source* platformi za strojno učenje. Razvijen je s namjernom da se inženjerima i znanstvenicima omogući brze provjere rješenja strojnog učenja. Inicijalno je razvijen u sklopu istraživanja na projektu ONEIROS (eng. *Open-ended Neuro-Electronic Intelligent Robot Operating System*).

Karakteristike Kerasa su:

1. Jednostavan je na način da reducira kognitivno opterećenje razvojnog programera.
2. Fleksibilan tako što prisvaja načelo progresivnog otkrivanja složenosti (eng. *progressive disclosure of complexity*).
3. Nudi industrijske performanse i skalabilnost. Koriste ga firme kao što su NASA (eng. *National Aeronautics and Space Administration*), YouTube i Waymo.

Tensorflow kombinira 4 ključne sposobnosti:

1. Efikasno izvodi jednostavne operacije s matricama korištenjem CPU-a, GPU-a ili TPU-a.
2. Izračunava gradijent proizvoljnih izraza koji se mogu derivirati.
3. Skalira izračune na različitim uređajima.
4. Izvodi programe na vanjske *runtime*-ove poput servera, web preglednika, mobilnih i ugradbenih uređaja.

Keras/Tensorflow 2.0 kompatibilan je s:

1. Python-ovim verzijama 3.5-3.8,
2. Ubuntu 16.04 i kasnijim verzijama,
3. Windows 7 i kasnijim verzijama,
4. macOS 10.12.6 i kasnijim verzijama.

4.1. SLOJEVI I MODELI U KERASU

Slojevi i modeli čine temeljne strukture podataka u Kerasu. Najjednostavniji model u Kerasu je sekvencijalni model; linearni stog slojeva. Od složenijih arhitektura, Keras raspolaže funkcionalnim API-jem koji omogućuje izgradnju proizvoljnih grafova slojeva.

Ovaj rad izrađen je korištenjem U-Net arhitekture potpuno konvolucijske neuronske mreže. Implementacija mreže u Kerasu nastala je slaganjem različitih slojeva što ju čini sekvencijalnim modelom. Slojevi koji su se koristili za implementaciju su:

1. *SeparableConv2D*
2. *BatchNormalization*
3. *Activation*
4. *MaxPooling2D*
5. *concatenate*
6. *Conv2DTranspose*
7. *Dropout*

4.1.1. *SeparableConv2D* sloj

SeparableConv2D odabran je umjesto Keras-ovog osnovnog *Conv2D* sloja jer smanjuje složenost podataka koji ulaze u sloj te ubrzava izračune. Prvo provodi dubinsku prostornu konvoluciju te nakon toga provodi konvoluciju po točkama (eng. *dot convolution*) [52].

Sloj može primiti više parametara čiji se popis nalazi na Keras-ovoj službenoj stranici [53]. Parametri korišteni u ovom radu su:

1. *filters*: određuje broj filtera u sloju. Ne posjeduje osnovno postavljenu vrijednost već očekuje broj filtera. Tijekom izrade ovog rada broj filtera se mijenjao ovisno o sloju u mreži.
2. *kernel_size*: određuje veličinu filtera koji prolazi po ulaznim vrijednostima. Također ne posjeduje osnovno postavljenu vrijednost već očekuje dimenziju filtera. U ovom radu je dimenzija postavljena na tri jer je to najčešće korištena osnovna dimenzija filtera.
3. *padding*: određuje hoće li se koristiti popuna ili ne radi očuvanja dimenzija ulaza. Osnovno postavljena vrijednost mu je '*valid*', dok se u ovom radu koristila vrijednost '*same*' kako bi se sačuvala dimenzije ulaza i jer će izlaz iz ovog sloja biti sažet prolaskom kroz sloj *MaxPooling2D*.

Ako se koristi *channel first* pristup, sloj na svoj ulaz prima 4D tenzor oblika (veličina grupe, broj kanala, broj redaka, broj stupaca) te na izlazu vraća 4D tenzor oblika (veličina grupe, filteri, novi redci, novi stupci).

4.1.2. BatchNormalization sloj

BatchNormalization sloj se koristi kako bi se normalizirali podaci koji izlaze iz konvolucijskog sloja. Izvodi transformacije koje srednju vrijednost izlaza zadržavaju blizu nule te standardnu devijaciju izlaza blizu jedan.

Sloj može primiti više parametara čiji se popis i čiji se opisi nalaze na Keras-ovoj službenoj stranici [54]. U ovom radu se nisu posebno postavljale vrijednosti nijednog parametra ovog sloja.

Ako se sloj, kao prilikom izrade ovog rada, ne pozove ni s jednim postavljenim argumentom, tada će vrijednosti parametara biti postavljene na osnovne. *BatchNormalization* sloj prima ulaz proizvoljnog oblika te vraća izlaz istog oblika kao i ulaz.

4.1.3. Activation sloj

U Kerasu, aktivacijske funkcije mogu se pozivati samostalno korištenjem aktivacijskog sloja ili u sklopu drugih slojeva. Jedan primjer bi bio postavljanjem parametra *activation* u *Conv2D* sloju. Tijekom izrade ovog rada, aktivacijske funkcije pozivane su korištenjem aktivacijskog sloja.

Aktivacijski sloj prima jedan *activation* parametar čija vrijednost označava ime aktivacijske funkcije koja se poziva. *Activation* sloj prima ulaz proizvoljnog oblika te vraća izlaz istog oblika kao i ulaz [55].

4.1.4. MaxPooling2D sloj

MaxPooling2D sloj izvodi operaciju sažimanja nad 2D podacima. On smanjuje prostorne dimenzije ulaza tako da uzima maksimalnu vrijednost na ulaznim okvirom. Ne smanjuje

dubinsku dimenziju ulaza, odnosno broj kanala. Okvir *MaxPooling2D* sloja pomiče se za onoliko mjesta po ulaznom tenzoru koliko je određeno *stride* parametrom.

Cijeli popis parametara koji ovaj sloj može primiti kao i njihova objašnjenja nalazi se na Keras-ovoj stranici [56]. Tijekom izrade ovog rada, korištene su osnovno postavljene vrijednosti parametara.

Ako se koristi *channels first* pristup, sloj prima 4D tenzor oblika (veličina grupe, kanali, redci, stupci) i na izlazu daje 4D tenzor oblika (veličina grupe, kanali, sažeti redci, sažeti stupci).

4.1.5. Conv2DTranspose sloj

Conv2DTranspose je sloj koji se koristi tijekom takozvanog 'puta razvijanja' (eng. *expansive path*) i koji izvodi takozvanu operaciju transponirane konvolucije ili dekonvolucije kojom se unaprjeđuju kontrast i rezolucija slika koje su prethodno prošle put sažimanja (eng. *contracting path*).

Cijeli popis parametara koji ovaj sloj može primiti nalazi se na službenoj Keras-ovoj stranici [57]. Tijekom izrade ovog rada, postavljeni su parametri:

1. *filters*: određuje broj filtera u sloju. Ne posjeduje osnovno postavljenu vrijednost već očekuje broj filtera. Tijekom izrade ovog rada broj filtera se mijenjao ovisno o sloju u mreži.
2. *kernel_size*: određuje veličinu filtera koji prolazi po ulaznim vrijednostima. Također ne posjeduje osnovno postavljenu vrijednost već očekuje dimenziju filtera. U ovom radu je dimenzija postavljena na tri jer je to najčešće korištena osnovna dimenzija filtera.
3. *padding*: određuje hoće li se koristiti popuna ili ne radi očuvanja dimenzija ulaza. Osnovno postavljena vrijednost mu je 'valid', dok se u ovom radu koristila vrijednost 'same' kako bi se sačuvala dimenzije ulaza i jer će izlaz iz ovog sloja biti sažet prolaskom kroz sloj *MaxPooling2D*.
4. *strides*: definira za koliko će se svaki filter pomicati po ulaznom tenzoru. Osnovno postavljena vrijednost mu je (1, 1) koja je u ovom radu postavljena na (2, 2) kako bi filter brže prolazio po svojim ulazima.

Ako se koristi *channels first* pristup, ulaz u ovaj sloj je 4D tenzor oblika (veličina grupe, broj kanala, broj redaka, broj stupaca), a izlaz iz sloja je 4D tenzor oblika (veličina grupe, filteri, novi redci rows, novi stupci).

4.1.6. Dropout sloj

Dropout sloj unaprijed postavljenom učestalošću nasumično postavlja vrijednosti ulaza na nulu te na taj način pomaže spriječiti pojavu prenaučivosti. Ovaj sloj se koristi samo tijekom procesa učenja mreže.

Popis svih parametara koje ovaj sloj može primiti nalazi se na službenoj Keras-ovoj stranici [58]. Tijekom izrade ovog rada postavljena je vrijednost parametra *rate* koja definira učestalost postavljanja vrijednosti na nulu jer parametar nema postavljenu osnovnu vrijednost. Parametar je postavljen na 0,5 što je česta praksa tijekom procesa učenja modela.

4.2. KOMPILIRANJE I UČENJE MODELA

Model se kompilira poziv *compile* metode sekvencijalnog modela koji konfigurira model za fazu učenja.

Popis argumenata koji ova metoda može primiti nalazi se na službenoj Keras-ovoj stranici [59]. Argumenti korišteni tijekom izrade ovog rada su sljedeći:

1. *optimizer*: definira ime algoritma za optimizaciju modela. U ovom radu je odabran Adam algoritam za optimizaciju. To je algoritam stohastičkog gradijentnog spusta koji se temelji na adaptivnoj procjeni momentuma prvog i drugog reda [60].
2. *loss*: funkcija koja računa koliko je model daleko od minimalnog gubitka. U ovom radu je odabrana *CategoricalCrossentropy* funkcija koja se koristi kada je potrebno dati predviđanja za dvije ili više oznaka [61].
3. *metrics*: funkcija kojom se ocjenjuje učenje modela. U ovom radu je odabrana *Accuracy* metrika koja računa koliko često se predviđanja modela poklapaju s unaprijed određenim oznakama za dani ulaz [62].

Proces učenja modela započinje poziv *fit* funkcije kojoj su tijekom izrade ovog rada postavljeni sljedeći argumenti:

1. *x*: skup ulaznih podataka, osnovno postavljena vrijednost je *None*
2. *y*: skup odgovarajućih oznaka za svaki pojedini ulaz, osnovno postavljena vrijednost je također *None*
3. *batch_size*: veličina skupa podataka koji se kao jedna cjelina prosljeđuju modelu, također mu je osnovno postavljena vrijednost *None*
4. *epochs*: broj epoha tijekom kojih će model učiti, osnovno postavljena vrijednost mu je jedan, dok je tijekom izrade ovog rada on bio postavljen na petsto
5. *callbacks*: lista *callback* instanci koje će se primijeniti tijekom procesa učenja, osnovno postavljena vrijednost mu je *None* dok je tijekom izrade ovog rada bio postavljen na listu s *ModelCheckpoint* vrijednostima.
6. *validation_data*: skup podataka za validaciju modela, osnovno postavljena vrijednost mu je također *None* dok je tijekom izrade ovog rada bio postavljen na skup podataka za validaciju

5. USPOREDBA AKTIVACIJSKIH FUNKCIJA

U ovom poglavlju će se usporediti aktivacijske funkcije koje se najčešće koriste za probleme semantičke segmentacije. To su u prvom redu ReLU i swish aktivacijske funkcije koje bi po [23] trebale dati najbolja predviđanja. Tu su još sigmoidalna i tangentno-hiperbolična funkcija koje su se dugo vremena koristile za probleme dubokog učenja, ali se manje koriste od pojave ReLU i swish funkcija. Na kraju je tu i softmax funkcija koja se najčešće koristi kao aktivacijska funkcija u posljednjem sloju neuronske mreže i koja daje konačna predviđanja koji bi piksel trebao biti označen kojom oznakom.

Model se trenirao korištenjem FESB MLID baze slika koja originalno sadrži 400 slika, 200 za treniranje i 200 za validaciju [63]. Za potrebe ovog diplomskog rada, skup je korištenjem *albumations* biblioteke [64] proširen s 400 na 1600 slika te je 1400 slika izdvojeno za potrebe treniranja, a 200 za potrebe validacije. Slike koje su se izdvojile u skup za validaciju odabrane su nasumično korištenjem *random* funkcije u Python-u kojoj je početni *seed* postavljen na 1337.

S obzirom na ograničenje računalne snage (AMD EPYC 7302P sa 16 jezgri, 128 GB RAM-a, DDR4 2933 MHz, GPU: Tesla T4; dio na kojem je izvođen kod za potrebe ovog rada sadrži 4 jezgre i 16GB RAM-a) sve slike su, ulazeći u model, redimenzionirane na veličinu 256x256. Veličina skupa koji kao jedan ulazi u model (eng. *batch size*) iznosi 100. Model je učio tijekom 500 epoha te je na kraju svake epohe bio evaluiran skupom za validaciju. Model sprema težine veza i to samo one koje rezultiraju najmanjim validacijskim gubitkom.

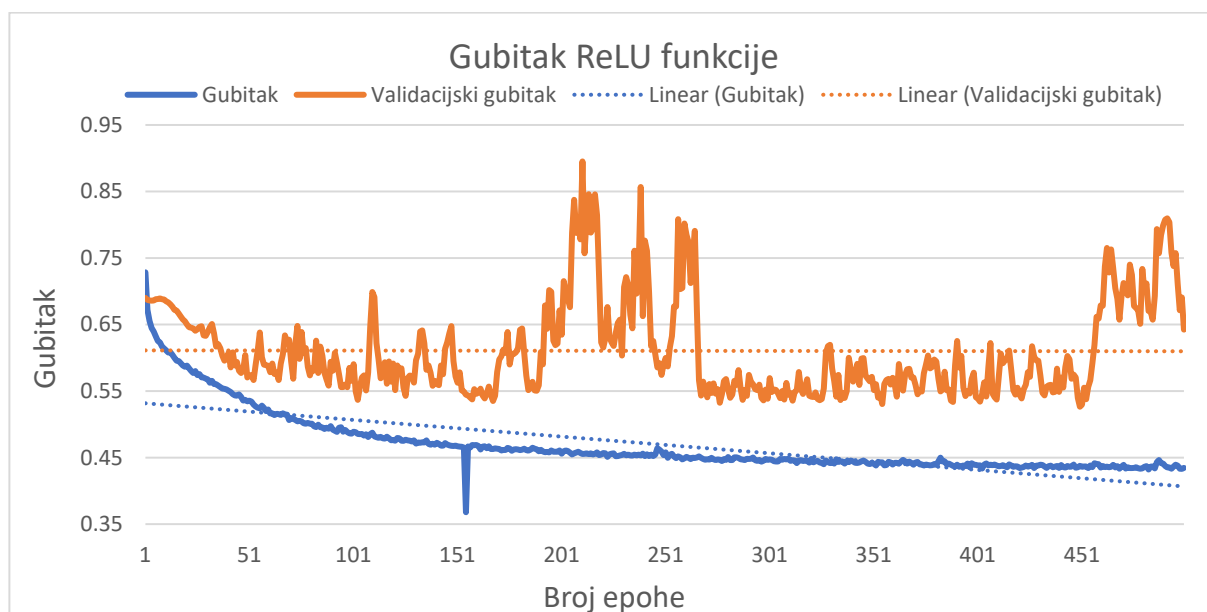
U prvom dijelu ovog poglavlja su analizirani rezultati izvršavanja svake pojedine epohe svake pojedine aktivacijske funkcije. Nakon toga su aktivacijske funkcije paralelno analizirane kako te su uspoređene prema najvišim i najnižim vrijednostima parametara te su spomenute međusobne sličnosti ako postoje.

5.1. ReLU AKTIVACIJSKA FUNKCIJA

Prosječno vrijeme izvođenja ReLU funkcije je 638,348 sekundi po epohi. Najdulje vrijeme izvođenja iznosi 648 sekundi, a najkraće 510 sekundi. Prosječno vrijeme za parametar

sekundi/koraku je 35,138 sekundi s najvišom vrijednosti 36 sekundi/koraku i najnižom vrijednosti od 25 sekundi/koraku. Prosječni gubitak tijekom faze učenja ReLU aktivacijske funkcije je 0,4691574 s minimalnim gubitkom od 0,3679 i maksimalnim gubitkom od 0,7285. Validacijski gubitak ima prosječnu vrijednost od 0,6105008 s minimalnim gubitkom od 0,5265 i maksimalnim gubitkom od 0,8952. Prosječna točnost ReLU funkcije tijekom faze učenja iznosi 0,39044742 s minimalnom točnošću od 0,2219 i maksimalnom točnošću od 0,7877. Prosječna validacijska točnost joj iznosi 0,418250494 s minimalnom točnošću od 0,000005722 i maksimalnom točnošću od 0,9864.

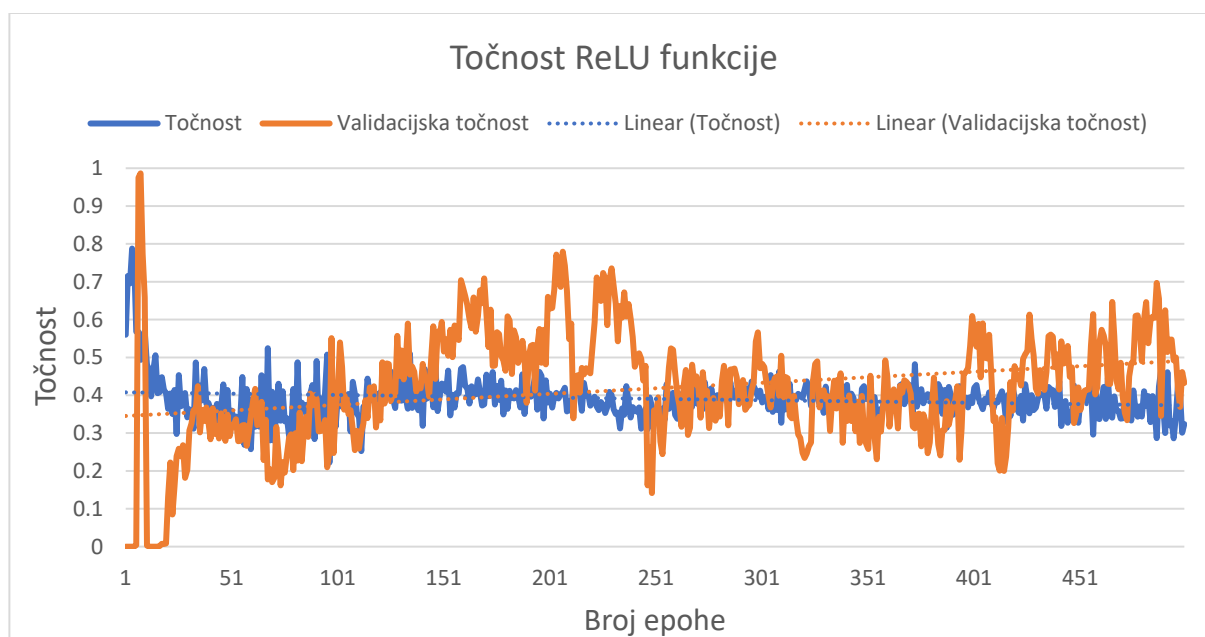
Usporedba gubitka tijekom faze učenja i validacijskog gubitka prikazana je na sljedećem grafu (Slika 5.1):



Slika 5.1: Usporedba gubitaka ReLU funkcije.

Kao što se može vidjeti na priloženom grafu, gubitak tijekom faze učenja brže pada od validacijskog gubitka koji pokazuje jako blagu tendenciju opadanja.

Usporedba točnosti funkcije tijekom faze učenja i validacijske točnosti prikazana je sljedećim grafom (Slika 5.2):



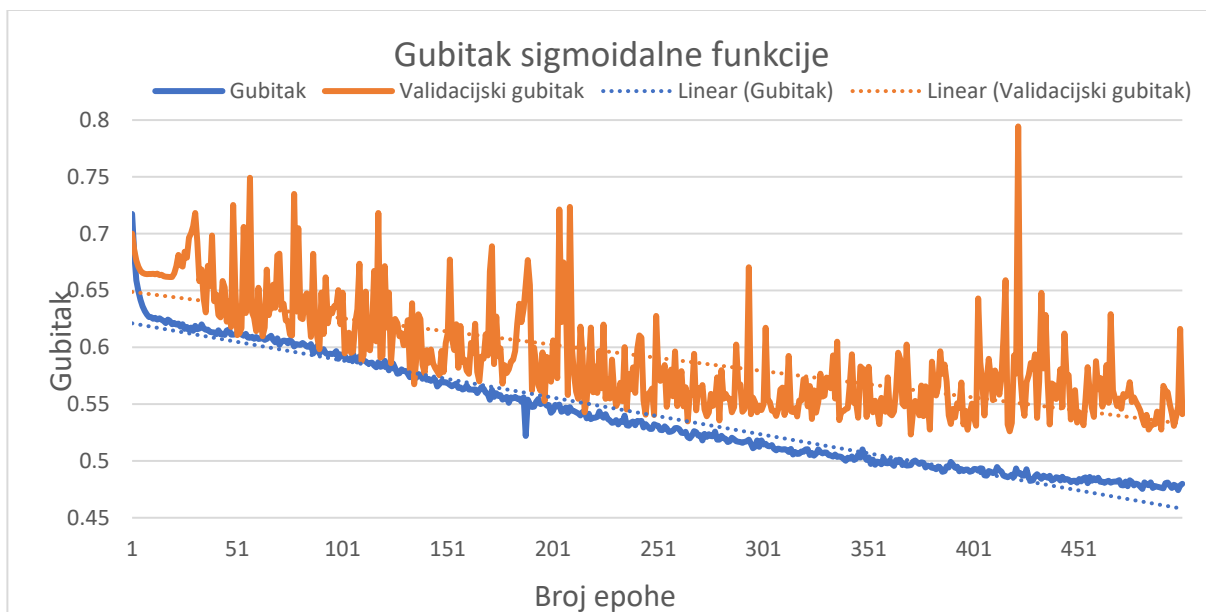
Slika 5.2: Usporedba točnosti ReLU funkcije.

Za razliku od gubitka, ovdje validacijska točnost raste brže od točnosti tijekom faze učenja koja u ovom slučaju čak pokazuje tendenciju opadanja.

5.2. SIGMOIDALNA AKTIVACIJSKA FUNKCIJA

Prosječno vrijeme izvođenja sigmoidalne aktivacijske funkcije je 723,168 sekundi po epohi s najduljim vremenom izvođenja od 1007 sekundi i najkraćim vremenom izvođenja od 586 sekundi po epohi. Prosječno vrijeme za parametar sekundi/koraku je 40,03 sekundi/koraku s najvišom vrijednošću 55 sekundi/koraku i najnižom vrijednošću od 33 sekunde/koraku. Prosječni gubitak sigmoidalne aktivacijske funkcije tijekom faze učenja iznosi 0,5395474 s minimalnim gubitkom od 0,4743 i maksimalnim gubitkom od 0,7174. Prosječna vrijednost validacijskog gubitka iznosi 0,5908033 s minimalnim gubitkom od 0,5233 i maksimalnim gubitkom od 0,7944. Prosječna točnost sigmoidalne aktivacijske funkcije tijekom faze učenja iznosi 0,3734298 s minimalnom točnošću od 0,1757 i maksimalnom točnošću od 0,8547. Prosječna validacijska točnost joj iznosi 0,38423111 s minimalnom točnošću od 1,5259E-05 i maksimalnom točnošću od 1.

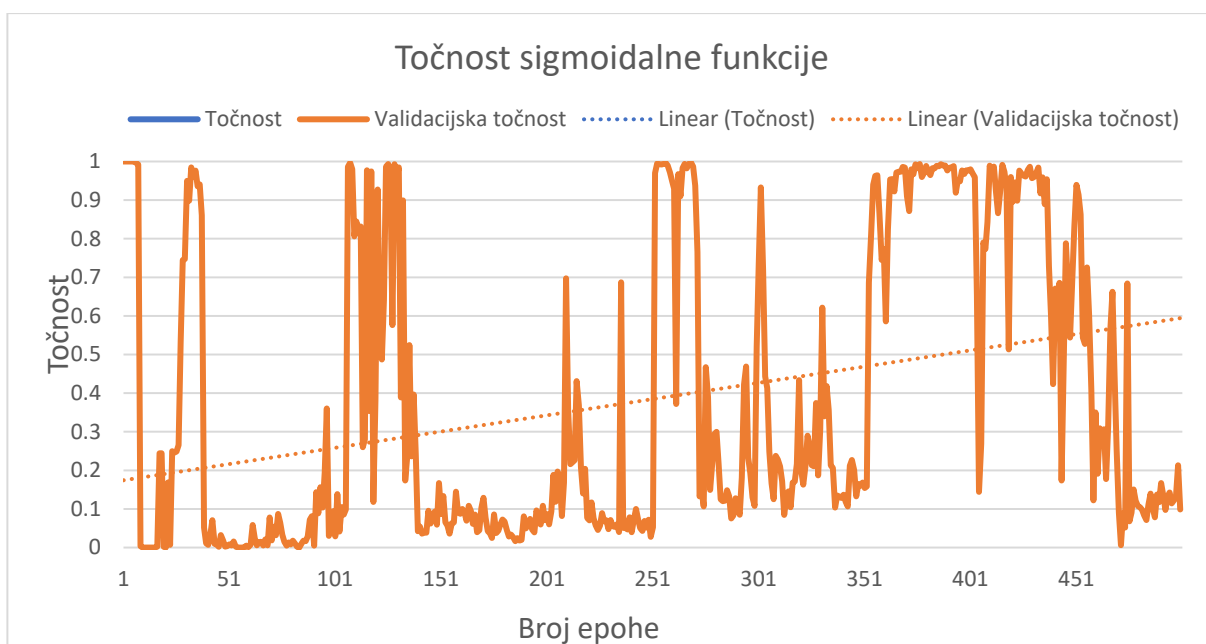
Usporedba gubitka tijekom faze učenja i validacijskog gubitka prikazana je na sljedećem grafu (Slika 5.3):



Slika 5.3: Usporedba gubitaka sigmoidalne funkcije.

I gubitak tijekom faze učenja i validacijski gubitak pokazuju tendenciju opadanja s tim da tendencija gubitka tijekom faze učenja konstantno ima manju vrijednost od tendencije validacijskog gubitka.

Usporedba točnosti funkcije tijekom faze učenja i validacijske točnosti prikazana je sljedećim grafom (Slika 5.4):



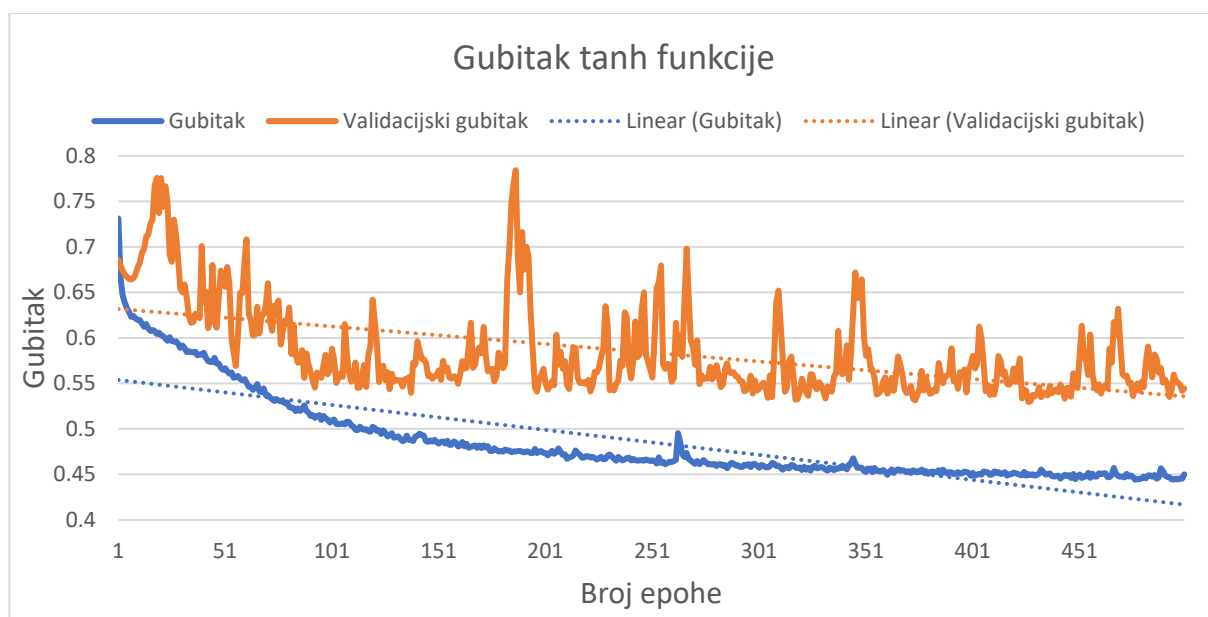
Slika 5.4: Usporedba točnosti sigmoidalne funkcije.

Iako tendencija točnosti tijekom faze učenja u početnoj fazi pokazuje bolju tendenciju rasta od tendencije validacijske točnosti, u drugom dijelu faze učenja tendencija validacijske točnosti poprima više vrijednosti.

5.3. TANGENTNO-HIPERBOLIČNA AKTIVACIJSKA FUNKCIJA

Prosječno vrijeme izvođenja tangentno-hiperbolične aktivacijske funkcije iznosi 718,478 sekundi po epohi s najduljim vremenom izvođenja od 976 sekundi i najkraćim vremenom izvođenja od 529 sekundi po epohi. Prosječno vrijeme za parametar sekundi/koraku je 39,786 sekundi/koraku s najvišom vrijednošću 54 sekunde/koraku i najnižom vrijednošću od 29 sekundi/koraku. Prosječni gubitak tangentno-hiperbolične aktivacijske funkcije tijekom faze učenja iznosi 0,4852694 s minimalnim gubitkom od 0,4441 i maksimalnim gubitkom od 0,7312. Prosječna vrijednost validacijskog gubitka iznosi 0,58380252 s minimalnim validacijskim gubitkom od 0,5296 i maksimalnim validacijskim gubitkom od 0,7842. Prosječna točnost tangentno-hiperbolične funkcije tijekom faze učenja iznosi 0,3697506 s minimalnom točnošću od 0,1671 i maksimalnom točnošću od 0,4533. Prosječna validacijska točnost joj je 0,40869941 s minimalnom točnošću od 0 i maksimalnom točnošću od 0,9288.

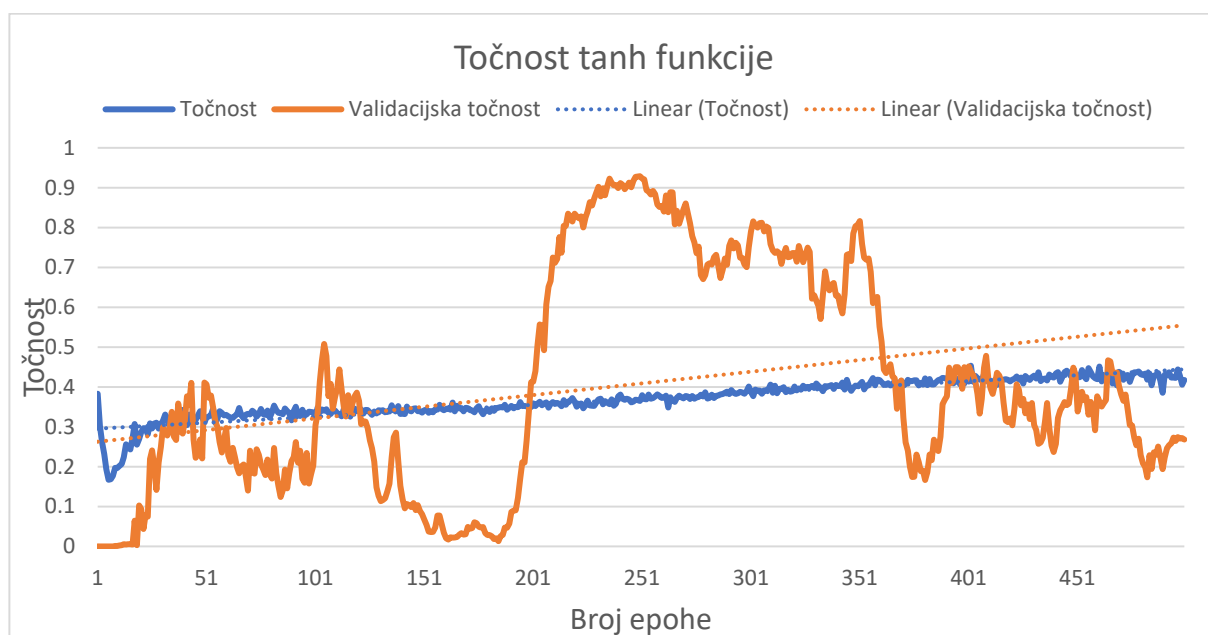
Usporedba gubitka tijekom faze učenja i validacijskog gubitka prikazana je na sljedećem grafu (Slika 5.5):



Slika 5.5: Usporedba gubitaka tangentno-hiperbolične funkcije.

Kao i kod sigmoidalne funkcije, i gubitak tijekom faze učenja i validacijski gubitak pokazuju tendenciju opadanja s tim da tendencija gubitka tijekom faze učenja konstantno ima manju vrijednost od tendencije validacijskog gubitka.

Usporedba točnosti funkcije tijekom faze učenja i validacijske točnosti prikazana je sljedećim grafom (Slika 5.6):



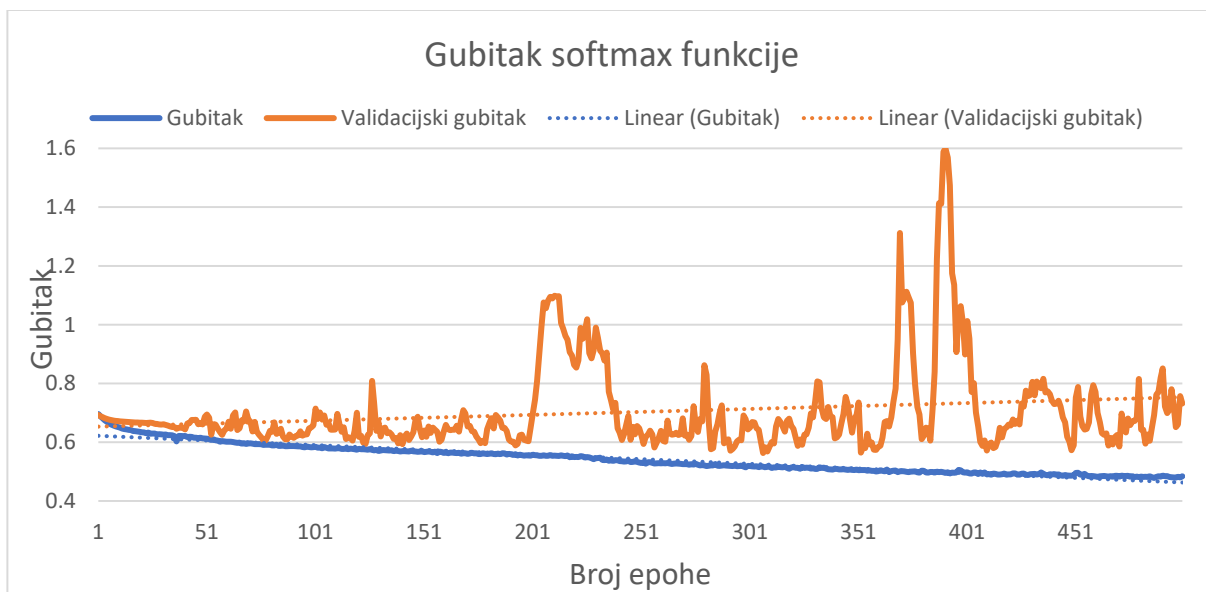
Slika 5.6: Usporedba točnosti tangentno-hiperbolične funkcije.

Kao i u slučaju sigmoidalne funkcije, tijekom 500 epoha validacijska točnost pokazuje strmiju tendenciju rasta od točnosti tijekom faze učenja. Jedan od razloga zašto je tako je taj da i sigmoidalna i tangentno-hiperbolična funkcija pripadaju logističkim sigmoidalnim funkcijama.

5.4. SOFTMAX AKTIVACIJSKA FUNKCIJA

Prosječno vrijeme izvođenja softmax aktivacijske funkcije iznosi 1352,388 sekundi po epohi s najduljim vremenom izvođenja od 2345 sekundi i najkraćim vremenom izvođenja od 1273 sekundi po epohi. Prosječna vrijednost sekundi/koraku iznosi 74,838 sekundi/koraku s najvišom vrijednošću 76 sekunde/koraku i najnižom vrijednošću od 70 sekundi/koraku. Ova funkcija pokazuje izrazito sporije vrijeme izvršavanja od ostalih aktivacijskih funkcija. Prosječni gubitak tijekom faze učenja softmax funkcije iznosi 0,5423836 s minimalnim gubitkom od 0,4783 i maksimalnim gubitkom od 0,6958. Prosječna vrijednost validacijskog gubitka iznosi od 0,703082 s minimalnim gubitkom od 0,5632 i maksimalnim gubitkom od 1,5988. Prosječna točnost softmax funkcije tijekom faze učenja iznosi 0,3536384 s minimalnom točnošću od 0,2379 i maksimalnom točnošću od 0,6779. Prosječna validacijska točnost joj iznosi 0,318353012 s minimalnom točnošću od 0,000014648 i maksimalnom točnošću 1.

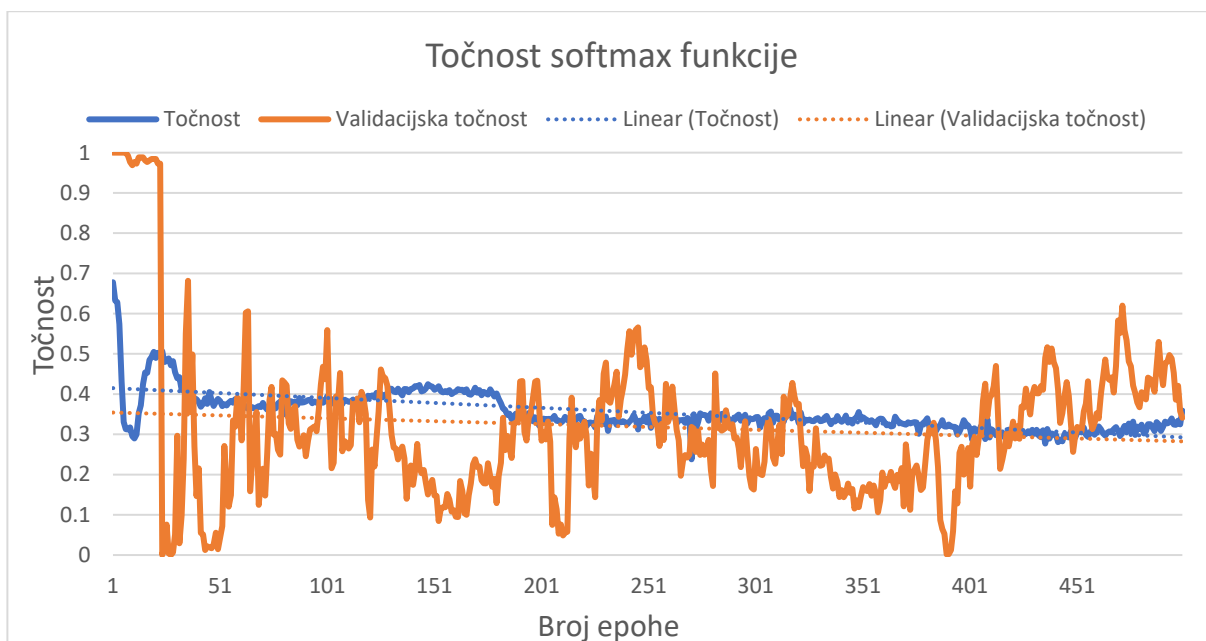
Usporedba gubitka tijekom faze učenja i validacijskog gubitka softmax funkcije prikazana je sljedećim grafom (Slika 5.7):



Slika 5.7: Usporedba gubitaka softmax funkcije.

Za razliku od gubitka tijekom faze učenja, validacijski gubitak softmax funkcije pokazuje linearnu tendenciju rasta dok vrijednosti gubitka tijekom faze učenja opadaju.

Usporedba točnosti funkcije tijekom faze učenja i validacijske točnosti prikazana je sljedećim grafom (Slika 5.8):



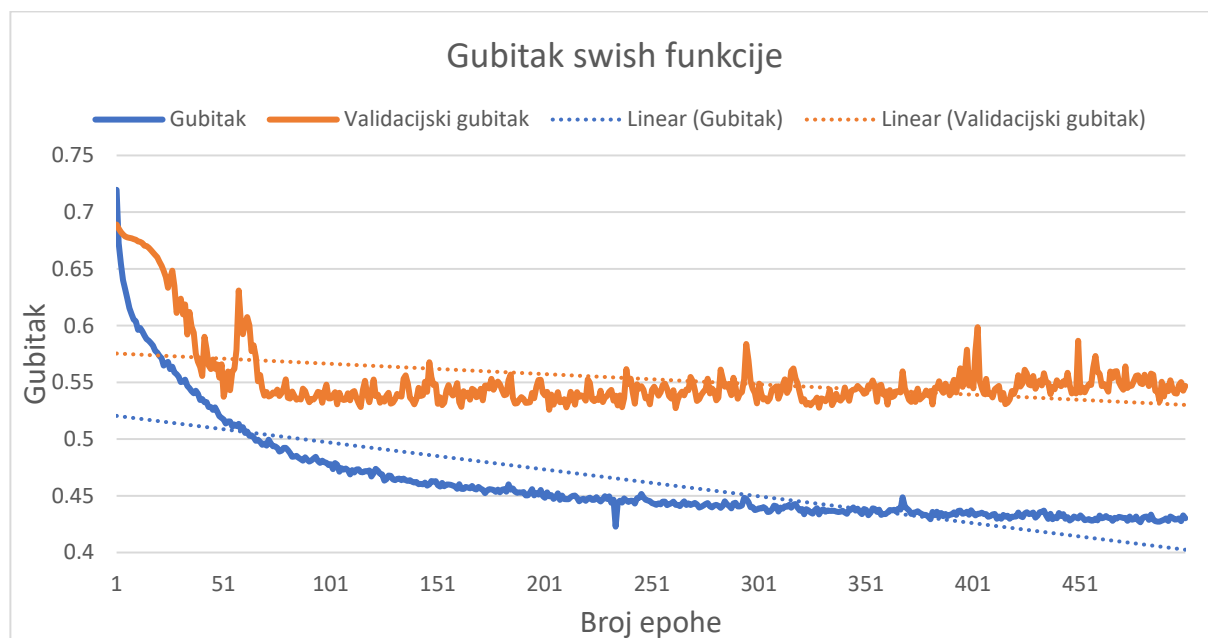
Slika 5.8: Usporedba točnosti softmax funkcije.

Kao i točnost tijekom faze učenja i validacijska točnost pokazuje linearnu tendenciju pada s tim da točnost tijekom faze učenja opada brže od validacijske točnosti.

5.5. SWISH AKTIVACIJSKA FUNKCIJA

Prosječno vrijeme izvođenja swish aktivacijske funkcije iznosi 951,662 sekunde po epohi s najduljim vremenom izvođenja od 972 sekunde i najkraćim vremenom izvođenja od 835 sekundi po epohi. Prosječna vrijednost parametra sekundi/koraku iznosi 52,822 sekunde/koraku s najvišom vrijednošću 54 sekunde/koraku i najnižom vrijednošću od 46 sekundi/koraku. Prosječni gubitak tijekom faze učenja swish funkcije iznosi 0,4614626 s minimalnim gubitkom od 0,4228 i maksimalnim gubitkom od 0,7197. Prosječna vrijednost validacijskog gubitka iznosi 0,55275978 s minimalnim gubitkom od 0,5258 i maksimalnim gubitkom od 0,6891. Prosječna točnost swish funkcije tijekom faze učenja iznosi 0,35967982 s minimalnom točnošću od 0,0386 i maksimalnom točnošću od 0,4847. Prosječna validacijska točnost joj iznosi 0,43563049 s minimalnom točnošću 0 i maksimalnom točnošću 1.

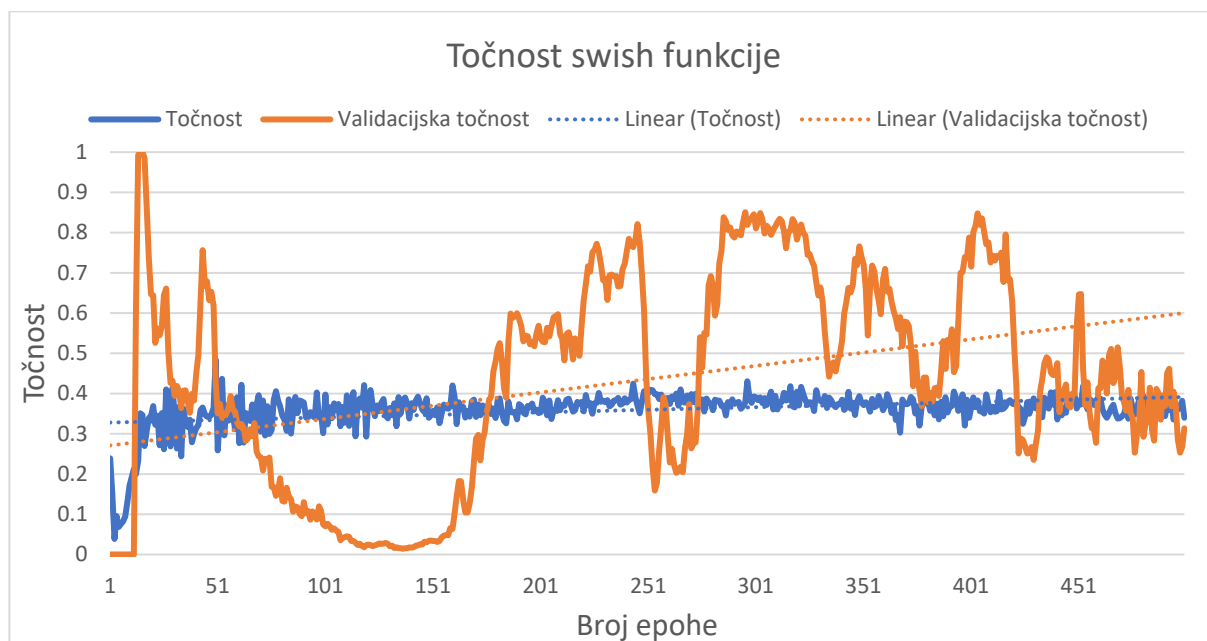
Usporedba gubitka tijekom faze učenja i validacijskog gubitka swish funkcije prikazana je sljedećim grafom (Slika 5.9):



Slika 5.9: Usporedba gubitaka swish funkcije.

Iako oba gubitka pokazuju linearnu tendenciju pada, gubitak tijekom faze učenja pokazuje strmiju tendenciju pada.

Usporedba točnosti funkcije tijekom faze učenja i validacijske točnosti prikazana je sljedećim grafom (Slika 5.10):



Slika 5.10: Usporedba točnosti swish funkcije.

Za razliku od gubitka tijekom faze učenja i validacijskog gubitka, validacijska točnost swish funkcije pokazuje strmiju linearnu tendenciju rasta u odnosu na točnost tijekom faze učenja iako obje točnosti pokazuju tendenciju rasta.

5.6. PARALELNE USPOREDBE AKTIVACIJSKIH FUNKCIJA

U ovom radu aktivacijske funkcije su se usporedile s obzirom na parametre spomenute u prethodnim poglavljima. Ti parametri su: gubitak, validacijski gubitak, točnost, validacijska točnost, vrijeme izvršavanja (po epohi) i parametar sekundi/koraku.

5.6.1. USPOREDBA GUBITAKA

S obzirom na parametar gubitka tijekom faze učenja, aktivacijske funkcije raspoređene su na način prikazan u tablici (Tablica 5.1):

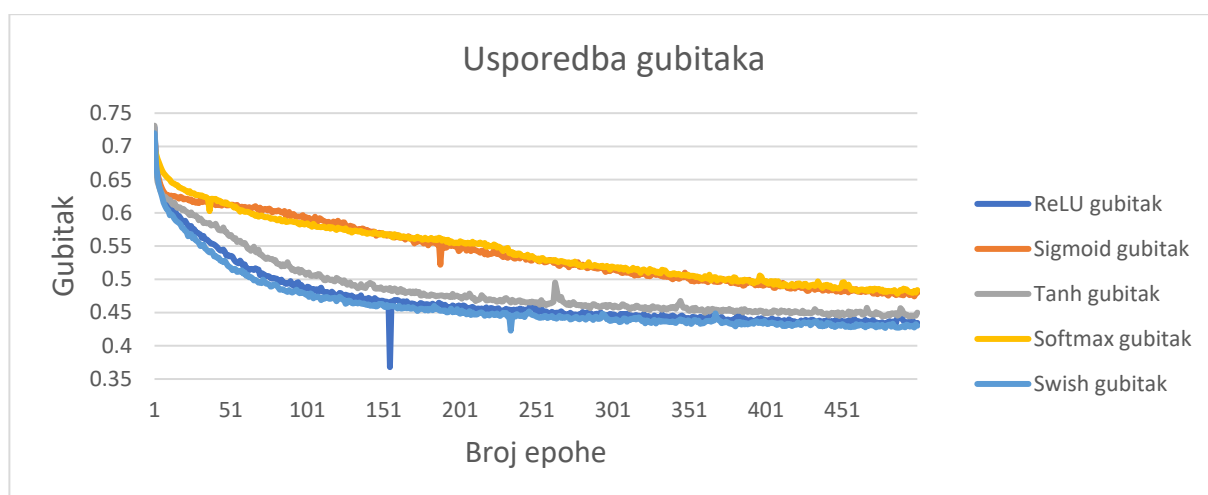
Tablica 5.1: Usporedba aktivacijskih funkcija s obzirom na parametar gubitka tijekom faze učenja

aktivacijska funkcija	prosječna vrijednost gubitka	minimalna vrijednost gubitka	maksimalna vrijednost gubitka
ReLU funkcija	0,4691574	0,3679	0,7285
sigmoidalna funkcija	0,5395474	0,4743	0,7174
tangentno-hiperbolična funkcija	0,4852694	0,4441	0,7312
softmax funkcija	0,5423836	0,4783	0,6958
swish funkcija	0,4614626	0,4228	0,7197

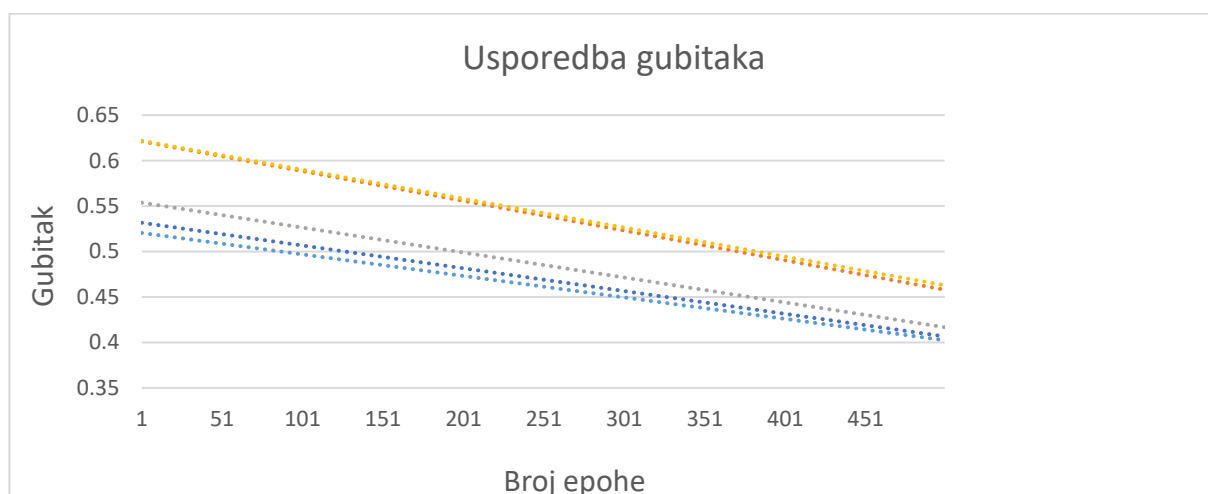
Najnižu prosječnu vrijednost tijekom faze učenja posjeduje swish aktivacijska funkcija. ReLU aktivacijska funkcija ima tek nešto višu prosječnu vrijednost gubitka. Razlika prosječnih vrijednosti gubitka tijekom faze učenja između ReLU i swish funkcije iznosi 0,0076948. Najvišu prosječnu vrijednost gubitka tijekom faze učenja ima softmax funkcija, dok tek nešto manju prosječnu vrijednost od nje ima sigmoidalna funkcija. Razlika prosječnih vrijednosti gubitka tijekom faze učenja između softmax i sigmoidalne funkcije iznosi 0,0028362.

Za razliku od prosječne vrijednosti gubitka, ReLU aktivacijska funkcija je postigla namanju vrijednost gubitka u odnosu na ostale aktivacijske funkcije. Kao i kod prosječne vrijednosti gubitka, softmax funkcija postiže minimalnu vrijednost gubitka koja je veća od minimalnih vrijednosti gubitka ostalih funkcija te opet sigmoidalna funkcija ima tek nešto manju minimalnu vrijednost gubitka. Razlika minimalne vrijednosti gubitka između softmax i sigmoidalne funkcije iznosi 0,004.

Kao što je ReLU funkcija bilježila najmanji gubitak tijekom faze učenja, tako bilježi i najveći gubitak koji iznosi 0,7285. S druge strane, softmax funkcija koja bilježi najveći prosječni gubitak, bilježi i najmanju maksimalnu vrijednost gubitka koja iznosi 0,6958. Najmanja razlika između maksimalnih vrijednosti gubitaka postoji između swish i sigmoidalne funkcije te ta razlika iznosi 0,0023.



Slika 5.11: Usporedba gubitaka aktivacijskih funkcija.



Slika 5.12: Usporedba linearnih tendencija gubitaka aktivacijskih funkcija.

Sve funkcije pokazuju linearnu tendenciju opadanja vrijednosti gubitka tijekom faze učenja (Slika 5.12). Sigmoidalna i softmax funkcija pokazuju veće vrijednosti gubitka u odnosu na ostale aktivacijske funkcije. Tangentno-hiperbolična funkcija se, u odnosu na ostale funkcije, nalazi u sredini. Iako je bolja od sigmoidalne i softmax funkcije, pokazuje veće vrijednosti gubitka od ReLU i swish funkcija. Swish funkcija pokazuje najmanje vrijednosti gubitka s tendencijom prema opadanju iako se po grafu može zaključiti da će u jednom trenutku ReLU funkcija imati manji gubitak od swish funkcije (Slika 5.11).

S obzirom na parametar validacijskog gubitka, aktivacijske funkcije su se podijelile na sljedeći način (Tablica 5.2):

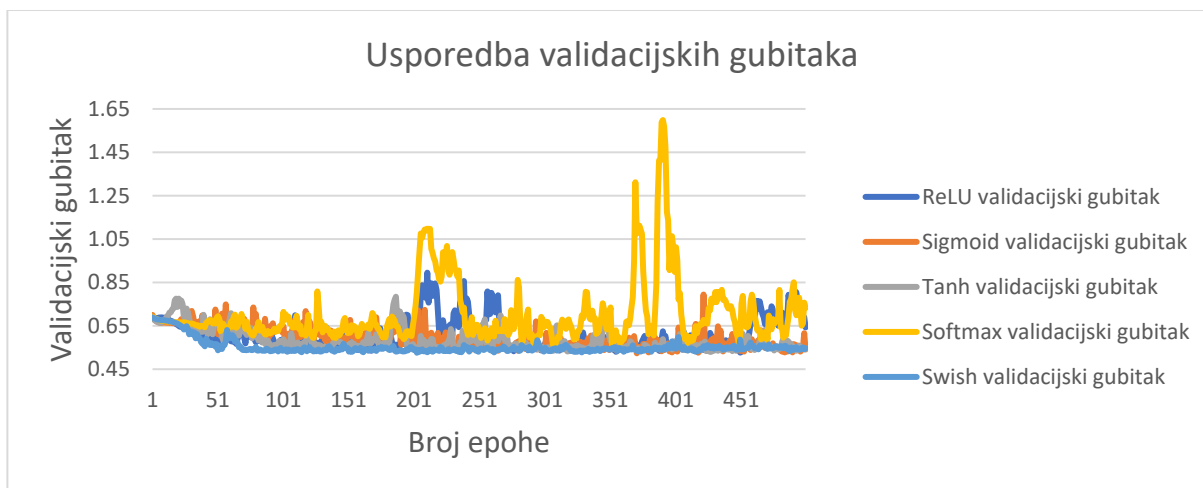
Tablica 5.2: Usporedba aktivacijskih funkcija s obzirom na parametar validacijskog gubitka

aktivacijska funkcija	prosječna vrijednost validacijskog gubitka	minimalna vrijednost validacijskog gubitka	maksimalna vrijednost validacijskog gubitka
ReLU funkcija	0,6105008	0,5265	0,8952
sigmoidalna funkcija	0,59	0,52	0,79
tangentno-hiperbolična funkcija	0,58380252	0,5296	0,7842
softmax funkcija	0,703082	0,5632	1,5988
swish funkcija	0,55275978	0,5258	0,6891

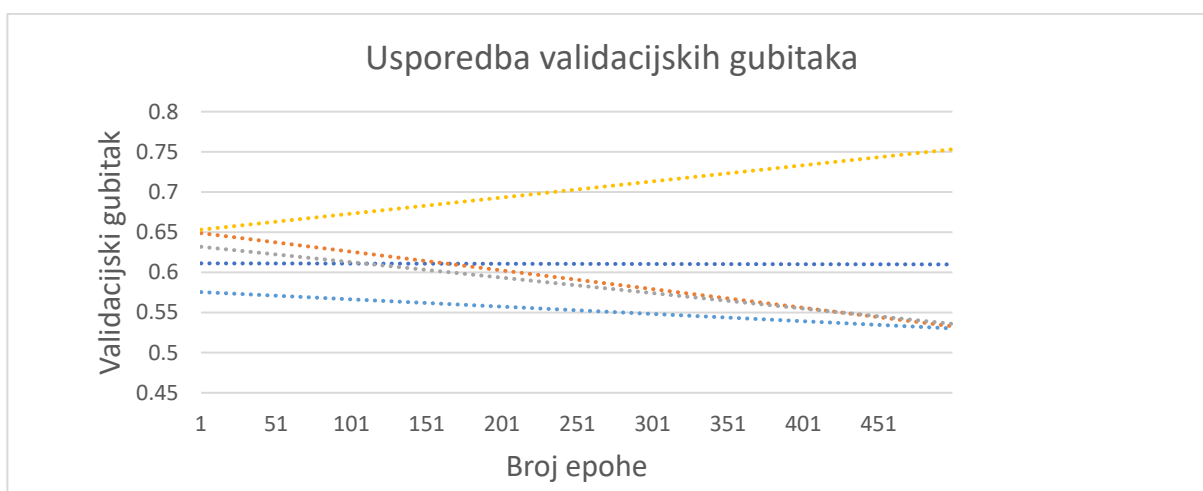
Kao i kod gubitka tijekom faze učenja, swish funkcija bilježi najmanju prosječnu vrijednost validacijskog gubitka. Ovog puta iza nje slijedi tangentno-hiperbolična funkcija od koje tek nešto veću vrijednost validacijskog gubitka bilježi sigmoidalna funkcija. Razlika njihovih validacijskih gubitaka iznosi 0,006197. I ovoga puta softmax funkcija bilježi najveće vrijednosti gubitka u odnosu na ostale funkcije.

Što se tiče minimalne vrijednosti validacijskog gubitka, sigmoidalna funkcija bilježi najmanju vrijednost u odnosu na ostale funkcije dok tek nešto više vrijednosti bilježe swish, ReLU i tangentno-hiperbolična funkcija. Razlika minimalne vrijednosti između swish i sigmoidalne funkcije iznosi 0,0058, između ReLU i swish funkcije iznosi 0,0007, a između tangentno-hiperbolične i ReLU funkcije iznosi 0,0031.

Najveću vrijednost maksimalnog validacijskog gubitka bilježi softmax funkcija, dok najnižu bilježi swish funkcija. Najmanja razlika maksimalnih vrijednosti validacijskog gubitka postoji između sigmoidalne i tangentno-hiperbolične funkcije te ona iznosi 0,0058.



Slika 5.13: Usporedba validacijskih gubitaka aktivacijskih funkcija.



Slika 5.14: Usporedba linearnih tendencija gubitaka aktivacijskih funkcija.

Kao što se može vidjeti na grafu (Slika 5.13), najveće vrijednosti validacijskih gubitaka, i to više puta, bilježi softmax funkcija te jedino ona bilježi linearnu tendenciju rasta (Slika 5.29). Najstrmiju linearnu tendenciju pada bilježe sigmoidalna i tangentno-hiperbolična funkcija te se po grafu tendencija (Slika 5.14) može zaključiti da će u jednom trenutku bilježiti manji gubitak od swish funkcije koja bilježi najmanji validacijski gubitak. Što se tiče ReLU funkcije, njen validacijski gubitak pokazuje vrlo blagu linearnu tendenciju pada.

5.6.2. USPOREDBA TOČNOSTI

S obzirom na parametar točnosti tijekom faze učenja, aktivacijske funkcije raspoređene su na način prikazan u tablici (Tablica 5.3):

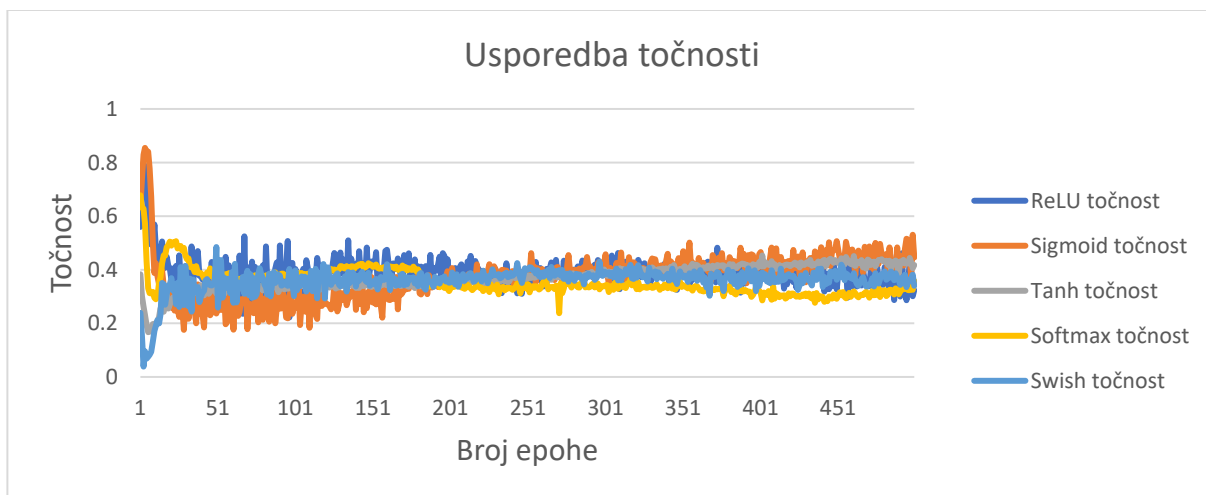
Tablica 5.3: Usporedba aktivacijskih funkcija s obzirom na parametar točnosti tijekom faze učenja

aktivacijska funkcija	prosječna vrijednosti točnosti	minimalna vrijednost točnosti	maksimalna vrijednost točnosti
ReLU funkcija	0,39044742	0,2219	0,7877
sigmoidalna funkcija	0,3734298	0,1757	0,8547
tangentno-hiperbolična funkcija	0,3697506	0,1671	0,4533
softmax funkcija	0,3536384	0,2379	0,6779
swish funkcija	0,35967982	0,0386	0,4847

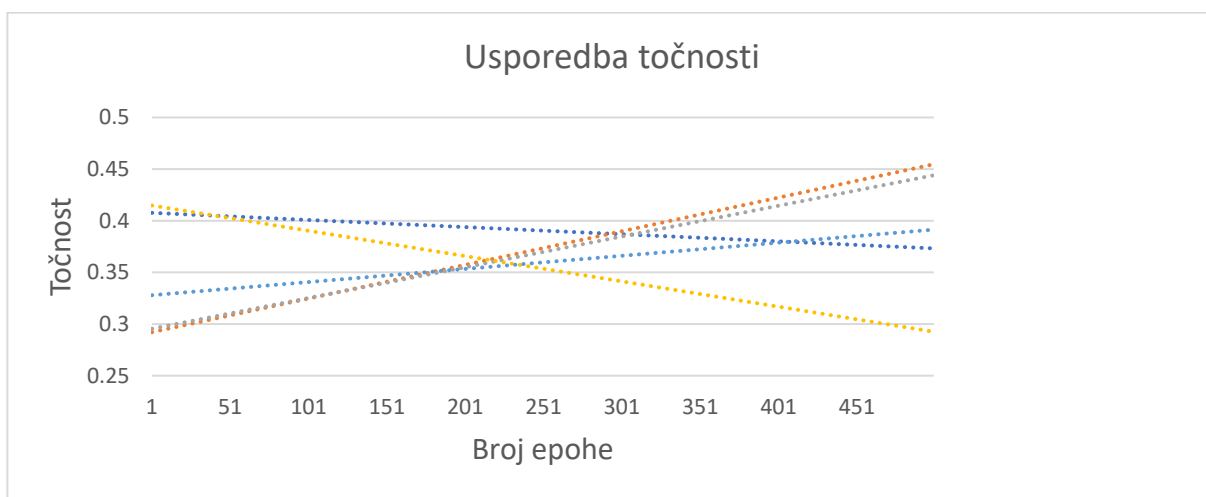
Najveću prosječnu vrijednost točnosti bilježi ReLU funkcija, dok najmanju prosječnu vrijednost bilježi softmax funkcija. U ovom slučaju swish funkcija bilježi tek nešto višu prosječnu vrijednost te razlika prosječne točnosti između swish i softmax funkcije iznosi 0,00604142. Sigmoidalna i tangentno-hiperbolična funkcija također bilježe slične prosječne vrijednosti točnosti te razlika između dviju funkcija iznosi 0,0036792.

Najmanju minimalnu vrijednost točnosti tijekom faze učenja bilježi swish funkcija, dok najveću minimalnu vrijednost bilježi softmax funkcija. Sigmoidalna i tangentno-hiperbolična funkcija ponovno bilježe najsličnije vrijednosti te njihova razlika iznosi 0,0086.

Najveću maksimalnu vrijednost točnosti tijekom faze učenja bilježi sigmoidalna funkcija. U ovom slučaju, najmanju maksimalnu vrijednost točnosti bilježi tangentno-hiperbolična funkcija. Razlika između tih dviju funkcija iznosi čak 0,4014.



Slika 5.15: Usporedba točnosti aktivacijskih funkcija.



Slika 5.16: Usporedba linearnih tendencija točnosti aktivacijskih funkcija.

Iako se na prvom grafu (Slika 5.15) može činiti da funkcije bilježe relativno slične vrijednosti točnosti tijekom faze učenja, ReLU i softmax aktivacijske funkcije pokazuju linearnu tendenciju pada (Slika 5.16) s tim da tendencija softmax funkcije pokazuje strmiji pad od ReLU funkcije. Sigmoidalna i tangentno-hiperbolična funkcija održavaju relativno jednake tendencije rasta. Swish funkcija također pokazuje tendenciju rasta s tim da njena tendencija nema toliko strmi rast u odnosu na sigmoidalnu i tangentno-hiperboličnu funkciju.

S obzirom na parametar validacijske točnosti, aktivacijske funkcije bilježe sljedeće vrijednosti prikazane u tablici (Tablica 5.4):

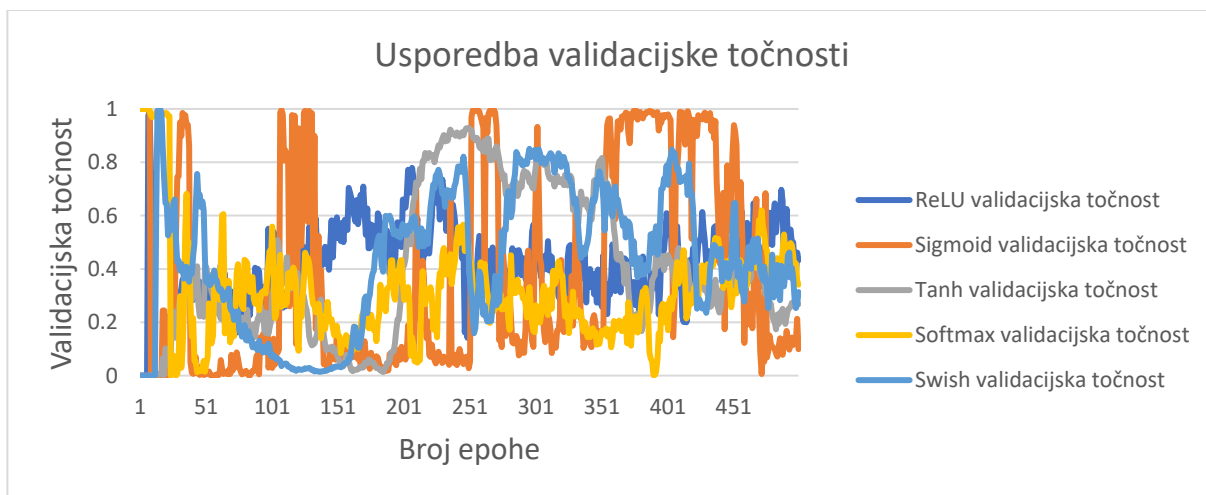
Tablica 5.4: Usporedba aktivacijskih funkcija s obzirom na parametar validacijske točnosti

aktivacijska funkcija	prosječna vrijednost validacijske točnosti	minimalna vrijednost validacijske točnosti	maksimalna vrijednost validacijske točnosti
ReLU funkcija	0,418250494	0,000005722	0,9864
sigmoidalna funkcija	0,384231107	0,000015259	1
tangentno-hiperbolična funkcija	0,40869941	0	0,9288
softmax funkcija	0,318353012	0,000014648	1
swish funkcija	0,43563049	0	1

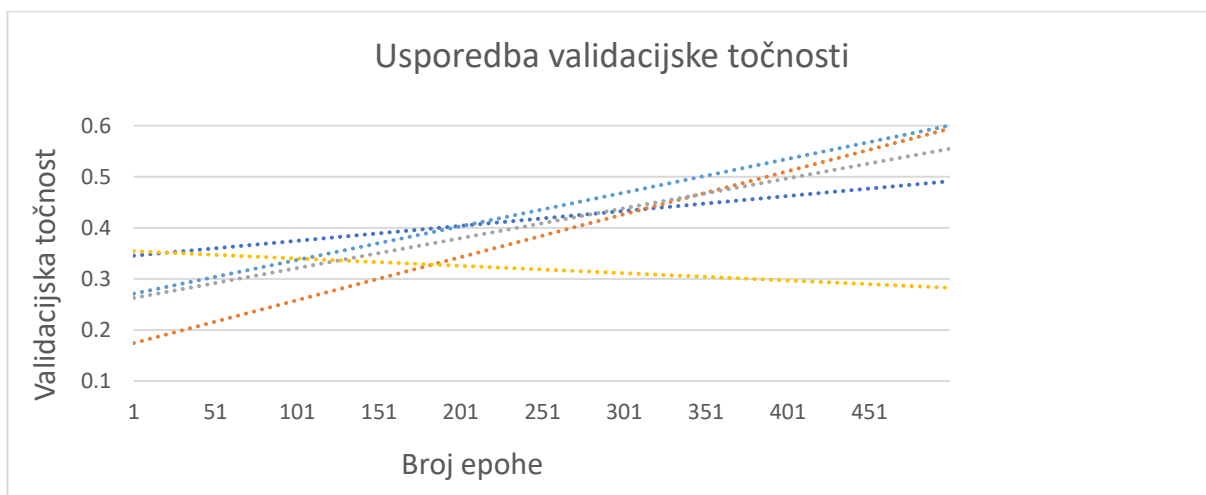
Najveću prosječnu vrijednost validacijske točnosti bilježi swish funkcija. Iza nje redom slijede ReLU funkcija, tangentno-hiperbolična funkcija i naposljetku softmax funkcija. Najmanja razlika validacijskih točnosti se nalazi između ReLU i tangentno-hiperbolične funkcije te ona iznosi 0,009551084.

Najmanju minimalnu vrijednost validacijske točnosti bilježe tangentno-hiperbolična i swish funkcija gdje obje bilježe minimalnu vrijednost validacijske točnosti 0. Najveću minimalnu vrijednost bilježi sigmoidalna funkcija te tek nešto manju minimalnu vrijednost bilježi softmax funkcija.

Najveću maksimalnu vrijednost validacijske točnosti bilježe čak tri funkcije: sigmoidalna, softmax i swish funkcija. Nakon njih slijedi ReLU funkcija udaljena za vrijednost 0,0136. Najmanju maksimalnu vrijednost bilježi tangentno-hiperbolična funkcija.



Slika 5.17: Usporedba validacijskih točnosti aktivacijskih funkcija.



Slika 5.18: Usporedba linearnih tendencija validacijskih točnosti aktivacijskih funkcija.

Kod prikaza validacijske točnosti koji se može vidjeti na prvom grafu (Slika 5.17), samo softmax funkcija održava linearnu tendenciju pada točnosti kao što je prikazano na drugom grafu (Slika 5.18). Ostale funkcije pokazuju tendenciju rasta s tim da sigmoidalna funkcija pokazuje najstrmiju tendenciju rasta. Najblažu tendenciju rasta pokazuje ReLU funkcija koja je u parametru točnosti tijekom faze učenja bilježila pad kao i softmax funkcija.

5.6.3. USPOREDBA VREMENA

Posljednja dva parametra po kojem su uspoređene aktivacijske funkcije su vrijeme izvršavanja pojedine epohe u sekundama i parametar sekundi po koraku. Ti parametri su međusobno

povezani jer rastom vremena izvršavanja po sekundi raste i vrijednost parametra sekundi/koraku.

Rezultati funkcija s obzirom na vrijeme izvršavanja po epohi (u sekundama) prikazani su na sljedećoj tablici (Tablica 5.5):

Tablica 5.5: Usporedba aktivacijskih funkcija s obzirom na parametar vremena izvršavanja po epohi

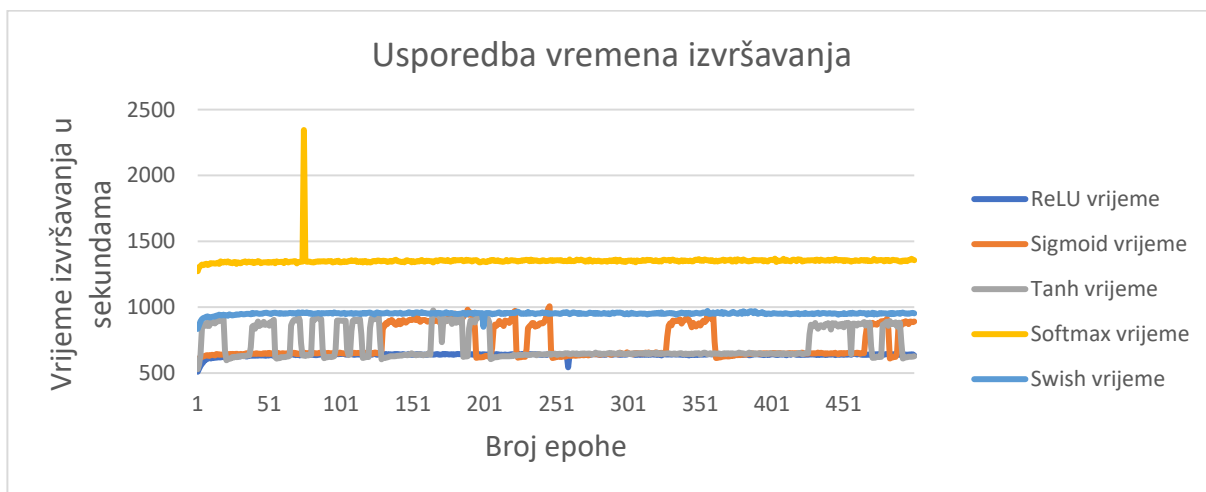
aktivacijska funkcija	prosječno vrijeme izvršavanja	minimalno vrijeme izvršavanja	maksimalno vrijeme izvršavanja
ReLU funkcija	638,348	510	648
sigmoidalna funkcija	723,168	586	1007
tangentno-hiperbolična funkcija	718,478	529	976
softmax funkcija	1352,388	1273	2345
swish funkcija	951,662	835	972

Softmax funkcija bilježi najdulje prosječno vrijeme izvršavanja u odnosu na ostale funkcije dok najkraće prosječno vrijeme bilježi ReLU funkcija. Tangentno-hiperbolična i Sigmoidalna bilježe najslićnija vremena izvršavanja te razlika između sigmoidalne i tangentno-hiperbolične funkcije iznosi 4,69 sekundi.

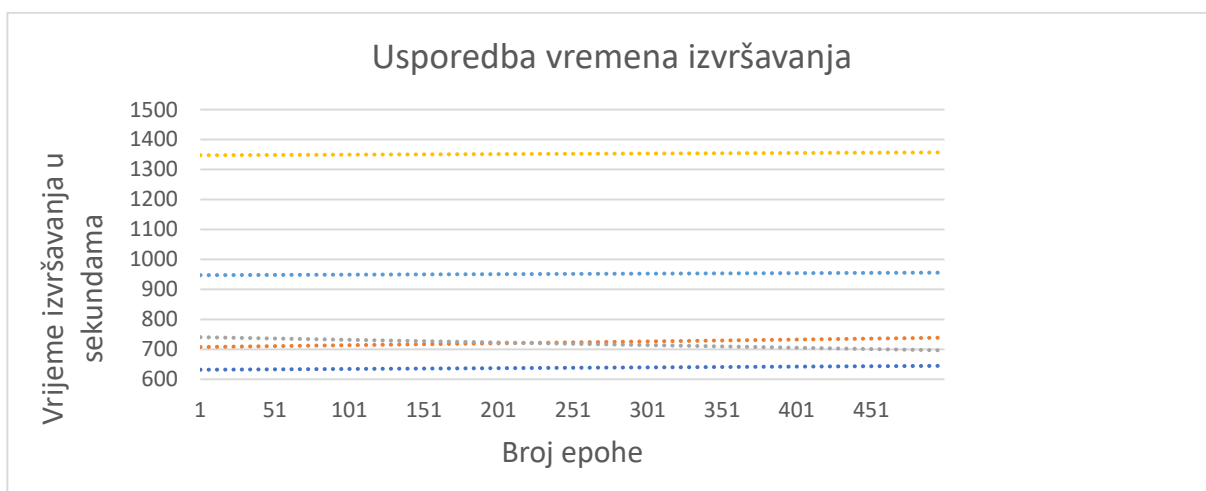
Kao i kod prosječnog vremena izvršavanja ReLU funkcija bilježi najkraće minimalno vrijeme izvršavanja dok softmax funkcija bilježi najdulje minimalno vrijeme izvršavanja. U ovom slučaju najmanja razlika postoji između tangentno-hiperbolične i ReLU funkcije te ona iznosi 19 sekundi.

Opet, najkraće maksimalno vrijeme izvršavanja bilježi ReLU funkcija dok najveće maksimalno vrijeme izvršavanja bilježi softmax funkcija. U ovom slučaju najmanja razlika postoji između tangentno-hiperbolične i swish funkcije te ona iznosi 4 sekunde.

Vremena izvršavanja po epohi i linearne tendencije vremena izvršavanja po epohi vizualno su prikazana na sljedećim grafovima (Slika 5.19 i Slika 5.20):



Slika 5.19: Usporedba vremena izvršavanja aktivacijskih funkcija.



Slika 5.20: Usporedba linearnih tendencija vremena izvršavanja aktivacijskih funkcija.

Sigmoidalna i tangentno-hiperbolična funkcija na prvom grafu (Slika 5.19) pokazuju učestalija odstupanja od zamišljene linije linearne tendencije koja je prikazana na drugom grafu (Slika 5.20). Swish aktivacijska funkcija pokazuje popriličnu monotonost na prvom grafu što se može vidjeti i na drugom grafu gdje pokazuje blagu tendenciju rasta vremena izvršavanja. Softmax i ReLU funkcija na prvom grafu pokazuju samo jedno veliko odstupanje u odnosu na ostala vremena izvršavanja. Kao i swish funkcija i softmax i ReLU funkcije pokazuju blage tendencije rasta vremena izvršavanja. Kao što je već spomenuto, tangentno-hiperbolična i sigmoidalna funkcija bilježe najslabija prosječna vremena izvršavanja s tim da sigmoidalna funkcija

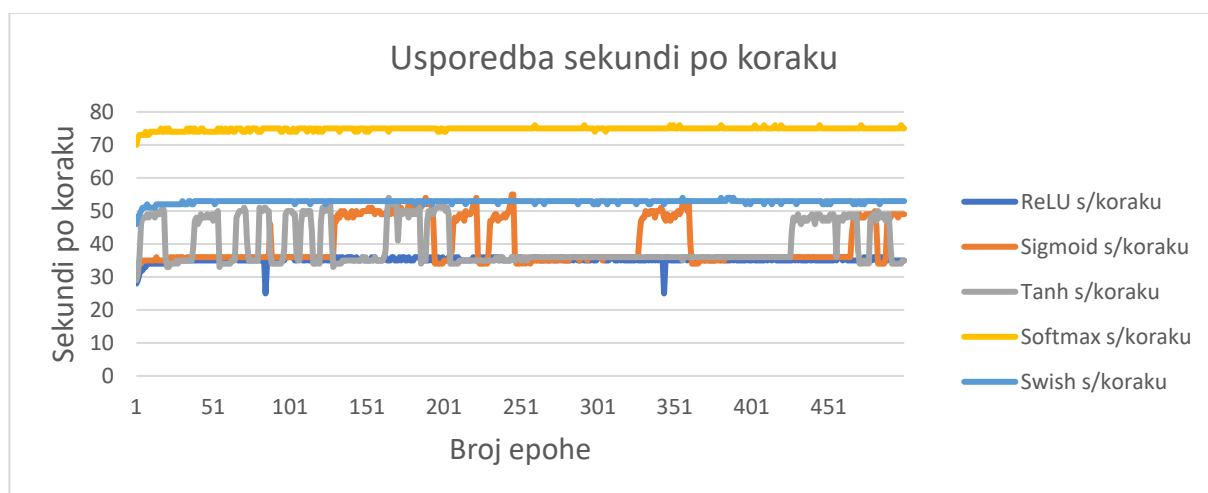
pokazuje tendenciju rasta dok tangentno-hiperbolična funkcija jedina bilježi tendenciju pada vremena izvršavanja po epohi. ReLU funkcija također bilježi tendenciju rasta vremena izvršavanja po epohi.

Što se tiče parametra sekundi/koraku, rezultati izvršavanja prikazani su u sljedećoj tablici (Tablica 5.6):

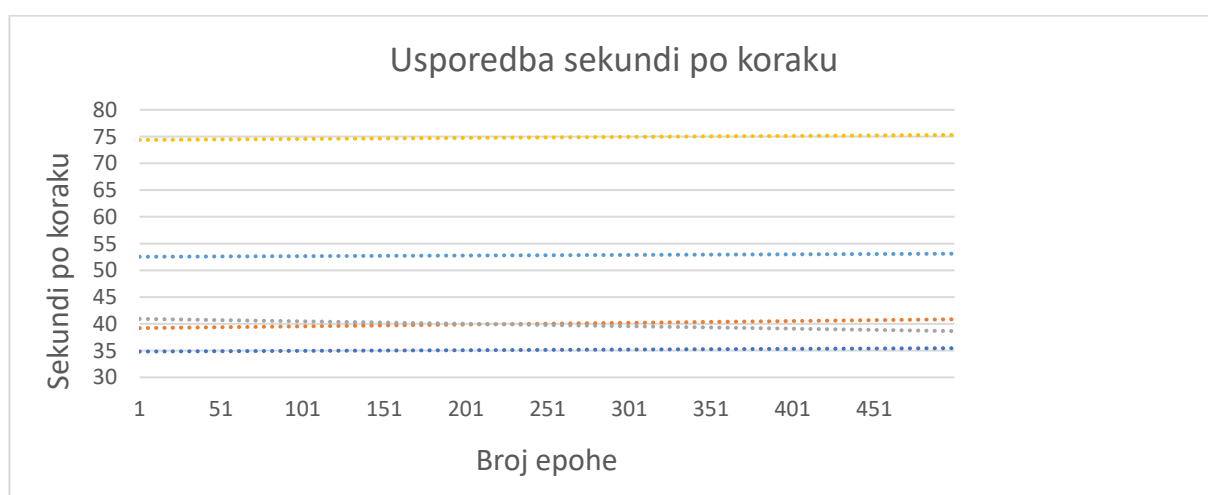
Tablica 5.6: Usporedba aktivacijskih funkcija s obzirom na parametar sekundi/koraku

aktivacijska funkcija	prosječna vrijednost sekundi/koraku	minimalna vrijednost sekundi/koraku	maksimalna vrijednost sekundi/koraku
ReLU funkcija	35,138	25	36
sigmoidalna funkcija	40,03	33	55
tangentno-hiperbolična funkcija	39,786	29	54
softmax funkcija	74,838	70	76
swish funkcija	52,822	46	54

Kod sva tri parametra ReLU aktivacijska funkcija bilježi najmanje vrijednosti dok softmax funkcija bilježi najviše vrijednosti. Kod prosječne i minimalne vrijednosti, najmanju razliku imaju sigmoidalna i tangentno-hiperbolična funkcija. U prvom slučaju ona iznosi 0,244 sekunde/koraku, dok u drugom iznosi 4 sekunde/koraku. Jako mala razlika postoji i kod bilježenja maksimalne vrijednosti gdje ona iznosi 1 sekundu/koraku s tim da u tom slučaju tangentno-hiperbolična i swish funkcija bilježe jednake vrijednosti.



Slika 5.21: Usporedba sekundi/koraku aktivacijskih funkcija.



Slika 5.22: Usporedba linearnih tendencija sekundi/koraku aktivacijskih funkcija.

Kao i kod parametra vremena izvršavanja po epohi sigmoidalna i tangentno-hiperbolična funkcija na prvom grafu (Slika 5.21) pokazuju učestalija odstupanja od zamišljene linije linearne tendencije koja je prikazana na drugom grafu (Slika 5.22). Swish aktivacijska funkcija i dalje pokazuje popriličnu monotonost na prvom grafu što se može vidjeti i na drugom grafu gdje opet pokazuje blagu tendenciju rasta sekundi/koraku. U ovom slučaju softmax funkcija ne bilježi ni jedno veliko odstupanje dok ReLU funkcija na prvom grafu pokazuje tek dva veća odstupanja u odnosu na ostale vrijednosti sekundi/koraku. Kao i swish funkcija, softmax, sigmoidalna i ReLU funkcije pokazuju blage tendencije rasta sekundi/koraku. Kao što je već spomenuto, tangentno-hiperbolična i sigmoidalna funkcija bilježe najbližnje vrijednosti sekundi/koraku s tim da tangentno-hiperbolična funkcija jedina bilježi tendenciju pada vrijednosti sekundi/koraku.

Na kraju, usporedba aktivacijskih funkcija na temelju svih prethodno opisanih parametara može se vidjeti na dolje prikazanoj tablici (Tablica 5.7):

Tablica 5.7: Usporedba aktivacijskih funkcija po svim parametrima.

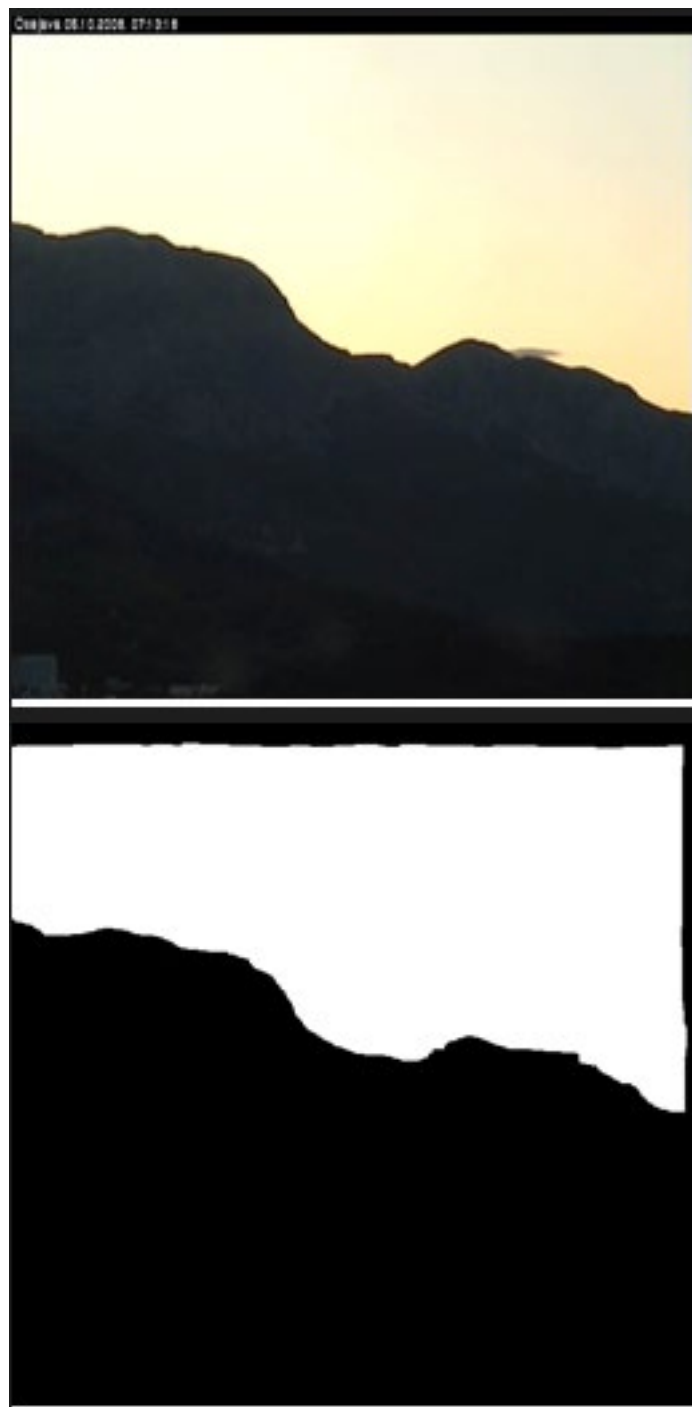
aktivacijska funkcija	ReLU funkcija	sigmoidalna funkcija	tangentno-hiperbolična funkcija	softmax funkcija	swish funkcija
prosječna vrijednost gubitka	0,4691574	0,5395474	0,4852694	0,5423836	0,4614626
minimalna vrijednost gubitka	0,3679	0,4743	0,4441	0,4783	0,4228
maksimalna vrijednost gubitka	0,7285	0,7174	0,7312	0,6958	0,7197
prosječna vrijednost validacijskog gubitka	0,6105008	0,59	0,58380252	0,703082	0,55275978
minimalna vrijednost validacijskog gubitka	0,5265	0,52	0,5296	0,5632	0,5258
maksimalna vrijednost validacijskog gubitka	0,8952	0,79	0,7842	1,5988	0,6891
prosječna vrijednost točnosti	0,39044742	0,3734298	0,3697506	0,3536384	0,35967982

minimalna vrijednost točnosti	0,2219	0,1757	0,1671	0,2379	0,0386
maksimalna vrijednost točnosti	0,7877	0,8547	0,4533	0,6779	0,4847
prosječna vrijednost validacijske točnosti	0,418250494	0,384231107	0,40869941	0,318353012	0,43563049
minimalna vrijednost validacijske točnosti	0,000005722	0,000015259	0	0,000014648	0
maksimalna vrijednost validacijske točnosti	0,9864	1	0,9288	1	1
prosječno vrijeme izvršavanja	638,348	723,168	718,478	1352,388	951,662
minimalno vrijeme izvršavanja	510	586	529	1273	835
maksimalno vrijeme izvršavanja	648	1007	976	2345	972
prosječna vrijednost sekundi/koraku	35, 138	40,03	39,786	74,838	52,822
minimalna vrijednost sekundi/koraku	25	33	29	70	46

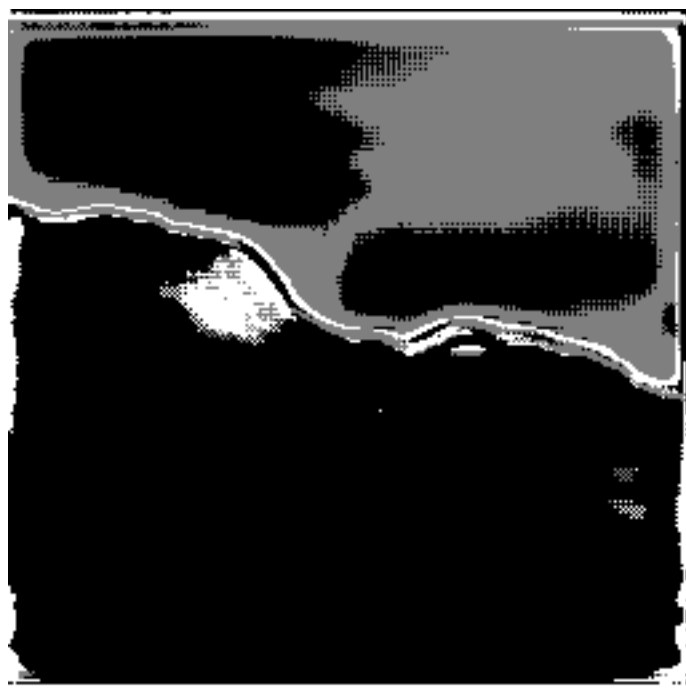
maksimalna vrijednost sekundi/koraku	36	55	54	76	54
---	----	----	----	----	----

5.6.4. VIZUALIZACIJA PREDVIĐANJA

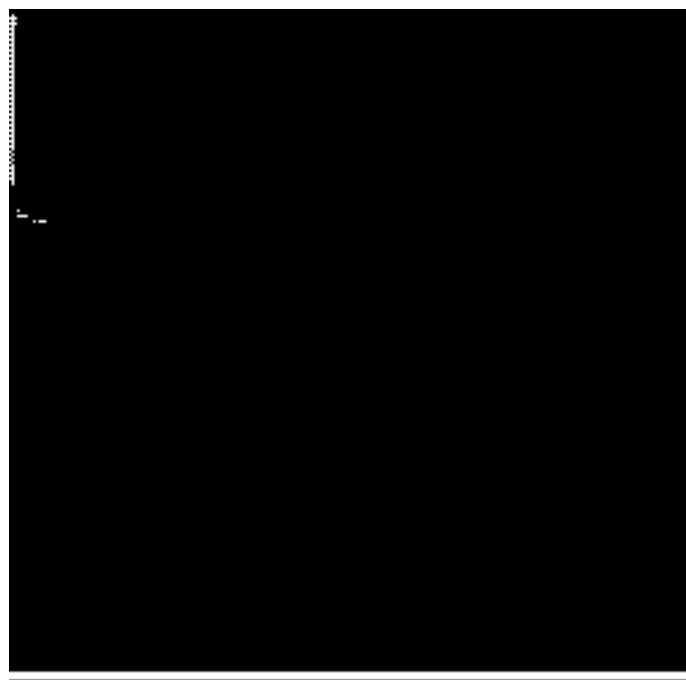
Sljedeća slika (Slika 5.23) prikazuje jednu validacijsku sliku i sliku s odgovarajućim oznakama za piksele te predviđanja svake pojedine mreže u odnosu na to koja se aktivacijska funkcija koristila. U idealnom slučaju, kada je gubitak nula, a točnost 1, predviđanja mreže bila bi jednaka slici s odgovarajućim oznakama.



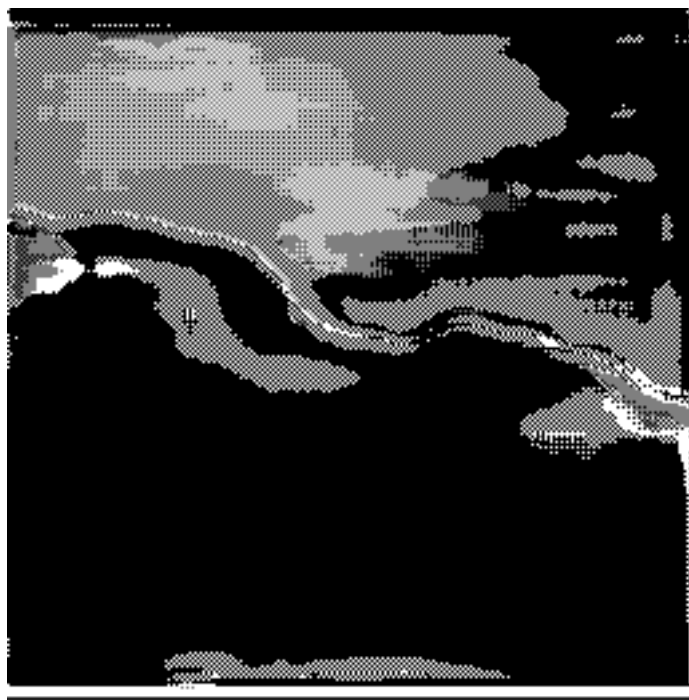
Slika 5.23: slika iz skupa za validaciju (gore) i njena slika s pravim oznakama (dolje).



Slika 5.24: predviđanje mreže s ReLU aktivacijskom funkcijom za danu validacijsku sliku.



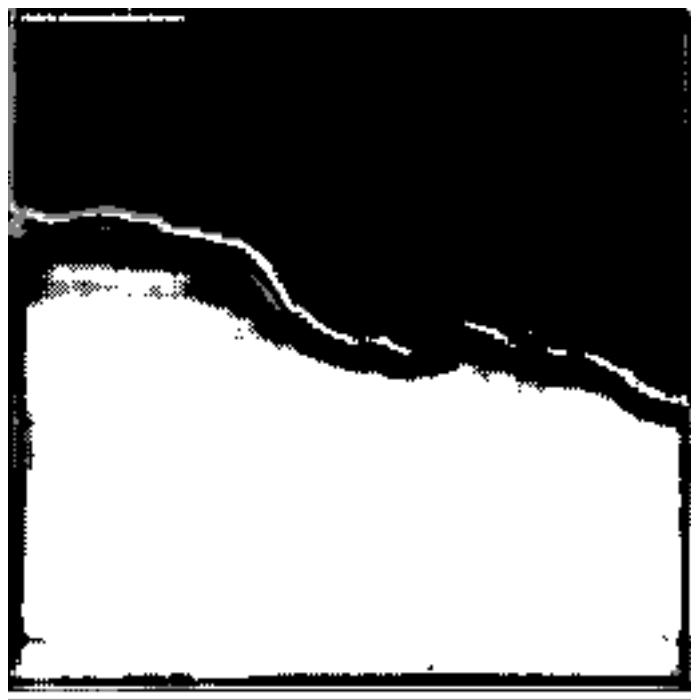
Slika 5.25: predviđanje mreže sa sigmoidalnom aktivacijskom funkcijom za danu validacijsku sliku.



Slika 5.26: predviđanje mreže s tangentno-hiperboličnom aktivacijskom funkcijom za danu validacijsku sliku.



Slika 5.27: predviđanje mreže sa softmax aktivacijskom funkcijom za danu validacijsku sliku.



Slika 5.28: predviđanje mreže sa swish aktivacijskom funkcijom za danu validacijsku sliku.

6. ZAKLJUČAK

Zadatak ovog diplomskog rada bio je proučiti različite aktivacijske funkcije koje se mogu koristiti u konvolucijskim neuralnim mrežama te konstruirati i implementirati jednostavnu konvolucijsku neuralnu mrežu za semantičku klasifikaciju digitalnih slika prirodnog krajolika kako bi se mogli analizirati rezultati segmentacije za različite aktivacijske funkcije.

U radu je ostvaren kratki i općeniti pregled konvolucijskih neuralnih mreža, njihove strukture i načina rada kako bi se dao uvid u to gdje pripadaju aktivacijske funkcije te koja je njihova funkcija i važnost. Nakon toga su matematički opisane aktivacijske funkcije koje su u kasnijem dijelu bile uspoređene na temelju parametara gubitka, validacijskog gubitka, točnosti, validacijske točnosti, vremena izvršavanja i sekundi/koraku. Aktivacijske funkcije koje su se uspoređivale su ReLU funkcija, sigmoidalna funkcija, tangentno-hiperbolična funkcija, softmax funkcija i swish funkcija.

Prije same usporedbe aktivacijskih funkcija dan je opis semantičke segmentacije. U sklopu tog opisa pozornost je skrenuta i na baze slika kao jednog od najvažnijih dijelova računalnog vida. Na kraju je dan pregled dijelova Keras biblioteke koji su se koristili za potrebe konstruiranja i implementacije mreže U-net arhitekture za klasifikaciju slika.

S obzirom na parametar gubitka, ReLU i Swish aktivacijske funkcije bilježe najnižu prosječnu vrijednost gubitka i najnižu minimalnu vrijednost gubitka u odnosu na ostale aktivacijske funkcije, dok najnižu maksimalnu vrijednost gubitka bilježi softmax funkcija koja se obično koristi tek u posljednjem sloju neuronskih mreža.

Swish funkcija također bilježi i najnižu prosječnu vrijednost validacijskog gubitka i najnižu maksimalnu vrijednost validacijskog gubitka u odnosu na ostale funkcije dok ReLU funkcija ima bolji rezultat prosječne vrijednosti validacijskog gubitka samo od softmax funkcije. Što se tiče najmanje minimalne vrijednosti validacijskog gubitka, ne bilježe ju ni ReLU ni swish funkcija, već sigmoidalna funkcija iza koje najmanju minimalnu vrijednost validacijskog gubitka bilježe swish i ReLU funkcija.

Za razliku od gubitka, ReLU funkcija bilježi najvišu prosječnu vrijednost točnosti dok swish funkcija ima bolju prosječnu vrijednost točnosti samo od softmax funkcije. ReLU funkcija bilježi dobre rezultate i kod minimalne vrijednosti točnosti gdje je od nje bolja samo softmax funkcija. Swish funkcija bilježi znatno najnižu vrijednost točnosti u odnosu na ostale funkcije, dok Sigmoidalna funkcija bilježi najvišu maksimalnu vrijednost točnosti te se iza nje nalazi ReLU funkcija.

Kod validacijske točnosti swish i ReLU funkcija bilježe bolje rezultate prosječnih vrijednosti od ostalih funkcija.

ReLU funkcija je tijekom izrade ovog rada potvrdila svoju glavnu karakteristiku, a to je velika brzina izvršavanja u odnosu na ostale aktivacijske funkcije te je zadržala solidne rezultate gubitka, točnosti i validacijske točnosti.

Iako swish funkcija pokazuje druge po redu najlošije rezultate koji se tiču vremena izvršavanja, pokazuje jako dobre rezultate gubitka, validacijskog gubitka i prosječne vrijednosti validacijske točnosti. Ona bilježi najbolje rezultate prosječne vrijednosti validacijskog gubitka i najniže maksimalne vrijednosti validacijskog gubitka te bilježi drugi najbolji rezultat najniže minimalne vrijednosti validacijskog gubitka gdje se nalazi iza sigmoidalne funkcije. Kao što je već spomenuto, tijekom faze učenja su spremene one težine veza za koje mreža bilježi najniže validacijske gubitke.

Softmax funkcija najčešće bilježi najgore rezultate u odnosu na ostale funkcije dok sigmoidalna i tangentno-hiperbolična funkcija često bilježe slične rezultate.

Ovo istraživanje se u budućnosti može proširiti na način da mreža uči tijekom većeg broja epoha kako bi se uvidjela eventualna veća ili čak manja razlika između aktivacijskih funkcija. Nadalje, umjesto ReLU funkcije se mogu odabrati i neke od izvedenica ReLU funkcije koje potencijalno ostvaruju bolje rezultate od osnovne ReLU funkcije. Također se istraživanje može nastaviti tako da se odaberu druge aktivacijske funkcije kako bi se i njihovi rezultati usporedili s rezultatima u ovom radu uspoređenih funkcija.

7. LITERATURA I REFERENCE

1. O'Shea K., Nash R.: „An Introduction to Convolutional Neural Networks“, 2.12.2015., [Internet] [Pristupljeno 9.2.2021.], Dostupno na: <https://arxiv.org/pdf/1511.08458.pdf>
2. Mathworks: „What Is Deep Learning? 3 things you need to know“, [Internet] [Pristupljeno 18.2.2021.], Dostupno na: <https://ch.mathworks.com/discovery/deep-learning.html>
3. Deeplizard.Homepage: [Internet] [Pristupljeno 14.2.2021.], Dostupno na: <https://deeplizard.com>
4. Missinglink: „The Complete Guide to Artificial Neural Networks: Concepts and Models“, [Internet] [Pristupljeno 3.3.2021.] Bilo dostupno na (autori izbrisali): <https://missinglink.ai/guides/neural-network-concepts/complete-guide-artificial-neural-networks/>
5. LeCun Y: „The MNIST database of handwritten digits“, 1998., [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://yann.lecun.com/exdb/mnist/>
6. Slika: <https://machinelearningmastery.com/wp-content/uploads/2019/02/Plot-of-a-Subset-of-Images-from-the-MNIST-Dataset-1024x768.png>, [Internet] [Pristupljeno 14.7.2021.]
7. Missinglink, [Internet] [Pristupljeno 11.2.2021.], Bilo dostupno na (autori izbrisali): <https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-networks-image-classification/>
8. Udofia U.: „Basic Overview of Convolutional Neural Network (CNN) The Principle of the Convolutional Layer, Activating Function, Pooling Layer and Fully-connected Layer“, 13.2.2018., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17>
9. Sharma S.: „Activation Functions in Neural Networks“, 6.9.2017., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
10. Missinglink: „Convolutional neural Network Tutorial Basic Advanced“, [Internet] [Pristupljeno 3.3.2021.], Bilo dostupno na (autori izbrisali): <https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-tutorial-basic-advanced/>

11. Keras: „Layer activation functions Usage of activations Sigmoid function“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://keras.io/api/layers/activations/#sigmoid-function>
12. Chaudhary M.: „Activation Functions: Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax“, 28.8.2020., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://medium.com/@cmukesh8688/activation-functions-sigmoid-tanh-relu-leaky-relu-softmax-50d3778dcea5>
13. Wood T.: „What is the Sigmoid Function?“, 27.9.2020., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://deeptai.org/machine-learning-glossary-and-terms/sigmoid-function>
14. Keras: „Layer activation functions Usage of activations ReLU function“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://keras.io/api/layers/activations/#relu-function>
15. Keras: „Layer activation functions Usage of activations Tanh function“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://keras.io/api/layers/activations/#tanh-function>
16. Paperswithcode: „Tanh Activation“, 2000., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://paperswithcode.com/method/tanh-activation>
17. Liu D.: „A Practical Guide to ReLU“, 30.11.2017., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7>
18. Ye A.: „Swish Activation Function by Google“, 22.10.2017., [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://medium.com/@neuralnets/swish-activation-function-by-google-53e1ea86f820>
19. Keras: „LeakyReLU layer“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: https://keras.io/api/layers/activation_layers/leaky_relu/
20. Keras: „PReLU layer“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: https://keras.io/api/layers/activation_layers/prelu/
21. Keras: „ThresholdedReLU layer“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: https://keras.io/api/layers/activation_layers/threshold_relu/
22. TensorFlow: „TensorFlow Core v2.5.0, Python, tf.nn.crelu“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: https://www.tensorflow.org/api_docs/python/tf/nn/crelu
23. Ye A.: „Swish: Booting ReLU from the Activation Function Throne How Swish Beats ReLU in the Deep Learning Activation Function Competition“, 3.3.2020., [Internet]

- [Pristupljeno 3.3.2021.], Dostupno na: <https://towardsdatascience.com/swish-booting-relu-from-the-activation-function-throne-78f87e5ab6eb>
24. TensorFlow: „TensorFlow Core v2.5.0., Python, tf.keras.activations.swish“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: https://www.tensorflow.org/api_docs/python/tf/keras/activations/swish
 25. Slika: https://miro.medium.com/max/546/1*w2iJqeyJ1Rs-Uc8MwjVmnw.png, [Internet] [Pristupljeno 14.7.2021.]
 26. Keras: „Layer activation functions Usage of activations Softmax function“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://keras.io/api/layers/activations/#softmax-function>
 27. Wood T.: „What is the Softmax Function?“, 17.5.2019. [Internet] [Pristupljeno 3.3.2021.], Dostupno na: <https://deepai.org/machine-learning-glossary-and-terms/softmax-layer>
 28. Slika: <https://d1zx6djv3kblv7.cloudfront.net/wp-content/media/2019/09/Deep-learning-27-i2tutorials.png>, [Internet] [Pristupljeno 14.7.2021.]
 29. Guo Y. et al.: „A Review of Semantic Segmentation Using Deep Neural Networks“, International Journal of Multimedia Information Retrieval, 24.11.2017.
 30. Garcia-Garcia A. et al.: „A Review on Deep Learning Techniques Applied to Semantic Segmentation“, 22.4.2017., arXiv: 1704.06857v1
 31. Long J. et al.: „Fully Convolutional Networks for Semantic Segmentation“, 8.3.2015., UC Berkley, arXiv:1411.4038v2
 32. Tsang S.: „Review: FCN – Fully Convolutional Network (Semantic Segmentation)“, 5.10.2018., [Internet] [Pristupljeno 14.6.2021.], Dostupno na: <https://towardsdatascience.com/review-fcn-semantic-segmentation-eb8c9b50d2d1>
 33. Matan O. et al.: „Multi-Digit Recognition Using a Space Displacement Neural Network“, 2.12.1991., NIPS. 488-95.
 34. Ronnenberg O. et al.: „U-Net: Convolutional Networks for Biomedical Image Segmentation“, 18.5.2015., arXiv: 1505.04597v1
 35. Slika: <https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/u-net-architecture.png>, [Internet] [Pristupljeno 14.6.2021.]
 36. The PASCAL Visual Object Classes Homepage, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://host.robots.ox.ac.uk/pascal/VOC/>
 37. The PASCAL-Context Dataset, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://cs.stanford.edu/~roozbeh/pascal-context/>

38. PASCAL-Part Dataset, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://roozbehm.info/pascal-parts/pascal-parts.html>
39. Hariharan B.: „Semantic Boundaries Dataset and Benchmark“, [Internet] [Pristupljeno 20.6.2021.], Dostupno na: <http://home.bharathh.info/pubs/codes/SBD/download.html>
40. COCO Dataset, Homepage, [Internet] [Pristupljeno 14.7.2021.] Dostupno na: <https://cocodataset.org/#home>
41. The SYNTHIA dataset, Homepage, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://synthia-dataset.net>
42. The Cityscapes Dataset, Homepage, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://www.cityscapes-dataset.com>
43. Brostow et al.: „Motion-based Segmentation and Recognition Dataset“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/>
44. Geiger et al.: „Vision meets Robotics: The KITTI Dataset“, International Journal of Robotics Research (IJRR), 2013., [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://www.cvlibs.net/datasets/kitti/>
45. Prest A. et al.: „Youtube-Objects dataset A large-scale database of object videos from YouTube“, 17.6.2012., [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://data.vision.ee.ethz.ch/cvl/youtube-objects/>
46. Bell S., Upchurch P. et al.: „Material Recognition in the Wild with the Materials in Context Database. Computer Vision and Pattern Recognition (CVPR)“, 2015., [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://opensurfaces.cs.cornell.edu/publications/minc/>
47. DAVIS: „Densely Annotated VIDEO Segmentation In-depth analysis of the state-of-the-art in video object segmentation“, Homepage, [Internet] [Pristupljeno 14.7.2021.] Dostupno na: <https://davischallenge.org>
48. DAGS: „Scene Understanding Datasets“, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <http://dags.stanford.edu/projects/scenedataset.html>
49. Nguyen Q.: „Sift Flow Dataset“, 2018., [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://www.kaggle.com/quanbk/sift-flow-dataset>
50. LabelMe: Homepage, [Internet] [Pristupljeno 14.7.2021.] Dostupno na: <http://labelme.csail.mit.edu/Release3.0/index.php>
51. Keras: Homepage, [Internet] [Pristupljeno 14.7.2021.], Dostupno na: <https://keras.io>

52. Stack overflow, Questions: „What is the difference between SeparableConv2D and Conv2D layers?“, 2.2019., [Internet] [Pristupljeno 15.6.2021.], Dostupno na: <https://stackoverflow.com/questions/54705367/what-is-the-difference-between-separableconv2d-and-conv2d-layers>
53. Keras: „SeparableConv2D layer SeparableConv2D class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/layers/convolution_layers/separable_convolution2d/
54. Keras: „BatchNormalization layer, BatchNormalization class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/layers/normalization_layers/batch_normalization/
55. Keras: „Activation layer Activation class“, [Internet] [Pristupljeno 30.6.], Dostupno na: https://keras.io/api/layers/core_layers/activation/
56. Keras: „MaxPooling2D layer MaxPooling2D class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/layers/pooling_layers/max_pooling2d/
57. Keras: „Conv2DTranspose layer Conv2DTranspose class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/layers/convolution_layers/convolution2d_transpose/
58. Keras: „Dropout layer Dropout class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/layers/regularization_layers/dropout/
59. Keras: „Model training APIs compile method“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/models/model_training_apis
60. Keras: „Adam Adam class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: <https://keras.io/api/optimizers/adam/>
61. Keras: „Probabilistic losses BinaryCrossentropy class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/losses/probabilistic_losses/#categoricalcrossentropy-class
62. Keras: „Accuracy metrics Accuracy class“, [Internet] [Pristupljeno 30.6.2021.], Dostupno na: https://keras.io/api/metrics/accuracy_metrics/#accuracy-class
63. Braović M. et al.: „FESB MLID dataset“, 30.9.2014., [Internet] [Pristupljeno 30.6.2021.], Dostupno na: http://wildfire.fesb.hr/index.php?option=com_content&view=article&id=66&Itemid=76
64. Albumentations: „Homepage“, [Internet] [Pristupljeno 1.4.2021.], Dostupno na: <https://albumentations.ai>

8. POPIS OZNAKA I KRATICE

AMD	<i>advanced micro devices</i>
ANN	<i>artificial neural network</i>
API	<i>application programming interface</i>
CNN	<i>convolutional neural network</i>
COCO	<i>common objects in context</i>
CPU	<i>central processing unit</i>
CReLU	<i>concatenated rectified linear unit</i>
eng	engleski
FCN	<i>fully convolutional network</i>
FESB	Fakultet elektrotehnike, strojarstva i brodogradnje
GB	gigabajt
GPU	<i>graphics processing unit</i>
LReLU	<i>leaky rectified linear unit</i>
MHz	megaherc
MLID	<i>mediterranean landscape image dataset</i>
MNIC	<i>materials in context</i>
MNIST	<i>modified national institute of standards and technology database</i>
NASA	<i>national aeronautics and space administration</i>
ONEIROS	<i>open-ended neuro-electronic intelligent robot operating system</i>
OS	<i>operating system</i>
PReLU	<i>parametric rectified linear unit</i>
RAM	<i>random access memory</i>
ReLU	<i>rectified linear unit</i>
RGB	<i>red, green, blue</i>
SBD	<i>semantic boundaries dataset</i>
Tanh	tangentno-hiperbolična funkcija
TPU	<i>tensor processing unit</i>
TReLU	<i>thresholded rectified linear unit</i>
VOC	<i>visual object classes</i>

SAŽETAK

Glavni zadatak ovog rada jest ostvariti pregled aktivacijskih funkcija u konvolucijskim neuronskim mrežama. To je ostvareno tako što je prvo dan kratki pregled strukture i načina rada konvolucijskih neuronskih mreža kako bi se stvorila podloga za predstavljanje aktivacijskih funkcija koje su se koristile tijekom izrade ovog rada.

Aktivacijske funkcije koje su se koristile su: sigmoidalna funkcija, ReLU funkcija, tangentno-hiperbolična funkcija, softmax i swish funkcija.

Nakon upoznavanja s konvolucijskim mrežama i matematičkim opisom aktivacijskih funkcija, dan je kratki pregled semantičke segmentacije. U sklopu pregleda semantičke segmentacije dan je kratki pregled U-net arhitekture konvolucijske mreže koja se koristila za potrebe ovog rada. Na kraju, u sklopu semantičke segmentacije spomenuti su i podaci kao jedan od najvažnijih dijelova strojnog učenja te su nabrojani najpopularniji skupovi podataka koji se koriste za potrebe treniranja modela semantičke segmentacije. Prije same usporedbe aktivacijskih funkcija, predstavljena je Keras biblioteka s funkcijama, klasama i njihovim argumentima koji su se koristili za potrebe ovog rada.

Aktivacijske funkcije su analizirane tako što je prvo konstruirana i implementirana jednostavna konvolucijska neuronska mreža U-net arhitekture za semantičku klasifikaciju te se implementacija mreže vrtjela na proširenom skupu FESB MLID digitalne baze slika prirodnog krajolika kroz 500 epoha tijekom kojih su spremene težine veza za koje mreža bilježi najmanji validacijski gubitak.

Rezultati izvođenja mreže sa svakom aktivacijskom funkcijom analizirani su na temelju parametara gubitka, validacijskog gubitka, točnosti, validacijske točnosti, vremena izvršavanja te parametra sekundi/koraku.

Dobiveni rezultati su pokazali da ReLU i swish funkcija najčešće bilježe bolje rezultate u odnosu na ostale funkcije s tima da ReLU funkcija bilježi najbolje rezultate u parametrima vremena izvođenja i sekundi/koraku, dok bilježi lošije rezultate kod parametara validacijskog gubitka. S druge strane je pokazano da swish funkcija bilježi druge po redu najgore rezultate

koji se tiču vremena izvršavanja i parametra sekundi/koraku, ali zato bilježi bolje rezultate gubitka, validacijskog gubitka i prosječne vrijednosti validacijske točnosti.

KLJUČNE RIJEČI: semantička klasifikacija, konvolucijska neuralna mreža, aktivacijske funkcije, Keras

SUMMARY

Main task of this thesis is to achieve an overview of activation functions in convolutional neural networks. This was accomplished by first giving a brief overview of convolutional neural networks and mathematical description of activation functions used in convolutional neural networks.

Activation functions used are sigmoid function, ReLU function, tanh function, softmax function and swish function.

After brief overviews of CNNs and mathematical descriptions of activation functions, a brief overview of semantic segmentation. As a part of the review of semantic segmentation, a brief overview of U-net architecture of CNNs is given. Finally, as part of semantic segmentation, data is mentioned as one of the most important parts of machine learning and computer vision and the most popular data sets used for training semantic segmentation model are listed. Before comparing the activation functions, the Keras library is presented along with the functions, classes and their arguments that are used for the purposes of this thesis.

Activation functions are analyzed by first constructing and implementing a simple CNN that is trained on the extended set of FESB mediterranean landscape image dataset over 500 epochs during which only the weights that cause the minimal validation loss are stored.

The obtained results showed that the ReLU and swish functions usually record better results compared to other functions. ReLU function records the best results in the execution time and seconds/step, but therefore records worse results of validation loss. On the other hand, swish function records second worst results in terms of the execution time and seconds/step, but therefore records better results in loss, validation loss and average validation accuracy.

TITLE: A Review of Activation Functions Used in Convolutional Neural Networks

KEY WORDS: semantic classification, convolutional neural network, activation functions, Keras