

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 000

# **Detekcija sigurnosnih atributa prometnica u snimkama**

Ivan Relić

Zagreb, lipanj 2017.

*Umjesto ove stranice umetnite izvornik Vašeg rada.*

*Da bi ste uklonili ovu stranicu obrišite naredbu \izvornik.*



# SADRŽAJ

<b>1. Uvod</b>	<b>1</b>
<b>2. Skupovi podataka i njihovo korištenje</b>	<b>3</b>
2.1. Video snimke projekta FTTS iRAP . . . . .	3
2.2. Ručno označen skup podataka . . . . .	4
2.3. Skup podataka označen konzistentno s projektom FTTS iRAP . . . . .	6
<b>3. Korištene metode, modeli i postupci prilikom oblikovanja sustava</b>	<b>8</b>
3.1. Unakrsna entropija kao funkcija gubitka . . . . .	9
3.2. Gradijentni spust . . . . .	10
3.2.1. Adam optimizator . . . . .	10
3.2.2. Prenaučenost modela . . . . .	12
3.3. Tipični slojevi dubokih modela za primjenu u računalnom vidu . . . . .	13
3.3.1. Potpuno povezani sloj . . . . .	13
3.3.2. Konvolucijski sloj . . . . .	16
3.3.3. Sloj sažimanja . . . . .	18
3.3.4. Sloj grupne normalizacije (engl. <i>batch normalization</i> ) . . . . .	18
3.4. Arhitekture korištene u eksperimentima . . . . .	20
3.4.1. Arhitektura VGG-16 modela . . . . .	20
3.4.2. Arhitektura za klasifikaciju pojedinačnih slika . . . . .	21
3.4.3. Arhitekture za klasifikaciju sekvenci slika . . . . .	22
3.5. Postupak pripreme skupova podataka . . . . .	25
3.5.1. Preprocesiranje slika . . . . .	25
3.5.2. Postupak kreiranja slika označenih konzistentno s projektom FTTS iRAP . . . . .	27
<b>4. Eksperimenti i rezultati</b>	<b>29</b>
4.1. Način učenja i testiranja u eksperimentima . . . . .	29

4.2. Klasifikacija ručno označenih slika na različitim rezolucijama . . . . .	31
4.2.1. Rezultati na rezoluciji 700x280 . . . . .	31
4.2.2. Rezultati na rezoluciji 525x210 . . . . .	33
4.2.3. Rezultati na rezoluciji 350x140 . . . . .	34
4.2.4. Rezultati na rezoluciji 175x70 . . . . .	36
4.3. Klasifikacija pojedinačnih slika označenih konzistentno s projektom FTTS iRAP na originalnoj rezoluciji . . . . .	37
4.4. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći vremensko-prostorno sažimanje . . . . .	39
4.5. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći povratne LSTM ćelije . . . . .	41
4.6. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći vremenski potpuno povezani sloj . . . . .	41
4.7. Analiza rezultata . . . . .	41
<b>5. Zaključak</b>	<b>42</b>
<b>Literatura</b>	<b>43</b>

# 1. Uvod

Računalni vid vrlo je značajno područje umjetne inteligencije koje se bavi razumijevanjem slike, odnosno scene na slici. Vrlo popularna područja računalnog vida danas su klasifikacija (određivanje pripadnosti scene slike jednoj ili više predefiniranih klasa), semantička segmentacija (podjela slike u dijelove i pridjeljivanje klase svakom od tih dijelova), stereoskopska rekonstrukcija (određivanje 3D geometrije scene iz slika do bivenih parom kamera) te mnoga druga. Važnost percepcije i razumijevanja okoline dovela je do toga da je računalni vid vrlo popularno područje umjetne inteligencije koje je u stalnom razvoju zbog svojih mnogih primjena.

Duboko učenje donijelo je veliku revoluciju na mnogim područjima umjetne inteligencije, pa tako i na području računalnog vida. Duboki modeli pomaknuli su marge najbolje ostvarenih rezultata na mnogim problemima. Primjerice, na problemu klasifikacije slika iz skupa podataka ImageNet [6] u jednu od tisuću mogućih klasa, najmanja pogreška ostvarena dubokim modelom manja je od pogreške koju na tom skupu podataka ostvaruju ljudi [9]. Upravo zbog vrlo dobrih rezultata koje duboki modeli ostvaruju, sve više zadatka u kojima ljudi lako grijese i koji su mentalno naporni povjeravaju se na obavljanje računalima.

Dobri rezultati dubokih modela na području računalnog vida svakako otvaraju mnoge primjene računala u praktičnim problemima. Jedan takav problem, kojim se ovaj rad bavi, jest inspekcija kvaliteta cesta. Organizacija iRAP (engl. *International Road Assessment Program*) ima za cilj upravo inspekciju kvaliteta ceste pridruživanjem različitih sigurnosnih atributa dijelovima prometnica. Zadatak se obavlja tako da se na temelju video sekvenci prometnica uz geolokacije pridružene video sekvcencama određuje prisutnost sigurnosnog atributa na tom dijelu prometnice. Na temelju prisutnih atributa moguće je procijeniti sigurnost promatranog dijela prometnice što je vrlo korisno.

Trenutno, pridruživanje atributa dijelovima prometnica obavljaju ljudi. Zadatak je vrlo naporan i skup te su moguće raznorazne greške – greške uzrokovane subjektivnošću, greške uzrokovane nekonzistentnim pridruživanjem različitih osoba, greške

uzrokovane umorom te mnoge druge. Prednost korištenja dubokog modela za taj zadatak je njegova veća brzina, manja cijena i objektivnost u procjeni u odnosu na čovjeka.

Ovaj rad kroz nekoliko faza opisuje implementaciju dubokog modela koji je sposoban detektirati prisutnost sigurnosnog atributa u video snimci. Prvi dio rada opisuje korišteni skup podataka – postupak njegovog prikupljanja te korištenje za učenje dubokog modela. Drugi dio rada opisuje korištene metode i modele koji na temelju učenja na skupu podataka ostvaruju željenu funkciju koju možemo primjeniti na konkretni problem. Završni dio rada opisuje provedene eksperimente i dobivene rezultate prilikom korištenja dubokog modela za problem detekcije sigurnosnih atributa u snimkama.

## 2. Skupovi podataka i njihovo korištenje

Sustav koji rad opisuje temelji se na prepoznavanju prisutnosti atributa na prometnici. Prepoznavanje prisutnosti atributa možemo okarakterizirati kao binarnu klasifikaciju čiji konačni izlaz nam govori je li atribut prisutan u sceni ili nije.

Modeli dubokog učenja danas uspješno rješavaju probleme klasifikacije. Rješavanje takvog problema temelji se na odabiru prikladnog modela uz dovoljan broj označenih podataka. Ključna stvar prilikom rješavanja problema klasifikacije jest izvlačenje znanja iz označenih podataka koje model koristi. Takav postupak nazivamo nadziranim učenjem [8]. Prilikom učenja ustvari se korigiraju slobodni parametri modela koji ostvaruju željenu funkciju preslikavanja.

Skup podataka možemo podijeliti na 3 podskupa koji imaju različite uloge:

- podskup za učenje – podskup koji se koristi za korekciju slobodnih parametara modela
- podskup za validaciju – podskup koji se koristi za korekciju hiperparametara, odnosno za provjeru sposobnosti generalizacije modela
- podskup za testiranje – podskup koji se koristi za konačno ispitivanje performanse naučenog modela

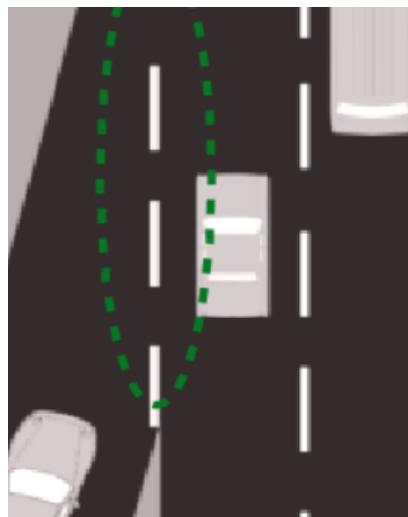
### 2.1. Video snimke projekta FTTS iRAP

Baza video snimaka prometnica sa stranice [1] korištena je kao osnova za kreiranje skupa podataka. Video snimke snimljene su automobilom koji se kretao po engleskim autocestama.

Stranica [1] nudi sučelje u kojem je moguće segmentima prometnica pridruživati ranije spomenute sigurnosne attribute. Standard organizacije iRAP propisuje 78 atributa [4] koje je moguće pridružiti segmentima prometnica. Pojedine video snimke je

moguće preuzeti uz pripadajuće informacije o geolokaciji scena.

Sustav koji ovaj rad opisuje koristi video snimke s navedene stranice za učenje. Konkretno, kako je riječ o binarnoj klasifikaciji, potrebni su nam označeni podaci. Atribut koji sustav prepoznaće jest pripajanje (engl. *merge lane*). Kako je riječ o video sekvencama s engleskih cesta, pripajanja koja se prepoznaju bit će s lijeve strane. Definicija atributa pripajanje prema [3] jest sljedeća – promet koji prilazi sa strane pripaja se preko linije za pripajanje na cestu kojoj se atribut dodjeljuje.



**Slika 2.1:** Simbolički prikaz pripajanja, preuzeto i prerađeno iz [3]

Slika 2.1 simbolički prikazuje što se smatra pripajanjem prilikom pridjeljivanja sigurnosnog atributa pripajanja. Iz video snimaka kreirana su dva tipa skupova podataka:

- ručno označen skup podataka
- skup podataka označen konzistentno s podacima projekta FTTS iRAP

U nastavku je opisan način kreiranja i uporaba pojedinih tipova skupova podataka.

## 2.2. Ručno označen skup podataka

Početna ideja bila je, neovisno o pridruženim atributima pripajanja sa stranice [1], ručno označiti pripajanja.

Dubokim modelima je, kao i ljudima, lakše izlučiti znanje iz podataka ukoliko u njima postoje diskriminativne značajke koje mogu odrediti pripadnost određenoj klasi. Dodjeljivanje oznaka prisutnosti atributa pripajanja prometnoj sceni iz tog se razloga vodilo tom mišlju – scene na kojima su prisutne karakteristične, diskriminativne značajke koje bi mogle odrediti prisutnost atributa pripajanja označene su kao pozitivne,

odnosno scene na kojima je atribut prisutan, dok su ostale označene kao negativne, odnosno scene na kojima atribut nije prisutan.



**Slika 2.2:** Prikaz ručno označene scene na kojoj je atribut pripajanja prisutan, preuzeto s [1]

Slika 2.2 prikazuje ručno označenu pozitivnu sliku. Na slici su vidljive karakteristične bijele trake u uzorku koji bi potencijalno mogao biti diskriminativan prilikom donošenja odluke modela o prisutnosti atributa pripajanja.



**Slika 2.3:** Prikaz ručno označene scene na kojoj atribut pripajanja nije prisutan, preuzeto s [1]

Slika 2.3 prikazuje ručno označenu negativnu sliku. Vidljiv je izostanak karakterističnih bijelih traka na cijeloj sceni.

Ručno označen skup podataka dobiven je obradom 7 videa preuzetih sa stranice [1]. Ukupan broj dobivenih označenih slika s raspodjelom u podskupove je sljedeći:

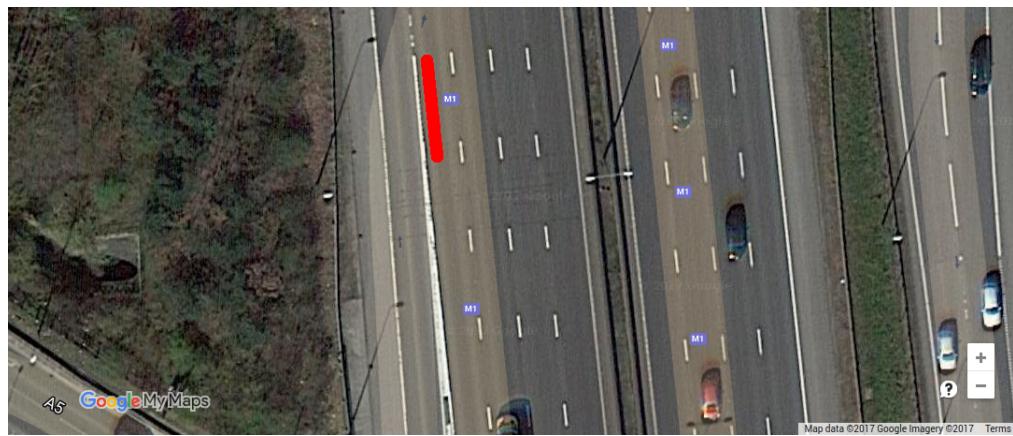
- podskup za učenje – 1796 slika, od toga 898 pozitivnih
- podskup za validaciju – 626 slika, od toga 313 pozitivnih
- podskup za testiranje – 594 slike, od toga 297 pozitivnih

Skup je kreiran na način da se iz svake video sekvene uzorkuju pozitivni primjeri (primjeri na kojima je prisutna diskriminativna značajka atributa pripajanja) te se zatim uzorkuje isti broj negativnih primjera na kojima diskriminativna značajka karakterističnog bijelog uzorka nije prisutna.

## 2.3. Skup podataka označen konzistentno s projektom FTTS iRAP

Kako je glavni cilj rada razviti sustav koji bi mogao automatski određivati prisutnost sigurnosnog atributa na prometnoj sceni, za učenje je potrebno kreirati takav skup podataka koji je označen konzistentno s pravilima koja propisuje iRAP.

U dogovoru s kreatorima stranice [1], izvučeni su podaci geolokacije svih segmenta prometnica na kojima je prisutan atribut pripajanja. Segmenti na kojima se određuje je li sigurnosni atribut prisutan ili nije dugački su 10m. Također, moguće je i otvoriti video snimku koja prikazuje trenutak kada započinje segment s atributom pripajanja.



Slika 2.4: Prikaz satelitske snimke s označenim segmentom atributa pripajanja, preuzeto s [2]

Promotrimo sliku 2.4. Na slici je crvenom linijom označen segment koji je na stranici [1] prema pravilima koje propisuje iRAP označen kao pripajanje. Već sa satelitske snimke je vidljivo kako bi moglo biti problema prilikom učenja zbog nedostatka diskriminativnih značajki koje bi mogle olakšati raspoznavanje pripajanja.



Slika 2.5: Prikaz video snimke s označenim segmentom atributa pripajanja, preuzeto s [1]

Slika 2.5 prikazuje isječak iz video snimke u trenutku kada započinje segment označen kao pripajanje. Iz same slike je vrlo teško odrediti da je u tom trenutku prisutan atribut

pripajanja.

Problem nedostatka informacije možemo riješiti tako da odluku ne temeljimo na jednoj slici nego na sekvenci slika koje su prethodile trenutnoj. Koristeći sekvencu slika koja prethodi slici označenoj kao početak segmenta označenog kao pripajanje, model će biti u mogućnosti lakše donijeti odluku zbog prisutnosti diskriminativne značajke (karakteristične bijele trake u uzorku). Prisutnost diskriminativne značajke u sekvenci koja prethodi početku pripajanja vidljiva je i iz satelitske snimke. Skup podataka označen konzistentno s projektom FTTS iRAP iz tog razloga ne sadrži statične, pojedinačne slike nego sekvence slika.

Skup podataka označen konzistentno s projektom FTTS iRAP dobiven je obradom 100 videa preuzetih sa stranice [1]. Ukupan broj dobivenih označenih sekvenci s raspodjelom u podskupove je sljedeći:

- podskup za učenje – 7554 sekvenci, od toga 3777 pozitivnih
- podskup za validaciju – 1720 sekvenci, od toga 860 pozitivnih
- podskup za testiranje – 1642 sekvenci, od toga 821 pozitivnih

Skup je kreiran na način da se iz svake video sekvence uzorkuju pozitivni primjeri na sljedeći način:

- odrede se rasponi sličica video snimke koji su prema stranici [1] označeni kao pozitivni (sadrže atribut pripajanja)
- za svaku sličicu označenu kao pozitivnu uzorkuje se i nekoliko sličica unatrag koje će zajedno činiti jednu pozitivnu sekvencu

Na sličan način uzorkuje se i jednak broj negativnih primjera za one sličice za koje je prema stranici [1] označeno da ne sadrže atribut pripajanja. Cjelokupni postupak kreiranja skupa podataka uz konkretnе podatke o duljini sekvence bit će dan u nastavku rada.

### 3. Korištene metode, modeli i postupci prilikom oblikovanja sustava

Sustav koji ovaj rad opisuje temelji se na algoritmima dubokog učenja. Zadatak koji je potrebno riješiti jest binarna klasifikacija, odnosno određivanje je li atribut pripajanja prisutan u prometnoj sceni. Model koji rješava zadatak tada možemo prikazati sljedećom funkcijom:

$$\mathbf{y} = f^*(\mathbf{x}; \phi), \quad (3.1)$$

gdje je  $f^*$  funkcija koju želimo naučiti koristeći skup podataka. Argumenti funkcije  $f^*$  su ulazni vektor podataka  $\mathbf{x}$  te skup slobodnih parametara,  $\phi$ . Vrijednost funkcije  $f^*$ , odnosno  $\mathbf{y}$  predstavlja upravo željenu informaciju koja nas zanima – vjerojatnost da je atribut pripajanja prisutan, odnosno vjerojatnost da atribut pripajanja nije prisutan na prometnoj sceni koja je predstavljena kao ulazni vektor podataka  $\mathbf{x}$ .

Modeli dubokog učenja prepostavljaju kompozitnu strukturu ulaznih podataka – npr. čovjek je građen od glave i tijela, glava je građena od očiju, usta, nosa i ušiju, oči su građene od rožnice, zjenice i šarenice, i sl. Iz tog razloga se nad ulaznim podacima obavljaju višestruke linearne i nelinearne operacije [19]. Funkciju  $f^*$  stoga možemo prikazati i na sljedeći način:  $f^*(\mathbf{x}; \phi) = f^{(n)}(f^{(n-1)}(\dots f^{(2)}(f^{(1)}(\mathbf{x}, \phi), \phi), \phi), \phi)$  [8]. Pojedine funkcije  $f^{(i)}$  predstavljaju slojeve dubokog modela koje obavljaju različite operacije.

Ocjenu prikladnosti funkcije  $f^*$  sa slobodnim parametrima  $\phi$  možemo mjeriti funkcijom gubitka. Funkcija gubitka nam govori koliko dobro naša funkcija sa slobodnim parametrima aproksimira stvarna preslikavanja koja su definirana na skupu podataka za učenje.

Učenje modela, odnosno optimizacija, tada se svodi na pretraživanje za onim skupom slobodnih parametara koji će minimizirati funkciju gubitka nad skupom podataka za učenje. Optimizacijski postupak koji se koristi je gradijentni spust nad funkcijom gubitka koju deriviramo po slobodnim parametrima modela.

Nastavak rada opisuje korištenu funkciju gubitka, optimizacijsku metodu za učenje modela te tipične slojeve dubokih modela koji se koriste za primjenu u računalnom vidu. Opisane su i konkretne arhitekture koje su se koristile prilikom provođenja eksperimenata te detaljni postupci kojima su generirani skupovi podataka.

### 3.1. Unakrsna entropija kao funkcija gubitka

Unakrsna entropija kao funkcija gubitka izvodi se kao negativni logaritam izglednosti modela nad podacima. Učenjem želimo maksimizirati izglednost parametara nad nekim podacima za učenje, odnosno želimo parametre modela za koje je najizglednije da minimiziraju razliku između funkcije  $f$  koju aproksimiramo i aproksimirane funkcije modela  $f^*$ .

Prepostavke za izvod unakrsne entropije kao funkcije gubitka su sljedeće:

- zadatak koji rješavamo jest zadatak klasifikacije slike u jednu od  $k$  različitih klasa
- izlazni vektor je dimenzionalnosti  $k$  i označava međusobno zavisne vjerojatnosti da ulazni podatak pripada svakoj od  $k$  klasa
- podaci za učenje su izvučeni iz iste razdiobe i međusobno su nezavisni

Izvedimo unakrsnu entropiju kao funkciju gubitka iz logaritma izglednosti modela nad podacima:

$$\begin{aligned}
 C &= -\ln \mathcal{L}(\phi, \mathcal{D}) = -\ln(p_{model}(\mathbf{x}, \mathbf{y} | \mathcal{D}, \phi)) \\
 &= -\ln \prod_{\mathbf{x} \in \mathcal{D}} \prod_{j=1}^k (p_{model}(\mathbf{y}_j | \mathbf{x}, \mathcal{D}, \phi)) p_{model}(\mathbf{x}) \\
 &= -\sum_{\mathbf{x} \in \mathcal{D}} \sum_{j=1}^k \ln(f_j^*(\mathbf{x})^{f_j(\mathbf{x})} (1 - f_j^*(\mathbf{x}))^{(1-f_j(\mathbf{x}))}) \\
 &= -\sum_{\mathbf{x} \in \mathcal{D}} \sum_{j=1}^k (\ln f_j^*(\mathbf{x})^{f_j(\mathbf{x})} + \ln(1 - f_j^*(\mathbf{x}))^{(1-f_j(\mathbf{x}))}) \\
 &= -\sum_{\mathbf{x} \in \mathcal{D}} \sum_{j=1}^k (f_j(\mathbf{x}) \ln f_j^*(\mathbf{x}) + (1 - f_j(\mathbf{x})) \ln(1 - f_j^*(\mathbf{x}))) \tag{3.2}
 \end{aligned}$$

gdje je  $k$  dimenzionalnost izlaznog vektora,  $f_j(\mathbf{x})$  je stvarni izlaz (0 ili 1), a  $f_j^*(\mathbf{x})$  je vrijednost  $j$ -te komponente izlaznog vektora za primjer  $\mathbf{x}$  iz skupa podataka za učenje. Vjerojatnost  $p_{model}(\mathbf{y} | \mathbf{x}, \mathcal{D}, \phi)$  predstavlja vjerojatnost označke primjera. Kako su izlazi modela iz intervala  $[0, 1]$ , navedenu vjerojatnost možemo zapisati kao  $f_j^*(\mathbf{x})^{f_j(\mathbf{x})} (1 - f_j^*(\mathbf{x}))^{1-f_j(\mathbf{x})}$ .

$f_j^*(\mathbf{x}))^{(1-f_j(\mathbf{x}))}$ . Član  $p_{model}(\mathbf{x})$  možemo slobodno zanemariti prilikom definiranja funkcije gubitka jer ne ovisi o vrijednostima parametara  $\phi$  [13].

## 3.2. Gradijentni spust

Izračun nove vrijednosti parametra  $\phi$  funkcije  $y$  čija se vrijednost minimizira definiran je kao:

$$\phi_{n+1} = \phi_n - \psi \frac{\partial y}{\partial \phi}, \quad (3.3)$$

gdje je  $\psi$  pozitivna, uobičajeno mala konstanta koja kontrolira faktor korekcije parametra.

U slučaju učenja dubokog modela, izraz 3.3 možemo zapisati kao:

$$\phi_{n+1} = \phi_n - \eta \frac{\partial C}{\partial \phi}, \quad (3.4)$$

gdje je  $\phi$  skup slobodnih parametara modela,  $\eta$  je parametar algoritma učenja koji nazivamo stopa učenja (engl. *learning rate*) kojim definiramo koliko će se po iznosu mijenjati parametri prilikom učenja. Funkcija koju optimiramo je funkcija gubitka,  $C$ .

Postoje različite varijante osnovnog algoritma za korekciju parametara modela. Korekciju parametara moguće je vršiti akumulirajući gradijent parametara po svim primjerima iz skupa za učenje, gdje govorimo o stvarnom gradijentu nad funkcijom gubitka,  $C$ . Akumulirani gradijent se zatim koristi za korekciju parametara prema izrazu 3.4. U tom slučaju govorimo o klasičnom gradijentnom spustu.

Varijanta u kojoj se gradijent izračunat u svakom primjeru odmah koristi za korekciju parametara naziva se stohastički gradijentni spust. Stohastičnost varijante očituje se u tome što ustvari ne računamo stvarni gradijent funkcije gubitka  $C$ , nego funkcije  $C_x$  koja definira gubitak nad jednim određenim primjerom  $x$  iz skupa za učenje. Stohastički gradijentni spust pokazuje bolju otpornost na zaglavljivanje u lokalnim optimumima, ali i brže učenje zbog češće korekcije parametara.

Varijanta u kojoj se gradijent akumulira nad određenim brojem primjera i zatim se koristi za korekciju parametara naziva se grupni (engl. *batched*) gradijentni spust [13].

### 3.2.1. Adam optimizator

Za potrebe učenja modela korištenog u ovom radu korišten je Adam optimizator. Adam optimizator je modificirana varijanta klasičnog algoritma gradijentnog spusta koja je robustnija i daje bolje rezultate [11].

Adam optimizator, za razliku od klasičnog algoritma gradijentnog spusta kombinira dvije napredne tehnike prilikom korekcije parametara. Uvode se metode momenta i adaptivnog pomaka što pospješuje učenje dubokih modela.

Metoda momenta pomaže ubrzanju učenja na način da prilikom korekcije parametara modela u obzir uzima i iznos prethodne korekcije — samo ime i analogija dolaze iz područja fizike pa metodu možemo poistovjetiti s tromostom tijela. Korekcija parametara se, u odnosu na izraz 3.4, definira na sljedeći način:

$$\mathbf{v}_{n+1} = \alpha \mathbf{v}_n - \eta \frac{\partial C}{\partial \phi} \quad (3.5)$$

$$\phi_{n+1} = \phi_n + \mathbf{v}_{n+1} \quad (3.6)$$

Parametar  $\alpha$  određuje jačinu momenta, odnosno utjecaj prethodne korekcije na trenutnu korekciju i njegov iznos bi u ranijim fazama učenja trebao biti niži, a u kasnijima viši jer tada želimo što usmjereniju pretragu. Osim ubrzanja učenja, momentum pospješuje i otpornost na zaglavljivanje u lokalnim optimumima jer je moguće “preskociti” lokalni optimum u koji bismo inače zaglavili slijepo prateći samo iznos gradijenta [8] [13].

Metoda adaptivnog pomaka pretpostavlja da je stabilnost gradijenta ovisna o svakoj komponenti parametara koji se optimiraju pa je smisleno koristiti zasebne stope učenja za svaku od njih i automatski ih adaptirati kroz postupak učenja. Konačan efekt jest da će se komponente parametara čije su parcijalne derivacije po funkciji gubitka vrlo visokih vrijednosti korigirati sa značajno manjim stopama učenja, a komponente parametara čije su parcijalne derivacije po funkciji gubitka relativno niskih vrijednosti, korigirat će se većim stopama učenja [8] [13].

Adam optimizator kombinira ove dvije metode i optimiranje parametara definira se na sljedeći način:

$$\text{inicijalne vrijednosti :} \quad (3.7)$$

$$\mathbf{m}_0 = \vec{0}$$

$$\mathbf{v}_0 = \vec{0}$$

$$\text{osvježavanje parametara :} \quad (3.8)$$

$$\mathbf{g}_t = \frac{\partial C}{\partial \phi_t}$$

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t$$

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2$$

$$\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1 - \beta_1^t}$$

$$\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1 - \beta_2^t}$$

$$\phi_{t+1} = \phi_t - \eta \frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon},$$

gdje je  $t$  vremenski korak koji se inkrementira sa svakom korekcijom parametara,  $\eta$  je stopa učenja,  $\beta_1$  i  $\beta_2$  su faktori korišteni za eksponencijalni pomični prosjek (engl. *exponential moving average*). Preporučene vrijednosti su  $\beta_1 = 0.9$  i  $\beta_2 = 0.999$  [11]. Parametar  $\epsilon$  služi za izbjegavanje dijeljenja s nulom i postavlja se na vrlo malu vrijednost (npr.  $10^{-8}$ ).

### 3.2.2. Prenaučenost modela

Slojevi u dubokim modelima često imaju vrlo velik broj parametara što može dovesti do problema prenaučenosti. Vrlo česta regularizacijska metoda koja nastoji sprječiti prenaučenost jest regularizacija metodom smanjenja težina (engl. *weight decay*). Regularizacija metodom smanjenja težina mijenja funkciju gubitka koja se optimira tako da uvodi član ovisan o normi parametara, i to na sljedeći način:

$$C^* = C + \lambda p(\phi), \quad (3.9)$$

gdje je  $C^*$  nova funkcija gubitka koju optimiramo,  $C$  je originalna funkcija gubitka,  $p(\phi)$  predstavlja funkciju norme nad parametrima  $\phi$ , a  $\lambda$  je regularizacijski faktor. Faktor  $\lambda$  određuje utjecaj regularizacije – što je on veći, regularizacija je jača i dobivamo jednostavnije modele koji nisu skloni prenaučenosti.

Motivacija za korištenje regularizacije metodom smanjenja težina leži u tome da će funkcije neuronskih mreža koje imaju niže vrijednosti težina biti zaglađenije, odnosno imat će nižu varijancu što je poželjno za bolju generalizaciju. Obično se prilikom regularizacije parametri koji modeliraju pomak ne regulariziraju jer je na njih dovedena konstantna vrijednost ulaza pa samim time utječu samo na pomak unutar prostora [13].

Često korištene funkcije norme  $p(\phi)$  za vektor slobodnih parametara dimenzionalnosti  $n$  jesu:

**L1 norma**  $p(\phi) = \sum_{i=1}^n |\phi_i|$  — L1 (engl. *Lasso*) regularizacija

**L2 norma**  $p(\phi) = \sqrt{\sum_{i=1}^n \phi_i^2}$  — L2 (engl. *Ridge*) regularizacija

### 3.3. Tipični slojevi dubokih modela za primjenu u računalnom vidu

Duboki modeli koji se koriste za primjenu u računalnom vidu tipično su građeni od nekoliko različitih slojeva koji imaju različite uloge. Slojevi obavljaju različite operacije nad podacima i tako ostvaruju željenu konačnu funkcionalnost koja je potrebna da se obavi neki zadatak.

#### 3.3.1. Potpuno povezani sloj

Operacija koju obavlja potpuno povezani sloj definirana je na sljedeći način:

$$\mathbf{y} = g(\mathbf{W}\mathbf{x}), \quad (3.10)$$

gdje je  $\mathbf{x}$  vektor ulaznih podataka,  $g$  je aktivacijska funkcija, a  $\mathbf{y}$  je vektor izlaznih podataka. Vektor ulaznih podataka obično se proširuje na oblik  $\mathbf{x} = \begin{bmatrix} 1 & x_1 & x_2 & \dots & x_n \end{bmatrix}^T$ . Vrijednost  $x_1$  se fiksno postavlja na vrijednost 1 iz razloga što želimo dozvoliti da model modelira pomak parametrima prvog stupca matrice  $\mathbf{W}$ . Primjetimo kako će upravo dimenzionalnost matrice  $\mathbf{W}$  određivati dimenzionalnost vektora izlaznih podataka.

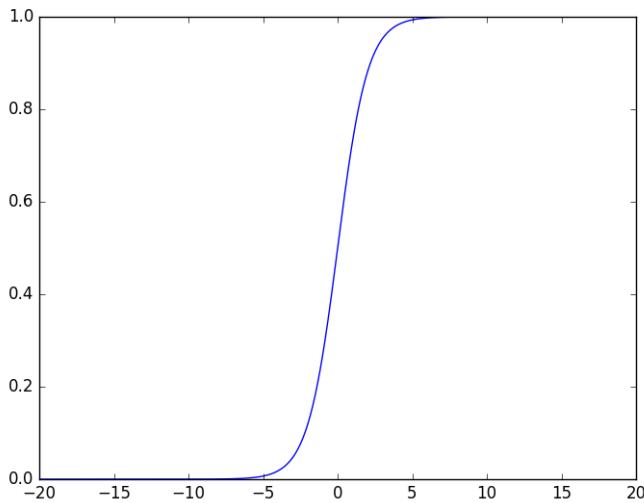
Aktivacijska funkcija  $g$  djeluje po elementima vektora dobivenog operacijom množenja matrice  $\mathbf{W}$  i vektora ulaznih podataka  $\mathbf{x}$ . Tipično se koriste nelinearne aktivacijske funkcije kako bismo dobili veću ekspresivnost dubokog modela. Postoje različite aktivacijske funkcije koje se u praksi koriste. U nastavku su opisane neke od njih.

#### Sigmoidalna aktivacijska funkcija

Sigmoidalna aktivacijska funkcija i njena derivacija definirane su kao:

$$g_j^{(i)}(\mathbf{x}) = \frac{1}{1 + e^{-\alpha x_j}}, \quad (3.11)$$

$$\frac{dg^{(i)}(\mathbf{x})}{d\mathbf{x}} = \alpha g^{(i)}(\mathbf{x})(1 - g^{(i)}(\mathbf{x})) \quad (3.12)$$



**Slika 3.1:** Graf sigmoidalne aktivacijske funkcije

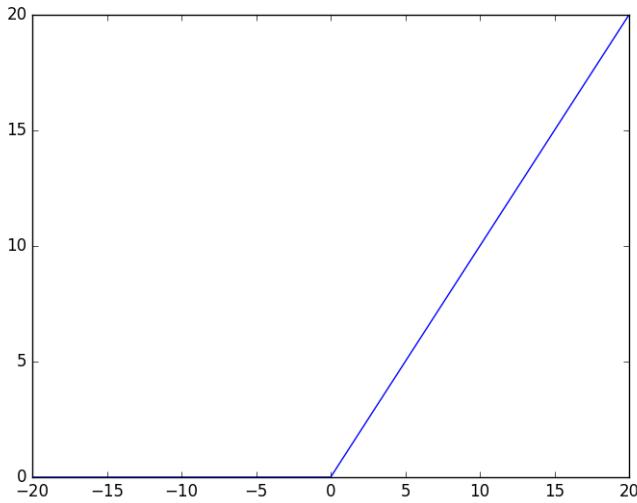
Sigmoidalna aktivacijska funkcija preslikava ulaznu vrijednost na vrijednost iz intervala  $\langle 0, 1 \rangle$ . Sigmoidalna aktivacijska funkcija je derivabilna na cijeloj domeni i njenim korištenjem dobivamo nelinearne modele. Slika 3.1 prikazuje graf sigmoidalne aktivacijske funkcije. Iz njega je vidljivo kako će gradijent za većinu vrijednosti sigmoidalne funkcije biti vrlo niska zbog toga što je vrijednost sigmoidalne funkcije gotovo na cijeloj domeni u zasićenju (engl. *saturating function*) — asimptotski se približava vrijednosti 0 ili 1. Kako je ranije navedeno, učenje modela obavljamo koristeći gradijentni spust funkcije gubitka po parametrima modela. Uglavnom nizak iznos gradijenta utječe na to da će već u zadnjem, izlaznom sloju korekcije parametara mreže biti vrlo male. Kako izračun korekcija propagiramo unatrag, slojevi koji su sve dalje od završnog sloja imat će sve manje iznose korekcija što će uvelike utjecati na brzinu i kvalitetu učenja koje će se drastično smanjiti. Sigmoidalne aktivacijske funkcije se u praksi zato ne koriste u dubokim modelima [13].

### Po dijelovima linearna aktivacijska funkcija

Po dijelovima linearna aktivacijska funkcija, ReLU (engl. *Rectified Linear Unit*) i njena derivacija definirane su kao:

$$g_j^{(i)}(\mathbf{x}) = \begin{cases} 0, & \text{ako je } x_j < 0 \\ x_j, & \text{ako je } x_j \geq 0 \end{cases}, \quad (3.13)$$

$$\frac{dg_j^{(i)}(\mathbf{x})}{d\mathbf{x}} = \begin{cases} 0, & \text{ako je } x_j < 0 \\ 1, & \text{ako je } x_j \geq 0 \end{cases} \quad (3.14)$$



**Slika 3.2:** Graf po dijelovima linearne aktivacijske funkcija

Po dijelovima linearna aktivacijska funkcija predstavlja funkciju praga na vrijednosti 0 ispod koje izlazne vrijednosti nisu propuštenе. Po dijelovima linearna aktivacijska funkcija je često korištena u dubokim modelima zbog zanimljivih svojstava koja pokazuje prilikom učenja takvih struktura. Vrijednost gradijenta je kod po dijelovima linearne aktivacijske funkcije konstantna za polovicu cijele njene domene. U drugoj polovici iznosi 0, što ponekad također može biti problem jer može dovesti do mrtvih dijelova slojeva kojima se parametri nikako ne korigiraju. Problem mrtvih dijelova slojeva može se riješiti korištenjem tzv. leaky ReLU prijenosne funkcije koja dozvoljava mali iznos gradijenta kada je vrijednost aktivacije 0 [8] [13].

### Normalizirajuća eksponencijalna aktivacijska funkcija

Normalizirajuća eksponencijalna aktivacijska funkcija (engl. *softmax*) i njena derivacija definirane su kao:

$$g_j^{(i)}(\mathbf{x}) = \frac{e^{x_j}}{\sum_{k=1}^n e^{x_k}}, \quad (3.15)$$

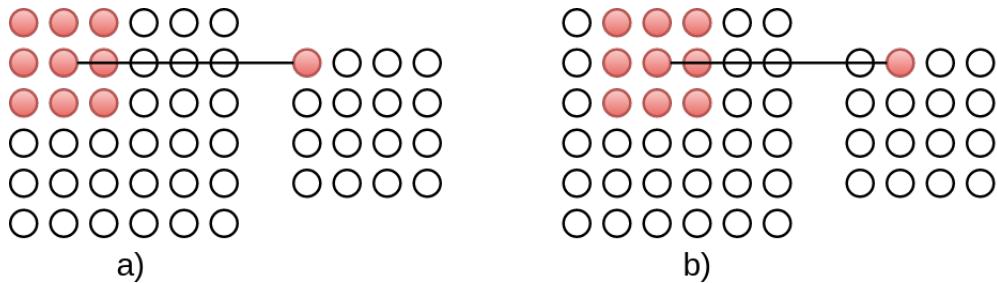
$$\frac{dg_j^{(i)}(\mathbf{x})}{\mathbf{x}} = \begin{cases} g_j^{(i)}(\mathbf{x})(1 - g_j^{(i)}(\mathbf{x})), & \text{ako je } j = k \\ -g_j^{(i)}(\mathbf{x})g_k^{(i)}(\mathbf{x}), & \text{ako je } j \neq k \end{cases} \quad (3.16)$$

Normalizirajuća eksponencijalna aktivacijska funkcija je generalizacija sigmoidalne funkcije u smislu da proširuje sigmoidalnu funkciju namijenjenu binarnoj klasifikaciji na klasifikaciju s  $n$  klasa.

Normalizirajuća eksponencijalna aktivacijska funkcija preslikava  $n$  - dimenzionalni vektor realnih vrijednosti u  $n$  - dimenzionalni vektor realnih vrijednosti koje se zbrajaju u vrijednost 1. Takav vektor realnih vrijednosti možemo promatrati kao vjerojatnosnu razdiobu i upravo iz tog razloga se normalizirajuća eksponencijalna aktivacijska funkcija često koristi u posljednjem sloju dubokih modela za probleme klasifikacije – daje nam vjerojatnost s kojom ulazni uzorak pripada svakoj od  $n$  mogućih klasa, slično kao što je sigmoidalna aktivacijska funkcija korištena kod binarne klasifikacije logističkom regresijom [13].

### 3.3.2. Konvolucijski sloj

Konvolucijski slojevi koriste se za stvaranje mapa značajki uporabom dvodimenzionalne konvolucije jezgre s podacima. Za definiciju konvolucije potrebno je prvo definirati veličinu jezgre (širinu i visinu), vrijednosti parametara jezgre te pomak jezgre po širini odnosno po visini prilikom izračuna.



**Slika 3.3:** Dva koraka konvolucije s dvodimenzionalnim podacima

Slika 3.3 prikazuje dva koraka konvolucije podataka s jezgrom. Slika a) prikazuje prvi korak konvolucije, a slika b) drugi korak. U oba koraka veličina jezgre je  $3 \times 3$ , a korak iznosi 1. Korak konvolucije (engl. *stride*) određuje pomak prozora po širini, odnosno visini ulazne mape značajki.

Općenito, vrijednost jednog elementa mape značajki u sloju  $l$  dobivene konvolucijom nad jednom ulaznom mapom značajki iz sloja  $l - 1$  definiramo kao:

$$x_{i,j}^l = \sum_{m,n=0}^{m=K_w, n=K_h} x_{i \cdot s_w + m, j \cdot s_h + n}^{l-1} w_{m,n}^l, \quad (3.17)$$

gdje je  $x_{i,j}^l$  vrijednost mape značajki u sloju  $l$ , za redak  $i$  i stupac  $j$ .  $K_w$  je širina jezgre,  $K_h$  je visina jezgre, a  $w_{m,n}^l$  vrijednost parametara jezgre za redak  $m$  i stupac  $n$ . Vrijednosti  $s_w$  i  $s_h$  predstavljaju pomak po širini, odnosno visini.

Izraz 3.17 opisuje izračun vrijednosti jednog elementa mape značajke koji je linearna kombinacija vrijednosti samo jedne mape značajki prethodnog sloja. U konvolucijskim arhitekturama se općenito koriste mape značajki koje su povezane s većim brojem mapa značajki iz prethodnog sloja. Za svaku od mapa značajki s kojom je povezana postoji po jedna jezgra koja se koristi za izračun vrijednosti. Primjerice, jedna mapa značajki ulaznog sloja može biti povezana s 3 mapa značajki - crvenom, plavom i zelenom komponentom slike u boji. Za svaku od komponenti boja će mapa značajki tada imati drugčiju jezgru.

Nadalje, izračun vrijednosti mapa značajki se često generalizira kao i kod potpuno povezanog sloja pa se dodaje dodatan parametar kao vrijednost praga te se koriste aktivacijske funkcije. Generalizirani izraz za izračun vrijednosti jednog elementa mape značajki u sloju  $n$  dobivene konvolucijom odgovarajućih jezgri sa svakom od mapa značajki iz sloja  $n - 1$  s kojima je povezana definiramo kao:

$$x_{k',i,j}^l = \sigma \left( \sum_{k \in F^{l-1}} \sum_{m,n=0}^{m=K_w^k, n=K_h^k} x_{k,i \cdot s_w + m, j \cdot s_h + n}^{l-1} w_{k,m,n}^l + b^l \right), \quad (3.18)$$

gdje je  $x_{k,i,j}^l$  vrijednost mape značajki  $k$  u sloju  $l$ , za redak  $i$  i stupac  $j$ .  $F^{l-1}$  je skup mapa značajki iz sloja  $l - 1$  s kojima je mapa značajki iz sloja  $l$  povezana.  $K_w^k$  je širina jezgre, a  $K_h^k$  je visina jezgre za mapu značajki  $k$  iz sloja  $l - 1$ .  $w_{k,m,n}^l$  je vrijednost parametara jezgre za mapu značajki  $k$  iz sloja  $l - 1$  za redak  $m$  i stupac  $n$ . Vrijednosti  $s_w$  i  $s_h$  predstavljaju pomak po širini, odnosno visini. Vrijednost  $b^l$  predstavlja vrijednost praga za mapu značajku iz sloja  $l$ . U pravilu se vrijednost praga dijeli za cijelu mapu značajki, no to nije nužno te je moguće za svaku od mapa značajki iz prethodnog sloja imati po jednu vrijednost praga. Konačno, funkcija  $\sigma$  predstavlja aktivacijsku funkciju.

Zbog svojih svojstava, konvolucijski slojevi mogu uvelike poboljšati učinkovitost modela za klasifikaciju slika. Konvolucijski slojevi su tipično rijetko povezani — nije nužno da su sve mape značajki sloja  $l - 1$  povezane sa svakom od mapa značajki iz sloja  $l$  [8], što uz jezgre koje su tipično mnogo manjih veličina od ulaza nad kojima djeluju u velikoj mjeri smanjuje broj parametara i povećava efikasnost učenja i računanja. Potpuno povezani slojevi koriste matrično množenje i svaki parametar je iskorišten samo jednom za opisivanje veze između ulaza i izlaza, dok se konvolucijom isti parametri koriste za više lokacija ulaza i predstavljaju svojevrsne značajke koje jezgra izlučuje na slici. Konvolucijske arhitekture korištenjem jezgara s dijeljenim parametrima uče skup parametara koji je otporan na translacije jer će za iste ulazne vrijednosti jezgra dati iste izlazne vrijednosti, bez obzira gdje na slici se one pojave. Naučeni dijeljeni skup parametara predstavlja jednu funkciju koju jezgra obavlja (primjerice, detekcija

rubova) – učenjem se skup parametara prilagođava cijeloj slici i uči se tako da obavlja apstraktne funkcije neovisne o lokaciji slike, a koje će pripomoći u samoj klasifikaciji. Nabrojana svojstva konvolucijske arhitekture donose svojevrsnu ugrađenu regularizaciju što povlači bolju generalizaciju uz veću efikasnost [8] [17] [13].

### 3.3.3. Sloj sažimanja

Slojevi sažimanja koriste se da se grupira određeni broj podataka mape značajki i predstavi ih kao jedan podatak koji statistički dobro predstavlja tu grupu. Sam postupak također smanjuje veličinu mapu značajki i to tako da ih skalira faktorom  $\frac{1}{S_w}$  po širini, odnosno faktorom  $\frac{1}{S_h}$  po visini, gdje je  $S_w$  širina, a  $S_h$  visina podmape značajki koja se grupira.

Korisnost postupka očituje se u vidu ostvarivanja invarijantnosti na lokalne translacije što može biti vrlo korisno u slučaju da nam je bitnije da je neka značajka prisutna od toga na kojoj je točno lokaciji [13].

Postoje različite vrste funkcija koje se koriste za sažimanje od kojih je u ovom radu korištena sljedeća:

**sažimanje maksimalnom vrijednošću** Sažimanje maksimalnom vrijednošću (engl. *max-pooling*) grupira značajke tako da iz podmape značajki nad kojom djeluje izluči samo maksimalnu vrijednost koja postaje predstavnik. Sažimanje maksimalnom vrijednošću je najpopularnija metoda sažimanja i pokazuje se da ima vrlo dobre rezultate i performanse [15].

### 3.3.4. Sloj grupne normalizacije (engl. *batch normalization*)

Učenje dubokih modela može biti vrlo komplikirano ukoliko bolje promotrimo njihovu arhitekturu, način učenja i sam utjecaj načina učenja na brzinu učenja. Učenjem mijenjamo parametre modela, a samim time, promjenom parametara u određenom sloju modela, mijenjamo distribuciju ulaznih podataka koje dobiva idući sloj. Sama dubina dodatno uvećava utjecaj promjena parametara modela – distribucija ulaza svakog sloja uvjetovana je promjenama na parametre modela svih slojeva koji mu prethode.

Konstantna promjena distribucije ulaza slojeva predstavlja problem jer je potrebno stalno se prilagođavati na novu distribuciju, što može uvelike usporiti učenje. Opisani problem nazivamo interni kovarijacijski pomak (engl. *internal covariate shift*). Metoda grupne normalizacije (engl. *Batch Normalization*) uklanja taj problem i uvelike

ubrzava učenje dubokih modela.

Uklanjanje problema internog kovarijacijskog pomaka rješavamo normalizacijom ulaza svakog sloja – linearno ga transformiramo tako da mu je srednja vrijednost jednaka 0, a varijanca mu je jednaka 1. Direktna normalizacija ulaza i izlaza nekog sloja ovisi o statistici svih ulaza tog sloja nad kompletним skupom podataka nad kojim učimo mrežu. Zbog izrazite računske složenosti koja bi bila potrebna za to, uvode se neka pojednostavljenja.

Prva pretpostavka za pojednostavljenje složenosti jest da umjesto normalizacije ulaza i izlaza združeno, normaliziramo svaku komponentu ulaza zasebno, s pretpostavkom da komponente nisu međusobno zavisne. Normalizacija  $k$ -te komponente ulaznog vektora  $\mathbf{x}$  nekog sloja definirana je na sljedeći način:

$$\hat{\mathbf{x}}^{(k)} = \frac{\mathbf{x}^{(k)} - \mathbb{E} [\mathbf{x}^{(k)}]}{\sqrt{\text{Var} [\mathbf{x}^{(k)}]}}, \quad (3.19)$$

gdje su očekivanje i varijanca izračunati nad skupom podataka za učenje. Normalizacija svakog ulaza nekog sloja može promijeniti njegovu ekspresivnu snagu pa iz tog razloga uvodimo transformaciju normaliziranog ulaza kojom osiguravamo da je iz normaliziranih ulaznih podataka moguće dobiti originalne vrijednosti. Transformacija  $k$ -te komponente prethodno normaliziranog ulaznog vektora  $\hat{\mathbf{x}}$  nekog sloja definirana je na sljedeći način:

$$\mathbf{y}^{(k)} = \gamma^{(k)} \hat{\mathbf{x}}^{(k)} + \beta^{(k)}, \quad (3.20)$$

gdje su  $\gamma^{(k)}$  i  $\beta^{(k)}$  parametri koji se uče zajedno s parametrima modela i koji omogućuju skaliranje i pomak normalizirane vrijednosti izlaza.

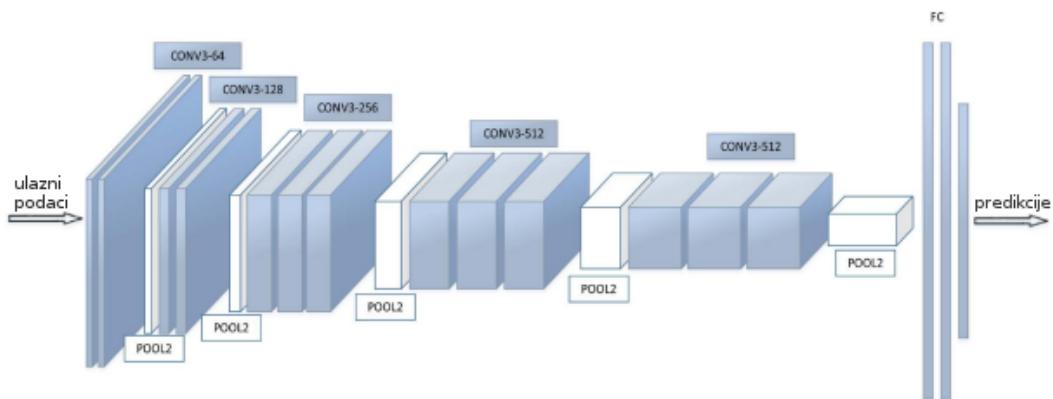
Druga pretpostavka za pojednostavljenje složenosti jest da normalizaciju umjesto nad statistikom cjelokupnog skupa podataka za učenje, obavljamo nad statistikama malih podskupova podataka (engl. *mini-batch*). Grupna normalizacija nad statistikom malog podskupa skupa podataka za učenje u nekom sloju tada je definirana kao normalizacija i linearna transformacija ulaza, definirane u izrazima 3.19 i 3.20.

Transformacija grupne normalizacije najčešće se koristi nad izračunatim težinskim sumama slojeva, prije djelovanja aktivacijske funkcije. Normalizirane vrijednosti  $\hat{\mathbf{x}}^{(k)}$  možemo promatrati kao ulaze u podsloj kojem je funkcija linearna, i to je upravo funkcija transformacije definirana u izrazu 3.20. Na izlaz podsloja tada djeluje procesiranje originalnog sloja. Također arhitekturom ostvarili smo da svi podslojevi na ulaze dobivaju podatke s fiksnom distribucijom (s očekivanjem 0 i varijancom 1), što ubrzava učenje podslojeva, a time posljedično i samih slojeva i kompletne mreže [10] [13].

## 3.4. Arhitekture korištene u eksperimentima

U poglavlju 2 opisana su dva tipa skupova podataka korištenih za eksperimente – skup podataka sa statičnim slikama i skup podataka sa sekvencama slika. Korišteno je nekoliko različitih modela koji su zasnovani na arhitekturi VGG-16 [16] treniranoj za natjecanje ILSVRC-2014 [14]. Model VGG-16 treniran je za klasifikaciju slika u 1000 različitih klasa na vrlo velikom broju slika – 1.3 milijuna [16]. Pokazuje se da model iz tog razloga vrlo dobro generalizira i na drugim problemima [16] pa je iz tog razloga odabran kao baza modela korištenih za eksperimente u ovom radu.

### 3.4.1. Arhitektura VGG-16 modela



Slika 3.4: Arhitektura VGG-16 modela, preuzeto i prerađeno iz [7]

Slika 3.4 prikazuje originalnu arhitekturu VGG-16 građenu od 16 slojeva.

Prednji dio sastoji se od 5 konvolucijskih blokova. Svaki od blokova izgrađen je na sličan način – nekoliko konvolucijskih slojeva iza kojih slijedi sloj za sažimanje maksimalnom vrijednošću (engl. *max pool*). Konvolucijski slojevi imaju jezgre dimenzija  $3 \times 3$  uz korak konvolucije (engl. *stride*) 1. Svi konvolucijski slojevi koriste po dijelovima linearnu funkciju (engl. *ReLU*) te su regularizirani L2 regularizacijom s faktorom regularizacije iznosa  $5 \cdot 10^{-4}$ . Slojevi za sažimanje maksimalnom vrijednošću imaju podmape za grupiranje veličine  $2 \times 2$  uz korak uzorkovanja podmapa 2. Konvolucijski blokovi građeni su kako slijedi:

- prvi konvolucijski blok:
  - konvolucijski sloj sa 64 izlaznih mapi značajki
  - konvolucijski sloj sa 64 izlaznih mapi značajki
  - sloj za sažimanje maksimalnom vrijednošću

- drugi konvolucijski blok:
  - konvolucijski sloj sa 128 izlaznih mapi značajki
  - konvolucijski sloj sa 128 izlaznih mapi značajki
  - sloj za sažimanje maksimalnom vrijednošću
- treći konvolucijski blok:
  - konvolucijski sloj s 256 izlaznih mapi značajki
  - konvolucijski sloj s 256 izlaznih mapi značajki
  - konvolucijski sloj s 256 izlaznih mapi značajki
  - sloj za sažimanje maksimalnom vrijednošću
- četvrti konvolucijski blok:
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - sloj za sažimanje maksimalnom vrijednošću
- peti konvolucijski blok:
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - konvolucijski sloj s 512 izlaznih mapi značajki
  - sloj za sažimanje maksimalnom vrijednošću

Stražnji dio građen je od 3 potpuno povezana sloja koji služe za klasifikaciju slika u 1000 različitih klasa. Prilikom korištenja VGG-16 arhitekture obično se uklanja stražnji ili svi potpuno povezani slojevi te se na prednji dio koji služi za izlučivanje značajki iz ulaznih slika nadodaju novi slojevi koji se treniraju za obavljanje specifičnog zadatka.

### **3.4.2. Arhitektura za klasifikaciju pojedinačnih slika**

Duboki model korišten za klasifikaciju pojedinačnih slika temeljen je na VGG-16 arhitekturi. Koristi se naučen prednji dio s 5 konvolucijskih blokova. Dobivene izlazne mape značajki pretvaraju se u jednodimenzionalan vektor koji se zatim vodi na potpuno povezane slojeve koji su definirani kako slijedi:

- potpuno povezani sloj dimenzionalnosti 200

- po dijelovima linearne (engl. *ReLU*) aktivacijska funkcija
- grupna normalizacija (engl. *batch normalization*)
- L2 regularizacija
- potpuno povezani sloj dimenzionalnosti 2 za binarnu klasifikaciju – određivanje je li atribut prisutan na slici ili nije
  - normalizirajuća eksponencijalna (engl. *softmax*) aktivacijska funkcija
  - L2 regularizacija

### 3.4.3. Arhitekture za klasifikaciju sekvenci slika

Za klasifikaciju sekvenci slika korišteno je nekoliko različitih modela temeljenih na naučenoj VGG-16 arhitekturi. Korišteni modeli su inspirirani arhitekturama iz [12].

Zajednički postupak koji svi modeli dijele jest postupak izlučivanja prostornih značajki – za svaku sliku sekvence se koristeći prednji dio naučenog VGG-16 modela izračunaju značajke koje se zatim dalje obrađuju na različite načine. Dobivene prostorne značajke svake pojedine slike su dimenzija  $H \times W \times 512$ , gdje dimenzije  $H$  i  $W$  ovise o dimenzijama ulaznih slika, a treća dimenzija je određena brojem izlaznih mapi značajki posljednjeg konvolucijskog sloja VGG-16 modela.

#### Klasifikacija sekvenci slika korištenjem vremensko-prostornog sažimanja

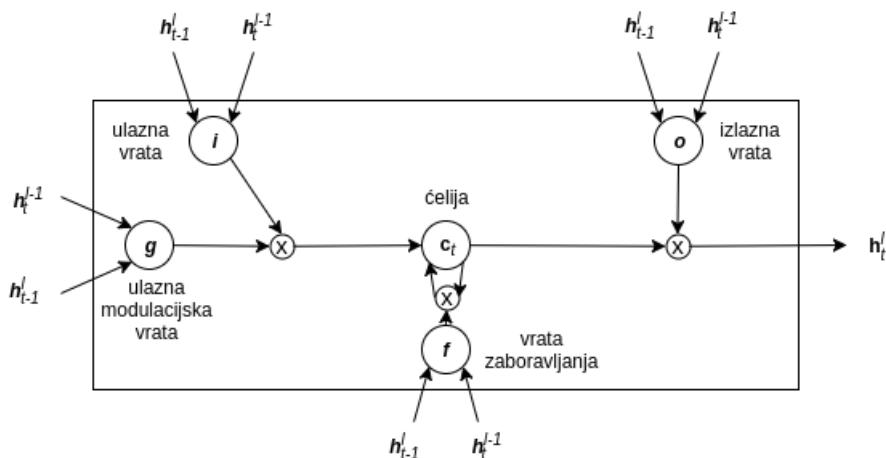
Model koji koristi vremensko-prostorno sažimanje prostorne značajke svake slike pretvori u jednodimenzionalni vektor dimenzija  $H \times W \times 512$ . Dobiveni jednodimenzionalni vektori se zatim oblikuju u dvodimenzionalnu strukturu dimenzija  $N \times (H \times W \times 512)$ , gdje je  $N$  duljina sekvence slika.

Dvodimenzionalna struktura koja se dobije povezivanjem prostornih značajki sadrži i vremenske i prostorne informacije. Dobivena dvodimenzionalna zatim se obrađuje kroz nekoliko slojeva kako slijedi:

- sloj za sažimanje maksimalnom vrijednošću s podmapama za grupiranje veličina  $2 \times 2$  uz korak uzorkovanja podmapa 2
  - operacija efektivno provodi vremensko-prostorno sažimanje značajki koje su bliske vremenski i prostorno
- pretvaranje dobivene dvodimenzionalne strukture u jednodimenzionalnu
- potpuno povezani sloj dimenzionalnosti 200
  - po dijelovima linearne (engl. *ReLU*) aktivacijska funkcija

- grupna normalizacija (engl. *batch normalization*)
- L2 regularizacija
- potpuno povezani sloj dimenzionalnosti 2 za binarnu klasifikaciju – određivanje je li atribut prisutan na slici ili nije
  - normalizirajuća eksponencijalna (engl. *softmax*) aktivacijska funkcija
  - L2 regularizacija

### Klasifikacija sekvenci slika korištenjem povratnih LSTM čelija (engl. *Long Short Term Memory*)



**Slika 3.5:** Arhitektura LSTM čelije, nacrtano prema [18]

Slika 3.5 prikazuje arhitekturu LSTM čelije. LSTM čelije pogodne su za korištenje nad sekvencama podataka iz razloga što sadrže memoriske čelije u kojima mogu čuvati informacije kroz dugačke vremenske periode.

U svakom vremenskom koraku potrebno je izračunati izlaz LSTM čelije i novi sadržaj memoriske čelije. Izlaz LSTM čelije i novi sadržaj memoriske čelije računaju se na temelju tri stvari:

- izlaz iz LSTM čelije iz prethodnog sloja za trenutni vremenski korak (ili ulazni podatak trenutnog vremenskog koraka ukoliko je riječ o prvom sloju LSTM čelija) –  $h_{t-1}^l$  na slici 3.5
- izlaz iz LSTM čelije trenutnog sloja za prethodni vremenski korak –  $h_l^{t-1}$  na slici 3.5
- sadržaj memoriske čelije trenutnog sloja za prethodni vremenski korak –  $c_t$  na slici 3.5

Izlaz LSTM čelije i novi sadržaj memorijske čelije računaju se sljedećim izrazima:

$$\mathbf{c}_t^l = \mathbf{f} \odot \mathbf{c}_{t-1}^l + \mathbf{i} \odot \mathbf{g} \quad (3.21)$$

$$\mathbf{h}_t^l = \mathbf{o} \odot \tanh(\mathbf{c}_t^l) \quad (3.22)$$

$$\mathbf{i} = \sigma(\mathbf{W}_i[\mathbf{h}_l^{t-1}, \mathbf{h}_{l-1}^t] + \mathbf{b}_i) \quad (3.23)$$

$$\mathbf{f} = \sigma(\mathbf{W}_f[\mathbf{h}_l^{t-1}, \mathbf{h}_{l-1}^t] + \mathbf{b}_f) \quad (3.24)$$

$$\mathbf{o} = \sigma(\mathbf{W}_o[\mathbf{h}_l^{t-1}, \mathbf{h}_{l-1}^t] + \mathbf{b}_o) \quad (3.25)$$

$$\mathbf{g} = \sigma(\mathbf{W}_g[\mathbf{h}_l^{t-1}, \mathbf{h}_{l-1}^t] + \mathbf{b}_g), \quad (3.26)$$

gdje je  $\tanh$  aktivacijska funkcija tangens hiperbolni,  $\sigma$  je sigmoidalna aktivacijska funkcija,  $\odot$  je operator množenja koji djeluje po elementima, a  $[\mathbf{x}, \mathbf{y}]$  je operacija spajanja vektora  $\mathbf{x}$  dimenzija  $n_1$  i vektora  $\mathbf{y}$  dimenzija  $n_2$  u vektor dimenzija  $n_1 + n_2$ .

Model koji koristi LSTM čelije prostorne značajke svake slike pretvori u jednodimenzionalni vektor dimenzija  $H \times W \times 512$ . Dobiveni jednodimenzionalni vektori se zatim redom obrađuju kako slijedi:

- LSTM sloj sa stanjem dimenzionalnosti 128
- LSTM sloj sa stanjem dimenzionalnosti 64
- LSTM sloj sa stanjem dimenzionalnosti 32
- potpuno povezani sloj dimenzionalnosti 2 za binarnu klasifikaciju – određivanje je li atribut prisutan na slici ili nije
  - ulaz u ovaj potpuno povezani sloj jest posljednji izlaz dobiven iz LSTM čelije prethodnog sloja koja je obradila cijelu sekvencu podataka (vektor dimenzionalnosti 32)
  - normalizirajuća eksponencijalna (engl. *softmax*) aktivacijska funkcija
  - L2 regularizacija

### **Klasifikacija sekvenci slika korištenjem vremenskog potpuno povezanog sloja**

Model koji koristi vremenski potpuno povezani sloj prostorne značajke pretvori u jednodimenzionalni vektor dimenzija  $H \times W \times 512$ . Svaki dobiveni jednodimenzionalni vektor prostornih značajki se zatim obrađuje kako slijedi:

- potpuno povezani sloj dimenzionalnosti 64
  - po dijelovima linearna (engl. *ReLU*) aktivacijska funkcija
  - grupna normalizacija (engl. *batch normalization*)
  - L2 regularizacija

- ovaj potpuno povezani sloj služi za predstavljanje prostornih značajki vektorom manjih dimenzija

Jednodimenzionalni vektori dobiveni nakon potpuno povezanog sloja se zatim oblikuju u dvodimenzionalnu strukturu dimenzija  $N \times (H \times W \times 512)$ , gdje je  $N$  duljina sekvence slika. Dobivena dvodimenzionalna zatim se obrađuje kroz nekoliko slojeva kako slijedi:

- pretvaranje dobivene dvodimenzionalne strukture u jednodimenzionalnu
- potpuno povezani sloj dimenzionalnosti 64
  - po dijelovima linearna (engl. *ReLU*) aktivacijska funkcija
  - grupna normalizacija (engl. *batch normalization*)
  - L2 regularizacija
  - ovaj potpuno povezani sloj služi za povezivanje dobivenih prostornih značajki po dimenziji vremena
- potpuno povezani sloj dimenzionalnosti 2 za binarnu klasifikaciju – određivanje je li atribut prisutan na slici ili nije
  - normalizirajuća eksponencijalna (engl. *softmax*) aktivacijska funkcija
  - L2 regularizacija

## 3.5. Postupak pripreme skupova podataka

Poglavlje 2 opisuje dva tipa skupova podataka korištenih u ovom radu. U nastavku je opisan postupak preprocesiranja slika dobivenih iz videa sa stranice [1] te postupak kreiranja slika označenih konzistentno s projektom FTTS iRAP.

### 3.5.1. Preprocesiranje slika

Slike korištene za kreiranje skupova podataka dobivene su iz video snimaka snimljenih automobilom koji se kretao engleskim autocestama. Zbog kretnje automobila uzorkovane slike bit će snimljene iz različitog kuta u odnosu na Sunce. Takva situacija za posljedicu ima da su neke slike mnogo svjetlijе, a neke mnogo tamnije od drugih.

Metoda kojom se može ublažiti velika varijacija kontrasta između slika jest metoda adaptivnog izjednačavanja histograma [5]. Izjednačavanje histograma za cilj ima izjednačiti razdiobu intenziteta svih kanala slike (crvenog, zelenog i plavog) na način da oni budu uniformno distribuirani. Metoda adaptivnog izjednačavanja histograma

za računanje histograma ne uzima u obzir cijelu sliku nego prozore određene veličine čime se dobiva na lokalnosti izjednačavanja.



**Slika 3.6:** Originalna slika, preuzeto s [1]



**Slika 3.7:** Slika s izjednačenim histogramom, preuzeto s [1]



**Slika 3.8:** Slika s adaptivno izjednačenim histogramom, preuzeto s [1]

Iz slika 3.6, 3.7 i 3.8 vidljivo je kako su, u odnosu na originalnu sliku, na slici s adaptivno izjednačenim histogramom najbolje naglašeni detalji zbog najbolje ublažene varijacije kontrasta. Metoda adaptivnog izjednačavanja histograma se iz tog razloga koristi kao korak pretprocesiranja za svaku sliku korištenu prilikom kreiranja skupova podataka.

### **3.5.2. Postupak kreiranja slika označenih konzistentno s projektom FTTS iRAP**

Za postupak kreiranja slika prometnih scena označenih konzistentno s projektom FTTS iRAP kreiran je automatizirani postupak. Ulazni podaci za automatizirani postupak su sljedeći:

- geolokacije na kojima je prisutan atribut pripajanja
- popis poveznica na sve video snimke u sustavu [1]
- broj video snimaka koji će se obraditi

Postupak će iz ulaznih podataka kreirati sljedeće:

- slika s oznakom prisutnosti atributa pripajanja na njoj te sekvenca slika koja joj prethodi (kao što je objašnjeno u poglavlju 2)
- slika s oznakom prisutnosti atributa pripajanja bez sekvence koja joj prethodi
- geolokacije označenih slika i sekvenci slika

Postupak kreira označene slike sa i bez sekvence koja joj prethodi kako bi se eksperimentima ustanovilo dobiva li se na točnosti postupka uvodeći više informacija kroz sekvencu slika.

Geolokacije video sekvence uzorkovane su rijetko – najčešće svakih  $0.5s$ . Iz tog razloga se za točnu geolokaciju konkretne sličice video sekvence vrši linearna interpolacija između dvije uzorkovane geolokacije koje su vremenski najbliže konkretnoj sličici. Geolokacija konkretnih sličica korisna je informacija iz razloga što je prilikom testiranja modela za svaki ulazni podatak kojem model dodijeli krivu oznaku moguće u sustavu FTTS iRAP provjeriti je li osoba koja je dodijelila vrijednost atributa tom segmentu to napravila točno ili je riječ o mogućoj pogrešci. Takvim postupkom moguće je na brži način validirati ispravnost dodijeljenih oznaka u sustavu.

Nastavak opisuje sažeti pseudokod automatiziranog postupka:

---

**Algoritam 1** Automatizirano kreiranje slika označenih konzistentno s projektom FTTS iRAP

---

**Sve dok** nije obrađen željen broj video sekvenci **Ponavljam**

```
v=preuzmi_video()
geolok,vrem=preuzmi_geolokacije_i_vremena(v)
sl=kreiraj_sličice(v)
ind_geolok,brz_aut=indeksiraj_geolokacije_i_izračunaj_brzine_automobila(geolok,vrem)
ras_poz=izračunaj_raspone_pozitiva(geolok_pripajanja,ind_geolok,brz_aut)
poz_slič,geolok_slič=kreiraj_positivne_sličice_s_geolokacijama(ras_poz,sl)
pohrani_sličice(poz_slič,geolok_slič)
neg_slič,geolok_slič=kreiraj_negativne_sličice_s_geolokacijama(ras_poz,sl)
pohrani_sličice(neg_slič,geolok_slič)
```

**Kraj**

---

Izračunavanje raspona sličica na kojima je prisutan atribut pripajanja (pozitivne sličice) radi se na sljedeći način:

- za svako pripajanje i njegovu geolokaciju pronađu se dvije najbliže geolokacije video sekvence
- ukoliko je bliža geolokacija video sekvence unutar praga udaljenosti pripajanju, uzima se da je pripajanje prisutno na tom dijelu video sekvence
- na temelju udaljenosti geolokacija video sekvence od geolokacije pripajanja i brzina kretanja automobila izračuna se točan raspon vremena video sekvence na kojem se nalazi segment atributa pripajanja
- iz dobivenog raspona vremena vremena izračuna se raspon sličica video sekvence na kojem se nalazi segment atributa pripajanja

# 4. Eksperimenti i rezultati

Za potrebe ovog rada provedeno je nekoliko eksperimenata. Postupak učenja i testiranja u svakom eksperimentu je jednak, razlikuju se samo u vrsti ulaznih podataka (statične slike ili sekvene slike) te u korištenoj arhitekturi.

## 4.1. Način učenja i testiranja u eksperimentima

Početni korak prilikom učenja jest normalizacija skupa podataka. Kako su podaci slike u boji, na podacima za treniranje se izračunava srednja vrijednost intenziteta piksela za svaki od kanala – crveni, zeleni i plavi. Dobivene srednje vrijednosti se zatim po kanalima oduzimaju svim podacima – podacima za treniranje, podacima za validaciju i podacima za testiranje.

Kako su sve arhitekture korištene za potrebe ovog rada zasnovane na naučenom prednjem dijelu arhitekture VGG-16, učenje se obavlja kroz 50 epoha i to na sljedeći način:

- 10 epoha učenja s početnom stopom učenja iznosa  $5 \cdot 10^{-4}$  uz optimiranje samo novo nadodanih parametara – svi parametri naučenog prednjeg dijela arhitekture VGG-16 ne mijenjaju se tijekom ove faze
- 40 epoha učenja s početnom stopom učenja iznosa  $1 \cdot 10^{-5}$  uz optimiranje kompletog skupa parametara (uključujući i parametre naučenog prednjeg dijela arhitekture VGG-16)
- korak učenja koristi gradiente izračunate nad mini-grupama veličina 5 ili 10, ovisno o memoriskoj zahtjevnosti modela
- optimizator korišten za učenje jest Adam optimizator
- stopa učenja se svakim korakom učenja eksponencijalno smanjuje:

$$\eta_t = \eta_0 \cdot 0.96^{\frac{t}{k}}, \quad (4.1)$$

gdje je  $t$  indeks koraka učenja, a  $k$  je broj koraka učenja u jednoj epohi

- slobodni parametri regularizirani su L2 regularizacijom s faktorom regularizacije iznosa  $5 \cdot 10^{-4}$
- podskup za validaciju korišten je za odabir slobodnih parametara modela
  - na kraju svake epohe mjeri se performansa modela na podskupu za validaciju
  - za testiranje se odabire onaj skup parametara koji postiže najbolju performansu na podskupu za validaciju – ovime se efektivno obavlja regularizacija jer odabiremo onaj skup parametara koji najbolje generalizira

Svi eksperimenti provedeni su na nVidia grafičkim karticama GTX 1070 i GTX Titan.

Zadatak ovog rada definiran kao binarna klasifikacija, pa možemo iskoristiti 4 definirana tipa predikcija modela:

- TP (engl. *true positive*) – slika koja je označena kao pozitivna (atribut pripajanja je prisutan) i za koju je predikcija modela također takva
- TN (engl. *true negative*) – slika koja je označena kao negativna (atribut pripajanja nije prisutan) i za koju je predikcija modela također takva
- FP (engl. *false positive*) – slika koja je označena kao negativna, a predikcija modela je pozitivna
- FN (engl. *false negative*) – slika koja je označena kao pozitivna, a predikcija modela je negativna

Mjere koje su uzimane u obzir prilikom testiranja jesu točnost (engl. *accuracy*), preciznost (engl. *precision*) i odziv (engl. *recall*). Točnost jest definirana na sljedeći način:

$$\text{točnost} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (4.2)$$

Odziv jest definirana na sljedeći način:

$$\text{odziv} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4.3)$$

Preciznost jest definirana na sljedeći način:

$$\text{preciznost} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (4.4)$$

## 4.2. Klasifikacija ručno označenih slika na različitim rezolucijama

Eksperimenti na ručno označenim slikama provedeni su koristeći arhitekturu za klasifikaciju pojedinačnih slika. Kako su eksperimenti nad sekvencama slika memorijski mnogo zahtjevniji od eksperimenata na pojedinačnim slikama, svrha ovog eksperimenta bila je utvrditi na kojoj rezoluciji ulaznih slika model gubi ekspresivnu moć i ne uspijeva dati zadovoljavajuću performansu.

Za početnu, originalnu rezoluciju pojedinačnih slika odabrana je rezolucija 700x280. Eksperiment je proveden redom na sljedećim rezolucijama: 700x280, 525x210, 350x140 i 175x70. Rezultati su dani u nastavku.

### 4.2.1. Rezultati na rezoluciji 700x280

Konačni rezultati dobiveni za eksperiment s ulaznim podacima rezolucije 700x280 su sljedeći:

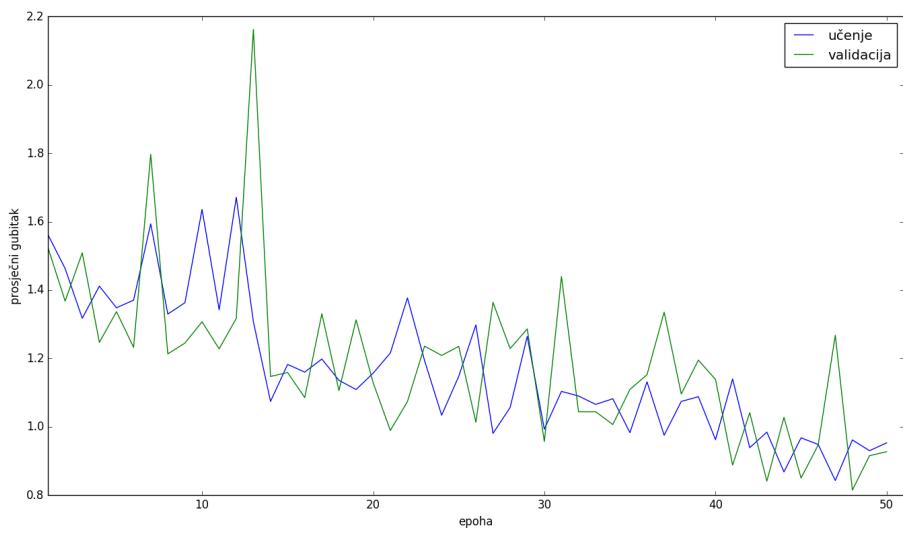
**Tablica 4.1:** Statistički rezultati na rezoluciji 700x280

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.2:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 700x280

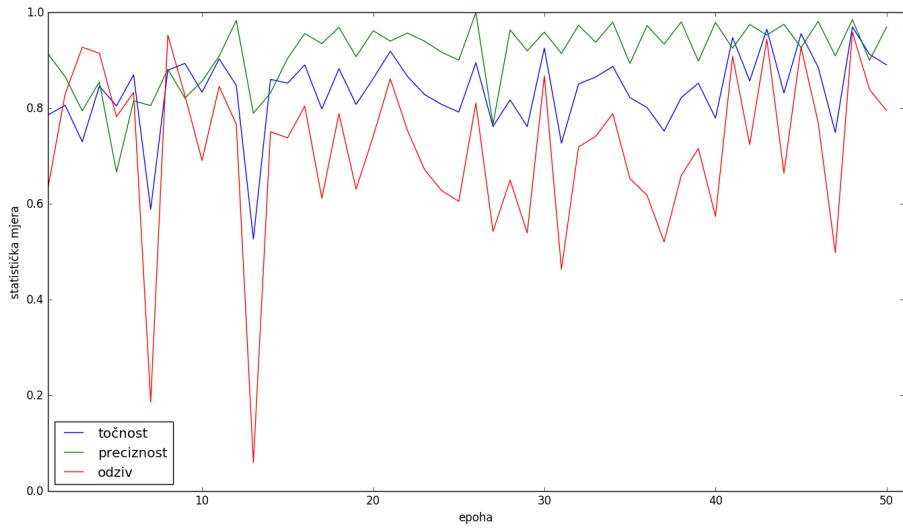
predikcija modela \ stvarna oznaka			
		0	1
0	0	TN	FP
	1	FN	TP

Tablica 4.1 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.2 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.1:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama

Na slici 4.1 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.2:** Kretanje točnosti, preciznosti i odziva po epohama

Na slici 4.2 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu za učenje i na podskupu za validaciju.

#### 4.2.2. Rezultati na rezoluciji 525x210

Konačni rezultati dobiveni za eksperiment s ulaznim podacima rezolucije 525x210 su sljedeći:

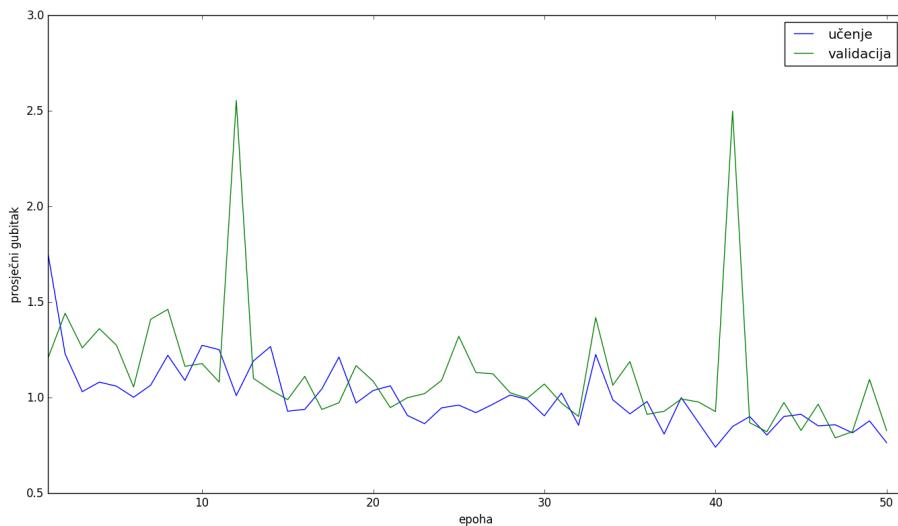
**Tablica 4.3:** Statistički rezultati na rezoluciji 525x210

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.4:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 525x210

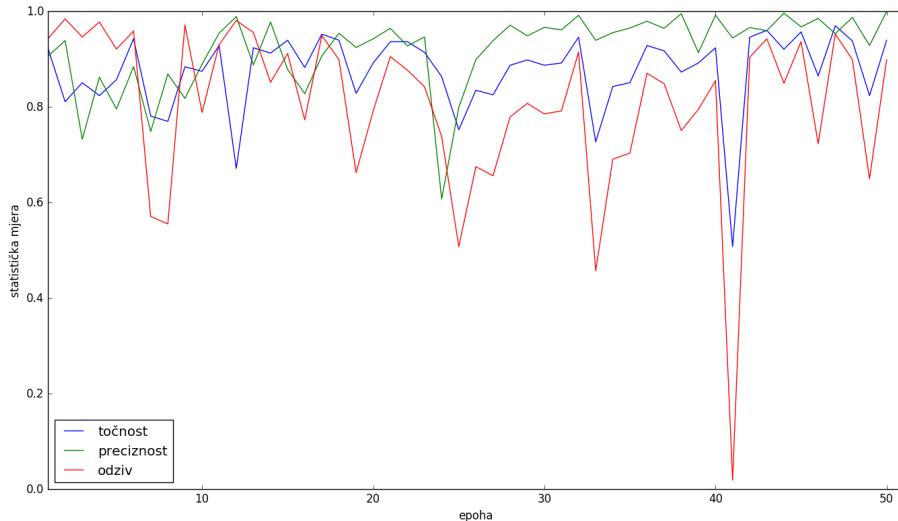
predikcija modela \ stvarna oznaka		0		1
		0	1	0
0	0	TN	FP	0
	1	FN	TP	1

Tablica 4.3 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.4 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.3:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama

Na slici 4.3 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.4:** Kretanje točnosti, preciznosti i odziva po epohama

Na slici 4.4 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu za učenje i na podskupu za validaciju.

#### 4.2.3. Rezultati na rezoluciji 350x140

Konačni rezultati dobiveni za eksperiment s ulaznim podacima rezolucije 350x140 su sljedeći:

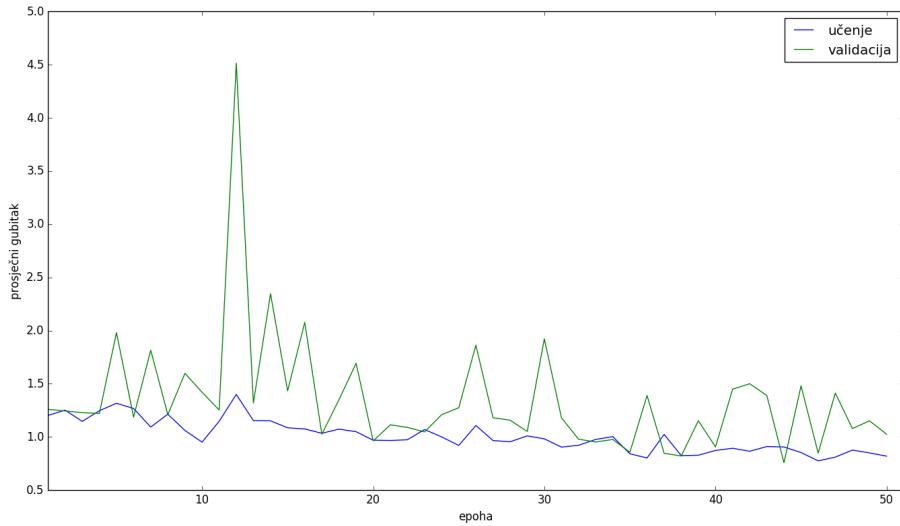
**Tablica 4.5:** Statistički rezultati na rezoluciji 350x140

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.6:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 350x140

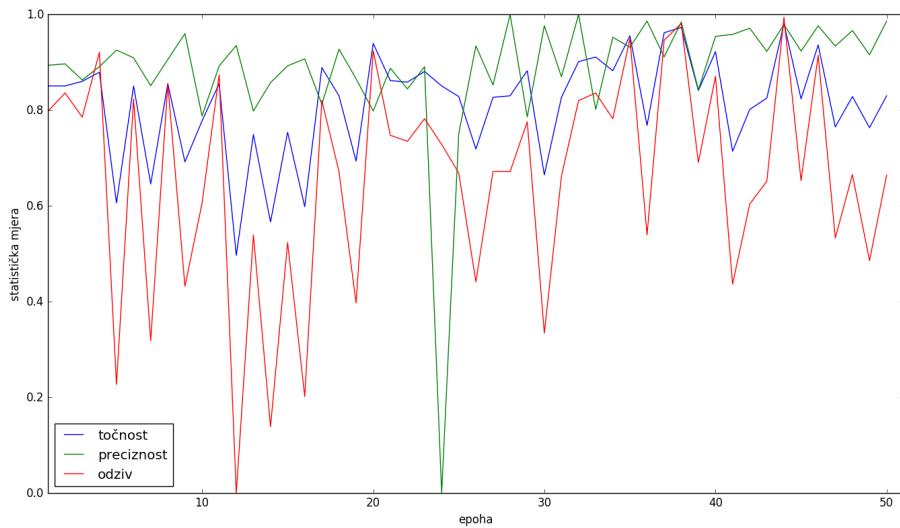
		stvarna oznaka	
		0	1
predikcija modela	0	TN	FP
	1	FN	TP

Tablica 4.5 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.6 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.5:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama

Na slici 4.5 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.6:** Kretanje točnosti, preciznosti i odziva po epohama

Na slici 4.6 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu

za učenje i na podskup za validaciju.

#### 4.2.4. Rezultati na rezoluciji 175x70

Konačni rezultati dobiveni za eksperiment s ulaznim podacima rezolucije 175x70 su sljedeći:

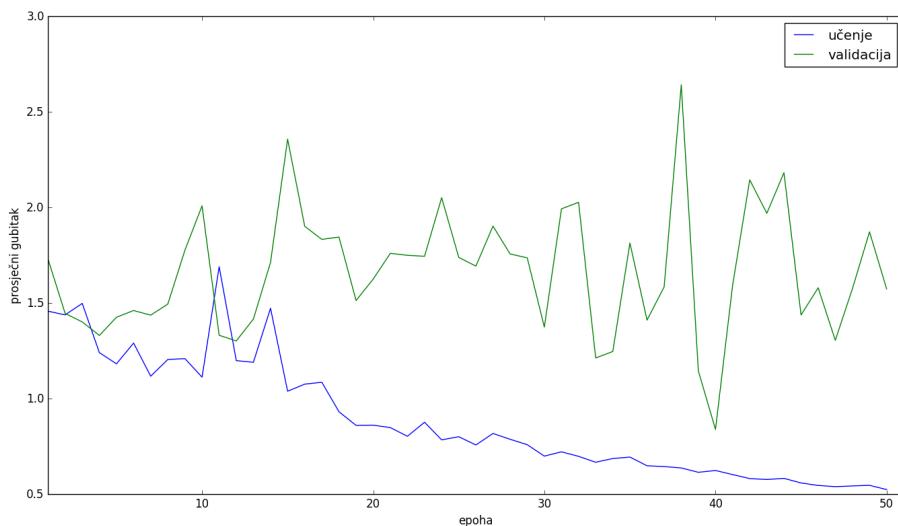
**Tablica 4.7:** Statistički rezultati na rezoluciji 175x70

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.8:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 175x70

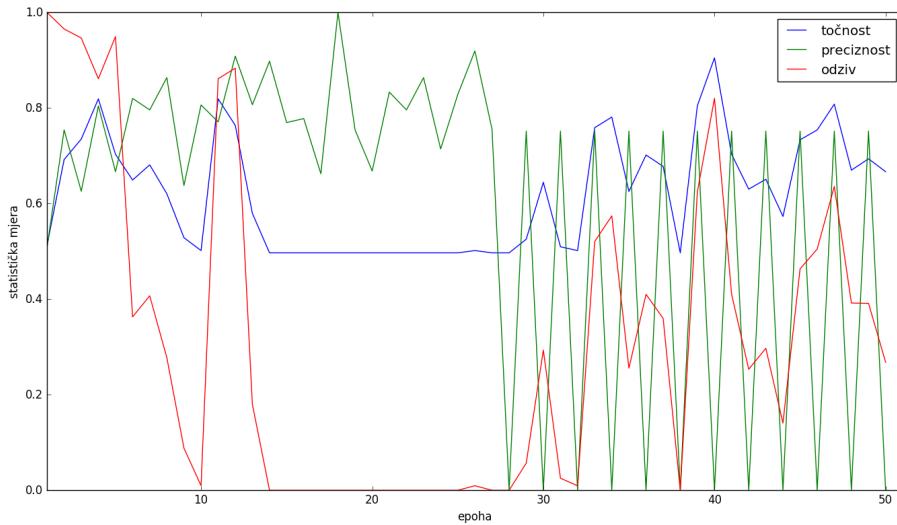
predikcija modela \ stvarna oznaka		0		1
		0	1	0
0	0	TN	FP	
	1	FN	TP	

Tablica 4.7 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.8 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.7:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama

Na slici 4.7 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.8:** Kretanje točnosti, preciznosti i odziva po epohama

Na slici 4.8 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu za učenje i na podskupu za validaciju.

Vidljivo je kako se tek na rezoluciji 175x70 pojavljuje značajniji pad performansi. Iz tog razloga se za eksperimente nad video sekvencama (zbog veće memorijске zah-tjevnosti) koriste rezolucije slika 350x140.

### 4.3. Klasifikacija pojedinačnih slika označenih konzistentno s projektom FTTS iRAP na originalnoj rezoluciji

Za razliku od prethodnih eksperimenata, ovaj eksperiment je proveden na skupu podataka označenom konzistentno sa sustavom FTTS iRAP. Eksperiment je proveden na pojedinačnim slikama kako bi se moglo usporediti poboljšava li rezultat uvođenje do-datnih informacija uvođenjem sekvence slika koja prethodi slici za koju je određena oznaka prisutnosti atributa pripajanja.

Konačni rezultati dobiveni za eksperiment s ulaznim podacima rezolucije 700x280 su sljedeći:

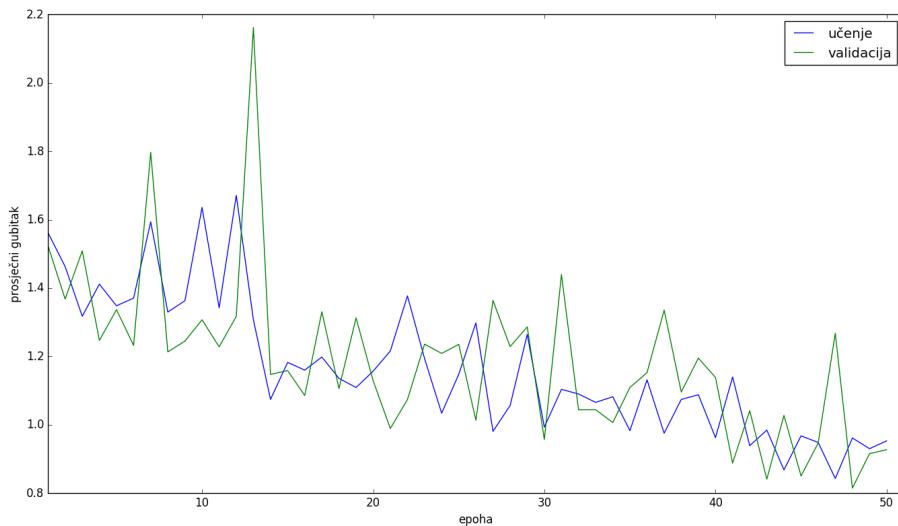
**Tablica 4.9:** Statistički rezultati na rezoluciji 700x280

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.10:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 700x280

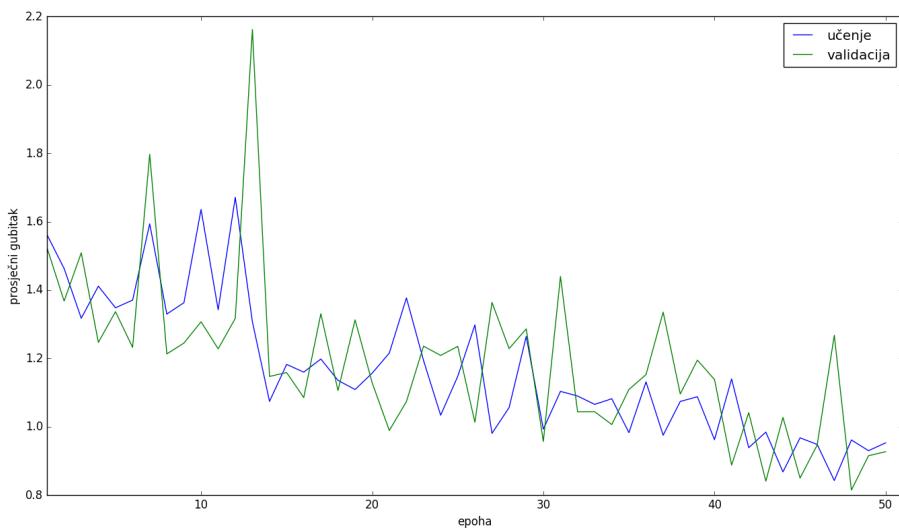
predikcija modela		stvarna oznaka	
		0	1
0	0	TN	FP
	1	FN	TP

Tablica 4.9 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.10 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.9:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama – promijeniti sliku

Na slici 4.9 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.10:** Kretanje točnosti, preciznosti i odziva po epohama – promijeniti sliku

Na slici 4.10 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu za učenje i na podskupu za validaciju.

#### 4.4. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći vremensko-prostorno sažimanje

Konačni rezultati dobiveni za eksperiment sa sekvencom ulaznih podataka rezolucije 350x140 su sljedeći:

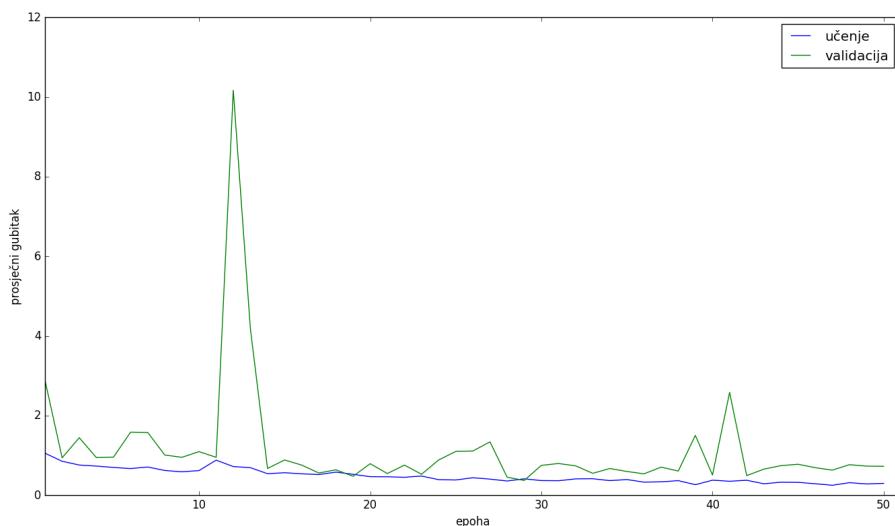
**Tablica 4.11:** Statistički rezultati na rezoluciji 350x140

podskup podataka	točnost	preciznost	odziv
podskup za učenje	0.5	0.8	1.1
podskup za validaciju	0.6	0.9	1.2
podskup za testiranje	0.7	1	1.3

**Tablica 4.12:** Odnos predikcija modela i stvarnih oznaka na rezoluciji 350x140

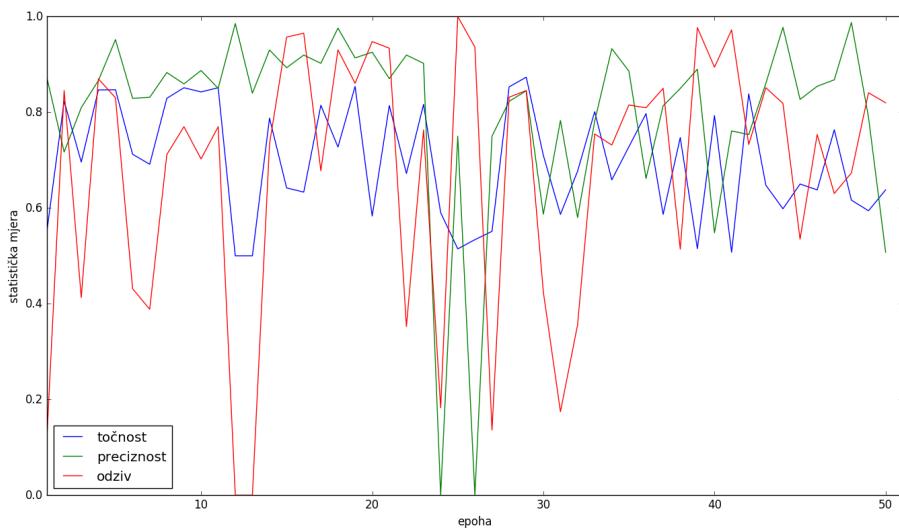
		stvarna oznaka	
		0	1
predikcija modela	0	TN	FP
	1	FN	TP

Tablica 4.11 prikazuje rezultate dobivene evaluacijom modela koji je postigao najbolju performansu na podskupu za validaciju prilikom učenja. Tablica 4.12 prikazuje odnos predikcija modela i stvarnih oznaka kroz matricu zabune.



**Slika 4.11:** Kretanje prosječnog gubitka na skupu za učenje i validaciju po epohama

Na slici 4.11 vidljivo je kretanje prosječnog gubitka po epohama na podskupu za učenje i na podskupu za validaciju.



**Slika 4.12:** Kretanje točnosti, preciznosti i odziva po epohama

Na slici 4.12 vidljivo je kretanje točnosti, preciznosti i odziva po epohama na podskupu za učenje i na podskupu za validaciju.

#### **4.5. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći povratne LSTM celije**

#### **4.6. Klasifikacija sekvenci slika označenih konzistentno s projektom FTTS iRAP koristeći vremenski potpuno povezani sloj**

#### **4.7. Analiza rezultata**

## **5. Zaključak**

Zaključak.

# LITERATURA

- [1] <https://he.ftts-irap.org/>, . [pristupano 14. lipnja 2017.].
- [2] <https://www.google.hr/maps?source=tldsi&hl=en>, . [pristupano 16. lipnja 2017.].
- [3] iRAP Star Rating and Investment Plan Coding Manual. [http://downloads.irap.org/docs/RAP-SR-2-2\\_Star\\_Rating\\_coding\\_manual.pdf](http://downloads.irap.org/docs/RAP-SR-2-2_Star_Rating_coding_manual.pdf), 2014. [pristupano 14. lipnja 2017.].
- [4] iRAP Methodology Fact Sheet #3 Road attributes. <http://www.irap.org/en/about-irap-3/methodology?download=132:irap-methodology-fact-sheet-3-road-attributes>, 2015. [pristupano 14. lipnja 2017.].
- [5] Dan C. Ciresan, Ueli Meier, i Jürgen Schmidhuber. Multi-column deep neural networks for image classification. *CoRR*, abs/1202.2745, 2012. URL <http://arxiv.org/abs/1202.2745>.
- [6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, i L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. U *CVPR09*, 2009.
- [7] H. El Khiyari i H. Wechsler. Face Recognition across Time Lapse Using Convolutional Neural Networks. *Journal of Information Security*, 7(3):141–151, 2016.
- [8] Ian Goodfellow, Yoshua Bengio, i Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015. URL <http://arxiv.org/abs/1502.01852>.
- [10] Sergey Ioffe i Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

- [11] Diederik P. Kingma i Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL <http://arxiv.org/abs/1412.6980>.
- [12] Joe Yue-Hei Ng, Matthew J. Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, i George Toderici. Beyond short snippets: Deep networks for video classification. *CoRR*, abs/1503.08909, 2015. URL <http://arxiv.org/abs/1503.08909>.
- [13] Ivan Relić. Klasifikacija slika dubokim konvolucijskim mrežama, 2016. seminarski rad.
- [14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, i Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.
- [15] Dominik Scherer, Andreas Muller, i Sven Behnke. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. 2010.
- [16] Karen Simonyan i Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014. URL <http://arxiv.org/abs/1409.1556>.
- [17] Vedran Vukotić. Raspoznavanje objekata dubokim neuronskim mrežama, 2014. diplomski rad.
- [18] Wojciech Zaremba, Ilya Sutskever, i Oriol Vinyals. Recurrent neural network regularization. *CoRR*, abs/1409.2329, 2014. URL <http://arxiv.org/abs/1409.2329>.
- [19] Siniša Šegvić. Uvodno predavanje. <http://www.zemris.fer.hr/~ssegvic/du/du0intro.pdf>, 2016. [pristupano 17. lipnja 2017.].

# **Detekcija sigurnosnih atributa prometnica u snimkama**

## **Sažetak**

Sažetak na hrvatskom jeziku.

**Ključne riječi:** Ključne riječi, odvojene zarezima.

## **Title**

## **Abstract**

Abstract.

**Keywords:** Keywords.