SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I
RAČUNARSTVA

SEMINARSKI RAD

# Prevođenje iz slike u sliku korištenjem uvjetnih suparničkih modela

*Kristijan Fugošić*

voditelj:
Prof.Dr.sc. Siniša Šegvić

Zagreb, svibanj 2019.

# Sadržaj

**1. Uvod** **1**

**2. Generativni suparnički modeli** **2**

2.1. Generator . . . . . 3

2.2. Diskriminator . . . . . 3

2.3. Mogući problemi . . . . . 3

2.4. Funkcija cilja . . . . . 4

2.5. Usporedba s drugim pristupima . . . . . 5

**3. Uvjetni suparnički modeli** **7**

**4. Prevođenje iz slike u sliku korištenjem uvjetnih suparničkih modela** **8**

4.1. Arhitektura modela pix2pix . . . . . 9

4.1.1. Generator . . . . . 9

4.1.2. Diskriminator . . . . . 12

4.2. Učenje . . . . . 13

4.3. Ulazni podatci i treniranje . . . . . 14

**5. Eksperimentalni rezultat** **16**

**6. Zaključak** **20**

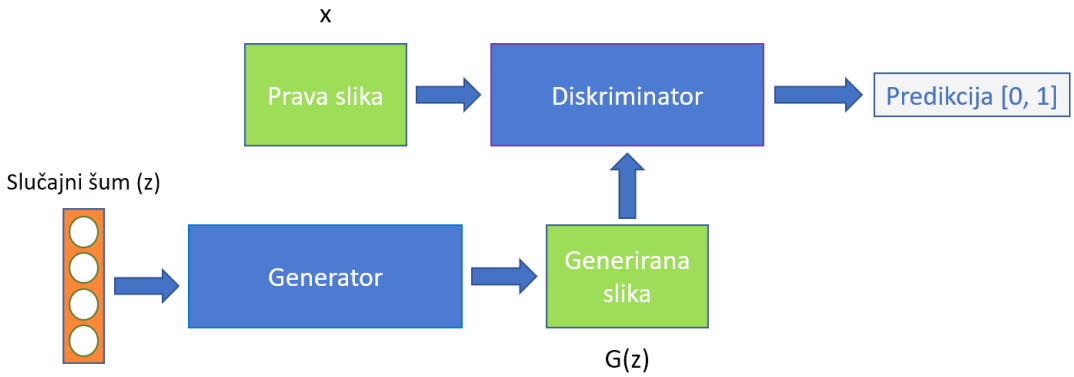
**Literatura** **21**

# 1. Uvod

Kroz ovaj rad bit će razmotrene generativni suparnički modeli čija je primarna namjena direktno generiranje uzoraka, bez procjene distribucije ulaznih podataka. Arthur Samuel je još 1959. godine pokazao da algoritmi mogu naučiti igrati damu igrajući jedni protiv drugih. U kontekstu neuronskih mreža, Ian Goodfellow je objavio svoj rad "Generative Adversarial Nets" 2014. godine i time potaknuo razvoj velikog broja različitih modela temeljenih na GAN-u. Primjene su raznolike, pa se današnji generativni suparnički modeli koriste za generiranje slika iz tekstualnih opisa i obratno, predviđanje budućih okvira u videozapisima, generiranje tlocrta iz satelitskih snimaka, segmentaciju slika, realistično podizanje rezolucije slika, ali i za potrebe dizajna, umjetnosti i zabave. Jedan konkretan primjer, pix2pix, bit će opisan kroz ovaj rad. pix2pix je model za prevođenje iz slike u sliku korištenjem uvjetnih suparničkih modela. Model je svestran i može se koristiti za različita prevođenja, poput prevođenja iz skice u stvarni objekt, iz segmentirane slike u fotografiju prometa ili iz noćne slike u dnevnu, potreban je jedino dovoljno velik skup podataka za treniranje.

# 2. Generativni suparnički modeli

Generativni suparnički modeli sastoje se od dvije neuronske mreže, generatora koji generira umjetne uzorke i diskriminatora koji ocjenjuje njihovu autentičnost. Preciznije, generator generira nove umjetne uzorke s ciljem da budu dovoljno slične pravim uzorcima da prođu provjeru diskriminatora, dok diskriminator za svaku ulaz koji dobije donosi sun pripada li uzorak na ulazu skupu za treniranje ili je generiran od strane generatora. Rezultat ovakvog načina njihovog rada jest natjecanje koje ih tjera da budu sve bolji i bolji u svojim nastojanjima, sve do trenutka kada generator generira umjetne uzorke koje diskriminator ne može razlikovati od pravih.




**Slika 2.1:** Generativni suparnički model.

Promotrimo sliku 2.1. Definirajmo izlaz diskriminatora kao vjerojatnost da je ulazni uzorak iz skupa za treniranje. Ako je na ulazu neka prava slika $x$ iz skupa za treniranje, želimo da rezultat diskriminatora $D(x)$ bude što bliže 1. S druge strane, ako slučajni šum označimo sa $z$, a sliku generiranu u generatoru kao $G(z)$, generator treniramo na način da nastoji izlaz diskriminatora $D(G(z))$ približiti jedinici, dok je istovremeno cilj diskriminatora prepoznati generiranu sliku i kao rezultat $D(G(z))$ vratiti vrijednost što bliže nuli.

## 2.1. Generator

Zadaća generatora je generiranje novih, realnih uzoraka, sličnih onima iz skupa za treniranje. Želimo da generator generira različite uvjerljive uzorke, stoga mu na ulazu dajemo slučajni šum ($z$) s određenom distribucijom $p(z)$, te je njegov zadatak da ulazni prostor skrivene varijable $z$ mapira u prostor podataka. $z$ možemo interpretirati kao izvor nasumičnosti koji omogućava da generator generira više različitih realističnih slika, a ne samo jednu. Generator označavamo sa $G(z, \theta_g)$, gdje sa $\theta_g$ označavamo parametre generatora.

## 2.2. Diskriminator

Diskriminator treniramo kako bi razlikovao stvarne uzorke iz skupa za treniranje od umjetnih uzoraka koje proizvodi generator. To postižemo nadziranim učenjem gdje na izlazu očekujemo vjerojatnost da ulazni uzorak nije nastao u generatoru već pripada skupu za treniranje. Diskriminator označavamo sa $D(x, \theta_d)$, gdje sa $\theta_d$ označavamo parametre diskriminatora, a $x$ ulazni uzorak.

## 2.3. Mogući problemi

Izgledna je situacija u kojoj će generator pronaći slučaj na koji diskriminator loše reagira. Generator će zatim sve svoje buduće generirane uzorke koncentrirati oko tog slučaja, sve dok diskriminator ne nauči taj konkretan slučaj. Problem je što generator nakon toga neće naučiti da mora generirati raznolike uzorke, nego će i dalje nove uzorke koncentrirati oko nekog novog specifičnog slučaja. Jedan od načina [11] za riješiti taj problem jest učenje diskriminatora na *minibatchu*, odnosno dopuštanje diskriminatoru da pri klasifikaciji pojedinog uzorka u obzir uzima i ostale članove *minibatcha*, te u slučaju prevelike sličnosti unutar *minibatcha* sve uzorke unutar njega označi kao lažne.
Neuravnoteženost dobrote odrađivanja generatora i diskriminatora za posljedicu ima nestabilnost. Ako je diskriminator predobar vraćat će vrijednosti toliko blizu nule ili jedinice da će generator imati problema s čitanjem gradijenata. S druge strane, ako je generator predobar iskorištavat će slabosti diskriminatora koje dovode do lažnih negativnih (ulazna slika je prava) predikcija. Djelomično rješenje ovoga problema dano je u nastavku.

## 2.4. Funkcija cilja

U ranije skiciranom modelu možemo prepoznati dvije povratne veze, diskriminatora s uzorcima iz skupa za treniranje i generatora s diskriminatorom.

Postupkom treniranja optimiziramo funkciju cilja s obzirom na parametre obje mreže, a to možemo prikazati sljedećim izrazom kao minimax igru:

$$\min_G \max_D V(D,G) = \mathbb{E}x \sim p_{data}(x)[logD(x)] + \mathbb{E}z \sim p_z(z)[log(1-D(G(z)))] \quad (2.1)$$

Gornji izraz možemo razdvojiti na funkcije gubitka diskriminatora i generatora:

$$J^{(D)} = -\mathbb{E}x \sim p_{data}(x)[logD(x)] - \mathbb{E}z \sim p_z(z)[log(1 - D(G(z)))] \quad (2.2)$$

$$J^{(G)} = -J^{(D)} \quad (2.3)$$

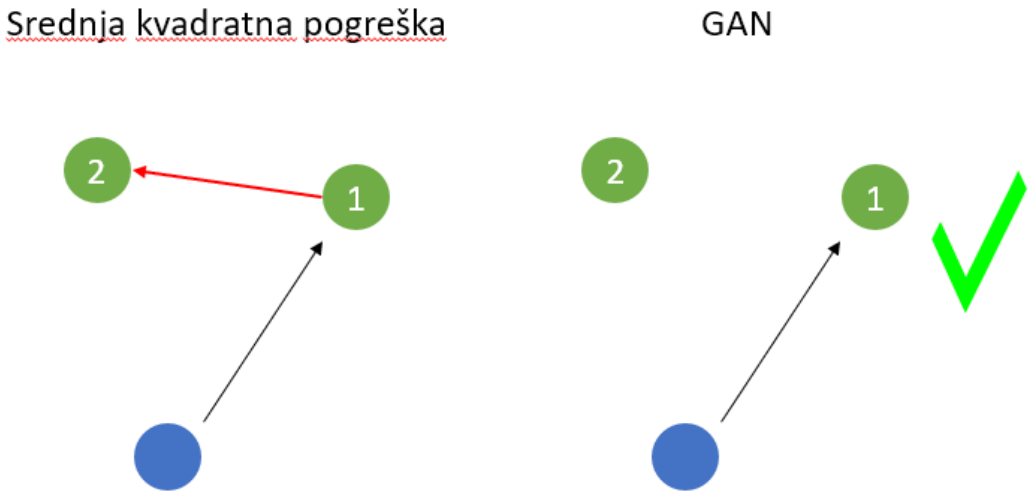
Možemo primijetiti da navedeni izraz odgovara unakrsnoj entropiji. Ako za primjer uzmemo klasičnu klasifikaciju uz korištenje unakrsne entropije, gradijenti padaju na nulu tek kada je mreža savršeno istrenirana i gradijent nam ionako više nije potreban. Kao što je ranije spomenuto, u slučaju generativnih suparničkih modela dobar diskriminator također dovodi do nestanka gradijenata, ali to nam predstavlja problem jer je taj gradijent i dalje potreban za učenje generatora.
Ako se prisjetimo uvodnog dijela ovog poglavlja, rekli smo da generator nastoji izlaz diskriminatora $D(G(z))$ približiti nuli. Na taj način možemo i napisati funkciju gubitka generatora i time riješiti problem nestajućih gradijenata. Dobivene funkcije gubitka sada su:

$$J^{(D)} = -\mathbb{E}x \sim p_{data}(x)[logD(x)] - \mathbb{E}z \sim p_z(z)[log(1 - D(G(z)))] \quad (2.4)$$

$$J^{(G)} = -\mathbb{E}z \sim p_z(z)[log(G(D(z)))] \quad (2.5)$$

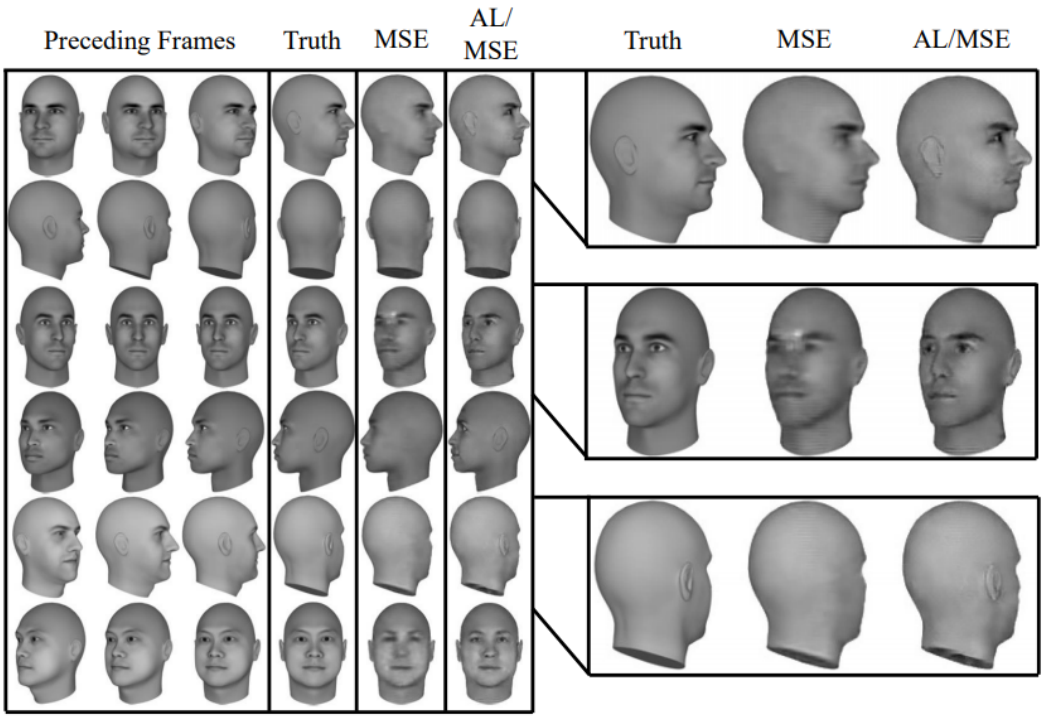
## 2.5. Usporedba s drugim pristupima

Kod generativnih modela cilj nam je generirati različite nove uzorke. Generativne suparničke mreže su veoma dobre upravo u tim slučajevima kada imamo više "točnih odgovora" odnosno kada smijemo generirati više različitih uzoraka, a jedini uvjet jest da izgledaju realno. Navedeno je prikazano na intuitivnoj razini na slici 2.2.




**Slika 2.2:** Ilustracija - usporedba srednje kvadratne pogreške i GAN-a.

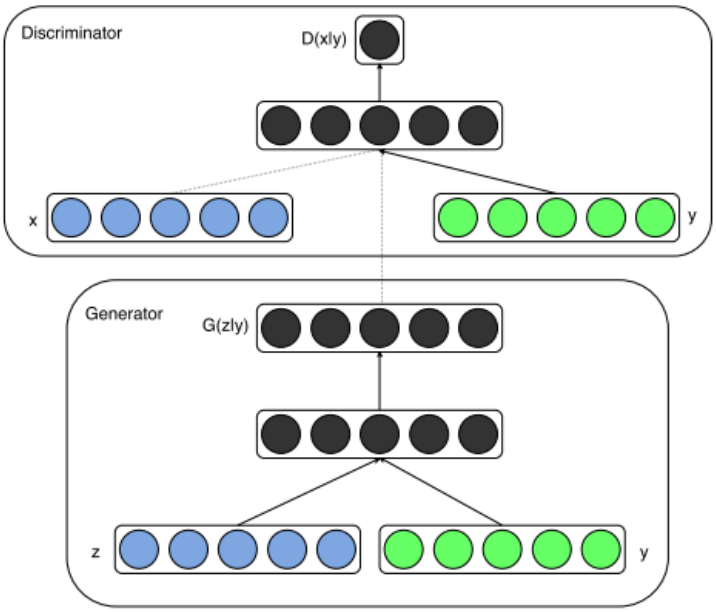
Na ilustraciji su prikazana dva točna rješenja. Pretpostavimo da je zelena točka broj 2 bila priložena oznaka za neki ulaz (plava točka), a naš model je kao rezultat dao zelenu točku broj 1. Ako koristimo srednju kvadratnu pogrešku, imat ćemo određeni gubitak ilustriran crvenom linijom, jer naša funkcija gubitka ne prepoznaje drugo točno rješenje. Nakon određenog vremena treniranja za određeni ulaz model će izbaciti srednju vrijednost svih točnih rezultata. Generativni suparnički modeli za učenje ne koriste parove ulaza i izlaza, već diskriminator uči kako se ulazi i izlazi mogu povezati te prihvaća različita točna rješenja. Na slici 2.3 vidimo kako to izgleda kod predviđanja sljedećeg okvira videozapisa. U gornjem desnom kutu možemo na primjeru uha, koje se može nalaziti na više bliskih pozicija, kako model trenirati srednjom kvadratnom pogreškom ne zna točnu lokaciju uha te na izlazu daje srednju vrijednost svih mogućih, što se očitava kao mutna slika u usporedbi s puno bistrijom slikom generiranom suparničkim modelom.




**Slika 2.3:** Predviđanje sljedećeg okvira za video rotirajućeg lica trenirana srednjom kvadratnom pogreškom (MSE) i suprotstavljenim mrežama (AL). [8]

# 3. Uvjetni suparnički modeli

Generativne suparničke mreže može se proširiti na uvjetni model ako su i generator i diskriminator uvjetovani nekom dodatnom informacijom $y$. $y$ može biti bilo koja vrsta pomoćnih informacija, kao što je naprimjer oznaka klase. Kondicioniranje možemo izvesti dodavanjem $y$ kao dodatni ulaz u generator i diskriminator.




**Slika 3.1:** Uvjetni suparnički model. [9]

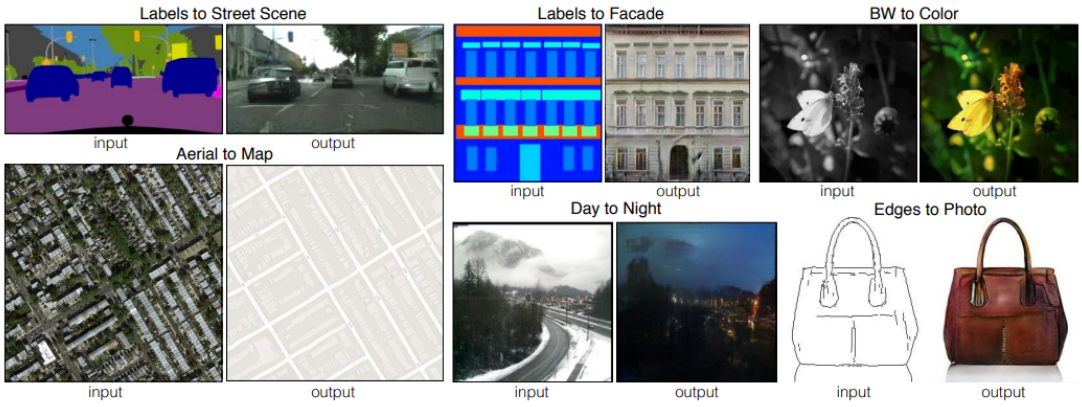
Funkciju cilja sada možemo prikazati kao:

$$\min_G \max_D V(D,G) = \mathbb{E}x \sim p_{data}(x)[\log D(x|y)] + \mathbb{E}z \sim p_z(z)[\log(1 - D(G(z|y)))]$$

Rezultati uvjetnih suparničkih modela ipak su često nadmašeni nekim drugim pristupima, pa čak i s običnim suparničkim modelima, no ipak imaju svoju primjenu kao što ćemo vidjeti u sljedećem poglavlju.

# 4. Prevođenje iz slike u sliku korištenjem uvjetnih suparničkih modela

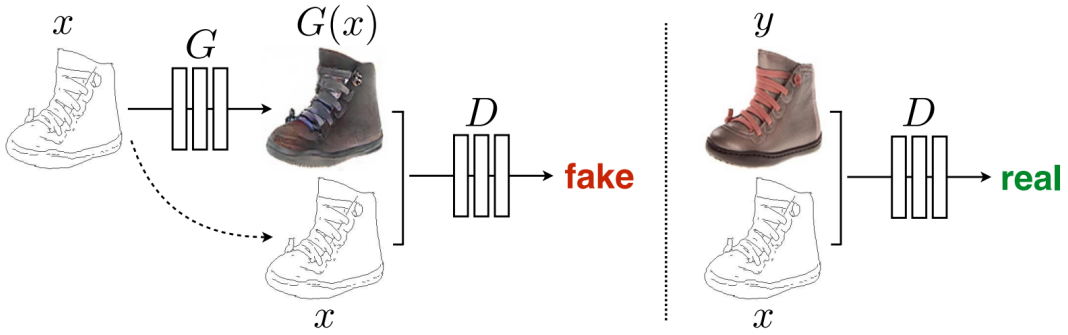
Mnoštvo problema na području računalnog vida i digitalne obrade i analize slike može se shvatiti kao prevođenje iz ulazne u izlaznu sliku, kao na primjer segmentacija slike, detekcija rubova i slično. Cilj autora rada[6] na kojem se bazira ovaj seminar jest ponuditi rješenje za sve probleme koji se baziraju na prevođenju iz slike u sliku uz istu arhitekturu mreže i funkciju cilja, a podatke za treniranje kao jedinu razliku.




**Slika 4.1:** Neki primjeri prevođenja iz slike u sliku. [6]

Iako su konvolucijske neuronske mreže sada već standardno rješenje za mnoge probleme računalnog vida, do problema nailazimo pri odabiru funkcije gubitka. Ranije smo pokazali da euklidska udaljenost između predviđenih i ciljnih vrijednosti nije dobra mjera zbog mutnih rezultata. Umjesto toga želimo cilj definirati na visokoj razini, poput "generiraj sliku koju ne možemo razlikovati od realnosti", i zatim automatski naučiti funkciju gubitka. To možemo postići korištenjem generativnih suparničkih modela, odnosno uvjetnih suparničkih modela gdje model

uvjetujemo ulaznom slikom, što je prikazano na slici 4.2. Treba spomenuti da postoje i radovi koji koriste neuvjetovane generativne suparničke modele, te se oslanjaju na metode poput L2 regularizacije kako bi izlaz uvjetovali na ulaz, koji također postižu dobre rezultate.




**Slika 4.2:** Skica uvjetnog suparničkog modela za prevođenje iz skiciranih rubnih linija u sliku. U usporedbi s GAN-om, u ovom modelu generator i diskriminator na ulazu dobivaju ulaznu sliku. Također, primijetimo da generator na ulazu ne dobiva slučajni šum $z$. Name, generator bi ga jednostavno naučio ignorirati, stoga autori šum dobivaju uz pomoć *dropouta* na nekolicini slojeva generatora. [6]

## 4.1. Arhitektura modela pix2pix

*pix*2*pix* model kao generator koristi arhitekturu baziranu na "U-Net" mreži[10], dok za diskriminator koristi konvolucijski "PatchGAN" klasifikator, sličan arhitekturi predloženoj u [7]. Različite implementacije dostupne su na službenom *pix*2*pix* github repozitoriju[2], a u ovom radu bit će opisane arhitekture mreža korištene u konkretnoj implementaciji na službenim stranicama tensorflowa[3].

### 4.1.1. Generator

Generator mapira ulaznu sliku visoke rezolucije u izlaznu sliku također visoke[1] rezolucije te su pritom globalne strukture ulaza i izlaza povezane, stoga koristimo mrežu s poveznicama sličnu mreži "U-Net" koja je prikazana na slici 4.3. *pix*2*pix* mreža je donekle izmijenjena. Sastoji se od 8 slojeva koji umanjuju(prepolavljaju) dimenziju slike (*en.downsampling*), te 7 slojeva koji dimenziju vraćaju na početnu (*en.upsampling*).

[1]U slučaju datasetova sa [2] rezolucija je 256x256.

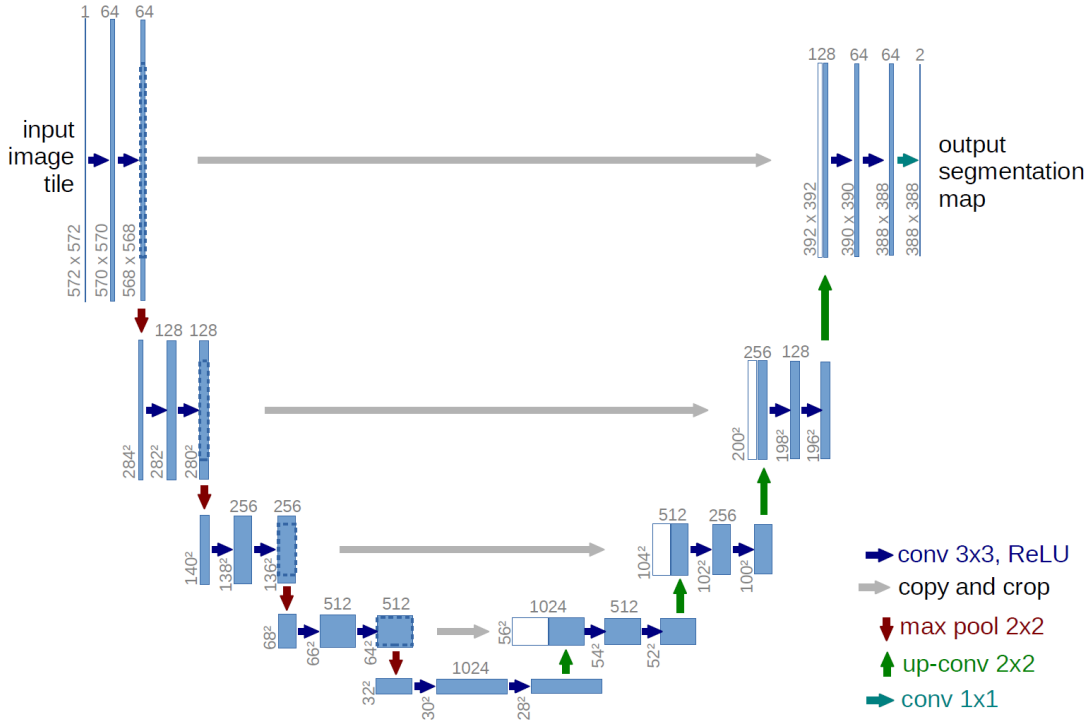
Riječ je o konvolucijskim slojevima sa sljedećim svojstvima:

- jezgre 4x4
- nadopuna rubova nulama (*samepadding*)
- korak 2
- leakyReLU aktivacija

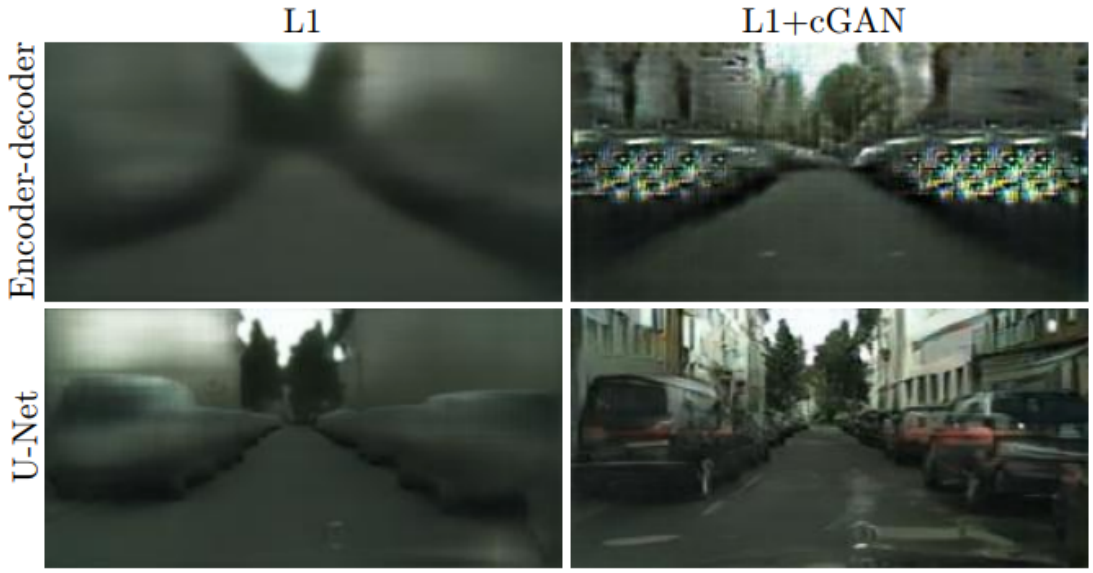
Također, u svakom od prvih 8 slojeva, izuzev prvog, radimo normalizaciju nad grupom, dok u prva 3 sloja drugog dijela mreže unosimo šum korištenjem *dropouta*. Niti u jednom sloju ne pridodajemo prag (en. bias). Izlaz svakog *downsampling* sloja pridodaje se izlazu odgovarajućeg *upsampling* sloja kao što je prikazano na 4.3. Takve veze poprilično poboljšavaju performanse generatora, kao što je vidljivo na 4.4. Završni sloj generatora je *upsampling* sloj koji rezoluciju slike vraća na početnu, broj kanala određen je zahtijevanim brojem kanala izlazne slike, a kao aktivacijska funkcija koristi se tangens hiperbolni. U nastavku su prikazane veličine jezgara i dimenzija izlaza pojedinog sloja.

```
downsampling
    64  #     128, 128, 64      Bez normalizacije nad grupom
    128 #     64, 64, 128
    256 #     32, 32, 256
    512 #     16, 16, 512
    512 #     8, 8, 512
    512 #     4, 4, 512
    512 #     2, 2, 512
    512 #     1, 1, 512


upsampling
    512 #     2, 2, 1024        Uz korištenje dropouta
    512 #     4, 4, 1024        Uz korištenje dropouta
    512 #     8, 8, 1024        Uz korištenje dropouta
    512 #     16, 16, 1024
    256 #     32, 32, 512
    128 #     64, 64, 256
    64  #     128, 128, 128
    3   #     256, 256, 3
```



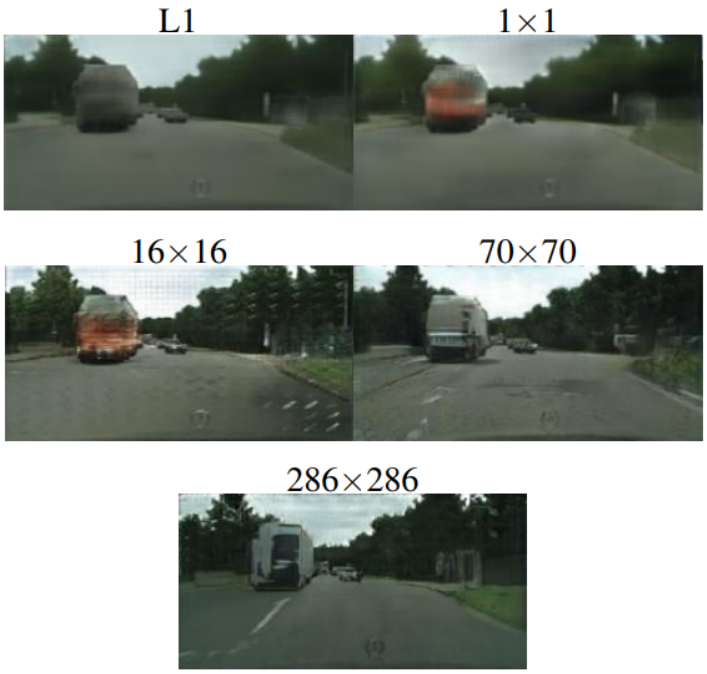

**Slika 4.3:** Primjer arhitekture "U-Net" mreže iz [10].




**Slika 4.4:** Prikaz razlike u performansama pri korištenju L1 i cGAN + L1 funkcije gubitka, te veliko poboljšanje rada generatora uz pomoć *skip* veza. [6]

### 4.1.2. Diskriminator

Diskriminator se temelji na "PatchGAN" klasifikatoru. Naša će mreža promatrati podudaranje strukture na nešto nižoj razini, dok odstupanje u globalnoj strukturi kažnjavamo L1 regularizacijom. To možemo postići na način da naš diskriminator promatra odlomke (*en.patches*) slike veličine NxN i za svaki odlomak donosi sud je li generiran ili stvaran. Diskriminator "pomićemo" poput konvolucijske jezgre sve dok ne obradi sve dijelove slike. U ovoj konkretnoj implementaciji uzimaju se dijelovi slike dimenzija 70x70 što na kraju donosi matricu rezultata dimenzije 30x30, koja se uprosječuje kako bi dobili konačnu odluku je li ulazna slika generirana ili stvarna.




| Discriminator receptive field | Per-pixel acc. | Per-class acc. | Class IOU |
|---|---|---|---|
| **1×1** | 0.39 | 0.15 | 0.10 |
| **16×16** | 0.65 | 0.21 | **0.17** |
| **70×70** | **0.66** | **0.23** | **0.17** |
| **286×286** | 0.42 | 0.16 | 0.11 |

**Slika 4.5:** Utjecaj različitih veličina *patcheva* u PatchGAN-u. Pri korištenju L1 nesigurne regije su mutne i bezbojne. 1x1 PixelGAN doprinosi različitosti boja. Uz pomoć 16x16 vidljive su jasne lokalne regije, ali globalno gledano stvara se efekt popločavanja. PatchGAN 70x70 donosi jasne i obojane rezultate, kao i potpuni 286x286. Na tablici ispod slika vidljivo je da su performanse PatchGAN-a 70x70 ipak bolje. [6]

## 4.2. Učenje

Iz ranije spomenutih razloga, $pix2pix$ funkciji gubitka uvjetnog suparničkog modela pridodaje L1 regularizaciju, pa je krajnji izraz:

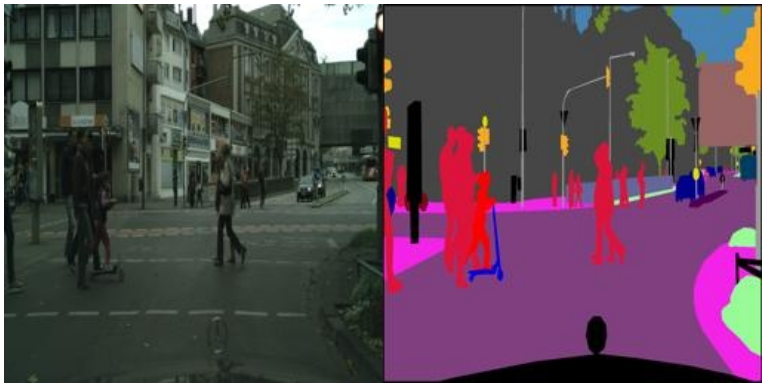
$$G^* = \underset{G}{arg\,min}\,\underset{D}{max}\, L_{cGAN}(G, D) + \lambda L_{L1}(G). \tag{4.1}$$

Naizmjenično se vrši jedan gradijentni spust na diskriminatoru i jedan na generatoru, s time da je funkcija cilja diskriminatora podijeljena sa 2 kako bi se usporila brzina učenja diskriminatora u usporedbi s generatorom.

Programski postupak učenja opisan je na [2] i [3].

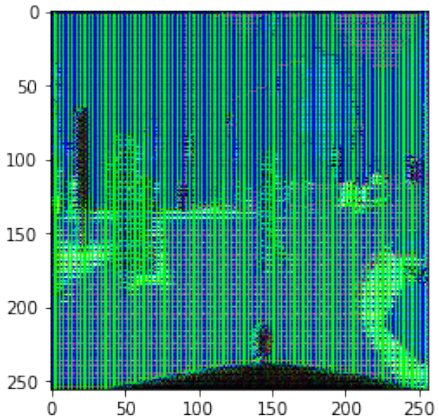
## 4.3. Ulazni podatci i treniranje

U sklopu pix2pix github repozitorija predloženi su sljedeći datasetovi: facades[2], cityscapes[3], maps[4], edges2shoes[5], edges2handbags[6] i night2day[7]. Za potrebe demonstracije rada modela u sklopu ovog rada korišten je skupi cityscapes[4].



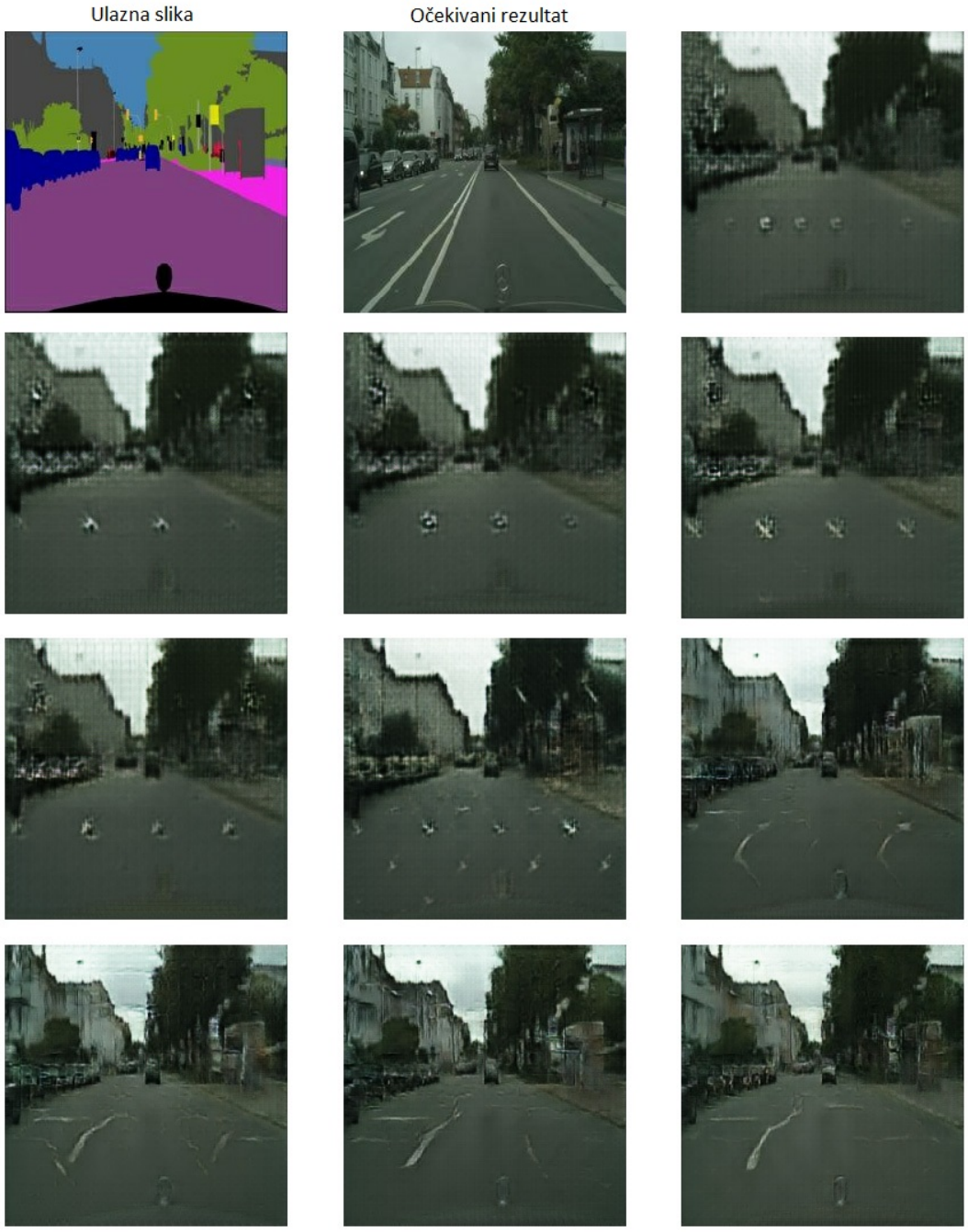
**Slika 4.6:** Primjer slike iz skupa za treniranje. [4]

Na slici 4.6 vidimo primjer slike iz skupa za treniranje. Primijetimo da pri treniranju model zahtjeva na ulazu i pravu i segmentiranu verziju slike. Na slici 4.7 prikazan je izlaz iz generatora u prvoj iteraciji treniranja.




**Slika 4.7:** Izlaz iz generatora u prvoj iteraciji treniranja.

[2]http://cmp.felk.cvut.cz/ tylecr1/facade/
[3]https://www.cityscapes-dataset.com/
[4]Slike uzorkovane sa Google Mapsa
[5]http://vision.cs.utexas.edu/projects/finegrained/utzap50k/
[6]https://github.com/junyanz/iGAN
[7]http://transattr.cs.brown.edu/

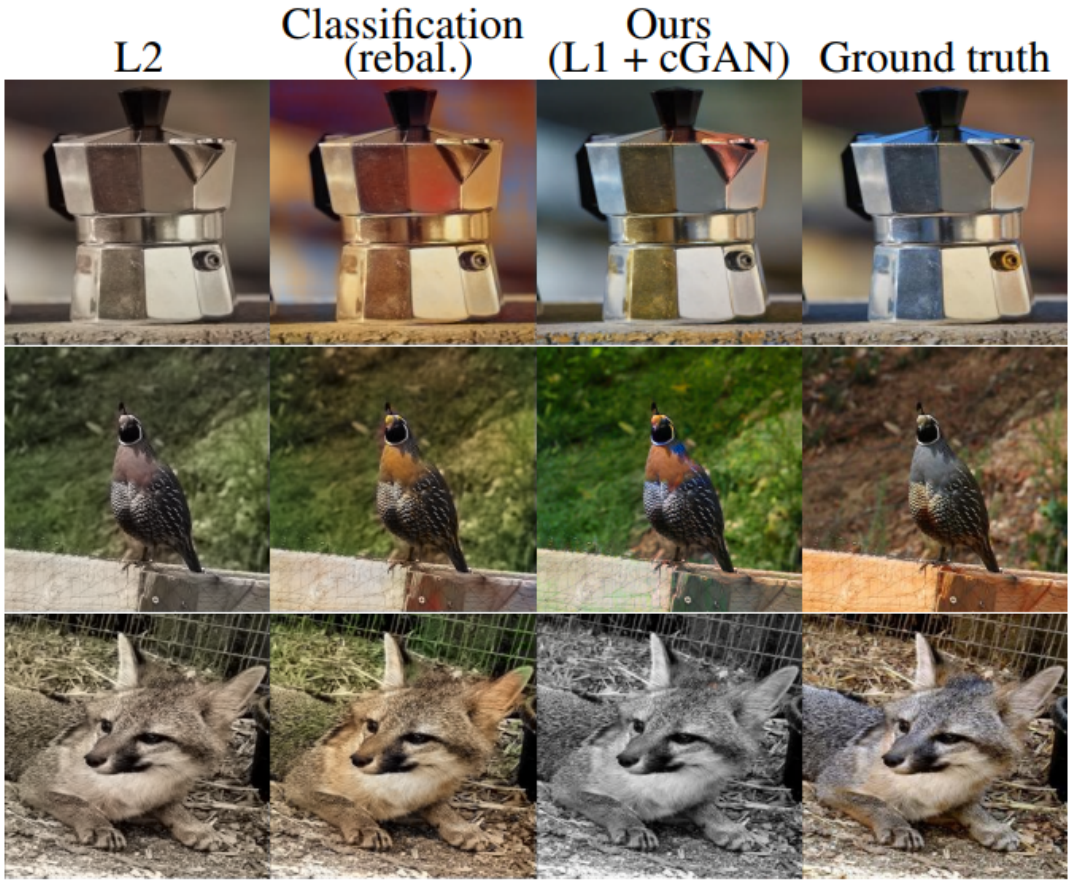



**Slika 4.8:** Prikaz učenja kroz epohe (s preskokom, raspon 1 - 141).

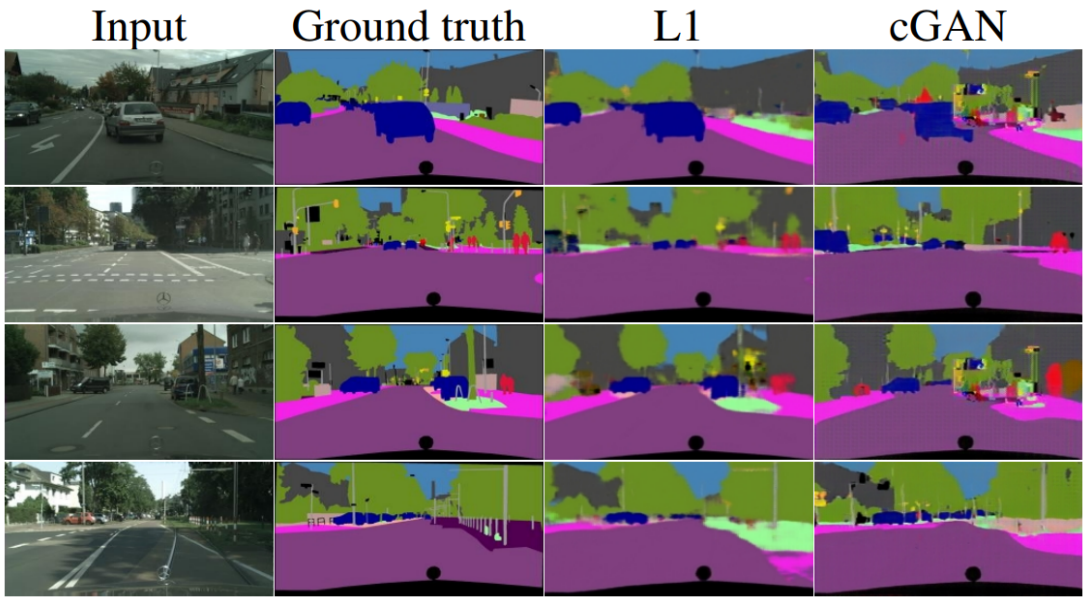
# 5. Eksperimentalni rezultat

Autori rada su generirane slike ocjenjivali uz pomoć usluge *AmazonMechanicalTurk*, postupak je detaljnije opisan u radu. Također, za evaluaciju su se nad generiranom slikom koristili tradicionalni modeli za segmentaciju slika, gdje je pretpostavka bila da će na kvalitetno generiranoj slici model dobro funkcionirati. U nastavku su prikazani dobiveni rezultati na različitim problemima.

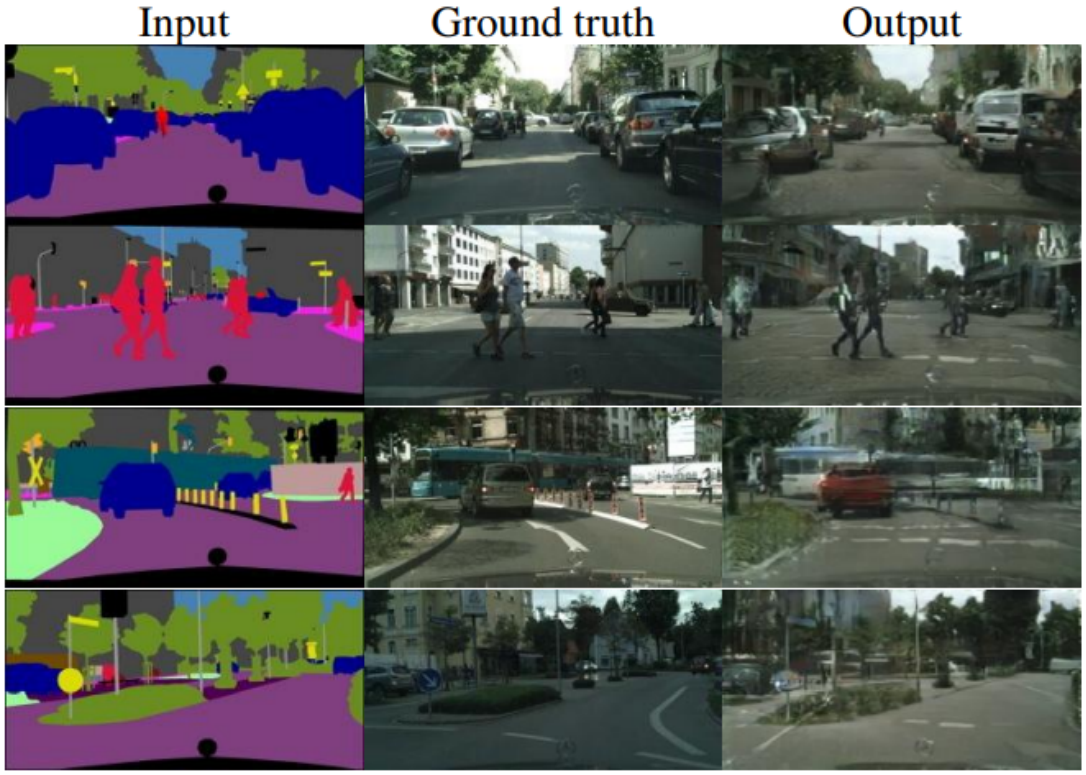

**Slika 5.1:** Primjeri rezultata dobivenih na Google Maps slikama rezolucije 512x512 (model je treniran na slikama rezolucije 256x256). Generirane satelitske snimke uspjele su prevariti ispitanike u 18.9% slučajeva, dok je pretvorba iz satelitskih snimaka u karte bila manje uspješna i prevarila je ispitanike u svega 6.1% slučajeva. [6]



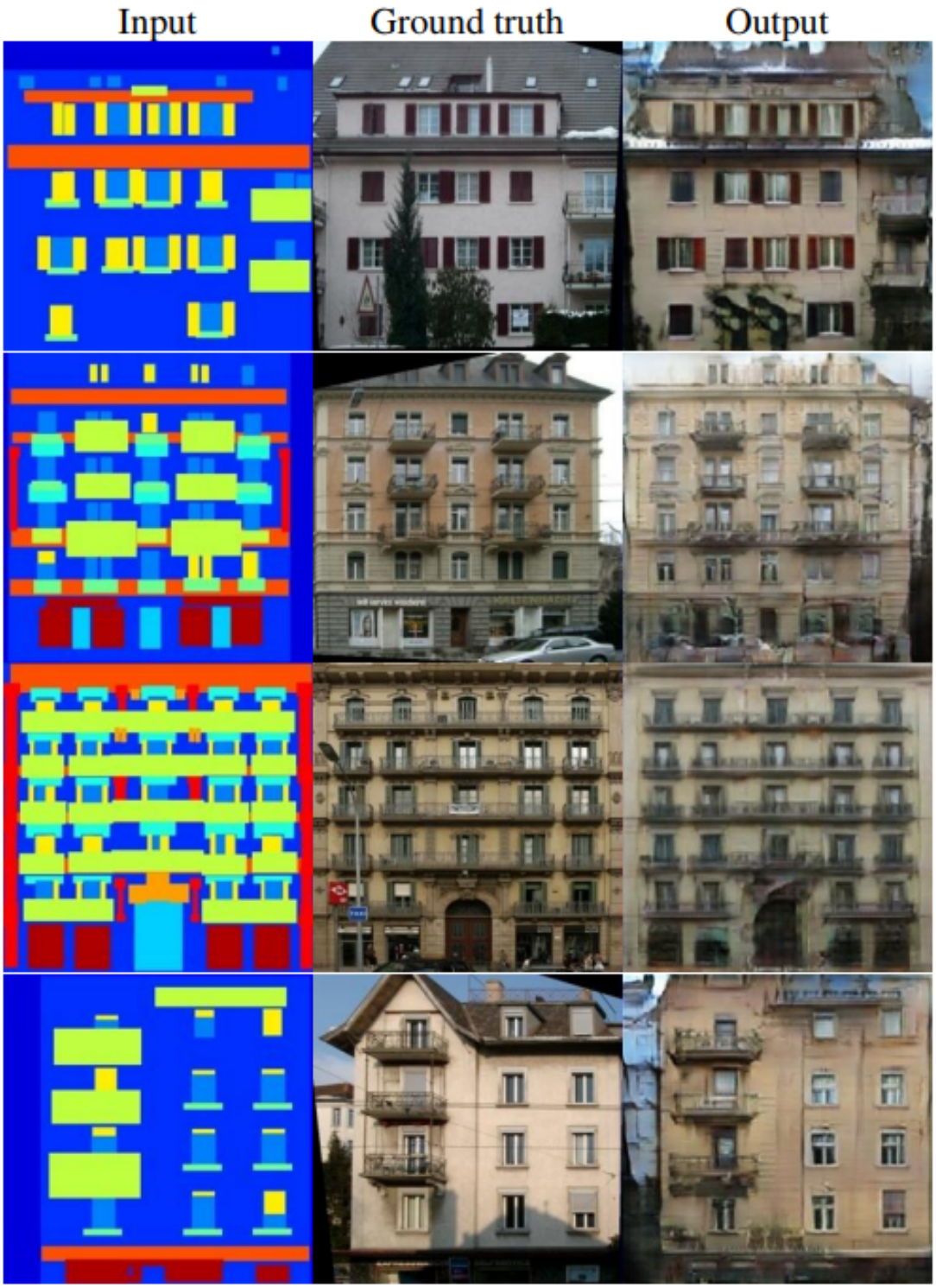

**Slika 5.2:** Rezultati retuširanja crno-bijelih fotografija. U zadnjem retku je prikazan neuspješan (bezbojan) rezultat. [6, 13]




**Slika 5.3:** Rezultati prevođenja iz cityscape slika u segmentirani oblik. [6]




**Slika 5.4:** Rezultati prevođenja iz segmentiranih reprezentacija slika u stvarne. [6]




**Slika 5.5:** Rezultati prevođenja iz skica u zgrade. [6]

# 6. Zaključak

Kroz ovaj seminarski rad dotakli smo se osnova generativnih suparničkih modela i pix2pix-a kao jedne od mogućih primjena takvih modela. Model pix2pix pokazao se kao svestran i prilagodljiv model koji ostvaruje dobre rezultate. Postoji još mnoštvo različitih verzija generativnih suparničkih modela kojih se nismo dotaknuli poput DCGAN-a, CycleGAN-a, BigGAN-a, StyleGAN-a, StackGAN-a i drugih. U trenutku pisanja generativni suparnički modeli su nova, interesantna i aktivna tema, te se novi pristupi i unaprjeđenja objavljuju iz mjeseca u mjesec. Za kraj valja spomenuti da će daljnji napredak ovakvih modela zasigurno potaknuti mnoga moralna i etička pitanja, kao što su neki modeli[1] to već napravili, pogotovo u trenutku kada ljudi ne budu mogli sa sigurnošću reći je li određena slika ili video stvaran ili umjetno generiran.

[1] https://en.wikipedia.org/wiki/Deepfake

# Literatura


[1] *A Beginner's Guide to Generative Adversarial Networks (GANs).* URL https://skymind.ai/wiki/generative-adversarial-network-gan.

[2] *Pix2Pix official github repository.* URL https://github.com/phillipi/pix2pix.

[3] *Pix2Pix notebook*, 2019. URL https://www.tensorflow.org/alpha/tutorials/generative/pix2pix.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, i Bernt Schiele. *The Cityscapes Dataset for Semantic Urban Scene Understanding*, 2016.

[5] Ian Goodfellow. *Introduction to GANs, Predavanje na konferenciji NIPS 2016.*, 2016. URL http://www.iangoodfellow.com/slides/2016-12-9-gans.pdf.

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, i Alexei A. Efros. *Image-to-Image Translation with Conditional Adversarial Networks*, 2016. URL https://arxiv.org/abs/1611.07004.

[7] Chuan Li i Michael Wand. *Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks*, 2016. URL https://arxiv.org/abs/1604.04382.

[8] William Lotter, Gabriel Kreiman, i David Cox. *Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning*, 2016. URL https://arxiv.org/abs/1605.08104.

[9] Mehdi Mirza i Simon Osindero. *Conditional Generative Adversarial Nets*, 2014. URL https://arxiv.org/abs/1411.1784.

[10] Olaf Ronneberger, Philipp Fischer, i Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*, 2015. URL `https://arxiv.org/abs/1505.04597`.

[11] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, i Xi Chen. *Improved Techniques for Training GANs*, 2016. URL `https://arxiv.org/abs/1606.03498`.

[12] Saining "Xie i Zhuowen" Tu. *Holistically-Nested Edge Detection*, 2015.

[13] Richard Zhang, Phillip Isola, i Alexei A. Efros. Colorful image colorization. 2016. URL `http://arxiv.org/abs/1603.08511`.

[14] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, i Alexei A. Efros. *Generative Visual Manipulation on the Natural Image Manifold*, 2016.

**Prevođenje iz slike u sliku korištenjem uvjetnih suparničkih modela**

**Sažetak**

Kroz rad je prikazan osnovni pregled generativnih i uvjetnih suparničkih modela, te korištenje istih u svrhu prevođenja iz slike u sliku. Opisana je arhitektura modela pix2pix i njegove performanse, te su prikazani eksperimentalni rezultati.

**Ključne riječi:** računalni vid, duboko učenje, neuronske mreže, generativni suparnički modeli, GAN, uvjetni suparnički modeli, CAN, slika-u-sliku, prevođenje slika