Generisanje logoa koristeći GAN

Milica Golubović Divna Mićić

Februar 2024

Sadržaj

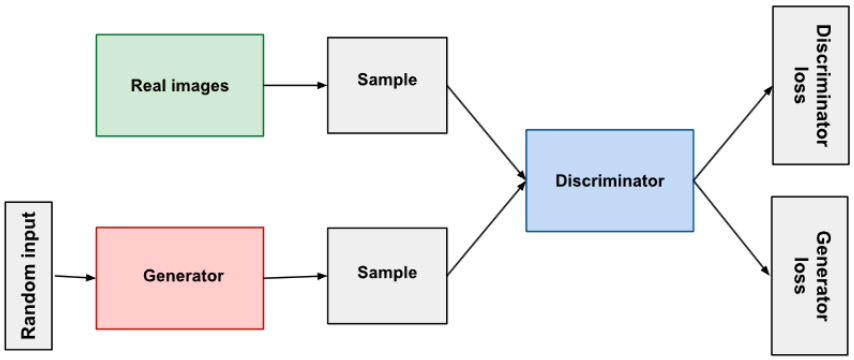
1 Uvod

Dizajniranja logoa je dugačak proces koji zahteva dosta saradnje između dizajnera i klijenta. Za svaku novu skicu potrebno je uložiti trud i vreme, što ceo ovaj proces čini komplikovanim i skupim. Razvojem novih tehnologija i dubokog učenja, omogućeno je da rešavanje ovog problema bude automatizovano. Jedan od načina kako to možemo da realizujemo je pomoću Generativnih kontradiktornih mreža (*eng. GAN*). Pored generisanja slika, GAN ima široku primenu u oblasti detekcije objekata, generisanje anime likova [1], nivoa u igricama (*Super Mario*) [2], previđanje frejmova u videu [3], generisanje slika visokih rezolucija [9], generisanje slika na osnovu teksta, uklanjanje šuma na slikama [5] itd.

2 Generisanje logoa pomoću GAN-a

2.1 Uopšteno o GAN-u

GAN (*eng. Generative Adversarial Networks*) tj. generativna kontradiktorna mreža je model mašinskog učenja koji je prvi put predstavio Ian Goodfellow 2014. godine [4]. U njemu se dve neuronske mreže takmiče koja će imati preciznije predviđanje. Jednu mrežu predstavlja generator, koji pokušava da napravi realistične slike i time prevari diskriminator. Drugu mrežu predstavlja diskriminator koja pokušava da razlikuje prave od lažno generisanih slika. Ovaj princip poznat je i kao igra nulte sume, gde gubitak jedne strane predstavlja dobitak druge. Naziv kontradiktorna proizilazi iz toga što se mogu analizirati alatima teorije igara.[7]

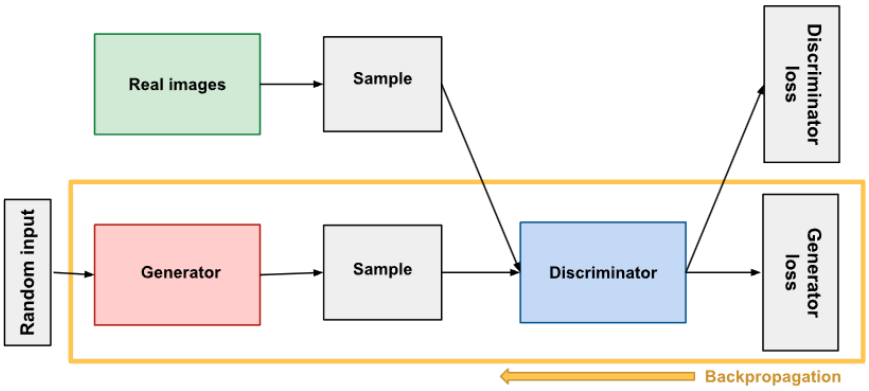


Slika 1: *Struktura GAN-a*

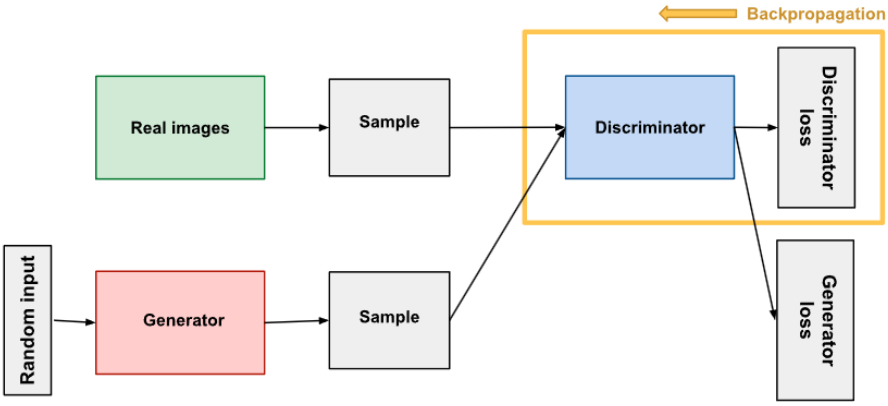
2.2 Detalji modela

Neuronskim mrežama je potrebna neka vrsta ulaza, kako bi mogle da generišu izlaze.

Generator G koristi šum z kao ulaz da bi generisao instancu $G(z)$. Nakon toga diskriminator D prima podatke od dva izvora; realni podaci instanca x iz skupa *pdata*; koriste se kao pozitivni primer tokom obučavanja i lažni podaci tj. izlaz generatora $G(z)$; koriste se kao negativni primer. Potom diskriminator klasifikuje generisanu instancu i kao izlaz vraća verovatnoću kojom je slika prava odnosno lažna. Ta informacija se povratnom spregom vraća do generatora i pomoću nje se dalje koriguju parametri modela.



Slika 2: *Povratna sprega za generator*



Slika 3: *Povratna sprega za diskriminator*

GAN koristi Minmax funkciju gubitka (kod generatora) gde generator pokušava da minimizuje datu funkciju, a diskriminator da je maksimizuje.

$$min_G max_D V(D,G) = E_{x\sim pdata}[logD(x)] + E_{z\sim pz}[log(1\text{-}D(G(z)))]$$

E_x predstavlja očekivanu vrednost za skup realnih podataka *pdata*, *D(x)* verovatnoća očekivanja za diskriminator ako je x pravo, *G(z)* je izlaz generatora za zadati šum z, *D(G(z))* je očekivana verovatnoća diskriminatora ako je lažno generisana slika prava, E_z je očekivanje za sve ulaze generatora tj. šumove.

Za diskriminator se koristi standardna unakrsna entropija.

Pošto ne želimo da nam se modeli potpuno menjaju mi ćemo umesto vrednosti koje vraćaju funkcije gubitka koristiti njihove gradijente.

Kako se generator poboljšava, diskriminator postaje gori jer ne može lako da uoči razliku između pravih i lažnih ulaza. Ako je generator uspešno generisao sliku identičnu pravoj onda diskriminator ima 50% tačnosti tj diskriminator baca novčić kako bi odlučio da li je slika prava ili lažna.

2.3 Problemi

Tokom treniranja mreže možemo naići na nekoliko problema [8]. Probaćemo da ih opišemo:

- Kolaps modela - U većini slučajeva želimo raznovrsne izlaze generatora. Kolaps modela se dešava kada generator proizvodi jednu te istu sliku ili skup sličnih slika. Ovo se dešava zato što je generator stalno u potrazi za slikom koja će najbolje prevariti diskriminator i onda proizvodi samo jedan izlaz.
- Nestajući gradijenti - Ako je diskriminator previše dobar kao izlaz vraća gradijente blizu nuli. Ta informacija je od malog značaja za generator i može usporiti ili zaustaviti njegovo učenje.

2.4 DCGAN

Za potrebe našeg problema mi ćemo koristiti modifikaciju GAN-a poznatu kao DCGAN *(Deep Convolutional GAN)*. Glavna karakteristika koja ih razlikuje od običnog GAN-a je korišćenje dubokih konvolutivnih neurnoskih mreža u generatoru i diskriminatoru. Navešćemo još nekiliko ključnih izmena:

1. U generatoru konvolucije su zamenjene transponovanim konvolucijama - služi za generisanje slike od šuma, počinjemo od gustog sloja pa ga povećavamo nekoliko puta dok ne dobijemo željenu veličinu slike.[6]
2. Korišćenje batch normalizacije u generatoru i diskriminatoru
3. Korišćenje ReLu aktivacione funkcije u svim slojevima osim za izlaz gde korisitmo Tanh, a u diskriminatoru korsitimo LeakyReLu u svim slojevima.

4. Korisitmo Adam kao optimizator umesto SGD

3 Pregled rezultata

Svi modeli su pokretani na mašinama sa sledećim specifikacijama:

- RAM - 16GB, procesor - AMD R7 3.10GHz, grafika - NVIDIA GT 1030
- RAM - 12GB, procesor - Intel i3-7020U 2.30GHz, grafika - Intel HD 620

3.1 2000 slika 1000 epoha

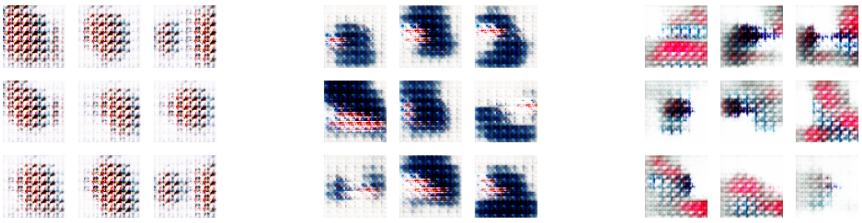
Prvi posmatrani uzorak se sastoji od 2000 različitih slika koje na sebi imaju neki logo. Te slike su podeljene u 25 kategorija, odnosno tema.

S obzirom na to koje je veličine dataset, izabrani batch je veličine 256.

Korišćeni slojevi za generator su Conv2DTranspose, praćeni normalizacijom i aktivacionim slojem LeakyReLU, čiji je parametar alpha=0.01. Redom imamo 128, 256, 64 i 3 filtera.

Diskriminator se sastoji od slojeva Conv2D, LeakyReLU sa parametrom alpha=0.01 i Dropout čiji je rate=0.3. Broj filtera je redom 128 i 64.

Trajanje jedne epohe je 1 minut i 54 sekunde. Ukupno izvršavanje svih 1000 epoha je 34 sata i 25 minuta.



Epoha 100 *Epoha 622* *Epoha 895*

Slika 4: *Rezultati različitih epoha*

3.2 18 slika 4750 epoha - Pikachu

Pošto na prethodnom primeru nismo dobili vizuelno zanimljive rezultate, modele smo malo prilagodili i testirali ih na manjem skupu. Posmatrali smo skup koji se sastoji od 18 slika, na kojima se nalazi Pikachu.

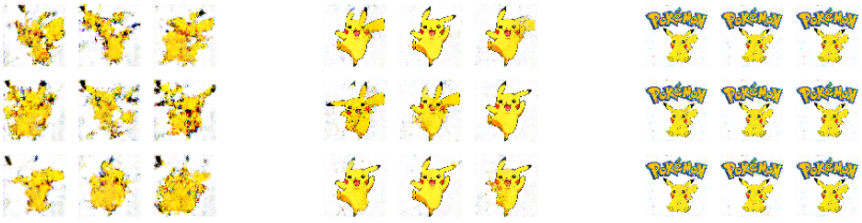
Zbog manjeg skupa slika, biramo da nam je batch veličine 2.

Slojevi koji čine generator su Conv2DTranspose, sa brojem filtera 64, 128, 256 i 3, BatchNormalization i LeakyReLU, čiji je parametar alpha=0.01.

Diskriminator se sastoji od slojeva Conv2D, gde je broj filtera 64, 128 i 256, LeakyReLU gde je alpha=0.2 i Dropout, čiji je rate=0.3.

Izvršavanje jedne epohe je trajalo 12 sekundi, dok se 4750 epoha izvršavalo 14 sati i 1 minut.

Ovaj put smo kao rezultat dobili smislene slike, gde je očigledno šta se na njima nalazi.



Epoha 1002 *Epoha 2103* *Epoha 4751*

Slika 5: *Rezultati različitih epoha*

Iako je naša tema vezana za logo, ovde vidimo da se algoritam može uspešno primeniti i na slike sa drugačijim temama.

3.3 64 slike 5000 epoha

Sledeći posmatrani uzorak je set od 64 slike, na kojima se nalazi neka varijanta Toyotinog prepoznatljivog logoa. Na ovom skupu smo testirali dve različite verzije modela. U oba slučaja biramo da nam je batch veličine 8.

3.3.1 Toyota 1

Kako smo ustanovili da su modeli iz prošlog primera dali interesantne rezultate, odlučili smo da ih testiramo i na ovom skupu.

Generator ima slojeve Conv2DTranspose, BatchNormalization i LeakyReLU, sa parametrom alpha=0.01. Broj filtera je redom 64, 128, 256 i 3.

Diskriminator je kombinacija slojeva Conv2D, sa 64, 128 i 256 filtera, LeakyReLU čiji je parametar alpha=0.2 i Dropout sa parametrom rate=0.3.

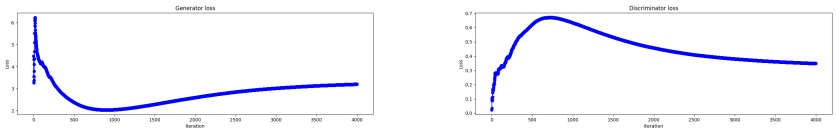
Vreme izvršavanja svih 5000 epoha je 7 sati i 40 minuta, pri čemu je izvršavanje jedne epohe trajalo 6 sekundi.

Epoha 100 *Epoha 2825* *Epoha 4831*

Slika 6: *Rezultati različitih epoha*

Kako bismo mogli bolje da uporedimo efikasnost ovih modela sa sledećim, čuvali smo i grafike koji predstavljaju gubitke generatora i diskriminatora, kao i tačnost diskriminatora pri određivanju lažnih i pravih slika.

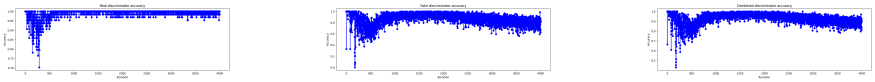
Na slikama se vidi kako se gubici menjaju dok naš model uči. Kada imamo male gubitke generatora, tada imamo velike gubitke diskriminatora i obrnuto.



Gubici generatora *Gubici diskriminatora*

Slika 7: *Gubici*

Sa grafika tačnosti se vidi da diskriminator na početku svog učenja ima lošiju tačnost u prepoznavanju i lažnih i pravih slika. Vremenom diskriminator počinje bolje da određuje prave slike i tačnost mu je bliža jedinici. Što se tiče lažnih slika, kako kroz epohe generator postaje bolji, tako tačnost diskriminatora počinje da opada i polako se kreće ka idealnoj teorijskoj vrednosti od 0.5.



Prave vrednosti *Lažne vrednosti* *Kombinovana tačnost*

Slika 8: *Tačnost diskriminatora u određivanju pravih i lažnih slika*

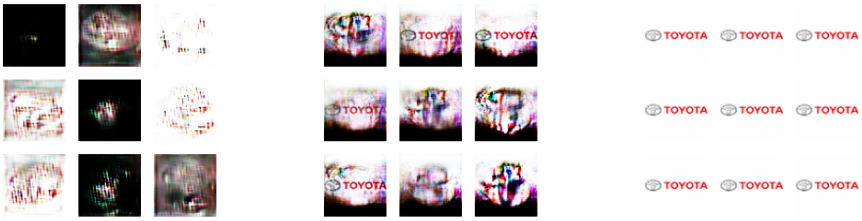
3.3.2 Toyota 2

Sledeće smo promenili slojeve koji čine modele generatora i diskriminatora i testirali njihovo ponašanje.

Generator se ovaj put sastoji od kombinacije slojeva UpSampling2D, Conv2D, BatchNormalization i ReLU. Broj filtera je redom 128, 64 i 3.

Za model diskriminatora su korišćeni slojevi Conv2D sa 64, 128 i 256 filtera, LeakyReLU sa parametrom alpha=0.2 i MaxPooling2D.

Vreme izvršavanja jedne epohe trajalo je 27 sekundi. Ukupno trajanje svih 5000 epoha je 36 sati i 27 minuta.

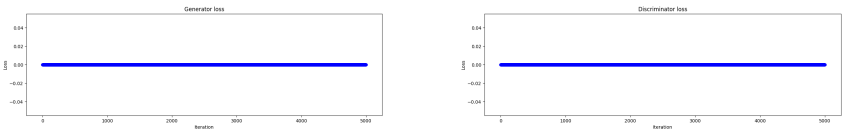


Epoha 180 *Epoha 1140* *Epoha 4840*

Slika 9: *Rezultati različitih epoha*

Osim što je vreme izvršavanja u ovom slučaju značajno duže u odnosu na prethodni primer, može se primetiti i razlika u dobijenim graficima.

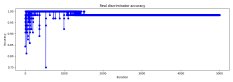
Gubici generatora i diskriminatora se u ovom primeru ne menjaju značajno kroz epohe, već se sve vreme nalaze oko nule. Za razliku od prošlog primera, ne vidi se da ovako izmenjeni modeli uče nove stvari i međusobno se unapređuju.



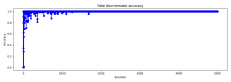
Gubici generatora *Gubici diskriminatora*

Slika 10: *Gubici*

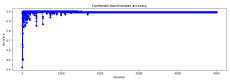
Na graficima tačnosti se ponovno u početku vidi lošija tačnost diskriminatora, ali ni s vremenom ona ne postaje bolja. Diskriminator u ovom slučaju sa velikom sigurnošću pogađa koje slike su lažne, čak bolje nego što to radi za prave slike. Odavde možemo zaključiti da je prethodni primer imao bolje modele generatora i diskriminatora, bilo po parametru brzine izvršavanja ili prema relevantnim metrikama kao što su gubici i tačnost odgovarajućih modela.



Prave vrednosti



Lažne vrednosti



Kombinovana tačnost

Slika 11: *Tačnost diskriminatora u određivanju pravih i lažnih slika*

4 Zaključak

Različiti rezultati se mogu dobiti u zavisnosti od toga koje slojeve biramo da koristimo u modelima generatora i diskriminatora. Kako je najveći problem dobro izbalansirati njihov rad, neki modeli se mogu pokazati kao bolji u odnosu na druge, što se može videti i u našim rezultatima.

Uzimajući u obzir mogućnosti našeg hardvera, jasno je da zbog toga postoje i određena ograničenja pri vršenju eksperimenata. Tako se ovi modeli mogu unaprediti dodavanjem većeg broja filtera u (transponovanim) konvolutivnim slojevima, kao i izvršavanjem nad većim skupom slika, po mogućnosti sa više različitih kategorija. Takođe, izvršavanjem većeg broja epoha, možemo dobiti relevantnije i preciznije rezultate.

5 Literatura

1. Y. Jin, J. Zhang, M. Li, Y. Tian, H. Zhu, and Z. Fang, *"Towards the automatic anime characters creation with generative adversarial networks"*, unpublished

2. V. Volz, J. Schrum, J. Liu, S. M. Lucas, A. M. Smith, and S. Risi, *"Evolving mario levels in the latent space of a deep convolutional generative adversarial network"*, in Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2018). New York, NY, USA: ACM, July 2018. [Online]. Available:

3. *"FutureGAN: Anticipating the Future Frames of Video Sequences using Spatio-Temporal 3d Convolutions in Progressively Growing GANs."* 2 October 2018.

4. Goodfellow, Ian J., Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron C., and Bengio, Yoshua. *"Generative adversarial nets"*. NIPS, 2014.

5. Zhang, H.; Sindagi, V.; Patel, V.M. *"Image de-raining using a conditional generative adversarial network"*. IEEE Trans. Circuits Syst. Video Technol. 2019, 30, 3943–3956

6. *"Deep Convolutional Generative Adversarial Network "* Tensorflow

7. *"NIPS 2016 Tutorial:Generative Adversarial Networks"* Ian Goodfellow OpenAI 2016

8. *"A review of Generative Adversarial Networks (GANs) and its applications in a wide variety of disciplines – From Medical to Remote Sensing"* ANKAN DASH, JUNYI YE, and GUILING WANG, Department of Computer Science, New Jersey Institute of Technology, USA

9. Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. 2017. *"Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. "*