GAN mreže

Dijana Ivezić

Fakultet elektrotehnike, računarstva i informacijskih tehnologija

Osijek, 2021

Sadržaj:

1. Uvod

1.1. Kratak opis GAN mreža

Neuronske mreže se često koriste pri klasičnim problemima klasifikacije i regresije. Takvi modeli su dobro poznati te kao prvi projekt sa dubokim mrežama se nerijetko klasificiraju brojevi iz MNIST baze podataka kako bi se objasnili novi pojmovi. Trenirati takvu mrežu je relativno jednostavno i brzo, a točan odgovor koji očekujemo od mreže je jednoznačno određen. Ukoliko se na slici nalazi broj x, očekujemo da mreža za taj broj x daje najveću vjerojatnost da se nalazi na slici, a ostale vjerojatnosti teže nuli.

Često se susrećemo s novim problemima za koje nije vjerojatno da ćemo znati točan odnos ulaznih i izlaznih podataka, a broj ulaznih varijabli može biti izuzetno velik i nepoznat. Na primjeru MNIST brojeva, vrlo teško možemo jednoznačno odrediti što svaki broj čini tim brojem, a da ta funkcija pokriva i još neviđene brojeve sa relativno dobrom preciznošću. Za navedeni primjer se uspješno koriste konvolucijske neuronske mreže relativno niske složenosti. Takvi modeli se nazivaju diskriminirajući modeli koji daju vjerojatnost pojave neke klase, ali ne znaju način na koji se takvi objekti kreiraju.

Za razliku od prethodnog primjera, zanima nas naučeno generiranje podataka na temelju naučenih značajki prilikom treniranja. Takav problem je nešto složeniji i zahtjeva kombinaciju dvije konvolucijske neuronske mreže koje zajedno natječući se dolaze do željenog ponašanja. Pojam koji opisuje takve mreže je *generative adversarial network* (generirajuća natječuća mreža u slobodnom prijevodu) koji će biti opisan u narednim poglavljima. GAN (skraćeno) ima svoje specifičnosti zbog kojih se teže treniraju, ovdje će biti dan osvrt na neke od njih.

Svrha navedenih mreža je generiranje uvjerljivih novih podataka iz nekog skupa od interesa s tim da ti novi podatci izgledaju kao da su iz originalnog skupa za treniranje, a da nisu jednostavno kopija pokazanih podataka tijekom treniranja. Glavno svojstvo GAN mreža i dubokih neuronskih mreža općenito je mogućnost prepoznavanja povezanih značajki i razumijevanje konteksta. U slučaju slika, umjesto da model računa vjerojatnosti pojave piksela određene boje na određenom mjestu, model uči važne odnose između piksela. Ako promatramo ponovno brojeve, mreža uči kreirati neki nasumični broj (ili čak zadani broj) koji osoba može pročitati umjesto da oboja piksele bijelo otprilike u sredini jer je visoka vjerojatnost da bi bili obojani. U tom slučaju to ne bi uopće izgledalo kao broj i takav sustav nije razumio kontekst podataka.

Primjeri korištenja GAN mreža bi bilo kreiranje novih slika na temelju opisa, kreiranje slika sličnih onima pri treningu, super rezolucija, oponašanje stila slikanja, popunjavanje nedostajućih dijelova slike, animacija lica i za ovaj projekt bojanje crno-bijelih slika. Trenutno se puno istraživanja provodi na unaprjeđivanju ovih mreža i razvoju novih modela.

1.2. Zadatak

Implementirati i trenirati model GAN mreže za bojanje slika iz CIFAR-10 baze slika koristeći programski jezik Python i Keras aplikacijsko korisničko sučelje nad Tensorflow bibliotekom. Opisati trenutno stanje u znanosti te dati teorijsku podlogu.

2. Trenutno stanje GAN mreža

2.1. GAN mreže sa spoznajom o široj slici

Navedene mreže su bazirane na konvolucijskim neuronskim mrežama i zbog prirode konvolucije obrađuju dio po dio slike u prozoru zadanih dimenzija tražeći značajke koje se propagiraju kroz mrežu i utječu na krajnji rezultat. Zhang et al. [1] predlažu drugačiji tip mreže gdje osim lokalnih značajki se uzimaju u obzir i druge značajke sa cijele slike kako bi diskriminator mogao naučiti odnose između značajki i različitih dijelova slike, a generator bolje i detaljnije generirao lažne podatke. Traži se ravnoteža između učinkovitosti i kompleksnosti mreže. Umjesto povećavanja broja konvolucijskih slojeva ili veličine jezgri koriste se već postojeće značajke s tim da mreža dobiva kapacitet da nauči važne veze između značajki, tzv. širu sliku.

Prije rada navedenih autora [1] koristila se spoznaja o široj slici, no ne u kontekstu GAN mreža, kao što predlažu. Kako bi mreža učila od nelokalnih piksela ili značajki na slici, implementiran je algoritam baziran na radu Wang et al. [2] koji kao glavnu ideju prati filter nelokalnih srednjih vrijednosti baziran u računalnom vidu. Taj filter uklanja šum sa slike kao neželjenu varijaciju u vrijednostima piksela na slici uslijed nesavršenosti optičkog senzora kamere i fizikalnih svojstava svjetla. Kako bi se uklonio šum na slici može se jednostavno kao vrijednost pojedinog piksela uzeti prosječna vrijednost piksela u nekoj njegovoj okolni što ima svoje nedostatke. Takav pristup uklanja piksele koji odskaču od prosječne vrijednosti, a koji su važni. Primjer takvih piksela bi često bili rubovi koje nije poželjno zamutiti. Umjesto da se uzme prosjek piksela u lokalnom prozoru promatranog piksela, traže se pikseli najsličniji promatranome i vrijednost promatranog piksela se mijenja tim prosjekom. Jednadžba (1) opisuje filter nelokalnih srednjih vrijednosti. [3]

$$\widehat{u_i}(p) = \frac{1}{C(p)} \sum_{q \epsilon B(p,r)} u_i(p) w(p,q), \qquad (1)$$
$$C(p) = \sum_{q \epsilon B(p,r)} w(p,q)$$

u predstavlja piksel gdje u_1 do u_3 predstavljaju različite kanale boja, p trenutno promatrani piksel, $B(p,r)$ okolinu oko trenutno promatranog piksela, $w(p,q)$ sličnost piksela u okolini

Gradeći nad opisanim filtrom dolazi se do nelokalnih neuronskih mreža autora Wang et al. [2] koje su stepenica do GAN mreža sa spoznajom o široj slici. Jednadžba (2) [2] je praktički identična jednadžbi (1), ona predstavlja općeniti oblik aktivacijske funkcije i od toga se grade različiti blokovi nelokalnih neuronskih mreža.

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j) g(x_j) \qquad (2)$$

i predstavlja indeks izlaza za kojeg se računa funkcija, j predstavlja indeks svih ostalih ulaza, x je ulazni, a y izlazni signal. Funkcija f predstavlja sličnost svih i i j, a funkcija g reprezentaciju ulaza na poziciji j.

Razlika naspram klasične konvolucije je ta što na izlazu nije uzeta u obzir samo lokalna okolina nego sve pozicije na ulazu.

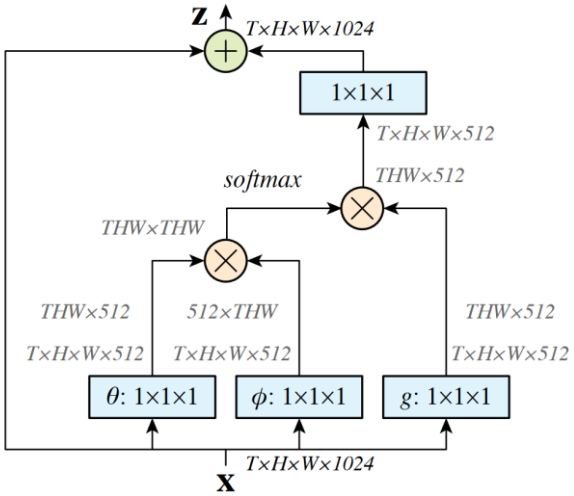
Biti će opisana samo jedna vrsta modela iz originalnoga rada [2], pokazali su kako odabir vrste ne utječe na učinkovitost mreže. Funkciju g se prikazuje samo kao matrica čije težine mreža treba naučiti, točnije, opisana je kao 1x1 konvolucija koja prepolavlja broj karata značajki na svom izlazu, dok funkcija f (jednadžba (3)) se razdvaja na dvije takve konvolucije čije matrice se nazivaju W_θ i W_ϕ.

$$f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)}, \quad \theta(x_i) = W_\theta x_i, \quad \phi(x_j) = W_\phi x_j \tag{3}$$

Uz definirane funkcije f i g, preostaje definirati $C(x)$ (jednadžba (4)) čime je potpuno definirana izlazna funkcija y. I takva vrsta modela se naziva *Embedded Gaussian*.

$$C(x) = \sum_{\forall j} f(x_i, x_j) \tag{4}$$

Model se lako umeće u postojeće modele neuronskih mreža tako da izlaz iz nekog neurona se zamijeni sa z prikazanim na slici 1 koji originalnoj vezi u mreži dodaje model pomnožen sa matricom koja može biti nula. Točnije: $z = W_z y_i + x_i$ gdje x_i predstavlja „staru“ vezu u mreži i ujedno i ulaz u nelokalni model, a y_i izlaz iz nelokalnog modela koji izlazi iz gornjeg produkta matrica (krugovi sa X u sredini). W_z je još jedna konvolucija 1x1 (plavi pravokutnici) koja može biti inicijalizirana na nulu, u tom slučaju se uopće ne koristi nelokalni model (dok mreža učenjem ne promijeni vrijednosti). Na slici su prikazane konvolucije 1x1x1 jer se originalni rad bazira na videima što također dodaje dodatnu dimenziju T prikazanim tenzorima.



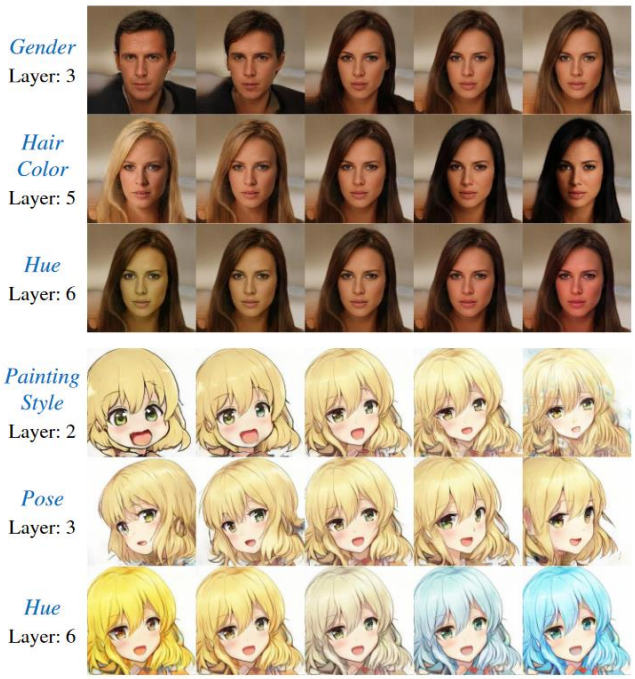
Slika 1: Arhitektura bloka nelokalnih neuronskih mreža [2]

GAN mreže sa spoznajom o široj slici iz [1] se baziraju na navedenom modelu nelokalnih neuronskih mreža uz spektralnu normalizaciju ne samo diskriminatora kao što je bio slučaj u nekim drugim radovima, nego i generatora.

Spektralnu normalizaciju predstavljaju Miyato et al. kako bi spriječili preveliki uspjeh diskriminatora zbog čega generator se ne može dobro trenirati. GAN mreže ovise o ravnoteži uspjeha oba modela. [4] Normalizacija se provodi tako da je l_2 norma matrice težina među slojevima jednaka 1 (korijen najveće svojstvene vrijednosti matrice treba biti 1). Osim navedene normalizacije koriste različite stope učenja za diskriminatora i generatora kako bi pomogli konvergenciji.

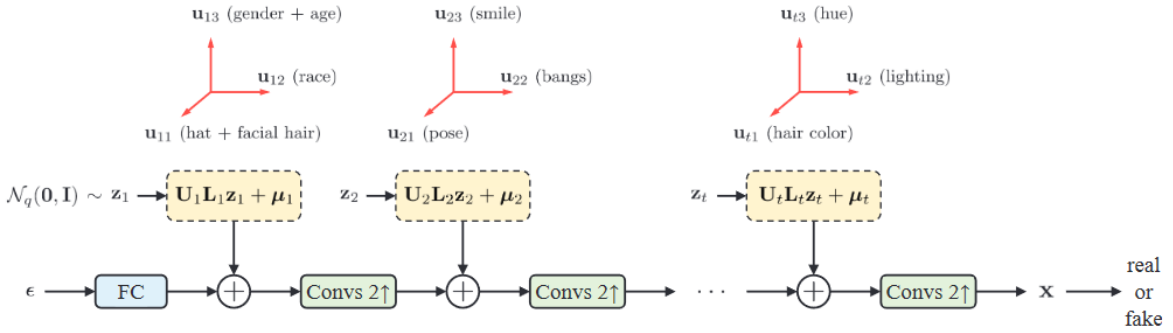
2.2. EigenGAN

Duboke neuronske mreže često se sastoje od konvolucijskih slojeva koji su filteri nad podacima. U slučaju da je ulaz u mrežu slika, ti filteri prepoznaju razne značajke, od onih jednostavnijih u plićim slojevima do onih kompliciranijih koji prepoznaju veze između značajki iz nižih slojeva. Primjer značajki sa plićih slojeva bi bili neki jednostavni oblici ili boje, dok bi spol osobe na slici bila puno kompliciranija značajka koju mreža ući u dubljim slojevima. He et al. [5] predlažu vrstu mreže koju su nazvali EigenGAN koja uči koji slojevi su zaduženi za pojedine značajke slike i omogućuje manipulaciju istima u svrhu postizanja zanimljivih rezultata na izlaznim slikama. Jedna od takvih manipulacija bi bila promjena spola na generiranoj slici osobe. Dodatni primjeri manipulacija se mogu vidjeti na slici 2.



Slika 2:Primjer promjene značajki [5]

Eigen dio imena mreže dolazi iz ideje o svojstvenim vrijednostima. Točnije, autori [5] predlažu da se u model generatora ubace potprostori sa svojim okomitim osima, gdje svaka os predstavlja jednu od značajki koje pojedini sloj može naučiti. Naučeni [5] spoznajama o manifestaciji značajki sa različitim razinama apstrakcije u različitim slojevima žele omogućiti mreži kontroliranje intenziteta značajke na generiranoj slici. Arhitektura mreže se može vidjeti na slici 3.



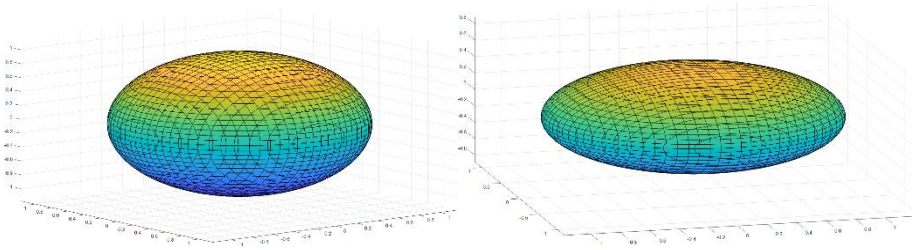
Slika 3:Arhitektura EigenGAN mreže [5]

X označava izlaznu sliku iz generatora na prethodnoj slici. Mreža je prikazana s lijeva na desno. Kao i klasična GAN mreža, generator uči iz latentnog prostora; komprimirani prikaz izlaznih slika, n-dimenzionalni vektor koji određuje kakva će biti izlazna slika za tako trenirani generator. U ovom slučaju omogućuje učenje onih značajki koje potprostori nisu uspjeli naučiti i taj prostor je označen sa ϵ, vektori su slučajna varijabla standardne Gaussove razdiobe. z_i predstavljaju isto što i ϵ za svaki potprostor, vrijednosti vektora z_i služe kao koordinate odgovarajućeg potprostora. U_i je koordinatni sustav potprostora čije osi označavaju različite značajke za taj sloj, L_i je dijagonalna matrica koja daje težinu pojedinoj osi iz pripadajućeg koordinatnog sustava, a μ_i označava ishodište pripadajućeg koordinatnog sustava. Convs predstavljaju konvolucije koje udvostručuju broj karata značajki (zeleni pravokutnici), a FC predstavlja potpuno povezani sloj (plavi pravokutnik). U narančastim pravokutnicima je upisana linearna kombinacija ($U_i L_i z_i + \mu_i$) koja se dodaje u ulaz trenutnog sloja uz izlaz iz prethodnog. Sve navedene parametre mreža uči kao i jezgre konvolucijskih slojeva.

3. Teorijska podloga

3.1. Latentni prostor

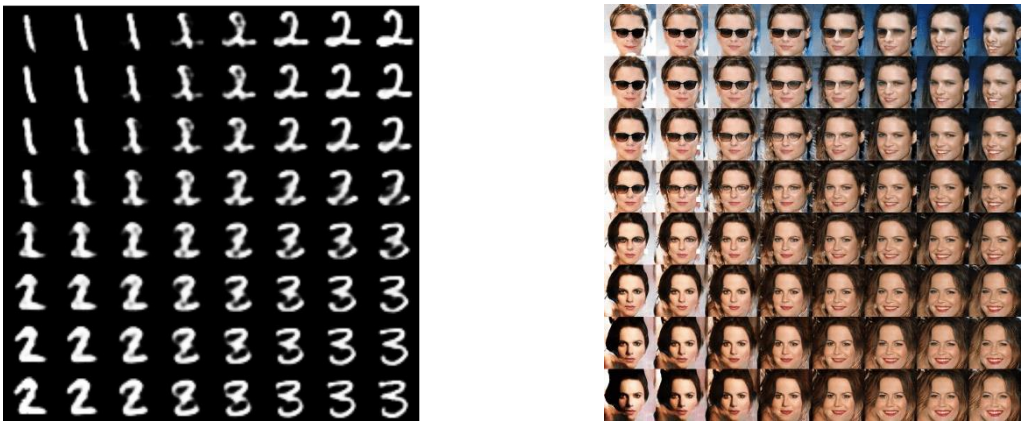
Kao osobe vrlo jednostavno zamišljamo objekte na temelju kratkog opisa njihovih značajki. Elipsoid može poslužiti kao jednostavan primjer. Lako zamišljamo izobličenje početne kugle u smjeru bilo koje osi tvoreći tako proizvoljne elipsoide jer znamo točno kako elipsoid izgleda. Potrebno je definirati samo duljine 3 poluosi, sa ta tri broja definiramo novi prostor u kojemu svaka točka uz određenu transformaciju daje spomenuti objekt.



Slika 4: Elipsoid

Za bilo koju točku u prostoru duljina poluosi možemo stvoriti objekt od interesa, saželi smo elipsoid na njegove tri značajke. Taj prostor značajki koje možemo podesiti nazivamo latentni prostor. Ako u obzir uzmemo pravac u x,y ravnini paralelan sa x osi tog prostora, svaka točka na tom pravcu bi predstavlja elipsoid koji se skuplja ili širi po jednoj poluosi ovisi u kojem smjeru putujemo po pravcu. Bilo kakva mala promjena od jedne do druge točke malo mijenja stvoreni objekt.

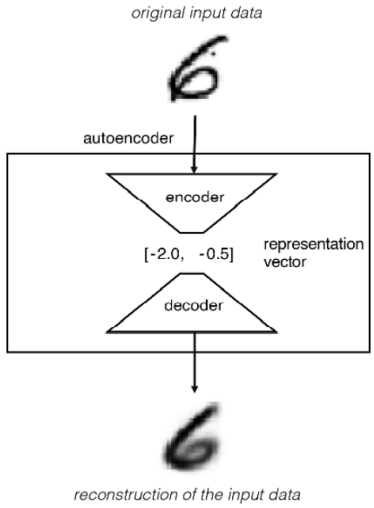
U ovom primjeru ovako definirati latentni prostor je prirodno, jednostavno iz formule smo izvukli značajke koje nas zanimaju. Ne trebaju nam neuronske mreže da definiramo taj prostor niti da stvaramo objekte na temelju točaka iz tog prostora, ali ideja se može primijeniti na neuronske mreže i stvaranje novih slika sličnih onima pri treniranju. U slučaju lica, vrlo teško možemo definirati latentni prostor (to bi bio neki višedimenzionalni prostor koji ne možemo ni zamisliti) i iz tih koordinata zamisliti neko lice pa pomalo mijenjati lice ovisno u kojem smjeru se krećemo od početne točke. Računala mogu naučiti sažeti lice na točku u n-dimenzionalnom prostoru i za bilo koju točku iz toga prostora stvoriti do sada neviđenu sliku lica što je prikazano na slici 5. Osim lica, mogu se vidjeti i prijelazi između brojeva MNIST baze slika.



Slika 5:Brojevi, lica i latentni prostor [6]

3.2. Autoenkoder

Vrsta nenadziranih neuronskih mreža koja uči sažeti informacije u latentni prostor i gotovo identično te informacije ponovno kreirati iz latentnog prostora. Cilj mreže nije da kopira ulazne podatke na svom izlazu nego da nauči prepoznati važne značajke na ulazu iz nestrukturiranih podataka. Kako bi se spriječilo ne tako korisno kopiranje podataka, ova vrsta mreža u svojoj strukturi ima usko grlo kroz koje sve informacije sa ulaza ne mogu proći, nema kapacitet da nauči kopirati podatke sa ulaza. To usko grlo je sažeta reprezentacija podataka, spomenuti latentni prostor.

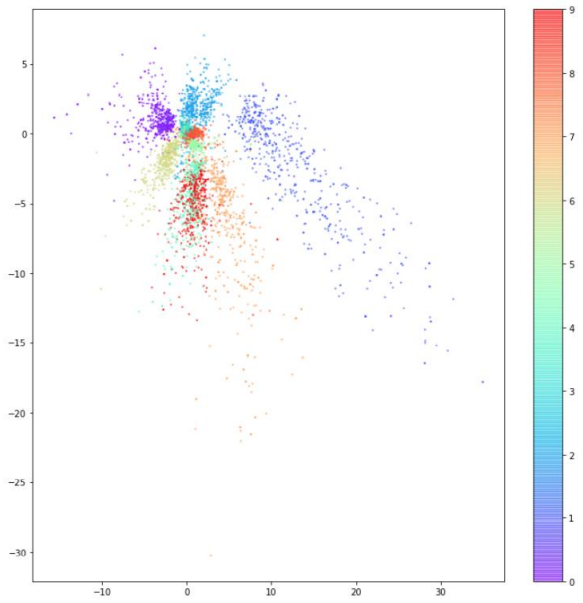


Slika 6: Idejna struktura autoenkodera [7]

Na slici 6 je prikazana struktura autoenkodera, sastoji se od dva dijela: enkodera i dekodera koji komuniciraju preko latentnog prostora. Enkoder uči najvažnije značajke skupa podataka za trening i te značajke upisuje u odgovarajuće mjesto u latentnom prostoru, dok dekoder iz poznatih značajki u latentnom prostoru rekonstruira originalne podatke. I enkoder i dekoder predstavljaju nelinearne funkcije koje neuronske mreže mogu aproksimirati. To mogu biti jednostavne plitke neuronske mreže, duboke neuronske mreže, konvolucijske neuronske mreže. Kompleksnost mreže ovisi o ulaznim podacima i primjeni. U slučaju konvolucijskih mreža enkoder se često tvori od niza konvolucijskih slojeva koji smanjuju dimenzionalnost ulaznih podataka, a dekoder se tvori od dekonvolucijskih slojeva koji uče interpretirati podatke iz nižih dimezija sve dok podatci ponovno ne dosegnu dimenzionalnost sa ulaza.

Kao i klasična neuronska mreža, za cilj ima minimizirati funkciju troška čiji se gradijent propagira unazad kroz mrežu tako trenirajući promjenjive parametre. Funkcija troška koju ove mreže minimiziraju je bilo koja funkcija (naravno, derivabilna) koja predstavlja sličnost ulaznog i izlaznog podatka, npr. srednja kvadratna pogreška.

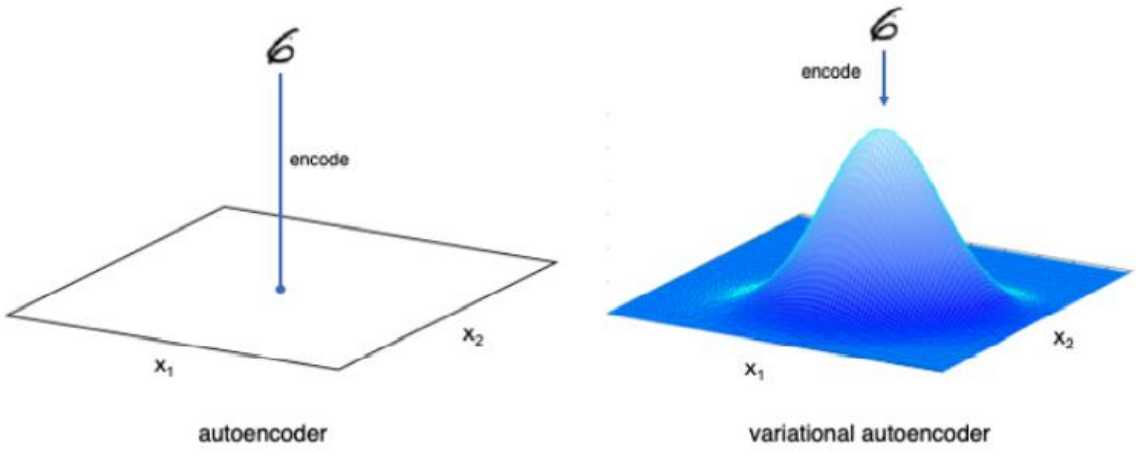
Problem autoenkodera je u latentnom prostoru. Nije sigurno da ako odaberemo nasumičnu točku u latentnom prostoru da bi dekoder mogao proizvesti neki podatak za koji bi rekli da je moguće da pripada skupu za trening. Na primjeru slika iz MNIST baze, neke točke u latentnom prostoru ne proizvode slike koje izgledaju kao ikakav broj, točke nisu smještene oko ishodišta toga prostora nego su točke nasumično raštrkane s tim da mreža može favorizirati neke brojeve, a druge ne. Distribucija toga latentnog prostora nije kontinuirana, ima spomenute „rupe“, niti je ograničena (slika 7).



Slika 7: Vizualizacija latentnog prostora autoenkodera za MNIST bazu slika [7]

3.3. Varijacijski autoenkoder

Nenadzirana neuronska mreža gotovo jednake arhitekture kao autonekoder s promjenama u enkoderu i aktivacijskoj funkciji. Latentni prostor ove mreže je kontinuiran i ograničen, točnije, ne sastoji se od točaka koje enkoder upisuje u latentni prostor kao u slučaju autoenkodera nego predstavlja multivarijantnu normalnu razdiobu (slika 8). [7]



Slika 8: Razlika u latentnom prostoru varijacijskog autoenkodera i autoenkodera [7]

Enkoder uči srednje vrijednosti i varijance te razdiobe tijekom treniranja, a kako bi upisao podatak kao njegovu odgovarajuću interpretaciju u latentnom prostoru uzima uzorak iz standardne normalne razdiobe i računa interpretaciju pomoću naučenih srednjih vrijednosti i varijanci kao što je prikazano jednadžbom 5 (sve veličine su vektorske veličine).

$$interpretacija = \mu + \sigma\epsilon \tag{5}$$

μ – srednje vrijednosti, σ – varijance, ϵ – uzorak iz standardne normalne razdiobe

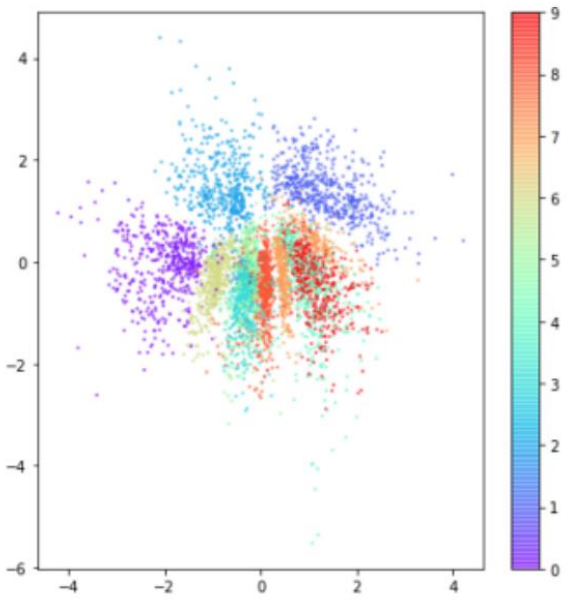
Funkcija troška (jednadžba 6) se mijenja na sljedeći način: umjesto samo funkcije koja predstavlja sličnost ulaznog i izlaznog podatka (npr. srednja kvadratna pogreška, jednadžba 7), dodaje se faktor koji predstavlja sličnost naučene razdiobe normalnoj razdiobi. Taj faktor se naziva Kullback–Leibler (KL) divergencija (jednadžba 8).

$$\mathcal{L}_{VAE} = \mathcal{L}_R + \mathcal{L}_{KL} \tag{6}$$

$$\mathcal{L}_R = \sum (y_{true} - y_{predicted})^2 \tag{7}$$

$$\mathcal{L}_{KL} = \frac{1}{2} \sum (1 + \log(\sigma)^2 - \mu^2 - \sigma^2) \tag{8}$$

Na slici 9 je prikazan napredak u grupiranju u latentnom prostoru kod varijacijskog autoenkodera. Vidimo da točke su centrirane oko 0 i otprilike ravnomjerno raspoređene oko tog središta što odgovara standardnoj normalnoj razdiobi zbog uvedenog faktora KL divergencije. To je korisno jer sada odabirući bilo koji vektor iz standardne normalne razdiobe, možemo ga dekodirati u uvjerljivu novu sliku koja izgleda kao da je iz originalne MNIST baze slika. Dodatan razlog zašto bi slike izgledale uvjerljivo je to što je naučena razdioba kontinuirana, stoga bilo koja točka daje uvjerljiv rezultat.



Slika 9: Vizualizacija latentnog prostora varijacijskog autoenkodera za MNIST bazu slika [7]

3.4. GAN

3.4.1. Uvod

Vrsta neuronskih mreža koja se sastoji od dva modela: generatora i diskriminatora. Generator je praktički dekoder varijacijskog autonekodera koji kao ulaz prima vektor iz normale razdiobe (šum) i generira nove podatke slične onima pri treningu. Diskriminator služi kao povratna informacija generatoru kako bi znao koliko su uvjerljivi podatci koje generira. U slučaju slika se za oba modela često koriste konvolucijske neuronske mreže.

Modeli se treniraju pojedinačno. Generator generira neke nove podatke iz ulaznog šuma. Ti podatci u početku su jako loše kvalitete i diskriminator vrlo jednostavno za prave podatke na izlazu daje 1, a za generirane (lažne) podatke 0 na svom izlazu. Na temelju novih saznanja o tome kako izgledaju lažni podaci, još bolji postaje u klasifikaciji. Diskriminator predstavlja binarni klasifikator, točnije daje vjerojatnost da je na njegovom ulazu stvarni podatak iz skupa za trening.

Nakon što je diskriminator dobio dodatne informacije o tome kako izgledaju lažni podaci, više ne uči, sada je red na generatoru. Generirani lažni podaci se predaju diskriminatoru kojeg se pokušava uvjeriti da su to stvarni podaci, a koliko je diskriminator siguran da bi to mogli biti stvarni podaci služi kao povratna informacija generatoru koji uči kako lakše zavarati diskriminatora, idealno uči generirati uvjerljive podatke.

Nakon duljeg treniranja (jedan model, pa drugi model i tako u krug), generator generira dovoljno dobre podatke tako da diskriminator više ne zna razlikovati stvarne od generiranih podataka. U tom slučaju odbacujemo diskriminatora i samo zadržavamo model generatora kojeg koristimo za generiranje podataka u budućnosti.

Na primjeru MNIST baze slika želimo kreirati model koji generira znamenke. Kao ulaz u generator dajemo vektor iz latentnog prostora, vektor iz normalne razdiobe. Na izlazu dobivamo sliku neke znamenke, ali za sada nemamo način da uvjetujemo koja točno znamenka će biti generirana. Pošto nema smisla čekati da se dogodi baš znamenka npr. 5, uvodimo pojam GAN s uvjetom.

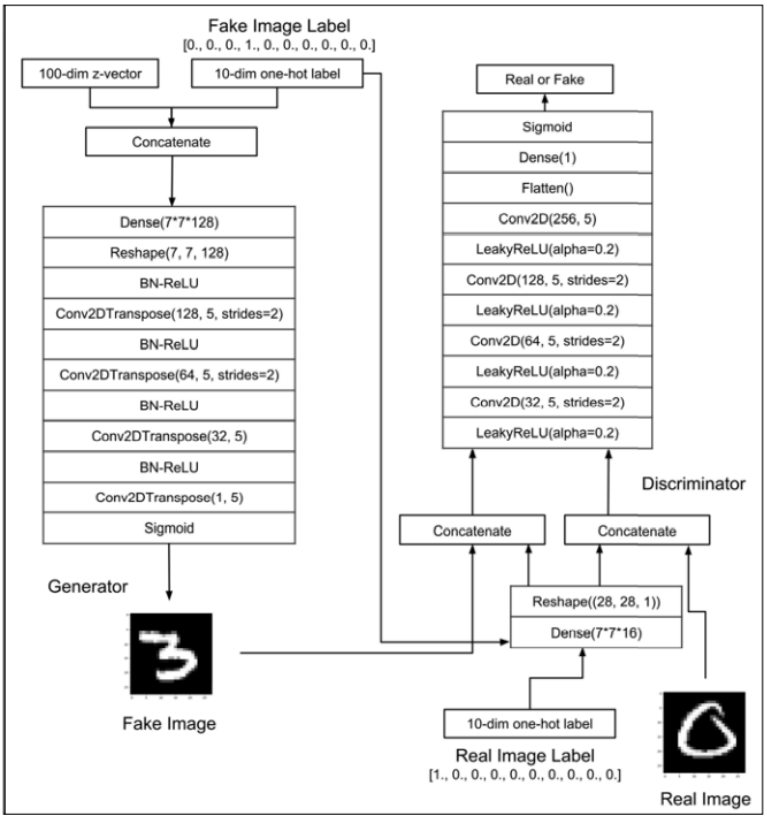
3.4.2. GAN s uvjetom

MNIST baza slika dolazi sa slikama i informaciji koja se točno znamenka nalazi na pojedinoj slici. Tu informaciju nećemo odbaciti nego ćemo ju iskoristiti za generiranje točno onih znamenaka koje odredimo. Razlika između „obične“ GAN mreže i mreže s uvjetom je u dodatnoj oznaci koja daje mreži kapacitet da zna koja slika predstavlja koju znamenku.

Na slici 10 je prikazan primjer konvolucijske GAN mreže s uvjetom. Počevši od gornje stanje slike prikazan je 100 dimenzionalni vektor (vektor iz latentnog prostora, normalna razdioba) kojemu je dodan 10 dimenzionalni vektor oznake gdje su svi elementi vektora 0 osim na onom mjestu koje predstavlja određenu znamenku (1000000000 bi predstavljalo znamenku 0 jer na nultom mjestu se nalazi jedinica, općenito takvi vektori (one-hot enconding vektori) su onolike dimenzije koliko set podataka ima klasa). U ovom primjeru odmah nakon konkatenacije latentnog vektora i vektora oznake dolazi potpuno povezani sloj (dense) koji je često prvi sloj generatora. Izlaz tog sloja se preoblikuje (mijenjaju se odnosi dimenzija) i dalje

propagira kroz slojeve koji su klasični za konvolucijske neuronske mreže. Lijevi dio slike predstavlja model generatora koji nakon oblikovanja izlaza iz potpuno povezanog sloja normalizira ulaze u iduće slojeve (BN na slici (Batch Normalization), ulazi imaju srednju vrijednost 0 i jediničnu varijancu). Aktivacijska funkcija generatora je ReLU (Rectified Linear Unit) koji za ulaze manje od 0 daje 0, a za ostale ulaze točno taj ulaz. Kako bi uspjeli „napuhati" latentni vektor i vektor oznake na uvjerljivu sliku koristimo slojeve obrnute konolucije sa prikazanim brojem filtera, veličinom jezgre 5 i promjenjivim korakom (2 ili 1). Kao izlazna aktivacijska funkcija koriste se sigmoid aktivacijska funkcija, tako slika na izlazu za svaki piksel ima vrijednosti od 0 do 1.

Model diskriminatora se nalazi na desnoj strani slike 10. Također sadrži informaciju o kojoj se znamenci u vektoru oznake. Ostali slojevi su jako slični, mreža je jednostavna mreža klasifikacije. Umjesto ReLU aktivacijske funkcije koristi se Leaky ReLU koja za negativne vrijednosti ne daje 0 nego ih smanji za određeni faktor, umjesto obrnute konvolucije, koristi se konvolucija, sažimamo ulaz koji su stvarne ili lažne slike sa svojim vektorom oznake u točno jednu vrijednosti između 0 i 1 (vjerojatnost da je slika iz originalne baze slika).



Slika 10: Model GAN mreže s uvjetom [8]

3.4.3. Problemi pri treniranju

U klasičnim GAN mrežama često se javlja problem oscilirajućeg troška gdje umjesto da se trošak smanjuje i konvergira, trošak se stohastički povećava i smanjuje bez vidljivog trenda smanjenja ili povećanja kroz epohe. Mreža ne može konvergirati.

Urušavanje moda (engl. *mode collapse*) događa se kada generator pronađe jedan podatak koji će sigurno prevariti diskriminatora. U tom slučaju mreža ne generira nove podatke pri svakoj iteraciji, niti nužno korisne (možda ne izgledaju kao podatci iz skupa podataka za treniranje). U tom slučaju gradijent funkcije troška je gotovo nula tako da mreža stagnira u tom lošem načinu djelovanja.

Ako je diskriminator previše uspješan (ukoliko su vrijednosti gotovo 1 ili gotovo 0) tada generator ne može naučiti ih grešaka, povratna informacija nije tako korisna pa generator stagnira. U suprotnom ako je diskriminator nedovoljno uspješan, javlja se problem nestajućeg gradijenta, u tom slučaju promjene gradijenta se ne uspiju propagirati kroz generator.

Parametri koji definiraju mrežu, kao što se dalo vidjeti u prethodnom odlomku o GANovima s uvjetom (sam model mreže, veličina jezgre, broj filtera, alpha parametar Leaky ReLU aktivacijske funckije itd.), jako utječu na konvergenciju GAN mreža i uz male njihove promjene. Kvalitetno izabrati parametre se često svodi na pokušavanje različitih parametara uz praćenje ponašanja funkcije troška.

4. Bojanje crno-bijelih slika

4.1. Originalni rad

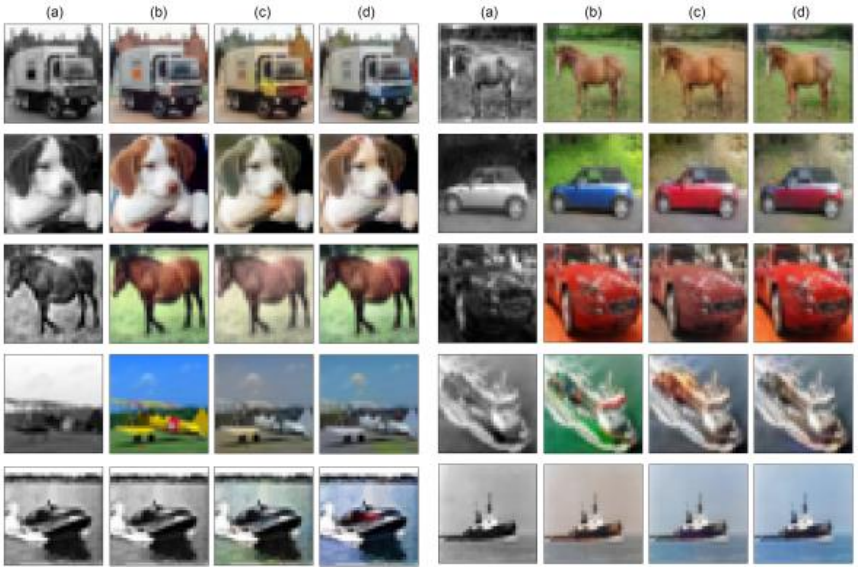
Nazeri et al. [9] predlažu svoj način bojanja crno bijelih slika koristeći konvolucijsku neuronsku GAN mrežu. Koriste baze slika CIFAR-10 i Places365. Vrsta GAN mreža je GAN uz uvjet koji je opisan u prethodnim poglavljima s tim da model prati U-Net arhitekturu mreža kako bi sačuvali važne informacije iz slojeva prije sažimanja podataka.

Umjesto korištenja RGB modela boja za slike, koriste L*a*b* model gdje L kanal predstavlja intenzitet svjetla slike, a a i b kanali predstavljaju kanale boje.

Jednadžba 9 prikazuje funkciju troška za model generatora. Maksimizira se vjerojatnost da je diskriminator u krivu (minimum negativne vrijednosti funkcije) zbog toga što je pokazano da takva funkcija ne divergira prema -∞ prilikom minimizacije. Osim toga se dodaje razlika između generirane obojane slike i originalne slike pomnoženo regularizacijskim parametrom λ koji je postavljen na 100.

$$\min_{\theta_G} J^{(G)^*}(\theta_D, \theta_G) = \min_{\theta_G} -\mathbb{E}_z[\log(D(G(z)))] + \lambda \parallel G(z) - y \parallel_1 \tag{9}$$

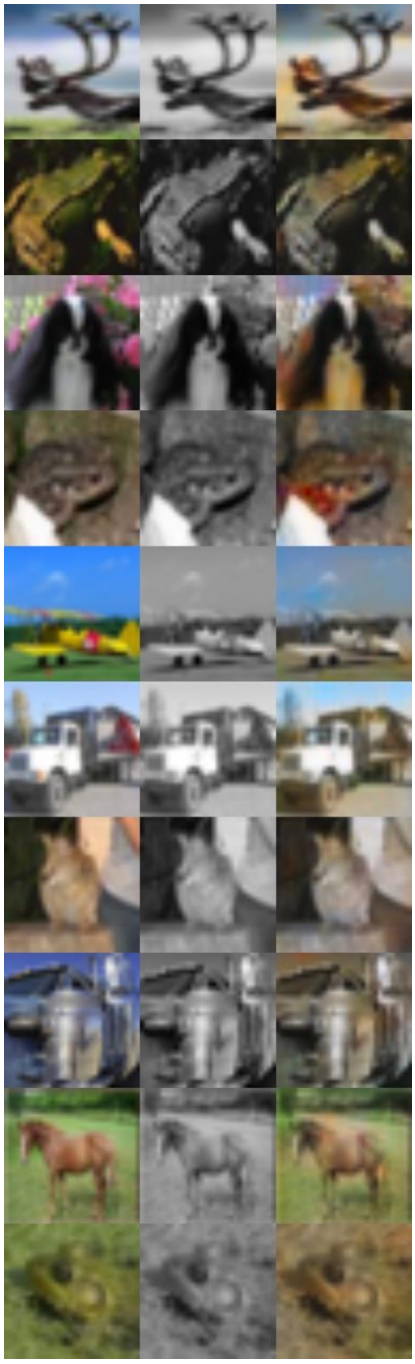
Prilikom treniranja predlažu korištenje navedene funkcije troška. Kako je diskrimniator binarni klasifikator koji daje vjerojatnost da su generirani podatci stvarni, vrijednosti znaju biti gotovo 0 ili gotovo 1 što se pokazalo da loše utječe na učenje mreža, stoga predlažu korištenje 0.9 umjesto 1 kao oznaku za stvarne podatke. Kao i primjer GAN mreže sa uvjetom na slici 10 koriste normalizaciju ulaza u iduće slojeve (engl. *Batch Normalization*) kao i konvolucijsku neuronsku mrežu i Leaky ReLU aktivacijsku funkciju. Koriste optimizator Adam uz smanjenje regularizacijskog parametra β_1 na 0.5. Rezultati originalnog rada se vide na slici 11.



Slika 11: Rezultati iz originalnog rada a)crno-bijela slika b)originalna slika c)obojana koristeći samo U-Net d)obojana koristeći GAN mreže [9]

4.2. Rezultati

Uz ovaj tekst priložen je kod pisan u programskom jeziku Python uz korištenje biblioteka OpenCV, Keras nad Tensorflow-om i dr. Rezultati bojanja slika se vide na slici 12. Također je korištena baza slika CIFAR-10 kao i arhitektura mreže iz originalnog rada. Prvi stupac predstavlja originalne slike, drugi stupac crno-bijele slike, a posljednji stupac predstavlja obojane slike. Mjesta za napredak ima puno, ali je nešto od čega bi se moglo početi.



Slika 12:Rezultati bojanja slika

Literatura

[1] H. Zhang, I. Goodfellow, D. Metaxas i A. Odena, Self-Attention Generative Adversarial Networks, 2019.

[2] X. Wang, R. Girshick, A. Gupta i K. He, Non-local Neural Networks, 2018.

[3] A. Buades, B. Coll i J.-M. Morel, Non-Local Means Denoising, 2011.

[4] T. Miyato, T. Kataoka, M. Koyama i Y. Yoshida, Spectral Normalization for Generative Adversarial Networks, 2018.

[5] Z. He, M. Kan i S. Shan, EigenGAN: Layer-Wise Eigen-Learning for GANs, 2019.

[6] M. Pieters i M. Wiering, Comparing Generative Adversarial Network Techniques for Image Creation and Modification, 2018.

[7] D. Foster, Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play, O'Reilly, 2019.

[8] R. Atienza, Advanced Deep Learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, Deep RL, Unsupervised Learning, Object Detection and Segmentation, and More, 2nd Edition, Packt, 2020.

[9] K. Nazeri, E. Ng i M. Ebrahimi, Image Colorization using Generative Adversarial Networks, 2018.