Sustav povećanja razlučivosti slike zasnovan na strojnom učenju

Anđelković, Karlo

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:168:390474

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-23**

Repository / Repozitorij:

FER Repository - University of Zagreb Faculty of Electrical Engineering and Computing repozitory

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 446

SUSTAV POVEĆANJA RAZLUČIVOSTI SLIKE ZASNOVAN NA STROJNOM UČENJU

Karlo Anđelković

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 446

SUSTAV POVEĆANJA RAZLUČIVOSTI SLIKE ZASNOVAN NA STROJNOM UČENJU

Karlo Anđelković

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 446

Pristupnik: **Karlo Anđelković (0036522628)**

Studij: Računarstvo

Profil: Računarska znanost

Mentor: izv. prof. dr. sc. Marin Šilić

Zadatak: **Sustav povećanja razlučivosti slike zasnovan na strojnom učenju**

Opis zadatka:

Povećanje razlučivosti slike zasnovano na strojnom učenju je postupak višestrukog povećanja razlučivosti slike uz zadržavanje njezinog sadržaja i detalja. Rezultat postupka je inačica izvorne slike visoke razlučivosti. Ovaj se postupak može koristiti za različite primjene kao što je poboljšanje kvalitete slike, poboljšanje vizualnih detalja i povećanje točnosti algoritama računalnog vida. Proučiti i opisati najuspješnije postojeće postupke za povećanje razlučivosti slike zasnovane na strojnom učenju s posebnim naglaskom na postupke koje koriste duboko učenje. Pronaći i prikupiti javno dostupne skupove podataka primjenjive za učenje i ispitivanje postupaka dubokog učenja za povećanje razlučivosti slike. Oblikovati i programski ostvariti sustav zasnovan na dubokom učenju za povećanje razlučivosti slike. Ispitati uspješnost ostvarenog sustava primjenom prikladno odabranih mjera te prikazati i opisati rezultate ispitivanja. Uz rad je potrebno predati i dokumentirati izvorni kod ostvarenog sustava, korištene skupove podataka te navesti korištenu literaturu.

Rok za predaju rada: 28. lipnja 2024.

Zahvaljujem mentoru Marinu Šiliću.

SADRŽAJ

1. Uvod **1**

2. Super rezolucija **3**

2.1. Duboko učenje 3

2.1.1. Kratak pregled 3

2.1.2. Definicija problema 4

2.2. Naduzorkovanje 4

2.2.1. Klasične metode 4

2.2.2. Metode temeljene na učenju 7

2.3. Ulazni podaci 9

2.3.1. *Single i multi-image sustavi* 9

2.3.2. Skupovi podataka 10

2.3.3. Augmentacija podataka 10

2.4. Poredak naduzorkovanja 11

2.5. Evaluacijske metrike 13

2.6. Funkcije gubitka 16

2.7. Arhitekture mreža 18

2.7.1. Rezidualne neuronske mreže 19

2.7.2. Rekurzivne neuronske mreže 20

2.7.3. Mreže s više puteva 21

2.7.4. Ansambl 22

2.7.5. Guste veze 22

2.7.6. Generativne suparničke mreže 23

2.7.7. Difuzijski modeli 24

2.7.8. Mehanizam pozornosti 25

2.7.9. Transformer arhitekture 27

2.8. Metode niskog nadzora 29

2.8.1. Polunadzirane metode 29

2.8.2. *Few-shot* metode 29
2.8.3. Nenadzirane metode 30
2.8.4. Samonadzirane metode 30

3. Implementacija **32**
3.1. Model 32
3.2. Skup podataka 33
3.3. Proces učenja 33
3.4. Rezultati 34
3.5. Korišteni alati i kod 35
3.5.1. Alati 35
3.5.2. Kod 36

4. Zaključak **38**

Literatura **39**

1. Uvod

Povećavanje razlučivosti slika jedan je od važnijih zadataka u procesuiranju slika i računalnom vidu. Sustavi za povećanje razlučivosti slika često se nazivaju i sustavima za super rezoluciju *(eng. super resolution, SR)*, a njihov zadatak jest povećanje rezolucije slika faktorom veličine 4 ili više, bez gubitka detalja i s održavanjem kvalitete slike. U rasterskoj grafici, rezolucija se odnosi na broj piksela koji se nalaze na slici te je njihov broj povezan s količinom detalja koju slika može prikazati. Što slika ima više piksela, može biti prostorno veća bez degradacije kvalitete, ali zbog konačnosti rasterske mreže uvijek postoji ograničenje. Za skaliranje rasterskih slika potrebno je generirati novu sliku s većom ili manjom rezolucijom. Ako se radi smanjivanje slike, postavlja se problem uzorkovanja postojećih piksela, a kada se slika povećava, potrebno je stvoriti nove piksele. Rezolucija se uglavnom izražava kao umnožak broja piksela po visini i širini slike ili mjernom jedinicom piksel po inču (PPI).

Sustavima za povećanje razlučivosti se na ulaz predaje jedna ili više slika niske rezolucije *(eng. low resolution, LR)* te se kao konačan rezultat dobiva verzija slike visoke rezolucije *(eng. high resolution, HR)*. Povećanjem razlučivosti slika dobiva nove piksele i time postaje jasnija i može se proširiti. Cilj sustava za povećanje razlučivosti jest odrediti vrijednost novih piksela. Postoji velik broj pristupa i algoritama za interpolaciju novih vrijednosti, no najsuvremenija rješenja koriste tehnike dubokog učenja te se upotrebljavaju vrlo različite arhitekture mreža, funkcije gubitka i strategije učenja za taj problem. Prijašnje metode bile su bazirane na statističkim principima, lokalnim vrijednostima i rekonstrukciji, ali takve su operacije nerijetko jako jednostavne ili koriste veliku količinu računalnih resursa bez zavidnih rezultata. Problem povećanja razlučivosti je zanimljiv zato što jedna slika niske rezolucije može odgovarati više različitih slika visoke rezolucije te je potrebno definirati precizne mjere po kojima se ocjenjuje uspješnost sustava. Duboko učenje omogućava sustavima da prepoznaju komplicirane uzorke u slikama te daje im sposobnost da samostalno izlučuju informacije iz njih.

Super rezolucija slika ima široku primjenu te se često koristi u samim procesima strojnog učenja i računalnog vida za dobivanje kvalitetnijih i točnijih modela kod zadataka prepoznavanja, detekcije i segmentacije. Najčešće su primjene u sigurnosti, nadzornim sustavima, medicinskom snimanju, satelitskim snimkama, svemirskoj fotografiji, mikroskopiji i restauraciji. Nadzorne kamere imaju ograničenu kvalitetu zbog memorije te zato snimaju u niskim rezolucijama pa je potrebno unaprijediti takve snimke, a medicinska snimanja poput magnetske rezonance i kompjuterizirane tomografije nemaju idealne uvjete ni jasan pogled na objekt zbog kojih kvaliteta snimke opada i otežava dijagnostiku. Također, korištenjem super rezolucije, stvaranje slika i videa visoke rezolucije postaje lakše i jeftinije te se u komercijalnom sektoru unapređuje korisničko iskustvo.

2. Super rezolucija

Super rezolucija slika definira se kao proces generiranja slike visoke rezolucije iz ulazne slike niže rezolucije. Zadatak super rezolucije uvršten je u sferu računalnog vida, točnije, digitalnu obradu slika te je srodan zadacima poput obnove slika, generacije slika i uklanjanje šuma. Suvremena rješenja oslanjaju se na duboko učenje i postižu iznimno dobre rezultate.

2.1. Duboko učenje

Duboko učenje grana je strojnog učenja koja se bavi neuronskim mrežama s velikim brojem slojeva. U jezgri svih algoritama dubokog učenja postoje tri neizbježne komponente: model, funkcija gubitka i optimizacija.

2.1.1. Kratak pregled

Model definira arhitekturu mreže, način na koji su umjetni neuroni u mreži ostvareni i povezani. Model je izgrađen od slojeva koji mogu imati različite uloge, ali u svim mrežama su prisutni ulazni i izlazni sloj. U zadacima računalnog vida česta je uporaba konvolucijskih slojeva koji obavljaju izlučivanje značajki i stvaraju mape. Važno svojstvo jest postojanje parametara koji se mogu učiti, a ono je ostvareno težinama neurona s kojima se množi ulaz.

Funkcija gubitka metoda je evaluacije koju model koristi za procjenu vlastite uspješnosti. Ulaz funkcije gubitka jest izlaz mreže i stvarna vrijednost podatka koji se pokušavao predvidjeti modelom. Funkcija najčešće ne gleda na koji način je predikcija kriva, već koliko je kriva.

Izlaz funkcije gubitka koristi se u optimizacijskom postupku, procesu učenja. Mreža se hrani podacima za koje predviđa izlaznu vrijednost, računa se iznos funkcije gubitka

i određuje se gradijent koji definira smjer ažuriranja parametara. Proces se obavlja od zadnjeg neurona prema početku mreže pa se zato naziva unazadna propagacija. Cilj učenja modela jest minimizacija funkcije pogreške.

2.1.2. Definicija problema

Slike niske rezolucije I_{LR} mogu se modelirati iz slika visoke rezolucije I_{HR} degradacijskom funkcijom koja smanjuje dimenzije i obavlja poduzorkovanje. Degradacijska funkcija koristi parametre δ koji određuju faktor skaliranja dolje ili unošenje šuma. Najjednostavnija definicija procesa izražava se kao u izrazu 2.1 gdje je $\mathcal{D}$ oznaka funkcije degradacije.

$$I_{LR} = \mathcal{D}(I_{HR}; \delta) \tag{2.1}$$

Super rezolucija može se zamisliti kao obrnuti postupak od degradacije, te modeli koji moraju povećati rezoluciju dobivene LR slike implementiraju postupak $\mathcal{F}$ koji također ima vlastite parametre θ (2.2), a ti parametri su težine neuronske mreže.

$$I'_{HR} = \mathcal{F}(I_{LR}; \theta) \tag{2.2}$$

Cilj je pronaći optimalne parametre θ postupkom optimizacije, dakle zadatak super rezolucije svodi se na obavljanje minimizacije funkcije gubitka 2.3.

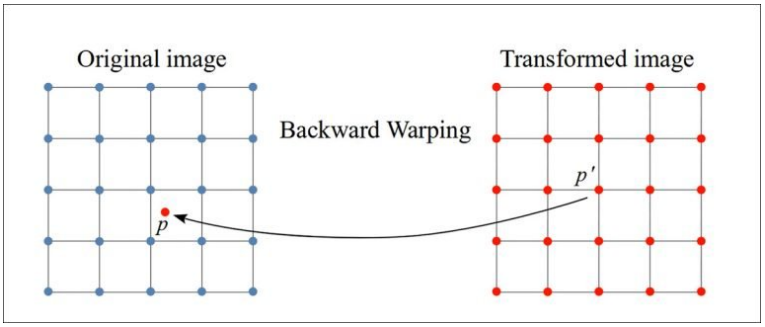
$$\theta' = argmin_{\theta} \mathcal{L}(I'_{HR}, I_{HR}) \tag{2.3}$$

Naravno, ovako postavljen problem je iznimno pojednostavljen i postoji pregršt arhitektura, operacija i oblika koje mogu poprimiti $\mathcal{D}$, $\mathcal{F}$, $\mathcal{L}$, δ i θ [1].

2.2. Naduzorkovanje

2.2.1. Klasične metode

Područje istraživanja super rezolucije postoji već dugi niz godina zbog potrebe skaliranja slika te su davno stvoreni algoritmi koji se danas koriste u brojnim programskim rješenjima kod zumiranja ili povećavanja slika. Ovi postupci najčešće se koriste nakon unatražne transformacije skaliranja, gdje se prvo stvori prazna matrica novih dimenzija, zatim se svakom pikselu izračuna odgovarajuća pozicija u izvornoj slici čije koordinate su većinski decimalni brojevi te se na temelju te pozicije izračunava nova vrijednost. Unatražna transformacija prikazana je na slici 2.1.



Slika 2.1: Unatražna transformacija [2]

Interpolacija najbližim susjedom jedan je od najjednostavnijih postupaka za povećavanje rezolucije slike u kojem se svakom pikselu izlaza pridjeljuje vrijednost najbližeg susjeda. Ovakav proces rezultira velikim brojem piksela iste vrijednosti što uzrokuje zupčaste rubove i nedostatak dubine na slici. Također, radi ubrzavanja postupka i čuvanja računalnih resursa, u implementacijama je uobičajeno izbjeći zaokruživanje i samo izbrisati decimalni dio koordinata. Ovakav pristup je brz i koristi se kada slike nemaju glatke rubove ni prijelaze, npr. u piksel umjetnosti.

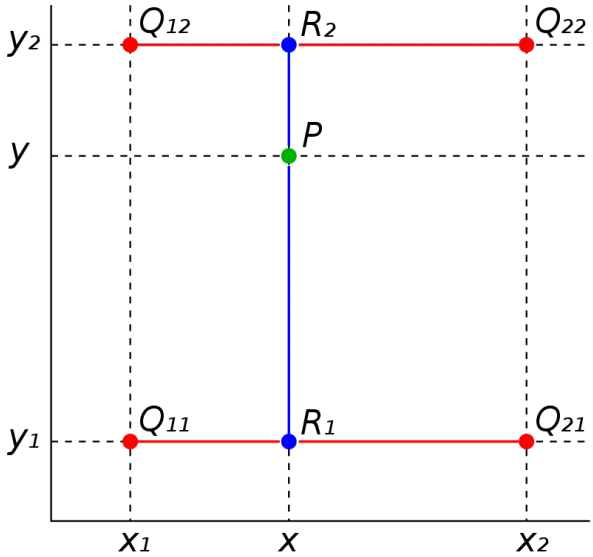
Bilinearna interpolacija metoda je ponovnog uzorkovanja koja koristi težinske vrijednosti bazirane na udaljenosti 4 najbliža piksela. Interpolacija uvodi kontinuitet u izlaznim vrijednostima i na mjestima gdje originalna slika ima diskretne prijelaze te smanjuje kontrast što nije poželjno na slikama na kojima prevladavaju linije. Moguće je prvo napraviti interpolaciju u jednom smjeru pa u drugom te odabir početnog smjera nije bitan:

$$f(x, y_1) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \tag{2.4}$$

$$f(x, y_2) = \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \tag{2.5}$$

$$f(x, y) = \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \tag{2.6}$$

Jednadžbe 2.4, 2.5 i 2.6 prikazuju kako bi se računala vrijednost piksela, prvo u smjeru x, a zatim u y. Oznaka f označava vrijednost piksela, x i y su koordinate piksela, a oznaka Q je sam piksel, kao što je prikazano na slici 2.2. Ova se metoda koristi kada je potrebno dobiti relativno dobar rezultat u kratkom vremenskom roku, poput zumiranja.



Slika 2.2: Bilinearna interpolacija [3]

Bikubična interpolacija sličan je postupak bilinearnoj, ali ovdje se koristi 4×4 piksela za izračun nove vrijednosti. Algoritam bolje održava detalje slike, ali je skuplji, stoga nije idealan kada je rezultat potrebno dobiti brzo. Ova se metoda često koristi za izgradnju SR skupova podataka kod smanjivanja slika i tijekom pretprocesiranja u samim sustavima za povećanje razlučivosti.

hqx je set 3 algoritma za povećavanje razlučivosti za faktore 2, 3 i 4. Algoritmi iteriraju po svim pikselima izvorne slike i pronalaze oblike tražeći slične boje. Svaki piksel se pretvara u 2×2, 3×3 ili 4×4 piksela ovisno o algoritmu, a proširenja su prije definirana u tablici ovisno o pronađenom uzorku. Najkompleksniji dio algoritma jest generiranje potrebnih tablica, ali nakon njihovog stvaranja moguće je koristiti algoritam u stvarnom vremenu.

Vektorizacija transformira sliku u vektorsku reprezentaciju neovisnu o rezoluciji te se pretvara u rastersku sliku tek po potrebi prikazivanja. Ovu metodu nije moguće koristiti za fotografije, ali geometrijske slike poput logotipa se najčešće spremaju na ovaj način.

2.2.2. Metode temeljene na učenju

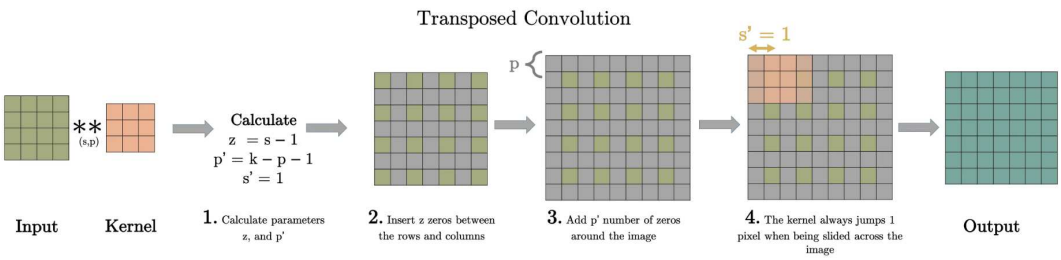
Transponirani konvolucijski sloj je sloj za naduzorkovanje koji generira mapu značajki većih dimenzija od ulazne te je sličan dekonvolucijskom sloju. Sloj obavlja obrnuti postupak od obične konvolucije i predviđa mogući ulaz za danu mapu značajki [1]. Isto kao i kod standardnog konvolucijskog sloja definira se podstava p *(eng. padding)*, korak s *(eng. stride)* i jezgra k *(eng. kernel)*. Ovi parametri odgovaraju i obrnutom postupku, dakle kada bi se obavljala konvolucija s istim vrijednostima na izlazu transponiranog sloja, rezultat bi bio jednakih dimenzija kao ulaz. Algoritam transponirane konvolucije obavlja se u 4 koraka:

1. Izračunaju se parametri
   $z = s - 1$
   $p' = k - p - 1$
   gdje je p debljina podstave, s veličina koraka i k dimenzija jezgre.

2. Između svakog retka i stupca postavi se z broj nula čime se dimenzije ulaza povećaju na $(2i - 1) \times (2i - 1)$, ako je i početna veličina kvadratnog ulaza.

3. Oko cijelog ulaza postavlja se podstava debljine p' broja nula.

4. Obavlja se standardna operacija konvolucije, ali s korakom veličine 1.



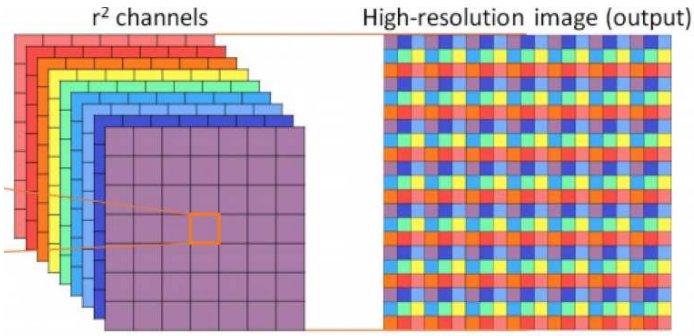
Slika 2.3: Koraci transponirane konvolucije [4]

Veličina izlaza za zadane i, k, p i s dana je u jednadžbi 2.7. Što je jezgra veća, svaki broj ulaza se raspršuje na šire područje što rezultira većim izlazom, a što su veći koraci, jezgra radi veće pomake pri stvaranju izlaza čime se on proširuje i postaje veći. Cilj transponiranih konvolucijskih slojeva je obaviti povećanje dimenzija s kraja na kraj mreže te osigurati kompatibilnost s ostalim slojevima. Jezgre sloja se adaptiraju kroz proces učenja kako bi što točnije i preciznije konstruirale izlaznu sliku, no mogući

su problemi nejednakih preklapanja jezgri sa slikom i umnožavanja istih rezultata koji štete učinku sustava ostavljajući rešetkaste artefakte [4].

$$o = (i - 1) * s + k - 2p \tag{2.7}$$

Subpikselni sloj obavlja operaciju koja se naziva miješanje piksela *(eng. pixel shuffle)*, radi s kraja na kraj mreže te je glavna ideja sloja povećati broj kanala ulaza korištenjem konvolucije, zatim preoblikovati dimenzije i prerasporediti vrijednosti kanala u širinu. Korištenjem operacija konvolucije generiraju se mape značajki jednake visine i širine te se spajaju u jednu cjelinu dimenzija $H \times W \times C * r^2$ gdje je H visina, W širina, C broj kanala, a r faktor skaliranja. Nakon miješanja dobiva se izlaz dimenzija $H * r \times W * r \times C$. Ulazu se na početku povećava dubina, a nakon toga se višak informacija preraspodjeli kako bi se dobila završna slika. Kanali jednog piksela se preurede u blok veličine $r \times r$, a njihov redoslijed je moguće adaptirati i učiniti ovisnim o okolini. Ova metoda ima veće receptivno polje zbog kojeg upija više informacija iz okoline i sustavima je dostupan kontekst potreban za stvaranje realističnih detalja, no ako se koristi neovisno predviđanje susjednih piksela u bloku, rezultat može izgubiti glatkoću [5].



Slika 2.4: Miješanje piksela [5]

***Meta upscale* modul** prvo je rješenje koje radi s proizvoljnim faktorom skaliranja te nije potrebno učiti ga sa svakim faktorom zasebno. Modul se sastoji od 3 glavna dijela, projekcija lokacije, predikcija težina i mapiranje značajki. Projekcija lokacije za svaki piksel izlaza (i, j) pronalazi odgovarajuću lokaciju (i', j') na slici niske rezolucije. Vrijednost piksela (i, j) bit će određena značajkom od (i', j'). Za mapiranje piksela koristi se operator dan jednadžbom 2.8:

$$(i', j') = T(i, j) = (\lfloor \frac{i}{r} \rfloor, \lfloor \frac{j}{r} \rfloor) \tag{2.8}$$

Ovaj dio modula omogućuje rad s proizvoljnim faktorima skaliranja r i ostvaruje frakcijske korake pa će tako za faktor skaliranja 2 svaki piksel dobiti 2 odgovarajuće točke, ali za faktor 1.5 neki će dobiti 2, a neki 1 odgovarajuću točku. Predikcija težina dinamički predviđa težine svake jezgre pomoću zasebne mreže, a broj jezgri se može razlikovati za svaki faktor skaliranja. Proces predikcije formuliran je na jednadžbi 2.9, gdje je W oznaka težina jezgre za piksel (i, j), φ funkcija predikcijske mreže, $\mathbf{v}_{ij}$ vektor srodnih točaka za piksel (i, j), a θ parametri predikcijske mreže [6].

$$W(i, j) = \varphi(\mathbf{v}_{ij}; \theta) \tag{2.9}$$

Mapiranje značajki odvija se na način da se izvuku značajke s odgovarajuće lokacije (i', j') i matrično se pomnože s predviđenim težinama za piksel (i, j) te se time dobije njegova vrijednost.

2.3. Ulazni podaci

2.3.1. *Single i multi-image sustavi*

Postoje dvije vrste postupaka super rezolucije prema količini podataka koja se unosi u sustav. U single-image sustavima dostupna je samo jedna slika niske rezolucije koja se mora mapirati u svoju verziju visoke rezolucije, a u multi-image sustavima više slika dolazi na ulaz, ali se sve koriste za konstrukciju samo jedne rezultantne slike na izlazu. Single-image sustavi rukuju s relativno malo informacija pa se na izlazu ponekad pojavljuju lažni uzorci koji nemaju uočljivu poveznicu s kontekstom izvorne slike visoke rezolucije. Ovo je vrlo nepoželjno svojstvo jer može uvesti dvosmislenost i loše utjecati na donošenje odluka u delikatnim područjima primjene poput medicine, no zadatak single-image sustava predstavlja veću prepreku i bliži je problematici stvarnoga svijeta [7].

Multi-image sustavi proizvode puno uvjerljivije rezultate zbog dodatnih informacija koje dobivaju na ulazu te mogu dobiti bolji uvid o stvarnim značajkama objekata na slici. Međutim, s količinom informacija povećava se i potreba za računalnim resursima te je dobivanje više različitih slika iste scene i niske rezolucije nije praktično niti je strogo definirano na koji način bi se slike trebale razlikovati. Uobičajeno je koristiti različite degradacijske funkcije na slici visoke rezolucije kako bi se dobilo više verzija slike u niskoj rezoluciji ili korištenje transformacija poput rotacija i podrezivanja na istoj slici. Ovakvi modeli bolje generaliziraju zbog raznolikosti ulaznih podataka, no

postoji mnogo načina na koje se koriste dodatne informacije. Na primjer, moguće je iz svake slike izvući mapu značajki istom mrežom pa ih spojiti i iskoristiti jednu mapu za stvaranje izlazne slike ili obraditi svaku sliku potpuno zasebno, zatim spojiti njihove rezultate [7].

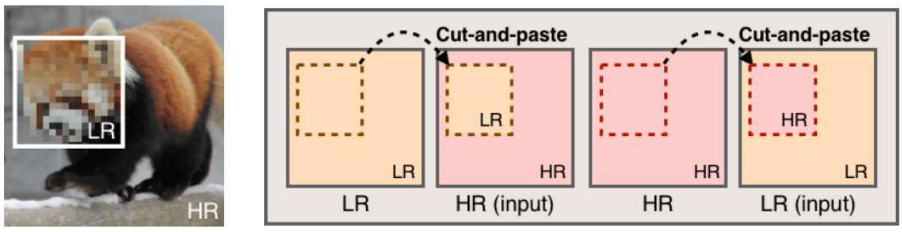
2.3.2. Skupovi podataka

Za sve problem koji se pokušavaju riješiti dubokim učenjem s nadziranim pristupom nužno je imati kvalitetne podatke. S obzirom na to da je dobivanje LR slike iz postojeće HR slike relativno jednostavan zadatak, mnogo skupova podataka nastaje upravo na taj način, sakupljanjem i degradacijom. Degradacije mogu biti jednostavne ili koristiti složenije postupke kako bi se približile originalnim slikama nastalim pomoću kamera niske rezolucije. Jedan od najpoznatijih skupova podataka nastao u sklopu *NTIRE2017* SR natjecanja jest *DIV2K* koji se sastoji od 800 slika za učenje, 100 za validaciju i 100 za testiranje, a slike prikazuju raznolike vrste scena i prosječna rezolucija im je 1972×1437 piksela. Sličan skup podataka jest *Flickr2K* koji se sastoji od ukupno 2650 slika preuzetih s *Flickr* platforme te je često kombiniranje ova dva skupa kako bi se iskoristilo što više podataka. Stariji skupovi podataka s nižim rezolucijama (u razinama od 200 do 1000 piksela maksimalno po dimenziji) koji se ponekad koriste za učenje jesu *BSD300*, *BSD500*, *T91* i *Urban100*. Dva najpoznatija skupa za testiranje koji su postali standard usporedbe jesu *Set5* i *Set14*. Sastoje se od malog broja slika (5 i 14) niskih dimenzija, ali su postali gotovo neizostavan dio procesa testiranja modela. Zanimljiv je i projekt *ImagePairs* koji koristi posebno hardversko rješenje s dvije kamere kako bi uhvatio isti kut i širinu scene, ali u dvije rezolucije od kojih je jedna 3 puta veća od druge. Skup podataka sastoji se od 8591 parova za učenje i 2830 za testiranje, a svrha je dobivanje velikog skupa stvarnih slika za super rezoluciju.

2.3.3. Augmentacija podataka

Jedan od najčešće korištenih postupaka za povećanje skupa podataka, ali i poboljšanje performansi modela jest augmentacija podataka. Pokazalo se da postupci koji stvaraju nerealistične uzorke u podacima mogu naštetiti učinku modela pa je važno pažljivo odabrati koje metode augmentacije koristiti. Klasične korisne metode jesu izrezivanje (korištenje segmenata umjesto cijele slike), rotacije i zrcaljenje. Dodatne metode razvijene sa super rezolucijom na umu jesu miješanje RGB kanala koje poboljšava performanse kod skupova podataka s neuravnoteženosti boja te *CutBlur* metoda (2.5) koja ubacuje isječak LR slike unutar odgovarajućeg dijela HR slike kako bi samo

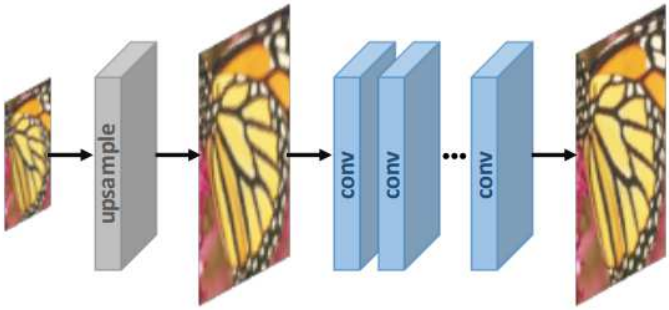
dio bio niske rezolucije i obrnuto, no takav postupak zahtijeva naduzorkovanje LR slike prije ulaska u mrežu [8].



Slika 2.5: *CutBlur augmentacija [8]*

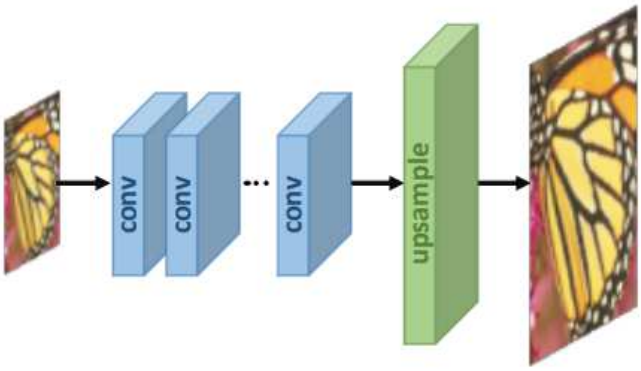
2.4. Poredak naduzorkovanja

Osim po ulaznim podacima, sustavi se razlikuju i po tome kada obavljaju naduzorkovanje i povećavanje dimenzija. Kod naduzorkovanja prije *(eng. pre-upsampling)*, slici se prvo povećaju dimenzija na zadanu veličinu korištenjem neke od klasičnih metoda poput bikubične interpolacije te se takva slika predaje na ulaz dubokog modela. Ovaj pristup je najučestaliji kod dubokih konvolucijskih mreža [9]. Takve mreže uglavnom sporo konvergiraju pri niskim stopama učenja, ali nije moguće samo povećati stope zbog eksplodirajućih gradijenata. Potrebno je koristiti tehnike poput rezidualnih veza, gdje se neki slojevi neuronske mreže povezuju iako nisu susjedni i rezanja gradijenata, metoda u kojoj se postavlja prag na derivaciju pogreške tijekom učenja, kako bi se adresirali ovi problemi. Naduzorkovanje ulaza prije predavanja mreži povećava računalni trošak, pogotovo kod konvolucijskih slojeva gdje broj koraka ovisi o početnoj rezoluciji, a sam postupak ne donosi nikakve nove informacije sustavu. Također, predefinirano naduzorkovanje dovodi do naglašavanja šuma i zamućivanja slike. No, ovakvi modeli čine proces učenja jednostavnijim i pretvaraju problem povećanja razlučivosti u problem rafiniranja slike te mogu raditi s proizvoljnim dimenzijama i faktorima skaliranja [7].



Slika 2.6: Pre-upsampling [1]

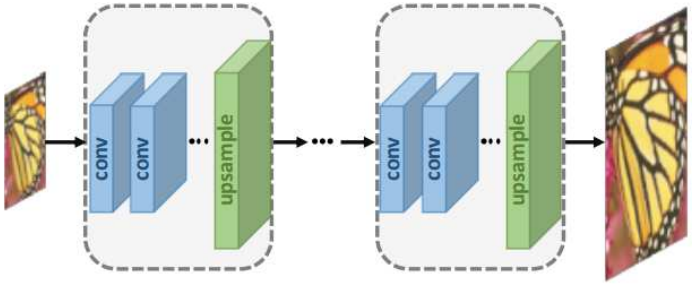
Kod sustava koji koriste naduzorkovanje poslije *(eng. post-upsampling)*, slika niske rezolucije prvo se obrađuje dubokim modelom i radi se izvlačenje značajki, a na kraju mreže obavlja se povećanje dimenzija prolaskom kroz poseban sloj. S obzirom na to da se najskuplji procesi odvijaju u prostoru niskih dimenzija, količina potrebnih računalnih resursa i složenost je značajno smanjena te su zbog toga sustavi s ovim poretkom jako popularni. Sloj za naduzorkovanje ne koristi klasične metode, već upotrebljava tehnike temeljene na učenju. Problem ovakvih sustava jest taj što se naduzorkovanje obavlja u jednom koraku, što otežava proces učenja kod većih faktora povećanja te svaki različiti faktor zahtijeva stvaranje i učenje novog modela [7].



Slika 2.7: Post-upsampling [1]

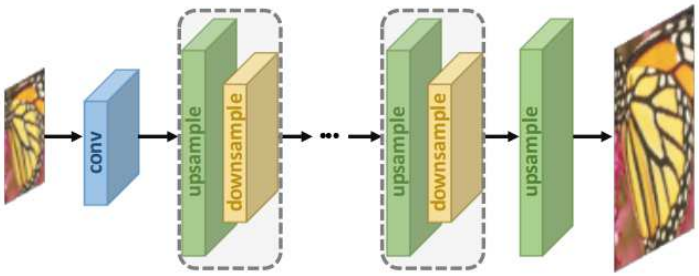
Progresivno naduzorkovanje postepeno obavlja povećanje dimenzija, čineći piramidalnu strukturu. U svakoj fazi modela slika prolazi kroz kaskadu konvolucijskih slojeva te joj se povećava rezolucija. Razlaganjem procesa na više jednostavnijih zadataka postiže se rasterećenje, model se lakše uči i smanjuju se prostorni i vremenski zahtjevi. Korištenje različitih faktora skaliranja također postaje jednostavnije jer mreža

ima međurezultate koje može iskoristiti kao završne. Arhitekture ovakvih sustava su složenije, što narušava njihovu stabilnost i uvodi potrebu za oblikovanjem robusnijih algoritama i strategija učenja [1].



Slika 2.8: Progresivno naduzorkovanje [1]

Iterativno uzorkovanje gore i dolje novija je metoda u kojoj se izmjenjuju slojevi za poduzorkovanje i naduzorkovanje. Slika prolazi kroz petlju u kojoj se nakon povećanja rezolucija ponovno smanjuje, računa se razlika ulazne i dobivene LR slike te se korigira korištenjem naduzorkovane razlike. Ovakvi se modeli ističu zbog naglašavanja veze između parova slika niske i visoke rezolucije te postižu rezultate visoke kvalitete, ali potrebno je daljnje istraživanje kako bi se ovakve mreže usavršile [1].



Slika 2.9: Iterativno uzorkovanje [1]

2.5. Evaluacijske metrike

Vizualna evaluacija uspješnosti sustava jako je subjektivna i teško je precizno ustanoviti njegovu učinkovitost. Potrebno je ustanoviti standardne mehanizme mjerenja učinkovitosti sustava pomoću kojih se oni uspoređuju

PSNR *(eng. peak signal-to-noise ratio)* objektivna je mjera koja računa kvalitetu slikovne rekonstrukcije nakon transformacije sa šumom. Mjera se računa između dvije

slike, a izražava se u decibelima (dB). Što je $PSNR$ viša vrijednost, veća je kvaliteta rekonstruirane slike. U izračunu se koristi maksimalna vrijednost piksela L i srednja kvadratna pogreška između dvije slike MSE. Prije izračuna samog $PSNR$, potrebno je izračunati MSE sumiranjem svih kvadrata razlika između istih piksela na izvornoj HR i rekonstruiranoj SR slici i dijeljenjem te sume s ukupnim brojem piksela u jednoj slici, kao što je prikazano u jednadžbi 2.10. M i N predstavljaju dimenzije slike, a m i n koordinate piksela, a $I(i)$ je intenzitet piksela slike.

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M * N} \tag{2.10}$$

$$PSNR = 10\log_{10}(\frac{L^2}{MSE}) \tag{2.11}$$

Ako se koriste 8-bitne reprezentacije za vrijednost piksela, vrijednost L iznosi 255. $PSNR$ je fokusiran na individualne razlike piksela te se kod slika s bojom ponekad računa samo na kanalu svjetlosnog intenziteta *(eng. luma channel)* koji je zapravo crno-bijela verzija slike. Vrijednost $PSNR$ nije idealan pokazatelj da će ljudsko oko percipirati sliku kao kvalitetnu, no i dalje je u vrlo čest uporabi zbog nedostatka dobrih perceptivnih metrika [10].

Indeks strukturne sličnosti *(eng. structural similarity index, SSIM)* zasniva se na sposobnosti ljudskog oka da opaža strukturu scene i uočava razlike između informacija izvučenih iz izvorne HR i izgrađene SR slike. Indeks iz slike izvlači 3 glavne značajke: svjetlinu, kontrast i strukturu te izvodi usporedbu na temelju njih. $SSIM$ izračunava vrijednost između -1 i 1, gdje pozitivni rezultati indiciraju da su slike slične, a negativni različite te se ponekad ove vrijednosti prilagode na raspon $[0, 1]$. Svjetlina slike označava se sa μ, a računa se kao prosjek intenziteta svih piksela kako je prikazano na jednadžbi 2.12.

$$\mu_I = \frac{1}{N}\sum_{i=1}^{N} I(i) \tag{2.12}$$

Kontrast se računa kao standardna devijacija (korijen varijance) svih intenziteta piksela. Označava se sa σ te se računa pomoću μ kao što je prikazano na jednadžbi 2.13.

$$\sigma_I = (\frac{1}{N-1}\sum_{i=1}^{N}(I(i) - \mu_I)^2)^{\frac{1}{2}} \tag{2.13}$$

Strukturna usporedba obavlja se konsolidiranom formulom, a izražava se tako da se slika normalizira, tj. signalu I se oduzima srednja vrijednost μ i dijeli se sa standardnom devijacijom σ:

$$\frac{I - \mu}{\sigma} \tag{2.14}$$

Jednadžbe usporedbe μ i σ koriste proizvoljne konstante K_i i maksimalnu vrijednost piksela L,

$$c_\mu(I, I') = \frac{2\mu_I \mu_{I'} + C_1}{\mu_I^2 + \mu_{I'}^2 + C_1} \tag{2.15}$$

$$c_\sigma(I, I') = \frac{2\sigma_I \sigma_{I'} + C_2}{\sigma_I^2 + \sigma_{I'}^2 + C_2} \tag{2.16}$$

gdje se C_i računa izrazom:

$$C_i = (K_i L)^2 \tag{2.17}$$

Jednadžba usporedbe strukture glasi,

$$c_s(I, I') = \frac{\sigma_{II'} + C_3}{\sigma_I \sigma_{I'} + C_3} \tag{2.18}$$

gdje je $\sigma_{II'}$ jednaka,

$$\sigma_{II'} = \frac{1}{N-1} \sum_{i=1}^{N} (I(i) - \mu_I)(I'(i) - \mu_{I'}) \tag{2.19}$$

te je na kraju potpuna jednadžba za izračun $SSIM$ dana u izrazu 2.20, u kojem se kontrolni parametri α, β i γ postavljaju ovisno o važnosti pojedinih značajki. No, pokazalo se kako je puno učinkovitije primjenjivati $SSIM$ na manjim regijama u usporedbi s korištenjem mjere na globalnoj razini slike, a onda im izračunati prosjek [11].

$$SSIM(I, I') = [c_\mu(I, I')]^\alpha * [c_\sigma(I, I')]^\beta * [c_s(I, I')]^\gamma \tag{2.20}$$

Srednja ocjena mišljenja *(eng. mean opinion score, MOS)* subjektivna je metoda koja daje dobar uvid u ljudsku perspektivu na slike, stoga se pokušava upotrebljavati kada je to moguće, ali njena provedba zahtijeva veću količinu ljudi. Od ljudi se traži da ocijene sliku na skali od 1 do 5 te se svi rezultati zbroje i izračuna se njihov prosjek. Subjektivne mjere pate jer nemaju strogo definirane linearne skale te ljudska pristranost i varijanca uvijek utječu na vrijednost, no korisno je mjeriti ljudsku percepciju na

taj način jer ponekad se događa da modeli koji nisu uspješni prema računskim skalama postižu visoke ocjene u subjektivnim mjerama.

Procjena kvalitete temeljena na učenju postupak je u kojem se modeli uče predviđati ocjene kvalitete slika. Cilj je stvoriti model koji s kraja na kraj može analizirati sliku i ocijeniti njenu kvalitetu. Za pristup je važno da radi i uči na velikim skupovima podataka kako bi imao što manje pristranosti te se mogu koristiti tehnike nenadziranog ili polunadziranog učenja gdje se modelu ne daje izvorna HR slika. Modeli se ponekad uče na slikama s objektivnim defektima i naglašenim šumom kao primjerima slika loše kvalitete. Potrebno je procijeniti je li važnija realističnost slika ili konzistentnost sa svojim izvornim HR pandanom [1].

2.6. Funkcije gubitka

Važan dio svakog algoritma strojnog učenja jesu funkcije gubitka. One određuju smjer optimizacije i vode proces učenja prema konvergenciji. U području super rezolucije postoji velik broj različitih funkcija gubitka koje uvelike utječu na učinkovitost i performanse modela te se ponekad kombinira korištenje više funkcija gubitka za jedan model [1].

Gubitak piksela *(eng. pixel loss)* intuitivna je i često upotrebljavana funkcija gubitka koja koristi isti princip kao i evaluacijska mjera $PSNR$. Cilj funkcije je minimizirati razlike između piksela izvorne HR i rezultantne SR slike, a za to koristi srednju apsolutnu (L1) ili srednju kvadratnu pogrešku (L2). Obje varijante gubitka prikazane su na jednadžbama 2.21 i 2.22. Simbol $\mathcal{L}$ predstavlja funkciju gubitka koja prima izvornu HR sliku I i sliku dobivenu pomoću SR I', a N sve vrijednosti piksela. Ako se radi o slici u boji, svaki kanal se promatra kao odvojena vrijednost [12].

$$\mathcal{L}(I', I) = \frac{1}{N} \sum_{i=1}^{N} |I'(i) - I(i)| \tag{2.21}$$

$$\mathcal{L}(I', I) = \frac{1}{N} \sum_{i=1}^{N} (I'(i) - I(i))^2 \tag{2.22}$$

Gubitak piksela prisiljava generiranu SR sliku da bude što bliža izvornoj HR slici u vrijednostima piksela, ali zbog naglaska na individualne piksele gubi se cjelokupni dojam i rezultatima nedostaje oštrina i detaljnost.

Charbonnier gubitak posebna je vrsta gubitka piksela kojoj se dodaje konstanta ϵ reda veličine oko 10^-3 kao kontrolni parametar koji održava stabilnost. Taj sitan dodatak uspio je poboljšati rezultate običnog pikselnog gubitka pa tako stvara oštrije slike i lakše se nosi s izuzecima [12]. Iznos gubitka računa se jednadžbom 2.23.

$$\mathcal{L}(I', I) = \frac{1}{N} \sum_{i=1}^{N} \sqrt{(I'(i) - I(i))^2 + \epsilon^2} \tag{2.23}$$

Gubitak teksture *(eng. texture loss)* funkcija je čiji je cilj održati konzistentan stil slike. Konvolucijske neuronske mreže jako su dobre u stvaranju tekstura visoke kvalitete. Stil slike je reprezentiran kao korelacija između kanala mape značajki te se definira kao Gramova matrica G. Svaki element matrice jest skalarni produkt vektoriziranih mapa značajki i i j na određenoj dubini l. Sve Gramove matrice su kvadratnih dimenzija koje se označavaju sa c_l. Jednadžba gubitka teksture na dubini l glasi:

$$\mathcal{L}(I', I, \phi, l) = \frac{1}{c_l^2} \sqrt{\sum_{i,j} (G_{i,j}^{(l)}(I') - G_{i,j}^{(l)}(I))^2} \tag{2.24}$$

Modeli koji učenje provode s gubitkom teksture generiraju slike koje su ljudskoj percepciji vrlo realistične, no važno je odabrati optimalnu veličinu okvira koji ih obuhvaća [13].

Gubitak sadržaja *(eng. content loss)* pokušava izjednačiti značajke visoke razine u generiranoj slici s onima od izvorne HR slike i tako održati perceptivni doživljaj. Umjesto učenja piksela da imaju egzaktnu vrijednost kao i na izvornoj HR slici, ovaj gubitak potiče generiranu sliku da ima slične reprezentacije u obliku mapi značajki. Funkcija računa $L2$ udaljenost između reprezentacija nastalih izvlačenjem značajki, a taj proces obavljaju unaprijed naučene neuronske mreže, što se označava sa $\phi(I)$. Neke od najpoznatijih mreža za izlučivanje značajki su VGG i *ResNet* te se njihove izlazne mape značajki uspoređuju kako bi se dobila vrijednost gubitka sadržaja. Jednadžba gubitka sadržaja dana je u izrazu 2.25, gdje HWC predstavlja dimenzije mapa značajki. Cilj funkcije je generirati vizualno zadovoljavajući rezultat koji je ugodan ljudskom oku pa se zato ovaj gubitak često primjenjuje u praksi [1].

$$\mathcal{L}(I', I, \phi) = \frac{1}{HWC} \|\phi(I') - \phi(I)\|_2^2 \tag{2.25}$$

Suparnički gubitak *(eng. adversarial loss)* posebna je vrsta funkcije gubitka koja se koristi u generativnim suparničkim mrežama *(eng. generative adversarial networks,*

GAN). U ovoj arhitekturi koriste se dvije neuronske mreže, generator i diskriminator. Generator uči replicirati slike iz neke distribucije, a diskriminator se kroz sljedstveno učenje priprema za obavljanje binarne klasifikacije u kojoj razlučuje prave primjerke od generiranih. Obje mreže sudjeluju u natjecanju gdje jedna pokušava nadmudriti drugu. Suparničko učenje potrebno je raditi iterativno, nakon poboljšanja generatora ponovno se trenira diskriminator i obrnuto. Model koji provodi super rezoluciju može se smatrati generatorom, a u sustav se uvodi nova mreža koja preuzima ulogu diskriminatora koji će prosuđivati je li slika originalna ili generirana. Jednadžbe gubitka nisu univerzalne pa tako postoje varijante s unakrsnom entropijom (2.26 i 2.27) i kvadratnom pogreškom (2.28 i 2.29). Oznaka D odnosi se na funkciju diskriminatora, tj. rezultat binarne klasifikacije, a I_S su nasumično odabrane izvorne HR slike. Svaka mreža ima zasebnu funkciju gubitka [12].

$$\mathcal{L}_{gen}(I'; D) = -logD(I') \tag{2.26}$$

$$\mathcal{L}_{dis}(I', I_S; D) = -logD(I_s) - log(1 - D(I')) \tag{2.27}$$

$$\mathcal{L}_{gen}(I'; D) = (D(I') - 1)^2 \tag{2.28}$$

$$\mathcal{L}_{dis}(I', I_S; D) = (D(I'))^2 + (D(I_S) - 1)^2 \tag{2.29}$$

Postoji još mnogo varijacija funkcija gubitaka poput gubitka cikličke konzistencije koji poduzorkuje rezultantnu SR sliku i računa gubitak piksela s ulaznom slikom, ili gubitak potpune varijacije koji računa apsolutne razlike susjednih piksela i sl. U praksi je uobičajeno kombinirati i isprobati više funkcija s težinskim koeficijentima kako bi se ostvario kompromis između percepcije i točnosti modela.

2.7. Arhitekture mreža

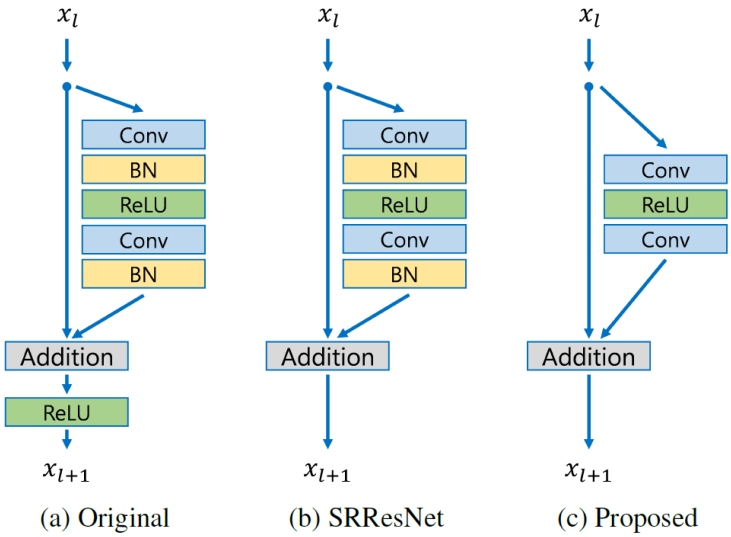
Arhitektura sustava vrlo je važan čimbenik koji utječe na uspješnost modela te isto kao i kod funkcija gubitka, postoji pregršt različitih principa i pristupa dizajnu mreže koji donose raznovrsne prednosti, ali i ograničenja sustavu.

2.7.1. Rezidualne neuronske mreže

Kako se duboko učenje i računala razvijaju, modeli postaju sve dublji s velikim brojem slojeva koji rade komplicirane računske operacije. Kod dubokih modela javlja se posebna vrsta problema nestajućih i eksplodirajućih gradijenata. Tijekom procesa učenja unazadnom propagacijom gradijenti mogu postati ekstremno mali ili veliki pa može doći čak i do prelijevanja što čini optimizaciju vrlo teškom za kontrolirati. Također se pojavljuje i problem degradacije ili stagniranja modela dodavanjem novih slojeva što ukazuje na njegovu prezasićenost [14].

Rezidualni blokovi glavna su komponenta rezidualnih mreža, a u njima se događa dodatna operacija dodavanja identiteta izlazu, dakle početni ulaz se nakon obrade ponovno zbraja u izlazu bloka. No, ovi blokovi ne obuhvaćaju svaki sloj mreže zasebno, već postoje preskočne veze pa se tako ulaz pribraja izlazu tek nakon nekoliko slojeva mreže. Na preskočnim vezama ne događa se nikakva dodatna obrada, već se samo prenosi ulaz prijašnjih slojeva unaprijed. Ovi blokovi zbog prisutnosti identiteta olakšavaju tok gradijenta te se slažu zajedno u kontinuitetu što omogućava građenje jako dubokih mreža. Arhitekture s rezidualnim blokovima često koriste sloj sažimanja prije zadnjeg sloja kako bi stvorili kompaktnu reprezentaciju cijele mape značajki. Većina modernih dubokih mreža koristi nekakvu vrstu rezidualnih blokova zato što oni dokazano unapređuju rad modela i olakšavaju proces učenja.

EDSR *(eng. Enhanced Deep Residual Network)* arhitektura bazirana je na *SRResNetu* i sastoji se od 32 sloja i 256 kanala značajki. U modelu je implementirana posebna vrsta rezidualnog bloka bez *BatchNorm* (BN) slojeva koji su normalizirali podatke nakon konvolucija jer su ograničavali raspon mreže (2.10). Uklanjanje tih slojeva dovelo je do napretka u točnosti, smanjilo je memorijsko opterećenje i ubrzalo proces učenja.



Slika 2.10: Rezidualni blokovi [15]

MDSR *(eng. Multi-scale Deep Residual Network)* ekstenzija je EDSR modela koja omogućuje skaliranje s više faktora (2x, 3x i 4x). Na početku mreže moduli su razdvojeni ovisno o faktoru skaliranja te obavljaju konvoluciju s 5×5 jezgrama kako bi mreža ostala plitka, a zadržala veliko receptivno polje. U sredini mreže slojevi su dijeljeni, a na kraju se ponovno granaju moduli za naduzorkovanje za svaki faktor [15].

2.7.2. Rekurzivne neuronske mreže

Klasa dubokih neuronskih mreža koja se bavi strukturiranim podacima jesu rekurzivne mreže. Takve mreže grade uređene izlaze rekurzivnom primjenom istih težina, što znači da apliciraju vlastit slojeve na svoj izlaz. Stablasta struktura mreže omogućuje rukovanje hijerarhijskim podacima. Izlazi čvorova više djece mogu se kombinirati u roditeljskim čvorovima i prisutno je dijeljenje parametara, tj. težina. Rekurzivni blokovi održavaju model kompaktnim i poboljšavaju točnost modela tako što mu daju dodatnu dubinu, a ne povećavaju broj parametara koje je potrebno optimizirati [12].

DRCN *(eng. Deep recursive convolutional network)* koristi ideju rekurzije pa tako mreža uključuje primjenu istih konvolucijskih slojeva više puta. Njeni konvolucijski slojevi nalaze se u rezidualnom bloku i dijeljeni su. Izlazi svih posrednih konvolucijskih slojeva šalju se u rekonstrukcijski sloj zajedno s ulazom gdje se gradi završna HR slika.

DRRN *(eng. Deep recursive residual network)* unosi napredak na DRCN mrežu tako što unutar rekurzivnih blokova koristi rezidualne blokove umjesto običnih konvolucijskih slojeva. Unutar tih rezidualnih blokova konvolucijski slojevi su dijeljeni. Mreža postiže dobre rezultate dok zadržava nizak broj parametara [16].

Regionalne rekurzivne mreže

Posebna vrsta rekurzivnih mreža jesu regionalne rekurzivne mreže. Većina modela uzima super rezoluciju kao zadatak u kojem generirani pikseli nemaju međuovisnosti. U regionalnim rekurzivnim mrežama predložen je pristup generiranja izlaza piksel po piksel te uporaba dvije mreže od kojih je jednoj zadatak uhvatiti globalni kontekst, a drugoj serijski zavisna generacija piksela. Ovakve metode imaju vrlo visoku MOS ocjenu te uspješno povećavaju rezoluciju kompleksnih tekstura poput ljudske kose i kože. Problem ovakvog pristupa je taj što se radi o izrazito računski skupom i dugotrajnom procesu.

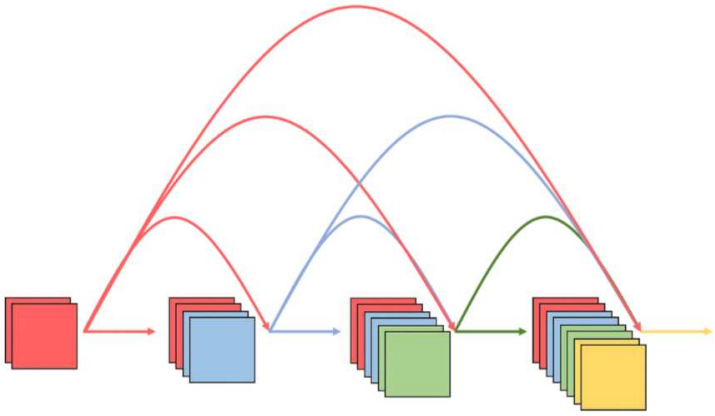
2.7.3. Mreže s više puteva

Mreže s više putova *(eng. multi-path networks)* provode iste podatke i značajke kroz nekoliko različitih puteva koji obrađuju iste informacije, ali drukčijim funkcijama. Postoje globalni, lokalni i putovi specifični faktoru skaliranja. Globalni putovi izvlače značajke različitih aspekata slike i protežu se kroz cijelu mrežu pa tako model može imati odvojene putove za značajke visoke i niske rezolucije. Putovi se mogu, ali ne moraju povezati na kraju, no njihovo spajanje dovodi do akumulacije raznolikih informacija s kojima se konstruiraju kvalitetnije slike. Lokalni putovi puno su kraći od globalnih te može biti riječ samo o dva konvolucijska sloja na istoj razini mreže koji primaju jednaki ulaz. Takvi slojevi najčešće imaju različite veličine jezgre i nakon procesa spajanja dodatno se radi konvolucija s jezgrom veličine 1×1 čija je uloga sažimanje kanala i smanjenje dimenzionalnosti. Ovaj pristup omogućava prikupljanje značajki s različitih skala na manjoj razini [1]. Putovi specifični faktoru skaliranja osiguravaju modelu sposobnost rada s više faktora skala. Ovisno o faktoru na ulazu se obavljaju različite operacije pretprocesiranja i naduzorkovanja, Proces učenja također izolira putove specifične skali te se samo zajednički slojevi uvijek optimiziraju. Primjer ovakve mreže jest spomenuti MDSR [15].

2.7.4. Ansambl

Ansambl *(eng. ensemble)* vrsta je arhitekture sustava koja koristi više jednakih ili različitih modela za stvaranje izlaza. *Bagging* ansambl koristi iste vrste modela, ali im daje različite podatke za učenje te se zaključivanje radi tako da se isti ulaz provuče kroz svaki model pa se izlaz svakog modela kombinira u završnu sliku. U području super rezolucije se ponekad koriste isti podaci za učenje kod svih modela, ali se svakom predaje rotirana verzija ulaza u vrijeme inferencije. *Stacking* ansambli koriste niz različitih modela s istim podacima. Završni izlaz nastaje tako što se posebno izdvojenom modelu predaju izlazi svih ostalih modela i on stvara rezultantnu sliku. Ovakav pristup također je moguć kod problema super rezolucije i daje bolje rezultate nego korištenje jednog jednostavnog modela, ali vremenski je zahtjevan proces [17].

2.7.5. Guste veze



Slika 2.11: Gusta mreža [18]

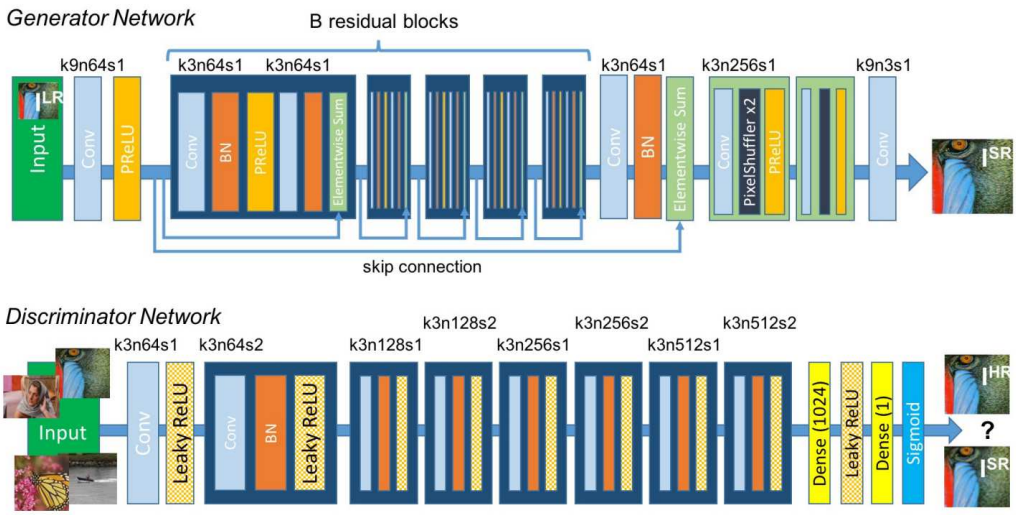
U klasičnim neuronskim mrežama guste veze odnose se na slojeve koji su potpuno povezani, gdje je svaki neuron povezan sa svakim neuronom sljedećeg sloja. Kod konvolucijskih mreža guste veze imaju drugačije značenje te se radi o konceptu sličnom rezidualnim vezama. Guste konvolucijske mreže *(eng. DenseNet)* za svaki sloj gustog bloka uzimaju mape značajki svih prijašnjih slojeva kao ulaz, što vodi do $l * (l - 1)/2$ veza u bloku sa l slojeva 2.11. Ova arhitektura maksimizira tok informacija kroz mrežu, značajno pomaže s problemom nestajućih gradijenata, smanjuje ukupan broj parametara i potiče ponovnu uporabu značajki. Za razliku od *ResNeta*, *DenseNet* ulančava identitetne mape kako bi ih učinkovitije sačuvao. Zbog toga što konvolucija smanjuje dimenzije na izlazu postavlja se pitanje kompatibilnosti svih ulaza, a odgovor na

taj problem su tranzicijski slojevi. Tranzicijski slojevi poduzorkuju mape značajki korištenjem normalizirajućeg *BatchNorm* sloja (ovaj sloj je često izbačen iz modela koji se bave super rezolucijom jer često ima negativan učinak), 1×1 konvolucijskog sloja i 2×2 sloja sažimanja. Gusti blokovi mogu imati i 1×1 konvolucijski sloj prije svake 3×3 konvolucije koji se naziva sloj uskog grla *(eng. bottleneck)*. Guste mreže imaju nizak hiperparametar k koji predstavlja stopu rasta zato što svaki sloj ima pristup svim prijašnjim mapama značajki i svjestan je globalnog stanja mreže [18]. *SRDenseNet* primjer je guste mreže za super rezoluciju, sastoji se od 69 slojeva i uvodi guste veze između samih gustih blokova [19].

2.7.6. Generativne suparničke mreže

Modeli koji koriste suparnički gubitak jesu generativne suparničke mreže. Temeljni koncept suparničkih mreža prisutan je u brojnim arhitekturama za super rezoluciju, no uvođenje ideja GANova nije tako jednostavno te samo dodavanje suparničkog gubitka ne bi postiglo željeni cilj. Natjecanje generatora i diskriminatora kroz koje se odvija učenje i dalje je prisutno te diskriminator i obavlja binarnu klasifikaciju rezultata generatora na stvarne i lažne primjerke. Potrebno je koristiti dodatnu funkciju gubitka uz suparnički gubitak kako bi se postiglo realistično i vizualno ugodno povećanje rezolucije. Također su moguće i promjene same građe kako bi se unaprijedila klasična GAN arhitektura za super rezoluciju.

SRGAN je prvi sustav koji je iskoristio suparnički gubitak u super rezoluciji te ga je kombinirao s gubitkom sadržaja i ostvario izvrsne rezultate koje ljudi percipiraju realističnima što je potvrdila i visoka MOS ocjena [20]. Generator je koristio arhitekturu *SRResNeta* dubine 16 blokova i gubitak sadržaja računan na mapama značajki generiranih pomoću VGG mreže. Cilj uvođenja diskriminatora bio je povećanje realističnosti generiranih slika, a sastoji se od 8 konvolucijskih slojeva s rastućim brojem jezgri, dva gusto povezana sloja i sigmoidnom funkcijom za klasifikaciju na samom kraju (2.12).



Slika 2.12: SRGAN arhitektura [20]

ESRGAN je nadogradnja na standardnu arhitekturu u kojoj se koristi relativistički diskriminator [12]. Takav diskriminator ne govori samo je li slika prava ili lažna, već procjenjuje koliko je stvarna slika više realistična od generirane i obrnuto. Izlaz diskriminatora više nije vjerojatnost da je slika prava, nego relativni "realizam" prave slike u usporedbi s lažnom. U implementacijama se često koristi prosječna vrijednost "realizma" serije slika. Ulaz u diskriminator više nije samo izlazna slika generatora, već par slika, prava i generirana verzija. Ovom zamjenom proces učenja postaje stabilniji jer relativna razlika stvarne i lažne slike pruža stabilnije gradijente te stvara kvalitetnije rezultate jer generator ne pokušava stvarati samo prolazne slike, već što realističnije slike. Postoji još dodatnih nadogradnji na SRGAN model poput Real-ESRGANa [21] koji je treniran na sintetičkim podacima modeliranim degradacijskim procesom visokog reda. Degradacijski model uzima u obzir brojne vrste defekta poput zamućenja fotoaparata, artefakata i šuma.

2.7.7. Difuzijski modeli

Generativni model koji progresivno dodaju šum podacima, a zatim uče obrnuti proces zovu se difuzijski modeli. Osnovni koncept modela jest postupno degradiranje podataka i rekonstrukcija u izvorni ili sasvim novi oblik, što zvuči vrlo slično problemu super rezolucije. Model se na početku uči uništavanju podataka uvođenjem šuma. Najčešće se uzorkuje Gaussova distribucija za šum koji se inkrementalno dodaje ulazu. Obrnuti proces uključuje prepoznavanje specifičnih uzoraka šuma dodanih

u unaprijednom procesu i učenje mreže uklanjanju šuma u skladu s tim. Model koristi svoje znanje kroz kompleksan Markovljev lanac, predviđa ukupni šum na svakom koraku i uklanja njegov dio. Razlikujemo vjerojatnosne modele *(eng. denoising diffusion probabilistic model, DDPM)* koji koriste statističko varijacijsko zaključivanje i modele bazirane na ocjeni *(eng. score-based generative model, SGM)* koji računaju procjenu je li slika vjerojatnija od susjednih koristeći stohastičke diferencijalne jednadžbe.

Problem difuzijskih modela jest njihova brzina rada. Difuzijski modeli obavljaju i po tisuće koraka tijekom uklanjanja šuma, a procesi ubrzanja uvelike štete rezultatima modela, pogotovo kod zadatka super rezolucije. *ResShift* primjer je učinkovitog difuzijskog modela za super rezoluciju koji koristi pomicanje reziduala između LR i HR slike te dobiva dobre rezultate u samo 15 koraka. Sustav koristi naduzorkovanje prije te se dimenzije slike ne mijenjaju tijekom samog procesa uklanjanja šuma [22].

2.7.8. Mehanizam pozornosti

Tehnika koja se koristi za unaprjeđenje modela u dubokom učenju tako što ih usredotočuje na relevantne informacije naziva se mehanizam pozornosti *(eng. attention mechanism)*. Ovaj mehanizam pruža modelu mogućnost da se selektivno posvećuje obradi pojedinih segmenata ulaznih podataka tako što pridaje različite stupnjeve važnosti elementima. Generiranje težina pozornosti za različite značajke i elemente slike jedan je od najvažnijih zadataka mehanizma. Te težine utvrđuju koliki će doprinos imati određene značajke izlazu modela, a računaju se na temelju sličnosti ili relevantnosti elemenata slike s postavljenim upitom.

Tri glavna čimbenika mehanizma jesu:

- upit *(eng. query, Q)* - predstavlja trenutni kontekst ili fokus modela
- ključ *(eng. key, K)* - predstavlja elemente ili značajke ulaznih podataka
- vrijednost *(eng. value, V)* - predstavlja egzaktnu vrijednost povezanu s ključem.

Svakom čimbeniku pridružena je kvadratna matrica težinskih parametara koji se mogu optimizirati učenjem. Mjerenjem sličnosti između upita q i ključeva k dolazi se do stupnjeva važnosti e (2.30) koji se provlače kroz softmax funkciju kako bi se dobile težine w (softmax pretvara niz brojeva u distribuciju vjerojatnosti, 2.30), a potom se one uparuju i množe s odgovarajućim vrijednostima v u ulazu čime se dobiva vektor konteksta (2.31). Vektor konteksta dalje se koristi za dobivanje izlaza. Mehanizam pozornosti čini modele učinkovitijima i preciznijima tako što im obraća pažnju na bitne

uzorke ili ovisnosti u podacima. Pozornost olakšava rad s varijabilnim veličinama podataka jer dinamički pristupa različitim dijelovima ulaza i interpretabilna je te pruža uvid u proces zaključivanja. Mehanizam pozornosti fundamentalna je komponenta transformera, a originalni koncept je zamišljen zbog fiksnih duljina enkodirajućih vektora zbog kojih je dekoder imao ograničen pristup informacijama sadržanim u ulaznim podacima. Modul pozornosti može imati više "glava" pomoću kojih je mreža sposobna istovremeno se baviti različitim dijelovima ulaznih podataka [23].

$$e_{q,k_i} = q * k_i \tag{2.30}$$

$$w_{q,k_i} = softmax(e_{q,k_i}) \tag{2.31}$$

$$attention(q, K, V) = \sum_i w_{q,k_i} * v_{k_i} \tag{2.32}$$

U području super rezolucije zanimljiva je ne-lokalna pozornost čija je zadaća pronalazak udaljenih zavisnosti na slici. Zbog ograničenih receptivnih polja modeli trebaju pomoć s usvajanjem takvih zavisnosti. Mreža RNAN *(eng. residual non-local attention networks)* dobiva dvije dodatne grane, lokalna grana koristi enkoder-dekoder strukturu za učenje lokalne pozornosti, a ne-lokalna grana procjenjuje uparene veze između značajki svih pozicija slike. Također, pozornost se ugrađuje i za analizu kanala slike. Predloženi "squeeze-and-excitation" blok eksplicitno modelira međuovisnosti kanala [24]. Svi kanali ulaza se sažimaju u opisnike koji se šalju u dva gusto povezana sloja u kojima se hvataju zavisnosti kanala. Opisnici se mapiraju u težine koje odgovaraju međuovisnostima kanala.

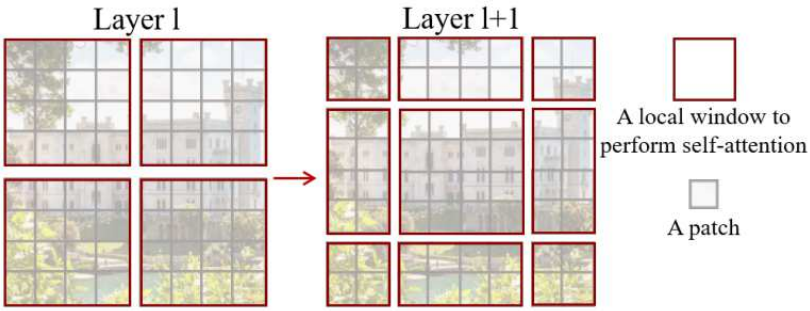
Primjer mreže koja koristi pozornost nad kanalima jest RCAN *(eng. residual channel attention network)*, čiji je cilj fokusiranje modela na informativnije kanale. Model se sastoji od rezidualnih blokova unutar rezidualnog bloka. Unutarnjih rezidualnih blokova je 10, a svaki taj blok sastoji se od 20 rezidualnih blokova pozornosti za kanale. Blokovi pozornosti sastoje se od sloja sažimanja, konvolucijskih slojeva, ReLU sloja i sigmoidne funkcije. Na kraju se izlaz sigmoide množi s ulazom u blok kako bi se primijenile dobivene težine [25].

2.7.9. Transformer arhitekture

Transformer arhitekture novija su pojava u području dubokog učenja, a specijalizirane su za učenje konteksta i razumijevanje sekvencijalnih podataka. Jezgreni dio transformer arhitektura jest mehanizam pozornosti pomoću kojeg se dobiva uvid u utjecaj i zavisnost udaljenih podataka. Transformeri crpe inspiraciju iz enkoder-dekoder arhitektura koje su se pojavile s rekurzivnim mrežama, ali za razliku od njih ne procesuira podatke u sekvencijalnom poretku, već koristi mehanizam samo-pozornosti koji računa ocjene za svaki element ulaza u usporedbi s ostalim elementima te tako uočava zavisnosti i veze između njih. Velika većina transformera koristi mehanizam pozornosti s više "glava" čiji se izlazi računaju paralelno i na kraju se linearno transformiraju u očekivanu dimenziju. Modeli nemaju rekurzivne veze niti koriste konvolucijske slojeve, ali koriste prednosti rezidualnih veza. Isto tako nemaju uvid u prostornu raspodjelu elemenata u podacima pa na ulazu primaju i enkodirane pozicije koje se sumiraju s ulaznom sekvencom.

Enkoder transformera sastoji se od nekoliko enkoder blokova koji imaju modul samo-pozornosti i potpuno povezani sloj te se između njih nalaze rezidualne veze i obavlja se operacija normalizacije. Dekoder je slične građe kao i enkoder te se sastoji od dekoder blokova koji u sebi isto imaju module za pozornost i potpuno povezane slojeve. Originalno predložena arhitektura transformera bila je namijenjena obradi teksta, no kasnije su razvijene posebne arhitekture koje se suočavaju s vizualnim zadacima.

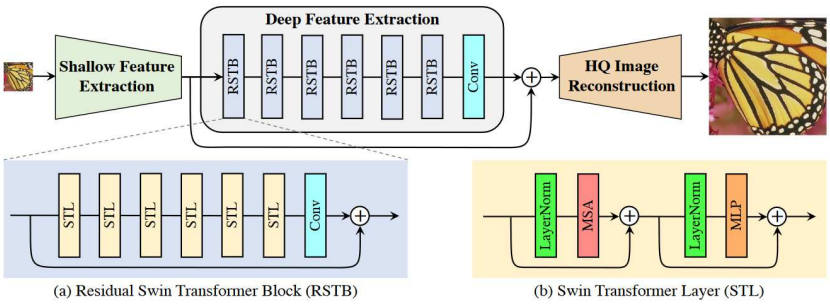
Vizualni transformer *(eng. Vision transformer, ViT)* prvi je modificirao izvornu arhitekturu transformera za rad sa slikovnim podacima, no on ne uključuje dekoder što je dosta česta pojava kod kasnije razvijenih modela. Prvi korak ViT arhitekture je tokenizacija slike, tj. pretvaranje slike u sekvenciju manjih elemenata, segmenata slike jednakih dimenzija. Segmenti slike se izravnavaju i mijenja im se duljina linearnom projekcijom. Segmenti se zatim procesiraju tako što im se pridružuje prazan token u koji će se upisivati izlaz i vektori pozicije. U enkodirajućem bloku model uči iz pripremljenih segmenata slike prolaskom kroz module pozornosti i normalizacije te se na kraju bloka nalazi modul neuronske mreže koji se sastoji od dva potpuno povezana sloja između kojih je aktivacijski sloj. Kako je ViT napravljen u svrhu klasifikacije slika, nakon enkoder blokova nalazi se komponenta za predikciju [26].



Slika 2.13: *Swin* pomični prozori [27]

Swin transformer je tip vizualnog transformera koji gradi hijerarhijske mape značajki i može se koristiti kao okosnica modela za super rezoluciju. *Swin* transformer blokovi zamjenjuju modul pozornosti s više glava s modulom koji računa samo-pozornost u lokalnim nepreklapajućim prozorima. Ti prozori nisu klizeći, ali čine pomake kroz slojeve kako bi se uvele veze između izrezaka slike (2.13). Veličina početnih segmenata slike koji se tokeniziraju jest 4×4, a kako se ide u dubinu mreže, broj tokena se smanjuje korištenjem slojeva za spajanje segmenata. Tako se izlučuju značajke na globalnoj i lokalnoj razini i grade se izlazne mape [27].

SwinIR model je za super rezoluciju koji koristi *Swin* transformer [28]. Sastoji se od plitkog izvlačenja značajki s konvolucijskim slojem, dubokog izvlačenja značajki sa *Swin* transformerom i rekonstrukcijskog subpikselnog sloja koji agregira plitke i duboke značajke i povećava rezoluciju. Njegova arhitektura dana je na slici 2.14.



Slika 2.14: Arhitektura *SwinIR* [28]

Transformer arhitekture trenutačno su prema $PSNRu$ najuspješnije u obavljanju zadatka super rezolucije te je najnapredniji model DRCT-L *(eng. dense residual connected transformer)* koji ima vrlo sličnu arhitekturu kao *SwinIR*, ali uvodi guste veze između rezidualnih blokova koje stabiliziraju tok podataka i izbjegavaju gubitak infor-

macija [29]. Najveća prepreka transformer arhitektura jest potreba za velikom količinom podataka za učenje zbog čega je započelo intenzivno istraživanje metoda nenadziranog i polunadziranog učenja.

2.8. Metode niskog nadzora

Većina popularnih modela za super rezoluciju koristi nadzirano učenje u kojem mreža uči iz parova HR i LR slika. Takvi pristupi postigli su značajan uspjeh i donijeli odlične rezultate, ali svi zahtijevaju veliku količinu podataka za učenje što predstavlja problem. Većina skupova podataka dobiva parove HR i LR slika koristeći degradacijske metode koje imaju unaprijed poznate parametre i postupke te se postavlja pitanje uče li modeli super rezolucije prirodno povećati rezoluciju ili samo obrnuti proces degradacije. Skupljanje izvornih parova slika iste scene, ali različite rezolucije zamoran je i teško izvediv postupak zbog kojeg je potrebno istraživati metode niskog nadzora.

2.8.1. Polunadzirane metode

Polunadzirane metode koriste oblik učenja gdje većina primjera iz skupa podataka nema izvornu HR sliku, dakle model ima pristup velikoj količini nestrukturiranih podataka, ali samo mali dio može iskoristiti za klasično učenje, a za ostatak minimizira razliku između sličnih podataka. Primjer ovakvog pristupa dan je kao TCR *(eng. transformation consistent regularization)*, regularizacijska metoda koja osigurava da niz geometrijskih transformacija LR slike ostane konzistentan s geometrijskim transformacijama SR slike [30]. Ovakav pristup rezultira pojavom da rotirane slike imaju identičan izlaz nakon super rezolucije [7]. Drugi pristupi koriste suparničke mreže s posebnim kombinacijama funkcija gubitka koje zasebno iskorištavaju označeni i neoznačeni primjeri. Neoznačeni primjeri često koriste vrste gubitka konzistencije koji zahtijeva određene sličnosti između LR i SR slika . Također postoje varijante sustava koji iz označenih parova uče degradaciju slika koju koriste za izgradnju vlastitog skupa podataka, ali takvi sustavi više pripadaju skupu nadziranih metoda.

2.8.2. *Few-shot* metode

Few-shot metode spadaju u posebnu vrstu algoritama kojima je cilj naučiti model kako da uči. Ovaj pristup koristi prije naučene modele za nove zadatke te nadogradnjom na već naučeni model, proširuju ga za rad s novom vrstom podataka koristeći

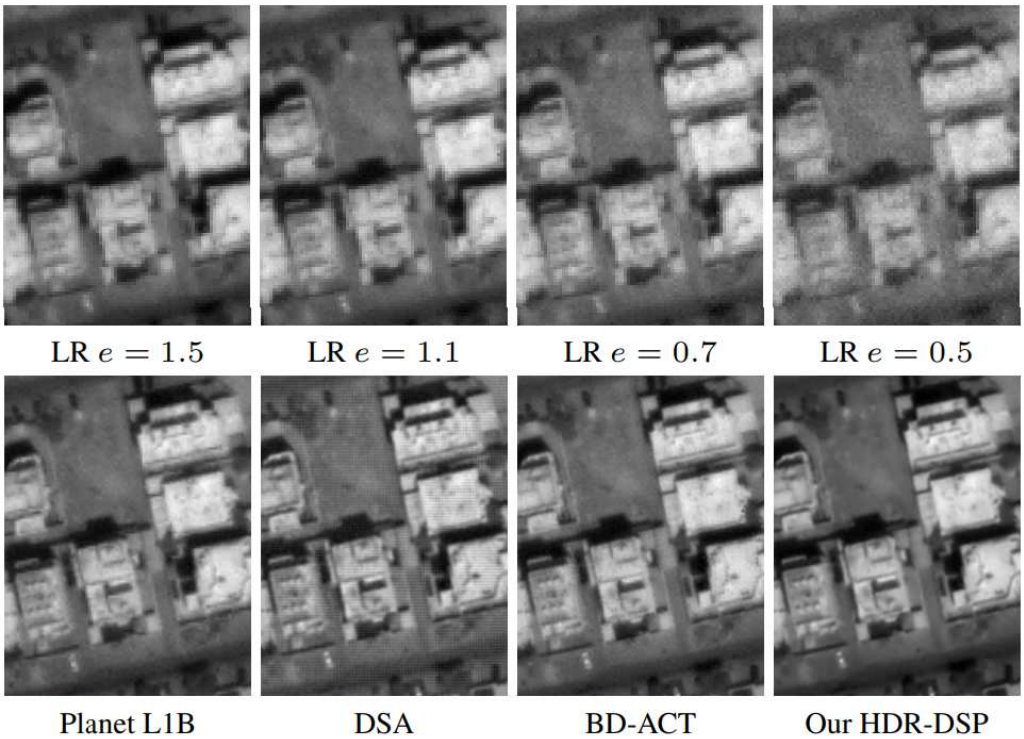
nekolicinu označenih podataka [7]. Korištenjem grafičkog modeliranja moguće je pripremiti mrežu za obavljanje super rezolucije. DRN *(eng. distortion relation network)* primjer je mreže koja je naučena raspoznavati veze između vrsta iskrivljenja slike poput šuma, artefakata, zamućenja te je stvorena banka predznanja za pohranu naučenih relacija iz pomoćnih zadataka koji su sintetički obavljali iskrivljenja. Kada se na ulazu mreže pojavi slika sa stvarnim iskrivljenjem ona može pretražiti svoju memoriju i pronaći odgovarajuće primjere. Na kraju se mjeri podudaranje iskrivljenja kosinusnom sličnosti te se odvojenom mrežom vođenom saznanjima DRNa gradi završni izlaz [31].

2.8.3. Nenadzirane metode

Nenadzirane metode učenja imaju isključivo neoznačene podatke, dakle samo LR slike, što predstavlja izrazito veliku prepreku za učenje super rezolucije. Jedna od metoda koja savladava ovu prepreku jest ciklus-u-ciklusu [7]. Sustav se sastoji od dva jednaka modela koja ukupno sadrže 4 generatora i 2 diskriminatora. Uloga jednog modela jest uklanjanje šuma na LR slici, a drugog povećanje rezolucije očišćene LR slike. Spajanjem dva povezana zadatka dobiva se pojačanje u učinkovitosti modela te se dodatno koristi i posebna kombinacija suparničkog gubitka, gubitka konzistencije i identiteta za održavanje boja.

2.8.4. Samonadzirane metode

Metode koje samostalno stvaraju vlastite nadzorne mehanizme nazivaju se samonadzirane metode. One se mogu svrstati i u skup nenadziranog učenja zbog toga što isto ne koriste označene podatke (podatke bez zadanog željenog izlaza), ali za njih je specifično stvaranje nadzornih elemenata iz čiste informacije ulaza. Ovakav pristup je moguć kod sustava koji primaju više slika na ulazu pa iskorištavaju jednu od njih kao referentnu vrijednost. Primjer ovakvog sustava jest HDR-DSP model učen da obavlja super rezoluciju na satelitskim snimkama kod kojih je dobivanje više verzija i različitih kuteva iste slike relativno dostupno [32]. Model radi posmak i sažimanje svih mapa značajki u jednu HR mapu te koristi mjere poput maksimalne i standardne devijacije piksela koje su invarijantne na permutacije (zbog više ulaznih slika) te postiže dosta bolje rezultate nego prijašnje metode na istom skupu podataka 2.15.

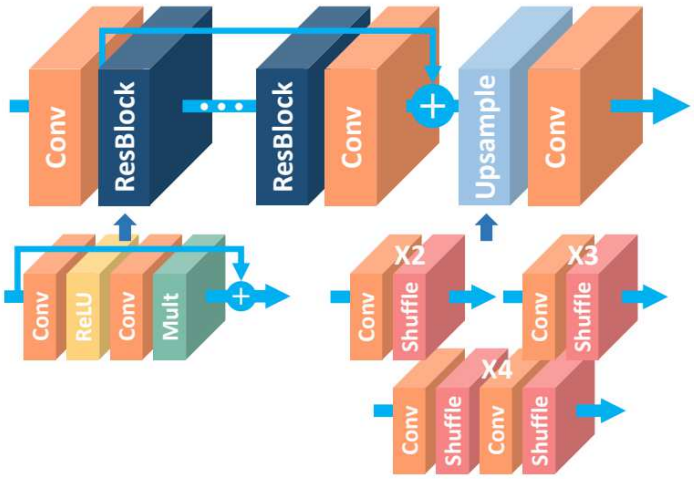


Slika 2.15: Rezultati HDR-DSP modela [32]

3. Implementacija

3.1. Model

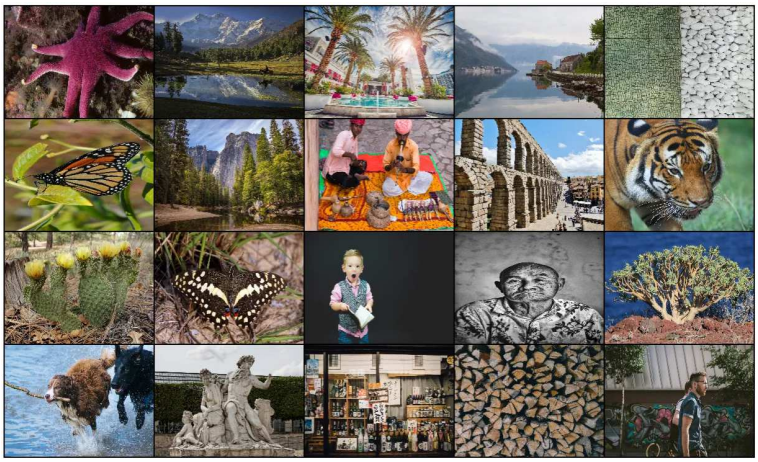
Odabrana arhitektura mreže za implementaciju sustava za povećanje razlučivosti je osnovni model EDSRa, pobjednik *NTIRE* 2017 natjecanja u super rezoluciji. Osnovna verzija modela sastoji se od 16 rezidualnih blokova koji su prikazani na slici 2.10, a sastoje se od 2 konvolucijska sloja, ReLU aktivacije između njih i pridodavanja identiteta. *BatchNorm* slojevi su izbačeni zato što se pokazalo da negativno utječu na performanse i učinkovitost modela, ali se na ulazu i izlazu obavlja normalizacija i denormalizacija RGB kanala. Svi konvolucijski slojevi osim zadnjeg sadrže 64 jezgre, a dodatan sloj se nalazi prije i poslije početka rezidualnih blokova te se između njih proteže i rezidualna veza. Operaciju naduzorkovanja obavlja subpikselni sloj koji koristi konvolucijske slojeve za povećanje broja kanala prije samog postupka miješanja piksela. Nakon naduzorkovanja ulaz prolazi kroz još jedan konvolucijski sloj s 3 jezgre zbog pretpostavke da izlaz ima 3 kanala (npr. RGB ili YCbCr). Sažeti prikaz arhitekture dan je na slici 3.1.



Slika 3.1: Arhitektura EDSR [15]

3.2. Skup podataka

Korišten je skup podataka *DIV2K* za učenje, a testiranje je provedeno na *Set14* skupu. Skup sadrži 800 primjera za učenje i 100 za validaciju, ali se validacija nije provodila temeljito svakih nekoliko epoha, već ručno koristeći kontrolne točke. Primjeri za učenje skalirani su dolje 4 puta kako bi se dobila ulazna LR slika. Od augmentacija nad podacima su se koristili samo klasični postupci koji ne uvode poremećaje uzoraka unutar slike, a to su izrezivanje, rotacije i zrcaljenje. Za učitavanje skupa podataka korišten je preuzeti učitavač podataka koji obavlja konverziju i skladištenje slika u priručnu memoriju za ubrzavanje procesa [33].



Slika 3.2: Primjeri slika iz *DIV2K* [34]

3.3. Proces učenja

U procesu učenja korištene su dvije funkcije gubitka. Model je prvo učen isključivo na pikselnom gubitku, konkretnije korištena je $L1$ verzija sa srednjom apsolutnom pogreškom zato što ponekad postiže bolje rezultate od $L2$ verzije koja je fokusirana na optimizaciju $PSNR$ metrike. Glavni nedostatak pikselnog gubitka jest što ignorira ljudsku percepciju i ne postiže oku ugodne i kvalitetne rezultate, stoga se obavlja dodatno detaljno podešavanje korištenjem perceptivnog gubitka po uzoru na SRGAN. Perceptivni gubitak kombinacija je gubitka sadržaja i suparničkog gubitka generatora (binarna unakrsna entropija), a njegova uloga jest izbjeći mutne teksture nastale kao posljedica korištenja pikselnog gubitka. Za ubrzanje konvergencije koristio se optimizator *Adam*, a detalji procesa učenja preuzeti su iz EDSR rada [15], no zbog vremenskih ograničenja platforme u oblaku koja se koristila za učenje, iznosi nekih parametara su smanjeni. Model je prvo učio u 100 epoha po 1000 koraka (ovdje epoha nema

klasično značenje jednog prolaza kroz cijeli skup podataka), a veličina serije podataka *(eng. batch size)* nakon kojih se obavlja ažuriranje težina je 4. Nakon toga, model uči koristeći perceptivni gubitak u još 100000 koraka, te se koriste VGG mreža za dobivanje mapa značajki i SRGAN diskriminator preuzeti iz [33]. Tijekom postupka su se koristili svi RGB kanali slika te su početne težine nasumično postavljene.

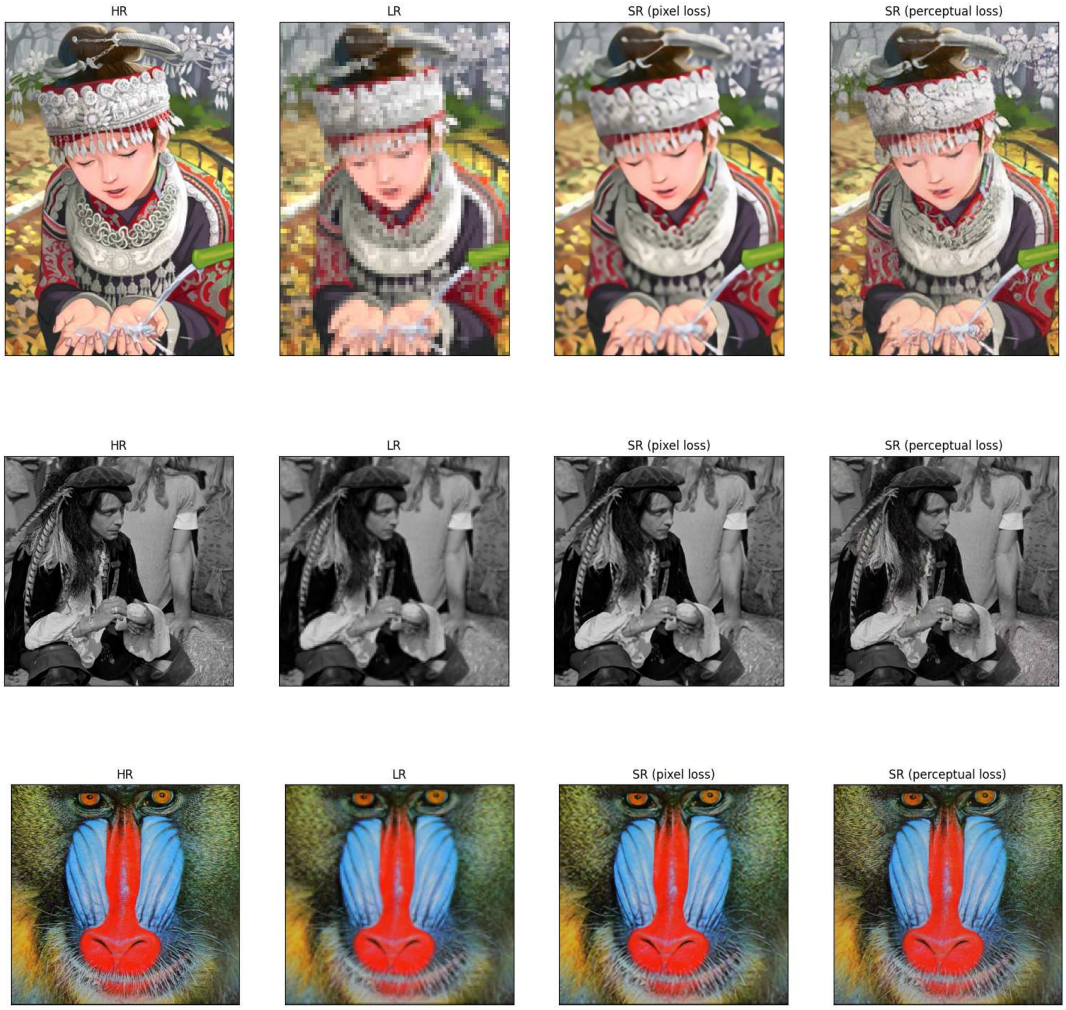
3.4. Rezultati

Zbog skraćenog vremena učenja i kretanja od nasumičnih težina model ne postiže jednake rezultate kao one navedene u EDSR radu [15]. Dobivena srednja $PSNR$ vrijednost na *Set14* skupu za model treniran samo s pikselnim gubitkom jest 26.58, a srednja $SSIM$ vrijednost 0.7513. Naime srednja $PSNR$ vrijednost u radu na *Set14* skupu navedena je kao 28.80, a srednja $SSIM$ vrijednost 0.7876, što je dosta velika razlika koja dokazuje važnost provođenja temeljitog učenja. Vrijednosti $PSNR$ i $SSIM$ dodatno opadaju nakon provođenja učenja s perceptivnim gubitkom, ali to je očekivano pa su tako srednje vrijednosti $PSNR$ i $SSIM$ 23.81 i 0.6405. Moguće je da je do velike razlike došlo i zbog različitog načina računanja metrika zato što se one ponekad računaju samo na svjetlosnom kanalu slike i različito se računa srednja vrijednost.

Set14	PSNR	SSIM
EDSR pixel	26.58	0.7513
EDSR pixel + perceptual	23.81	0.6405

Tablica 3.1: Tablica rezultata

Vizualni rezultati su dosta zadovoljavajući te je očita razlika u primjeni različitih funkcija gubitka. Rezultati samo pikselnog gubitka imaju zamućene i nejasne teksture i neobično naglašene rubove. Uspješnost perceptivnog gubitka posebno se vidi u dijelovima slika sa složenim teksturama poput krzna ili kose. Pikselni gubitak ne može samostalno ostvariti realistične slike, ali se i dalje često koristi zbog unaprjeđenja objektivnih metrika, što može naštetiti modelu. Primjeri rezultata oba modela, zajedno s izvornim LR i HR slikama dani su na prikazu 3.3.



Slika 3.3: Primjer rezultata na *Set14* skupu

3.5. Korišteni alati i kod

3.5.1. Alati

EDSR model je implementiran u programskoj jeziku *Python*, a koristili su se paketi *TensorFlow* i *Keras* koji pružaju radni okvir za stvaranje rješenja koja koriste strojno učenje i omogućava uporabu GPU za optimiziranje procesa s tenzorima. Paketi sadrže funkcije koje grade slojeve mreže, automatiziraju proces učenja te računaju gubitak i evaluacijske metrike. Za uporabu paketa nužno je koristiti CUDA i cuDNN alate koji osposobljavaju grafičke procesorske jedinice za rad s dubokim neuronskim mrežama. Zbog nedostatka računalnih resursa proces učenja se djelomično obavljao na platformi u oblaku *Google Colab*.

3.5.2. Kod

Kod modela nalazi se u direktoriju "model" u datoteci "edsr.py" i sastoji se od tri funkcije prikazane na slici 3.4. Funkcija $edsr$ stvara objekt modela definiran u *TensorFlow* paketu, u njoj se obavlja postavljanje zadanih slojeva koji čine mrežu, argument $scale$ govori modelu na kojem faktoru skaliranja radi, argument B označava broj rezidualnih blokova, a argument F broj jezgri u konvolucijskim slojevima. Funkcija res_block definira strukturu rezidualnog bloka, a $upsample$ sloj naduzorkovanja koji koristi miješanje piksela implementirano pomoću *TensorFlow* funkcije $depth_to_space$.

```
6   def edsr(scale, F=64, B=8):
7       input = Input(shape=(None, None, 3))
8
9       x = normalize(input)
10
11      x = b = Conv2D(F, 3, padding='same')(x)
12      for i in range(B):
13          b = res_block(b, F)
14      b = Conv2D(F, 3, padding='same')(b)
15      x = Add()([x, b])
16      if scale == 2:
17          x = upsample(x, scale, F)
18      elif scale == 3:
19          x = upsample(x, scale, F)
20      elif scale == 4:
21          x = upsample(x, 2, F)
22          x = upsample(x, 2, F)
23
24      x = Conv2D(3, 3, padding='same')(x)
25
26      x = denormalize(x)
27
28      return Model(input, x)
29
30
31  def res_block(input, F):
32      x = Conv2D(F, 3, padding='same', activation='relu')(input)
33      x = Conv2D(F, 3, padding='same')(x)
34      x = Add()([input, x])
35      return x
36
37
38  def upsample(x, scale, F):
39      x = Conv2D(F * (scale ** 2), 3, padding='same')(x)
40      return pixel_shuffle(x, scale)
41
```

Slika 3.4: edsr.py

Učenje je jednostavno implementirano u "train.py" pomoću funkcije fit iz $Model$ klase. Klasa omogućava samostalno postavljanje funkcije gubitka, optimizatora i parametara učenja. Model je potrebno kompilirati pomoću funkcije $compile$ prije učenja kako bi bile poznate sve pozicije težina koje se mogu ažurirati.

```
23  model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
24      filepath=checkpoint_filepath,
25      monitor='val_loss',
26      mode='min',
27      save_best_only=True)
28
29  model.fit(train_dataset, epochs=100, steps_per_epoch=1000, callbacks=[model_checkpoint_callback])
30
31  model.save_weights(os.path.join(weights_dir, 'weights-edsr.h5'))
```

Slika 3.5: train.py

Kod za učenje s perceptivnim gubitkom koji koristi diskriminator i generator preuzet je iz [33] i nalazi se u "trainPerceptual.py". Na slici 3.6 prikazan je korak učenja, lr je ulazna LR slika, a hr je referentna HR slika te se na početku osigurava konzistentnost brojeva pretvorbom u decimalne brojeve $float32$. Izlaznu SR sliku stvara $generator$, a $discriminator$ daje procjene diskriminatora pomoću kojih se računa suparnički gubitak sadržan u gen_loss i $disc_loss$. Gubitak sadržaja con_loss računa se pomoću mreže VGG te se kombinira s gubitkom generatora gen_loss pomnoženim konstantom u perceptivni gubitak $perc_loss$. Gradijenti pomoću kojih se ažuriraju težine računaju se odvojeno pomoću posebne *TensorFlow* strukture $GradientTape$ koja prepoznaje promjenjive varijable modela i odmah se može primijeniti pomoću funkcije $apply_gradients$.

```
98   @tf.function
99   def train_step(lr, hr):
100      with tf.GradientTape() as gen_tape, tf.GradientTape() as disc_tape:
101          lr = tf.cast(lr, tf.float32)
102          hr = tf.cast(hr, tf.float32)
103
104          sr = generator(lr, training=True)
105          hr_output = discriminator(hr, training=True)
106          sr_output = discriminator(sr, training=True)
107
108          con_loss = content_loss(hr, sr)
109          gen_loss = generator_loss(sr_output)
110          perc_loss = con_loss + 0.001 * gen_loss
111          disc_loss = discriminator_loss(hr_output, sr_output)
112
113      gradients_of_generator = gen_tape.gradient(perc_loss, generator.trainable_variables)
114      gradients_of_discriminator = disc_tape.gradient(disc_loss, discriminator.trainable_variables)
115
116      generator_optimizer.apply_gradients(zip(gradients_of_generator, generator.trainable_variables))
117      discriminator_optimizer.apply_gradients(zip(gradients_of_discriminator, discriminator.trainable_variables))
118
119      return perc_loss, disc_loss
```

Slika 3.6: trainPerceptual.py

4. Zaključak

Suvremene implementacije sustava za povećanje rezolucije primjenjuju tehnike dubokog učenja i postižu kvalitetne rezultate. Raznovrsni pristupi i arhitekture iskazuju prednosti u različitim sferama super rezolucije te je potrebno dobro definirati problem kako bi se odabrala pravilna struktura i podaci za izgradnju rješenja. Prema objektivnim metrikama poput $PSNRa$ transformer arhitekture postižu najbolje rezultate, ali za ovaj zadatak je vrlo bitna subjektivna mjera i ljudska percepcija prema kojoj se GAN arhitekture posebno ističu realističnim rezultatima. Kombiniranjem različitih funkcija gubitka, poput perceptivnog i pikselnog, može se postići dobra ravnoteža između objektivno preciznih i vizualno ugodnih rezultata. Najveća prepreka današnjih algoritama strojnog učenja jest potreba za velikom količinom podataka i opsežnost istih. Kako bi modeli mogli bolje generalizirati i obavljati zadatke na svim vrstama ulaza važno je nastaviti istraživati metode učenja niskog nadzora i oblikovati mehanizme prenašanja znanja. Posebno je zanimljiv i novouočeni problem stvaranja izvornih parova HR i LR slika koji se pokušava riješiti specijalnim ulančavanjem kamera ili ublažiti istraživanjem različitih degradacijskih funkcija. Također je moguć napredak u samim modulima naduzorkovanja i osmišljavanje novih postupaka temeljenih na učenju za proširenje dimenzija zato što je izbor opcija trenutno ograničen. Trenutne evaluacijske metrike previše su fokusirane na vrijednost samih piksela te potiču modele na korištenje pikselnog gubitka koji samostalno ne proizvodi vizualno najugodnije rezultate. Razvijanjem evaluacijskih metrika koje prosuđuju harmoniju slika i funkcije gubitka koje nagrađuju realističnost slika smjer istraživanja bi se pomaknuo u pravom smjeru prema društveno korisnim modelima. Unatoč već dobrim rezultatima, jasno je da su daljnja poboljšanja u brojnim aspektima moguća, ali se može zaključiti da je odabir dubokih modela prikladan za rješavanje zadatka super rezolucije.

LITERATURA

[1] Z. Wang, J. Chen, i S. C. H. Hoi, "Deep learning for image super-resolution: A survey", https://arxiv.org/abs/1902.06068, 2020., pristup: 04. 05. 2024.

[2] M. Anderson. (2023) Temporally coherent stable diffusion videos via a video codec approach. https://blog.metaphysic.ai/temporally-coherent-stable-diffusion-videos-via-a-video-codec-approach//. Pristup: 03. 05. 2024.

[3] cloudfactory. (2024) Downsampling and upsampling in machine learning. https://wiki.cloudfactory.com/docs/mp-wiki/key-principles-of-computer-vision/upsampling-and-downsampling-techniques-in-machine-learning. Pristup: 03. 05. 2024.

[4] A. Anwar. (2020) What is transposed convolutional layer? https://towardsdatascience.com/what-is-transposed-convolutional-layer-40e5e6e31c11. Pristup: 08. 05. 2024.

[5] E. Gomede. (2024) nhancing image clarity with subpixel convolution: A cutting-edge approach to super-resolution. https://pub.aimind.so/enhancing-image-clarity-with-subpixel-convolution-a-cutting-edge-approach-to-super-resolution-f6c95d94b929. Pristup: 08. 05. 2024.

[6] X. Hu, H. Mu, X. Zhang, Z. Wang, T. Tan, i J. Sun. (2019) Meta-sr: A magnification-arbitrary network for super-resolution. https://arxiv.org/abs/1903.00875. Pristup: 08. 05. 2024.

[7] R. Kundu. (2022) Deep learning for image super-resolution [incl. architectures]. https://www.v7labs.com/blog/image-super-resolution-guide. Pristup: 03. 05. 2024.

[8] J. Yoo, N. Ahn, i K.-A. Sohn. (2020) Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy. https://arxiv.org/abs/2004.00448. Pristup: 25. 05. 2024.

[9] C. Dong, C. C. Loy, K. He, i X. Tang. (2015) Image super-resolution using deep convolutional networks. https://arxiv.org/abs/1501.00092. Pristup: 03. 05. 2024.

[10] MathWorks. (2024) Psnr. https://www.mathworks.com/help/vision/ref/psnr.html. Pristup: 10. 05. 2024.

[11] P. Datta. (2020) All about structural similarity index (ssim): Theory + code in pytorch. https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e. Pristup: 10. 05. 2024.

[12] A. C. N. Matcha. (2020) Image super-resolution: A comprehensive review. https://blog.paperspace.com/image-super-resolution/. Pristup: 03. 05. 2024.

[13] M. S. M. Sajjadi, B. Schölkopf, i M. Hirsch. (2017) Enhancenet: Single image super-resolution through automated texture synthesis. https://arxiv.org/abs/1612.07919. Pristup: 15. 05. 2024.

[14] A. I. (2023) Understanding resnet architecture: A deep dive into residual neural network. https://medium.com/@ibtedaazeem/understanding-resnet-architecture-a-deep-dive-into-residual-neural-network-2c792e6537a9. Pristup: 17. 05. 2024.

[15] B. Lim, S. Son, H. Kim, S. Nah, i K. M. Lee. (2017) Enhanced deep residual networks for single image super-resolution. https://arxiv.org/abs/1707.02921. Pristup: 03. 05. 2024.

[16] Y. Tai, J. Yang, i X. Liu. (2020) Image super-resolution via deep recursive residual network. https://openaccess.thecvf.com/content_cvpr_2017/papers/Tai_Image_Super-Resolution_via_CVPR_2017_paper.pdf. Pristup: 17. 05. 2024.

[17] J. Brownlee. (2021) Gentle introduction to ensemble learning algorithms. https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/. Pristup: 08. 05. 2024.

[18] A. I. Aramendia. (2024) Densenet : A complete guide. https://medium.com/@alejandro.itoaramendia/densenet-a-complete-guide-84fedef21dcc. Pristup: 17. 05. 2024.

[19] S.-H. Tsang. (2019) Review: Srdensenet — densenet for sr (super resolution). https://towardsdatascience.com/review-srdensenet-densenet-for-sr-super-resolution-cbee599de7e8. Pristup: 17. 05. 2024.

[20] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, i W. Shi. (2017) Photo-realistic single image super-resolution using a generative adversarial network. https://arxiv.org/abs/1609.04802. Pristup: 18. 05. 2024.

[21] X. Wang, L. Xie, C. Dong, i Y. Shan. (2021) Real-esrgan: Training real-world blind super-resolution with pure synthetic data. https://arxiv.org/abs/2107.10833. Pristup: 08. 05. 2024.

[22] Z. Yue, J. Wang, i C. C. Loy. (2023) Resshift: Efficient diffusion model for image super-resolution by residual shifting. https://arxiv.org/abs/2307.12348. Pristup: 18. 05. 2024.

[23] O. T. Taiwo. (2024) What are attention mechanisms in deep learning? https://www.freecodecamp.org/news/what-are-attention-mechanisms-in-deep-learning/. Pristup: 17. 05. 2024.

[24] J. Hu, L. Shen, S. Albanie, G. Sun, i E. Wu. (2019) Squeeze-and-excitation networks. https://arxiv.org/abs/1709.01507. Pristup: 17. 05. 2024.

[25] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, i Y. Fu. (2018) Image super-resolution using very deep residual channel attention networks. https://arxiv.org/abs/1807.02758. Pristup: 17. 05. 2024.

[26] S. J. Callis. (2024) Vision transformers, explained. https://towardsdatascience.com/vision-transformers-explained-a9d07147e4c8#1d10. Pristup: 18. 05. 2024.

[27] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, i B. Guo. (2021) Swin transformer: Hierarchical vision transformer using shifted windows. https://arxiv.org/abs/2103.14030. Pristup: 18. 05. 2024.

[28] J. Liang, J. Cao, G. Sun, K. Zhang, L. V. Gool, i R. Timofte. (2021) Swinir: Image restoration using swin transformer. https://arxiv.org/abs/2108.10257. Pristup: 18. 05. 2024.

[29] C.-C. Hsu, C.-M. Lee, i Y.-S. Chou. (2024) Drct: Saving image super-resolution away from information bottleneck. https://arxiv.org/abs/2404.00722. Pristup: 18. 05. 2024.

[30] A. Mustafa i R. K. Mantiuk. (2020) Transformation consistency regularization-a semi-supervised paradigm for image-to-image translation. https://arxiv.org/abs/2007.07867. Pristup: 25. 05. 2024.

[31] X. Li, X. Jin, J. Fu, X. Yu, B. Tong, i Z. Chen. (2023) A close look at few-shot real image super-resolution from the distortion relation perspective. https://arxiv.org/abs/2111.13078. Pristup: 25. 05. 2024.

[32] N. L. Nguyen, J. Anger, A. Davy, P. Arias, i G. Facciolo. (2022) Self-supervised super-resolution for multi-exposure push-frame satellites. https://arxiv.org/abs/2205.02031. Pristup: 25. 05. 2024.

[33] M. Krasser. (2019) Single image super-resolution with deep neural networks. https://krasserm.github.io/2019/09/04/super-resolution/. Pristup: 04. 05. 2024.

[34] E. Agustsson i R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study", u *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

Sustav povećanja razlučivosti slike zasnovan na strojnom učenju

Sažetak

Super rezolucija je postupak dobivanja slike visoke rezolucije iz slike niske rezolucije. Duboko je učenje najpopularniji i najkvalitetniji pristup rješavanju problema te postoje raznolike tehnike kojima se može pristupiti zadatku. U odabiru arhitektura ističu se generativne suparničke mreže i transformeri, a za funkciju gubitka često se uzimaju pikselni, perceptivni i gubitak teksture. Za zadatak je ključno imati dobar i opsežan skup podataka te procijeniti uspješnost modela prikladnom evaluacijskom metrikom poput $PSNR$ ili $SSIM$ mjere. Pri izradi modela važno je odabrati dobru ravnotežu između stvaranja vizualno ugodnih i objektivno sličnih rezultata. Područje super rezolucije već postiže izvrsne rezultate, ali napredak u brojnim aspektima je i dalje moguć.

Ključne riječi: super rezolucija, duboko učenje, naduzorkovanje, generativni modeli

Image Super-Resolution System Based on Machine Learning

Abstract

Super resolution is the process of obtaining a high-resolution image from its low-resolution counterpart. Deep learning is the most popular and high-quality approach to solving this problem, and there are various techniques that can be applied to the task. In the selection of architectures, generative adversarial networks and transformers stand out, and for the loss function, pixel, perceptual, and texture loss are often used. For this task, it is crucial to have a good and extensive dataset and to assess the success of the model with an appropriate evaluation metric such as PSNR or SSIM measures. When creating the model, it is important to strike a good balance between generating visually pleasing and objectively similar results. The field of super resolution is already achieving excellent results, but progress in many aspects is still possible.

Keywords: super resolution, deep learning, upsampling, generative models