Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Gašper Spagnolo

Lokalizacija brezpilotnih letalnikov

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Luka Čehovin Zajc
Somentor: asist. Matej Dobrevski

Ljubljana, 2023

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Kandidat: Gašper Spagnolo
Naslov: Lokalizacija brezpilotnih letalnikov
Vrsta naloge: Diplomska naloga na univerzitetnem programu prve stopnje Računalništvo in informatika
Mentor: doc. dr. Luka Cehovin Zajc
Somentor: asist. Matej Dobrevski

Opis:
V zadnjem času postaja uporaba brezpilotnih letalnikov vse bolj razširjena in se uporablja na različnih področjih, kot so agrikultura, kartiranje, vojaške operacije idr. Kljub njihovi vsestranskosti pa se poraja ključno vprašanje: kako se brezpilotni letalniki obnašajo, ko izgubijo stik z sistemom za določanje položaja? Diplomska naloga se osredotoča na to tematiko in predlaga metodo za lokalizacijo brezpilotnih letalnikov ob izgubi sistema za določanje položaja.

Title: UAV localization

Description:
In recent times, the use of unmanned aerial vehicles (UAVs) has become increasingly prevalent, finding applications in various fields such as agriculture, mapping, military operations, and many others. Despite their versatility, a critical question arises: how do drones behave when they lose connection to the positioning system? This thesis focuses on this issue and proposes a method for localizing UAVs in the event of a positioning system signal loss.

Zahvaljujem se mentorju, doc. dr. Luki Čehovinu Zajc, in somentorju, asist. Mateju Dobrevskemu, za strokovno pomoč in usmeritve pri izdelavi diplomske naloge. Zahvala gre tudi družini za podporo pri pisanju diplomskega dela.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
UAV	unmanned aerial vehicle	brezpilotni letalnik
SAT	satellite	satelit
CNN	convolutional neural network	konvolucijska nevronska mreža
RDS	relative distance score	ocena relativne razdalje
MSE	mean squared error	srednja kvadratna napaka
FPI	finding point in an image	iskanje točke v sliki
WAMF	Weight-Adaptive Multi Feature fusion	Uteženo združevanje več značilk
PCPVT	Pyramid Vision Transformer with Conditional Positional encodings	Piramidni vision transformer s pogojnimi pozicijskimi kodiranji
HANN	Hanning loss function	Hanningova kriterijska funkcija
GWMSE	Gaussian Weighted Mean Squared Error	Gaussova utežena srednja kvadratna napaka
HWMSE	Hanning Weighted Mean Squared Error	Hanningova utežena srednja kvadratna napaka
CWMSE	Cross-Weighted Mean Squared Error	Križno utežena srednja kvadratna napaka

Povzetek

Naslov: Lokalizacija brezpilotnih letalnikov

Avtor: Gašper Spagnolo

Diplomsko delo predstavlja implementacijo trenutno vodilne metode za geo-lokalizacijo brezpilotnih letalnikov, ob izgubi sistema za določanje položaja. V okviru dela smo ustvarili novo podatkovno zbirko, ki vsebuje pare slik iz brezpilotnega letalnika in pripadajočih satelitskih posnetkov. Osredotočili smo se na uporabo naprednih nevronskih mrež, zlasti konvolucijskih mrež, transformerske arhitekture in njenih derivatov, kot sta Vision Transformer (ViT) in Piramidni vision transformer (PVT). Ključno vlogo je imela Siamska nevronska mreža za primerjavo vzorcev med obema vrstama slik. Metodologija je bila podprta z različnimi optimizacijskimi strategijami, vključno z uporabo stratificiranega vzorčenja, Hanningovega okna in regularizacijskih tehnik. Rezultati potrjujejo učinkovitost predlagane metode za natančno geolokalizacijo brezpilotnih letalnikov. Delo zaključujemo s poudarkom na ključnih ugotovitvah in potencialu razvite metode.

Ključne besede: Lokalizacija UAV, geo-lokalizacija, globoko učenje, transformer.

Abstract

Title: UAV localization

Author: Gašper Spagnolo

The thesis presents the implementation of the currently leading method for geolocation of unmanned aerial vehicles, in the event of a loss of the positioning system. As part of the work, we created a new database containing pairs of images from unmanned aerial vehicles and corresponding satellite images. We focused on the use of advanced neural networks, especially convolutional networks, transformer architecture, and its derivatives, such as Vision Transformer (ViT) and Pyramid Vision Transformer (PVT). The Siamese neural network played a crucial role in comparing samples between the two types of images. The methodology was supported by various optimization strategies, including the use of stratified sampling, Hanning window, and regularization techniques. The results confirm the effectiveness of the proposed method for accurate geolocation of unmanned aerial vehicles. We conclude the work by emphasizing the key findings and the potential of the developed method.

Keywords: UAV localization, geo-localization, deep learning, transformer.

Poglavje 1

Uvod

"Brezpilotni letalniki so postali nepogrešljivi v številnih sektorjih, vključno s kmetijskim nadzorom, reševalnimi operacijami in vojaškimi operacijami Kljub njihovi široki uporabi pa se soočajo z izzivi pri avtonomni navigaciji, še posebej v okoljih, kjer je le-ta omejen ali nezanesljiv. V idealnih razmerah brezpilotni letalniki za svojo navigacijo uporabljajo sisteme za določanje položaja, kot so GPS [1], GLONASS [2] in drugi podobni sistemi. Vendar lahko te signale motijo naravne in človeške ovire, kot so visoke stavbe, gorske formacije ali celo elektronske motnje. Izguba sistema za določanje položaja lahko postane kritična, še posebej v tistih trenutkih, ko je natančna lokacija letalnika ključna za njegovo nalogo, zato je iskanje alternativne metode za lokalizacijo brezpilotnih letalnikov nujno.

Zgodnje metode, kot so navedene v virih [8, 17, 16, 13], so se osredotočale predvsem na uporabo ročno izdelanih značilnosti. To pomeni, da so raziskovalci uporabljali specifične, predhodno definirane vzorce iz slik za določanje lokacije. Čeprav so te metode predstavljale pomemben začetek, so bile omejene v svoji natančnosti in prilagodljivosti.

S prihodom globokih konvolucijskih nevronskih mrež (CNN) in njihove dokazane sposobnosti v obdelavi vizualnih podatkov so raziskovalci začeli

[1] GPS: `https://en.wikipedia.org/wiki/Global_Positioning_System`
[2] GLONASS: `https://en.wikipedia.org/wiki/GLONASS`

avtomatsko pridobivati kompleksne in prilagodljive značilnosti neposredno iz podatkov med učenjem mreže. Raziskave v [24] so bile med prvimi, ki so se lotile tega področja z izvlečkom značilnosti za izziv geolokalizacije s pomočjo različnih pogledov, uporabljajoč vnaprej naučen CNN. Ugotovljeno je bilo, da visokonivojske plasti v CNN vsebujejo bogate semantične informacije, ki lahko pripomorejo k boljši geolokalizaciji. Nadaljnje raziskave v [15] so razširile ta koncept z natančnim prilagajanjem predhodno naučenih mrež, da bi zmanjšali razdaljo značilnosti med satelitskimi slikami in slikami iz brezpilotnega letalnika.

V [18] je bil predstavljen pristop z uporabo modificirane siamske mreže. Ta pristop uporablja kontrastno izgubo za optimizacijo parametrov mreže, kar omogoča boljše razlikovanje med podobnimi in različnimi lokacijami. V [14] so bile predstavljene metode, ki so optimizirale opise slik, da so postale odporne na masivne spremembe perspektive, kot je pogled iz zraka proti tlem ali obratno. V [25] so predstavljene inovacije, ki uporabljajo prostorske informacije za izboljšanje globalnega koraka agregacije pri izvlečku značilnosti. Z uporabo mehanizma prostorske pozornosti so še dodatno izboljšali natančnost geolokalizacije.

Tradicionalne metode prepoznavanja slik se v kontekstu lokalizacije brezpilotnih letalnikov zdijo kot obetavna alternativa [2] in [27], takšen pristop pa lahko ima težave. Vsaka posodobitev ali sprememba v osnovni nevronski mreži, ki se uporablja za prepoznavanje slik, zahteva ponovno obdelavo celotne slikovne baze. Slednje ne le da je časovno potratno, ampak tudi zviša stroške, saj morajo vse slike ponovno potekati skozi postopek predprocesiranja in razpoznavanja. Ko brezpilotni letalnik zajame sliko za primerjavo, mora biti ta slika primerjana z vsako sliko v bazi, da se ugotovi najboljše ujemanje. V praksi to pomeni, da ko imamo bazo sestavljeno iz milijonov slik, bo vsaka nova poizvedovalna slika potrebovala milijone primerjav, kar je precej časovno potratno in računsko intenzivno.

V kontekstu omejitev tradicionalnih metod prepoznavanja slik so raziskovalci razvili pristop, imenovan FPI (Finding Point with Image) [5]. FPI

sprejme dva vhodna podatka: sliko posneto z brezpilotnim letalnikom in pripadajočo satelitsko sliko. V kontekstu te satelitske slike je mesto, kjer je bila slika iz brezpilotnega letalnika posneta. Za obdelavo vsake slike se uporablja posebna nevronska mreža, kjer vsaka mreža obdeluje svoj nabor podatkov brez deljenja uteži z drugo. Ko sta sliki obdelani in njihove značilke izluščene, se med njima izvede operacija korelacije. Ta mera podobnosti se predstavi v obliki toplotne karte, ki prikazuje stopnjo ujemanja med sliko brezpilotnega letalnika in satelitsko sliko. Najvišja vrednost na toplotni karti natančno označuje mesto, kjer je brezpilotni letalnik posnel svojo sliko na večji satelitski sliki. Informacija se nato neposredno prevede v natančno lokalizacijo brezpilotnega letalnika na satelitski sliki.

Inovacije v znanstvenem raziskovanju pogosto vodijo do nadaljnjih metodoloških izboljšav. Nadgradnja metode FPI, znana kot WAMF-FPI, je dodatno izboljšala natančnost in učinkovitost lokalizacije brezpilotnih letalnikov [5]. Ta pristop je integriral koncepte iz območja sledenja objektov za potrebe lokalizacije ob soočanju z izzivi, ki jih predstavljajo razlike med slikami zajetimi z brezpilotnim letalnikom in satelitskimi slikami. Z uporabo dveh različnih uteži za izvleček značilnosti iz slik posnetih z brezpilotnim letalnikom in satelitskih slik, WAMF-FPI omogoča natančnejše in bolj zanesljivo ujemanje slik. Dodatna optimizacija je bila dosežena z vključitvijo WAMF modula in uporabo Hanningove kriterijske funkcije, ki sta povečala učinkovitost modela.

WAMF-FPI je evolucija osnovne metode FPI. Ključna prednost WAMF-FPI je njegova napredna piramidna struktura izluščenja značilk, ki omogoča bolj natančno in raznoliko analizo vhodnih podatkov. Z uporabo te piramidne strukture se značilke izluščijo na več različnih ravneh, nato pa se skalirajo in medsebojno primerjajo, kar pridobi bolj robusten in natančen sklop informacij. Poleg tega WAMF-FPI optimizira kompresijske zmogljivosti, kar pripomore k hitrejšemu in učinkovitejšemu procesiranju podatkov. Medtem ko je bila v osnovni FPI metodi končna velikost značilk stisnjena na 16-krat manjšo od izvorne satelitske slike, v WAMF-FPI ta kompresijski faktor znaša

samo štirikrat manjšo velikost. To omogoča WAMF-FPI-ju, da ohrani več informacij ter pridobi boljšo lokalizacijsko natančnost ob hkratnem zmanjšanju računske obremenitve.

Kljub številnim obstoječim zbirkam, kot so CVUSA [11], CVACT [10] in University-1652 [26], ki so namenjene za zgoraj omenjene tradicionalne metode prepoznavanja slik, večina ne zajema vseh realnih situacij s katerimi se srečuje brezpilotni letalnik. Zbirka CVUSA [11] je osredotočena predvsem na zgradbe. Zbirka University-1652 [26] uporablja posnetke univerz, vendar nima dovolj raznolikih posnetkov, saj je omejena le na univerzitetna okolja. Poleg tega so objekti v sredini slike. Podatkovna zbirka UL14, omenjena v [5], je edina s pogledom od zgoraj navzdol, vendar avtorji zbirke žal niso javno delili. Zato smo se odločili za ustvarjanje lastne zbirke, osredotočene na pogled iz brezpilotnega letalnika, z uporabo Google Earth Studia[3]. Naša zbirka obsega 11 evropskih mest. Glavni cilj izdelave te zbirke je bil zagotoviti raznolike podatke, ki bi služili kot robustna osnova za testiranje in validacijo pristopov. S tem smo nameravali zagotoviti, da naša implementacija lahko obravnava različne scenarije, ki jih morebiti sreča brezpilotni letalnik v realnem svetu. Cilj izdelave zbirke je zagotoviti raznolike podatke, ki bi lahko služili kot robustna osnova za testiranje in validacijo naše implementacije WAMF-FPI.

Cilj diplomske naloge je raziskati in implementirati metodo WAMF-FPI, predstavljeno v [22], saj je ta metoda trenutno prepoznana kot vodilna in najnaprednejša na področju geolokalizacije brezpilotnih letalnikov. Poleg tega smo želeli tudi ustvariti podatkovno zbirko, ki bo omogočala nadaljnje raziskave na tem področju.

Diplomska naloga je razdeljena na šest osnovnih poglavij. V Poglavju 1 so predstavljena temeljna izhodišča in namen raziskave. Poglavje 2 obsega metodologijo, kjer so podrobno opisane uporabljene tehnike, kot so konvolucijske nevronske mreže, transformerska arhitektura in različne oblike Vision Transformerja. Poglavje 3 obravnava podatkovno množico, ki vključuje slike

[3]Google Earth Studio: `https://www.google.com/earth/studio/`

brezpilotnih letalnikov, satelitske slike in oznake. V Poglavju 4 je opisana implementacija, s posebnim poudarkom na metodi WAMF-FPI. Poglavje 5 se osredotoča na eksperimentalno evalvacijo, kjer so predstavljeni rezultati različnih eksperimentov in analize. V Poglavju 6 so podane sklepne ugotovitve in zaključki naloge. Diplomsko delo zaključujemo z relevantno literaturo in dodatki, ki vključujejo primere izračuna RDS in primerjavo toplotnih kart.

Poglavje 2

Metodologija

V tem poglavju bomo predstavili osnovne komponente, ki jih uporabljamo v našem modelu. Začeli bomo s konvolucijskimi nevronskimi mrežami, ki so temeljni gradnik večine modelov za obdelavo slik in nudijo močno orodje za izluščenje značilnosti iz vizualnih podatkov. Nadaljevali bomo s predstavitvijo transformerske arhitekture, ki je revolucionirala področje obdelave naravnega jezika in se v zadnjem času vedno bolj uporablja tudi v računalniškem vidu. Podrobneje se bomo osredotočili na zgradbo transformerja in njegove ključne komponente. V nadaljevanju se bomo posvetili strukturi Vision Transformer (ViT) in razširjeni različici – Pyramid Vision Transformer (PVT). Posebno pozornost bomo posvetili prilagojeni različici PVT, imenovani PCPVT, saj njeni deskriptorji zagotavljajo prostorsko skladnost in natančno poravnavo. Zaključili bomo s siamskimi nevronskimi mrežami, ki predstavljajo ključno komponento v primerjavi vzorcev. Te mreže so še posebej pomembne, ko nameravamo primerjati dva ali več podobnih vzorcev in ugotoviti, ali med njimi obstajajo razlike.

Z vključitvijo vseh teh komponent in tehnik v naš model WAMF-FPI nameravamo razviti robusten in natančen sistem za lokalizacijo točk na slikah. V nadaljevanju poglavja bomo vsako od teh komponent podrobno raziskali, da bi bolje razumeli njihove lastnosti in kako prispevajo k celotnemu modelu.

2.1 Konvolucijske nevronske mreže

Konvolucijske nevronske mreže (ang. Convolutional Neural Networks – CNN) so metoda globokega učenja, specializirana za obdelavo vizualnih podatkov, zasnovana tako, da avtomatsko in adaptivno izvaja izvleček značilnosti iz slik.

2.1.1 Struktura in delovanje

Osnovni gradniki CNN obsegajo štiri glavne vrste plasti: konvolucijsko, aktivacijsko, združevalno (ang. pooling) in polno povezano.

1. **Konvolucijska plast**: vsak nevron v tej plasti je povezan le z majhnim območjem v prejšnji plasti, namesto da bi bil povezan z vsemi nevroni, kot je to v običajnih nevronskih mrežah. Ko se filter (ali jedro) premika preko slike, izvaja konvolucijsko operacijo:

$$(I * K)(x, y) = \sum_m \sum_n I(m, n) \cdot K(x - m, y - n). \tag{2.1}$$

2. **Aktivacijska funkcija**: po konvolucijski operaciji se uporabi aktivacijska funkcija za vsak izhod. Najpogosteje se uporablja funkcija ReLU:

$$\text{ReLU}(x) = \max(0, x). \tag{2.2}$$

3. **Združevalna plast**: po konvolucijski in aktivacijski operaciji sledi združevalna plast, ki zmanjšuje dimenzije slike z uporabo operacij, kot je "max pooling":

$$P(i, j) = \max_{m,n \in R} I(i + m, j + n). \tag{2.3}$$

4. **Polno povezane plasti**: delujejo kot klasične plasti v običajnih nevronskih mrežah. Vsak nevron je povezan z vsemi izhodi prejšnje plasti.

$$O_i = \sum_j W_{ij} \cdot I_j + b_i, \tag{2.4}$$

kjer je O_i izhod, W_{ij} matrika uteži, I_j vhod in b_i pristranskost (ang. bias).

2.2 Transformerska arhitektura

V tem podpoglavju bomo obravnavali razvoj in lastnosti transformerske arhitekture. Predstavljeno bo ozadje, vključno s prejšnjimi mehanizmi, kot so rekurentne nevronske mreže, ter podrobnosti o njihovi zgradbi in delovanju.

2.2.1 Predhodni mehanizmi

Preden so obstajali transformerji, so bile najpogostejše metode za obvladovanje zaporedij v jezikovnih modelih rekurentne nevronske mreže (ang. Recurrent Neural Networks – RNN) in njihove različice, kot so dolgi kratkotrajni spomini (ang. Long Short-Term Memory – LSTM) in obogatene RNN (ang. Gated Recurrent Units – GRU). Najpogostejša uporaba teh modelov v kontekstu strojnega prevajanja ali drugih nalog pretvarjanja zaporedja v zaporedje je bila uporaba strukture kodirnik-dekodirnik. V tej strukturi je bilo zaporedje vhodnih besed ali kodirano v latentni prostor z uporabo RNN (kodirnik), ta latentni vektor pa je bil nato uporabljen za generiranje zaporedja izhodnih besed ali žetonov z uporabo drugega RNN (dekodirnik). Problem s to strukturo je bil, da je bil latentni prostor omejen na velikost fiksne dolžine in je moral vsebovati vse informacije iz izvornega zaporedja, ki so potrebne za generiranje ciljnega zaporedja. To je omejevalo model pri obvladovanju dolgih zaporedij, saj je bilo težko ohraniti informacije iz zgodnjega dela zaporedja do konca. Da bi to težavo rešili, so raziskovalci vključili mehanizem pozornosti, ki je omogočil dekodirniku, da se osredotoči na različne dele izvornega zaporedja na različnih stopnjah generiranja ciljnega zaporedja. Slednje je bil velik napredek, ki je omogočil boljše obvladovanje dolgih zaporedij [1].

2.2.2 Razlaga RNN kodirnik-dekodirnik arhitekture

Definirajmo problem strojnega prevajanja kot iskanje najboljše ciljne sekvence $\vec{E} = (e_0, e_1, ..., e_m)$ glede na dane izvorne besede $\vec{F} = (f_0, f_1, ..., f_n)$. Ta problem lahko izrazimo kot optimizacijo pogojne verjetnosti $P(\vec{E}|\vec{F})$. Začnimo z opisom RNN-kodirnik-dekodirnik arhitekture. Imamo dva RNN

modela, kodirnik RNNenc in dekodirnik RNNdec. Kodirnik z zaporedjem vektorjev $\vec{F}$ proizvede skrito stanje h_n:

$$h_n = \text{RNNenc}(f_n, hn - 1). \tag{2.5}$$

Začetno stanje h_0 je pogosto postavljeno na nič ali se ga mreža nauči. Dekodirnik nato uporablja to skrito stanje, da generira ciljno zaporedje $\vec{E}$:

$$e_t = \text{RNNdec}(et - 1, h_{t-1}). \tag{2.6}$$

Opomba: pri učenju se za e_{t-1} pogosto uporablja dejanska vrednost iz ciljnega zaporedja (ne izhod modela), kar je znano kot "teacher forcing"[19]. Izvorna zaporedja besed $\vec{F}$ se tako vnašajo v kodirnik, ki generira skrita stanja za vsako besedo:

$$\vec{H} = \text{Kodirnik}(\vec{F}). \tag{2.7}$$

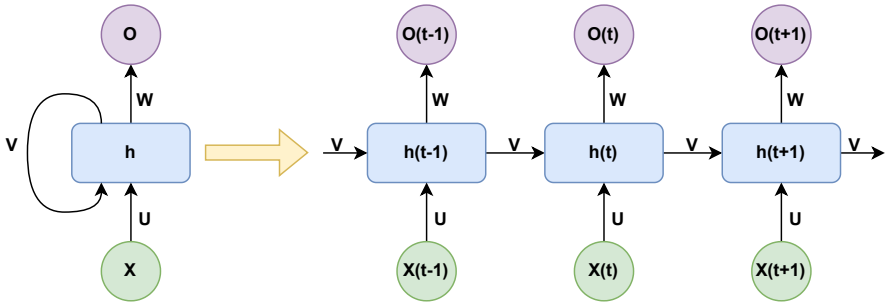
Za vsako besedo v ciljnem zaporedju $\vec{E}$ se potem izračuna utežena vsota skritih stanj iz kodirnika:

$$\vec{a}t = \text{Pozornost}(\vec{H}, et - 1). \tag{2.8}$$

Potem se ta vektor uporabi za napoved ciljne besede:

$$e_t = \text{Dekodirnik}(\vec{a}t, et - 1). \tag{2.9}$$

Ta pristop omogoča, da dekodirnik upošteva vse besede v izvornem zaporedju, ne samo prejšnje besede v ciljnem zaporedju, kar izboljša kakovost prevoda. Vendar je to zgolj matematična formulacija koncepta. Dejanski detajli, kot so vrste in struktura kodirnika in dekodirnika, so odvisni od specifičnega modela, ki ga uporabljamo. Na sliki 2.1 je prikazana skica RNN modela.



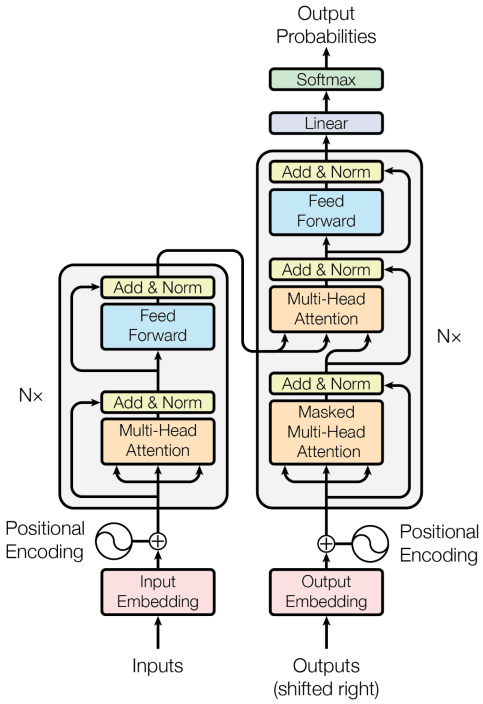
Slika 2.1: Skica RNN modela

2.3 Zgradba transformerja

Avtorji v članku [21] so predstavili novo arhitekturo za strojno prevajanje, ki se osredotoča na mehanizme pozornosti in se izogiba omejitvam RNN. Glavna inovacija je zamenjava RNN in njihovih skritih stanj z bolj učinkovitimi operacijami na osnovi pozornosti. Transformer model je model kodirnika-dekodirnika. Kodirnik sestavljajo N blokov na levi, dekodirnik pa N blokov na desni, vidno na sliki 2.2.

Med učenjem se vhodne besede $\vec{F} = (f_0, ..., f_n)$ hkrati prenesejo v prvi blok kodirnika, izhod tega bloka pa se nato prenese v njegovega naslednika. Postopek se ponavlja, dokler vseh N blokov kodirnika ni obdelalo vhoda. Vsak blok ima dve komponenti: plast večglave samopozornosti (ang. Multi-Head Self-Attention), ki ji sledi polno povezana plast z aktivacijami ReLU, ki obdeluje vsak element vhodne sekvence vzporedno. Tako večglav sloj pozornosti kot polno povezana plast sledita koraku *Dodaj in Normiraj* – *dodaj* se nanaša na residualno povezavo, ki doda vhod vsake plasti na izhod, *normiraj* pa se nanaša na normalizacijo plasti. Ko je vhod prešel skozi vse bloke kodiranja, ostane kodirana predstavitev $\vec{F}$.

Dekodirnik pa sestoji iz treh korakov: maske večglave samopozornosti, večglave plasti pozornosti, ki povezuje kodirano izvorno predstavitev z dekodirnikom, in polno povezane plasti z aktivacijami ReLU. Tako kot v kodirniku, vsaki plasti sledi plast *Dodaj in Normiraj*. Dekodirnik sprejme vse



Slika 2.2: Izgled transformerja, iz članka "Attention is all you need"[21].

ciljne besede $\vec{E} = (e_0, ..., e_m)$ kot vhod. V procesu napovedovanja besede e_i ima dekodirnik dostop do prej generiranih besed. Ne more pa imeti dostopa do besed, ki sledijo e_i, saj te še niso bile generirane. Obstaja nekaj ključnih razlik v primerjavi s kodirnikom - ena je, da so vhodi v prvo operacijo pozornosti v blokih dekodirnika maskirani, zato tudi ime plasti. To pomeni, da se lahko katera koli beseda v ciljnem izhodu nanaša samo na besede, ki so prišle pred njo. Razlog za to je preprost: med sklepanjem generiramo predvideni prevod $\vec{E}$ besedo za besedo z uporabo izvornega stavka $\vec{F}$.

Druga razlika od kodirnika je druga večglava plast pozornosti, ki se imenuje tudi plast pozornosti kodirnika-dekodirnika. Za razliko od plasti pozornosti na začetku blokov kodirnika in dekodirnika ta plast ni plast samopozornosti.

2.3.1 Utežena točkovna produktna pozornost

Utežena točkovna produktna pozornost (ang. Scaled Dot-Product Attention) se uporablja v vseh plasteh pozornosti v transformerju. Scaled Dot-Product Attention je skoraj identičen Dot-Product Attention-u, že omenjenem pri Luongu [1].

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.10}$$

Edina razlika je, da je vhod v softmax skaliran s faktorjem $\frac{1}{\sqrt{d_k}}$. Avtorji pozornosti omenjajo, da delijo vhode v softmax funkcijo z $\sqrt{(}d_k)$, da bi ublažili učinke velikih vhodnih vrednosti, ki bi vodile do majhnih gradientov med učenjem [21].

V članku [21] in predhodni literaturi [1] se vrstice $Q \in \mathbb{R}^{m \times d_k}$ imenujejo poizvedbe, vrstice $K \in \mathbb{R}^{n \times d_k}$ ključi, in vrstice $V \in \mathbb{R}^{n \times d_v}$ vrednosti. Upoštevati je potrebno, da se mora za izvedbo število ključev in vrednosti n ujemati, vendar se lahko število poizvedb m razlikuje. Prav tako se mora ujemati dimenzionalnost ključev in poizvedb, vendar se lahko dimenzionalnost vrednosti razlikuje.

Postopek izračuna utežene točkovne produktne pozornosti je naslednji:

1. Izračunamo produkt med matrikama poizvedb Q in ključev K^T.
2. Produkt normaliziramo z delitvijo z $\sqrt{d_k}$, kjer d_k predstavlja dimenzijo ključev.
3. Na dobljen rezultat uporabimo funkcijo *softmax*, da pridobimo matriko uteži pozornosti.
4. Matriko uteži pomnožimo z matriko vrednosti V, da pridobimo končni izhod.

Vektorji poizvedbe in ključev se med seboj primerjajo preko skalarnega produkta. Ta produkt nam pove, koliko pozornosti naj določen ključ nameni

določeni poizvedbi. Utežena vsota vektorskih vrednosti določa, koliko informacij iz vsakega ključa se upošteva v končnem izhodu. V tem postopku so uporabljene le matrične in vektorske operacije, brez dodatnih učljivih parametrov.

2.3.2 Večglava pozornost

Večglava pozornost, ključna komponenta v arhitekturi transformatorja, je razširitev mehanizma Scaled Dot-Product Attention, omenjenega v prejšnjem podpoglavju. V večglavi pozornosti se vhodni podatki (poizvedbe, ključi in vrednosti) najprej transformirajo v več različnih prostorov z uporabo linearnih preslikav. Nato se za vsak niz izračuna funkcija pozornosti Scaled Dot-Product Attention. Rezultati teh funkcij pozornosti se nato združijo skupaj v eno matriko. Končno se ta matrika preslika nazaj v izviren prostor z uporabo druge linearne preslikave, da se pridobi končni rezultat večglave pozornosti. Avtorji to izrazijo v spodnji obliki [21]:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W_O. \tag{2.11}$$

Vsak head_i je rezultat izvajanja Scaled Dot-Product Attention na i-tem nizu transformiranih poizvedb, ključev in vrednosti:

$$\text{head}_i = \text{Attention}(QW_{Qi}, KW_{Ki}, VW_{Vi}), \tag{2.12}$$

kjer so $Q \in \mathbb{R}^{m \times d_{\text{model}}}$, $K \in \mathbb{R}^{n \times d_{\text{model}}}$, in $V \in \mathbb{R}^{n \times d_{\text{model}}}$. Poleg tega, ob upoštevanju hiperparametra h, ki označuje število glav pozornosti, velja: $W_{Qi} \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_{Ki} \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_{Vi} \in \mathbb{R}^{d_{\text{model}} \times d_v}$, in $W_O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

Vsak izračun glave ima drugačno linearno preslikavo za matrike ključev, poizvedb in vrednosti. Vsaka od teh preslikav se nauči med učenjem.

2.3.3 Pozornost kodirnik-dekodirnik

Tretja in zadnja uporaba pozornosti v članku [21] je pozornost kodirnik-dekodirnik, ki se uporablja v blokih dekodirnika neposredno po sloju maske

večglave pozornosti, da se povežejo izvorne in ciljne sekvence. Medtem ko so pri samopozornosti vsi trije vhodi enaka matrika, to tukaj ne velja.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W_O, \quad \text{head}_i = f(Q, K, V)$$

Ko govorimo o pozornosti med kodirnikom in dekodirnikom, je edina razlika od prej v tem, da Q izhaja iz sloja maske večglave pozornosti, medtem ko sta K in V kodirani predstavitvi $\vec{F}$. Lahko bi razmišljali o tem tako, da model zastavlja vprašanje o tem, kako se vsak položaj v ciljni sekvenci nanaša na izvor, in pridobiva predstavitve izvora za uporabo pri generiranju naslednje besede v cilju. Pomembno je poudariti, da vsi bloki dekodirnika prejmejo enake podatke od kodirnika. Od prvega do N-tega bloka dekodirnika vsak uporablja kodirano izvorno sekvenco kot ključe in vrednosti.

2.4 Vision Transformer (ViT)

Transformerji so prvotno bili omejeni na obdelavo zaporedij, kar je idealno za jezik, vendar ne nujno za slike, ki so običajno dvodimenzionalne. To se je spremenilo z razvojem Vision Transformerja (ViT) [7]. Namesto da bi slike obdelovali kot dvodimenzionalne mreže pikslov (kot to počnejo konvolucijske nevronske mreže), Vision Transformer slike obravnava kot zaporedje majhnih kvadratov ali zaplat. Slednje omogoča uporabo istih tehnik samopozornosti, ki so bile učinkovite v jezikovnih modelih, tudi za obdelavo slik. Ta pristop je pokazal obetavne rezultate, saj je Vision Transformer dosegel ali presegel učinkovitost konvolucijskih nevronskih mrež na številnih nalogah računalniškega vida [7].

2.4.1 Arhitektura ViT

Arhitektura ViT obravnava slike dimenzij $H \times W \times C$ tako, da jih razdeli na zaplate dimenzij $P \times P$. Pri tem sta H in W višina in širina slike, C je število barvnih kanalov, P pa predstavlja dimenzijo zaplate. Kot rezultat

tega postopka dobimo $(H \cdot W)/P^2$ zaplat, ki se vsaka zravna v 1D vektor dolžine $P^2 \cdot C$.

Vsak 1D vektor x se nato prenese skozi linearni model:

$$z = Wx + b. \tag{2.13}$$

Ker transformerji ne vsebujejo inherentne informacije o poziciji vložkov v zaporedju, je treba dodati pozicijske vložke:

$$e_i = z_i + p_i. \tag{2.14}$$

Zaporedje vložkov se nato prenese skozi bloke transformerja, ki vsebujejo večglavo samopozornost in feed-forward mreže:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W_O. \tag{2.15}$$

Za končno klasifikacijo slike se uporabi naslednja klasifikacijska glava:

$$y = \text{softmax}(W_2\text{ReLU}(W_1 e)). \tag{2.16}$$

2.5 Piramidni ViT (PVT)

Piramidni ViT (PVT) [23] je bil razvit z namenom vključitve piramidne strukture v okviru Transformerja. Arhitektura PVT je razdeljena na štiri stopnje. Vsaka od teh stopenj je sestavljena iz plasti za vdelavo zaplat (ang. patch embedding) in iz več plasti transformerskega kodirnika. Značilnost te arhitekture je, da se izstopna ločljivost štirih stopenj postopoma zmanjšuje, kar sledi piramidni strukturi. Na najvišji stopnji je ločljivost značilnostne mape največja, medtem ko se na najnižji stopnji zmanjša.

Za boljše razumevanje si poglejmo podrobneje prvo stopnjo: vhodna slika velikosti $H \times W \times 3$ je razdeljena na zaplate velikosti $4 \times 4 \times 3$. To pomeni, da je število zaplat enako $HW/4^2$. Vsaka zaplata je nato sploščena in prenesena v linearno projekcijo, kar rezultira v vdelavi zaplat velikosti $HW/4^2 \times C1$. Te

vdelane zaplate, skupaj z dodano vdelavo položaja, prehajajo skozi Transformerski kodirnik z $L1$ plastmi. Izhod iz tega kodirnika je nato preoblikovan v značilnostno mapo $F1$ velikosti $H/4 \times W/4 \times C1$.

Matematično to lahko izrazimo kot:

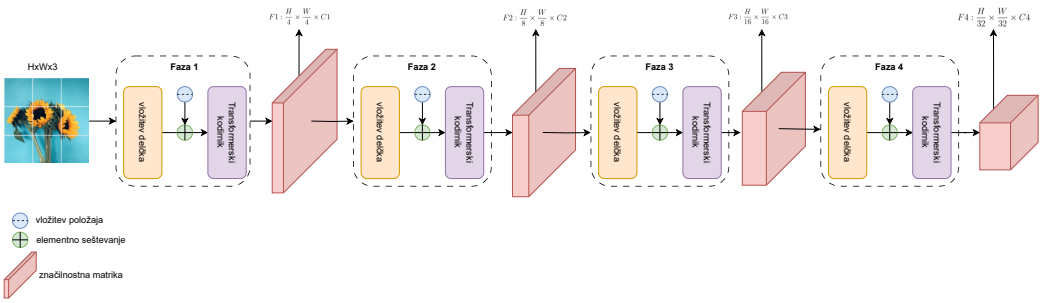
$$F1 = \frac{H}{4} \times \frac{W}{4} \times C1. \tag{2.17}$$

Naslednje stopnje PVT sledijo podobnemu pristopu, vendar z različnimi ločljivostmi in dimenzijami. Na primer, značilnostne mape $F2, F3$ in $F4$ so pridobljene z različnimi koraki, ki so 8, 16 in 32 slikovnih pik glede na vhodno sliko.

Ena izmed ključnih inovacij v PVT je uporaba pozornosti za zmanjšanje prostorskega obsega (ang. Spatial Reduction Attention – SRA) namesto tradicionalne večglave pozornostne plasti (ang. Multi Headed Attention – MHA). Ta pristop omogoča PVT-ju, da učinkovito obdela značilnostne mape visoke ločljivosti.

V primerjavi z ViT, PVT prinaša večjo prilagodljivost, saj lahko generira značilnostne mape različnih meril/kanalov v različnih fazah. Poleg tega je bolj vsestranski, saj se lahko enostavno vključi in uporabi v večini modelov za spodnje naloge. Prav tako je bolj prijazen do računalniških virov in spomina, saj lahko obdela značilnostne mape višje ločljivosti.

Na sliki 2.3 je prikazana skica PVT modela.



Slika 2.3: Skica PVT modela

2.6 Piramidni ViT z uporabo lokalnih značilnosti (PCPVT)

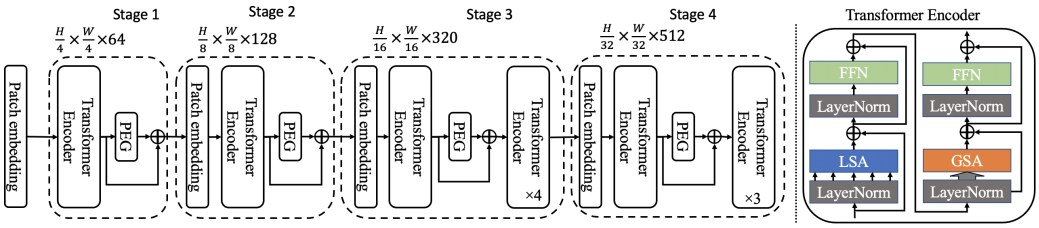
Twins-PCPVT [4] je zasnovan na osnovi PVT in CPVT [3]. Glavna razlika med Twins-PCPVT in PVT je v načinu uporabe pozicijskih kodiranj. V PVT so uporabljena absolutna pozicijska kodiranja, medtem ko Twins-PCPVT uporablja pogojna pozicijska kodiranja (ang. Conditional Positional Encoding – CPE), ki so bila predlagana v CPVT.

PVT je uvedel piramidno večstopenjsko strukturo z namenom boljšega obravnavanja nalog goste napovedi, kot so zaznavanje objektov in semantična segmentacija. Vendar je bilo ugotovljeno, da je manjša učinkovitost PVT-ja v veliki meri posledica uporabe absolutnih pozicijskih kodiranj. Absolutna pozicijska kodiranja se soočajo s težavami pri obdelavi vhodov različnih velikosti, kar je pogosto v nalogah goste napovedi.

V Twins-PCPVT so absolutna pozicijska kodiranja nadomeščena s pogojnimi pozicijskimi kodiranji (CPE), ki so odvisna od vhodov in se tako lahko naravno izognejo zgoraj omenjenim težavam. Generator pozicijskega kodiranja (ang. Positional Encoding Generator – PEG), ki generira CPE, je postavljen za prvim kodirnim blokom vsake stopnje. Uporablja najpreprostejšo obliko PEG, tj. 2D globinsko konvolucijo brez normalizacije serij.

$$CPE = f(PEG(E_1, E_2, ..., E_n)) \tag{2.18}$$

Kjer je CPE pogojno pozicijsko kodiranje, f je funkcija, ki generira kodiranje na podlagi vhodnih značilnosti, in E_i so značilnosti iz različnih stopenj kodirnika. Twins-PCPVT združuje prednosti tako PVT-ja kot CPVT-ja, kar ga naredi enostavnega za učinkovito implementacijo. Eksperimentalni rezultati so pokazali, da ta preprosta zasnova lahko doseže zmogljivost nedavno predlaganega Swin transformerja [12]. Na sliki 2.4 je prikazana skica PCPVT modela.



Slika 2.4: Skica PCPVT modela, iz članka o modelu Twins [4]

2.7 Siamska nevronska mreža za primerjavo vzorcev

Siamske nevronske mreže predstavljajo sodoben pristop v domeni primerjave vzorcev v računalniškem vidu. Z zmožnostjo učinkovite primerjave med paroma slik so siamske mreže pridobile pozornost v številnih aplikacijah, kjer je ključnega pomena zanesljiva ocena podobnosti. V tem podpoglavju bomo obravnavali osnovno arhitekturo siamske mreže, metodologijo njenega učenja ter aplikacije in prednosti, ki jih ta prinaša v prakso.

2.7.1 Osnovna arhitektura siamske mreže za primerjavo vzorcev

Klasična siamska mreža za primerjavo vzorcev sestoji iz dveh identičnih podmrež, ki delijo enake uteži. Vsaka podmreža prejme sliko: ena je ciljna slika, druga pa je iskana slika. Oba vhoda se preoblikujeta v značilnostne vektorje prek teh podmrež. Nato se izračuna razdalja med obema vektorjema, običajno z evklidsko razdaljo, da se ugotovi, kako podobni sta sliki.

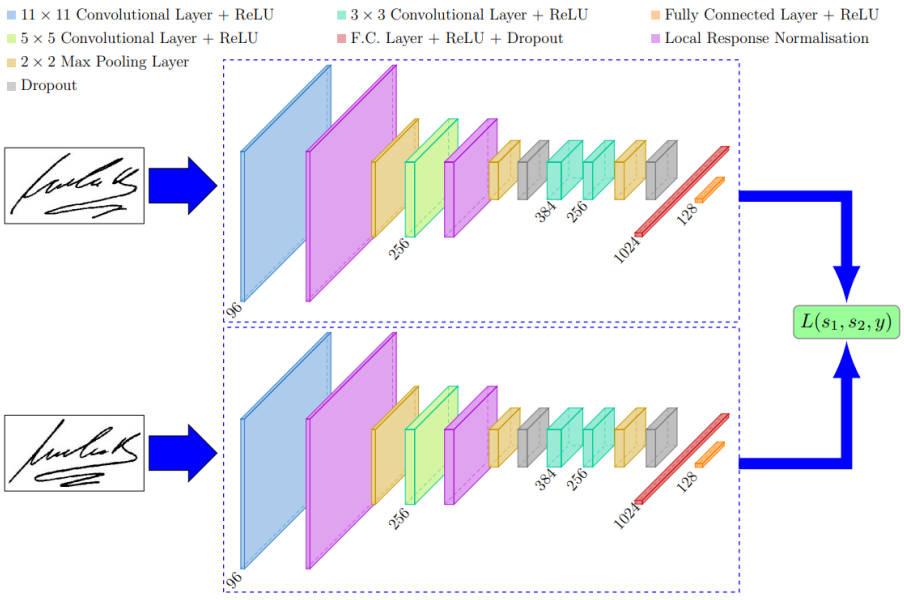
Matematično, za dve sliki x_1 in x_2, podmreži proizvedeta predstavitve $f(x_1; \theta)$ in $f(x_2; \theta)$. Razdalja D med tema dvema predstavitvama je določena kot:

$$D(f(x_1;\theta), f(x_2;\theta)) = |f(x_1;\theta) - f(x_2;\theta)|_2. \tag{2.19}$$

Da bi siamsko mrežo naučili za učinkovito primerjavo vzorcev, potrebujemo nabor učnih podatkov, ki vsebuje pare podobnih in različnih slik. Med učenjem je cilj zmanjšati razdaljo med podobnimi slikami in povečati razdaljo med različnimi slikami. Kriterijska funkcija, običajno uporabljena pri učenju siamskih mrež za primerjavo vzorcev, je kontrastna kriterijska funkcija, definirana kot:

$$L(y, D(f(x_1;\theta), f(x_2;\theta))) = y \cdot \frac{1}{2}D^2 + (1-y) \cdot \frac{1}{2}\max(0, m-D)^2, \tag{2.20}$$

kjer y označuje oznako podobnosti (1 za podobne in 0 za različne), m pa je prag, ki določa mejo med podobnimi in različnimi slikami. Na sliki 2.5 je prikazana skica siamske mreže uporabljene za primerjavo podpisov.



Slika 2.5: Skica siamske mreže, model SigNet [6]

2.7.2 Aplikacije in prednosti

Siamske mreže za primerjavo vzorcev so se izkazale za izjemno koristne v številnih aplikacijah, kot so prepoznavanje in sledenje objektom, biometrija ter varnost in nadzor. V primerjavi s tradicionalnimi metodami imajo siamske mreže večjo odpornost na variacije v svetlobi, rotaciji, lestvici in drugih deformacijah. Zaradi globje hierarhične predstavitve slike so sposobne zaznati in primerjati kompleksne značilnosti, ki jih manj kompleksne metode morda ne bi opazile.

Poglavje 3

Podatkovna množica

V svetu raziskovanja je podatkovna množica ključnega pomena za razvoj, testiranje in validacijo modelov. Kljub obstoju številnih zbirk, kot so CVUSA [11], CVACT [10] in University-1652 [26], večina ne zajema vseh realnih situacij, s katerimi se srečuje brezpilotni letalnik. Konkretno, CVUSA se osredotoča na zgradbe, medtem ko University-1652 predstavlja predvsem univerzitetna okolja. Poleg tega so objekti v sredini slike. Zaradi pomanjkljivosti obstoječih zbirk in ker zbirka UL14 iz [5] ni dostopna, smo se soočili z izzivom pridobivanja ustreznih podatkov za analizo. Zbirka vsebuje posnetke s pogledom od zgoraj navzdol in je osredotočena na pogled iz brezpilotnega letalnika.

Da bi premostili to vrzel, smo se odločili za ustvarjanje lastne zbirke. Za pridobivanje slik iz brezpilotnega letalnika smo uporabili orodje Google Earth Studio [1] in pridobili slike iz 11 evropskih mest. Te slike odražajo raznolikost terena, vključno z zgradbami, parki, zelenimi in vodnimi površinami. Dodatno smo uporabili Mapbox API [2] za pridobitev pripadajočih satelitskih slik.

Skupno naša podatkovna baza vključuje več kot 11.000 slik. Slike so bile pridobljene s simulacijo letenja in so razvrščene v koherentnem časovnem

[1] Google Earth Studio: `https://www.google.com/earth/studio/`

[2] Mapbox API: `https://www.mapbox.com/api-documentation/`

zaporedju. V članku [5, 22] so avtorji uporabili podatkovno množico UL14, ki vključuje 6.768 slik za učenje in 2.331 slik za validacijo. Ta zbirka se osredotoča večinoma na slike stavb večjih kitajskih univerz. V nasprotju s tem naša zbirka ponuja širši spekter značilnosti za analizo in bolje odraža realne okoliščine. Cilj izdelave naše zbirke je bil zagotoviti raznolike podatke, ki bi lahko služili kot robustna osnova za testiranje in validacijo naše implementacije WAMF-FPI. Poleg tega je bil namen, da so slike posnete iz zgornjega pogleda, osredotočene na pogled brezpilotnega letalnika. Želimo se prepričati, da je naš pristop robusten in da lahko obravnava različne scenarije, ki jih lahko sreča brezpilotni letalnik v realnem svetu.

3.1 Slike brezpilotnega letalnika

Nabor podatkov, ki ga predstavljamo, je bil zasnovan z namenom raziskovanja in analize lokalizacije brezpilotnih letalnikov v različnih mestnih scenarijih. Osredotoča se na dve ključni območji:

1. gosto pozidana mestna območja z zgradbami in

2. odprte zelene površine, kot so parki in travniki.

Zajem slik je bil izveden na naključnih poteh po mestu, kar omogoča širok spekter scenarijev. V mestnih območjih je poudarek na razumevanju, kako se brezpilotni letalniki lokalizirajo in navigirajo med visokimi zgradbami, kjer so lahko GPS signali zmanjšani ali moteni. V zelenih območjih je cilj razumeti, kako se brezpilotni letalniki obnašajo v okoljih, kjer so vizualni vzorci manj unikatni. V naboru podatkov za učenje je 10.000 slik iz desetih mest, pri čemer vsako mesto prispeva 1.000 slik. Vsaka slika je opremljena z oznakami lokacije kamere v sistemu ECEF. Sistem ECEF (ang. Earth Centered, Earth Fixed) je globalni koordinatni sistem z izhodiščem v središču Zemlje. Brezpilotni letalniki so bili kalibrirani na višini 150 metrov nad navedeno nadmorsko višino mesta. Kamere na brezpilotnih letalnikih imajo

vidno polje 80 stopinj in so usmerjene pravokotno na središče Zemlje. Vse slike so bile ustvarjene z uporabo orodja Google Earth Studio [9].

V naboru so mesta s tipično evropsko arhitekturo, kombinacijo zelenja in stavb. Mesta, vključena v nabor podatkov, so:

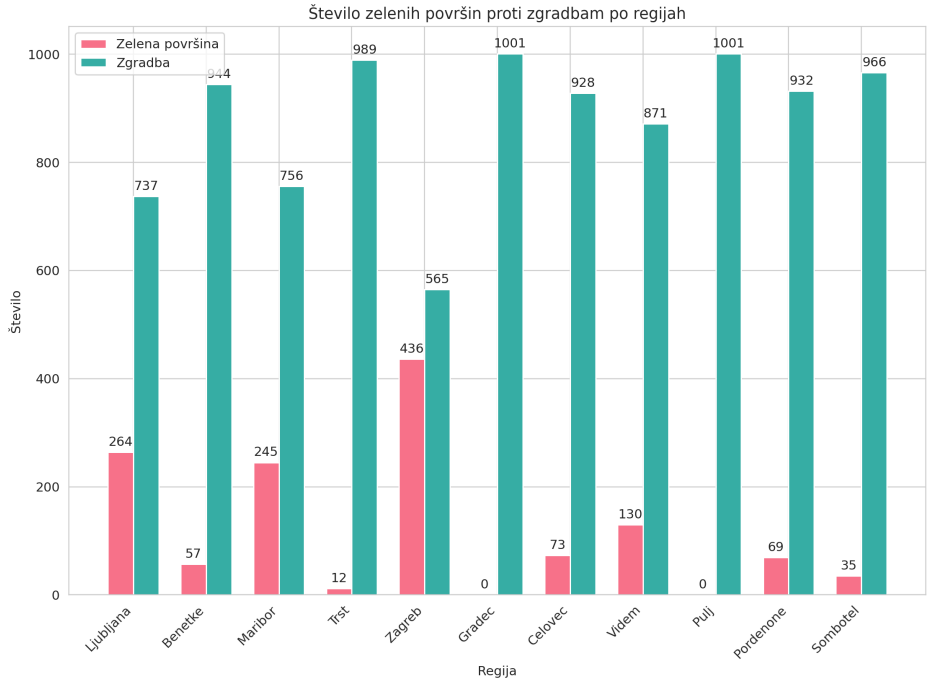
- **Maribor:** nadmorska višina: 272 m, višina brezpilotnega letalnika: 150 m, skupaj: 422 m nad morsko gladino.
- **Trst:** nadmorska višina: 23 m, višina brezpilotnega letalnika: 150 m, skupaj: 173 m nad morsko gladino.
- **Zagreb:** nadmorska višina: 158 m, višina brezpilotnega letalnika: 150 m, skupaj: 308 m nad morsko gladino.
- **Gradec:** nadmorska višina: 353 m, višina brezpilotnega letalnika: 150 m, skupaj: 503 m nad morsko gladino.
- **Celovec:** nadmorska višina: 446 m, višina brezpilotnega letalnika: 150 m, skupaj: 596 m nad morsko gladino.
- **Videm:** nadmorska višina: 113 m, višina brezpilotnega letalnika: 150 m, skupaj: 263 m nad morsko gladino.
- **Pula:** nadmorska višina: 17 m, višina brezpilotnega letalnika: 150 m, skupaj: 167 m nad morsko gladino.
- **Pordenone:** nadmorska višina: 24 m, višina brezpilotnega letalnika: 150 m, skupaj: 174 m nad morsko gladino.
- **Szombathely:** nadmorska višina: 212 m, višina brezpilotnega letalnika: 150 m, skupaj: 362 m nad morsko gladino.
- **Benetke:** nadmorska višina: -1 m, višina brezpilotnega letalnika: 150 m, skupaj: 149 m nad morsko gladino.
- **Ljubljana:** nadmorska višina: 295 m, višina brezpilotnega letalnika: 150 m, skupaj: 445 m nad morsko gladino.

Na Sliki 3.2 je prikazana razdelitev zelenih površin in stavb za različna mesta, temelječa na analizi slik, ki smo jih zajeli v našem podatkovnem naboru. Vsako mesto razkriva svojo edinstveno strukturo in raven urbanizacije. Te razlike so ključnega pomena pri razumevanju izzivov, s katerimi se srečujejo brezpilotni letalniki pri lokalizaciji in navigaciji v različnih mestnih okoljih.

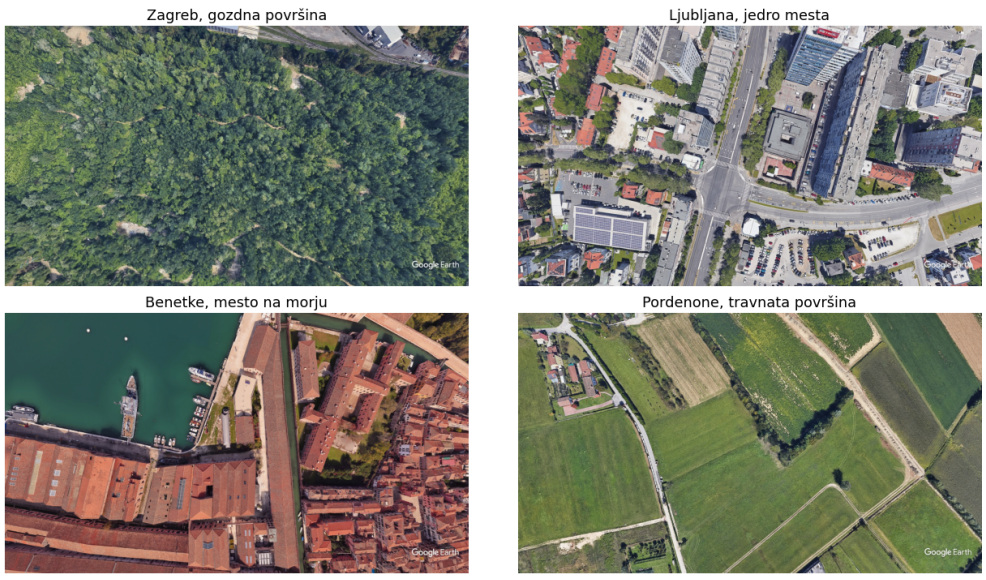
Slika 3.1: Slika prikazuje lokacije mest, ki so vključena v nabor podatkov.

Nekatera mesta, kot sta Gradec in Pula, kažejo višjo stopnjo urbanizacije z minimalno prisotnostjo zelenih površin. To pomeni, da bodo brezpilotni letalniki v teh okoljih večinoma navigirali med zgradbami. Na drugi strani pa mesta, kot je Zagreb, predstavljajo večjo mešanico zgradb in zelenih površin. Takšne razlike lahko vplivajo na algoritme lokalizacije in navigacije brezpilotnih letalnikov, saj se morajo prilagajati različnim scenarijem in oviram.

Na slikah 3.3 in 3.4 so prikazani raznoliki primeri zajeti z brezpilotnim letalnikom.



Slika 3.2: Graf prikazuje razmerje med zelenimi površinami in stavbami za vsako mesto.



Slika 3.3: Raznoliki primeri slik, zajetih z brezpilotnim letalnikom.

Slika 3.4: Raznoliki primeri slik, zajetih z brezpilotnim letalnikom.

3.2 Satelitske slike

Za vsako sliko posneto z brezpilotnim letalnikom smo poiskali ustrezno satelitsko zaplato. Ta korak je zagotovil, da so satelitske slike popolnoma usklajene s slikami posnetimi iz brezpilotnega letalnika v smislu geografske lokacije. Ko smo identificirali ustrezno satelitsko zaplato, smo jo prenesli neposredno iz Mapbox API-ja [3], vira za visokokakovostne satelitske slike. Da bi zagotovili dodatno globino in kontekst za vsako lokacijo, nismo prenesli samo osrednje zaplate, temveč tudi vse njene sosednje zaplate. Te sosednje zaplate smo nato združili z osrednjo zaplato za ustvarjanje enotne TIFF datoteke.

Pretvorbo geografskih koordinat (latitudo in longitudo) v zaplatne koordinate (x, y) na določeni ravni povečave z uporabo Mercatorjeve projekcije, lahko izrazimo:

[3]Mapbox API: `https://www.mapbox.com/api-documentation/`

- Pretvorba geografskih koordinat v radiane:

$$\text{lat}_{\text{rad}} = \text{latitude} \times \frac{\pi}{180},$$
$$\text{lon}_{\text{rad}} = \text{longitude} \times \frac{\pi}{180}.$$

- Pretvorba radianov v normalizirane koordinate Mercatorja:

$$x = \frac{\text{lon}_{\text{rad}} = +\pi}{2\pi},$$
$$y = \frac{\pi - \log(\tan(\frac{\pi}{4} + \frac{\text{lat}_{\text{rad}}}{2}))}{2\pi}.$$

- Pretvorba normaliziranih koordinat v zaplatne koordinate:

$$\text{tile}_{\text{x}} = \text{floor}(x \times 2^z),$$
$$\text{tile}_{\text{y}} = \text{floor}(y \times 2^z).$$

Na slikah 3.5 in 3.6 so prikazani primeri pripadajočih satelitskih slik za slike zajete z brezpilotnim letalnikom.

3.3 Oznake

V okviru raziskave smo iz visokoločljivostnih satelitskih TIFF datotek naključno izrezali regije velikosti 400 x 400 pikslov. Pri vsaki iteraciji je bil izrez drugačen, s poudarkom na vključevanju referenčne točke lokalizacije v izrez. Ta pristop zagotavlja izpostavljenost modela različnim scenarijem ob ohranjanju natančnosti lokalizacijskih podatkov. Slike, pridobljene z brezpilotnimi letalniki, so bile obdelane s tehniko *osrednjega izreza* in različnimi stopnjami povečave, združujoč detajlnost teh slik z obsežnostjo satelitskih posnetkov. Spodaj na slikah 3.7, 3.9, 3.8, 3.10 in 3.11 je prikazanih nekaj primerov takšnih izrezov. Na vsaki sliki je s pomočjo rdečega kroga označen center izreza, ki predstavlja referenčno točko lokalizacije iz brezpilotnega letalnika, s čimer je omogočeno lažje prepoznavanje osredotočenosti izreza.

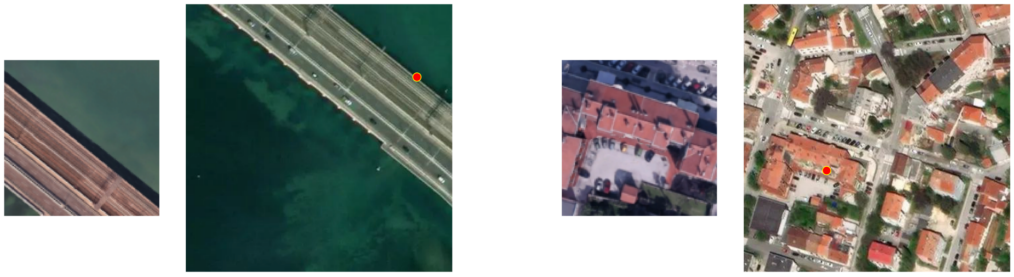
Slika 3.5: Primer pripadajočih satelitskih slik.

Slika 3.6: Primer pripadajočih satelitskih slik.

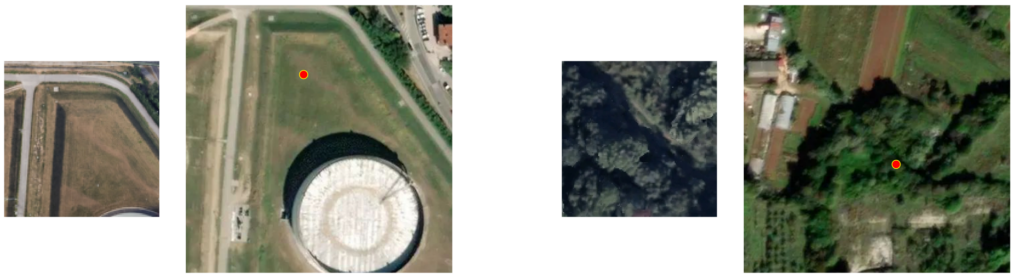
Slika 3.7: Leva slika prikazuje Gradec z dvakratno povečavo slike iz brezpilotnega letalnika, desna pa Trst z 2,5-kratno povečavo slike iz brezpilotnega letalnika.

Slika 3.8: Leva slika prikazuje mesto Szombathely z 1,5-kratno povečavo slike iz brezpilotnega letalnika, desna pa z dvakratno povečavo slike iz brezpilotnega letalnika.

Slika 3.9: Leva slika prikazuje Zagreb z 1,5-kratno povečavo slike iz brezpilotnega letalnika, desna pa mesto Szombathely z 2,5-kratno povečavo slike iz brezpilotnega letalnika.



Slika 3.10: Leva slika prikazuje Benetke s trikratno povečavo slike iz brezpilotnega letalnika, desna pa mesto Pula z dvakratno povečavo slike iz brezpilotnega letalnika.



Slika 3.11: Leva slika prikazuje Trst s 3,5-kratno povečavo slike iz brezpilotnega letalnika, desna pa mesto Pula z enkratno povečavo slike iz brezpilotnega letalnika.

Poglavje 4

Implementacija

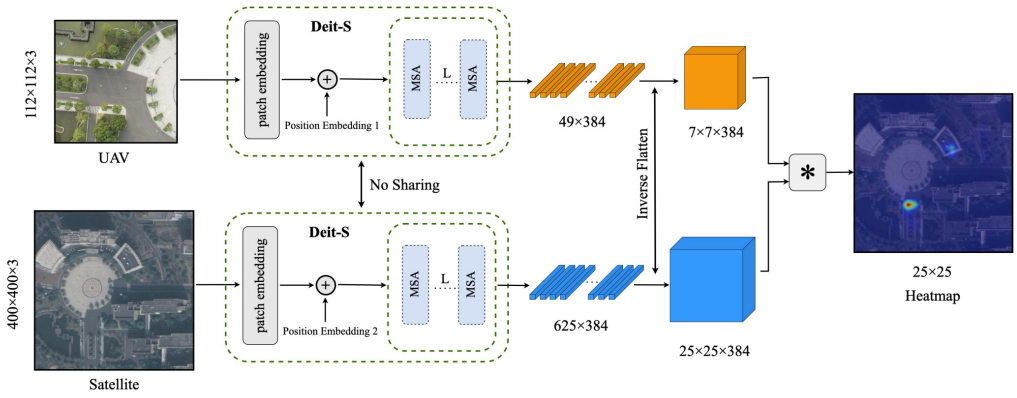
V tem poglavju se bomo osredotočili na implementacijo modela WAMF-FPI, RDS metriko ter samo učenje modela. Odločili smo se za lastno implementacijo metode, saj originalna koda iz članka [22] ni bila javno dostopna. V naslednjem podpoglavju bomo podrobneje predstavili postopek implementacije.

4.1 Implementacija metode WAMF-FPI

Sledenje objektov v okviru računalniškega vida običajno temelji na izračunu podobnosti med referenčno in iskalno podobo v trenutnem okviru. Medtem ko temeljna metoda za iskanje točk znotraj slike izhaja iz metodologije sledenja objektov, je prva v primerjavi z drugo bolj zapletena. To je posledica različnih perspektiv med predlogo (sliko posneto z brezpilotnim letalnikom) in iskalno sliko (satelitsko sliko), ki povzročajo veliko variacijo.

Metoda iskanja točk uporablja satelitsko sliko kot referenčno in sliko iz brezpilotnega letalnika kot poizvedbo. Obe sliki – posneti z brezpilotnim letalnikom in satelitsko sliko relevantnega območja – se nato preneseta v end-to-end mrežo. Po obdelavi je rezultat toplotna karta, kjer točka z najvišjo vrednostjo predstavlja lokacijo brezpilotnega letalnika, kot jo predvideva model. Lokacijo nato preslikamo na satelitsko sliko, pri čemer položaj brezpilo-

tnega letalnika določimo na podlagi geografske širine in dolžine, ki jih vsebuje satelitska slika. V [5] avtorji kot modul za izluščenje značilnosti uporabljajo dva Deit-S [20] brez deljenih uteži za vertikalne poglede slike brezpilotnega letalnika in satelitske slike. Izluščene značilnosti nato uporabimo za izračun podobnosti in izdelavo toplotne karte. Lokacijo z najvišjo vrednostjo toplotne karte nato preslikamo na satelitsko sliko, da določimo lokacijo brezpilotnega letalnika. Na sliki 4.1 je prikazana skica modela FPI [5].

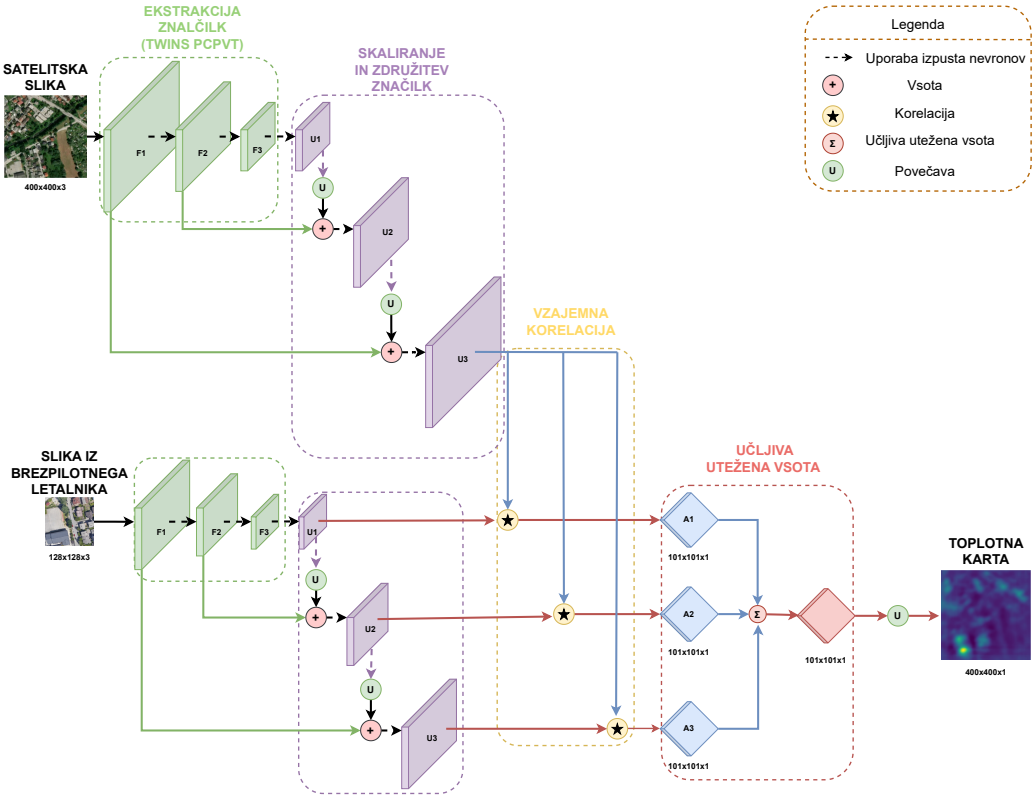


Slika 4.1: Skica modela FPI, iz članka [5].

V FPI je za izračun podobnosti uporabljena zadnja plast zemljevidnih značilnosti [5]. Zaradi tega, ker je izhodna toplotna karta 16-krat manjša od vhodne satelitske slike, model izgubi veliko prostorskih informacij, kar vodi v znatno izgubo natančnosti pri določanju lokacije.

Da bi izboljšali lokalizacijske sposobnosti modela, smo uporabili strukturo piramidnih značilnosti (Twins-PCPVT) in modul utežno prilagodljivega združevanja večznačilnostnih lastnosti (WAMF). K osnovnemu modelu so bile dodane izboljšave z vključitvijo dveh močnejših PCPVT-S modulov za izluščenje značilnosti iz slik brezpilotnega letalnika in satelitskih slik. Da bi bolje zajeli informacije na različnih ločljivostih in ohranili več prostorskih informacij, so bile prvotno izluščene značilnosti poslane v omrežje značilnostne piramide za nadaljnjo obdelavo. Modul WAMF je bil nato uporabljen za

izračun podobnosti in združevanje različnih značilnosti. Končne združene značilnosti so bile razširjene za izdelavo končne izhodne napovedne mape. Rezultat je toplotna karta iste velikosti kot vhodna satelitska slika v modelu WAMF-FPI. Na sliki 4.2 je prikazana skica arhitekture modela WAMF-FPI.



Slika 4.2: Skica arhitekture modela WAMF-FPI

4.1.1 Modul za izluščenje značilnosti

Model WAMF-FPI temelji na strukturi, ki je podobna siamski arhitekturi, vendar se od tradicionalnih metod sledenja objektom loči v določenih ključnih točkah, ki jih bomo v tem razdelku opisali. Zaradi občutne razlike med satelitskimi slikami in slikami brezpilotnega letalnika, ki izvirajo iz različnih naprav, veji modela WAMF-FPI za vsako od teh vrst slik ne uporabljata

metode deljenja uteži. WAMF-FPI kot vhod uporablja satelitske slike dimenzij 400 × 400 × 3 in slike brezpilotnega letalnika dimenzij 128 × 128 × 3. Značilnosti obeh vrst slik so izluščene s pomočjo PCPVT-S.

V modelu smo odstranili zadnjo stopnjo PCPVT-S in uporabili samo prve tri stopnje za izluščene značilnosti. Pri dimenzijah vhodnih slik 400 x 400 x 3 in 128 x 128 x 3 oba pristopa pridobita značilnostne mape z obliko 25 x 25 x 256 in 8 x 8 x 320.

V primerjavi z Deit-S [20], ki je bil uporabljen v FPI [5], ima PCPVT-S piramidno strukturo. Ta struktura je bolj prilagojena za naloge goste napovedi. Pravzaprav uporaba piramidne strukture zagotavlja osnovo za kasnejšo integracijo modula WAMF. Poleg tega mreža s piramidno strukturo lahko zmanjša obseg potrebnih izračunov in s tem izboljša hitrost procesiranja, kar je ključno za učinkovito uporabo metode v praksi.

Po obdelavi slike s pomočjo PCPVT-S se podobnost neposredno izračuna na zadnjih značilnostnih mapah. Kljub temu je končni izhod stisnjen samo za faktor štiri v primerjavi z vhodom, kar je potem z bikubično interpolacijo povečano nazaj na velikost vhodne satelitske slike.

Pristranskost, ki je posledica nizke ločljivosti značilnostne mape, je bila odstranjena že na samem začetku. Ker značilnostna mapa z visoko ločljivostjo vsebuje več prostorskih informacij, je bila združena z globoko značilnostno mapo, bogato s semantičnimi informacijami, preko lateralne povezovalne strukture.

Prva faza obdelave v metodi WAMF-FPI je uporaba konvolucijskega jedra velikosti ena, ki prilagodi kanalsko dimenzijo tri-stopenjske značilnostne mape, pridobljene s pomočjo PCPVT-S. Število izhodnih kanalov je bilo nastavljeno na 64, kar zagotavlja kompaktno in učinkovito zastopanje značilnosti. Po tej fazi sledi upsampling operacija na značilnostnih mapah zadnjih dveh stopenj, ki poveča njihovo ločljivost in s tem omogoča bolj precizno lokalizacijo. Te mape se nato kombinirajo z značilnostnimi mapami istega merila iz osnovnega modela.

Končno, značilnosti so dodatno izluščene s pomočjo konvolucijskega je-

dra velikosti 3, kar modelu omogoča izluščenje bolj kompleksnih značilnosti iz združenih map. Rezultat je združena značilnostna mapa, ki združuje plitve (prostorske) in globoke (semantične) informacije. Ta bogata kombinacija modelu omogoča učinkovito prepoznavanje in lokalizacijo objektov na vhodnih slikah.

4.1.2 Arhitektura utežno-prilagodljivega združevanja večznačilnostnih lastnosti (WAMF)

Modul za združevanje značilnosti je zasnovan tako, da združuje informacije iz dveh ločenih vhodnih tokov, v tem primeru iz UAV (brezpilotnega letalnika) in SAT (satelita). Ta modul uporablja piramido značilnosti iz obeh in izračuna korelacije med njimi, da jih združi v en sam izhod.

Za začetek se izvedejo konvolucijske operacije na značilnostnih mapah UAV in SAT. Konvolucijske operacije so izvedene s konvolucijskimi jedri velikosti 1×1, kar omogoča prilagoditev kanalskih dimenzij značilnostnih map.

Za UAV značilnostne mape:

$$U1_{\text{UAV}} = \text{Conv1UAV}(s3\text{UAV}) \tag{4.1}$$

$$U2_{\text{UAV}} = \text{Povečava}(U1_{\text{UAV}}) + \text{Conv2UAV}(s2\text{UAV}) \tag{4.2}$$

$$U3_{\text{UAV}} = \text{Povečava}(U2_{\text{UAV}}) + \text{Conv3UAV}(s1\text{UAV}). \tag{4.3}$$

Za SAT značilnostne mape:

$$U1_{\text{SAT}} = \text{Conv1SAT}(s3\text{SAT}) \tag{4.4}$$

$$U2_{\text{SAT}} = \text{Povečava}(U1_{\text{SAT}}) + \text{Conv2SAT}(s2\text{SAT}) \tag{4.5}$$

$$U3_{\text{SAT}} = \text{Povečava}(U2_{\text{SAT}}) + \text{Conv3SAT}(s1\text{SAT}), \tag{4.6}$$

kjer je Povečava funkcija, ki poveča prostorsko resolucijo značilnostne mape z uporabo bikubične interpolacije.

$$A1 = \text{corr}(U1_{\text{UAV}}, U3_{\text{SAT}}) \tag{4.7}$$

$$A2 = \text{corr}(U2_{\text{UAV}}, U3_{\text{SAT}}) \tag{4.8}$$

$$A3 = \text{corr}(U3_{\text{UAV}}, U3_{\text{SAT}}), \tag{4.9}$$

kjer je corr funkcija za izračun korelacije med dvema značilnostnima mapama.

Korelacija v kontekstu obdelave slik je postopek izračuna podobnosti med dvema slikama ali značilnostnima mapama. V osnovi ena značilnostna mapa (poimenovana poizvedba) drsi čez drugo značilnostno mapo (poimenovana iskalna regija) in izračuna podobnost med njima na vsaki lokaciji. Rezultat tega postopka je nova značilnostna mapa, imenovana korelacijska mapa, kjer vsaka vrednost predstavlja stopnjo podobnosti med poizvedbo in delom iskalne mape na določeni lokaciji.

Matematično je korelacija med dvema funkcijama f in g definirana kot:

$$(f \star g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t + \tau) d\tau. \tag{4.10}$$

V kontekstu diskretnih signalov, kot so slike ali značilnostne mape, je korelacija definirana kot:

$$(f \star g)[n] = \sum_{m=-\infty}^{\infty} f[m] g[n + m]. \tag{4.11}$$

Nazadnje se izvede uteženo združevanje teh treh koreliranih značilnostnih map s pomočjo naučljivih uteži:

$$\text{združena\_mapa} = w_1 \cdot A1 + w_2 \cdot A2 + w_3 \cdot A3. \tag{4.12}$$

Za dokončanje postopka se uporabi bikubična interpolacija, da se združena mapa poveča na velikost vhodne satelitske slike. Na izhodu dobimo toplotno karto iste velikosti kot vhodna satelitska slika v WAMF-FPI.

4.1.3 RDS metrika

Da bi lahko ovrednotili in primerjali zmogljivost našega modela, uporabljamo metriko RDS [22]. Zaradi različnih meril podatkov v naboru podatkov vsak piksel v različnih satelitskih slikah predstavlja različno razdaljo. Čeprav model morda najde točko, ki je na satelitski sliki blizu dejanske lokacije, lahko v resničnem prostoru povzroči veliko napako. Da bi se izognili težavam zaradi spremembe merila, RDS izračuna relativno razdaljo na ravni pikslov med napovedano in dejansko točko.

Enačba za izračun RDS je naslednja:

$$RDS = e^{-k \times \frac{\sqrt{\left(\frac{dx}{w}\right)^2 + \left(\frac{dy}{h}\right)^2}}{2}}, \tag{4.13}$$

kjer so:

- w širina v pikslih satelitske slike,
- h višina v pikslih satelitske slike,
- dx pikselska razdalja med vodoravnimi koordinatami napovedane pozicije in dejanske pozicije,
- dy pikselska razdalja med navpičnimi koordinatami napovedane pozicije in dejanske pozicije,
- k je faktor merila, ki je v tem delu postavljen na 10.

Za lažje razumevanje delovanja RDS metrike smo dodali dodatek A k diplomskemu delu, ki vsebuje tri primere izračuna metrike.

4.1.4 Učenje modela

Model smo učili na računalniškem sistemu s procesorjem Intel(R) Xeon(R) CPU E5-2690 v3 @ 2.60GHz z 12 jedri ter grafično kartico NVIDIA GeForce RTX 3060 z 12 GB pomnilnika. Razvoj je temeljil na platformi Ubuntu z

uporabo Python [1] knjižnice PyTorch [2]. V času učenja našega modela ni prišlo do povečane energetske porabe, saj je bil računalnik neprestano napajan iz lokalne sončne elektrarne. To pomeni, da je bil celoten postopek učenja izveden na okolju prijazen način, brez dodatnega obremenjevanja električnega omrežja ali uporabe fosilnih goriv.

Za dosego optimalnih rezultatov smo uporabili naslednje specifične hiperparametre in nastavitve:

Hitrost učenja: uporabljena sta bila dva različna parametra: $lr\_fusion = 0.0004$ za združevanje in $lr\_backbone = 0.0001$ za osnovno arhitekturo.

Prilagajanje hitrosti učenja: $gamma = 0.2$ z mejniki na epohah 2, 3 in 5.

Delovni procesi: skupno 24 hkratnih delovnih procesov (`num_workers = 24`).

Epoh: Model je bil učen skozi 10 epoh.

Velikost serije: `batch_size = 16`.

Mešanje podatkov: podatki so bili premešani pred vsako epoho.

Funkcija izgube: uporabljena je bila `hanning` funkcija.

Vizualizacija: vključena za spremljanje napredka učenja.

Za vsako iteracijo učenja smo iz vsake satelitske TIFF datoteke naključno izrezali regijo velikosti 400 x 400 pikslov. Ključnega pomena je bilo, da se je točka lokalizacije vedno nahajala nekje znotraj te izrezane regije. Med postopkom učenja smo izvajali osrednji izrez (ang. center crop) velikosti 128 x 128 pikslov iz slike z ločljivostjo 1920 x 1080. Ta pristop nam je omogočil simulacijo različnih višin brez potrebe po generiranju podatkovnega nabora z različnimi višinami. Z uporabo povečav s faktorji [1.0, 1.5, 2.0, 2.5, 3.0, 3.5]

[1] Programski jezik Python: `https://www.python.org/`
[2] Knjižnica Pytorch: `https://pytorch.org/`

in nadaljnjim osrednjim izrezom smo ustvarili umetno elevacijo. Ta metoda nam je zagotovila, da je bil model izpostavljen širokemu naboru scenarijev in kontekstov, hkrati pa smo ohranili natančnost in relevantnost lokalizacijskih podatkov. S tem pristopom smo uspešno sestavili nabor podatkov, ki združuje najboljše iz obeh svetov: detajlnost slik posnetih z brezpilotnim letalnikom in širino satelitskih slik, kar omogoča poglobljeno analizo in učinkovito učenje.

Poglavje 5

Eksperimentalna evalvacija

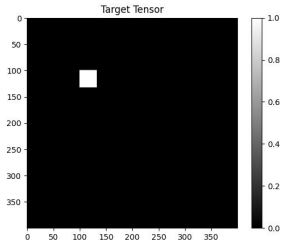
V tem poglavju so podrobno predstavljeni rezultati, doseženi v različnih fazah eksperimentalne evalvacije modela WAMF-FPI. Začeli smo z iskanjem optimalne kriterijske funkcije, da bi bolje razumeli, katera funkcija bi lahko prinesla najboljše rezultate. Nadaljevali smo s preučevanjem stratificiranega vzorčenja, tehnike, ki bi lahko pripomogla k izboljšanju natančnosti in robustnosti modela. Pregledali smo tudi vpliv Hanningovega okna in analizirali, kako različne velikosti tega okna vplivajo na končne rezultate. V zaključni fazi naših eksperimentov smo se osredotočili na regularizacijo, predvsem na tehniko izpuščanja nevronov, ter raziskali možnosti in prednosti uporabe prednaučene mreže. Vsako od teh področij je v nadaljevanju podrobno obravnavano, pri čemer so podane analize, interpretacije in ključne ugotovitve.

5.1 Izbira kriterijske funkcije

Zanimalo nas je, kako se bo model obnesel pri uporabi različnih kriterijskih funkcij. Predvidevamo, da bo Hanningovo okno kot kriterijska funkcija prineslo najboljše rezultate, saj jih je tudi v [22], medtem ko pričakujemo, da bo Krizno utežena srednja kvadratna napaka prav tako pokazala dobre rezultate

5.1.1 Hanningova kriterijska funkcija

V članku WAMF-FPI [22] so avtorji predlagali uporabo Hanningove kriterijske funkcije. Prvi pomemben vidik te funkcije izgube je dodelitev uteži vzorcem. Namesto enakega pomena vseh pozitivnih vzorcev, kriterijska funkcija Hanning dodeli različne uteži glede na lokacijo vzorca.



Slika 5.1: Primer vzorca, središče je točka lokacije vzorca.

To je zato, ker je pomembnost središčnega položaja veliko večja kot pomembnost robovnih položajev, kar je v kontekstu satelitskih slik logično in smiselno. Za normalizacijo teh pozitivnih uteži se uporablja Hanningovo okno, za normalizacijo negativnih uteži pa 1/#negativnih vzorcev. Uteži so dodeljene tako, da je vsota uteži pozitivnih in negativnih vzorcev enaka 1. A ker je število negativnih vzorcev običajno večje od števila pozitivnih vzorcev, postane utež negativnih vzorcev manjša. Da bi slednje popravili, se uvede hiperparameter, imenovan Negativna utež (NG), ki prilagodi utež negativnih vzorcev.

Hanningova funkcija:

$$\text{Hanning}(n) = \begin{cases} 0.5 + 0.5\cos\left(\frac{2\pi n}{M-1}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases} \tag{5.1}$$

Uteži primerov:

- Utež negativnih vzorcev:

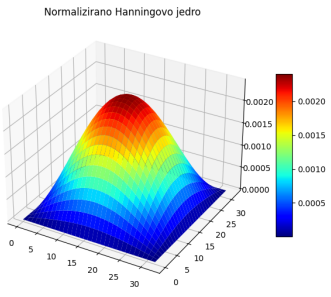
$$w_{pos} = NG/(NN(NW+1)).$$

- Utež pozitivnih vzorcev:

$$w_{neg} = HN(n)/(NW + 1),$$

kjer je:

- **NG** je Negativna utež
- **NN** je število vseh uzorcev
- **NW** je normalizacijski faktor
- **HN(n)** je vrednost Hanningove funkcije na lokaciji.



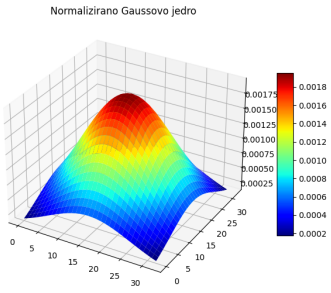
Slika 5.2: Normalizirano Hanningovo jedro.

5.1.2 Gaussovo utežena srednja kvadratna napaka

Gaussovo utežena srednja kvadratna napaka (ang. Gaussian Weighted Mean Squared Error – GWMSE) je spremenjena funkcija izgube, namenjena izboljšanju modelov, ki obravnavajo podatke, kot so satelitske slike. Glavna značilnost GWMSE je dodeljevanje uteži vzorcem na zelo podoben način kot pri Hanningovi funkciji izgube. Namesto enakega pomena vseh pozitivnih vzorcev, GWMSE različnim vzorcem dodeljuje različne uteži glede na njihovo lokacijo. Za normalizacijo teh uteži se uporablja Gaussova funkcija.

Gaussova funkcija:

$$\text{Gauss}(n) = \begin{cases} \exp\left(-\frac{(n-\mu)^2}{2\sigma^2}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases}. \tag{5.2}$$



Slika 5.3: Normalizirano Gaussovo jedro

5.1.3 Hanningovo utežena srednja kvadratna napaka

Hanningovo utežena srednja kvadratna napaka (ang. Hanning Weighted Mean Squared Error – HWMSE) je spremenjena funkcija izgube, namenjena izboljšanju modelov, ki obravnavajo podatke, kot so satelitske slike. Glavna značilnost HWMSE je dodeljevanje uteži vzorcem na zelo podoben način kot pri Gaussovi funkciji izgube. Namesto enakega pomena vseh pozitivnih vzorcev, HWMSE različnim vzorcem dodeljuje različne uteži glede na njihovo lokacijo. Za normalizacijo teh uteži se uporablja Hanningovo okno.

Hanningova funkcija je podana kot:

$$\text{Hanning}(n) = \begin{cases} 0.5 + 0.5\cos\left(\frac{2\pi n}{M-1}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases} \tag{5.3}$$

5.1.4 Križno utežena srednja kvadratna napaka

Funkcija izgube križno utežena srednja kvadratna napaka (ang. Cross-Weighted Mean Squared Error – CWMSE) je različica standardne srednje kvadratne

napake (Mean Squared Error – MSE), ki vključuje uteževanje dveh različnih skupin vzorcev: tistih, katerih resnična vrednost je večja od 0 (t. i. "resničnih"vzorcev) in tistih, katerih resnična vrednost je manjša ali enaka 0 (t. i. "neresničnih"vzorcev). Končna funkcija izgube se izračuna kot utežena kombinacija srednjih kvadratnih napak za "resnične"in "neresnične"vzorce, pri čemer se uteži vzorcev različnih skupin prekrižajo. Ta pristop se formalno izraža z naslednjo enačbo:

$$\text{loss} = \frac{\text{true\_weight} \cdot N_{\text{true}} \cdot \text{MSE}_{\text{false}} + \text{false\_weight} \cdot N_{\text{false}} \cdot \text{MSE}_{\text{true}}}{N_{\text{all}}}. \quad (5.4)$$

- N_{true}: število vzorcev, katerih resnična vrednost je večja od 0.
- N_{false}: število vzorcev, katerih resnična vrednost je enaka ali manjša od 0.
- N_{all}: skupno število vzorcev.
- MSEtrue = $\frac{1}{N\text{true}} \sum_{i=1}^{N_{\text{true}}} (y_i - \hat{y}_i)^2$ za vzorce, katerih resnična vrednost je večja od 0.
- MSEfalse = $\frac{1}{N\text{false}} \sum_{i=1}^{N_{\text{false}}} (y_i - \hat{y}_i)^2$ za vzorce, katerih resnična vrednost je enaka ali manjša od 0.
- true_weight in false_weight: uteži, dodeljene skupinama *true* in *false*.

5.1.5 Primerjava rezultatov

V kontekstu geolokalizacije brezpilotnih letalnikov v modelu WAMF-FPI je Hanningova kriterijska funkcija izkazala izjemno učinkovitost glede na vrednosti RDS. Kot je razvidno iz Tabele 5.1, razmerje RDS_{train} za Hanningovo kriterijsko funkcijo je 0.893, kar kaže na visoko natančnost pri učenju modela. Čeprav se razmerje RDS_{val} zmanjša na 0.709, je še vedno precej višje v primerjavi z drugimi preučevanimi kriterijskimi funkcijami. V primerjavi s

Hanningovo kriterijsko funkcijo so druge kriterijske funkcije praktično neuporabne, kar potrjuje, da je Hanningova kriterijska funkcija optimalna izbira za geolokalizacijo brezpilotnih letalnikov v obravnavanem modelu [22].

Kriterijska funkcija	vrednost	RDS_{train}	RDS_{val}	$\overline{\Delta_m}$ [m]
HANN	8.49	0.893	0.709	43.42
GWMSE	0.001	0.077	0.074	234.48
HWMSE	4.04e-06	0.061	0.059	232.55
CWMSE	0.007	0.07	0.06	242.70

Tabela 5.1: Rezultati ob uporabi različnih kriterijskih funkcij, kjer je $\overline{\Delta_m}$ povprečna napaka v metrih.

V tabeli 5.2 prikazujemo napako v odstotkih za različne kriterijske funkcije glede na razdaljo. Iz tabele je razvidno, kolikšen delež primerov ima napako manjšo od določene razdalje.

Kriterijska funkcija	$<$ 10m [%]	$<$ 20m [%]	$<$ 50m [%]	$<$ 100m [%]
HANN	65.22	71.66	75.87	81.45
GWMSE	0.10	0.37	1.96	8.08
HWMSE	0.00	0.10	1.35	6.71
CWMSE	0.05	0.22	1.93	8.59

Tabela 5.2: Rezultati ob uporabi različnih kriterijskih funkcij, kjer je prikazan odstotek primerov z napako manjšo od določene razdalje.

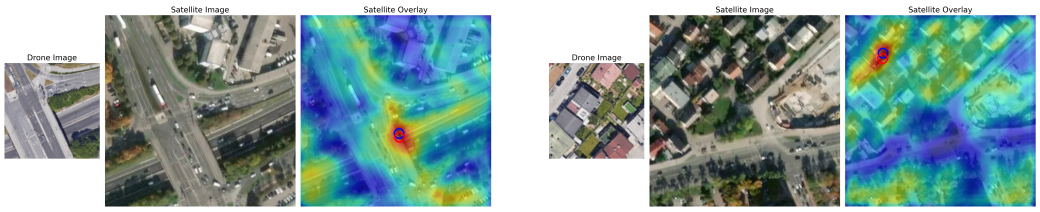
5.1.6 Analiza rezultatov

Hanningova kriterijska funkcija, ki je značilna po dodeljevanju uteži vzorcem glede na njihovo lokacijo, je na učni množici dosegla skupno vrednost 8.49 in RDS_{train} vrednost 0.893. Kljub temu, da je na validacijski množici dosegla nekoliko nižjo RDS_{val} vrednost 0.709, to kaže, da se je znanje dobro preneslo na validacijsko množico. Zaradi teh pozitivnih rezultatov smo Hanningovo kriterijsko funkcijo uporabljali v nadaljnjem testiranju.

V spodnjem razdelku so na slikah 5.4, 5.5, 5.6, 5.7, 5.8 in 5.9 predstavljeni primeri lokalizacije z modelom WAMF-FPI na vzorcu iz podatkovne množice za Ljubljano. Ti izbrani primeri osvetljujejo uspešnost in pomanjkljivosti modela pri obvladovanju kompleksnih scenarijev lokalizacije. S pomočjo teh primerov lahko podrobneje razumemo zmogljivosti in omejitve uporabljenega modela v praksi. Na vsaki sliki je z rdečim krogcem označena dejanska lokacija (ang. ground truth), medtem ko je z modrim krogcem označena predikcija modela, ki predstavlja najvišjo točko v toplotni karti.

Primeri dobre lokalizacije

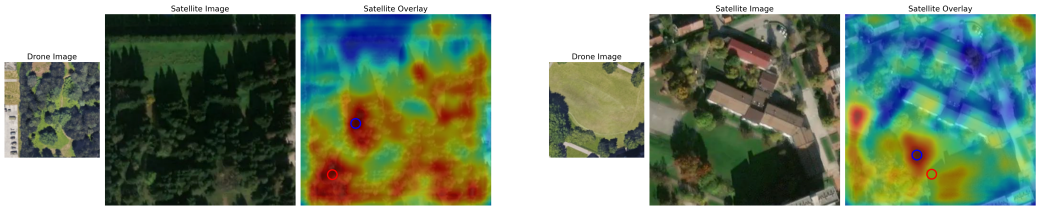
Slika 5.4: Leva slika prikazuje stanovanjsko hišo v stanovanjskem naselju z napako 6.93 m in RDS vrednostjo 0.90. Na desni vidimo parkirišče z monotono okolico. Napaka je 3.77 m, RDS pa 0.94.



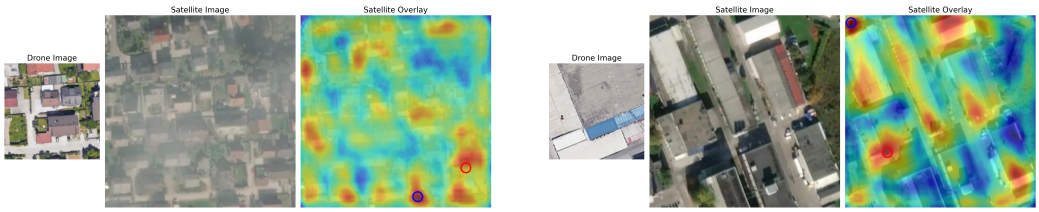
Slika 5.5: Leva slika prikazuje izsek avtoceste z napako 6.55 m in RDS vrednostjo 0.91. Desna slika okolico stanovanjske hiše v naselju z napako 5.73 m in RDS vrednostjo 0.90.

Slika 5.6: Na levi je industrijska stavba v industrijski coni z napako 3.99 m in RDS vrednostjo 0.95. Desna slika pa prikazuje travnik ob stanovanjskih hišah v naselju z napako 2.07 m in RDS vrednostjo 0.97.

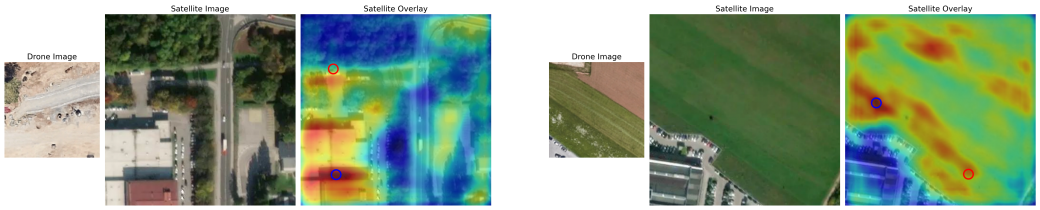
Primeri slabe lokalizacije



Slika 5.7: Obe sliki prikazujeta travnike in drevesa. Napaka na levi sliki je 50.74 m z RDS vrednostjo 0.31, na desni pa 30.16 m z RDS vrednostjo 0.42.



Slika 5.8: Leva slika prikazuje stanovanjsko hišo, kjer satelitsko sliko prekriva oblak, otežujoč izvleček značilnosti. Napaka je 54.24 m in RDS je 0.30. Desna slika prikazuje industrijsko stavbo v monotoni industrijski coni z napako 150.47 m in RDS vrednostjo 0.10.



Slika 5.9: Leva slika prikazuje gradbišče prisotno na sliki iz brezpilotnega letalnika, odsotno na satelitski sliki z napako 169.43 m in RDS vrednostjo 0.09. Desna slika pa polje v monotoni okolici z napako 155.47 m in RDS vrednostjo 0.10.

5.2 Učenje s stratificiranim vzorčenjem

Stratificirano vzorčenje igra ključno vlogo pri ocenjevanju kakovosti modela v heterogenih podatkovnih zbirkah. V tem podpoglavju bomo raziskali kaj so prednosti in slabosti stratificiranega vzorčenja.

5.2.1 Stratificirano vzorčenje

Stratificirano vzorčenje zagotavlja, da so vse podkategorije v populaciji ustrezno zastopane v vzorcu, kar izboljša natančnost ocen. Vendar pa model morda ni pripravljen na povsem nove, nevidene podatke in se lahko preveč prilagodi specifični distribuciji podatkov.

5.2.2 Rezultati

Način	HANN_{val}	$\text{RDS}_{\text{train}}$	RDS_{val}	$\overline{\Delta_m}$ [m]
Originalno učenje	8.49	0.893	0.709	43.42
Učenje s strat. vzorčenjem	3.17	0.750	0.731	17.89

Tabela 5.3: Rezultati ob uporabi stratificiranega vzorčenja, kjer je $\overline{\Delta_m}$ povprečna napaka v metrih.

Kriterijska funkcija	< 10m [%]	< 20m [%]	< 50m [%]	< 100m [%]
Originalno učenje	65.22	71.66	75.87	81.45
Učenje s strat. vzorčenjem	71.11	81.18	87.97	95.35

Tabela 5.4: Rezultati ob uporabi stratificiranega vzorčenja, kjer je prikazan odstotek primerov z napako manjšo od določene razdalje.

Iz rezultatov 5.3 in 5.4 je razvidno, da stratificirano vzorčenje pozitivno vpliva na natančnost modela. Model, naučen s to metodo, je dosegel rahlo višjo uspešnost na validacijski množici in boljšo generalizacijo. Kljub temu je treba upoštevati omejitve stratificiranega vzorčenja, kot so omejena generalizacija in težave pri podatkih, ki se močno razlikujejo od originalne distribucije.

5.3 Vpliv velikosti Hanningovega okna

Hanningova kriterijska funkcija je ključna za določanje uteži vzorcev v satelitskih slikah. Spreminjanje velikosti njenega okna neposredno vpliva na razporeditev in obliko uteži, kar ima posledično vpliv na kakovost rezultatov.

5.3.1 Dinamika različnih velikosti Hanningovih oken

Majhna velikost okna omejuje območje vzorcev, ki ga zajema. Takšna omejitev lahko zmanjša učinkovitost povratnega razširjanja med učenjem modela, saj kriterijska funkcija nima dovolj širokega vpliva na celotno mrežo. Nasprotje predstavlja preveliko okno, ki zajema široko paleto vzorcev. Kljub širšemu zajemu, lahko detajli v sliki postanejo manj opazni, kar zmanjšuje natančnost predikcij.

5.3.2 Eksperimentalni rezultati

V eksperimentu smo vsak model posebej naučili z različnimi velikostmi Hanningovega okna, da bi ocenili vpliv velikosti oken na modelovo natančnost. Za testiranje smo uporabili kombinacijo slike iz brezpilotnega letalnika in satelitske slike, zagotavljajoč enake vhodne podatke za vse modele. Referenčni sliki za testiranje sta prikazani na sliki 5.10.

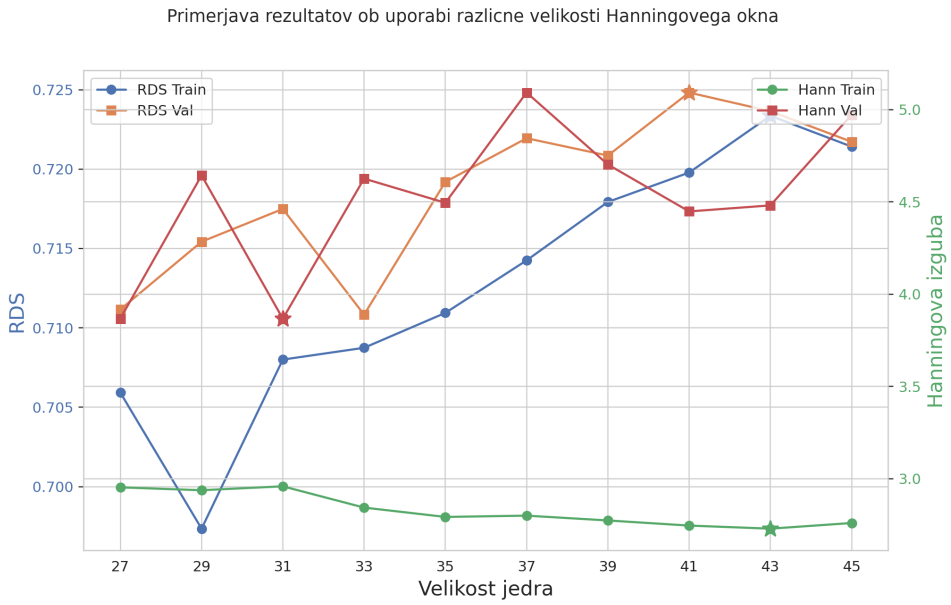
Slika 5.10: Primer referenčnih slik, ki smo jih uporabili za testiranje.

Eksperimenti so bili izvedeni z različnimi velikostmi oken, da bi ugotovili njihov vpliv na uspešnost modela. Primeri so prikazani na slikah B.1. Podatki kažejo na optimalno ravnovesje med velikostjo oken in natančnostjo modela. Najboljše uspešnosti so bile dosežene z okni velikosti 31 in 33. Te velikosti sovpadajo s priporočili iz literature, kjer je bila optimalna velikost okna določena na 33 [22].

Čeprav imajo nekatera druga okna boljšo vrednost kriterijske funkcije (vidno v dodatku B), je analiza slik pokazala, da je najmanj šuma prav pri oknih velikosti 31 in 33. Okna, ki imajo manjše ali večje jedro od teh velikosti, začnejo vnašati šum na različnih lokacijah, kar vodi do zmanjšane natančnosti pri lokalizaciji. Ta šum lahko moti interpretacijo satelitskih slik

in zmanjša zanesljivost modela.

Zaključimo lahko, da je izbira prave velikosti Hanningovega okna ključna za doseganje optimalnih rezultatov.



Slika 5.11: Primerjava rezultatov ob uporabi različnih velikosti Hanningovega okna, na celotni validacijski množici.

5.4 Regularizacija

V tem podpoglavju raziskujemo tehniko izpuščanja nevronov kot sredstvo regularizacije v nevronskih mrežah. Ocenjujemo njen vpliv na model Twins, predstavimo pa tudi, kako različni parametri te tehnike vplivajo na uspešnost modela.

5.4.1 Izpuščanje nevronov

V svetu strojnega učenja je regularizacija ključna tehnika, ki se uporablja za preprečevanje prekomernega prilagajanja modela. Prekomerno prilagajanje

se pojavi, ko model postane preveč specifičen za učni nabor podatkov, kar pomeni, da se "preveč nauči"podrobnosti in šuma v učnih podatkih, kar vodi v slabo zmogljivost na novih, nevidenih podatkih. Med različnimi tehnikami regularizacije je "izpuščanje nevronov"(ang. dropout) ena izmed najbolj priljubljenih in učinkovitih metod za nevronske mreže. Koncept izpuščanja nevronov je preprost, a močan: med učenjem se določen odstotek nevronov v mreži naključno "izklopi"ali izpusti. To pomeni, da se med posameznim prehodom vnaprej določeni nevroni (in njihove povezave) začasno odstranijo iz mreže.

V modelu smo uporabili izpuščanje nevronov na več ključnih mestih:

1. **v modelu Twins**: izpuščanje nevronov je bilo uporabljeno za regulacijo različnih komponent modela, vključno z deli, kot so `attn_drop`, `proj_drop`, `head_drop`, `mlp_drop1`, `mlp_drop2` in `pos_drops`. Vsaka od teh komponent ima svojo specifično vlogo v arhitekturi modela. Z dodajanjem izpuščanja nevronov na te komponente smo dodali dodatno raven regularizacije, ki pomaga preprečiti prekomerno prilagajanje.

2. **v modulu za Združevanje značilnosti**: po vsaki konvolucijski operaciji v fuzijskem delu modela smo dodali izpuščanje nevronov. Konvolucijske plasti lahko hitro postanejo kompleksne in se prekomerno prilagodijo podatkom, zlasti ko delujejo na visokodimenzionalnih značilnostih. Z dodajanjem izpuščanja nevronov po vsaki konvolucijski plasti smo zmanjšali to tveganje in povečali robustnost modela.

Izpuščanje nevronov je ena izmed najbolj učinkovitih tehnik regularizacije za nevronske mreže. Z njegovo uporabo v modelu smo zagotovili, da je model bolj robusten in manj nagnjen k prekomernemu prilagajanju na učne podatke. V kompleksnih modelih, kot je Twins, kjer je veliko komponent, ki se lahko prekomerno prilagodijo podatkom, je uporaba izpuščanja nevronov ključnega pomena za zagotavljanje natančnih in zanesljivih rezultatov.

5.4.2 Rezultati

Parameter	**UAV**	**Satelit**	**Združevanje**
dropout	0.1	0.1	0.1
attn_drop	0.1	0.1	-
proj_drop	0.1	0.1	-
head_drop	0.1	0.1	-
mlp_drop1	0.1	0.1	-
mlp_drop2	0.1	0.1	-
pos_drops	0.05	0.05	-

Tabela 5.5: Parametri z uravnovešenim izpustom nevronov.

Parameter	**UAV**	**Satelit**	**Združevanje**
dropout	0.15	0.05	0.05
attn_drop	0.15	0.05	-
proj_drop	0.15	0.05	-
head_drop	0.15	0.05	-
mlp_drop1	0.15	0.05	-
mlp_drop2	0.15	0.05	-
pos_drops	0.1	0.05	-

Tabela 5.6: Parametri z neuravnovešenim izpuščanjem nevronov.

Način	HANN_{val}	$\text{RDS}_{\text{train}}$	RDS_{val}	$\overline{\Delta_m}$ [m]
Brez izpuščanja	8.49	0.893	0.709	43.42
Uravnovešeno izpuščanje	5.49	0.725	0.690	21.67
Neuravnovešeno izpuščanje	5.42	0.725	0.719	18.11

Tabela 5.7: Rezultati ob uporabi različnih izpuščanj nevronov, kjer je $\overline{\Delta_m}$ povprečna napaka v metrih.

5.5 Uporaba prednaučene mreže

V tem podpoglavju raziskujemo vpliv uporabe prednaučene mreže Twins za izluščenje značilnosti pred združitvijo v modulu za združevanje značilnosti. Cilj je oceniti, kako uporaba prednaučene mreže vpliva na uspešnost modela WAMF-FPI.

5.5.1 Prednaučena mreža za izluščenje značilnosti

Uporaba prednaučenih modelov v strojnem učenju omogoča izkoristek že obstoječega znanja modela za pospešitev in izboljšanje učenja na novem naboru podatkov. Zlasti v kontekstu globokih nevronskih mrež so prednaučeni modeli dragoceni, saj lahko pomagajo modelom hitreje konvergirati in v nekaterih primerih doseči boljše rezultate.

5.5.2 Rezultati

Naši rezultati 5.8 in 5.9 kažejo , da je uporaba prednaučene mreže Twins privedla do boljših rezultatov v primerjavi z modelom, ki ni uporabljal prednaučene mreže. To poudarja prednost prenosa znanja iz prednaučenih modelov na specifične naloge.

Način	HANN_{val}	$\text{RDS}_{\text{train}}$	RDS_{val}	$\overline{\Delta_m}$ [m]
Prednaučena mreža	8.49	0.893	0.709	43.42
Brez prednaučene mreže	8.21	0.627	0.630	60.23

Tabela 5.8: Rezultati ob uporabi prednaučene mreže, kjer je $\overline{\Delta_m}$ povprečna napaka v metrih.

Uporaba prednaučene mreže Twins je omogočila boljše zajemanje in interpretacijo značilnosti iz našega nabora podatkov, kar je vodilo k izboljšanim rezultatom. To potrjuje, da so prednaučeni modeli lahko zelo koristni v nekaterih scenarijih, še posebej, ko želimo izkoristiti že obstoječe znanje za izboljšanje uspešnosti na novih nalogah.

Kriterijska funkcija	< 10m [%]	< 20m [%]	< 50m [%]	< 100m [%]
Prednaučena mreža	65.22	71.66	75.87	81.45
Brez prednaučene mreže	34.98	34.45	53.43	64.79

Tabela 5.9: Rezultati ob uporabi prednaučene mreže, kjer je prikazan odstotek primerov z napako manjšo od določene razdalje.

Poglavje 6

Sklepne ugotovitve

Lokalizacija brezpilotnih letalnikov je ključnega pomena za njihovo avtonomno delovanje, zlasti v okoljih, kjer so tradicionalni navigacijski signali omejeni ali moteni. Da bi se spopadli s tem izzivom, smo se v tej raziskavi osredotočili na raziskovanje in implementacijo metode WAMF-FPI za lokalizacijo brezpilotnih letalnikov na podlagi slik.

V okviru naše raziskave smo razvili podatkovno zbirko, ki zajema slike 11 evropskih mest s pogledom od zgoraj navzdol. Poleg tega smo preizkusili različne aspekte metode WAMF-FPI, vključno z regularizacijo z izpustom nevronov, različnimi vrednostmi Hanningovega okna in primerjavo med uporabo predhodno naučene mreže za izluščenje značilnosti in mreže, ki ni bila predhodno naučena. Naša implementacija WAMF-FPI je pokazala obetavne rezultate, pri čemer je Hanningova kriterijska funkcija izstopala kot najbolj učinkovita med vsemi preizkušenimi.

Kljub obetavni učinkovitosti metode smo identificirali več področij za izboljšave. Med njimi je raziskava različnih osnovnih arhitektur (ang. backbone) za izluščenje značilnosti, naprednejše združevanje značilnosti, bolj usmerjeno učenje z uporabo pomožnih izgub (ang. auxiliary losses) ter uporaba segmentacije. Poleg tega smo prepoznali potrebo po bolj napredni primerjavi značilnosti drona in satelita, pri čemer bi lahko uporabili pristope iz vizualnega sledenja, saj je to soroden problem.

V prihodnosti nameravamo razširiti našo podatkovno zbirko z večjim številom mest, slikami iz različnih višin, različnih kotov in pogledov ter z realnimi podatki. Prav tako nameravamo raziskati uporabo naprednejših tehnik za združevanje značilnosti in boljše usmerjeno učenje.

Naša raziskava je postavila trdne temelje za nadaljnji razvoj in implementacijo metode v realnih sistemih brezpilotnih letalnikov. Naslednji koraki bi vključevali nadaljnje optimizacije modela, razširitev podatkovnih zbirk, uporabo časovne informacije, vzpostavitev povratne zanke (ang. feedback loop) in končno implementacijo na dejanskih brezpilotnih letalnikih.

Dodatek A

Primeri izračuna RDS

Za boljše razumevanje, kako se RDS izračuna in kaj nam predstavlja, si oglejmo tri različne primere.

Primer 1: Za $w = 400px$, $h = 400px$, $dx = 0px$, $dy = 0px$ in $k = 10$ dobimo:

$$RDS_1 = e^{-10\times\frac{\sqrt{\left(\frac{0}{400}\right)^2+\left(\frac{0}{400}\right)^2}}{2}} = 1. \tag{A.1}$$

Ker sta dx in dy oba 0, je RDS za ta primer enak 1 (kar pomeni, da je napovedana pozicija točno na dejanski poziciji).

Primer 2: Za $w = 400px$, $h = 400px$, $dx = 2px$, $dy = 0px$ in $k = 10$ dobimo:

$$RDS_2 = e^{-10\times\frac{\sqrt{\left(\frac{2}{400}\right)^2+\left(\frac{0}{400}\right)^2}}{2}} = 0.975. \tag{A.2}$$

Tukaj je napovedana pozicija rahlo odmaknjena samo v vodoravni smeri za dva piksla. $RDS = 0.975$ kaže na minimalno odstopanje napovedane pozicije od dejanske.

Primer 3: Za $w = 400px$, $h = 400px$, $dx = 10px$, $dy = 14px$ in $k = 10$ dobimo:

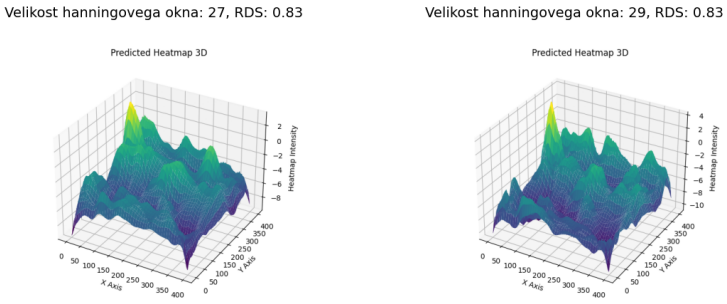
$$RDS_3 = e^{-10\times\frac{\sqrt{\left(\frac{10}{400}\right)^2+\left(\frac{14}{400}\right)^2}}{2}} = 0.806. \tag{A.3}$$

V tem primeru je napovedana pozicija odmaknjena tako v vodoravni kot navpični smeri. RDS vrednost 0.806 kaže na večjo relativno napako v primerjavi s prejšnjim primerom.

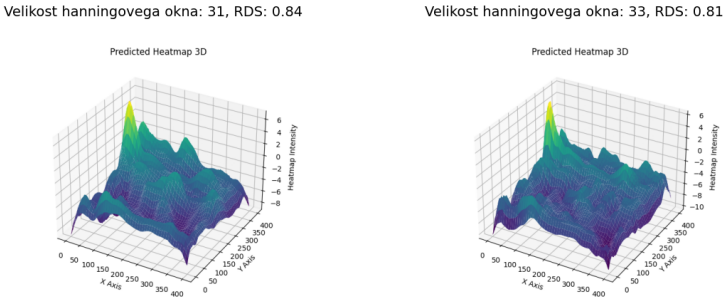
RDS metrika nam omogoča kvantitativno oceno natančnosti napovedane pozicije v primerjavi z dejansko pozicijo. Višja kot je vrednost RDS, bližje je napovedana točka dejanski točki. V obratnem primeru, nižja kot je vrednost RDS, večja je napaka med napovedano in dejansko točko.

Dodatek B

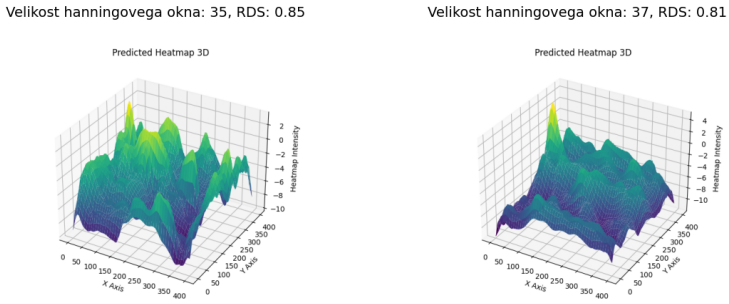
Primerjava toplotnih kart



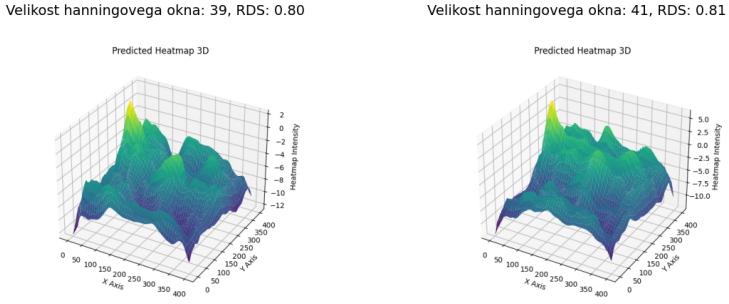
Slika B.1: Primerjava toplotnih map ob uporabi velikosti 27 in 29 Hanningovega okna.



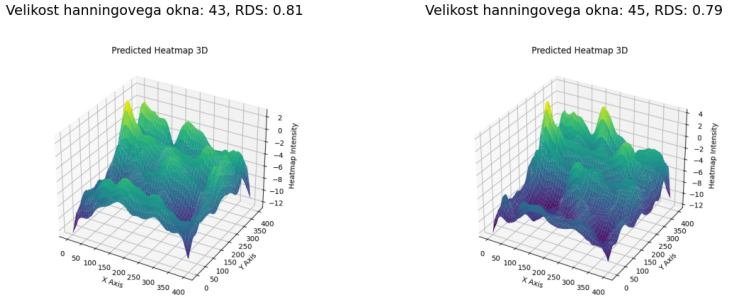
Slika B.2: Primerjava toplotnih map ob uporabi velikosti 31 in 33 Hanningovega okna.



Slika B.3: Primerjava toplotnih map ob uporabi velikosti 35 in 37 Hanningovega okna.



Slika B.4: Primerjava toplotnih map ob uporabi velikosti 39 in 41 Hanningovega okna.



Slika B.5: Primerjava toplotnih map ob uporabi velikosti 43 in 45 Hanningovega okna.

Literatura

[1] Dzmitry Bahdanau, Kyunghyun Cho in Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". V: *arXiv preprint arXiv:1409.0473* (2015).

[2] Mollie Bianchi in Timothy D. Barfoot. "UAV Localization Using Autoencoded Satellite Images". V: *arXiv preprint arXiv:2102.05692* (2021). URL: http://arxiv.org/abs/2102.05692v1.

[3] Xiangxiang Chu in sod. "Conditional positional encodings for vision transformers". V: *arXiv preprint arXiv:2102.10882* (2021).

[4] Xiangxiang Chu in sod. "Twins: Revisiting the design of spatial attention in vision transformers". V: *Advances in Neural Information Processing Systems* 34 (2021), str. 9355–9366.

[5] Ming Dai in sod. "Finding Point with Image: An End-to-End Benchmark for Vision-based UAV Localization". V: *arXiv preprint arXiv:2208.06561* (2022).

[6] Sounak Dey in sod. "SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification". V: *arXiv preprint arXiv:1707.02131* (2017). URL: http://arxiv.org/abs/1707.02131v2.

[7] Alexey Dosovitskiy in sod. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". V: *arXiv preprint arXiv:2010.11929* (2020).

[8] A. Zamir F. Castaldo in sod. "Semantic cross-view matching". V: *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)* (2015). DOI: https://doi.org/10.1109/iccvw.2015.137.

[9] Google. *Google Earth Studio*. Dostopano: 30.8.2023. 2021. URL: https://www.google.com/earth/studio/.

[10] Liu Liu in Hongdong Li. "Lending Orientation to Neural Networks for Cross-view Geo-localization". V: *arXiv preprint arXiv:1903.12351* (2019). URL: http://arxiv.org/abs/1903.12351v1.

[11] Liu Liu in Hongdong Li. "Predicting Ground-Level Scene Layout from Aerial Imagery". V: *arXiv preprint arXiv:1612.02709* (2016). URL: http://arxiv.org/abs/1612.02709v1.

[12] Ze Liu in sod. "Swin transformer: Hierarchical vision transformer using shifted windows". V: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, str. 10012–10022.

[13] H. S. Sawhney M. Bansal in sod. "Geo-localization of street views with aerial image databases". V: *Proceedings of the 19th ACM international conference on Multimedia - MM '11* (2011). DOI: https://doi.org/10.1145/2072298.2071954.

[14] M. Feng S. Hu in sod. "CVM-net: Cross-view matching network for image-based ground-to-aerial geo-localization". V: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018). DOI: https://doi.org/10.1109/cvpr.2018.00758.

[15] R. Souvenir S. Workman in N. Jacobs. "Wide-area image geolocalization with aerial reference imagery". V: *2015 IEEE International Conference on Computer Vision (ICCV)* (2015). DOI: https://doi.org/10.1109/cvpr.2005.202.

[16] T. Senlet in A. Elgammal. "A framework for global vehicle localization using stereo images and satellite and road maps". V: *2011 IEEE International Conference on Computer Vision Workshops*

(ICCV Workshops) (2011). DOI: https://doi.org/10.1109/iccvw.2011.6130498.

[17] S. Belongie T.-Y. Lin in J. Hays. "Cross-view image geolocalization". V: *2013 IEEE Conference on Computer Vision and Pattern Recognition* (2013). DOI: https://doi.org/10.1109/cvpr.2013.120.

[18] Y. Cui T.-Y. Lin in sod. "Learning deep representations for ground-to-aerial geolocalization". V: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). DOI: https://doi.org/10.1109/cvpr.2015.7299135.

[19] *Teacher forcing*. https://en.wikipedia.org/wiki/Teacher_forcing. Dostopano: 30.8.2023. 2023.

[20] Hugo Touvron in sod. "Training data-efficient image transformers & distillation through attention". V: *arXiv preprint arXiv:2012.12877* (2020). URL: http://arxiv.org/abs/2012.12877v2.

[21] Ashish Vaswani in sod. "Attention is all you need". V: *Advances in neural information processing systems* 30 (2017).

[22] Guirong Wang in sod. "WAMF-FPI: A Weight-Adaptive Multi-Feature Fusion Network for UAV Localization". V: *Remote Sensing* 15.4 (2023), str. 910.

[23] Wenhai Wang in sod. "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions". V: *arXiv preprint arXiv:2102.12122* (2021).

[24] S. Workman in N. Jacobs. "On the location dependence of convolutional neural network features". V: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2015). DOI: https://doi.org/10.1109/cvprw.2015.7301385.

[25] X. Yu Y. Shi in sod. "Optimal feature transport for cross-view image geo-localization". V: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.07 (2020), str. 11 990–11 997. DOI: https://doi.org/10.1609/aaai.v34i07.6875.

[26] Zhedong Zheng in sod. "University-1652: A Multi-view Multi-source Benchmark for Drone-based Geo-localization". V: *arXiv preprint arXiv:2002.12186* (2020). URL: http://arxiv.org/abs/2002.12186v1.

[27] Sijie Zhu, Mubarak Shah in Chen Chen. "TransGeo: Transformer Is All You Need for Cross-view Image Geo-localization". V: *arXiv preprint arXiv:2204.00097* (2022). URL: http://arxiv.org/abs/2204.00097v1.