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 v različnih področjih, kot so agrikultura, kartiranje, vojaške operacije in še mnogo drugih. Kljub njihovi vsestranskosti pa se poraja ključno vprašanje: kako se droni obnašajo, ko izgubijo stik z GPS sistemom? Diplomska naloga se osredotoča na to tematiko in predlaga metodo za lokalizacijo brezpilotnih letalnikov ob izgubi GPS signala.

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 GPS system? This thesis focuses on this issue and proposes a method for localizing UAVs in the event of a GPS signal loss.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
UAV	unmanned aerial vehicle	brezpilotni letalnik
DNN	deep neural network	globoka nevronska mreza
CNN	convolutional neural network	konvolucijska nevronska mreza
RDS	relative distance score	ocena relativne razdalje
MSE	mean squared error	srednja kvadratna napaka
FPI	finding point in an image	iskanje tocke v sliki
WAMF	Weight-Adaptive Multi Feature fusion	Uteženo združevanje več značilk
PCPVT	Pyramid Vision Transformer with Conditional Positional encodings	Piramidni vizualni transformer s pogojnimi pozicijskimi kodiranji

Povzetek

Naslov: Lokalizacija brezpilotnih letalnikov

Avtor: Gašper Spagnolo

Diplomsko delo se osredotoča na uporabo naprednih nevronskih mrež za lokalizacijo brezpilotnih letalnikov s pomočjo satelitskih slik. V uvodu je predstavljena osnovna terminologija in koncepti s področja nevronskih mrež. V metodološkem delu so podrobno razložene konvolucijske nevronske mreže, transformerska arhitektura ter njeni derivati, kot sta Vision Transformer (ViT) in Piramidni vision transformer (PVT). Posebno pozornost je namenjena Siamski nevronski mreži, ki je ključna za primerjavo vzorcev med satelitskimi in dronskimi slikami. Podatkovna množica, uporabljena za učenje in testiranje, vključuje slike brezpilotnega letalnika in satelitske slike. V razdelku rezultatov so predstavljene ključne faze implementacije, učenja modela, izbire kriterijske funkcije ter različne optimizacijske strategije, kot je uporaba Stratificiranega Vzorčenja. Poudarjena je tudi vloga Hanningovega okna in regularizacijskih tehnik, kot je izpuščanje nevronov. Delo zaključuje s sklepnimi ugotovitvami, ki poudarjajo potencial in učinkovitost predlagane metode za natančno lokalizacijo brezpilotnih letalnikov.

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

Abstract

Title: UAV localization

Author: Gašper Spagnolo

The thesis focuses on the use of advanced neural networks for the localization of drones using satellite imagery. In the introduction, foundational terminology and concepts from the realm of neural networks are presented. The methodology section provides a detailed exploration of convolutional neural networks, transformer architecture, and its derivatives like the Vision Transformer (ViT) and the Pyramid Vision Transformer (PVT). Particular emphasis is given to the Siamese Neural Network, which is pivotal for comparing patterns between satellite and drone images. The dataset used for training and testing encompasses drone images and satellite photos. In the results section, key stages of implementation, model training, criterion function selection, and various optimization strategies, such as Stratified Sampling, are discussed. The role of the Hanning window and regularization techniques like dropout is also highlighted. The work concludes with final observations that underscore the potential and efficiency of the proposed method for precise drone localization.

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

Poglavje 1

Uvod

Brezpilotni letalniki so postali nepogrešljivo orodje v številnih sektorjih, od vojaških operacij do kmetijskega nadzora. Kljub njihovi široki uporabi pa se soočajo z izzivi pri avtonomni navigaciji, še posebej v okoljih, kjer je satelitski signal omejen ali nezanesljiv. V idealnih razmerah brezpilotni letalniki za svojo navigacijo uporabljajo GPS signale. Vendar pa lahko te signale motijo naravne in človeške ovire, kot so visoke stavbe, gorske formacije ali celo elektronske motnje. Izguba GPS signala 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 postalo nujno.

Tradicionalne metode prepoznavanja slik se v kontekstu lokalizacije brezpilotnih letalnikov zdijo kot obetavna alternativa [2, 15]. Vendar pa se ob njihovi uporabi pojavi cela paleta izzivov. Prvič, potrebujemo ogromno slikovno bazo, ki vključuje kompresirane satelitske slike območij, nad katerimi letalnik leti. Velikost in obseg te baze lahko povzročita precejšnje računske in pomnilniške zahteve, kar lahko oteži njeno integracijo v realnočasovnih sistemih, kot so brezpilotni letalniki. Drugič, vsaka posodobitev ali sprememba v osnovni nevronski mreži, ki se uporablja za prepoznavanje slik, zahteva ponovno obdelavo celotne slikovne baze. To ne le da je časovno potratno, ampak tudi poveča stroške, saj morajo vse slike ponovno potekati skozi po-

stopek predprocesiranja in razpoznavanja. Tretjič, ko brezpilotni letalnik zajame sliko za primerjavo, mora ta slika biti primerjana z vsako sliko v bazi, da se ugotovi najboljše ujemanje. V praksi to pomeni, ko imamo bazo sestavljeno iz milijonov slik, bo vsaka nova poizvedovalna slika potrebovala milijone primerjav, kar je zelo časovno potratno in računsko intenzivno.

TLE MANJKA RELATED WORK ODSTAVEK (nevem kaj je misljeno s tem)

V luči omejitev tradicionalnih metod prepoznavanja slik so raziskovalci razvili inovativen pristop, imenovan FPI (Finding Point with Image) [5]. Ta pristop se razlikuje od običajnih metod v smislu strukture in delovanja. 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 z drona posneta.

Za obdelavo vsake slike se uporablja posebna nevronska mreža, kjer vsaka mreža obdeluje svoj nabor podatkov brez deljenja uteži z drugo. Ko sta obe 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 dronsko in satelitsko sliko. Najvišja vrednost na toplotni karti natančno označuje mesto, kjer je dron posnel svojo sliko na večji satelitski sliki. Ta informacija se nato neposredno prevede v natančno lokalizacijo drona 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 dronov [5]. Ta pristop je integriral koncepte iz območja sledenja objektov za potrebe lokalizacije, ob soočanju z izzivi, ki jih predstavljajo razlike med slikami UAV in satelitskimi slikami. Z uporabo dveh različnih uteži za izvleček značilnosti UAV 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.

Tle manjka popravek, stevilne

WAMF-FPI je evolucija osnovne metode FPI in prinaša številne izboljšave pri procesiranju slik. Ključna prednost WAMF-FPI je njegova napredna piramidna struktura izluščenja značilk, ki omogoča bolj natančno in ra-

znoliko 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 v osnovni FPI metodi končna velikost značilk bila stisnjena na 16-krat manjšo od izvorne satelitske slike, v WAMF-FPI ta kompresijski faktor znaša samo 4-krat manjšo velikost. To omogoča WAMF-FPI-ju, da ohrani več informacij ter pridobi boljšo lokalizacijsko natančnost ob hkratnem zmanjšanju računske obremenitve.

Velik problem nam je predstavljala odsotnost javno dostopnep podatkovne zbirke. Zaradi te ovire smo se odločili za ustvarjanje lastne zbirke. To smo storili s pomočjo Google Earth Studio. Naša zbirka vključuje 11 večjih evropskih mest z raznoliko strukturo. Cilj izdelave te zbirke je bil zagotoviti raznolike podatke, ki bi lahko služili kot robustna osnova za testiranje in validacijo naše implementacije WAMF-FPI. Zato smo se odločili, da bomo v tej diplomski nalogi implementirali WAMF-FPI, kakor je opisano v izvornem članku, in preverili njegovo delovanje. Implementirali smo vse, kakor je v članku opisano, z namenom dobiti objektivno sliko o učinkovitosti in natančnosti metode. V tej diplomski nalogi bomo podrobno raziskali te tehnike, njihove prednosti in pomanjkljivosti ter potencialne aplikacije in izboljšave za prihodnost. Analizirali bomo njihovo učinkovitost in natančnost, s poudarkom na njihovi uporabi v realnih scenarijih lokalizacije brezpilotnih letalnikov. Naš cilj je ponuditi temeljito analizo metode WAMF-FPI in njenih aplikacij, da bi olajšali nadaljnji razvoj in uporabo v industriji brezpilotnih letalnikov.

Struktura naloge?

Poglavje 2

Metodologija

Tukaj sem izbrisal kar celoten odstavek

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 Vision Transformerju (ViT) in njegovi razširjeni verziji - Piramidnem Vision Transformerju (PVT). Posebno pozornost bomo posvetili tudi prilagojeni različici PVT, ki upošteva lokalne značilnosti, imenovani PCPVT. Zaključili bomo s siamskimi nevronskimi mrežami, ki predstavljajo ključno komponento pri primerjavi vzorcev. Te mreže so še posebej pomembne, ko želimo 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 želimo 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 (CNN – Convolutional Neural Networks) 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

Osnovna struktura CNN vključuje štiri glavne vrste plasti: konvolucijsko, aktivacijsko, združevalno (pooling) in polno povezano plast.

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.

2.1.2 Učenje in optimizacija

Mislim to je kljucno za vse metode, iguess je potrebno. Lahko tudi vrzemo ven

Glavni mehanizem učenja v CNN je vzvratno širjenje napak (backpropagation). To je iterativna metoda, ki minimizira kriterijsko funkcijo, da bi ugotovili optimalne uteži mreže. Za mnoge naloge v obdelavi slik je kvadratna napaka (MSE) izbrana kot kriterijska funkcija:

$$J(w) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.4}$$

Za optimizacijo uteži se uporablja gradientni sestop. Za vsako iteracijo se uteži posodobijo v smeri negativnega gradienta kriterijske funkcije:

$$w_{\text{novi}} = w_{\text{stari}} - \eta \nabla J(w_{\text{stari}}) \tag{2.5}$$

2.1.3 Značilnosti in prednosti

Konvolucijske mreže so sposobne avtomatskega zaznavanja hierarhičnih značilnosti. Na nižjih ravneh mreže se zaznavajo nizkonivojske značilnosti, kot so robovi in teksture, na višjih ravneh pa se zaznavajo kompleksnejše strukture, kot so oblike in objekti. Ta hierarhična značilnost je tisto, kar omogoča CNN, da doseže izjemno natančnost pri različnih nalogah obdelave slik.

2.2 Transformerska arhitektura

Dodaj uvod v podpoglavje

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 dolgokratni

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. To je bil velik napredek, ki je omogočil boljše obvladovanje dolgih zaporedij.

Tukaj sem izbrisal kar celoten odstavek

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:

$$h_n = \text{RNNenc}(f_n, hn - 1) \tag{2.6}$$

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

$$e_t = \text{RNNdec}(et - 1, h_{t-1}) \tag{2.7}$$

Opomba: pri učenju se za e_{t-1} pogosto uporablja dejanska vrednost iz

ciljnega zaporedja (ne izhod modela), kar je znano kot "teacher forcing"[11]. Izvorna zaporedja besed $\vec{F}$ se tako vnašajo v kodirnik, ki generira skrita stanja za vsako besedo:

$$\vec{H} = \text{Kodirnik}(\vec{F}) \tag{2.8}$$

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

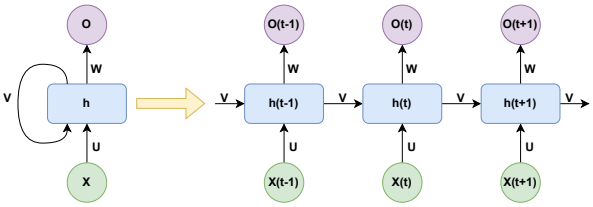
$$\vec{a}t = \text{Pozornost}(\vec{H}, et - 1) \tag{2.9}$$

Potem se ta vektor uporabi za napoved ciljne besede:

$$e_t = \text{Dekodirnik}(\vec{a}t, et - 1) \tag{2.10}$$

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.



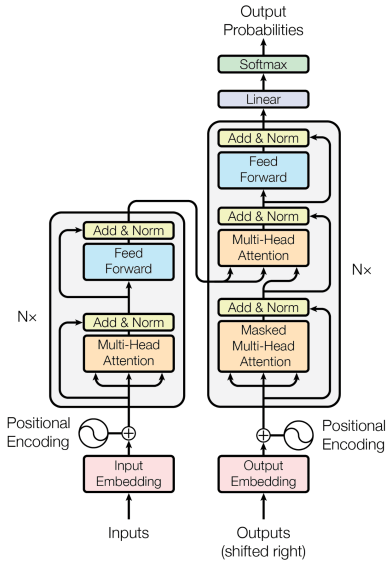
Slika 2.1: Skica RNN modela

2.3 Zgradba transformerja

V kontekstu strojnega prevajanja so avtorji v članku [12] o pozornosti predstavili novo vrsto arhitekture, ki se izogiba mnogim pastem modelov, ki temeljijo na RNN. Kljub vsem napredkom pri kodirnikih-dekodirnikih RNN,

ki smo jih obravnavali zgoraj, je ostalo dejstvo, da so RNN težko paralelizirani, ker zaporedno obdelujejo vhod. Ključna inovacija tega članka je, da so RNN in njihova skrita stanja v celoti nadomeščeni z operacijami na osnovi pozornosti, ki so v mnogih problematičnih režimih bolj učinkoviti.

Transformer model je model kodirnika-dekodirnika. Kodirnik sestavljajo N blokov na levi, dekodirnik pa N blokov na desni, vidno na sliki 2.2.



Slika 2.2: Izgled transformerja, iz članka "Attention is all you need"[12].

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 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. Maskiranje med učenjem nam omogoča, da posnemamo pogoje, s katerimi se bo model soočil med sklepanjem. Obstaja nekaj ključnih razlik od kodirnika - ena je, da so vhodi v prvo operacijo pozornosti v blokih dekodirnika maskirani, zato 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 pozornot (ang. Scaled Dot-Product Attention) se uporablja v vseh plasteh pozornosti v transformerju. Za zdaj bomo razčlenili matematiko za to operacijo, le da dobimo občutek, katera števila gredo kam. Kasneje se bomo osredotočili na njegove aplikacije v članku.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.11}$$

Scaled Dot-Product Attention je skoraj identičen Dot-Product Attentionu, omenjenem prej pri Luongu [1]. 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 [12].

V članku in predhodni literaturi 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 za izvedbo mora število ključev in vrednosti n ujemati, vendar se lahko število poizvedb m razlikuje. Prav tako se mora dimenzionalnost ključev in poizvedb ujemati, vendar se lahko dimenzionalnost vrednosti razlikuje.

Postopek izračuna utežena 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 (ang. Multi-Head Attention) je razširitev mehanizma pozornosti Scaled Dot-Product Attention. 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:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W_O \tag{2.12}$$

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

$$\text{head}_i = \text{Attention}(QW_{Qi}, KW_{Ki}, VW_{Vi}) \tag{2.13}$$

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 Maskiranje vhodov

En način za maskiranje vhodov je preprosto dodajanje matrike M k argumentu ki vsebuje 0 v spodnjem trikotniku in $-\infty$ povsod drugje:

$$\frac{1}{\sqrt{d_k}} Q' K'^T + M = \tag{2.14}$$

$$\begin{pmatrix} \frac{1}{\sqrt{d_k}}\vec{q}'_0 \cdot \vec{k}'_0 & \vec{q}'_0 \cdot \vec{k}'_1 & \cdots & \vec{q}'_0 \cdot \vec{k}'_n \\ \vec{q}'_1 \cdot \vec{k}'_0 & \vec{q}'_1 \cdot \vec{k}'_1 & \cdots & \vec{q}'_1 \cdot \vec{k}'_n \\ \vdots & \vdots & \ddots & \vdots \\ \vec{q}'_n \cdot \vec{k}'_0 & \vec{q}'_n \cdot \vec{k}'_1 & \cdots & \vec{q}'_n \cdot \vec{k}'_n \end{pmatrix} + \begin{pmatrix} 0 & -\infty & -\infty & \cdots & -\infty \\ -\infty & 0 & 0 & \cdots & -\infty \\ -\infty & \vdots & \vdots & \vdots & \vdots \\ -\infty & 0 & 0 & \cdots & 0 \\ -\infty & 0 & 0 & \cdots & 0 \end{pmatrix}$$

$$= \frac{1}{\sqrt{d_k}} \begin{pmatrix} \vec{q}'_0 \cdot \vec{k}'_0 & -\infty & -\infty & \cdots & -\infty \\ \vec{q}'_1 \cdot \vec{k}'_0 & \vec{q}'_1 \cdot \vec{k}'_1 & -\infty & \cdots & -\infty \\ \vdots & \vdots & \ddots & \vdots & \\ \vec{q}'_{n-1} \cdot \vec{k}'_0 & \vec{q}'_{n-1} \cdot \vec{k}'_1 & \vec{q}'_{n-1} \cdot \vec{k}'_2 & \cdots & -\infty \\ \vec{q}'_n \cdot \vec{k}'_0 & \vec{q}'_n \cdot \vec{k}'_1 & \vec{q}'_n \cdot \vec{k}'_2 & \cdots & \vec{q}'_n \cdot \vec{k}'_n \end{pmatrix}$$

Nato ima izvajanje softmaxa na vsaki vrstici učinek pošiljanja vseh $-\infty$ celic na 0, pri čemer ostanejo samo veljavni izrazi za pozornost.

2.3.4 Pozornost kodirnik-dekodirnik

Tretja in zadnja uporaba pozornosti v članku [12] 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) s strani Google-a [6]. 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. To omogoča uporabo istih tehnik samo-pozornosti, 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.

2.4.1 Arhitektura ViT

- Razdelitev slike na zaplate: Slika velikosti $H \times W \times C$ se razdeli na zaplate velikosti $P \times P$, kjer je H višina, W širina, C število barvnih kanalov in P velikost zaplate. To ustvari $(H \cdot W)/P^2$ zaplat. Vsaka zaplata se nato zravna v 1D vektor dolžine $P^2 \cdot C$.

- Linearne projekcije: Vsak 1D vektor x se prenese skozi enostaven linearni model (npr. polno povezano plast), da se pretvori v vektorski vložek. To se lahko zapiše kot:

$$z = Wx + b$$

  kjer sta W in b uteži in pristranskost linearne plasti.

- Dodajanje pozicijskih vložkov: Ker transformerji ne vsebujejo nobene inherentne informacije o relativni ali absolutni poziciji vložkov v zaporedju, se dodajo pozicijski vložki. To so enaki vektorji, ki se dodajo vložkom zaplat, da bi modelu dali nekaj informacij o tem, kje se zaplata nahaja v sliki. Če je z_i vložek i-te zaplate in p_i pozicijski vložek, potem je končni vložek e_i določen kot:

$$e_i = z_i + p_i$$

- Bloki transformerja: Zaporedje vložkov (zdaj z dodanimi pozicijskimi vložki) se nato prenese skozi več blokov transformerja. Ti bloki vsebujejo večglavo samopozornost in mreže feed-forward, ki omogočajo modelu, da se nauči, kako povezati različne dele slike. Večglava samopozornost se lahko zapiše kot:

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

kjer je $\text{head}_i = \text{Attention}(QWQi, KW_{Ki}, VW_{Vi})$, Q, K in V so poizvedbe, ključi in vrednosti, W_{Qi}, W_{Ki}, W_{Vi} in W_O so uteži, ki se naučijo, in Attention je funkcija samopozornosti.

- Klasifikacijska glava: Na koncu se uporabi klasifikacijska glava (ponavadi ena polno povezana plast), da se izračuna končna napoved za dano nalogo (npr. klasifikacija slik). To se lahko zapiše kot:

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

kjer sta W_1 in W_2 uteži polno povezanih plasti, e je vložek, ki izhaja iz transformerskih blokov, in ReLU in softmax sta aktivacijski funkciji.

2.5 Piramidni ViT (PVT)

Piramidni ViT (PVT) [14] 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 Transformer 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:

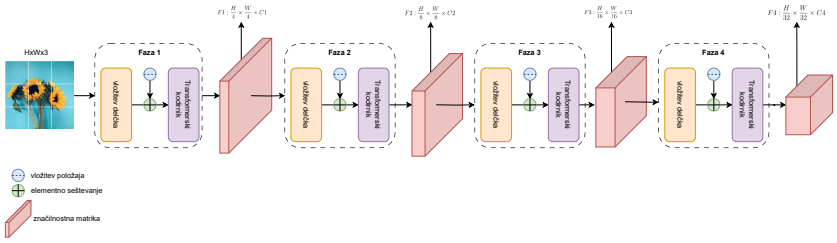
$$F1 = \frac{H}{4} \times \frac{W}{4} \times C1 \tag{2.15}$$

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 vision transformer 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.

$$CPE = f(PEG(E_1, E_2, ..., E_n)) \tag{2.16}$$

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 [8].

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 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:

$$D(f(x_1;\theta), f(x_2;\theta)) = |f(x_1;\theta) - f(x_2;\theta)|_2 \tag{2.17}$$

2.7.2 Učenje siamske mreže za primerjavo vzorcev

Da bi siamsko mrežo usposobili 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:

$$L(y, D(f(x_1;\theta), f(x_2;\theta))) = y \cdot \frac{1}{2}D^2 + (1-y) \cdot \frac{1}{2}\max(0, m-D)^2 \tag{2.18}$$

Kjer y označuje oznako podobnosti (1 za podobne in 0 za različne), m pa je prag, ki določa mejo med podobnimi in različnimi slikami.

2.7.3 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 raziskovalnem svetu je podatkovna množica ključnega pomena za razvoj, testiranje in validacijo modelov. Kljub pomembnosti modela WAMF-FPI avtorji niso javno delili originalne podatkovne množice. To je postavilo pred nas in vse druge, ki bi jih ta metoda lahko zanimala, izziv pri zbiranju primerne podatkovne osnove za analizo. Za doseganje konsistentnosti in kakovosti rezultatov smo se odločili samostojno kreirati in kurirati našo lastno podatkovno množico, ki odraža realne pogoje in scenarije uporabe.

Podatkovna množica, ki smo jo oblikovali, temelji na dveh glavnih virih vizualnih podatkov. Prvi vir predstavljajo slike, pridobljene z brezpilotnimi letalniki. Te slike so bile pridobljene preko orodja Google Earth Studio [7], ki omogoča natančno in realno reprezentacijo terenskih značilnosti iz ptičje perspektive. Te slike nudijo bogate detajle in so ključne za razumevanje fine strukture terena.

Drugi vir podatkov predstavljajo satelitske slike, pridobljene preko Mapbox API [9]. Satelitske slike prinašajo širši pogled na regijo in omogočajo razumevanje večjih geografskih in prostorskih vzorcev. V kombinaciji z dronskimi slikami te slike nudijo celovito sliko terena z različnih višin in ločljivosti.

Skupno naša podatkovna množica vključuje več kot 11.000 slik posnetih z brezpilotnim letalnikom in njihovih pripadajočih satelitskih slik. Ta obsežna zbirka podatkov nam omogoča, da model WAMF-FPI testiramo in

validiramo v številnih različnih scenarijih in pogojih, s čimer zagotavljamo njegovo robustnost in splošno uporabnost.

Za primerjavo, v članku so uporabili podatkovno množico UL14, ki vključuje 6.768 slik za učenje in 2.331 slik za validacijo [5, 13]. Ta množica predvsem vsebuje slike stavb večjih kitajskih univerz. V nasprotju z UL14, naša podatkovna množica ne zajema samo stavb, temveč tudi parke, zelene površine in druge značilnosti terena, kar prinaša širši spekter značilnosti za analizo, ter predstavlja bolj realne okoliščine.

3.1 Slike brezpilotnega letalnika

Nabor podatkov, ki ga predstavljamo, je bil zasnovan z namenom raziskovanja in analize dronov 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. 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 [7].

Mesta, vključena v učni nabor podatkov, so:

- **Maribor:** Nadmorska višina: 272m, Višina drona: 150m, Skupaj: 422m nad morsko gladino.

- **Trst:** Nadmorska višina: 23m, Višina drona: 150m, Skupaj: 173m nad morsko gladino.
- **Zagreb:** Nadmorska višina: 158m, Višina drona: 150m, Skupaj: 308m nad morsko gladino.
- **Gradec:** Nadmorska višina: 353m, Višina drona: 150m, Skupaj: 503m nad morsko gladino.
- **Celovec:** Nadmorska višina: 446m, Višina drona: 150m, Skupaj: 596m nad morsko gladino.
- **Videm:** Nadmorska višina: 113m, Višina drona: 150m, Skupaj: 263m nad morsko gladino.
- **Pula:** Nadmorska višina: 17m, Višina drona: 150m, Skupaj: 167m nad morsko gladino.
- **Pordenone:** Nadmorska višina: 24m, Višina drona: 150m, Skupaj: 174m nad morsko gladino.
- **Szombathely:** Nadmorska višina: 212m, Višina drona: 150m, Skupaj: 362m nad morsko gladino.
- **Benetke:** Nadmorska višina: -1m, Višina drona: 150m, Skupaj: 149m nad morsko gladino.

Dodatno je bil v nabor dodan tudi testni nabor podatkov za Ljubljano, ki vključuje 1.000 slik. Vsaka slika je opremljena z oznakami lokacije kamere v sistemu ECEF. Sistem ECEF (Earth Centered, Earth Fixed) je globalni koordinatni sistem z izhodiščem v središču Zemlje.

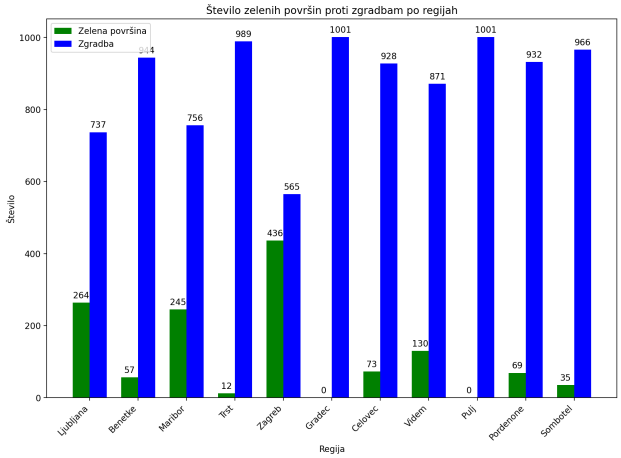
A tuki je misljeno da za vsako mesto dam primer slike?

Na Sliki 3.2 je prikazana vizualna 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

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

urbanizacije. Te razlike so ključnega pomena pri razumevanju izzivov, s katerimi se srečujejo brezpilotni letalniki pri lokalizaciji in navigaciji v različnih mestnih okoljih.

Nekatera mesta, kot sta Gradec in Pula, kažejo višjo stopnjo urbanizacije z minimalno prisotnostjo zelenih površin. To pomeni, da bodo droni 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 dronov, saj se morajo prilagajati različnim scenarijem in oviram.



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

Na sliki 3.3 je prikazan primer slike brezpilotnega letalnika, zajete v Ljubljani.

Slika 3.3: Primer slike brezpilotnega letalnika.

3.2 Satelitske slike

Za vsako dronsko sliko smo poiskali ustrezen satelitski "tile"ali ploščico. Ta korak je bil ključnega pomena, saj je zagotovil, da so satelitske slike popolnoma usklajene z dronskimi slikami v smislu geografske lokacije. Ko smo identificirali ustrezno satelitsko ploščico, smo jo prenesli neposredno iz Mapbox API-ja, priznanega vira za visokokakovostne satelitske slike. Da bi zagotovili dodatno globino in kontekst za vsako lokacijo, nismo prenesli samo osrednje ploščice, temveč tudi vse njene sosednje ploščice. Te sosednje ploščice smo nato združili z osrednjo ploščico za ustvarjanje enotne TIFF datoteke.

Ko govorimo o ploščicah v kontekstu kartografije in GIS (Geografski informacijski sistem), se običajno nanašamo na kvadratne segmente, ki pokrivajo Zemljo in se uporabljajo za hitrejše in učinkovitejše prikazovanje zemljevidov na spletu. Sistem ploščic je zelo priljubljen v spletnih kartografskih aplikacijah, kot je Google Maps.

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

- 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 ploscicne koordinate:

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

Na sliki 3.4 je prikazan primer pripadajoče satelitske slike za dronsko sliko.

Slika 3.4: Primer pripadajoče satelitske slike za dronsko sliko.

Tukaj nimam idej kako bi dodal vec primerov in kako bi polepsal vizualizacijo.

Poglavje 4

Rezultati

V tem poglavju so podrobno predstavljeni rezultati, doseženi v različnih fazah implementacije in optimizacije modela WAMF-FPI. Naš izhodiščni korak je bil zagotoviti stabilno osnovo, kar smo dosegli z implementacijo modela skladno z metodologijo, opisano v izvirnem članku. Ta pristop nam je zagotovil referenčno točko, od katere smo izvajali nadaljnje optimizacije in izboljšave.

Med optimizacijo modela smo se posvetili iskanju optimalne kriterijske funkcije. Da bi bolje razumeli, katera funkcija bi lahko prinesla najboljše rezultate v našem primeru, smo izvedli serijo eksperimentov z različnimi funkcijami ter jih evalvirali glede na njihovo učinkovitost in zanesljivost. Kot naslednji korak smo preučili stratificirano vzorčenje, tehniko, ki bi lahko pripomogla k izboljšanju natančnosti in robustnosti modela z zagotavljanjem bolj uravnoteženega učnega nabora. Pregledali smo tudi vpliv Hanningovega okna ter analizirali, kako različne velikosti tega okna vplivajo na končne rezultate modela.

V zaključni fazi naših eksperimentov smo se osredotočili na regularizacijo, predvsem na tehniko izpuščanja nevronov. Zaradi kompleksnosti modelov globokega učenja smo želeli razumeti, kako bi taka regularizacija lahko pomagala preprečiti prekomerno prilagajanje ter izboljšala splošno učinkovitost modela. Vsako od teh področij je v nadaljevanju podrobno obravnavano, pri čemer so podane analize, interpretacije in ključne ugotovitve, ki smo jih pri-

dobili v tem procesu.

4.1 Implementacija

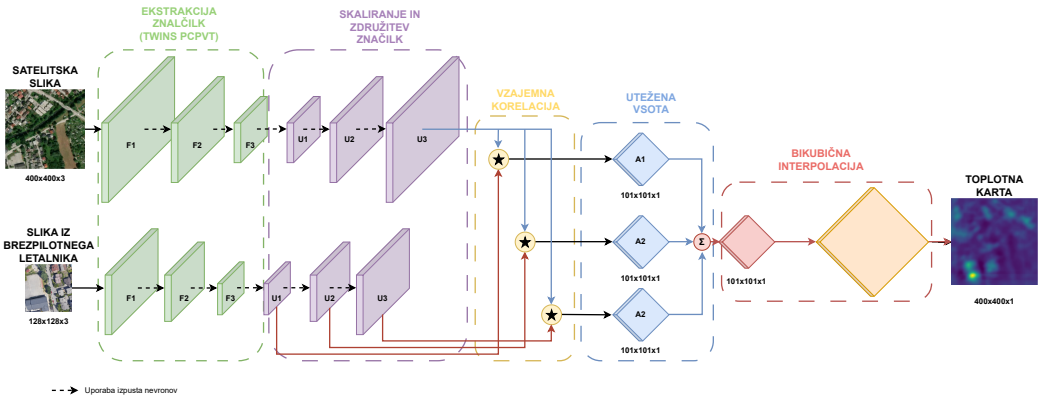
Sledenje objektov v okviru računalniškega vida običajno temelji na izračunu podobnosti med referenčno in iskalno podobo v trenutnem okviru. Medtem ko temeljna metoda za iskanje točk znotraj slike izhaja iz metodologije sledenja objektov, je prva v primerjavi z drugo bolj zapletena. To je posledica različnih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

Metoda iskanja točk uporablja satelitsko sliko kot referenčno in dronsko sliko kot poizvedbeno. Obe sliki – posneto z dronom in satelitsko sliko relevantnega območja – se nato prenesejo v end-to-end omrežje. Po obdelavi je rezultat toplotna karta, kjer točka z najvišjo vrednostjo predstavlja lokacijo drona, kot jo predvideva model. Lokacijo nato preslikamo na satelitsko sliko, pri čemer položaj drona določimo na podlagi geografske širine in dolžine, ki jih vsebuje satelitska slika. V FPI avtorji kot modul za izluščenje značilnosti uporabljajo dva Deit-S brez deljenih uteži za vertikalne poglede dronske in satelitske slike [5]. Ekstrahirane značilnosti nato uporabimo za izračun podobnosti in izdelavo toplotne karte. Lokacijo z najvišjo vrednostjo toplotne karte nato preslikamo na satelitsko sliko, da določimo lokacijo drona.

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 pozicioniranja.

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 dronskih 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 4.1 je prikazana skica arhitekture modela WAMF-FPI.



Slika 4.1: Skica arhitekture modela

4.1.1 Modul za izluščenje značilnosti

WAMF-FPI temelji na strukturi, ki je podobna siamski arhitekturi, vendar se od tradicionalnega sledenja objektom loči v ključnih aspektih. 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.

Konkretno, WAMF-FPI kot vhod uporablja satelitske slike dimenzij 400 $\times$ 400 $\times$ 3 in slike brezpilotnega letalnika dimenzij 128 $\times$ 128 $\times$ 3. Značilnosti obeh vrst slik so izluščene s pomočjo PCPVT-S.

Natančneje, 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 oziroma.

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

Po izluščanju informacij iz 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 s 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.

WAMF-FPI uporablja konvolucijske mreže za izluščenje značilnosti iz vhodnih slik. Konvolucija je ključna operacija, ki modelu omogoča, da prepoznava vzorce in značilnosti v slikah.

Prva faza obdelave v WAMF-FPI je uporaba konvolucijskega jedra velikosti ena, ki prilagodi kanalsko dimenzijo tri-stopnjske 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 se dodatno izluščene s pomočjo konvolucijskega jedra 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 mo-

delu 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:

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

Kjer je f^* kompleksno konjugirana funkcija f.

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 [13]. 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 piksla satelitske slike,
- h višina piksla 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.

4.2 Učenje modela

Model smo učili na računalniški konfiguraciji, opremljeni z visokozmogljivim procesorjem Intel(R) Xeon(R) CPU E5-2690 v3 @ 2.60GHz s 12 jedri. Dodatno je računalnik vseboval grafično kartico NVIDIA GeForce RTX 3060 s 12 GB pomnilnika, kar je omogočalo efektivno paralelizacijo in optimizacijo operacij, ki jih zahteva model med učenjem. Naš razvoj je temeljil na platformi Ubuntu z uporabo python knjižnice PyTorch [10].

Da bi povečali produktivnost in optimizirali proces, smo razvili avtomatizirane skripte, imenovane `push-ml-node` in `train-ml-node`. Prva skripta je bila uporabljena za sinhronizacijo virov z računalnikom, medtem ko je bila druga skripta uporabljena za zagon samega učenja.

Za dosego optimalnih rezultatov smo uporabili specifične hiperparametre in nastavitve:

Naprava: Učenje je potekalo na `cuda:0`, ki se nanaša na uporabo NVIDIA grafične kartice.

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 9, 13 in 15.

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

Epoh: Model je bil učen skozi 24 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.

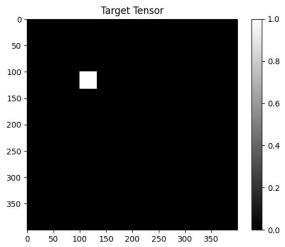
Za vsako iteracijo učenja smo iz vsake satelitske TIFF datoteke naključno izrezali regijo velikosti 400x400 pikslov. Ključnega pomena je bilo, da se je točka lokalizacije vedno nahajala nekje znotraj te izrezane regije. Ta metoda nam je zagotovila, da je bil model izpostavljen širokemu naboru scenarijev 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 dronskih slik in širino satelitskih slik, kar omogoča poglobljeno analizo in učinkovito učenje.

4.3 Izbira kriterjiske funkcije

Zanimalo nas je, kako se bo model obnesel, ko izbiramo različne kriterijske funkcije.

4.3.1 Hanningova kriterijska funkcija

V članku WAMF-FPI [13] 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 4.2: Primer vzorca, sredisce je tocka lokacije vzorca.

To je zato, ker je pomembnost središčnega položaja veliko večja kot pomembnost robovih položajev, kar v kontekstu satelitskih slik logično 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. Toda ker je število negativnih vzorcev običajno večje od števila pozitivnih vzorcev, postane utež negativnih vzorcev manjša. Da bi to popravili, se uvede hiperparameter, imenovan Negativna utež (NG), ki prilagodi utež negativnih vzorcev.

Hanningova funkcija:

$$\text{Hanning}(n) = \begin{cases} 0.5 + 0.5\cos\left(\frac{2\pi n}{M-1}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases} \tag{4.14}$$

Utezi primerov:

- Utez negativnih vzorcev:

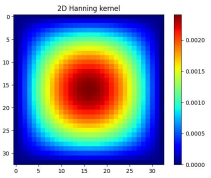
$$w_{pos} = NG/(NN(NW + 1))$$

- Utez 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 n



Slika 4.3: Normalizirano Hanningovo jedro

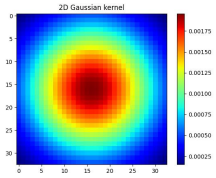
4.3.2 Gaussovo utezena srednja kvadratna napaka

Gaussova utežena srednja kvadratna napaka (Gaussian Weighted Mean Squared Error - GWMSE) je modificirana 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:

$$\text{Gauss}(n) = \begin{cases} \exp\left(-\frac{(n-\mu)^2}{2\sigma^2}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases} \tag{4.15}$$



Slika 4.4: Normalizirano Gaussovo jedro

4.3.3 Hanningovo utežena srednja kvadratna napaka

Hanningova utežena srednja kvadratna napaka (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:

$$\text{Hanning}(n) = \begin{cases} 0.5 + 0.5\cos\left(\frac{2\pi n}{M-1}\right) & \text{za } 0 \leq n \leq M-1 \\ 0 & \text{sicer} \end{cases} \tag{4.16}$$

4.3.4 Križno utežena srednja kvadratna napaka

Funkcija izgube križno utežena srednja kvadratna napaka (Cross-Weighted Mean Squared Error - CW-MSE) 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. "ne-resničnih"vzorcev). Končna funkcija izgube se izračuna kot utežena kombinacija srednjih kvadratnih napak za "resnične"in "ne-resnične"vzorce, pri čemer se uteži vzorcev različnih skupin prekrižajo. Ta pristop se formalno izraža z naslednjo enačbo:

$$\text{loss} = \frac{\text{true\_weight} \cdot N_{\text{true}} \cdot \text{MSE}_{\text{false}} + \text{false\_weight} \cdot N_{\text{false}} \cdot \text{MSE}_{\text{true}}}{N_{\text{all}}} \quad (4.17)$$

- 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*.

4.3.5 Primerjava rezultatov

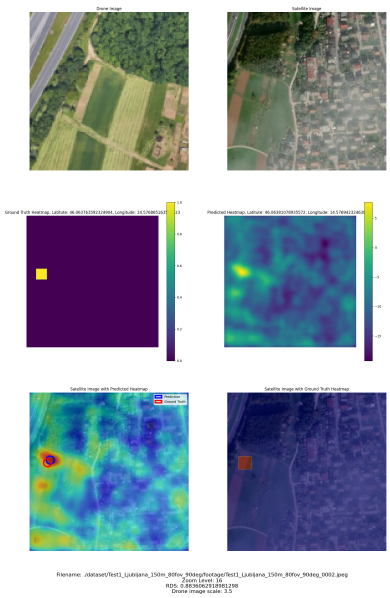
Kriterijska funkcija	vrednost	RDS_{train}	RDS_{val}
Hanningova kriterijska funkcija	8.49	0.893	0.709
Gaussovo utezena srednja kvadratna napaka	0.001	0.077	0.074
Hanningovo utežena srednja kvadratna napaka	4.04e-06	0.061	0.059
Križno utežena srednja kvadratna napaka	0.007	0.07	0.06

Tabela 4.1: Rezultati ob uporabi razlicnih kriterijskih funkij.

4.3.6 Analiza rezultatov

4.3.7 Hanningova kriterijska funkcija

Hanningova kriterijska funkcija, znana tudi po svoji značilnosti dodeljevanja uteži vzorcem glede na njihovo lokacijo, je v testiranju pokazala dobre rezultate. S skupno vrednostjo 8.49 in RDS_{train} vrednostjo 0.893 na učni množici se je izkazala kot izredno učinkovita za trening set. Čeprav je bila njena učinkovitost na validacijski množici, kjer je dosegla RDS_{val} vrednost 0.709, nekoliko nižja, so rezultati še vedno zelo obetavni. Primer je viden na sliki 4.5.



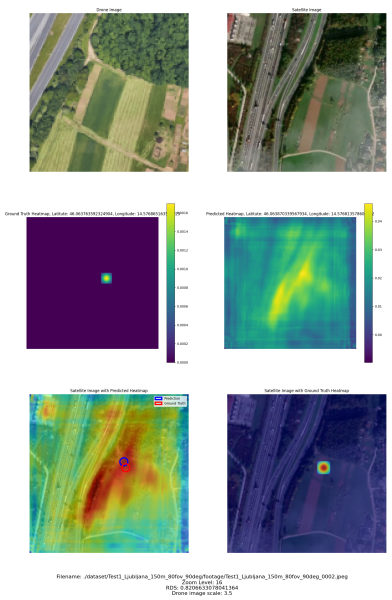
Slika 4.5: Primer izhoda ob uporabi Hanningove kriterijske funkcije

Ključna prednost Hanningove funkcije je v njeni zmožnosti prilagajanja uteži vzorcem glede na njihov položaj, kar se zdi še posebej primerno pri analizi satelitskih slik. V teh slikah je središčni položaj pogosto bistven, medtem ko robovi morda niso tako pomembni. To naravno prilagodljivost

Hanningove funkcije lahko opazimo v njenih rezultatih, ki jih dosegla v obravnavanem primeru.

4.3.8 Gaussovo utežena srednja kvadratna napaka

Čeprav je Gaussova utežena srednja kvadratna napaka prav tako zasnovana na principu dodeljevanja uteži glede na lokacijo vzorca, rezultati kažejo, da ne dosega enake uspešnosti kot Hanningova funkcija. Z RDS_{train} vrednostjo 0.077 na učni množici in RDS_{val} vrednostjo 0.74 na validacijski množici so njeni rezultati precej slabši v primerjavi s Hanningovo funkcijo. Primer je viden na sliki 4.6.

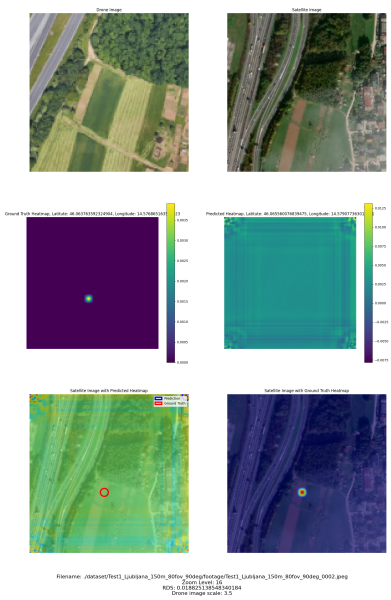


Slika 4.6: Primer izhoda ob uporabi Gaussovo utežene srednje kvadratne napake

Čeprav obe funkciji temeljita na podobnem principu, se zdi, da Hanningova funkcija bolje odraža posebnosti in značilnosti satelitskih slik.

4.3.9 Hanningovo utežena srednja kvadratna napaka

Pri tej funkciji se je izkazalo, da mreža ni dosegla želenih rezultatov. Namesto, da bi se mreža naučila prepoznati in interpretirati relevantne značilnosti satelitskih slik, se je večinoma učila šuma. Praktično, model se ni naučil nič koristnega, kar nakazuje, da Hanningovo utežena srednja kvadratna napaka morda ni primerna za to vrsto podatkov ali za uporabljeni model. Primer je viden na sliki 4.7.

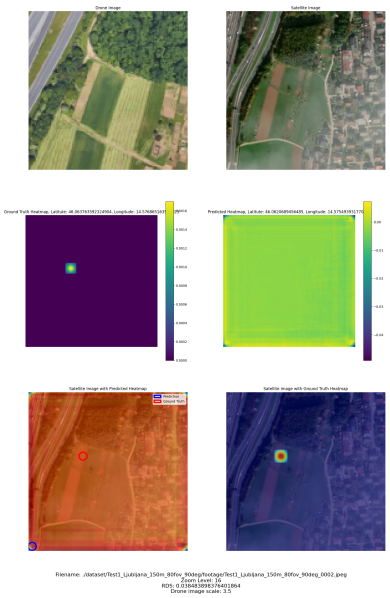


Slika 4.7: Primer izhoda ob uporabi Hanningove utežene srednje kvadratne napake

4.3.10 Križno utežena srednja kvadratna napaka

Podobno kot pri Hanningovi uteženi srednji kvadratni napaki se je tudi pri Križno uteženi srednji kvadratni napaki pokazalo, da mreža večinoma prepoznava in se uči šuma. Rezultati so bili nezadovoljivi in kažejo na to, da

ta funkcija ni najbolj primerna za analizo satelitskih slik s tem pristopom. Primer je viden na sliki 4.8.



Slika 4.8: Primer izhoda ob uporabi Križno uteženе srednje kvadratne napake

Zaključek: Hanningova kriterijska funkcija se je v obravnavanem primeru izkazala kot najbolj učinkovita. Njena edinstvena sposobnost prilagajanja uteži glede na lokacijo vzorca se zdi še posebej primerna za obravnavo satelitskih slik, kar je morda razlog za njeno premoč nad ostalimi obravnavanimi funkcijami izgube.

4.4 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.

4.4.1 Stratificirano vzorčenje

Stratificirano vzorčenje je metoda vzorčenja, pri kateri se celoten nabor podatkov razdeli na ločene podskupine ali strate. Vsak stratum predstavlja določeno kategorijo ali razred v naboru podatkov. V kontekstu mest bi lahko vsako mesto predstavljalo svoj stratum. Namen stratificiranega vzorčenja je zagotoviti, da je vsak vzorec reprezentativen za celoten nabor podatkov.

Zakaj je stratificirano vzorčenje pomembno?

1. **Ohranjanje Distribucije**: Stratificirano vzorčenje zagotavlja, da se razmerje vzorcev v vsakem stratumu ohranja enako kot v celotnem naboru podatkov. To je še posebej pomembno, ko je distribucija podatkov v vsakem stratumu (v tem primeru mesto) ključnega pomena za analizo. Na primer, če želimo, da je naš vzorec reprezentativen za različna mesta, bi uporabili stratificirano vzorčenje, da zagotovimo, da so vsa mesta ustrezno zastopana.

2. **Natančnost**: Stratificirano vzorčenje lahko poveča natančnost ocen, saj zmanjšuje variabilnost znotraj vsakega strata. To pomeni, da so vzorci iz vsakega strata bolj homogeni, kar lahko vodi do natančnejših rezultatov.

Slabosti stratificiranega vzorčenja:

1. **Omejena Generalizacija**: Čeprav stratificirano vzorčenje zagotavlja, da so vse kategorije ali razredi v naboru podatkov ustrezno zastopani v vzorcu, to lahko pomeni, da model morda ni tako dobro pripravljen na povsem nove, nevidene podatke. Model je lahko optimiziran za specifično distribucijo podatkov, ki je bila uporabljena med ucenjem in validacijo.

2. **"In-Distribution"Validacija** Ker se vzorci za ucenje in validacijo izbirajo iz iste distribucije (stratificirane distribucije), model morda ne bo dobro deloval na "out-of-distribution"podatkih. To pomeni, da

čeprav model morda kaže visoko natančnost na validacijskem naboru, to ne zagotavlja, da bo enako dobro deloval na podatkih, ki se močno razlikujejo od originalne distribucije.

4.4.2 Rezultati

Nacin	Hanningova izguba	RDS_{train}	RDS_{val}
Originalno učenje	8.49	0.893	0.709
Učenje s stratificiranim vzorčenjem	3.17	0.750	0.731

Tabela 4.2: Rezultati ob uporabi stratificiranega uzorcenja.

Iz rezultatov 4.2 je razvidno, da je uporaba stratificiranega vzorčenja pozitivno vplivala na rezultate.

Za boljše razumevanje uspešnosti modelov je ključno upoštevati tudi njihovo zmogljivost na validacijskih naborih podatkov. To je še posebej pomembno, saj nam validacija daje vpogled v to, kako dobro model predvideva rezultate na nevidenih podatkih. Če primerjamo rezultate RDS_{val} med obema pristopoma, opazimo, da je model, ki je bil naučen s stratificiranim vzorčenjem, dosegel rahlo višjo uspešnost (0.731) v primerjavi z modelom, ki je bil naučen s tradicionalno metodo "train-test split"(0.709). To kaže, da se je model, ki je bil naučen s stratificiranim vzorčenjem, nekoliko bolje spoprijel s generalizacijo na nevidenih podatkih. To dejstvo podkrepi tudi zmanjšana razlika med uspešnostjo na učni in validacijski množici v primeru stratificiranega vzorčenja. Večja konsistentnost rezultatov med učno in validacijsko množico je lahko pokazatelj, da model ni pretirano prilagojen in se lahko bolje generalizira na nove podatke. Torej, medtem ko je tradicionalna "train-test split"metoda dosegla višjo uspešnost na učni množici, se zdi, da stratificirano vzorčenje ponuja bolj zanesljive in stabilne rezultate na validacijski množici, kar je ključnega pomena za ocenjevanje realne zmogljivosti modela. V našem primeru se zdi, da stratificirano vzorčenje ponuja bolj robusten in stabilen model za obravnavane satelitske slike. Vendar pa je po-

membno upoštevati tudi omejitve stratificiranega vzorčenja, kot so omejena generalizacija in potencialne težave pri "out-of-distribution"podatkih.

4.5 Vpliv velikosti Hanningovega okna

Hanningovo okno, ključni element za določanje uteži vzorcev v satelitskih slikah, se prilagaja glede na svojo velikost. Spreminjanje velikosti okna neposredno vpliva na razporeditev in obliko uteži, kar ima posledično vpliv na kakovost rezultatov.

4.5.1 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.

4.5.2 Eksperimentalni rezultati

Pogoji eksperimenta so bili naslednji:

- Vsak model je bil posebej natreniran z različno velikostjo Hanningovega okna. To smo storili, da bi preverili vpliv različnih velikosti oken na uspešnost in natančnost modela.

- Po treniranju vsakega modela smo za testiranje uporabili enostaven primer kombinacije dronske in satelitske slike. To nam je omogočilo direktno primerjavo delovanja modelov na isti vhodni podatki in tako odpravilo morebitne nejasnosti ali napake, ki bi jih prinesli različni vhodni podatki.

- Referenčni sliki, ki smo ju uporabili za testiranje, je prikazana na sliki 4.9. Slika brezpilotnega letanika predstavlja tipičen primer slike, s katero se naš model srečuje v praksi in vsebuje različne značilnosti terena, ki so pomembne za lokalizacijo.

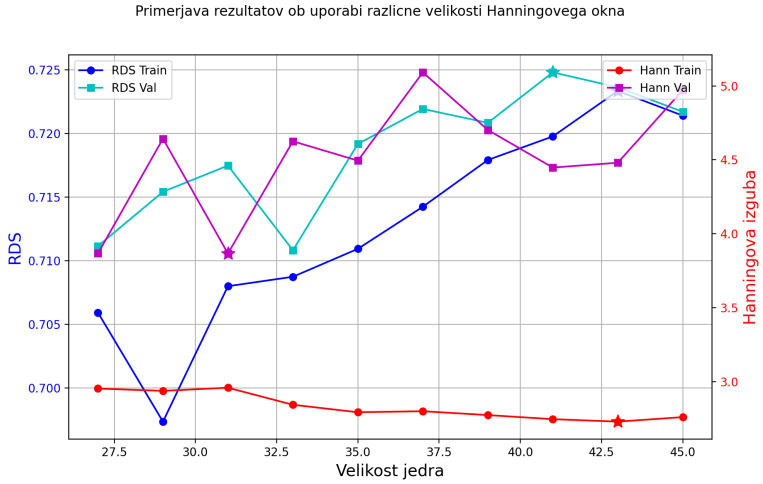
Slika 4.9: Primer referenčnih slik, ki smo jih uporabili za testiranje.

Eksperimenti so bili izvedeni z različnimi velikostmi oken, da bi ugotovili njihov vpliv na uspešnost modela. Primeri so prikazani na slikah 4.11 in 4.12. 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 [13].

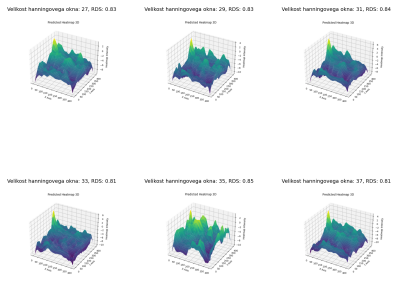
Čeprav imajo nekatera druga okna boljšo vrednost kriterijske funkcije (vidno na sliki 4.10), je analiza slik pokazala, da je najmanj šuma prav pri oknih velikosti 31 in 33. Okna, ki imajo manjše ali večje jedro od teh velikosti, začnejo vnašati šum na različnih lokacijah, kar vodi do zmanjšane natančnosti 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.

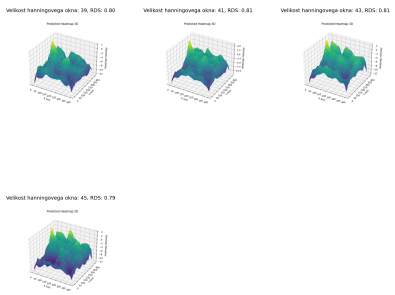
Popravi te slike!!!



Slika 4.10: Primerjava rezultatov ob uporabi različnih velikosti Hanningovega okna, na celotni validacijski množici.



Slika 4.11: Primerjava toplotnih map ob uporabi različnih velikosti Hanningovega okna.



Slika 4.12: Primerjava toplotnih map ob uporabi različnih velikosti Hanningovega okna.

4.6 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.

4.6.1 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 ucenjem se določen odstotek nevronov v mreži naključno "izklopi"ali izpusti. To pomeni, da se med posameznim prehodom naprej 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 povečal 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.

4.6.2 Rezultati

dodaj section kako se je mreza obnasala ko je bila pretrained in ko ni bila pretrained

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 4.3: Parametri z uravnovesenim 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 4.4: Parametri z neuravnovesenim izpustom nevronov.

Nacin	Hanningova izguba	RDS_{train}	RDS_{val}
Brez izpuscanja nevronov	8.49	0.893	0.709
Z uravnovesenim izpuscanjem nevronov	5.49	0.725	0.690
Z neuravnovesenim izpuscanjem nevronov	5.42	0.725	0.719

Tabela 4.5: Rezultati ob uporabi razlicnih izpustov.

Poglavje 5

Sklepne ugotovitve

Brezpilotni letalniki predstavljajo revolucionarni korak v tehnologiji, ki je našel svojo uporabo v številnih sektorjih, od vojaških operacij do kmetijskega nadzora. Kljub njihovi široki uporabi pa se soočajo z več ključnimi izzivi, zlasti na področju avtonomne navigacije. V diplomski nalogi smo se osredotočili na raziskovanje in implementacijo metode WAMF-FPI za lokalizacijo brezpilotnih letalnikov na podlagi slik. Spodaj so izpostavljene naše glavne ugotovitve in predlogi:

1. **Učinkovitost in natančnost metode**: Naša implementacija WAMF-FPI je pokazala, da je metoda izjemno obetavna. Ugotovili smo, da je sama arhitektura izredno učinkovita in da lahko zagotovi natančno lokalizacijo brezpilotnih letalnikov tudi v zahtevnih pogojih.

2. **Potencial za izboljšave**: Kljub izjemni učinkovitosti metode WAMF-FPI smo identificirali nekaj ključnih področij, kjer bi se lahko izvedle izboljšave. Eden od predlogov je uporaba močnejše nevronske mreže za izluščenje značilnosti. Morda bi bile konvolucijske nevronske mreže novejše generacije ali nekatere druge arhitekture bolj primerne za ta namen.

3. **Optimizacija združevanja značilnosti**: Med našo analizo smo opazili, da bi lahko del združevanja značilnosti optimizirali z uporabo me-

tode pozornosti, kar bi omogočilo še boljše ujemanje med dronskimi in satelitskimi slikami.

4. **Iskanje pripadajoče satelitske slike**: Ena od glavnih težav, s katerimi se metoda še vedno sooča, je identifikacija prave satelitske slike, ki ustreza dronski sliki. To predstavlja izziv, še posebej v bazah z milijoni slik, in je eno od področij, ki zahteva nadaljnje raziskave.

5. **Raziskava različnih kriterijskih funkcij**: V okviru naše analize smo preizkusili več kriterijskih funkcij, vključno s Hanningovo kriterijsko funkcijo, Gaussovo uteženo srednjo kvadratno napako, Hanningovo uteženo srednjo kvadratno napako ter križno uteženo srednjo kvadratno napako. Rezultati so pokazali, da je Hanningova kriterijska funkcija izstopala kot najbolj učinkovita med vsemi preizkušenimi. Te ugotovitve so v skladu z implementacijo in rezultati, predstavljenimi v izbranem članku.

6. **Regularizacija in računske obremenitve**: Ugotovili smo, da ima regularizacija v modelu z uporabo izpuščanja nevronov pomembno vlogo pri preprečevanju prenaučenja. Vendar pa je treba skrbno uravnotežiti med računskimi obremenitvami in natančnostjo modela.

7. **Praktična uporaba**: Naša največja ambicija za prihodnost je preizkusiti metodo WAMF-FPI na dejanskem brezpilotnem letalniku. S tem bi lahko dobili boljšo predstavo o realni učinkovitosti in uporabnosti metode v praksi.

Metoda WAMF-FPI predstavlja pomemben korak naprej v lokalizaciji brezpilotnih letalnikov, še posebej v okoljih, kjer je satelitski signal omejen ali nezanesljiv. Kljub obetavni učinkovitosti metode pa obstajajo še nekateri izzivi in priložnosti za izboljšave. 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 in končno implementacijo na dejanskih brezpilotnih letalnikih.

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

[7] Google. *Google Earth Studio*. Dostopano: 30.08.2023. 2021. URL: https://www.google.com/earth/studio/.

[8] 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.

[9] Mapbox. *Mapbox API Dokumentacija*. Dostopano: 30.08.2023. 2021. URL: https://www.mapbox.com/api-documentation/.

[10] PyTorch. *PyTorch: Odprtokodni knjižnica za strojno učenje*. Dostopano: 30.08.2023. 2021. URL: https://pytorch.org/.

[11] *Teacher forcing*. https://en.wikipedia.org/wiki/Teacher_forcing. Dostopano: 30.08.2023. 2023.

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

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

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

[15] 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.