Večmodalna detekcija ovir na vodni površini

Jon Muhovič

10. februar 2025

Povzetek

V okviru te seminarske naloge študenti rešujejo problem detekcije in segmentacije ovir na vodni površini z uporabo različnih modalitet. Namen naloge je, da študentje preizkusijo state-of-the-art metode na področju interpretacije slik in prednaučene modele doučijo z novimi podatki.

1 Uvod

Vaša naloga je usposobiti detekcijo in segmentacijo na podatkih, zajetih s kamero, montirano na čolnu. Vhodni podatki so dveh tipov: barvne slike s kamere ZED in termalne slike s kamere Device-ALab SmartIR384L. Barvne slike so bile ročno anotirane za problem semantične segmentacije, oznake tako vsebujejo štiri razrede: *water*, *sky*, *static obstacle*, *dynamic obstacle*. Razreda ovir predstavljata statične ovire, torej obalo in pomole ter dinamične ovire kot so ladje, boje in plavalci. Preko kalibracije ekstrinzičnih parametrov sistema so bile oznake nato prenešene na slike termalne kamere. Bolj podroben opis sistema se nahaja v članku [1].

2 Podatkovna baza

Baza je sestavljena iz dveh naborov slik (barvne in termalne), ustreznih semantičnih oznak in dveh datotek `.json`, ki vsebujeta oznake ovir z očrtanimi pravokotniki.

Bazo lahko najdete tukaj.

V python skripti `NMRV_viewer.py` se nahajajo funkcije, ki za vsako od štirih podnalog naložijo in prikažejo podatke in ustrezne anotacije. Te funkcije služijo hitremu pregledu podatnih podatkov, za uporabo v programskih ogrodjih (angl. framework) pa boste morali sami napisati ustrezen data loader in po potrebi preurediti očrtane pravokotnika (npr. v YOLO format). Barvne in termalne slike z istim imenom so bile zajete istočasno (to za vaše naloge sicer ni pomembno).

Med procesom anotiranja smo uporabili več razredov, ki jih je treba pri učenju ustrezno nasloviti. To so razredi *Water (hole)* (5), *Ignore* (6) in *Recording Boat* (7). (prim. `classes_map` v `NMRV_viewer.py`). Razreda z oznakama 6 in 7 lahko med učenjem ignorirate, razred 5 pa upoštevate kot del vode (t.j. kot razred 3). Način (ne)upoštevanja določenih pikslov med procesom učenja je odvisen od vaše implementacije (uporabite lahko recimo binarno masko pri računanju funkcije izgube).

3 Termalne slike

Termalne slike se v globokem učenju uporabljajo redkeje, zato je večina modelov prilagojena barvnim slikam (t.j. trije 8-bitni kanali). Zaradi širokega razpona vrednosti v termalnem spektru so termalne slike ponavadi zapisane s 16 biti. Pred uporabo v nevronski mreži jih morate zato pretvoriti v ustrezen format. Na področju interpretacije termalnih slik ni ustaljenih metod za ta postopek, zato lahko preizkusite različne pristope, kot so:

1. Slike lahko normalizirate na območje med 0 in 255 z uporabo funkcije `preprocess_thermal`, nato pa vrednosti premaknete na razpon med 0 in 1 ter pretvorite v zapis s plavajočo vejico.
2. 16-bitne slike lahko direktno skalirate na razpon med 0 in 1 (mogoče bo to pokvarilo kontrast in se bo mreža učila slabše)
3. Uporabite lahko adaptivne metode, kot je npr. Fieldscale

Osnoven pristop k predprocesiranju termalnih slik, tako da so razločne z golim očesom, je implementiran v naslednji python funkciji:

```
def preprocess_thermal(thermal):
    thermal = cv2.cvtColor(thermal, cv2.COLOR_GRAY2RGB)
    thermal = cv2.normalize(thermal, dst=None, alpha=0, beta=65535,
    norm_type=cv2.NORM_MINMAX)
    thermal = cv2.convertScaleAbs(thermal, alpha=255/(2**16))

    return thermal
```

Funkcija vrednosti najprej vrednosti raztegne čez celotno domeno 16-bitnih vrednosti, nato pa vrednosti skalira na razpon `up.uint8`. To večinoma deluje dovolj dobro, v prisotnosti zelo toplih objektov (npr. pogled proti soncu) pa lahko proizvede slike s slabim kontrastom. V takem primeru je bolje uporabiti adaptivne metode, kot je Fieldscale.

4 Naloge

4.1 Detekcija

Detekcija objektov v slikah je problem, kjer iščemo vse pojavitve objektov želenih razredov v sliki. Razširjeni prednaučeni modeli, kot je npr. YOLO, ponavadi vsebujejo 80 razredov, kot so definirani v podatkovni zbirki COCO [2]. Odvisno od problema in podatkov, ki jih imamo na voljo lahko to število povečamo ali zmanjšamo. V primeru naših podatkov boste modele za detekcijo učili samo na enem razredu, torej na dinamičnih ovirah. Predikcije modela bodo torej očrtani pravokotniki (ali binarne maske), ki označujejo objekte na vodni površini.

Za reševanje tega problema lahko uporabite katerokoli ustaljeno metodo za detekcijo, sledi nekaj predlogov:

1. DETR [3]
2. DINO [4]

3. YOLOv11 [5]

Za izhodišče lahko uporabite prednaučene modele in preizkusite njihovo delovanje na barvnih slikah. Model boste nato morali prilagoditi tako, da bo detektiral samo en razred, navadno to pomeni, da je treba zamenjati glavo modela in ga nato učiti na novih podatkih. Odvisno od ogrodja, ki ga uporabljate, boste morali anotacije pretvoriti v ustrezno obliko.

Metrika uspešnosti za detekcijo je F1. Posamezno detekcijo lahko upoštevate kot TP, če je prekrivanje z ustrezno anotacijo večje ali enako 0.5.

Opomba: Pri kakršnikoli pretvorbi podatkov si *vedno* prikažite pretvorjene podatke (torej narišite oznake na slike), da preverite če so oznake tam, kjer morajo biti. Isto velja za avgmentacije. Isto preverjanje lahko naredite tudi med učenjem (torej, da vaš model dela vsaj približno smiselne predikcije).

4.2 Semantična segmentacija

Semantična segmentacija vsakemu izmed pikslov vhodne slike pripiše semantično (t.j. pomensko) oznako razreda. Problem lahko razumemo tudi kot hkratno klasifikacijo vseh pikslov slike. Vhodni podatki so navadno 3-kanalne barvne slike, modeli pa kot izhod ponavadi vrnejo N matrik velikosti vhodne slike, kjer je N število razredov za klasifikacijo. Za pridobitev končnih predikcij je treba nad temi matrikami izvesti funkcijo `softmax`, nato pa za vsak piksel poiskati razred z največjo verjetnostjo (funkcija `argmax`).
Za semantično segmentacijo lahko uporabite katero od ustaljenih metod, sledi nekaj primerov:

1. U-Net [6]
2. SegFormer [7]
3. Mask2Former [8]
4. SETR [9]
5. K-Net [10]

Modele lahko učite z naključnimi (oz. smiselno inicializiranimi) utežmi, lahko pa uporabite uteži, ki so bile prednaučene na obstoječih podatkovnih zbirkah (ponavadi je to ImageNet). Glede na to, da podatkovna zbirka, ki ste jo dobili, ni zelo velika (dobili ste samo manjši del), lahko za učenje RGB modelov uporabite tudi druge zbirke (npr. LaRS [11]).
Metrika uspešnosti za semantično segmentacijo je mIoU, torej povprečna vrednost IoU za vse razrede.

5 Razdelitev podatkov

V podatkovni zbirki so samo slike iz ene od sekvenc, ki smo jih zajeli (*lj3*). Slike so razdeljene glede na smer plovbe, za testno podmnožico je izbran del plovbe od zapornic na Vrazovem trgu do Špice. To nanese na 489 učnih in 91 testnih slik. V podatkovni zbirki imate podani datoteki `train.txt` in `test.txt`, ki vsebujeta ustrezna seznama imen slik. Vseh barvnih slik je 580, vseh termalnih pa (zaradi tehničnih problemov med zajemanjem) 513. Če nekaterih imen med termalnimi slikami torej ne najdete, je to pričakovano.

6 Praktični nasveti

V tem poglavju sledijo praktična navodila, kako strukturirati meta proces učenja, shranjevanja in uporabe globokih modelov.

Porabite nekaj časa, da vzpostavite sistem za shranjevanje naučenih modelov. Glavna elementa tega sta shranjevanje uteži med učenjem in shranjevanje učnih parametrov. Redno shranjevanje naučenih uteži nam omogoča analizo delovanja modela preko učnih epoh (a pazite, da velikih modelov ne shranjujete prepogosto, ker lahko zasedejo ogromno prostora na disku). Lahko pa tudi zaženete evalvacijo modela po vsaki epohi in uteži shranite samo, če se je delovanje modela izboljšalo. Shranjevanje učnih parametrov pa je ključno za analizo delovanja modelov z različnimi hiperparametri (avgmentacije, resolucija, learning rate, različne funkcije izgube itd.).

Vzpostavite si grafičen prikaz delovanja modela med učenjem. V nekaterih ogrodjih je to že implementirano preko ogrodja tensorboard ali česa podobnega. Sicer pa lahko vsakih nekaj iteracij učenja izberete nekaj učnih slik in na njih prikažete predikcije, ki jih model proizvede. Sproti lahko na grafu prikazujete tudi padanje funkcije izgube in tako spremljate potek učenja. Na tak način lahko enostavneje ugotovite, če vaš model proizvaja smiselne predikcije, in če se te predikcije skozi čas izboljšujejo.

Proces učenja za barvne in termalne slike se načelno ne razlikuje. Če vaš izbrani model ne podpira enokanalnih slik, boste morali termalne slike ustrezno kopirati, da dobite 3-kanalno sliko. Barvne slike so bile anotirane na polni resoluciji (2k), zato jih boste za učenje morali verjetno zmanjšati. Zadovoljive rezultate se da dobiti že pri četrtini polne resolucije (prbl. 500 x 300px).

Literatura

[1] Jon Muhovič and Janez Perš. Joint calibration of a multimodal sensor system for autonomous vehicles. Sensors, 23(12):5676, 2023.

[2] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. CoRR, abs/1405.0312, 2014.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In European conference on computer vision, pages 213–229. Springer, 2020.

[4] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. arXiv preprint arXiv:2203.03605, 2022.

[5] Rahima Khanam and Muhammad Hussain. Yolov11: An overview of the key architectural enhancements. arXiv preprint arXiv:2410.17725, 2024.

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted

intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, pages 234–241. Springer, 2015.

[7] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in neural information processing systems, 34:12077–12090, 2021.

[8] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 1290–1299, 2022.

[9] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 6881–6890, 2021.

[10] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. Advances in Neural Information Processing Systems, 34:10326–10338, 2021.

[11] Lojze Žust, Janez Perš, and Matej Kristan. Lars: A diverse panoptic maritime obstacle detection dataset and benchmark. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 20304–20314, 2023.