Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Boštjan Bohte

Sistem za delno avtomatsko štetje polipov na slikah

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: prof. dr. Igor Kononenko
Somentor: doc. dr. Matjaž Kukar

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Za morske biologe je pomembno napovedovanje pričakovanega števila meduz, česar se lotevajo s fotografiranjem polipov na morskem dnu, iz katerih se pozneje razvijejo meduze. Ročno štetje polipov na fotografijah je zamudno in naporno delo. Naloga dela je implementirati sistem za delno avtomatsko štetje polipov na fotografijah morskega dna. Sistem naj podpira avtomatsko označevanje polipov z možnostjo ročnega popravljanja. Pri tem naj se uporabijo različne tehnike analize slik, kot so odstranjevanje ozadja, razvrščanje, iskanje geometrijskih struktur, detekcija robov in avtomatska klasifikacija želenih oblik s pomočjo algoritmov nenadzorovanega, delno nadzorovanega, nadzorovanega ter aktivnega strojnega učenja. Različni pristopi naj se pretestirajo na realni bazi fotografij polipov in naj se primerja dosežena točnost.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Boštjan Bohte sem avtor diplomskega dela z naslovom:

Sistem za delno avtomatsko štetje polipov na slikah

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Igorja Kononenka in somentorstvom doc. dr. Matjaža Kukarja,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 28. januarja 2015 Podpis avtorja:

Zahvaljujem se vsem, ki so s svojimi nasveti, napotki in finančno omogočanjem študija pripomogli k izdelavi mojega diplomskega dela. Med njimi bi posebej izpostavil mentorja prof. dr. Igorja Kononenka in somentorja doc. dr. Matjaža Kukarja, ki sta me s svojo strokovno pomočjo usmerjala s pravimi nasveti ter tako omogočila izdelavo diplomskega dela.

The last 29 days of the month
are the hardest.

Nikola Tesla

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
MAPE	mean absolute percentage error	povprečna absolutna procentna napaka
RGB	red green blue	rdeča zelena modra
L	lightness	svetlost
SMOreg	support vector machine	metoda podpornih vektorjev
TP	true positive	pravilno pozitivni
TN	true negative	pravilno negativni
FP	false positive	nepravilno pozitivni
FN	false negative	nepravilno negativni

Povzetek

Cilj diplomskega dela je implementirati sistem za delno avtomatsko štetje polipov na fotografijah morskega dna. Dobili smo veliko fotografij, pri čemer ročno označeni smo vzorec teh uporabili za učno množico, na podlagi te pa smo izvedli strojno učenje. Fotografije je bilo treba predobdelati z ustreznimi metodami, ki so popravile napačno osvetlitev, in tako bolj poenotiti vse slike. Najprej smo naučili model, ki odstrani ozadje na sliki, pri tem pa pusti vse, kar je del polipa. Sliko z odstranjenim ozadjem ponovno predobdelamo za model avtomatskega označevanja polipov. Uporabnik se nato odloči, ali bo avtomatsko označene polipe popravil in s tem dobil točno število polipov, ali pa bo to prepustil modelu napovedovanja števila polipov. Ta je naučen tako, da predvideva napako prejšnjih modelov in ustrezno napove bolj točno število polipov. Celotni sistem smo testirali nad testnimi slikami in ugotovili, da nekatere slike niso primerne za celotni sistem ter zato jih ne moremo uporabljati v tem sistemu. Pri ročnem popravljanju avtomatskih oznak polipov smo ugotovili, da je občutno hitrejše od ročnega označevanja polipov na celotni sliki. Model napovedi polipov je bil tudi pozitivno ocenjen, saj v povprečju zmanjša napako modela avtomatskega označevanja polipov.

Ključne besede: model, točnost, slika, polipi, označevanje, večnivojski perceptron, napaka, predobdelava. .

Abstract

The objective of the thesis is to implement a system for semi-automatic counting of polyps on the photographs of sea-bed. Numerous photographs have been obtained, wherein the manually labelled sample of those was used for the train set, and on basis of this the machine learning was implemented. Photographs had to be re-processed with appropriate methods in order to correct the bad lighting and furthermore even out the differences. First a model was used, which removes the background in the image while leaving everything that is part of the polyp. Image with the removed background is re-processed again for the model for automatic labelling of polyps. The user can then decide whether automatically labelled polyps will be repaired and thereby the exact number of polyps is obtained, or leave it to the model for predicting the number of polyps. This model is taught to foresee the error of previous models and to predict the number of polyps more accurately. The entire system was tested by test images and it was established that some images are not suitable for the entire system, and, therefore cannot be used in this system. While manually correcting automatic labels of polyps it was established that it is significantly faster than the manual labelling of polyps in the overall photographs. The model for predicting polyps was also positively assessed, as it reduces the error of the model for automatic labelling of polyps on average.

Keywords: model, accuracy, image, polyps, labeling, multilayer perceptron, error, pretreatment..

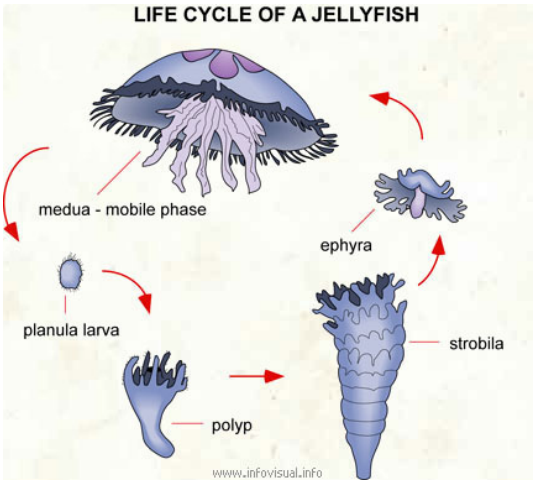
Poglavje 1

Uvod

Diplomska naloga se ukvarja z avtomatskim štetjem polipov, ki so ena od faz življenja meduz, prikazano na sliki 1.1. Da bi lahko bolje razumeli populacijsko dinamiko meduz in njihovih obdobij, biologi proučujejo populacijo polipov, njihovo gostoto in čas, ko se aseksualno razmnožujejo v meduze. Taka študija lahko pomaga odkriti, kako nekateri faktorji, kot recimo temperatura ali slanost vode, vplivajo na populacijo meduz. Da bi to izvedeli, se v rednih intervalih vzete fotografirajo z lokacije polipov, kjer ti ponavadi uspevajo. Polipi na teh slikah so prešteti in izmerjena je njihova gostota, ki se je spremenila v nekem pretečenem času. Te podatke nato primerjajo s faktorji, ki jih zanimajo. Šteti jih na roke, je zelo zamudno in nadležno delo, saj zahteva analizo več slik, ki vsebujejo več sto polipov. Avtomatično štetje polipov bi zato skrajšalo čas, da se ta študija opravi [9].

Cilj diplomskega dela je implementirati sistem za delno avtomatsko štetje polipov na fotografijah morskega dna. Pri tem smo morali sistem razdeliti na več pomembnih modelov in za njih ustvariti učno množico iz danih fotografij polipov. Celotni sistem smo naredili v javanskem okolju, pri tem pa smo si pomagali z različnimi znanimi orodji in metodami, ki so bili namenjeni strojnemu učenju ter računalniškem vidu. Struktura naloge je naslednja.

V poglavju 2 so opisana javansko okolje in orodja, ki smo jih uporabili za delo. Opisane so tudi metode, ki jih uporabljamo za strojno učenje in



Slika 1.1: Slika prikazuje življenjske cikle meduze in pojasnjuje, kaj je polip. Vir slike [1].

računalniški vid. V poglavju 3 opisujemo postopek ustvarjanja učne množice iz priloženih fotografij. Na kratko je tudi opisan postopek celotnega sistema, ki pojasnjuje, kako so posamezni modeli povezani in katero učno množico uporabljajo, če sploh. V poglavju 4 opisujemo model za odstranjevanje ozadja. Prvi razdelek razlaga, katere predobdelave je treba narediti na vhodu modela. Drugi razdelek pa opisuje postopek učenja modela z ustreznimi atributi in prikaz končnega rezultata na posamezni sliki. V poglavju 5 je najprej opisan model avtomatskega označevanja polipov, ta s sliko odstranjenega ozadja na njegovem vhodu poskuša označiti vse polipe, njegove napake pa lahko nato ročno popravimo. Sledi opis modela, ki kot vhod dobi avtomatsko označene polipe, pri tem pa predvideva, da so prisotne napake in poskuša napovedati bolj točno število polipov na sliki. Na koncu poglavja pa so vsi trije modeli testirani. Opisani so rezultati, kaj nam pomenijo in v katerih primerih bi bili ocenjeni negativno. Diplomsko delo se v poglavju 6 zaključi z razlago sklepnih ugotovitev in s predstavitvijo morebitnega nadaljnjega dela.

Poglavje 2

Uporabljene metode in orodja

V diplomskem delu je razvita aplikacija v celoti implementirana v Javi. To je programski jezik, ki je objektno orientiran in ima sintakso podobno programskemu jeziku C. Za programerje se je izkazala kot priljubljen jezik, saj je enostavna za pisanje robustne in razhroščene kode [6]. Na podlagi tega programskega jezika so bila izbrana ustrezna orodja za računalniški vid in strojno učenje, ki so kompatibilna z Javo.

2.1 OpenCV

OpenCV (*Open Source Computer Vision Library*) [12] je odprtokodna programska knjižnica računalniškega vida in strojnega učenja. OpenCV je bil zgrajen za zagotavljanje skupne infrastrukture za aplikacije računalniškega vida in za povečanje strojnega zaznavanja v komercialne namene. Knjižnica ima več kot 2500 optimiziranih algoritmov, ki vključujejo celovit nabor klasičnih in sodobnih algoritmov računalniškega vida ter strojnega učenja. Algoritmi se lahko uporabljajo za zaznavanje obrazov, prepoznavo objektov, zaznavo gibov v videoposnetkih itd. Najpogosteje se uporablja v podjetjih, raziskovalnih skupinah in vladnih organih [12]. Ima vmesnik za C++, C, Python, Java in MATLAB ter podporo za Windows, Linux, Mac OS in Android [12].

Za potrebe tega diplomskega dela se uporablja javanski vmesnik uporabniškega programa različice OpenCV2.4.7, ki je izšla 11. 11. 2013. Uporabljena je bila za manipulacijo nad slikami, kot so pretvorba formata slik, točkovne operacije, filtriranje, morfološke operacije, odkrivanja robov in izračun momentov Hu.

2.1.1 Pretvorba formata slike

Slika je dvodimenzionalna matrika celih števil oziroma dvodimenzionalna funkcija celoštevilskih koordinat (2.1).

$$I(u, v) \in \mathbb{P} \text{ in } u, v \in \mathbb{N} \tag{2.1}$$

Ponavadi imamo opravka s pravokotnimi slikami, z določenim številom stolpcev M in vrstic N, ki vsebujejo določeno število pikslov. Ločljivost slike je enaka številu vseh pikslov. Velikost slike v realnem svetu pa dobimo tako, da izračunamo število pikslov na kilometer razdalje. V diplomski nalogi uporabljamo intenzitetne in barvne slike.

- Pri intenzitetnih slikah imamo samo en kanal oziroma matriko. Ponavadi so to 8-bitne slike, katere vsebujejo 256 različnih intenzit. Slika, ki vsebuje samo dve različni intenziteti, pravimo da je binarna slika.
- Barvne slike vsebujejo tri kanale oziroma matrike, ki skupaj tvorijo eno sliko.

Obstaja zelo veliko formatov slik, delimo jih na rastrsko (bitno) in vektorsko grafiko [4].

- vektorski formati: SVG, CGM, DXF, AI, PICT, WMF, PS, EPS, PDF
- rastrski formati: TIFF, GIF, PNG, JPEG, JFIF, EXIF, BMP, PBM, RGB, RAS, TGA,...

V diplomski nalogi se ukvarjamo z rastrskimi slikami v obliki matrik. Pri tem spreminjamo formate slik iz RGB v sivinsko sliko (2.2) in iz RGB v Lab (2.3).

- RGB ↔ sivinska [12]:

$$\text{sivinska} = 0,299 \cdot R + 0,587 \cdot G + 0,114 \cdot B \tag{2.2}$$

- RGB ↔ L*a*b, izračunamo po enačbi (2.3) [12],

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0,412453 & 0,357580 & 0,180423 \\ 0,212671 & 0,715160 & 0,072169 \\ 0,019334 & 0,119193 & 0,950227 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.3}$$

$$L = \begin{cases} 116Y^{1/3} & \text{za } Y > 0,180423 \\ 903,3Y & \text{za } Y \leq 0,008856 \end{cases} \tag{2.4}$$

$$a = 500(f(X) - f(Y)) + delta \tag{2.5}$$

$$b = 200(f(Y) - f(Z)) + delta \tag{2.6}$$

kjer je $f(t)$ izračunan po enačbi (2.7)

$$f(t) = \begin{cases} t^{1/3} & \text{za } t > 0,008856 \\ 7,787t + 16/116 & \text{za } t \leq 0,008856 \end{cases} \tag{2.7}$$

in $delta = 128$ za 8-bitno sliko.

Nato pa še prilagodimo vrednosti za posamezno matriko, v našem primeru je to 8-bitna matrika, ki ima enačbo (2.8).

$$L = L255/100,\ a = a + 128,\ b = b + 128 \tag{2.8}$$

2.1.2 Histogram in točkovne operacije

Histogram je zelo enostavna statistika slike, ki opisuje frekvenco posameznih intenzitetnih vrednosti v sliki. Pri tem je $h(i)$ = število pikslov slike I z intenziteto i. Z njim lahko detektiramo probleme pri zajemanju slik, ne moremo pa zajeti

prostorske informacije slik. V primeru, da ima slika veliko različnih intenzitet(ponavadi več kot 256), se naredi, da posamezna vrednost v histogramu pokriva več vrednosti pikslov. Celoten obseg vrednosti z $max.$ vrednostjo B razdelimo na K intervalov dolžine $k_B = K/B$ (2.9).

$$h_j = card\{(u,v)|a_j \leq I(u,v) < a_{j+1}\} \text{ za } 0 \leq j < B \tag{2.9a}$$

$$a_j = j\frac{K}{B} = j \cdot k_B \tag{2.9b}$$

Pri barvnih slikah, ki imajo več kanalov, računamo histogram za vsak kanal posebej.

Točkovne operacije so operacije, ki se izvajajo na pikslih, pri čemer se velikost, geometrija in lokalna struktura slike ne spremeni. Nova vrednost piksla je odvisna samo od prejšne vrednosti istoležnega piksla. Poznamo homogene, kot je spreminjanje svetlosti in kontrasta, globalno upragovanje,... ter nehomogene operacije, kot je lokalno spreminjanje svetlosti in konstrasta. V diplomski nalogi uporabljamo homogeno operacijo izenačenja histograma, za povečanje konstrasta slike, ki je kombinacija obsega intenzitetnih vrednosti in razlike med maksimalno in minimalno vrednostjo piksla, po enačbi (2.10).

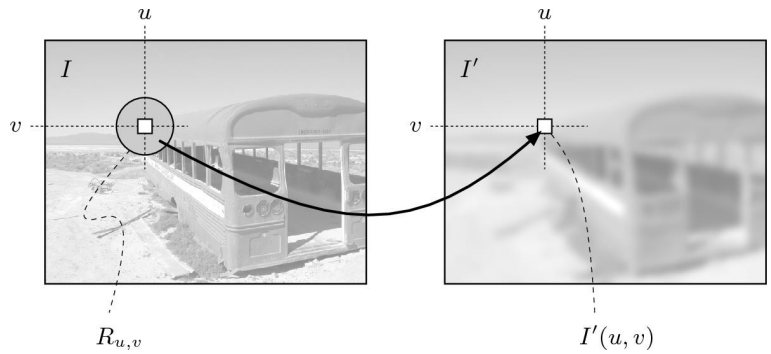
$$f_{eq}(a) = \left| H(a)\frac{K}{N \times M} \right| \tag{2.10}$$

Uporabljamo tudi nehomogeno operacijo izenačenja lokalnega histograma, ki izračuna intenziteto posameznega piksla, glede na njegov histogram regije [4].

2.1.3 Filtriranje slik

Filtriranje je operacija, ki ne spremeni geometrije slike in se uporabi več kot en piksel za izračun vrednosti novega piksla. Za izračun $I'(u,v)$ se uporabi celotno regijo $R_{u,v}$ s slike I. Primer je prikazan na sliki 2.1. Po matematični lastnosti se filtri delijo na linearne in nelinearne.

- Linearni filtri kombinirajo vrednosti pikslov s filtriranjem regije na linearen način kot uteženo vsoto. Pri tem se uporablja filtrina dvodimenzionalna matrika (ali maska) $H(i,j)$, ki ima koordinatni sistem z



Slika 2.1: Slika prikazuje primer filtra, ki uporabi celotno regijo za izračun enega piksla. Vir slike [4].

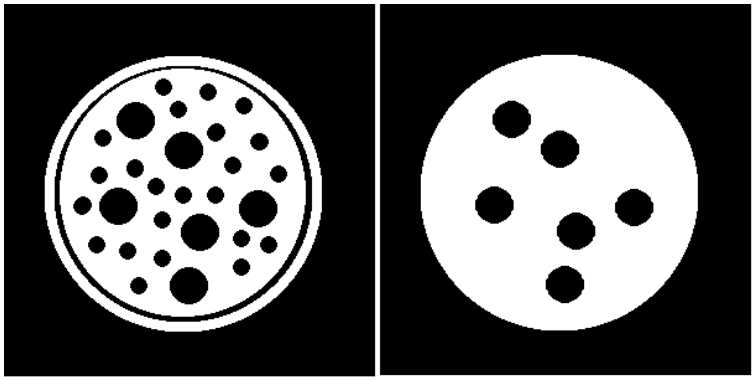
izhodiščem v središčnem elementu in je definirana s koeficienti filtra. V diplomski nalogi uporabljamo Gaussov filter za glajenje robov. Postopek glajenja poteka tako, da center filtrine matrike poravnamo s pikslom na sliki. Zmnožimo vse koeficiente filtra z vrednosti istoležnih pikslov in nato vsoto shranimo kot novo vrednost piksla. Enačba Gaussovega filtra za posamezni koeficient matrike (2.11).

$$G_\sigma(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.11}$$

- Nelinearni filtri tudi računajo vrednost piksla s pomočjo vrednosti pikslov z določene regije na originalni sliki, pri tem pa uporabljajo nelinearno funkcijo. Z nelinearnim filtrom se izognemu uniformnemu glajenju. Pri diplomski nalogi uporabljamo medianin filter, za glajenje regij polipov. Deluje tako, da pikslu dodelimo vrednost mediane okoliške regije (2.12).

$$I'(u,v) = medianI(u+i, v+j)|(i,j) \in \mathbb{R} \tag{2.12}$$

Mediana je robustna statistika, pri katerem posamezni odstopajoči elementi nimajo velikega vpliva. Tako se robovi slike ohranijo, šum se pa zamenja z okolico prevladujočih pikslov [4].



Slika 2.2: Slika prikazuje pred uporabo operatorja zaprtje in po njej. Vir slike [2].

2.1.4 Morfološki filtri

Morfološki filtri, ki jih ponavadi izvajamo na binarnih slikah, so namenjeni spreminjanju lokalne strukture slike, kot so:

- odstranjevanje majhnih elementov,
- polnjenje lukenj in
- iskanje obrisov.

Osnovne morfološke operacije so definirane s strukturnim elementom, ki je matrika binarnih vrednosti in definira lastnost morfološkega filtra [4].

V diplomski nalogi uporabljamo operator zaprtje. Ta se uporablja za polnjenje lukenj in je sestavljen iz dveh osnovnih morfoloških operacij, to sta širitev in erozija. Primer učinka zapiranja je prikazan na sliki 2.2.

2.1.5 Metoda za odkrivanje robov

Odkrivanje robov je ena izmed temeljnih operacij računalniškega vida. Obstajajo številni pristopi odkrivanja robov in eden izmed njih je postopek

Canny. Razvil ga je John F. Canny [5] leta 1986. Odkrivanje robov s postopkom Canny je najbolj rigorozno definiran in velikokrat uporabljen operator. Njegovo popularnost lahko pripišemo trem kriterijem, in sicer so to:

- nizek delež napak (dobro zazna samo prave robove),
- dobra lokalizacija (razdalja med odkritim robom in pravim robom mora biti minimalna) in
- minimalni odziv (samo eden odzivni detektor na rob).

Tipična implementacija odkrivanje robov s postopkom Canny sledi korakom, ki so navedeni spodaj:

- Preden poskušamo najti in detektirati rob, moramo sliko pogladiti z Gaussovim filtrom, da zmanjšamo šum na sliki. Gaussov filter uporablja enostavno masko, ki je ponavadi veliko manjša od same slike. Z masko potuje po sliki in s konvolucijsko metodo manipulira vsak njen piksel. Večji kot je filter po dolžini, manjši je vpliv šuma na sliko.
- Po glajenju določimo magnitudo roba, pri katerem izračunamo gradient slike. Tega izračunamo po Sobelovem operatorju. Sobel uporablja dva filtra, pri tem gre eden po osi x (2.13) in drugi po osi y (2.14). Nato se izračuna absolutna magnituda posameznega gradienta (2.15).

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{2.13}$$

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{2.14}$$

$$|G| = |G_x| + |G_y| \tag{2.15}$$

- Izračunamo usmerjenost roba glede na gradient G_x in G_y. Če ima gradient G_x vrednost 0, ima lahko rob usmerjenost 90 ali 0 stopinj, odvisno od gradienta G_y. Če ima G_y vrednost 0, je usmerjenost roba 0 stopinj, drugače je pa 90 stopinj. Za vse ostale primere, ko G_x nima vrednosti 0, se računa usmerjenost roba po enačbi (2.16):

$$\theta = \text{atan2}(\frac{\text{G}_\text{y}}{\text{G}_\text{x}}) \tag{2.16}$$

- Naslednji korak je, da izračunane vrednosti diskretiziramo tako, da lahko rob predstavimo dvema sosednjima piksloma. Rob ima lahko usmerjenost 0 stopinj (horizontalno), 45 stopinj (pozitivna diagonala), 90 stopinj (vertikalno) in 135 stopinj (negativna diagonala). Rob dobi diskretno vrednost glede na to, kateri usmerjenosti se najbolj približuje.

- Ko je usmerjenost robov znana, se vsi piksli, ki niso kandidati za rob, pobrišejo. S tem se naredi tanka črta, ki predstavlja rob.

- Na koncu se izbere robove glede na prag 1 in prag 2. Če bi imeli samo en prag, bi zaradi šuma bili nekateri robovi večkrat prekinjeni in zato bi bili tudi robovi na sliki predstavljeni kot črtkana črta. Da se temu izognemo, uporabljamo dva praga. Vsi robovi, ki so nad pragom 2, so avtomatično zaznani kot rob. Vsi robovi, ki so nad pragom 1 in pod pragom 2, se morajo držati roba, ki je nad pragom 2, da se lahko označijo kot rob. S to tehniko omilimo učinek črtkanee črte. Pri delu naloge se uporablja odkrivanje robov s postopkom Canny za ločitev med polipi nad posamezno regijo [7].

2.1.6 Momenti Hu

Momenti in s tem invariantne povezave so bili med različnimi aplikacijami obširno analizirani za označevanje vzorcev v slikah. Znani momenti vključujejo geometrične, zernike, rotacijske in kompleksne momente. Invariantni momenti so bili prvič predstavljeni v [16]. Ti imajo šest absolutnih pravokotnih

invariant in eno nagnjeno pravokotno invarianto na podlagi algebrskih invariant, ki so ne le neodvisne od pozicije, velikosti ter rotacije, ampak tudi od vzporedne projekcije. Momenti Hu so se izkazali za ustrezne pri prepoznavanju vzorcev s slik ne glede na njihovo pozicijo, velikost in rotacijo, s predpostavko, da je funkcija slike zvezna in da na sliki ni nobenega šuma.

Slika $f(x, y)$ ima dvodimenzionalni moment, ki se izračuna po enačbi (2.17):

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) \mathrm{d}x \mathrm{d}y \tag{2.17}$$

kjer je p,q=0,1,2,... Če se sliki spremeni pozicija, velikost ali rotacija, se spremeni tudi moment m_{pq}. Da se temu izognemo, izračunamo centralni moment, ki ga zapišemo z enačbo (2.18):

$$u_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \overline{x})^p (y - \overline{y})^q f(x, y) \mathrm{d}x \mathrm{d}y \tag{2.18}$$

kjer $p, q = 0, 1, 2, ...$, pri čemer je točka piksla $(\overline{x}, \overline{y})$ centroid od slike $f(x, y)$. $\overline{x}$ se izračuna po enačbi (2.19) in $\overline{y}$ po enačbi (2.20).

$$\overline{x} = \frac{m_{10}}{m_{00}} \tag{2.19}$$

$$\overline{y} = \frac{m_{01}}{m_{00}} \tag{2.20}$$

Centralni moment u_{pq}, ki je izračunan po sliki $f(x, y)$, je ekvivalenten momentu m_{pq}, če postavimo center slike na centroid. Tako je centralni moment neodvisen od pozicije slike. Da bi bil centralni moment še neodvisen od velikosti slike, ga normaliziramo po enačbi (2.21):

$$n_{pq} = \frac{u_{pq}}{u_{00}^{y}}, y = (p + q + 2)/2, p + q = 2, 3, ... \tag{2.21}$$

Glede na normalizirani centralni moment je Hu [16] predstavil sedem invariantnih momentov:

$$\varphi_1 = n_{20} + n_{02} \tag{2.22}$$

$$\varphi_2 = (n_{20} + n_{02})^2 + 4n_{11}^2 \tag{2.23}$$

$$\varphi_3 = (n_{30} + 3n_{12})^2 + (3n_{21} - u_{03})^2 \tag{2.24}$$

$$\varphi_4 = (n_{30} + 3n_{12})^2 + (n_{21} + u_{03})^2 \tag{2.25}$$

$$\begin{aligned}\varphi_5 = (n_{30} - 3n_{12})(n_{30} - u_{12})[(n_{30} + n_{12})^2 - 3(n_{21} + n_(03)^2)] \\ + (3n_{21} - n_{03})(n_{21} + n_{03})[3(n_{30} + n_{12})^2 - (n_{21} + n_{03})]\end{aligned} \tag{2.26}$$

$$\begin{aligned}\varphi_6 = (n_{20} - n_{02})[(n_{30} + n_{12})^2 - (n_{12} + n_{03})^2] \\ + 4n_{11}(n_{30} + n_{12})(n_{21} + n_{03})\end{aligned} \tag{2.27}$$

$$\begin{aligned}\varphi_7 = (3n_{21} - n_{03})(n_{30} + n_{12})[(n_{30} + n_{12})^2 - 3(n_{21} + n_{03})^2] \\ - (n_{30} - 3n_{12})(n_{21} + n_{03})[3(n_{30} + n_{12})^2 - (n_{21} + n_{30})^2]\end{aligned} \tag{2.28}$$

Teh sedem momentov ima koristno lastnost, saj se ne spremenijo, če sliko premaknemo, skaliramo ali rotiramo [16].

2.2 Monte media library

Monte media library je javanska knjižnica za procesiranje medijskih podatkov. V diplomski nalogi jo uporabljamo za predobdelavo slik pred odstranjevanjem ozadja. Uporabljamo metodo whiteBalanceGreyworld, ki je navedena v njihovi dokumentaciji. Metoda izboljša lastnosti slik, posnetih v slabi osvetlitvi. Dodatne informacije in dokumentacija so dostopne na njihovi spletni strani [13].

2.2.1 Metoda Grey-World

Algoritem Grey-World je definiran za RGB barvni prostor in sloni na predpostavki, da je povprečna barva odbita od objektov na sliki enaka barvi osvetlitve. Odstopanje povprečne osvetlitve posameznih barv slike od standardne je tako indikator, da je bila slika posneta pod nestandardnimi pogoji. Če želimo takim slikam popraviti osvetlitev, moramo popraviti vrednost osvetlitve vseh slikovnih elementov.

Algoritmu najprej izračunamo povprečno intenziteto posameznega piksla na področju celotne slike (2.29), nato v skladu z enačbami (2.30) in (2.31) popravimo še vrednosti intenzitete vseh pikslov na sliki [15].

$$R_{pov} = \frac{\sum_{i=0}^{x}\sum_{j=0}^{y} R(i,j)}{xy} \tag{2.29a}$$

$$G_{pov} = \frac{\sum_{i=0}^{x}\sum_{j=0}^{y} G(i,j)}{xy} \tag{2.29b}$$

$$B_{pov} = \frac{\sum_{i=0}^{x}\sum_{j=0}^{y} B(i,j)}{xy} \tag{2.29c}$$

$$\tag{2.29d}$$

Pri tem so R_{pov},G_{pov},B_{pov} povprečne intenzitete barve, $R(i,j)$,$G(i,j)$,$B(i,j)$ intenzitete posamezne barve, x, y pa je velikost slike.

$$S_R = \frac{R_{std}}{R_{pov}}\ S_G = \frac{G_{std}}{G_{pov}}\ S_B = \frac{B_{std}}{B_{pov}} \tag{2.30}$$

Pri tem so S_R, S_G, S_B faktorji množenja za posamezno barvo in R_{std}, G_{std}, B_{std} standardne intenzitete za posamezno barvo.

$$R_n(i,j) = R(i,j)S_R \tag{2.31a}$$

$$G_n(i,j) = G(i,j)S_G \tag{2.31b}$$

$$B_n(i,j) = B(i,j)S_B \tag{2.31c}$$

Pri tem so $R(i,j)$, $G(i,j)$, $B(i,j)$ stare barvne intenzitete posameznega piksla in $R_n(i,j)$, $G_n(i,j)$, $B_n(i,j)$ popravljene vrednosti barvne intenzitete piksla.

2.3 WEKA

WEKA (*ang. TheWaikato Environment for Knowledge Analysis*) [8] se je razvila ob vedno večji potrebi raziskovalcev po najsodobnejših tehnikah strojnega učenja. Projekt se je začel razvijati leta 1992. Na voljo so bili različni

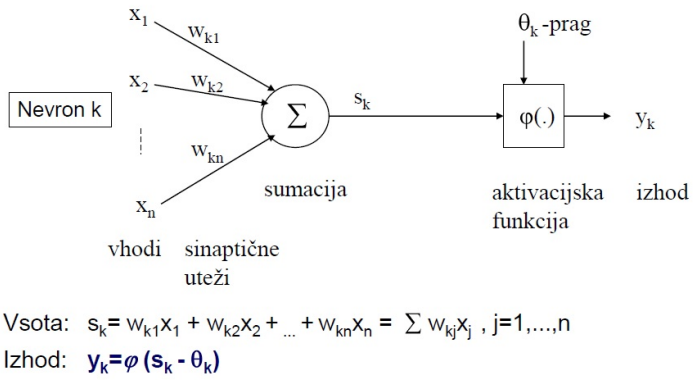
učni algoritmi, ki so delovali na različnih delovnih okoljih in oblikah podatkov. Danes je WEKA prepoznavni znak podatkovnega rudarjenja in strojnega učenja. Je splošno sprejeta v različnih akademskih in poslovnih krogih ter je postala široko uporabljeno orodje za raziskavo podatkovnega rudarjenja. Izšla je kot odprtokodna programska oprema in s tem omogočila številnim skupnostim, da razvijajo ter lažje delajo na projektih, ki širijo ali vključujejo WEKO. Na začetku projekta je bila večinoma napisana v jeziku C, od WEKA 3.0 naprej pa je v celoti napisana v programskem jeziku Java. Cilj projetka WEKA je zagotoviti celovito zbirko algoritmov strojnega učenja in orodij za predobdelavo podatkov. Uporabnikom omogoča, da hitro preizkušajo in testirajo različne metode strojnega učenja nad različnimi podatki. Njena modularna in razširljiva arhitektura omogoča sofisticirane procese podatkovnega rudarjenja, pri tem pa se lahko uporablja velika zbirka učnih algoritmov in orodij. Delovna miza omogoča algoritme za regresijo, klasifikacijo, razvrščanje, asociacijska pravila rudarjenja in izbor atributov. Za predhodno raziskovanje podatkov je dobro poskrbljeno z vizualizacijo uporabljenih podatkov in veliko orodji za predobdelavo podatkov [8]. Za potrebe tega diplomskega dela se uporablja različica WEKA 3.6. Metode, ki jih uporabljamo, so:

- linearna regresija,(razdelek 2.3.2)
- večnivojski perceptron, (razdelek 2.3.1)
- iskanje gruč s simple K means. ([9])

Delali smo tudi z REPTree, ZeroR in SMOreg modeli, vendar smo jih pozneje izključili iz dela. Programska oprema in dokumentacija je na voljo na njihovi spletni strani [3].

2.3.1 Večnivojski perceptron

Nevron je osnovna procesna enota nevronskih mrež. Ta pa je paralelni model računanja z različno stopnjo kompleksnosti, ki vključuje gosto povezane



Slika 2.3: Prikaz modela nevrona. Vir slike: [14].

adaptivne procesne enote. Razporeditev in povezanost nevronov tvorita arhitekturo nevronske mreže. Te so primerne za področje aplikacij, kjer imamo majhno ali nepopolno razumevanje problema, ki ga rešujemo, in kjer imamo na voljo učne podatke, kot so meritve, izračuni ... Učenje nevronske mreže je proces, v katerem se uteži (W)nevronske mreže prilagajajo nenehnim spodbujanjem okolja. Enačba uteži: $W(t+1) = W(t) + \Delta W(t)$.

Model nevrona

Nevron je informacijska procesna enota, ki vsebuje:

- množico sinaps ali povezav,
- sumacijo (vhodni signali se seštejejo z utežmi) in
- aktivacijsko funkcijo (določa izhod nevrona).

Slika 2.3 pojasnjuje model nevrona.

Večnivojski perceptron

V diplomski nalogi uporabljamo model: večnivojski perceptron. Sestavljen je iz:

- vhodnega nivoja, kjer so postavljeni vsi vhodi v model,
- vsaj enega ali več skritih nivojev nevronov in
- izhodnega nivoja nevronov.

Nevron vsakega nivoja je povezan z vsemi izhodi predhodnega nivoja. Pri učenju večnivojskega perceptrona se uporablja algoritem vzvratnega procesiranja napake. Na začetku učenja modela se inicializirajo sinaptične uteži na majhne naključne vrednosti. Nato pošljemo na vhod vzorec učnih podatkov iz učne množice in dobimo vrednost na izhodu. Izračuna se napaka, nato pa se v smeri od izhoda proti vhodu popravljajo sinaptične uteži. Računanje ponavljamo z novim vzorcem učnih primerov, dokler povprečna kvadratna napaka učne množice ni dovolj majhna [14].

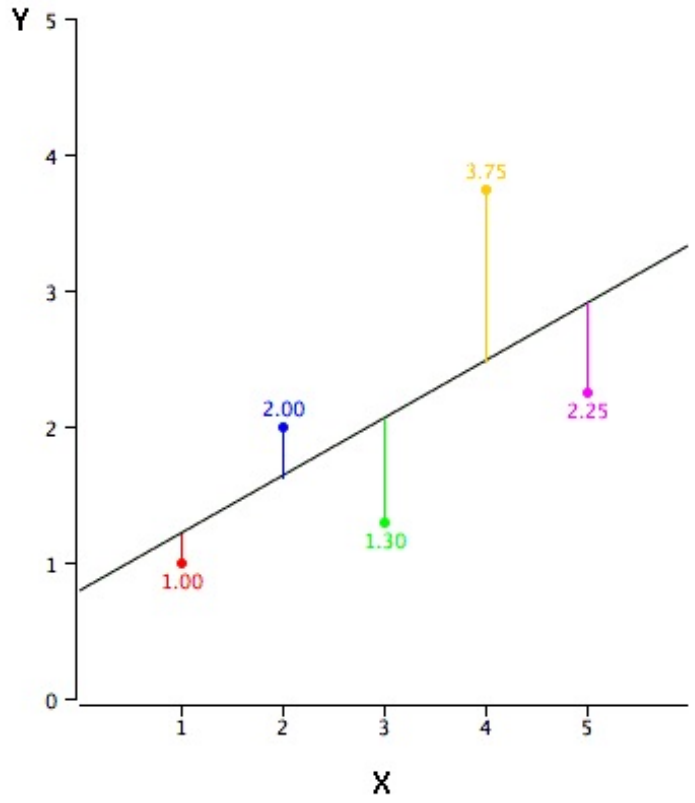
2.3.2 Linearna regresija

Pri enostavni linearni regresiji napovedujemo rezultat odvisne spremenljivke glede na rezultat druge odvisne spremenljivke. Spremenljivki, ki ji napovedujemo vrednost, pravimo kriterijska spremenljivka in se omenja kot Y. Spremenljivka, na podlagi katere vrednosti napovedujemo, se imenuje prediktor in jo omenjamo kot X. Enostavna linearna regresija se imenuje zato, ker ima samo en prediktor. Funkcija, kjer napovedujemo vrednost Y glede na vrednost X, tvori na grafu ravno črto — premico. Linearna regresija izračuna premico, ki se najbolj prilega točkam (X, Y). Tej premici pravimo regresijska premica. Kriterij za najboljše prileganje je največkrat minimalna vsota kvadratnih napak. Primer regresijske premice prikazuje slika 2.4

Če ima linearna regresija več spremenljivk prediktorjev, se imenuje multipla regresija. Ta poskuša najti linearno kombinacijo vseh predikcijskih spremenljivk, da izračuna regresijsko premico. Posamezne prediktorje utežimo z utežmi, ki jih zapišemo z enačbo (2.32),

$$\mathrm{Y}_i^{'} = a + \sum_{j=1}^{P} k_j X_{ij} \tag{2.32}$$

kjer je $Y_i^{'}$ napovedana vrednost, k_j utež posameznega prediktorja in a je presečišče z ordinatno osjo [11].



Slika 2.4: Črna premica je regresijska premica, točke so vrednosti Y glede na X. Navpične barvne črte pa predstavljajo napako regresije pri posamezni točki. Vir slike [10].

Poglavje 3

Eksperimentalna metodologija

Za reševanje naloge je bilo posredovanih 1000 slik s polipi in gradivo [9], ki je služilo kot ogrodje diplomske naloge. Da bi lahko naučili model štetja polipov na slikah, je bilo treba najprej ročno prešteti polipe na sliki, da dobimo število polipov, nato pa nad znanim številom iz preštetih slik izvesti strojno učenje in napovedati rezultat. Ker je 1000 slik preveč za ročno štetje polipov, je bil vzet vzorec 49 slik, nad katerimi se je nato učilo in testiralo točnost posameznega modela.

3.1 Ročno označevanje slik za učno množico

Izbranih je bilo 49 slik, ki so se med seboj razlikovale, in sicer z namenom, da bi pokrili raznolikost slik. Vsem izbranim slikam smo morali ročno označiti polipe, da smo lahko učili nadaljnje modele. Ker so polipi okroglaste oblike, je bil za označevanje uporabljen krogec. Pri označevanju smo pazili, da je površina krogca čim bolj pokrila površino polipa. Pri tem naj ne bi bilo vključeno nobeno ozadje. Vsak krogec predstavlja število enega polipa. Notranjost krogca predstavlja območje polipa, zunanjost krogca pa območje ozadja. S tem dobimo dve različni vrsti podatkov: območje polipa in ozadja ter število polipov na sliki. Krogec ne pokrije celotne površine polipa, saj polip ni pravilne kroglaste oblike. Prisotna pa je tudi človeška

napaka, saj obstaja verjetnost, da je pri vsaki ročno označeni sliki kakšen polip neoznačen ali narobe označen. Zato ročno označevanje prikazuje približek točnega označevanja polipov na sliki. Učno množico uporabljata dva modela: model za odstranjevanje ozadja in model za napovedovanje števila polipov. Pri tem si pomagata z ročno označenimi slikami.

3.1.1 Učna množica modela za odstranjevanje ozadja

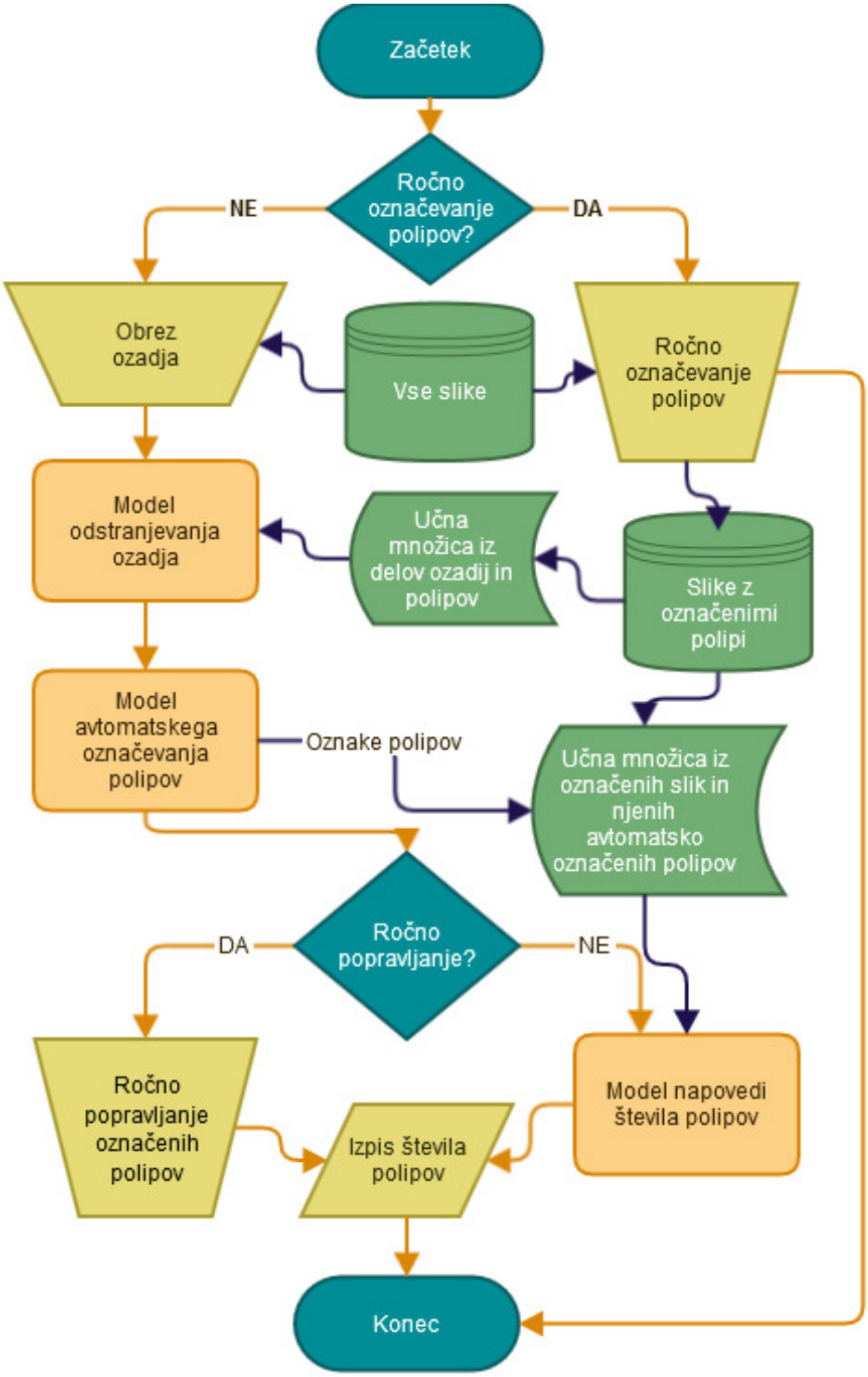
Model je naučen z učno množico, ki vsebuje primere ozadij in polipov pri določeni sliki. 10×10 pikslov velik del slike, ki ima srednji piksel znotraj krogca, se klasificira kot primer polipa. V nasprotnem primeru se klasificira kot primer ozadja. Vsakemu delu slike se izračuna še historam sivinske slike iz katere je bil odvzet.

3.1.2 Učna množica modela za napovedovanje števila polipov

Model je naučen z učno množico, ki vsebuje primere avtomatsko označenih polipov in histograma sivinske slike, nad katero so bili označeni polipi. Za razredni atribut uporablja število ročno označenih polipov na sliki.

3.2 Sistem za štetje polipov

Sistem deluje tako, da na vhod pošljemo sliko s polipi in ji obrežemo odvečno ozadje. Sliko nato obdela model za odstranjevanje ozadja, ki poskuša avtomatsko odstraniti vse, kar ni del polipa, in obdelano sliko pošlje modelu za avtomatsko označevanje polipov. Ta nato s predznanjem o obliki polipov poskuša označiti vse polipe, ki so ostali na sliki. Nato se mora uporabnik odločiti, ali bo sam popravil napake označevanja in tako dobil točno število polipov ali pa bo prepustil modelu napovedovanja števila polipov. Ta model poskuša ugotoviti napake obeh prejšnjih modelov in nato napove število polipov na celotni sliki. Podroben potek opisuje shema na sliki 3.1.



Slika 3.1: Shema prikazuje potek celotnega sistema.

Poglavje 4

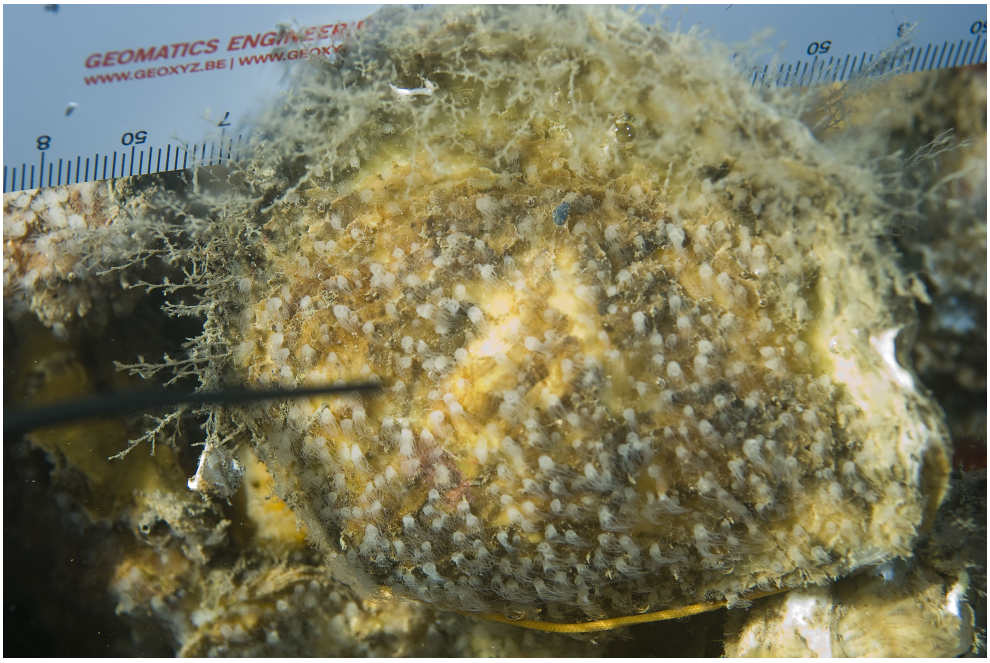
Odstranjevanje ozadja

4.1 Predobdelava slik za odstranjevanje ozadja

Slike se razlikujejo po kakovosti, različnosti obarvanosti in same strukture podlage, na kateri rastejo polipi. Na posamezni sliki se vidita območje polipov in ozadje. Primer kaže slika 4.1. Nekatere slike vsebujejo tudi ravnilo, kjer lahko nato izračunamo gostoto polipov. Ker pa nimajo vse slike ravnila, tega v tej nalogi ne računamo. Da bi si bile slike med seboj čim bolj podobne, je treba sliko barvno uravnotežiti. Pri tem se uporablja metoda *greyworld*, ki sliko pobeli. Ker to še zmeraj ni bilo dovolj, je bilo treba lokalno izenačiti histogram po svetlosti slike. To naredimo tako, da sliko, ki je v formatu RGB, pretvorimo v format Lab in nad kanalom L izenačimo lokalni histogram. Končno predobdelavo slike prikazuje primer slike 4.2.

4.2 Odstranjevanje ozadja

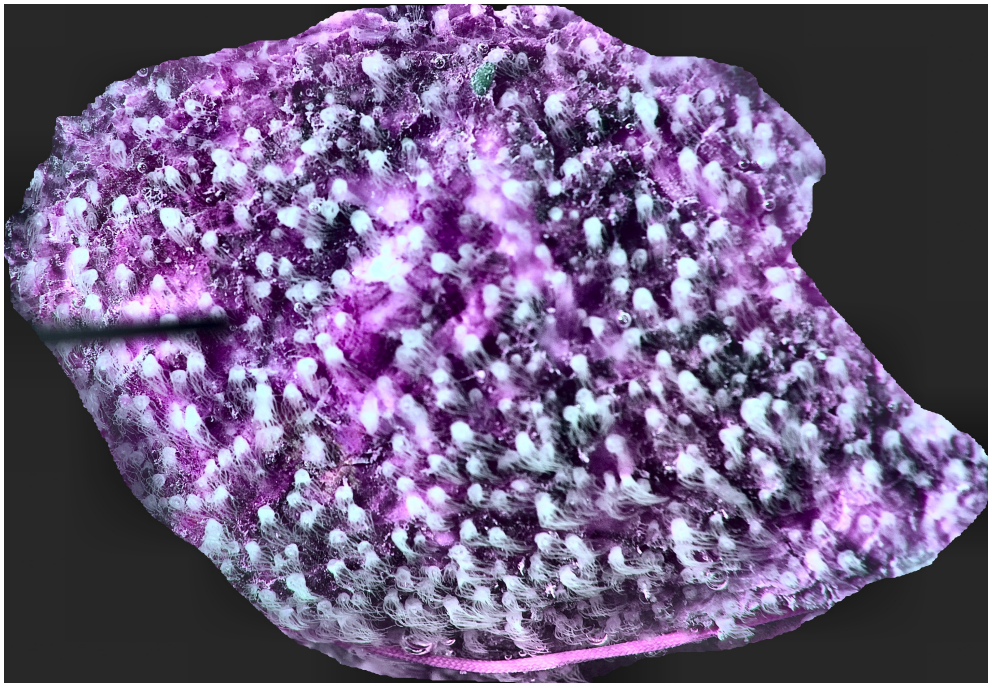
Za odstranjevanje ozadja smo izbrali model večnivojskega perceptona iz knjižnice WEKA, ker je že v [9] ugotovljeno, da je ta najbolj primeren. Model smo učili iz učne množice slik, na kateri so ročno postavljeni krogci na polipih. Območje znotraj krogca označimo kot območje polipa, območje zunaj krogca pa označimo kot ozadje. Pri tem se pojavi težava, da je na robu



Slika 4.1: Primer slike s polipi.

krogca še vedno nekaj območja polipa ali ozadja na napačni strani krogca, s čimer učimo model z napako. Primer prikazuje slika 4.4. Temu se delno izognemo tako, da za primer ozadja vsem krogcem povečamo njihov radij na $radij = 1,5 \times radij$, pri primeru polipa pa vsem krogcem zmanjšamo njihov radij na $radij = 0,5 \times radij$. To je prikazano na sliki 4.5. Najmanjši možni element, ki ga lahko dobimo s slike, je piksel, vendar lahko iz njega izvemo samo podatke o intenziteti posamezne barve. Da bi dobili informacije o strukturi dela slike, je treba zajeti več pikslov hkrati.

Vzamemo 10×10 pikslov velik del slike in ga obdelamo. V nadaljevanju izračunamo njegovo povprečno rdečo, zeleno in modro barvo ter momente Hu. Predobdelano sliko pretvorimo v sivinsko in vrednosti pikslov postavimo na interval $[0, 255]$. Izračunane vrednosti nato uporabimo kot atribute za učenje modela. Podrobne informacije o rabi atributov za učenje modela so prikazane v tabeli 4.1. Vsaki sliki iz učne množice naključno vzamemo 100 učnih primerov, od tega je 70 primerov ozadja in 30 primerov polipa. Ker je



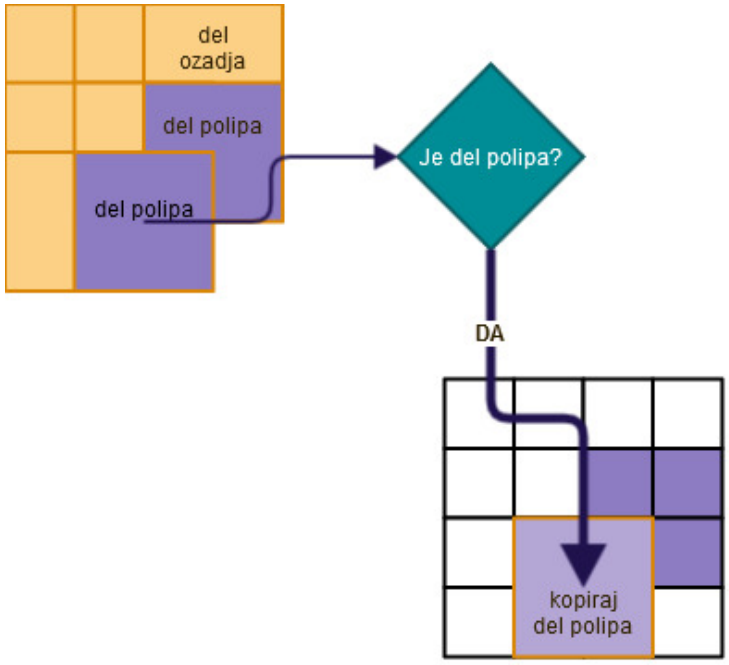
Slika 4.2: Primer iste slike po predobdelavi.

slik 49, skupaj tvorijo 4900 učnih primerov.

Pri odstranjevanju ozadja vzamemo dele slike, ki se med seboj prekrivajo. Če je del slike zaznan kot polip, se celotni del prepiše v prazno matriko oziroma sliko. Primer prikazuje shema na sliki 4.3. Pri odstranjevanju ozadja si želimo, da bi bilo odstranjenega čim več ozadja, z malo izgube izbrisanih polipov.

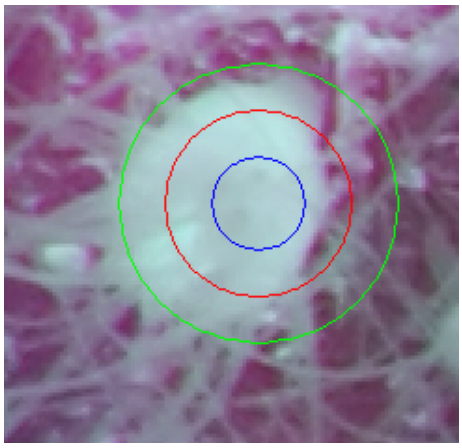
Atributi	opis
Moment Hu 1	enačba (2.22)
Moment Hu 2	enačba (2.23)
Moment Hu 3	enačba (2.24)
Moment Hu 4	enačba (2.25)
Moment Hu 5	enačba (2.26)
Moment Hu 6	enačba (2.27)
Moment Hu 7	enačba (2.28)
Rdeča barva	povprečna intenziteta rdeče barve celotnega dela slike
Zelena barva	povprečna intenziteta zelene barve celotnega dela slike
Modra barva	povprečna intenziteta modre barve celotnega dela slike
Histogram slike 1	seštevek vseh sivinskih pikslov na sliki z intenziteto na intervalu $[0, 63]$
Histogram slike 2	seštevek vseh sivinskih pikslov na sliki z intenziteto na intervalu $[64, 127]$
Histogram slike 3	seštevek vseh sivinskih pikslov na sliki z intenziteto na intervalu $[128, 191]$
Histogram slike 4	seštevek vseh sivinskih pikslov na sliki z intenziteto na intervalu $[192, 255]$
Ozadje	klasifikacijski atribut—ozadje ali polip

Tabela 4.1: Opis atributov, ki se uporabljajo za model odstranjevanja ozadja.

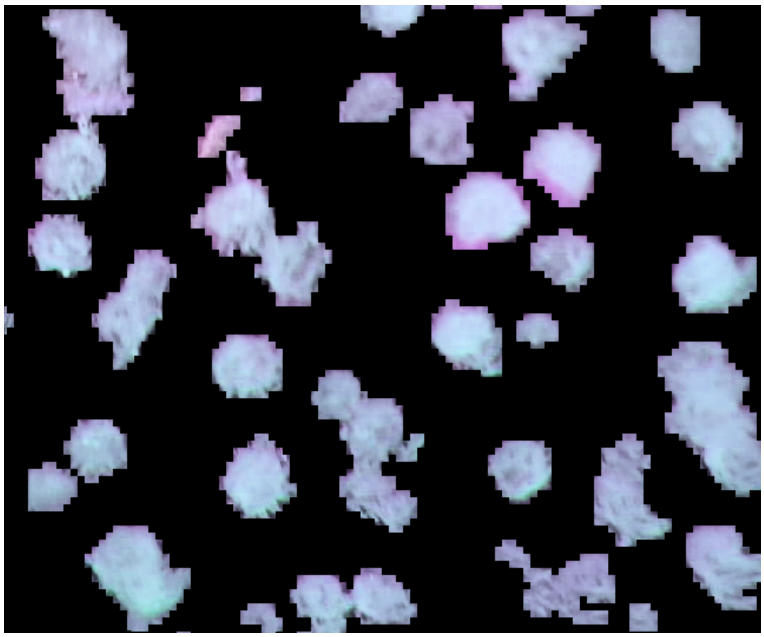


Slika 4.3: Prikaz sheme, kako deluje prepis dela slike v prazno matriko.

Slika 4.4: Primer označenega polipa, ki ne označuje celotnega polipa, označuje pa tudi malo ozadja.



Slika 4.5: Rdeči krogec je ročno postavljeni krogec, zeleni je povečan krogec na $radij = radij \times 2$ in se uporablja za določitev ozadja na sliki, modri je pomanjšani krogec na $radij = radij \times 0.5$ in se uporablja za določitev polipa na sliki. S tem smo omilili problem ročnega označevanja, ki je bil prikazan na sliki 4.4.



Slika 4.6: Slika prikazuje rezultat odstranjenega ozadja.

Poglavje 5

Avtomatsko označevanje in napovedovanje števila polipov

5.1 Avtomatsko označevanje polipov

Polipi se avtomatsko označujejo na slikah, ki imajo že odstranjeno ozadje. Če je bil polip odstranjen v postopku odstranjevanja ozadja, ga model ne more označiti, saj ima na voljo samo nepobrisani del na sliki. Točnost avtomatskega označevanja polipov je zato odvisna tudi od točnosti odstranjevanja ozadja na sliki.

5.1.1 Predobdelava podatkov za avtomatsko označevanje polipov

Pred začetkom avtomatskega označevanja polipov sliko z odstranjenim ozadjem primerno predobdelamo, da prilagodimo vhod za model avtomatskega označevanja. Pri prvem koraku na sliki poiščemo robove s postopkom Canny, pri čemer uporabimo prvi prag 50 in drugi prag 100. Sliko z najdenimi robovi pogladimo z Gaussovim filtrom, velikim 3×3 piksle, in s standardnim odklonom 2, po smeri x, da so robovi bolj široki. Sliko, glede na intenziteto

barve pikslov in roba, upragovimo s pragom 50, kot kaže enačba (5.1):

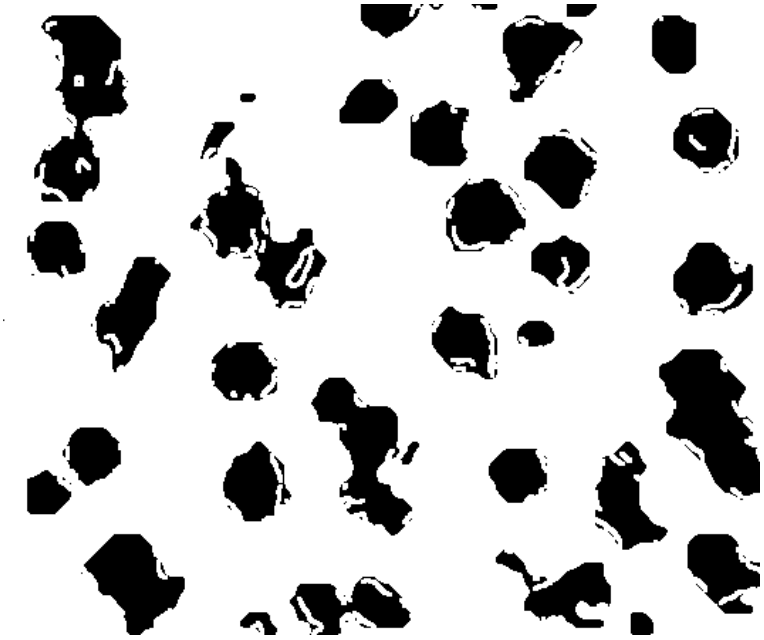
$$f(barvniPiksel) = \begin{cases} kanal_R < 50 & \text{piksel=255} \\ kanal_G < 50 & \text{piksel=255} \\ kanal_B < 50 & \text{piksel=255} \\ drugo & \text{piksel=0} \end{cases} \tag{5.1}$$

Pri tem dobimo binarno sliko, ki kaže na območje polipov in ozadja, kot kaže slika 5.1. Nad to sliko izvedemo morfološko operacijo zaprtje, ki uporablja za strukturni element krog s premerom 12 pikslov. Pri tem zmanjšamo šum na sliki in zarobimo regijo tako, da nima kvadratkov. Sledi barvanje regij. Vsako regijo pobarvamo in jo shranimo v tabelo regij. Te so med seboj ločene in vse nadaljnje predobdelave se izvajajo posamično. Nad vsako regijo ponovno naredimo zaprtje nad binarno sliko, ki je inverzna binarni sliki regije. S to operacijo poskušamo zapreti luknjice v posamezni regiji in s tem še zmanjšamo napako modela za odstranjevanje ozadja. Obdelanim regijam nato še povrnemo barvno strukturo, ki so jo vsebovale pred začetkom predobdelave za iskanje polipov, kot je prikazano na sliki 5.2.

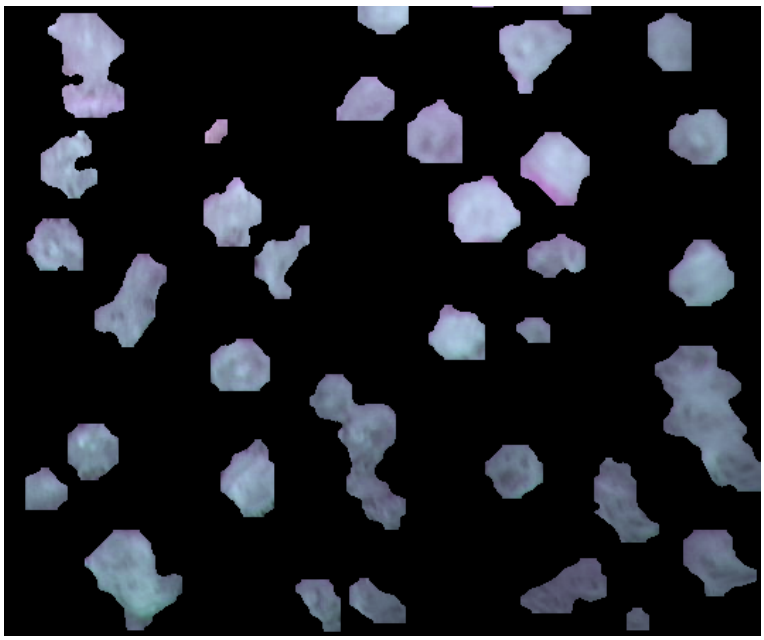
5.1.2 Avtomatsko označevanje polipov

Polipe označujemo na prej pobarvanih regijah velikih $N \times M$ pikslov, ki so shranjene v tabeli. Te se med seboj ne vidijo, zato ne more gruča regij predstavljati enega polipa, je pa lahko več polipov na posamezni regiji. Prvi del postopka je nenadzorovano iskanje gruč z metodo, ki jo je že [9] uporabil pri svojem delu. Vsako gručo nato pobarvamo z različno barvo in dobimo sliko regije, nad katero so najdene in pobarvane gruče. Primer je pokazan na sliki 5.3. Nad to sliko se iščejo polipi.

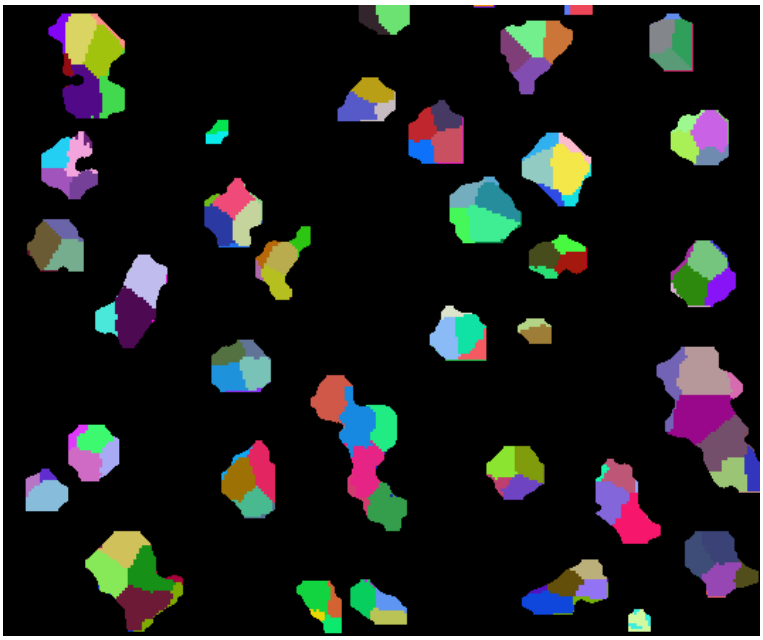
Za klasifikacijo polipa mora biti ustrezen pogoj. Da je pogoj izpolnjen, se mora nad regijo vrisati krogec, ki deluje po ustreznih pravilih. Vnaprej predpostavimo, da je na vhodni sliki samo območje polipov (brez območja ozadja) in da ni nobenega polipa, čigar radij je manjši od 7 pikslov, saj želimo, da se polip zazna na vsaj 4 ploščicah klasificiranega polipa. Če so polipi na



Slika 5.1: Slika prikazuje primer upragovanja slike z odkrivanjem robov s postopkom Canny.

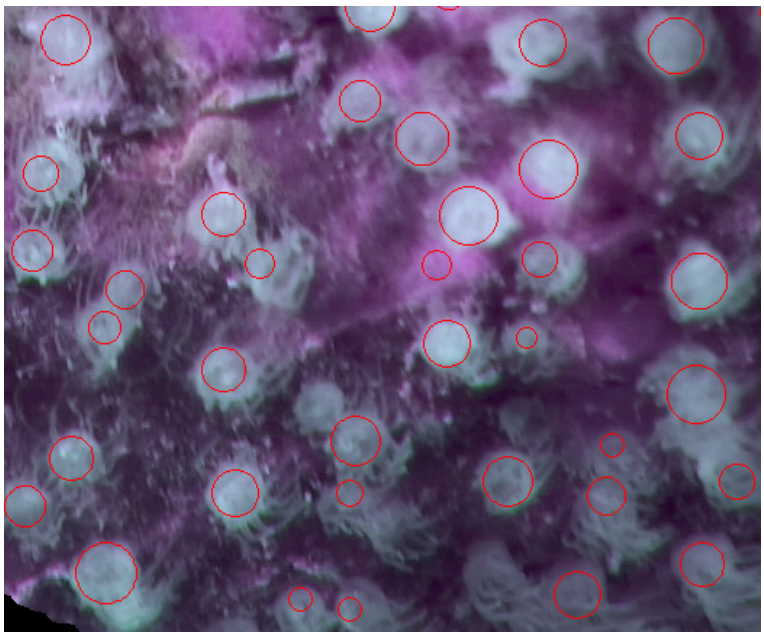


Slika 5.2: Končni rezultat predobdelave. Dosegli smo, da so se nekateri polipi odcepili drug od drugega (primer sta polipa levo zgoraj), odstranil se je odvečni del na polipu (lovke) in struktura slike ni več kvadratkaste oblike. Naredi se pa tudi napaka, saj so nekateri polipi zdaj popačeni in nimajo več kroglaste oblike.



Slika 5.3: Primer iskanja in barvanja gruč na regiji.

sliki manjši od radija 7 pikslov, model ne deluje zaradi slabe kakovosti slike. Pri iskanju začnemo s krogcem, ki ima $radij = \min\{N, M\}/2$ pikslov, in ga pomikamo po celotni regiji. Krogec se vriše, če njegova površina zajema manj kot 5% prazne površine. Če se krogec vriše, se pobrišejo vse barve na regiji, ki jih krogec zajema. Ker barve ponavadi sežejo čez vrisani krogec, se s tem pobriše večje območje okoli krogca. S to metodo želimo pobrisati celoten polip, saj je vrisani krogec manjši od polipa in bi tako njegovi nepobrisani deli motili zaznavo ostalih polipov. Ko se krogec ne more več nikamor vrisati v regijo, se mu zmanjša radij za 5 pikslov in ponovno poskusi z vrisavanjem. Radij se zmanjšuje, dokler ni manjši od 7 pikslov. Ker se radij začne z največjim krogcem, s tem onemogočimo zaznavo več manjših krogcev nad isto površino. Primer avtomatsko označenih polipov prikazuje slika 5.4.



Slika 5.4: Slika prikazuje primer avtomatsko označenih polipov s krogci. Na njej se vidi, da nekateri polipi niso označeni in da je prisoten nepravilno postavljen krogec, saj označuje ozadje.

5.1.3 Ročno popravljanje avtomatsko označenih polipov

Program omogoča ročno popravljanje avtomatsko označenih polipov. Napaka se pojavi, če oznaka ne označuje polipa ali če na polipu ni oznake. Napačno oznako polipa odstranimo s klikom na oznako. Novo oznako polipa pa ustvarimo s klikom na neoznačeno mesto. Ročno popravljanje napak avtomatskega označevanja je hitrejše od ročnega označevanja polipov, če je treba popraviti manj napak, kot je število polipov na sliki. Točnost ročno popravljenih oznak je odvisna od točnosti uporabnika, saj je od njega odvisno, kako dobro bo opazil napako in jo popravil. Če se odločimo za ročno popravljanje, s tem zaključimo avtomatsko štetje polipov, ker je nadaljnji model napovedi števila polipov na sliki naučen z napakami avtomatskih oznak.

5.2 Model za napovedovanje števila polipov

Polipi na slikah imajo različno velikost in pri tem so tudi oznake polipov, ki so predstavljene s krogci, različne velikosti. S tem so tudi napake avtomatskega označevanja polipov različne pri različnih velikostih krogcev. Pri majhnih krogcih je ponavadi večje število napačno zaznanih polipov, medtem ko je pri povprečni velikosti krogcev več neoznačenih polipov. Enako lastnost najdemo tudi pri sami sliki. Svetle slike imajo naprimer več napačno označenih polipov, temne pa več neoznačenih polipov. Izdelati želimo model, ki bi na podlagi števila različno velikih krogcev in podatkov celotne slike napovedal število polipov na sliki ter bil pri tem dovolj točen. Število polipov na sliki lahko dobimo tudi iz števila avtomatsko označenih polipov, saj število oznak predstavlja število polipov. Da model za napovedovanje števila polipov ocenimo kot uspešen, mora biti bolj točen od avtomatsko označenih polipov.

5.2.1 Predobdelava podatkov

Slika, s katere dobivamo podatke, je kopija slike, ki je bila obdelana za predobdelavo odstranjevanja ozadja, vendar pa na njej ni izenačen lokalni histogram. Pretvorimo jo v sivinko in vse njene intenzitete pikslov, razen intenzitete pikslov vrednosti 0, ki jih izpustimo iz podatkov, postavimo na interval $[Min, Max]$, pri tem je $Min = 0$, $Max = 99$. Intenzitete z vrednostjo 0 ne vključujemo, ker je s to vrednostjo zapisano obrezano ozadje in ga nočemo vključevati v učno množico. Najmanjša vrednost normaliziranega piksla je Niz, največja vrednost pa je Vis. Vrednost posamezne vrednosti se izračuna po enačbi (5.2). Normalizirati moramo tudi oznake polipov, ker so polipi fotografirani z različnih razdalj in zato so polipi na različnih slikah različne velikosti, kar pa vpliva tudi na oznake. Pri tem bi velikost istega krogca pomenila pri eni sliki velik krogec, pri drugi pa majhen krogec. S tem bi bila točnost modela odvisna tudi od razdalje fotografiranega polipa. Da bi se temu izognili, ploščino krogcev normaliziramo na intervalu $[0, 99]$. Pri tem se posamezno vrednost izračuna po enačbi (5.2).

$$h(i) = Min + (i + Niz)\frac{Max - Min}{Vis - Niz} \tag{5.2}$$

5.2.2 Regresijski model

Na podlagi podatkov histograma krogcev in histograma slike, ki smo ju naredili v predobdelavi, želimo narediti model, ki bo napovedal število polipov na sliki. Za učenje modela uporabimo atribute, navedene v tabeli 5.1. Model smo naučili na vseh 49 slikah, ki jih imamo za učno množico. Po principu — *leave-one-out* smo na učni množici primerjali točnost različnih regresijskih modelov. S knjižnico WEKA smo testirali večnivojski perceptron, REPTree, ZeroR, SMOreg in linearno regresijo. Slednji model se je izkazal kot najboljši, ker je imel najmanjši MAPE (5.9).

Atributi	opis
Vrednost krogcev 1	število krogcev s ploščino na intervalu $[0, 24]$
Vrednost krogcev 2	število krogcev s ploščino na intervalu $[24, 49]$
Vrednost krogcev 3	število krogcev s ploščino na intervalu $[50, 74]$
Vrednost krogcev 4	število krogcev s ploščino na intervalu $[75, 99]$
Vrednost pikslov 1	število pikslov z intenziteto na intervalu $[0, 63]$
Vrednost pikslov 2	število pikslov z intenziteto na intervalu $[64, 127]$
Vrednost pikslov 3	število pikslov z intenziteto na intervalu $[128, 191]$
Vrednost pikslov 4	število pikslov z intenziteto na intervalu $[192, 254]$
Pravo število	ročno prešteto število polipov za učenje regresije

Tabela 5.1: Opis atributov za regresijski model napovedovanja števila polipov na sliki.

5.3 Testiranje modela

Izmed še neoznačenih slik smo jih naknadno označili 42 in jih uporabili kot neodvisno testno množico.

5.3.1 Rezultati odstranjevanja ozadja

Na neodvisni testni množici izvedemo testiranje modela za odstranjevanje ozadja. Za vhodni podatek vzamemo matriko slike od odstranjevanja ozadja

Kratica	`opis`
`TP`	število delcev pravilno zaznanih kot del polipa
`TN`	število delcev pravilno zaznanih kot del ozadja
`FP`	število delcev nepravilno zaznanih kot del polipa
`FN`	število delcev nepravilno zaznanih kot del ozadja

Tabela 5.2: Definicija TP, TN, FP in FN pri testiranju modela za odstranjevanje ozadja.

in podatke ročno vrisanih krogcev na njeni izvorni sliki. Za izračun točnosti moramo določiti frekvence TP, TN, FP, FN, ki so definirani v tabeli 5.2. Pri izračunu točnosti si v celoti pomagamo z ročno vrisanimi krogci. Krogci so bili vrisani v polip, zato vse, kar je znotraj posameznega krogca, označimo kot polip. Ker pa so polipi večji od krogcev, ne moremo narediti enako za ozadje, saj bi dejanski delec polipa lahko označili za ozadje, kar pa bi bilo narobe. Pri določitvi ozadja vsem krogcem povečamo radij na $radij = 2 \times radij$. Vse kar se nahaja zunaj vseh povečanih krogcev, je ozadje. Točnosti območja med krogcem in povečanim krogcem, zaradi nezmožnosti določitve ozadja ali polipa, ne računamo. Vsem testnim slikam izračunamo:

- senzitivnost (5.3), ki nam pove, kolikšno je razmerje med delci polipov in pobrisanimi delci polipov;

$$\text{Senzitivnost} = \frac{TP}{TP + FN} \tag{5.3}$$

- specifičnost (5.4), ki nam pove, kolikšno je razmerje med odstranjenenim ozadjem in neodstranjenim ozadjem;

$$\text{Specifičnost} = \frac{TN}{FP + TN} \tag{5.4}$$

	Rezultati
Senzitivnost	$94,0\%$
Specifičnost	$88,7\%$
Preciznost	$34,4\%$
Klasifikacijska točnost	$89,1\%$

Tabela 5.3: Tabela prikazuje povprečne vrednosti izračunov pri testiranju modela za odstranjevanje ozadja.

- preciznost (5.5), ki nam pove, kolikšen delež delcev na sliki dejansko pripada polipu;

$$\text{Preciznost} = \frac{TP}{TP + FP} \tag{5.5}$$

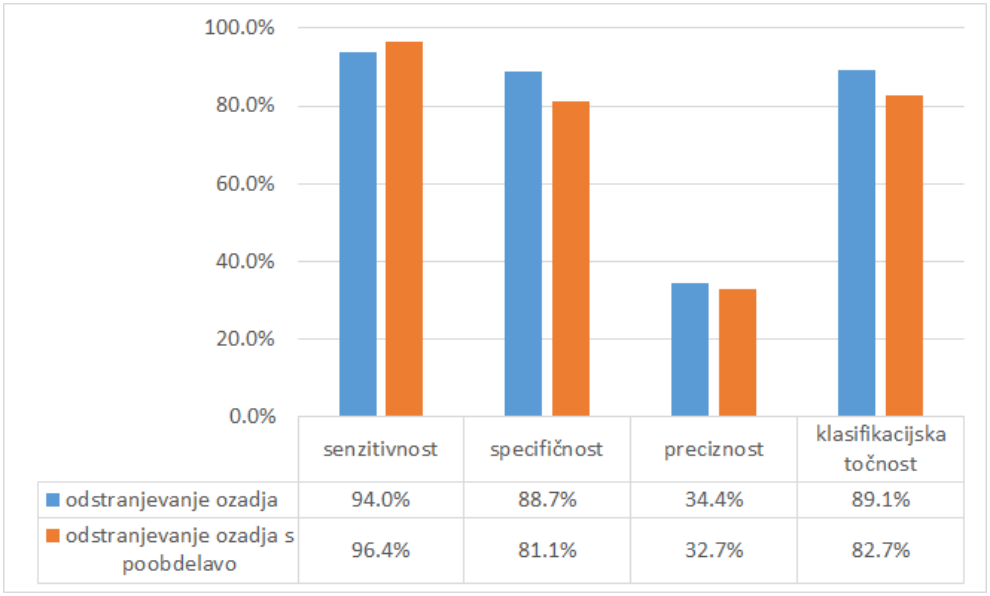
- klasifikacijsko točnost (5.6), ki nam pove skupno točnost odstranjenega ozadja in delcev polipov.

$$\text{Klasifikacijska točnost} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.6}$$

Testiranje je pokazalo, da je povprečna klasifikacijska točnost odstranjenega ozadja $89,1\%$, vendar pa je povprečna preciznost samo 34%. Vrednosti so prikazane v tabeli 5.3. S poobdelovanjem slike se je točnost poslabšala, kar je nasprotno od pričakovanega, saj se je pri testiranju z učno množico pokazalo, da izboljša točnost. Kljub temu pa je pomembno, da se poobdelava izvede, saj pripomore k boljši postavitvi avtomatično vrisanega krogca. Razlike v točnosti pred poobdelavo in po njej prikazuje graf na sliki 5.5. Kljub slabi preciznosti pa pri nadaljnjih modelih vseeno dobimo natančne rezultate.

5.3.2 Avtomatsko označevanje polipov

Avtomatsko označevanje polipov z vrisanimi krogci je bilo testirano nad 42 ročno označenih slikah, ki niso bile del učne množice. Da je bil avtomatsko vrisani krogec zaznan kot TP (podrobno v tabeli: 5.4), se je moral dotikati vsaj enega ročno vrisanega krogca. Če se več avtomatsko vrisanih krogcev



Slika 5.5: Slika prikazuje graf, na katerem so povprečne točnosti vseh testnih slik.

dotika ročno vrisanega krogca, se kot pravega izbere tistega, ki je najbližje ročno vrisanemu krogcu. Vsak krogec je lahko samo enkrat zaznan. Avtomatsko vrisani krogci, ki so ostali nezaznani, se štejejo kot FP. Neznane ročno vrisane krogce pa štejemo kot FN. Pri takšnem testiranju obstaja možnost, da sta krogca na enakem polipu, vendar se ne dotikata zaradi drugačne postavitve avtomatsko vrisanega krogca kot ročno vrisanega. V tem primeru se namesto TP štejeta dve napaki: FP in FN. To težavo omilimo tako, da krogcem pri testiranju povečamo njihov radij na: radij $= 2 \times$ radij. S tem se krogci, ki so na istem polipu, dotikajo drug drugega, čeprav se pred tem niso. Ker model predpostavi, da je na njegovi vhodni sliki samo območje polipov in označuje samo polipe, ne moremo definirati TN. Zato klasifikacijske točnosti ni bilo mogoče izračunati. Na vsaki posamezni sliki smo izračunali:

- senzitivnost (5.3), ki nam pove razmerje med pravilno avtomatsko označenimi in avtomatsko neoznačenimi polipi, ter
- preciznost (5.5), ki nam pove, koliko avtomatično označenih polipov je

Kratica	opis
TP	število pravilno avtomatsko označenih polipov
TN	ga ne moremo definirati
FP	število nepravilno avtomatsko označenih polipov
FN	število nezaznanih polipov

Tabela 5.4: Definicija TP, TN, FP in FN pri testiranju modela za avtomatsko označevanje polipov.

	Rezultati
Senzitivnost	79%
Preciznost	68%
F1 score	72%

Tabela 5.5: Tabela prikazuje povprečne vrednosti izračunov pri testiranju modela za avtomatsko označevanje polipov.

dejansko pravilno označilo polip.

S senzitivnostjo in preciznostjo smo lahko izračunali F1 score (5.7), ki nam pove harmonično povprečje senzitivnosti in preciznosti.

$$\text{F1 score} = \frac{2TP}{2TP + FP + FN} \tag{5.7}$$

Povprečni rezultati vseh slik so prikazani v tabeli 5.5.

5.3.3 Rezultati napovedovanja števila polipov na sliki

Število ročno označenih polipov na posamezni sliki predstavlja pravo vrednost števila polipov na sliki. Rezultat regresijskega modela pa predstavlja napovedano vrednost števila polipov na sliki. Za izračun napake smo uporabili $MAPE$ (5.9). Pri vsaki testni sliki smo izračunali napako po enačbi (5.8)

in nato iz celotne testne množice dobili povprečno napako (5.9). Da lahko rezultate ocenimo kot uspešne, moramo primerjati regresijski model za napovedovanje števila polipov na sliki s številom avtomatsko označenih polipov. Če s številom avtomatsko vrisanih krogcev dobimo manjšo napako od regresijskega modela, bi bil model neuspešen, saj bi poslabšal rezultat. Pri avtomatsko označenih polipih je bila izračunana povprečna napaka 0.30, pri modelu napovedi pa 0.32. $MAPE$ nam pove, da regresijski model ni izboljšal napovedi, saj ima večjo povprečno napako. Da bi bolje razumeli napako, naredimo še histogram napak za avtomatsko označene polipe (slika 5.6) in regresijski model (slika 5.7).

Vsak histogram ima 21 intervalov dolžine 0.1 na skupnem intervalu $[0, 2]$. Posamezni razred pa prikazuje frekvenco razreda, to je vsota napak, ki pripadajo istemu razredu. Ena napaka predstavlja eno sliko iz testne množice. Primerjava histogramov avtomatsko vrisanih krogcev in regresijskega modela pokaže, da je regresijski model izboljšal napoved za večino slik, ki so imele napako večjo od 0.4. Pri tem pa imamo z regresijskim modelom manj slik z napako 0.2 in več slik z napako 0.3 ter eno veliko napako, ki je 4.04, zato je tudi $MAPE$ regresijskega modela slabši od avtomatsko vrisanih krogcev. Kljub temu je regresijski model zmanjšal razpon napak in je zaradi tega pozitivno ocenjen. Slika, ki je bila pri regresijskem modelu ocenjena z napako 4.04 in je imela pri avtomatsko vrisanih krogcih napako samo 0.96, je bila zelo slabe kakovosti, kot je prikazano na sliki 5.9.

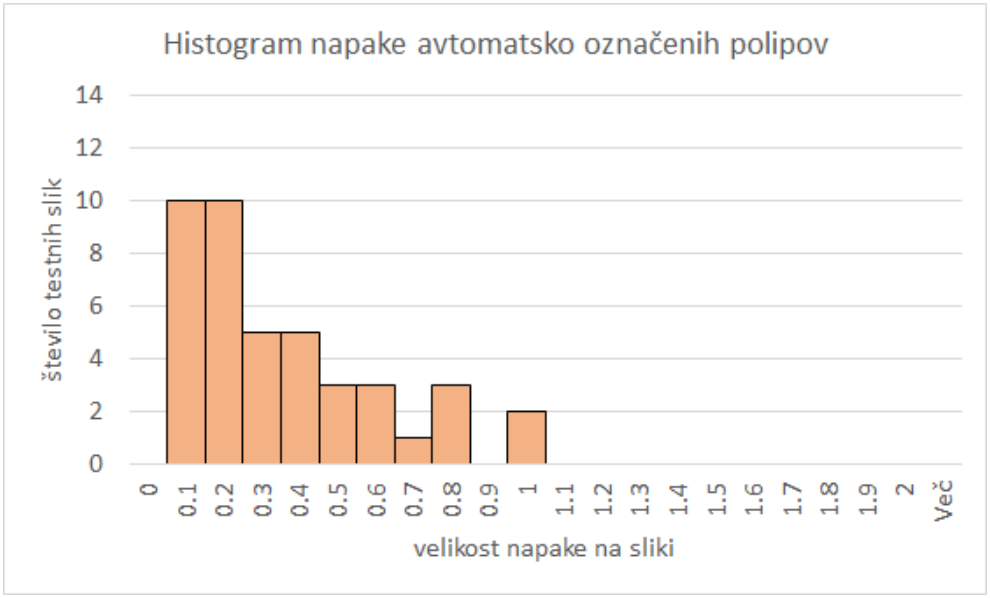
Da se izognemo slabim rezultatom, je treba pri štetju polipov odstraniti slike s slabo kakovostjo, kar bi pri ročnem štetju naredil tudi človek. Model tudi ne deluje dobro nad slikami s črnimi polipi (primer slika 5.8), zato je tudi te odsvetovano uporabljati pri štetju polipov. Po odstranitvi slike s slabo kakovostjo in slike s črnimi polipi iz testne množice slik dobimo $MAPE$ za avtomatsko vrisane krogce 0.27 in regresijski model 0.21, kar je zdaj pozitivni rezultat. Rezultati so prikazani v tabeli 5.6.

$$\text{napaka} = \left| \frac{\text{prava vrednost} - \text{napovedana vrednost}}{\text{prava vrednost}} \right| \tag{5.8}$$

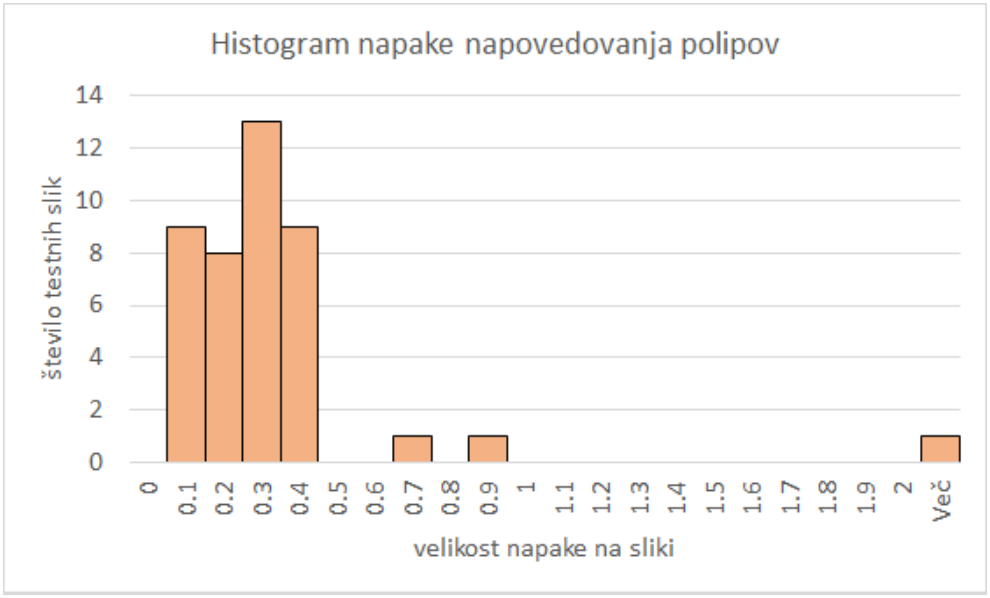
Kratica	`Vrednost`
$MAPE$ `modela za avtomatsko označevanje polipov`	0.20
$MAPE$ `modela za napovedovanje števila polipov`	0.32
$MAPE$ `modela za avtomatsko označevanje polipov po odstranitvi slabih slik`	0.27
$MAPE$ `modela za napovedovanje števila polipov po odstranitvi slabih slik`	0.21

Tabela 5.6: Tabela prikazuje povprečne vrednosti izračunov pri testiranju modela za avtomatsko označevanje polipov in modela napovedovanja števila polipov.

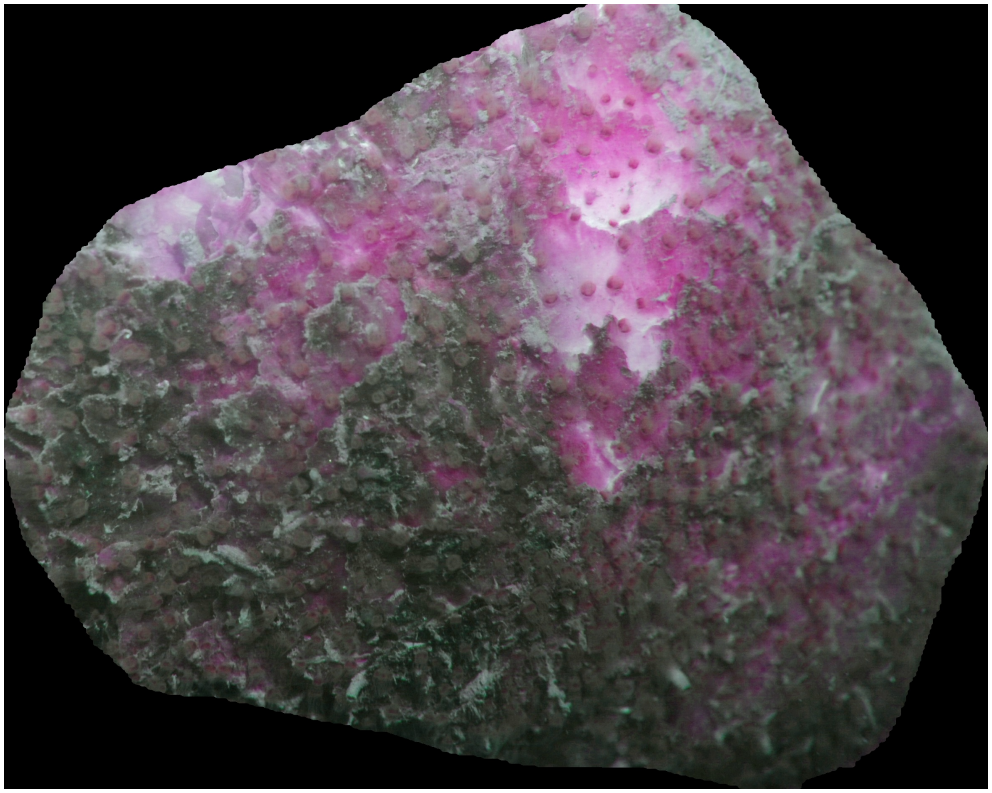
$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{\text{prava vrednost}_t - \text{napovedana vrednost}_t}{\text{prava vrednost}_t}\right| \tag{5.9}$$



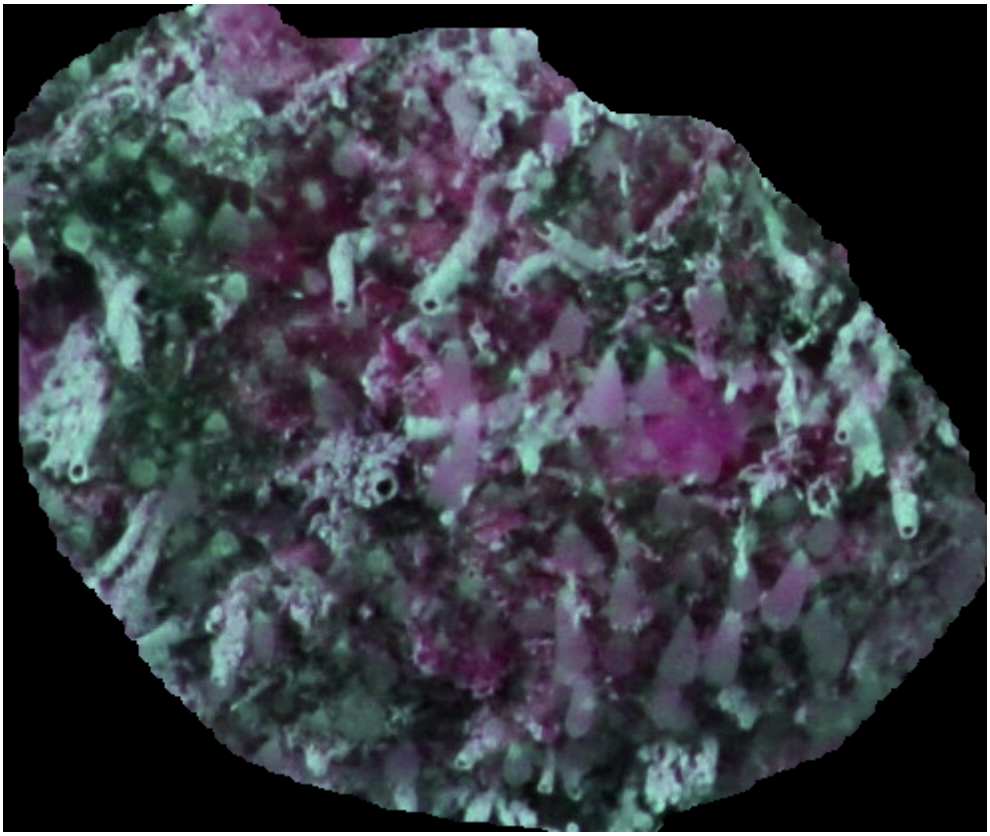
Slika 5.6: Slika prikazuje histogram modela avtomatičnega vrisovanja krogcev, ki kaže frekvenco napak slik posameznega intervala.



Slika 5.7: Slika prikazuje histogram modela za napovedovanje polipov, ki kaže frekvenco napak slik posameznega intervala.



Slika 5.8: Slika prikazuje primer črnih polipov.



Slika 5.9: Slika prikazuje primer slabe kakovosti slike za napoved števila polipov na sliki.

Poglavje 6

Sklepne ugotovitve

Cilj diplomske naloge je bilo delno avtomatsko štetje polipov na sliki, ki bi pripomoglo k temu, da se hitreje opravi študija za napovedovanje števila meduz v morju. Za ogrodje smo dobili delo [9], ki je imel že narejen sistem za avtomatsko štetje polipov. Ta sistem smo nato poskušali izboljšati in nadgraditi. Popolnoma avtomatičnega sistema nismo dosegli, saj mora uporabnik na začetku ročno obrezati sliko, da se izognemo večjim napakam sistema.

Naučili smo model, da sam odstrani ozadje na sliki, tako ni več treba klikati primerov ozadja in polipa. Omogočili smo ročno popravljanje napačno zaznanih polipov. Naredili smo model, ki na osnovi podanih avtomatsko zaznanih polipov in histograma slike napove število polipov in s tem še dodatno izboljša točnost. Preizkušali smo sistem na večjem številu slik in s tem izračunali točnost modelov. Ugotovili smo, da sistem ni učinkovit pri slikah s črnimi polipi in slikah s slabo kakovostjo ter s tem kvari natančnost skupnega štetja slik. Rezultati so pokazali, da z izločanjem slabih slik regresijski model napove število polipov s povprečno napako 21%.

Uporabnik lahko sistem uporablja kot pomoč pri štetju polipov ali pa kot napoved števila polipov. Za pomoč pri štetju polipov sistem avtomatsko označi polipe, ki jih nato uporabnik popravi. Povprečni F1 score je 72%, kar pomeni, da mora uporabnik ročno popraviti samo 28% polipov na sliki, kar je hitreje kot ročno označiti 100% polipov. Če se uporabnik odloči za napoved

števila polipov na sliki, mora obrezati sliko in nato prepustiti sistemu, da sliko obdela. Pri tem ne more vedeti, s kakšno natančnostjo je model napovedal število, in mu mora v celoti zaupati. Priporočljivo je, da se meri več slik skupaj zaporedoma, s čimer se dobi skupno število polipov, saj pri nekaterih slikah napove preveč polipov, pri nekaterih pa premalo. S tem se izniči napaka.

6.1 Možnosti za nadaljnje delo

Možnosti za izboljšavo je veliko. Že pri samem fotografiranju bi lahko potapljači bili pozorni glede kakovosti slik. Paziti je treba, da so slike ostre in svetle, a ne bleščeče. V vodi ne sme biti plavajočih delcev, ki bi se poznali na sliki, saj povzročajo nepotrebno napako pri odstranjevanju ozadja.

Namesto uporabe ravnil bi lahko uporabljali manjši vidni okrogli predmet. Glede na njegovo velikost bi izvedeli dolžino polipa, s tem pa bi hkrati omogočili avtomatično meritev slike, saj se okrogli predmeti enostavno in robustno zaznajo.

Modele bi lahko naučili s še večjo učno množico, saj se slike razlikujejo po strukturi in številu polipov na slikah. Primer so črni polipi — z uporabo več primerkov črnih polipov v učni množici bi lahko model deloval učinkovito tudi na črnih polipih. Hkrati bi tudi regresijski model za napovedovanje števila polipov pokril večje število učnih primerkov in s tem še bolj zanesljivo napovedal število polipov.

Pri avtomatskem označevanju polipov bi lahko dodali še dodaten atribut za klasificiranje polipa, saj se zdaj krogec vriše povsod, kjer ni ozadja. S tem bi izboljšali preciznost, še vedno pa bi imeli težave pri izpuščenih polipih.

Pri sami napovedi bi lahko izpustili korak avtomatskega označevanja polipov in napovedali število polipov glede na odstranjeno ozadje. Odstranjevanje ozadja ima boljšo točnost (klasifikacijsko točnost 89%) od avtomatskega označevanja polipov, ki ima F1 score 72%. Pri tem bi morali pretvoriti ploščino pikslov v ploščino metrov, da bi vedeli, koliko ploščine ima posame-

zni polip.

Za dejansko uporabo je treba urediti uporabniški vmesnik, da bi bil enostaven za uporabo. Diplomsko delo ni bilo namenjeno uporabniškemu vmesniku, ampak postopku obdelave slik, ki bi zaznaval in s tem tudi prešteval polipe na slikah.

Literatura

[1] Life cycle of a jellyfish. Slikovno gradivo. Dostopno na: `http://www.infovisual.info/02/013_en.html`. 2015.

[2] The Hypermedia Image Processing Reference. learning resources—Morphology, Closing, slikovno gradivo. Dostopno na: `http://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm`. 2015.

[3] Machine Learning Group at the University of Waikato. Data Mining Software in Java. Dostopno na: `http://www.cs.waikato.ac.nz/ml/index.html`. 2015.

[4] Wilhelm Burger and Mark J Burge. *Digital image processing: an algorithmic introduction using Java*. Springer Science & Business Media, 2009. pogl. 1, 4, 5, 6.

[5] Lijun Ding and Ardeshir Goshtasby. On the canny edge detector. *Pattern Recognition*, 34(3):721–725, 2001.

[6] David Flanagan. *Java in a Nutshell*. "O'Reilly Media, Inc.", 2005. pogl. 1.

[7] Bill Green. Canny edge detection tutorial. Dostopno na: `http://sites.google.com/site/setiawanhadi2/1CannyEdgeDetectionTutorial.pdf`. 2015.

[8] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[9] Vito Janko. Automated polyp identification and counting. *Machine learning course*, 2014. Seminar work, FRI, 2014.

[10] David M. Lane. Introduction to Linear Regression. Dostopno na: `http://onlinestatbook.com/2/regression/intro.html`. 2015.

[11] Douglas C Montgomery, Elizabeth A Peck, and G Geoffrey Vining. *Introduction to linear regression analysis*, volume 821. John Wiley & Sons, 2012. pogl. 2, 3.

[12] OpenCV. Open source computer vision. Uradna spletna stran OpenCV in dokumentacija knjižnjice. Dostopno na: `http://opencv.org/`. 2015.

[13] Werner Randelshofer. Monte media library. Dostopno na: `http://www.randelshofer.ch`. 2015.

[14] Mira Trebar. Porazdeljene strukture. Učno gradivo predmeta Logika in sistemi, Fakulteta za računalništvo, Ljubljana. Dostopno na: `www.fri.uni-lj.si/file/102663/nevronske-mreze.pdf/`. 2015.

[15] Štruc Vitomir. Ovrednotenje sistemov za samodejno razpoznavanje obrazov. Diplomsko delo, Fakulteta za elektrotehniko, Ljubljana, 2005.

[16] Jinsong Leng Zhihu Huang. Analysis of Hu's Moment Invariants on Image Scaling and Rotation. *2nd International Conference on Computer Engineering and Technology*, 7, 2010. 2015.

[17] GNU General Public Licence, dostopno na: https://www.gnu.org/copyleft/gpl.html, 2015