Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01664/2010
Datum: 05.04.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **DOMEN TABERNIK**

Naslov: **OPISOVANJE VIZUALNIH KATEGORIJ Z ATRIBUTI**
DESCRIBING VISUAL CATEGORIES BY ATTRIBUTES

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Proučite problem opisovanja vizualnih kategorij z atributi. Implementirajte in s poskusi ovrednotite pristop, ki ga je predlagal Farhadi s sodelavci [Computer Vision and Pattern Recognition 2009]. Na osnovi analize originalnega pristopa predlagajte možne izboljšave ter jih preverite na standardnih slikovnih podatkovnih zbirkah.

Mentor:
prof. dr. Aleš Leonardis

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

Dekan:
prof. dr. Franc Solina

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Domen Tabernik

Opisovanje vizualnih kategorij z atributi

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: prof. dr. Aleš Leonardis

Ljubljana, 2010

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01664/2010
Datum: 05.04.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **DOMEN TABERNIK**

Naslov: **OPISOVANJE VIZUALNIH KATEGORIJ Z ATRIBUTI**
DESCRIBING VISUAL CATEGORIES BY ATTRIBUTES

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Proučite problem opisovanja vizualnih kategorij z atributi. Implementirajte in s poskusi ovrednotite pristop, ki ga je predlagal Farhadi s sodelavci [Computer Vision and Pattern Recognition 2009]. Na osnovi analize originalnega pristopa predlagajte možne izboljšave ter jih preverite na standardnih slikovnih podatkovnih zbirkah.

Mentor:
prof. dr. Aleš Leonardis

Dekan:
prof. dr. Franc Solina

Fakulteta za računalništvo in informatiko
2.
UNIVERZA V LJUBLJANI

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani/-a Domen Tabernik,

z vpisno številko 63050118,

sem avtor/-ica diplomskega dela z naslovom:

Opisovanje vizualnih kategorij z atributi

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom prof. dr. Aleš Leonardis
- 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 v zbirki "Dela FRI".

V Ljubljani, dne 22.06.2010 Podpis avtorja/-ice:

Zahvala

Zahvalil bi se mentorju prof. dr. Alešu Leonardisu za vodenje in pomoč pri izdelavi diplomske naloge. Prav tako bi se zahvalil še Sanji Fidler za strokovno pomoč in nasvete pri pisanju diplome ter dr. Markotu Bobnu za tehnično pomoč pri uporabi modela LHOP.

Posebna zahvala gre tudi mojim staršem, ki so me tekom študija brezpogojno podpirali tako finančno kot tudi moralno.

Kazalo

Seznam uporabljenih kratic in simbolov

LHOP — naučena hierarhija delov (ang. *Learned Hierarchy of Parts)*

HOG — histogram orientiranih gradientov (ang. *Histogram of Oriented Gradients*)

ANN — aproksimirani najbližji sosed (ang. *Approximate Nearest Neighbors*)

FLANN — hitra aproksimacija najbližjega soseda

fMRI — funkcionalna magnetna resonanca

DOG — razlika gausov (ang. *Difference of Gaussians*)

MSER — maksimalno stabilne ekstremne regije (ang. *Maximally Stable Extremal Regions*)

SIFT — vizualni deskriptor (ang. *Scale-Invariant Feature Transform*)

SURF — vizualni deskriptor (ang. *Speeded Up Robust Features*)

HMAX — standardni model vizualnega prepoznavanja objektov modeliran po bioloških sistemih

SVM — algoritem strojnega učenja z uporabo podpornih vektorjev

ROC — krivulja ROC (ang. *Receiver Operating Characteristic*)

AUC — ploščina pod krivuljo ROC

Povzetek

V diplomski nalogi se osredotočamo na enega izmed težjih problemov s področja računalniškega zaznavanja, tj. splošna kategorizacija objektov. V uvodu natančneje predstavljamo problem ter možne rešitve, ki so se izoblikovale skozi zadnjih 30 let, nato pa se osredotočimo zlasti na enega izmed najnovejših pristopov do obravnavanega problema, kot so si to zamislili Ali Farhadi idr. Sistem, ki so ga predlagali, vsakemu objektu pripiše določene semantične in diskriminativne atribute, na osnovi katerih lahko naredimo dobro kategorizacijo objektov. Poleg tega pa smo opazili, da sistem omogoča še veliko drugih prednosti, kot so hitrejše učenje večjega števila kategorij, ugotavljanje nenavadnih lastnosti objektov ter celo učenje in kategorizacija objektov na podlagi le besednega opisa. Zaradi vseh teh prednosti smo mnenja, da je predlagani pristop korak v pravilno smer, zato smo v tej nalogi podrobneje predstavili koncept učenja z atributi. Ponovili smo tudi nekatere eksperimente z našo različico implementacije in pri tem pokazali, da dobimo podobne rezultate kot Ali Farhadi idr. Pri tem smo tudi opazili, da bi bilo smiselno vpeljati še nekaj dodatnih izboljšav, kot je spremenjena implementacija diskriminativnih atributov ter uporaba ločene lokalizacije, s katero omogočimo boljše učenje semantičnih atributov na eni strani ter boljše učenje kategorij na drugi strani. Naše izboljšave so se v skoraj vseh primerih izkazale za uporabne, kar smo tudi podprli z ustreznimi eksperimenti. Za atributni sistem smo tudi opazili, da ima določene skupne lastnosti z metodo LHOP, zato smo oba združili, tako da smo dele LHOP do 3. nivoja uporabili v bazni značilki namesto deskriptorja za robove in HOG-a ter pokazali, da se obnese podobno dobro kot originalna metoda atributnega učenja. Vendar je pri tem še vedno veliko prostora za izboljšave, zato smo na koncu predlagali nekaj možnih poti za nadaljnje raziskave.

Ključne besede:
Računalniško zaznavanje, kategorizacija objektov, atributi, strojno učenje, kategorije.

Abstract

This thesis focuses on one of the most challenging problems in the field of computer vision, i.e. general object recognition. In the introduction we first delineate the problem and possible solutions that have been formulated over the past thirty years. We proceed by concentrating on mostly one of the latest approaches introduced by Ali Farhadi et al, who have suggested a system which ascribes certain semantic and discriminative attributes to each object, which then act as a basis for performing quite satisfactory object recognition. We observe that this system has many additional advantages such as faster learning of greater number of categories, reporting unusual object traits as well as even learning and recognizing objects on the basis of word description alone. Given this we believe that the suggested approach indicates a step into the right direction, therefore this thesis offers a more detailed presentation of the attribute learning concept. In addition, we repeated certain experiments with our version of implementation, demonstrating that the produced results are similar to those of Ali Farhadi et al. Doing this we observed that it would be logical to introduce additional improvements as for example transformed implementation of discriminative attributes and the use of separate localization, which on one hand enables better learning of semantic attributes, and on the other hand better category leaning. In practically every case, our improvements turned out to be useful, which we have supported with appropriate experiments. The attribute system also showed to have certain similarities with the LHOP method; therefore we decided to combine both the attribute system and the LHOP method by employing LHOP parts up to the third level in the base feature instead of the edge descriptor and HOG descriptor. This approach illustrated that it works similarly well as the original attribute learning method; however, in view of the fact that there is always room for improvements, we suggest some possible directions for future research.

Key words:
Computer vision, object recognition, attributes, machine learning, categories.

Poglavje 1

Uvod

Odkar so se računalniki začeli v drugi polovici 20. stoletja vse pogosteje uporabljati, so strokovnjaki poskušali poiskati različne algoritme in matematične modele za to, kar je človeku skoraj samoumevno, vizualno zaznavanje in prepoznavanje objektov. Področje, ki zajema te raziskave, lahko označimo z besedno zvezo vizualno računalniško zaznavanje, vendar pri tem ne mislimo zgolj na zgoraj omenjeno zaznavanje oz. prepoznavanje objektov, temveč bi lahko rekli, da zajema veliko širše področje analize in obdelave 2D, 3D in 4D signalov z namenom pridobivanja dodatnih informacij. To področje se je skozi leta izkazalo za interdisciplinarno, saj se zaradi svoje narave problem močno prekriva in povezuje tako s tehničnimi kot tudi z netehničnimi disciplinami. Močne povezave tako obstajajo s področji, kot so analiza slik, digitalno procesiranje signalov in strojno učenje na eni strani, ter v zadnjem času tudi s področjij kot sta psihologija in nevroznanost na drugi strani.

Čeprav sega začetek raziskav več kot 30 let nazaj, pa je do ogromnega napredka prišlo šele v zadnjih desetih letih s pojavom vse močnejših računalnikov in vse bolj naprednih algoritmov. Posledično se v zadnjem času pojavlja vse več aplikacij z raznovrstnih področij, ki izkoriščajo te najnovejše napredke. Tako je v nekaterih panogah industrije mogoče videti različne tehnološko napredne proizvodne linije, ki se za učinkovito delovanje močno zanašajo na računalniški vid. Predvsem pa se tehnologijo že nekaj časa uporablja za nadzor kakovosti izdelkov [33, 34], saj s tem podjetja močno pridobijo na dodani vrednosti. Tehnologija se že kar nekaj časa uporablja tudi na področju nadzora in varnosti [35, 36]. Tam obstaja ogromna količina posnetkov, ki jih človek ni sposoben hitro analizirati, s pomočjo algoritmov računalniškega zaznavanja in močnih računalnikov pa je možno te podatke hitro predelati in analizirati. Tudi veliko naprav za identifikacijo oz. avtorizacijo, kot so zaz-

nava prstnih odtisov, roženice ali obraza [12, 13], ne bi bilo brez naprednih algoritmov s tega področja. Vse večja je uporaba tudi v eni izmed najmanj fleksibilnih industrij, avtomobilski industriji, kjer je v serijski proizvodnji že možno dobiti vozilo, ki je sposobno zaznati prometne znake in ustrezno opozoriti voznika. Zaradi vse večje prisotnosti digitalnih fotoaparatov in kamer je prav tako mogoče videti veliko uporabnih aplikacij v vsakodnevnih napravah. Skoraj nemogoče je dandanes dobiti digitalni fotoaparat ali pametni telefon, ki ne bi bil sposoben zaznati obraza in ustrezno prilagoditi svojega delovanja za najboljšo kakovost slike. Nekateri od njih so celo sposobni zaznati nasmeh in posneti sliko v najboljšem trenutku. Poleg zgoraj omenjenih primerov obstaja še skoraj neskončna množica področij, kjer se uporablja računalniško zaznavanje, npr. medicina (fMRI – obdelava slik v pri postopku magnetne resonance), šport (analiza posnetkov), robotika (avtonomna vozila), vojska (nadzor, pametne bombe), zabavna industrija (prepoznavanje obraza na slikah) itd. Glede na obseg različnih področij bi lahko rekli, da skoraj ne obstaja področje na katerega raziskave iz računalniškega zaznavanja ne bi imele vpliva.

Navkljub vse večji uporabi teh algoritmov je pojav omenjenih novih aplikacij le vrh ledene gore, saj na tem področju še vedno obstaja veliko odprtih problemov. Z rešitvijo teh problemov tehnologija računalniškega zaznavanja obljublja še več sprememb. Mnogo raziskav tako poteka na področju uporabniških vmesnikov, kjer veliko obetata tehnologiji sledenja pogleda (ang. *gaze-tracking*) [31, 32] in prepoznave gest [37, 38]. Mnogo raziskav pa poskuša rešiti enega izmed težjih problemov tega področja, splošno kategorizacijo objektov [6, 7]. V zadnjih desetih letih se je sicer veliko pozornosti usmerilo v prepoznavanje specifičnih kategorij kot, so vozila [39, 40, 41] in pešci [14], zato obstajajo zelo dobri detektorji za ti dve kategoriji, vendar pa so se detektorji za bolj splošne kategorije izkazali za veliko bolj kompleksne.

1.1 Problem splošne kategorizacije

Na področju računalniškega zaznavanja pod problemom *splošne kategorizacije* razumemo postopek, pri katerem iz vizualnega signala prepoznamo oz. zaznamo objekt ter mu nato pripišemo eno ali več kategorij. Splošna kategorizacija se popolnoma razlikuje od problema *identifikacije*, kjer se iz učnega signala poskuša naučiti specifičen objekt oz. instanco objekta ter ga nato kasneje prepoznati in identificirati. V nasprotju z identifikacijo pa pri problemu kategorizacije poskušamo prepoznati tudi objekte, ki jih sicer nismo še nikoli videli, smo pa zato videli druge podobne primere, ki pripadajo isti kategoriji.

Primer identifikacije bi bil, da iz slike zaznamo le obraz Janeza, medtem ko pri problemu kategorizacije iz slike zaznamo katerikoli obraz. V tej diplomski nalogi se bomo ta problem imenovali tudi le kot *kategorizacija*.

S problemom splošne kategorizacije se vsak od nas srečuje vsakodnevno in ga pri tem rešujemo brez truda ter velikokrat tudi ne da bi se zavedali. Rešujemo ga pri vožnji, ko prepoznamo velik tovornjak med avtomobili, rešujemo ga, ko želimo vstopiti v stavbo in znamo razlikovati med vrati in okni, rešujemo ga, ko se želimo usesti in prepoznamo stol, praktično ga rešujemo skoraj na vsakem koraku. Človeški vizualni sistem je pri svojem delu tako zelo učinkovit, da za kategorizacijo večini ljudi ni potrebno niti malo napora. Zato se zdi presenetljivo, da nam kljub več kot 30 letom raziskovalnega napora še vedno ni uspelo popolnoma rešiti tega problema. Izkaže se, da je problem veliko težji, kot se sprva zdi. Njegovo kompleksnost si je delno mogoče razlagati z okolico, v kateri mora sistem delovati. Ta okolica je v realnih pogojih zelo kompleksna in močno spremenljiva. Veliko vlogo igra pri tem osvetljenost oz. svetlobni pogoji, ki so zelo nepredvidljivi in se lahko v realnem okolju spreminjajo iz minute v minuto. Realno okolje je velikokrat tudi posejano z velikim številom različnih objektov, ki se lahko med sabo zakrivajo, in s tem močno otežujejo prepoznavanje. Poleg tega je realno okolje skoraj vedno tridimenzionalno, vizualni sistem pa običajno gleda iz določene točke, in zato vidi le dve dimenziji oz. v najboljšem primeru 2,5 dimenzije, kadar poznamo tudi globino. Zaradi tega mora biti sistem sposoben prepoznati objekt iz veliko različnih perspektiv, kar prinese še dodatne težave. Vsaka rešitev za problem kategorizacije, ki deluje v realnih pogojih, mora tako upoštevati zgoraj omenjene težave iz okolice, s tem pa se kompleksnost problema dramatično poveča.

Prve rešitve tega problema so se začele pojavljati v 1970-ih z uveljavitvijo prvih prototipnih modelov [42, 43, 44], ki so bili objektno usmerjeni ter so sloneli na volumetričnih telesih, kot so posplošeni cilindri, geoni in superkvadratniki. Primer tega modela je Brooksov sistem ACRONYM [49] iz leta 1973. Celoten koncept teh sistemov temelji na hierarhičnih modelih, ki abstrahirajo določeno kategorijo ter so sestavljeni iz volumetričnih teles. Tako npr. obstaja en hierarhični model za vse različne variacije lončkov za kavo. Gledano s stališča teorije je bil model povsem dober, vendar je vseboval preveč predpostavk in omejitev. Le-te so se nanašale tako na same objekte, ki so morali biti enostavni in brez tekstur, kot tudi na kontrolirane pogoje osvetlitve. Zaradi teh predpostavk model ni bil primeren za uporabo v realnih okoljih, ampak le v primerih, kjer je bilo okolico mogoče nadzorovati.

Naslednji korak proti boljši kategorizaciji objektov je prišel v 1980-ih s pojavitvijo 3D modelov CAD [45, 46, 47, 48], ki so zamenjali predhodni koncept

hierarhičnih teles. Posledično se je s tem zmanjšal razkorak med modelom in sliko, povečal pa se je poudarek na lokalnih značilnostih ter na ekstrakciji robov in kotov [1]. Zaradi tega je bilo sedaj mogoče kategorizirati tudi bolj kompleksne oblike, a to je prineslo dodatne težave ter še vedno so veljale omejitve za teksture. Dodatne težave so bile predvsem v vprašanju, kako zgraditi kompleksen 3D model, ga posplošiti za celotno kategorijo ter ta model ustrezno poiskati iz različnih pogledov in zakritih slik.

Navkljub izboljšavam je bil ta sistem v praksi še vedno neuporaben, zaradi česar se je v 1990-ih začel uveljavljati nov pristop, ki temelji na osnovi videza (ang. *appearance-based*) [56]. Koncept izhaja iz spoznanja, da je treba modele popolnoma približati dejanskim slikam, zato se sedaj namesto robov in kotov uporablja vse slikovne elemente, te vrednosti pa se predstavi kot vektorje v podprostoru. Vsak objekt je predstavljen s skupino slik oz. vektorjev iz različnih perspektiv, prepoznavanje objekta pa poteka s primerjavo vektorjev vseh znanih objektov. Ta pristop sicer zmanjša razmik med modeli in dejansko sliko ter je uporaben tudi za zelo kompleksne objekte, vendar ima še vedno svoje težave. Za vsak objekt je treba naprej poznati sliko iz mnogo različnih pogledov ter z različno osvetlitvijo, kar je zaradi velikega števila različnih kombinacij zelo zahtevno. Zaradi tega ima sistem veliko problemov z robustnostjo, saj je objekte skoraj nemogoče zaznati v scenah z množico različnih objektov ter prav tako ima probleme pri zaznavaju delno zakritih objektov. Glavni problem z vidika kategorizacije pa je, da je sistem zmožen prepoznavati le tiste primerke objektov, ki jih je predhodno že videl ter znotraj modela ni sposoben zajeti velike variablinosti različnih oblik, ki so potrebne za predstavitev celotne kategorije.

Za zgoraj omenjeni sistem bi lahko rekli, da je usmerjen v globalni videz, saj za vsak objekt poznamo celotno sliko iz določene perspektive. Zaradi te lastnosti sistem ni dovolj robusten in ni povsem primeren za kategorizacijo, zato se je v zadnjih desetih letih veliko napora vložilo v sistem, osnovan na lokalnem videzu oz. lokalnih značilkah. Princip sloni na nadgradnji detektorjev kotov, razvitih v 1980-ih [1] tako, da bodo zaznali le tiste značilnosti (npr. kote), ki jih je mogoče videti iz različnih perspektiv. Te značilnosti poimenujemo tudi invariantne točke. Okolico teh točk se nato uporabi za izgradnjo lokalnih vektorjev oz. lokalnih značilk. Vsak objekt je sedaj predstavljen s skupino teh lokalnih značilk, kar naredi sistem veliko bolj robusten in primeren za kategorizacijo objektov. Poleg DOG, MSER [3] ter Kadir Brady detektor [4] je za detekcijo invariantnih točk še najbolj znan Harris-affine [2], za lokalno značilko pa se je zelo dobro izkazal SIFT [5]. S stališča kategorizacije se sedaj veliko truda vlaga v iskanje modelov, ki bi bili na osnovi teh bogatih značilk

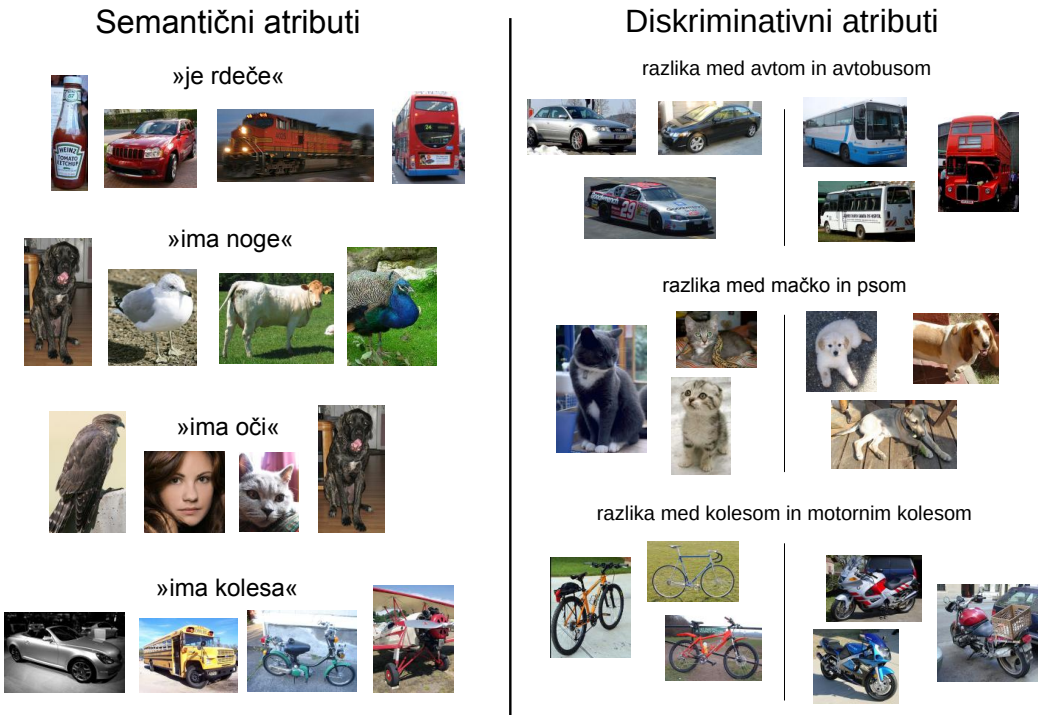
sposobni narediti dobro kategorizacijo. Nekateri, kot [8] in [9] so se usmerili v uporabo geometrijskih značilnosti in dodatno razširili starejše modele [10] v t. i. model ozvezdja (ang. *Constellation model*). Drugi [11, 15] so preizkušali različne kombinacije algoritmov strojnega učenja in lokalnih značilk. Veliko pozornosti se je usmerilo tudi v povezave z nevroznanostjo in s principi delovanja vizualnega sistema na bioloških osnovah [18]. Tako je eden izmed znanih modelov tudi HMAX [16], ki izhaja iz eksperimentalnih podatkov o preučevanju vizualnega korteksa opice makak (ang. *macaque*) [17] in podaja osnovo za bolj hierarhično usmerjene modele. Primer takega hierarhičnega modela je tudi LHOP [25], ki je podrobneje predstavljen v kasnejših poglavjih.

1.2 Atributi in kategorizacija objektov

Čeprav je v zadnjih desetih letih področje kategorizacije objektov močno napredovalo, še vedno ni prišlo do točke, kjer bi bili algoritmi pripravljeni za množično uporabo v praktičnih aplikacijah. Poti za nadaljnje raziskave so različne. Nekateri bi se radi usmerili predvsem v prototipne modele, kot so bili v 1970-ih in 80-ih, drugi so bolj za modele podobne biološkim vizualnim sistemom. V zadnjih letih se je pojavila še ena izmed možnih poti, za katero bi lahko rekli, da je delno blizu biološkim sistemom. Koncept sloni na spoznanju nekaterih raziskav, da se uspešnost kategorizacije izboljša, če upoštevamo dejstvo, da imajo nekatere kategorije med seboj skupne določene vizualne značilnosti [19, 20, 21]. Tako je npr. skupna značilnost kategorijama vozil in hiš, da imata okno, prav tako bi lahko rekli, da imajo skoraj vse mačke in psi, štiri noge, rep in dlako. V. Ferrari in A. Zisserman sta že v [22] poskusila te skupne značilnosti obravnavati kot enostavne vizualne atribute v obliki barv in vzorcev (rdeča, rumena, modra, črtasto, pikčasto itd.), vendar sta te atribute uporabila le za lokalizacijo objekta, in ne za kategorizacijo.

Šele v [23] so raziskovalci bolj splošne skupne značilnosti definirali kot *semantične atribute*, bolj diskriminativne oz. specifične značilnosti pa kot *diskriminativne atribute*. Prvi predstavljajo lastnosti, ki jih je enostavno identificirati, kot so: »ima rep«, »ima štiri noge«, »ima okno«, »ima kolesa« itd., medtem ko diskriminativni atributi predstavljajo lastnosti, ki jih ne moremo identificirati kot semantične atribute, vendar predstavljajo pomembno razliko med določenimi kategorijami. Primer diskriminativnih atributov bi bile lastnosti, ki jih imajo psi, vendar jih nimajo mačke. Nekateri primeri semantičnih in diskriminativnih atributov so prikazani na sliki (1.1).

Pokazali so tudi, da je lahko kategorizacija z uporabo takih atributov boljša



Slika 1.1: Različni primeri semantičnih (levo) in diskriminativnih (desno) atributov.

ali vsaj enako dobra, kot če bi se učili brez atributov. Glavni razlog za vpeljavo atributov pa ni le potencialno boljša kategorizacija, ampak so lahko veliko bolj pomembne druge prednosti in predvsem nove zmožnosti:

- učenje z majhnim številom učnih primerov
- delno inkrementalno učenje
- učenje večje množice kategorij
- učenje in prepoznavanje brez dejanskih slik (uporaba le besedilnega opisa)
- ugotavljanje nepričakovanih oz. nenavadnih atributov
- čeprav kategorije še ne poznamo, lahko še vedno nekaj uporabnega povemo o objektu (katere atribute ima itd.)

S tem ko imajo kategorije skupnih večino atributov, lahko pri učenju kategorij uporabimo veliko manj učnih primerov in obenem še vedno ohranimo dobro

klasifikacijsko natančnost. V [23] so vzeli le 4 učne primere in dobili isto klasifikacijsko točnost, kot če bi pri učenju brez atributov vzeli 20 primerov. Razlika v številu učnih primerov se pri uporabi več primerov le še povečuje v prid učenju z atributi. Poleg tega skupni atributi omogočajo, da se lahko naučimo večje število različnih kategorij, saj je pri dodajanju novih kategorij treba dodati le manjše število novih atributov. Po drugi strani pa to pomeni, da je učenje novih kategorij lahko delno inkrementalno, saj je dodajanje atributov neodvisno od ostalih atributov.

Velika prednost atributnega sistema je tudi v tem, da lahko nekatere atribute za učenje in klasifikacijo dobimo samo iz besednega opisa, medtem ko je v pri ostalih metodah nujno potrebna vsaj slika. To pomeni, da če imamo npr. opis: »ima vrata, ima okna, ima kolesa, je kovinsko«, lahko ta opis razbijemo v ustrezne semantične atribute in na osnovi teh atributov naredimo kategorizacijo. To lastnost so pokazali tako v [24] kot tudi v [23], kjer se izkaže, da je učenje z le besednim opisom ekvivalento učenju brez atributov z 8 učnimi primeri, ter učenju s semantičnimi in diskriminativnimi atributi s 3 učnimi primeri. Potencialna uporabnost te lastnosti je lahko velika; zelo primerna bi bila pri internetnih iskalnikih, kjer bi uporabnik lahko vnesel sliko vozila in ukaz »želim objekt, podoben tej sliki, vendar brez strehe«. Skupaj z ustrezno analizo naravnega jezika in atributnim učenjem objektov bi to bilo povsem izvedljivo.

Vpeljava atributov ima tudi lepo lastnost, da kadar zaznamo znan ali neznan objekt, lahko informacijo o atributih uporabimo za dodatne sklepe. Tako lahko v sliki prepoznamo vozilo, obenem pa zaznamo atribut »plastično«, iz česar bi bilo možno sklepati, da gre za igračo v obliki vozila. Na drugi strani pa bi lahko ugotovili, če kakšen atribut manjka, glede na prepoznano kategorijo. Primer tega bi lahko bil, ko zaznamo letalo, ne zaznamo pa atributa »letalsko krilo«. S temi dodatnimi informacijami lahko posledično povemo veliko več kot le kategorijo, kaj je lahko v pomoč bodisi uporabniku bodisi kakšnemu drugemu sistemu za analizo. Tudi v primeru, da nimamo nobenih informacij o kategoriji oz. zaznamo, da objekt ne spada v nobeno izmed znanih kategorij, lahko še vedno veliko povemo o objektu, ki ga vidimo, preko zaznanih atributov.

Vse te prednosti kažejo na dejstvo, da je splošna kategorizacija z vizualnimi atributi veliko močnejša od konvencionalnih pristopov, zato se bomo v tej diplomski nalogi osredotočili na koncept atributnega učenja, kot so si ga zamislili v [23]. Podrobneje bomo predstavili sam koncept ter našo implementacijo, kjer bomo poskušali vpeljati še nekaj dodatnih izboljšav, kot je uporaba ločene lokalizacije. Metodo učenja z atributi bomo poskušali tudi delno združiti s hierarhičnim modelom LHOP ter preveriti, kako dobro se ob-

neseta. Pri tem bomo vpeljali tudi nove diskriminativne atribute, ki se v naši implementaciji obnesejo bolje od že obstoječih.

1.3 Organizacija diplomske naloge

Glavni del diplomske naloge je sestavljen iz treh poglavij. V prvem poglavju bomo podrobneje predstavili metodo učenja vizualnih kategorij z atributi, specifično bomo tudi definirali semantične in diskriminativne atribute in izpostavili na kakšne težave lahko naletimo pri uporabi atributov ter kako jih lahko rešimo. V tem poglavju bomo tudi podrobneje predstavili našo implementacijo atributnega sistema. Specifično bomo izpostavili tudi nekaj izboljšav, ki bi lahko še dodatno izboljšale kategorizacijo. V drugem poglavju bomo podrobneje predstavili model LHOP ter kako bi ga lahko integrirali v atributni sistem. V zadnjem poglavju bomo predstavili rezultate eksperimentov originalnega atributnega sistema, s čimer bomo potrdili nekatere rezultate iz [23], predstavili pa bomo tudi rezultate atributnega sistema, kombiniranega z modelom LHOP, s katerim poskušamo še dodatno izboljšati kategorizacijo. Na koncu bomo v zaključku povzeli ugotovitve ter predstavili možne poti za nadaljnje raziskave.

Poglavje 2

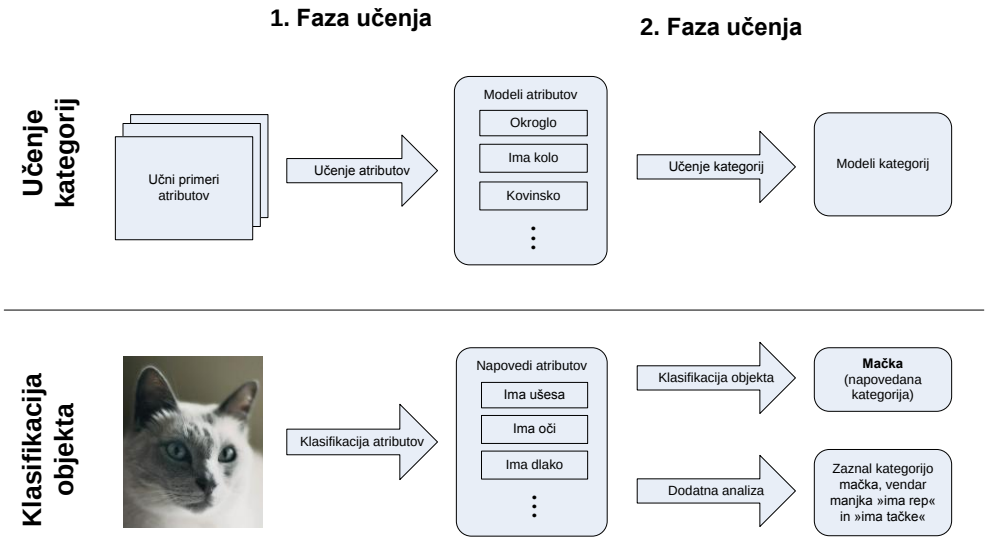
Učenje kategorij na podlagi atributov

2.1 Koncept atributnega učenja

Model učenja kategorij z atributi ne sloni več na tradicionalnem določanju vizualnih kategorij iz enostavnih značilk, ampak na opisovanju kategorij. To nam omogoča vpeljava vizualnih atributov, ki jih lahko definiramo bodisi kot semantične bodisi kot diskriminativne atribute. S temi atributi lahko sedaj kategorijo vozil opišemo, kot npr. »ima okna«, »ima vrata«, »ima ogledalo«, »ima kolesa«, »ima luči«, »je kovinsko« itd. Ko vemo, kateri atributi zastopajo določeno kategorijo, lahko to uporabimo za klasifikacjo objekta v sliki. Za objekt potrebujemo le seznam ustreznih atributov, ki jih zaznamo v sliki. Tak način klasifikacije pa pomeni, da se velik del odgovornosti prenese na atribute. Le-te moramo biti sedaj sposobni dobro zaznati v sliki, zato se jih moramo predhodno ustrezno naučiti.

Celoten model atributnega učenja lahko tako razdelimo v učenje ter klasifijacijo, kot je to prikazano na sliki (2.1). V učnem delu poteka učenje kategorij, ki ga lahko delimo v dve fazi. V prvi fazi se učimo vse različne semantične in diskriminativne atribute, v drugi fazi pa na osnovi teh atributov poteka učenje kategorij. V klasifikacij pa s pomočjo predhodno naučenih atributov in kategorij določamo vizualno kategorijo specifičnega objekta, ki ga zaznamo na sliki. Tudi klasifikacija poteka v dveh fazah, in sicer v prvi fazi klasificiramo oz. zaznamo atribute v sliki ter iz tega naredimo ustrezen seznam napovedanih atributov, nato pa na osnovi tega seznama poiščemo najbolj ustrezno kategorijo. V drugi fazi lahko poleg klasičnega določanja kategorije opravimo tudi dodatno analizo napovedanih atributov, ki lahko nato poda

veliko več informacij o objektu na sliki.



Slika 2.1: Prikaz postopka učenja kategorij in klasfikacije objekta s pomočjo atributov.

2.1.1 Učenje kategorij in klasifikacija objektov

Kot je bilo že omenjeno, učenje kategorij poteka v dveh fazah. V prvi fazi poteka učenje semantičnih in diskriminativnih atributov, kar se lahko naredi s poljubnim algoritmom za strojno učenje. Ta algoritem se požene za vsak atribut ločeno in načeloma lahko za učne atribute uporabimo katerokoli zvrst značilk, ki jih izvlečemo s slike. Uporabimo lahko tako Harrisov detektor kotov kot tudi deskriptor SIFT ali SURF . Vendar je pri izbiri značilk treba upoštevati dejstvo, da se atributi med seboj močno razlikujejo, in zato atributov, ki se nanašajo na material ali teksturo, ni mogoče zaznati le z značilkami za obliko. Prav tako ni mogoče zaznati oblike objekta le z značilkami za teksturo. Zaradi tega ni primerno uporabiti le ene vrste značilk, ampak več različnih vrst. V [23] uporabijo štiri različne tipe značilk: histogram barve, značilke za teksturo, HOG in robove.

V drugi fazi učenja se nato naučimo kategorije, tako da uporabimo želeni algoritem strojnega učenja, za učne atribute pa uporabimo semantične in diskriminativne atribute. To pomeni, da če imamo 64 semantičnih in diskriminativnih atributov, potem za vsak objekt dobimo 64 binarnih učnih atributov,

ki označujejo, katere atribute vsebuje objekt. Na osnovi teh učnih algoritmov nato poženemo strojno učenje, ki nam vrne ustrezen model.

Učenje novih kategorij lahko poteka delno inkrementalno. V prvi fazi moramo naučiti le nove semantične in diskriminativne atribute, ki se pojavijo z novimi kategorijami in jih predhodno nismo poznali. Starih atributov pri tem ni treba ponovno učiti, vendar lahko s ponovnim učenjem malenkost izboljšamo tudi točnost le-teh. Drugo fazo učenja je običajno smiselno ponoviti skupaj s starimi kategorijami, v večini pa je to odvisno od izbranega algoritma strojnega učenja, saj mora le-ta podpirati inkrementalno učenje. V vsakem primeru pa je smiselno učenje v 2. fazi popolnoma ponoviti tudi za obstoječe kategorije, če je bil dodan nov atribut, saj je mogoče, da prav ta novi atribut izboljša uspešnost ostalih kategorij.

Določanje kategorije novega objekta oz. klasifikacija objekta poteka na podlagi naučenih modelov. Z vsemi modeli semantičnih in diskriminativnih atributov napovemo seznam atributov, za katere algoritmi strojnega učenja predvidevajo, da se nahajajo na sliki. Na podlagi tega seznama in naučenega modela kategorij lahko nato algoritem strojnega učenja napove ustrezno kategorijo. Pri tem velja opomniti, da se lahko seznam semantičnih atributov pridobi tudi brez slike, pri čemer lahko atribute poda eksplicitno sam uporabnik ali pa se jih izlušči iz besednega opisa. Za diskriminativne atribute to običajno ni mogoče, in zanje potrebujemo vizualni signal. Poleg kategorije lahko iz seznama atributov razberemo tudi druge lastnosti objekta, kot so zaznani nenavadni atributi, pomanjkanje nekaterih atributov itd.

2.1.2 Semantični atributi

Kot semantične atribute predstavimo vse tiste značilnosti objektov, ki so skupne različnim kategorijam, oz. tudi vse značilnosti, ki sicer pripadajo le eni kategoriji, vendar so vizualno zelo razpoznavne, in jih je zato mogoče enostavno definirati. Te značilnosti so lahko zelo enostavne lastnosti, kot sta na primer barva in oblika. Primer teh bi lahko bili »je rdeče«, »je modro«, »je kvadratasto« ali »je okroglo«. Značilnosti se lahko nanašajo tudi na bolj kompleksne lastnosti, kot so material (npr. »kovinsko, »svetlikajoče«), zapletene teksture (npr. »progasto«, »pikčasto«), ali pa celo na bolj kompleksne dele. Primeri takih kompleksnih delov bi lahko bile značilnosti, kot so »ima rep«, »ima ušesa«, »ima okno«, »ima kolo« itd. Semantične atribute tako dobimo iz značilnosti, ki jih lahko razdelimo v tri skupine: oblika, material in kompleksen del. Vsak objekt ima tako enega ali več semantičnih atributov, in na osnovi teh atributov nato slonijo vsi postopki učenja in kategorizacije, zato je

zelo pomembno, da poznamo vse različne atribute, ki so pomembni za vsako kategorijo. V [23] predhodno definirajo 64 različnih atributov, ki so primerni za želene razrede.

Slika 2.2: Primer razreda »avto«, za katerega lahko najdemo pripadajoče semantične atribute: »ima kolesa«, »ima vrata«, »ima sprednje luči«, »ima stranska ogledala«, »ima okna«, »ima streho«, »je kovinsko«, »je modro«, »je podolgovato«, »je škatlasto«.

Učenje kategorij sloni na semantičnih atributih, in zato je zelo pomembno, da se te atribute čim bolj pravilno nauči. Na primer, če imamo atribut »ima kolo«, potem je z vidika semantike zelo pomembno, da se pri učenju atributov nauči dejanske značilnosti kolesa, in ne drugih značilnosti, kot na primer »kovinsko«. To se sicer sliši povsem logično, vendar lahko velikokrat naletimo na učno množico, kjer bi se lahko ti atributi pojavili istočasno, oz. lahko rečemo, da obstaja močna korelacija med dvema naučenima atributoma. Čeprav bi načeloma lahko poskrbeli, da se taki primeri izločijo iz učne množice, je včasih to nemogoče, ker potem ne bi imeli dovolj učnih primerov, zato je še posebno pomembno, da se tak problem reši pri učenju. Ta problem sicer ni tako izrazit, kadar so bili semantični atributi in kategorije naučene na isti učni množici ter z istimi kategorijami, saj se bodo korelirani atributi večinoma pojavljali skupaj. Vendar je rešitev tega problema še vedno pomembna, saj je ena od poti do boljše natančnosti kategorizacije tudi preko čim večje natančnosti semantičnih atributov. Problem pride do izraza šele, ko želimo naučiti nove kategorije, ki niso bile zajete v učno množico pri učenju semantičnih atributov, ter ko želimo čimvečjo natančnost pri učenju samo s tekstovnimi opisi. Slednje je s stališča uporabnika seveda zelo pomembno, saj ko uporabnik poda atribut »ima kolo«, po navadi ne misli na atribut »kovinsko«. Pri učenju novih kategorij imamo dejansko t. i. problem generalizacije, kar pomeni, da se morajo

atributi posplošiti, tako da so primerni tudi za učenje novih kategorij, ki jih pri učenju semantičnih atributov še ni bilo. Z rešitvijo problema generalizacije se lahko posledično nove kategorije naučimo veliko bolj uspešno in prav tako se lahko bolj uspešno naučimo tudi originalne kategorije. Ta problem v [23] rešujejo z vpeljavo postopka *izbiranje značilk*, ki je podrobneje predstavljen pri implementaciji.

2.1.3 Diskriminativni atributi

Pri semantičnih atributih se je treba zavedati, da običajno zajemajo le *skupne* značilnosti kategorij, in velikokrat ti atributi niso dovolj specifični, da bi lahko ločili med nekaterimi kategorijami. Če iz slike izvlečemo npr. atribute »ima dlako«, »ima rep«, »ima ušesa« in »ima štiri noge«, potem na podlagi tega ni mogoče enoznačno določiti kategorije, saj je ta opis preveč splošen in lahko zajema veliko različnih živali. Ne glede na to, koliko novih semantičnih atributov bi ustvarili, lahko v zgornji opis še vedno štejemo tako kategorijo »pes« kot tudi kategorijo »mačka« ali pa kakšno drugo podobno žival. Med nekaterimi kategorijami posledično še vedno ni mogoče najti semantičnega atributa, ki bi ga lahko označili z besedo in bi lahko ločil te kategorije, vendar pa vemo, da med njimi obstaja določena vizualna razlika. Ta problem je rešen z vpeljavo *diskriminativnih atributov*, ki zajemajo ravno te specifične vizualne podrobnosti raznih kategorij.

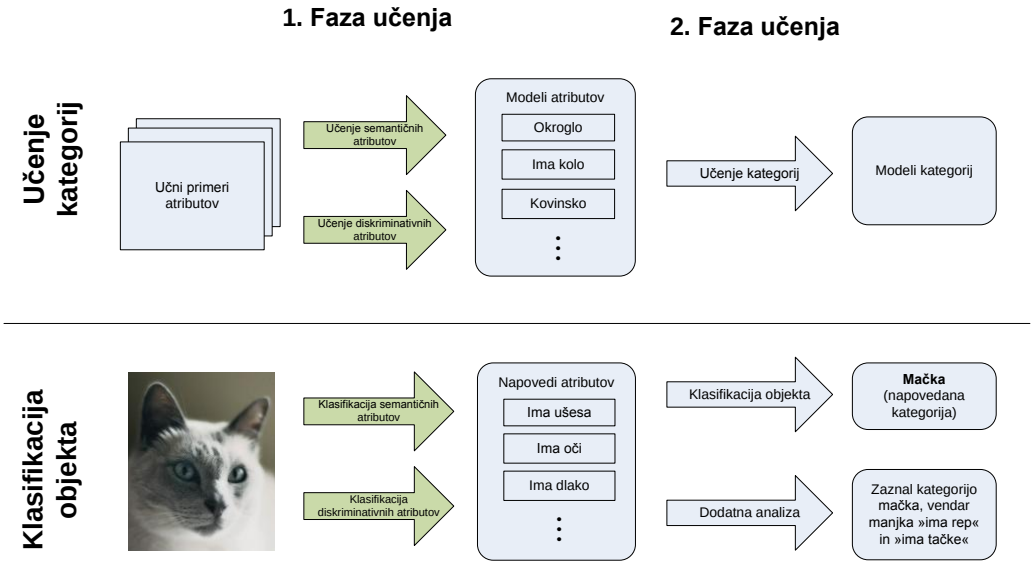
2.2 Predstavitev implementacije

Množico učnih slik označimo $\mathcal{U}$, množico testnih slik pa s $\mathcal{T}$, pri čemer je posamezna slika $\mathcal{S} \in \mathcal{U} \cup \mathcal{T}$. Vse atribute, tako semantične kot diskriminativne, označimo z množico $\mathcal{A} = \mathcal{A}_s \cup \mathcal{A}_d$, kjer je posamezen atribut $a \in \mathcal{A}$ oz. specifično semantičen atribut $a_s \in \mathcal{A}_s$ in diskriminativen atribut $a_d \in \mathcal{A}_d$. Kategorije označimo kot $\mathcal{K}$, ki vsebuje posamezne razrede $r \in \mathcal{K}$. Postopek učenja kategorij lahko predstavimo z dvema funkcijama, $f_{attribut}$ in $f_{kategorije}$, ki sta definirani v (2.1) in (2.2).

$$f_{atribut} : \mathcal{U} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\} \longmapsto m_{atribut} \tag{2.1}$$

$$f_{kategorije} : \mathcal{U} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\} \longmapsto m_{kategorije} \tag{2.2}$$

Pri učenju obe funkciji sprejmeta učno množico slik, za katere poznamo ustrezen razred r in vse semantične atribute a_s (tj. *'ground truth'* podatki). Diskriminativnih atributov pri tem ni treba poznati, saj se izračunajo naknadno



Slika 2.3: Vpeljava diskriminativnih atributov v učenje in klasifikacijo objektov z atributi.

iz učne množice. Torej za vsako učno sliko mora obstajati preslikava $p : \mathcal{S} \in \mathcal{U} \mapsto (r \in \mathcal{K}, \{a_{s_1}, a_{s_2}, ...\} \subseteq \mathcal{A}_s)$. Obe funkciji na koncu vrneta bodisi naučene klasifikatorje oz. modele posameznega atributa $m_{atribut}$ bodisi modele naučenih kategorij $m_{kategorije}$. Ti modeli so rezultat algoritma strojnega učenja in se nato uporabijo pri klasifikaciji objekta, katere postopek je definiran s funkcijama $g_{atribut}$ in $g_{kategorije}$ v (2.3) in (2.4).

$$g_{atribut} : m_{atribut} \times \mathcal{S} \in \mathcal{T} \longmapsto a \in \mathcal{A} \tag{2.3}$$

$$g_{kategorije} : m_{kategorije} \times \{a_1, a_2, ...\} \longmapsto r \in \mathcal{K} \tag{2.4}$$

2.2.1 Učenje atributov in kategorij

Postopek učenja atributov iz enačbe (2.1) se prične z izračunom bazne značilke $\mathcal{B}$ za vsako učno sliko $\mathcal{S} \in \mathcal{U}$, kot je prikazano na sliki (2.4). Izračun bazne značilke je podrobneje predstavljen v naslednjem poglavju. Po tem, ko dobimo bazne značilke za vse učne slike, se postopek razdeli na učenje diskriminativnih atributov in učenje semantičnih atributov. Za slednje moramo predhodno za vsako sliko opraviti izbiro pomembnega dela bazne značilke $\mathcal{B}_{izbrana} \subseteq \mathcal{B}$.

Učenje diskriminativnih atributov ter izbira pomembnih značilk sta prav tako podrobneje predstavljena v naslednjih podpoglavjih.

Preden nadaljujemo z učenjem semantičnih atributov, definirajmo še algoritem strojnega učenja v (2.5) in (2.6). Tega lahko splošno označimo kot funkcijo $h_{učenje}$ in $h_{klasifikacija}$, pri čemer je p_i posamezen učni primer z ustreznimi učnimi atributi oz. učnim vektorjem $\vec{u}_i \in \mathbb{R}^n$ in kategorijo oz. razredom $r_i \in \mathcal{K}$. Pri učenju dobimo vrnjen naučen model m, ki ga nato uporabimo skupaj s testnim vektorjem $\vec{t}_i \in \mathbb{R}^n$ v funkciji klasifikacije. Dimenziji učnega in testnega vektorja se morata pri tem ujemati.

$$h_{učenje} : \{p_1, p_2, ...p_k\} \longmapsto m \tag{2.5}$$

$$h_{klasifikacija} : m \times \vec{t}_i \longmapsto r \in \mathcal{K} \tag{2.6}$$

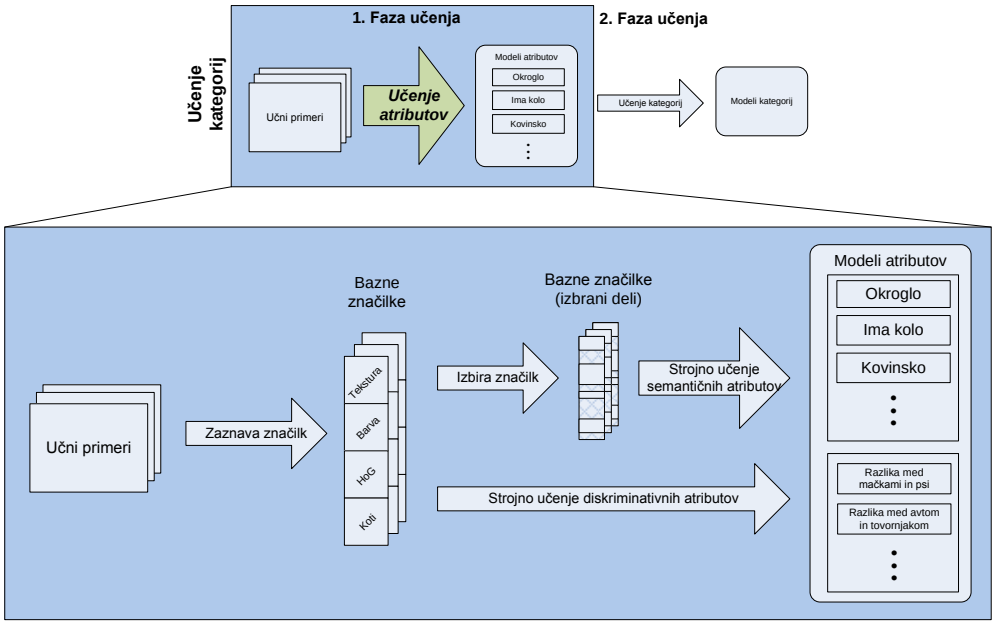
Učenje semantičnih atributov poteka za vsak atribut a_s ločeno, tako da preslikamo vse učne slike $\mathcal{S}_i \in \mathcal{U}$ v ustrezne učne primere $p_i = (\vec{u}_i, r_i)$ za strojno učenje. Pri tem določimo za učne atribute izbrane bazne značilke za specifičen atribut $\vec{u}_i \leftarrow \mathcal{B}_{izbrane_i}$ ter določimo tudi razred r_i, glede na to, ali učna slika vsebuje izbrani semantčni atribut a_s:

$$r_i \leftarrow \begin{cases} 1; & (r, \mathcal{A}_i) = p\left(\mathcal{S}_i\right) \wedge a_s \in \mathcal{A}_i \\ 0; & sicer \end{cases} \tag{2.7}$$

Vse učne primere za izbrani semantični atribut $\{p_1, p_2, ..., p_n\}_{a_s}$ sedaj uporabimo kot parameter za strojno učenje $h_{učenje}$, ki nam nato vrne uszrezen model m_{a_s}. V našem primeru smo za učenje atributov preizkusili različne algoritme strojnega učenja: SVM iz OpenCV ter SVM^{perf} in SVC iz libLinear. Za vsak semantičen atribut si sedaj shranimo ustrezne atribute ter povrnjene modele v množico $\mathcal{M}_{semantični} = \left\{\left(a_s, m_{a_s}\right)_j\right\}$. Skupaj z množico diskriminativnih atributov oz. modelov sedaj ti množici predstavljata naučene modele atributov, potrebnih v nadaljnjem postopku učenja in klasifikacije.

$$\mathcal{M}_{atributi} = \mathcal{M}_{semantični} \cup \mathcal{M}_{diskriminativni} \tag{2.8}$$

Potem ko naučimo vse atribute, sledi učenje kategorij, ki je sedaj relativno preprosto. Za vsako učno sliko $\mathcal{S}_i \in \mathcal{U}$ pripravimo za algoritem strojnega učenja ustrezne učne primere p_i, tako da za razred r_i vzamemo ustrezni razred iz *'ground truth'* podatkov $(r_i, \mathcal{A}_{s_i}) = p\left(\mathcal{S}_i\right)$. Učni vektor $\vec{u}_i$ pa sestavimo iz semantičnih in diskriminativnih atributov. Vsak atribut $a_j \in \mathcal{A}$ predstavlja eno dimenzijo v učnem vektorju $\vec{u}_i \in \mathbb{R}^{|\mathcal{A}|}$ tako, da če učna slika $\mathcal{S}_i$ vsebuje



Slika 2.4: Podroben prikaz postopka učenja atributov.

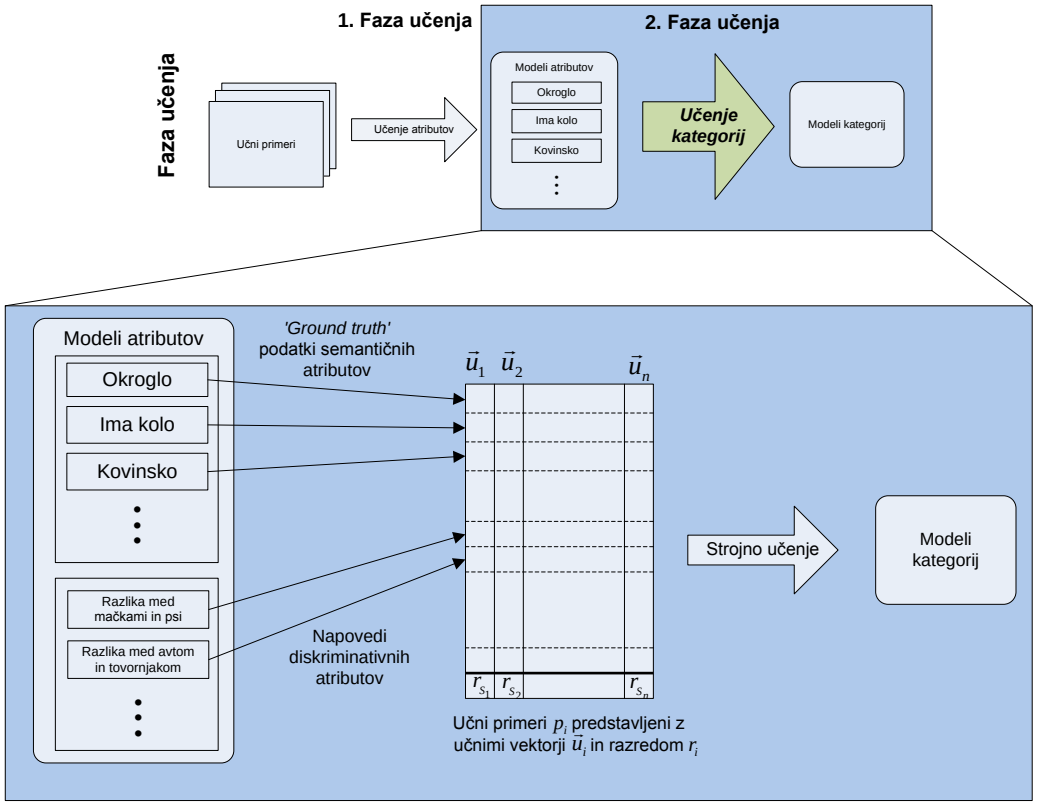
ta atribut a_j, potem je na tem mestu vrednost vektorja $\vec{u}_i[j] = 1$, če ga ne vsebuje, pa je vrednost $\vec{u}_i[j] = 0$. Za semantičen atribut a_{s_j} to pomeni:

$$\vec{u}_i[j] = \begin{cases} 1; & (r_i, \mathcal{A}_i) = p(\mathcal{S}_i) \wedge a_{s_j} \in \mathcal{A}_i \\ 0; & sicer \end{cases} \tag{2.9}$$

Za vsak diskriminativni atribut $a_{d_j} = \left(c_j, m_{disk.atr._j}\right)$ pa je treba izračunati napoved s $h_{klasifikacija}$. Pri tem je treba izpostaviti, da uporabimo celotno bazno značilko $\mathcal{B}_i$:

$$\vec{u}_i[j] = \begin{cases} 1; & h_{klasifikacija}\left(m_{disk.atr._j}, \mathcal{B}_i\right) = r_{j,i} \wedge r_{j,i} = 1 \\ 0; & sicer \end{cases} \tag{2.10}$$

Potem ko dobimo vse usztrezne učne primere $\vec{p}_i$ za celotno učno množico $\mathcal{U}$, lahko uporabimo algoritem strojnega učenja in dobimo ustrezen model kategorije $m_{kategorije}$, ki ga shranimo za klasifikacijo. Za učenje smo preizkusili algoritme SVM iz OpenCV in SVM^{multiclass}, ter Logistično Regresijo iz libLinear s privzetimi nastavitvami.

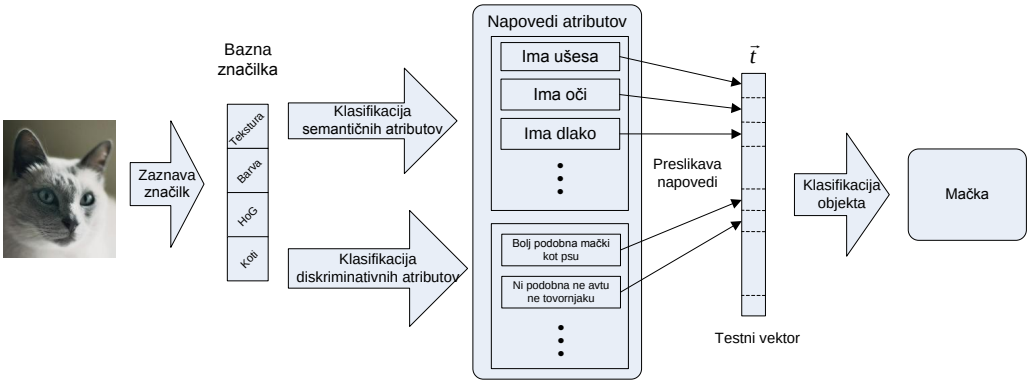


Slika 2.5: Podroben prikaz postopka učenja kategorij.

2.2.2 Klasifikacija objekta

Klasifikacija objekta iz testne slike $\mathcal{S} \in \mathcal{T}$ se prične z ustrezno napovedjo oz. klasifikacijo vsakega semantičnega in diskriminativnega atributa $a_j \in \mathcal{A}$. Preden lahko napovemo atribute, je potrebno za sliko $\mathcal{S}$ izračunati bazno značilko $\mathcal{B}$. Nato lahko za vsak atribut a_j uporabimo ustrezen model atributa m_{a_j}, ki smo ga izračunalni pri učenju atributov, ter bazno značilko za testni vektor $\vec{t}_{atributi} \leftarrow \mathcal{B}$ in preko funkcije $h_{klasifikacija}$ izračunamo ustrezno napoved, ali sliki $\mathcal{S}$ pripada atribut a_j. Na podlagi teh napovedi lahko sedaj sestavimo nov testni vektor $\vec{t}_k \in \mathbb{R}^{|\mathcal{A}|}$ za kategorije, kjer vsak atribut a_j predstavlja eno dimenzijo vektorja:

$$\vec{t}_k\,[j] = \begin{cases} 1; & h_{klasifikacija}\left(m_{a_j}, \mathcal{B}\right) = r_j \wedge r_j = 1 \\ 0; & sicer \end{cases} \tag{2.11}$$



Slika 2.6: Primer podrobnega postopka klasifikacije objekta.

Izračunani vektor $\vec{t}_k$ sedaj uporabimo pri klasifikaciji končne kategorije objekta $r \in \mathcal{K}$ iz slike $\mathcal{S}$:

$$h_{klasifikacija}\left(m_{kategorije}, \vec{t}_k\right) = r_{\mathcal{S}} \tag{2.12}$$

2.2.3 Izračun baznih značilk

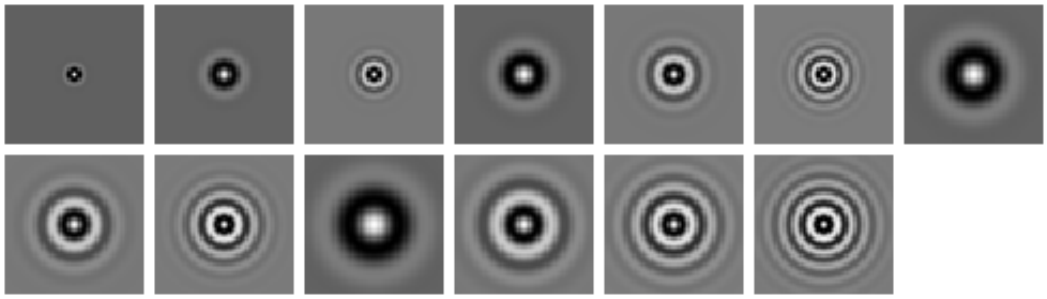
Za vsako sliko $\mathcal{S} \in \mathcal{U} \cup \mathcal{T}$ pred učenjem in klasifikacijo izvlečemo bazno značilko $\mathcal{B} = \left[\ \mathcal{C},\ \ \mathcal{Q},\ \ \mathcal{D},\ \ \mathcal{H}\ \right]$, ki je sestavljena iz štirih ločenih deskriptorjev: barva $\mathcal{C}$, tekstura Q, robovi $\mathcal{D}$, HOG $\mathcal{H}$. Prvi dve značilki $\mathcal{C}$ in $\mathcal{Q}$ poskrbita za splošen videz, medtem ko drugi dve predstavljata obliko. S tem se zajame celoten spekter različnih zvrst značilk.

2.2.3.1 Deskriptor teksture

Izračun deskriptorja teksture oz. tekstona poteka po zelo podobnem postopku kot v [26]. Za banko filtrov (ang. *filterbank*) so v omenjenem članku primerjali štiri različne množice:

- množica 48 filtrov Leung-Malik (LM) [53]
- množica 13 filtrov Schmid (S) [52]
- dve množici 38 filtrov maksimalnega odziva ($MR8$ in $MR4$)

Množica *MR8* se pri tem izkaže za najboljšo, takoj za njo pa je bila druga najboljša množica filtrov *S*. Glede na to, da je v *MR8* 38 različnih filtrov, je izračun odziva vseh filtrov lahko relativno počasen, zato smo za našo banko filtrov izbrali množico S s 13 različnimi filtri.



Slika 2.7: Množica 13 različnih filtrov S, uporabljenih za izračun tekstona oz. deskriptor teksture $\mathcal{T}$.

Izbrano množico filtrov označimo s $\mathcal{F}_B = (\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_{13})$. Za sliko $\mathcal{S}$ s konvolucijo izračunamo odziv na vsak filter $\mathcal{F}_1, \mathcal{F}_2$,..., $\mathcal{F}_{13} \in \mathcal{F}_\mathcal{B}$, ter odziv shranimo v ustrezen $\mathcal{V}_i$.

$$\begin{aligned} \mathcal{S} * \mathcal{F}_1 &= \mathcal{V}_1 \\ \mathcal{S} * \mathcal{F}_2 &= \mathcal{V}_2 \\ &\vdots \\ \mathcal{S} * \mathcal{F}_{13} &= \mathcal{V}_{13} \end{aligned} \tag{2.13}$$

Za vsak slikovni element slike $x_{i,j} = \mathcal{S}[i,j]$ lahko sedaj iz dobljenih odzivov sestavimo ustrezen 13-dimenzionalni vektor $\vec{v}_{i,j} \in \mathbb{R}^{13}$, tako da za vsako dimenzijo uporabimo ustrezen odziv $\mathcal{V}$.

$$\vec{v}_{i,j} = \left[\begin{array}{cccc} \mathcal{V}_1[i,j] & \mathcal{V}_2[i,j] & ... & \mathcal{V}_{13}[i,j] \end{array} \right] \tag{2.14}$$

Vektor $\vec{v}$ se sedaj kvantizira na najbližjega izmed 256 različnih centrov $\vec{v} \mapsto \vec{v}_c \in \{c_1, c_2, ..., c_{256}\}_{TEX}$, ki se jih predhodno izračuna iz čim večje množice učnih slik. Najbližji center smo izračunali s pomočjo algoritma ANN oz. implementacije FLANN [50] iz knjižice OpenCV. Za celotno množico kvantiziranih vektorjev $\{\vec{v}_{c_1}, \vec{v}_{c_2}, ..., \vec{v}_{c_n}\}$ iz določene slike velikosti n slikovnih elementov sedaj naredimo histogram, ki na koncu predstavlja 256-dimenzionalni deskriptor teksture $\mathcal{Q} \in \mathbb{N}^{256}$.

$$\mathcal{S} \longmapsto \{\vec{v}_{c_1}, \vec{v}_{c_2}, ..., \vec{v}_{c_n}\} \longmapsto \mathcal{Q} \in \mathbb{N}^{256} \tag{2.15}$$

Izračun 256 različnih centrov poteka s pomočjo algoritma K-Means [51]. Iz učne množice $\mathcal{U}$ za vsako kategorijo psevdonaključno vzamemo določeno število slik $\mathcal{U} \mapsto \mathcal{U}_{TEX} = \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m\}$. Pri tem so slike izbrane tako, da za je število slikovnih elementov na kategorijo približno enakomerno pri vseh kategorijah. Za vsako sliko $\mathcal{S}_i \in \mathcal{U}_{TEX}$ po zgornjem postopku izračunamo množico nekvantiziranih vektorjev $\mathcal{V}_i = \{\vec{v}_1, \vec{v}_2, ..., \vec{v}_n\}$, ter vse vektorje $\vec{v}_j$ vseh slik $\mathcal{S}_i \in \mathcal{U}_{TEX}$ shranimo v skupno množico $\mathcal{P} = \mathcal{V}_1 \cup \mathcal{V}_2 \cup ... \cup \mathcal{V}_m$. S pomočjo algoritma K-Means nato iz množice $\mathcal{P}$ izračunamo 256 grup in centrov, ki bodo najbolje predstavljali različne vektorje $\mathcal{P} \xrightarrow[gručenje]{K\text{-}Means} \{c_1, c_2, ..., c_{256}\}_{TEX}$.

2.2.3.2 Deskriptor barve

Deskriptor barve izračunamo iz trikanalne barvne slike $\mathcal{S}$, ki pa jo predhodno spremenimo v format LAB $\mathcal{S} \mapsto \mathcal{S}_{LAB}$. Vsak slikovni element $x_{LAB} \in S_{LAB}$ se predstavi kot tri-dimenzionalni vektor $\vec{y} \in \mathbb{N}^3$, ki se ga kvantizira na najbližjega izmed 128 različnih centrov $\vec{y} \mapsto \vec{y}_c \in \{c_1, c_2, ..., c_{128}\}_{COL}$. Iskanje najbližjega centra poteka z algoritmom FLANN. Iz tega za vsako sliko $\mathcal{S}$ velikosti n slikovnih elementov dobimo seznam kvantiziranih vektorjev $\{\vec{y}_{c_1}, \vec{y}_{c_2}, ..., \vec{y}_{c_n}\}$, iz katerega izračunamo histogram. Ta histogram na koncu predstavlja 128-dimenzionalni deskriptor barve $\mathcal{C} \in \mathbb{N}^{128}$.

$$\mathcal{S} \longmapsto \{\vec{y}_{c_1}, \vec{y}_{c_2}, ..., \vec{y}_{c_n}\} \longmapsto \mathcal{C} \in \mathbb{N}^{128} \tag{2.16}$$

Za kvantizacijo potrebujemo ustrezne centre, ki jih izračunamo po podobnem postopku kot centre teksture. Za vsako sliko $\mathcal{S}_i \in \mathcal{U}$ iz učne množice velikosti m slik po zgornjem postopku izračunamo množico vektorjev $\mathcal{Y}_i = \{\vec{y}_1, \vec{y}_2, ..., \vec{y}_n\}$, ter vse vektorje shranimo v skupno množico $\mathcal{Y}_{vsi} = \mathcal{Y}_1 \cup \mathcal{Y}_2 \cup ... \cup \mathcal{Y}_m$. Iz množice $\mathcal{Y}_{vsi}$ poiščemo le različne vektorje $\vec{y} \in \mathbb{N}^3$, za katere tudi izračunamo število ponovitev k in sestavimo par $(\vec{y} \in \mathbb{N}^3, k \in \mathbb{N})$. Iz vsakega para naredimo nov štiridimenzionalni vektor $\vec{z} \in \mathbb{N}^4$, tako da $\vec{y}$ dodamo k kot vrednost četrte dimenzije. $(\vec{y} \in \mathbb{N}^3, k \in \mathbb{N}) \mapsto \vec{z} \in \mathbb{N}^4$. Vse vektorje $\vec{z}$ shranimo v množico $\mathcal{P}$. S tem sicer povečamo dimenzijo vektorjev, vendar pa je velikost množice $\mathcal{P}$ mnogo manjša od velikosti množice $\mathcal{Y}_{vsi}$. To pomeni, da lahko uporabimo celotno učno množico $\mathcal{U}$. Z algoritmom K-Means iz množice vektorjev $\vec{z}$ sedaj izračunamo ustreznih 128 gruč ter njihove centre $c'_1, c'_2, \ldots, c'_{128}$. Preden lahko centre uporabimo, jim zmanjšamo dimenzijo, tako da enostavno odstranimo zadnjo dimenzijo.

$$\mathcal{P} \xrightarrow[gručenje]{K\text{-}Means} \{c'_1, c'_2, ..., c'_{128}\} \xrightarrow[dimezijo]{odstranimo} \{c_1, c_2, ..., c_{128}\}_{COL} \tag{2.17}$$

2.2.3.3 Deskriptor robov

Za deskriptor robov se uporablja standardni canny detektor [27], tako da se za vsako sliko izračuna število robov, ki ga ta detektor vrne ter ustrezno orientacijo v tem robu. Vsako sliko $\mathcal{S}$ se sprva filtrira s Sobelovim filtrom velikosti 3×3 v smeri x in y.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * \mathcal{S},\ G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathcal{S} \tag{2.18}$$

Iz rezultatov se nato izračuna magnitudo $G = \sqrt{G_x^2 + G_y^2}$ in orientacijo $\Theta = \arctan\left(\frac{G_y}{G_x}\right)$. Nadalje se v sliki naredi *Non-Maximum Suppression* ter uporagovanje s spodnjim pragom 50 in zgornjim pragom 80. Po tej operaciji dobimo seznam robov $\mathcal{R}$, kjer imamo podane njihove magnitude ter orientacije $\mathcal{R} = \{(m \in \mathbb{R}, \theta \in [0, 2\pi])_i\}$. Preden se nadaljuje se za vsak rob v $\mathcal{R}$ kvantizira orinetacijo na 8 smeri $\theta \in [0, 2\pi] \mapsto \theta_c \in \left\{\frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}, 2\pi\right\}$. Nato lahko iz vseh robov v $\mathcal{R}$ sedaj sestavimo histogram orientacij, kar pomeni, da izračunamo pogostost pojavitve določene orientacije. Iz tega dobimo 8-dimenzionalni vektor $\vec{d} \in \mathbb{N}^8$, ki mu dodamo še eno dimenzijo, ki predstavlja število robov v sliki oz. moč množice robov $|\mathcal{R}|$. Novi 9-dimenzionalni vektor tako predstavlja deskriptor robov $\mathcal{D} \in \mathbb{N}^9$.

2.2.3.4 Deskriptor HOG

Zadnji deskriptor, ki ga uporabljamo za vizualne besede, je implementacija HOG-a (ang. *Histogram of Orineted Gradients*) iz [28]. Za vsako trikanalno barvno sliko $\mathcal{S}_{RGB}$ sprva izračunamo gradient G_x v horizontalni in G_y v vertikalni smeri z jedrom $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$ in $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}^T$. Ta postopek naredimo ločeno za vsak kanal, tako da na koncu za vsak slikovni element slike $x_{RGB} = (x_R, x_G, x_B)$ dobimo ustrezen gradient $g_{RGB} = (g_R, g_G, g_B)$, pri čemer je

$$\begin{aligned} g_R &= (m_R \in \mathbb{R}, \theta_R \in [0, 2\pi)) \\ g_G &= (m_G \in \mathbb{R}, \theta_G \in [0, 2\pi)) \\ g_B &= (m_B \in \mathbb{R}, \theta_B \in [0, 2\pi)) \end{aligned} \tag{2.19}$$

Za vsak g_{RGB} nato izračunamo končni gradient , tako da vzamemo tisti gradient kanala, ki ima največjo moč $m = \max(m_R, m_G, m_B)$ in njegovo ustrezno orientacijo θ. Slednjo je treba pred nadaljevanjem kvantizirati na 9 različnih smeri, tako da dobimo na koncu gradient $g_c = \left(m \in \mathbb{R}, \theta_c \in \left\{\frac{2\pi}{9}, \frac{4\pi}{9}, ..., 2\pi\right\}\right)$.

Celotno sliko $\mathcal{S}_{RGB}$ velikosti $n \times m$ slikovnih elementov sedaj razdelimo v majhne celice $C_{i,j}$ velikosti 8×8 slikovnih elementov. Ta razdelitev nam vrne $\left\lceil \frac{n}{8} \right\rceil \times \left\lceil \frac{m}{8} \right\rceil$ novih celic, pri čemer vsaki pripada ustreznih 64 kvantiziranih gradientov $\{g_{c_1}, g_{c_2}, ..., g_{c_{64}}\}$. Iz teh gradientov za vsako celico $C_{i,j}$ izračunamo histogram orientacij $\vec{c}_{i,j}$, kjer vsak gradient prispeva s svojo močjo m.

$$C_{i,j} = \{g_{c_1}, g_{c_2}, ..., g_{c_{64}}\} \xrightarrow[histograma]{izračun} \vec{c}_{i,j} \in \mathbb{R}^9 \tag{2.20}$$

Iz postopka (2.20) dobimo za vsako sliko $\mathcal{S}_{RGB}$ množico histogramov $\{\vec{c}_{1,1}, \vec{c}_{1,2}, ..., \vec{c}_{k,l}\}$, kjer pa je treba vsak hisotgram še normalizirati. Normalizacijo naredimo tako, da pogledamo vse možne skupine celic velikosti 2×2 v okolici, ki bi vsebovale izbrano celico. Za vsako celico $C_{i,j}$ naredimo 4 možne skupine iz vseh 8 sosednih celic v taki kombinaciji, da vsaka skupina vedno vsebuje celico $C_{i,j}$ ter še tri skupne sosede. V skupino za vsako celico podamo ustrezen histogram $\vec{c}$ ter normaliziramo $\vec{c}_{i,j}$ vsake skupine glede na energijo vseh celic v skupini.

$$\begin{aligned}
\{\vec{c}_{i,j}, \vec{c}_{i-1,j-1}, \vec{c}_{i-1,j}, \vec{c}_{i,j-1}\} &\xrightarrow{normalizacija} \left\|\vec{c}^{\,1}_{i,j}\right\|_1 \\
\{\vec{c}_{i,j}, \vec{c}_{i-1,j}, \vec{c}_{i-1,j+1}, \vec{c}_{i,j+1}\} &\xrightarrow{normalizacija} \left\|\vec{c}^{\,2}_{i,j}\right\|_1 \\
\{\vec{c}_{i,j}, \vec{c}_{i,j-1}, \vec{c}_{i+1,j-1}, \vec{c}_{i,j+1}\} &\xrightarrow{normalizacija} \left\|\vec{c}^{\,3}_{i,j}\right\|_1 \\
\{\vec{c}_{i,j}, \vec{c}_{i,j+1}, \vec{c}_{i+1,j}, \vec{c}_{i+1,j+1}\} &\xrightarrow{normalizacija} \left\|\vec{c}^{\,4}_{i,j}\right\|_1
\end{aligned} \tag{2.21}$$

Vse štiri noralizirane histograme, ki jih dobimo po zgornjem postopku, združimo sedaj v nov v normaliziran vektor celice $\|\vec{c}_{i,j}\| = (\left\|\vec{c}^{\,1}_{i,j}\right\|, \left\|\vec{c}^{\,2}_{i,j}\right\|, \left\|\vec{c}^{\,3}_{i,j}\right\|, \left\|\vec{c}^{\,4}_{i,j}\right\|)$ dolžine $36 = 4 \times 9$.

Vse normalizirane vektorje $\{\|\vec{c}_{1,1}\|, \|\vec{c}_{1,2}\|, ..., \|\vec{c}_{k,l}\|\}$, ki pripadajo sliki $\mathcal{S}_{RGB}$, zdaj še dodatno grupiramo v 4×4 velike skupine sosednjih celic, pri čemer za vsako skupino dobimo nov vektor $\vec{d}$ dolžine $576 = 36 \times 4 \times 4$, ki je sestavljen iz vseh normaliziranih vektorjev celic. Skupine se oblikuje tako, da okno za vsako naslednjo skupino premaknemo le za eno celico, in ne za štiri celice, kar pomeni, da se sosednje skupine prekrivajo v najmanj 3×3 celicah (lahko tudi v 3×4 celicah). Za sliko $\mathcal{S}_{RGB}$ tako dobimo množico 576-dimenzionalnih vektorjev $\left\{\vec{d}_1, \vec{d}_2, ..., \vec{d}_o\right\}$. Celoten zgornji postopek ponavljamo na vse manjših slikah $\mathcal{S}'_{RGB}, \mathcal{S}''_{RGB}$..., ki jih skaliramo s faktorjem $\sigma = \sqrt{2}$. Iz vseh zgornjih slik tako dobimo skupno množico 576-dimenzionalnih vektorjev $\mathcal{D} = \left\{\vec{d}_1, \vec{d}_2, ..., \vec{d}_o\right\} \cup \left\{\vec{d'}_1, \vec{d'}_2, ..., \vec{d'}_p\right\} \cup \left\{\vec{d''}_1, \vec{d''}_2, ..., \vec{d''}_r\right\} \cup ...$

Množico vseh vektorjev iz celotne piramide slik $\mathcal{D}$ je na koncu treba še kvantizirati na enega izmed 1000 centrov $\vec{d}_i \mapsto \vec{d}_{c_i} \in \{c_1, c_2, ..., c_{1000}\}_{HOG}$,

kar naredimo z algoritmom FLANN iz knjižice OpenCV. Na koncu za celotno množico kvantiziranih vektorjev izračunamo še končni histogram $\mathcal{H} \in \mathbb{N}^{1000}$.

$$D \longmapsto \left\{ \vec{d}_{c_1}, \vec{d}_{c_2}, ..., \vec{d}_{c_q} \right\} \longmapsto \mathcal{H} \in \mathbb{N}^{1000} \tag{2.22}$$

Izračun centrov, potrebnih za kvantizacijo, poteka po podobnem postopku kot izračun centrov za deskriptor teksture. Iz učne množice $\mathcal{U}$ za vsako kategorijo psevdo-naključno vzamemo določeno število slik $\mathcal{U} \mapsto \mathcal{U}_{HOG} = \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m\}$. Pri tem so slike izbrane tako, da je število slikovnih elementov na kategorijo približno enakomerno pri vseh kategorijah. Za vsako sliko $\mathcal{S}_i \in \mathcal{U}_{HOG}$ po zgornjem postopku izračunamo množico nekvantiziranih vektorjev $D_i = \left\{ \vec{d}_1, \vec{d}_2, ..., \vec{d}_n \right\}$ ter vse vektorje $\vec{d}_j$ vseh slik $\mathcal{S}_i \in \mathcal{U}_{TEX}$ shranimo v skupno množico $\mathcal{P} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_m$. S pomočjo algoritma K-Means nato iz množice $\mathcal{P}$ izračunamo 1000 grup in centrov, ki bodo najbolje predstavljali različne vektorje $\mathcal{P} \xrightarrow[gručenje]{K\text{-}Means} \{c_1, c_2, ..., c_{1000}\}_{HOG}$.

2.2.3.5 Normalizacija vektorja

Preden lahko uporabimo bazno značilko za učenje semantičnih atributov, je treba poskrbeti, da bo značilko možno uporabiti za slike različnih velikosti. Za to je treba narediti normalizacijo vsakega histograma oz. značilke posebej. Histogram teksture, barve in kotov se normalizira z normo ℓ^2 tako, da se vsako vrednost histograma deli s korenom vsote kvadratov. Za histogram HOG-a se uporablja normalizacijo ℓ^2-Hys, tako se podobno kot pri normi ℓ^2 sprva vsako vrednost deli s korenom vsote kvadratov, nato pa se odreže pri določeni vrednosti ter ponovno normalizira na skalo med 0 in 1. Z normalizacijo vseh histogramov se tako znebimo vplivov velikosti slike na bazno značilko.

$$\mathcal{B} = \left[\ \frac{\mathcal{C}}{\|\mathcal{C}\|_2}, \quad \frac{\mathcal{Q}}{\|\mathcal{Q}\|_2}, \quad \frac{\mathcal{D}}{\|\mathcal{D}\|_2}, \quad \frac{\mathcal{H}}{\|\mathcal{H}\|_{2\text{-}Hys}} \ \right] \tag{2.23}$$

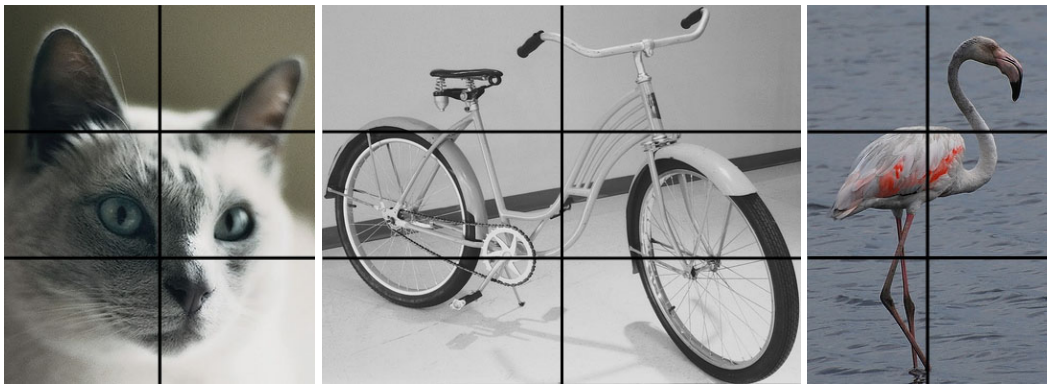
2.2.3.6 Lokalizacija

Ena izmed pomembnih lastnosti, uporabljenih v [23] za izračun bazne značilke, je tudi uporaba dodatnega vzorčenja za boljšo lokalizacijo. Lokalizacija poteka tako, da se celotna slika $\mathcal{S}$ razdeli na tri vertikale in dve horizontali v 6 celic $L_{i,j} \subset \mathcal{S}$, tako da vsaka celica najbolje zajame lokalne lastnosti. Vse tipe deskriptorjev je zato treba izračunati tako za celotno sliko $\mathcal{S}$ kot tudi za vsako lokalno celico $L_{i,j}$. Skupaj vsi različni deskriptorji oz. histogrami vseh celic

in celotne slike sedaj predstavljajo končno 9751-dimenzionalno bazno značilko $\mathcal{B} \in \mathbb{R}^{9751}$.

$$\mathcal{B} = \left[\; \mathcal{B}_{\mathcal{S}}, \;\; \mathcal{B}_{L_{1,1}}, \;\; \mathcal{B}_{L_{1,2}}, \;\; \mathcal{B}_{L_{2,1}}, \;\; \mathcal{B}_{L_{2,2}}, \;\; \mathcal{B}_{L_{3,1}}, \;\; \mathcal{B}_{L_{3,2}} \; \right] \tag{2.24}$$

Lokalizacija je uporabna predvsem pri kategorizaciji objektov, pri katerih se nekateri atributi vedno nahajajo v določeni celici. Primer tega bi bila kategorija obraz, kjer se oči skoraj vedno nahajajo na vrhu slike, nos na sredini, usta spodaj in ušesa ob straneh. Poleg tega so v [23] pokazali, da je pri nekaterih primerih možno ugotoviti lokacijo semantičnih atributov ravno s pomočjo omenjene lokalizacije.



Slika 2.8: Primer slik, razdeljenih na tri vertikale in dve horizontali. Za vsako sliko skupaj dobimo 6 celic, za katere ločeno izračunamo bazno značilko.

2.2.4 Izbiranje pomembnih značilk

Pri postopku izbire pomembnih značilk (ang. *feature selection*) poskušamo iz bazne značilke $\mathcal{B}$ izločiti določene dele oz. vrednosti, ki bi znale negativno vplivati na uspešnost učenja semantičnega atributa $a_s \in \mathcal{A}$. Vsak semantičen atribut se dejansko učimo popolnoma neodvisno od ostalih, zato so lahko za različne atribute pomembni različni deli značilk, in je zaradi tega treba postopek izbire značilk izvesti za vsak atribut ločeno. Rezultat tega postopka je za vsak semantičen atribut a_{s_j} binarni vektor maske $\vec{b}_{m_j} \in \{0,1\}^{9751}$, na podlagi katerega se izbriše nepomembne vrednosti iz vsake bazne značilke $\mathcal{B}_i$ slike $\mathcal{S}_i$. Izbrano bazno značilko nato pred postopkom učenja atributa a_{s_j} izračunamo po enačbi (2.25), kjer pika predstavlja množenje po dimenzijah ter $\mathcal{B}_{izbrana_i}, \mathcal{B}_i \in \mathbb{R}^{9751}$.

$$\mathcal{B}_{izbrana_i} = \mathcal{B}_i \cdot \vec{b}_{m_j} \tag{2.25}$$

Postopek izračuna binarnega vektorja $\vec{b}_m$ za semantični atribut a_s pričnemo z iskanjem vseh kategorij oz. razredov $r_i \in \mathcal{K}$, pri katerih se atribut a_s pojavi v učni množici $\mathcal{U}$. Vse te razrede shranimo v množico $\mathcal{R} = \{r_i \mid (r_i, \mathcal{A}_j) = p\left(\mathcal{S}_j \in \mathcal{U}\right) \wedge a_s \in \mathcal{A}_j\}$. Iz množice teh razredov izberemo prvega $r_0 \in \mathcal{R}$ in na podlagi tega razreda poiščemo vse učne primere $\mathcal{S}_j$, ki pripadajo razredu r_0, ter jih razdelimo v dve novi učni množici $\mathcal{U}_a$ in $\mathcal{U}_b$, glede na to, ali slika $\mathcal{S}_j$ vsebuje semantičen atribut a_s. Definicija teh učnih množic je podana v spodnji enačbi (2.26).

$$\begin{aligned} U_a &= \{S_j \mid (r_j, \mathcal{A}_j) = p\left(\mathcal{S}_j \in \mathcal{U}\right),\ r_j = r_0,\ a_s \in \mathcal{A}_j\} \\ U_b &= \{S_j \mid (r_j, \mathcal{A}_j) = p\left(\mathcal{S}_j \in \mathcal{U}\right),\ r_j = r_0,\ a_s \notin \mathcal{A}_j\} \end{aligned} \tag{2.26}$$

To razdelitev se sedaj naučimo z algoritmom strojnega učenja $h_{učenje}$. Za vsako učno sliko $\mathcal{S}_j$ iz množice $\mathcal{U}_a$ in $\mathcal{U}_b$ ustvarimo ustrezen učni primer $p_j = (\vec{u}_j, r_j)$. Za učni vektor uporabimo celotno bazno značilko slike $\vec{u}_j \leftarrow \mathcal{B}_j$, medtem ko za učni razred r_j izberemo bodisi vrednost 0, če se S_j nahaja v pri učni množici bodisi vrednost 1, če se slika nahaja v drugi učni množici.

$$r_j = \begin{cases} 1; & \mathcal{S}_j \in \mathcal{U}_a \\ 0; & \mathcal{S}_j \in \mathcal{U}_b \end{cases} \tag{2.27}$$

Na podlagi množice učnih primerov p_j se sedaj z L1-regularizirano logistično regresijo iz libLinear ($C = 60$) naučimo to razdelitev, ter pri tem dobimo ustrezen model za trenutni izbrani razred m_{r_0}.

$$m_{r_0} = h_{učenje}\left(\{p_1, p_2, ...\}\right) \tag{2.28}$$

Za logistično regresijo vrnjeni model vsebuje pomemben vektor $\vec{w}_{r_0} \in \mathbb{N}^{9751}$, ki ga lahko uporabimo pri setavljanju binanrnega vektorja makse $\vec{b}_m$. To naredimo tako, da ustvarimo vektor maske in ga napolnimo z ničlami, če ga še nimamo, ter nato, če vektor $\vec{w}_{r_0}$ vsebuje neničelno vrednost, to preslikamo kot vrednost 1, sicer pa ohranimo ničlo. Postopek lahko strnemo v naslednjo enačbo:

$$\vec{b}_m\left[k\right] = \begin{cases} 1; & \vec{w}_{r_0}\left[k\right] \neq 0 \\ \vec{b}_m\left[k\right]; & sicer \end{cases} \tag{2.29}$$

Ko prekopiramo vse ustrezne vrednosti iz $\vec{w}$ za trenutni razred r_0, ponovimo celoten postopek še za vse ostale razrede r_i iz množice $\mathcal{R}$, kjer vsak ustrezno posodobi vektor $\vec{b}_m$.

Celoten postopek izbire značilk lahko demonstriramo na primeru semantičnega atributa a_s = »ima kolo«. Kolo lahko najdemo na objektih, kot so »avto«, »vlak«, »motorno kolo«, »voz«, »kočija« itd. Pri vseh, razen pri kategorijah »voz« in »kočija«, je mogoče opaziti, da obstaja velika verjetnost, da bodo objekti vsebovali tudi atribut »je kovinsko«. Zaradi tega obstaja velika verjetnost, da se bo učni algoritem namesto »ima kolo« naučil »je kovinsko«. Postopek izbire nadaljujemo tako, da izberemo prvo kategorijo, na primer r_0 = »avto« in poiščemo vse učne primere kategorije »avto«. To množico sedaj razdelimo na $\mathcal{U}_a$ = del objektov »avto« z atributom »ima kolo« in $\mathcal{U}_b$ = del objektov »avto« brez atributa »ima kolo« (2.26). S tem, ko se naučimo to razdelitev, smo izbrali le tiste dele značilk, ki se nanašajo na atribut »ima kolo«, saj je samo ta atribut le v enem delu učne množice, medtem ko so vsi ostali atributi (npr. »je kovinsko«) v obeh delih množice. Nato nadaljujemo z ostalimi kategorijami, kot na primer r_1 = »kočija«, in poiščemo značilke, ki dobro ločijo med kočijami s kolesom in kočijami brez kolesa. Vse pomembne značilke iz vseh petih kategorij združimo v skupni vektor maske $\vec{b}_m$ in tega nato uporabimo pred učenjem semantičnih atributov.

Učinkovitost tega pristopa je še vedno delno odvisna od učne množice, saj se bodo v primeru, da vsi avtomobili vsebujejo tudi kolo, pomembne značilke za to kategorijo še vedno lahko pomešale z drugimi atributi. V takih primerih lahko opazimo, da se »slabi« deli prenesejo v množico vseh izbranih značilk. Četudi bi imeli veliko ostalih kategorij, za katere bi dobili povsem »dobre« pomembne značilke, je dovolj le ena kategorija z vrnjenimi »slabimi« značilkami, da pokvari celoten postopek. Zaradi tega je pomembno, da se take primere kategorij že vnaprej izloči in se jih ne uporabi pri postopku izbire pomembnih značilk.

Postopek izbire pomembnih značilk se lahko zaradi velikega števila kategorij ponovi mnogokrat, zato je pomembno, da je učenje razdelitev za vsako kategorijo zelo hitro. Uporabljeni algoritem L1-regularizirane logistične regresije je zelo hiter in je bil tudi že uporabljen za podobno izbiranje značilk [29], zato je ta algoritem povsem primeren za dani problem.

2.2.5 Diskriminativni atributi

Diskriminativne atribute poiščemo po sledečem postopku. Za celotno učno množico $\mathcal{U}$ naključno izberemo neko razdelitev c_i množice na dva dela $c_i = (\mathcal{K}_a, \mathcal{K}_b \subseteq \mathcal{U})$. To naredimo tako, da izmed množice vseh kategorij $\mathcal{K}$ naključno

izberemo razrede $r \in \mathcal{K}$ in jih dodelimo bodisi množici $\mathcal{K}_a$ bodisi množici $\mathcal{K}_b$. Vsaki podmnožici dodelimo maksimalno 5 različnih razredov in pri tem poskrbimo, da podmnožici nimata skupnih razredov $\mathcal{K}_a \cap \mathcal{K}_b = \emptyset$. Iz te razdelitve ustrezno definiramo učni množici $\mathcal{U}_a$ in $\mathcal{U}_b$:

$$\begin{aligned} \mathcal{U}_a &= \{\mathcal{S}_i \mid (r, \vec{a}_s) = p(\mathcal{S}_i),\ r \in \mathcal{K}_a\} \\ \mathcal{U}_b &= \{\mathcal{S}_i \mid (r, \vec{a}_s) = p(\mathcal{S}_i),\ r \in \mathcal{K}_b\} \end{aligned} \tag{2.30}$$

To razdelitev se sedaj poskušamo naučiti s pomočjo željenega algoritma strojnega učenja. V našem primeru smo uporabili L1-regularizirano logistično regresijo iz libLinear s privzetimi atributi ($C = 1$), vzamemo pa lahko tudi SVM ali katerikoli drug algoritem strojnega učenja, ki se lahko zelo dobro nauči binarne razrede. Pri učenju ne uporabimo celotne bazne značilke, ampak le določeno vrsto značilke. Izmed barve $\mathcal{C}$, teksture $\mathcal{T}$, robov $\mathcal{D}$ in HOG $\mathcal{H}$ naključno izberemo le eno značilko $\mathcal{B}_{naključna}$. Za vsako učno sliko $\mathcal{S}_j \in \mathcal{U}_a \cup \mathcal{U}_b$ določimo učni primer $p_j = (\vec{u}_j, r_j)$, tako da za učni vektor vzamemo željeno značilko $\vec{u}_j \leftarrow \mathcal{B}_{naključna_j}$, razred r_j pa določimo glede na to v katero razdelitev spada učna slika S_j:

$$r_j = \begin{cases} 0; & \mathcal{S}_j \in \mathcal{U}_a \\ 1; & \mathcal{S}_j \in \mathcal{U}_b \end{cases} \tag{2.31}$$

Vse ustvarjene učne primere sedaj uporabimo pri strojnem učenju $h_{učenje}(\{p_1, p_2, ..., p_n\}_{c_i}) = m_{disk.\,atr._i}$. Ko se celotno razdelitev c_i ustrezno naučimo preverimo, kako dober klasifikator oz. model $m_{disk.\,atr._i}$ smo dobili, tako da ga pretestiramo na celotni učni množici $\mathcal{U}$. V primeru, da ocenimo klasifikator razdelitve z vrednostjo AUC vsaj 0,8, razdelitev obdržimo skupaj z ustreznim naučenim modelom ter jo dodamo v množico dobrih razdelitev $\mathcal{R} = \mathcal{R} \cup \{(c_i, m_{disk.\,atr._i})\}$, v nasprotnem primeru pa razdelitev enostavno zavržemo. Postopek nato ponovimo od začetka in ustvarimo novo radelitev c_j ter vse skupaj ponavljamo, dokler ne dobimo zadostnega števila različnih razdelitev. V [23] so uporabili 1000 različnih razdelitev, v našem primeru pa smo se omejili le na 200, saj je učenje oz. iskanje dobrih razdelitev zaradi naključnosti relativno počasno. Vsaka izmed razdelitev $c_i \in \mathcal{R}$ v končni množici tako predstavlja en diskriminativen atribut $(c_i, m_{disk.\,atr._i}) = a_{d_i}$, skupno pa imamo 200 različnih diskriminativnih atributov. Vse diskriminativne atribute si hranimo v množico $\mathcal{M}_{diskriminativni}$.

$$\mathcal{M}_{diskriminativni} = \{a_{d_1}, a_{d_2}, ..., a_{d_{200}}\} \tag{2.32}$$

Praktičen primer ene izmed dobrih razdelitev bi bila kategorija »mačka« na eni strani, kategorija »pes« na drugi strani, za bazno značilko pa je izbran deskriptor HOG $\mathcal{H}$. Shranjene razdelitve in njihove klasifikatorje iz množice $\mathcal{M}_{diskriminativni}$ nato uporabimo pri učenju kategorij, kot je razloženo v enem izmed zgornjih poglavij.

Vrednost AUC, uporabljeno za ocenjevanje uspešnosti razdelitve, smo aproksimirali iz ene točke ROC. Podrobnosti o tem, kako smo izračunali vrednost AUC so predstavljene v poglavju *Testi in rezultati.*

2.3 Naše dodatne izboljšave

Vse zgoraj omenjene lastnosti implementacije so povzete po konceptih iz [23], vendar se je skozi potek implementiranja in testiranja izkazalo, da bi bilo smiselno vpeljati nekatere dodatne izboljšave. Čeprav je povsem možno, da so nekatere od teh izboljšav bile že originalno uporabljene v zgoraj omenjenem članku, tega ni mogoče zanesljivo sklepati iz samega članka, zato so te izboljšave v pričujoči nalogi omenjene kot dodatne.

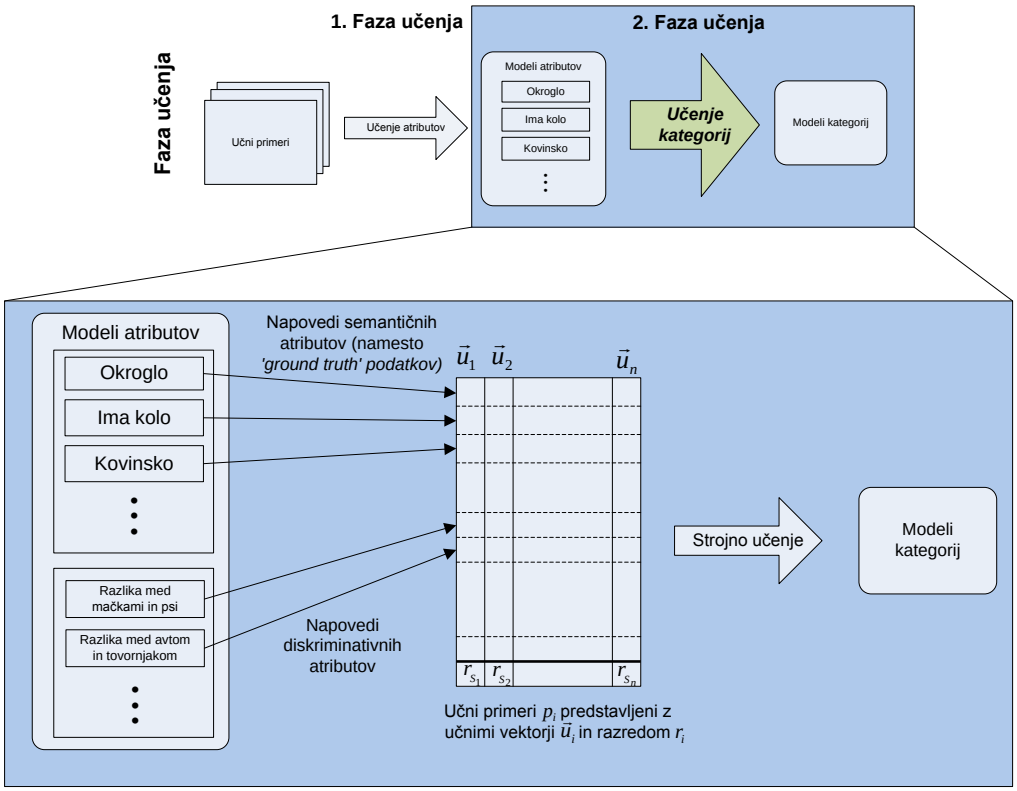
Prve tri izboljšave, učenje na nanapovedanih atributih, ločena lokalizacija ter popravilo pristranskosti, smo uporabili že pri prvem sklopu eksperimentov, kjer smo uspešnost naše implementacije primerjali z implementacijo iz članka, medtem ko smo ostali dve izboljšavi, krožno vzorčenje in nove diskriminativne atribute, uporabili šele v drugem sklopu testov, kjer smo preverjali uspešnost integracije modela LHOP v učenje z atributi.

2.3.1 Učenje kategorij na napovedanih semantičnih atributih

Pri učenju kategorij običajno semantične atribute iz slike $\mathcal{S}_i$ preslikamo v učne vektorje $\vec{u}_i$ glede na *'ground truth'* podatke. Vendar pa obstaja tudi možnost, da za učne atribute uporabimo kar napovedi iz naučenih klasifikatorjev za semantične atribute. Podobno, kot bi naredili za diskriminativne atribute, lahko uporabimo klasifikatorje semantičnih atributov, ki smo jih izračunalni v prvi fazi učenja. Za vsako učno sliko $\mathcal{S}_i \in \mathcal{U}$ sedaj zgradimo ustrezen učni vektor $\vec{u}_i$ iz vsakega modela semantičnega atributa $\left(a_{s_j}, m_{a_{s_j}}\right) \in \mathcal{M}_{semantični}$:

$$\vec{u}_i[j] = \begin{cases} 1; & h_{klasifikacija}(m_{a_{s_j}}, \mathcal{B}_{izbrane_i}) = r_{j,i},\ r_{j,i} = 1 \\ 0; & sicer \end{cases} \tag{2.33}$$

Spremenjen postopek učenja kategorij je možno videti na sliki (2.9). S tem sicer v učenje vpeljemo dodaten šum, ki se ustvari z napačnimi napovedmi semantičnih atributov $r_{j,i}$, vendar se potem iste klasifikatorje oz. modele $m_{a_{s_j}}$, ki so ustvarili ta šum, uporabi tudi pri klasifikaciji atributov novih objektov. Zaradi tega se ta napaka izniči, ter se celo izkaže, da se s tem pridobi nekaj procentov na klasifikacijski točnosti. Za manjšo nevšečnost se izkaže tudi dejstvo, da je pri učenju za celotno učno množico treba izračunati te napovedi za vsak semantičen atribut ločeno, kar je lahko pri velikih bazah z velikim številom atributov zelo počasno. Vendar se ta nevšečnost pojavi le pri učenju in se izkaže, da ne predstavlja prevelikega problema.



Slika 2.9: Podroben prikaz postopka učenja kategorij, kjer za učni vektor uporabimo dejanske napovedi semantičnih atributov namesto *'gorund truth'* podatkov.

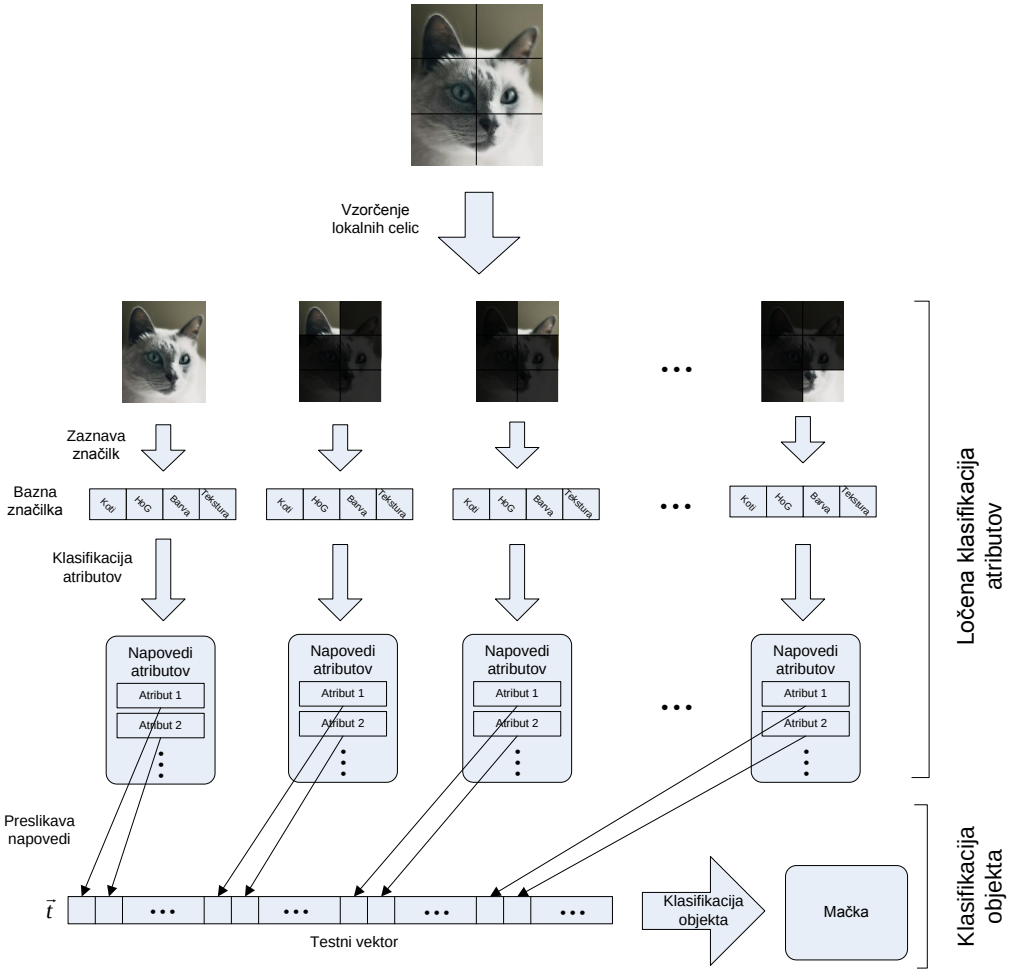
2.3.2 Učenje kategorij z ločeno lokalizacijo

Pri izračunu baznih značilk smo omenili, da sliko razdelimo na manjše celice $L_{i,j}$ za katere nato ločeno izračunamo lokalno bazno značilko $\mathcal{B}_{L_{i,j}}$, saj naj bi veliko kategorij imelo določene atribute na specifičnih delih slike (npr. avto ima kolesa skoraj vedno spodaj). Te lokalne bazne značilke shranimo skupaj z bazno značilko celotne slike $\mathcal{B}_{\mathcal{S}}$ v eno veliko bazno značilko, ko je prikazano v enačbi (2.24). Skozi testiranje pa se je izkazalo, da te značilke ni najbolje združiti v eno bazno značilko, saj lahko pride do napak tako pri učenju atributov kot tudi pri učenju objektov. Ilustrirajmo na primeru atributa »ima kolo«. Recimo, da se učimo ta semantični atribut le na primerih kategorij »motorno kolo« in »avto«. Tedaj se bo atribut »ima kolo« skoraj vedno pojavil v spodnjem delu slike, in klasifikator se bo naučil, da je v sliki kolo, kadar se kolo pojavi le na spodnjih lokalnih celicah. Iz tega je mogoče opaziti, da bomo, kadar bi ta atribut želeli uporabiti še za nove kategorije, naleteli na problem generalizacije. Zaradi tega se bo novo kategorijo veliko težje naučiti.

Omenjeni problem lahko rešimo s tem, da pri učenju semantičnih atributov uporabimo le bazno značilko iz cele slike $\mathcal{B}_{\mathcal{S}}$, medtem ko značilke iz lokalnih celic $\mathcal{B}_{L_{i,j}}$ za to učenje enostavno izpustimo. Ta rešitev bo pripomogla, da se algoritem strojnega učenja ne bo naučil le lokalnih značilk. Kljub rešitvi enega problema, pa s tem izgubimo prednosti lokalizacije, zato je treba lokalizacijo uporabiti v drugem koraku učenja, in sicer pri učenju kategorij $f_{kategorije}$ (2.2) ter nato ustrezno tudi pri klasifikaciji $h_{kategorije}$ (2.4). Lokalizacijo lahko vpeljemo v učenje kategorij tako, da se vsak semantičen atribut $a_{s_k} \in \mathcal{A}_s$ sedaj ne preslika v eno dimenzijo učnega vektorja $\vec{u}$, kot je to prikazano v enačbah (2.9) in (2.33), ampak se pogleda, ali se a_{s_k} nahaja na celotni sliki $\mathcal{S}$ ter v vsaki lokalni celici $L_{i,j}$ ločeno. To pomeni, da če imamo 6 lokalnih celic $\{L_{1,1}, L_{1,2}, ..., L_{3,2}\}$, potem se atribut a_{s_k} sedaj preslika v 7 dimenzij učnega vektorja $\vec{u}$. Atributi iz celotne slike S se preslikajo v eno dimenzijo po enačbi (2.9) ali (2.33), medtem ko je treba za lokalne celice uporabiti naučene modele semantičnih atributov ter se zanje atribut a_{s_k} sedaj preslika po enačbi (2.33). Pri lokalnih celicah se lahko uporabi le ena enačba, saj v *'ground truth'* podatkih običajno nimamo podanih informacij o lokalnih atributih, in je treba te atribute napovedati iz semantičnih modelov m_{a_s}, ki smo se jih naučili v prvi fazi. Velikost učnega vektorja spremeni $\vec{u} \in \mathbb{R}^{7\times|\mathcal{A}_s|+|\mathcal{A}_d|}$. Diskriminativnih atributov pri tem nismo računali ločeno za vsako celico, saj je računanje slednjih že za celotno sliko relativno počasno.

Podobno se sedaj spremeni tudi pri klasifikaciji objekta, tako da izračunamo bazno značilko za celotno sliko $\mathcal{B}_{\mathcal{S}}$ in vsako celico $\mathcal{B}_{L_{i,j}}$, ter nato za vsakega

ločeno izračunamo ustrezen semantičen atribut a_{s_j} in ga ustrezno preslikamo v testni vektor za klasifikacijo objekta $\vec{t} \in \mathbb{R}^{7\times|\mathcal{A}_s|+|\mathcal{A}_d|}$, ki mora ustrezati velikosti učnega vektorja $\vec{u}$ iz učenja kategorij.



Slika 2.10: Podroben prikaz postopka klasifikacije objekta z ločeno lokalizacijo.

Problem te rešitve je, da je sedaj treba napovedovati za vsako sliko $\mathcal{S}$, za vsako celico $L_{i,j}$ in za vsak semantični atribut a_{s_j}. Če smo pred tem imeli časovno zahtevnost le $O(N \times M)$, kjer je N število slik in M število semantičnih atributov, imamo sedaj časovno zahtevnost $O(N \times M \times (K+1))$, kjer je K število lokalnih celic ($K = 6$ v našem primeru). Ta slabost se zaradi narave rešitve sedaj pojavi tako pri učenju kot tudi pri klasifikaciji objektov,

zato je potreba dodatne mere previdnosti pri izbiri števila lokalnih celic.

Izkaže se, da je to izboljšavo vredno vpeljati, kadar se želimo naučiti nove kategorije z že obstoječimi klasifikatorji za atribute $m_{atribut}$. V teh primerih se lahko klasifikacijsko točnost izboljša za skoraj 50 odstotkov, medtem ko se v primerih kategorizacije že obstoječih kategorij izboljša le za nekaj odstotkov.

2.3.3 Popravilo pristranskosti pri strojnem učenju

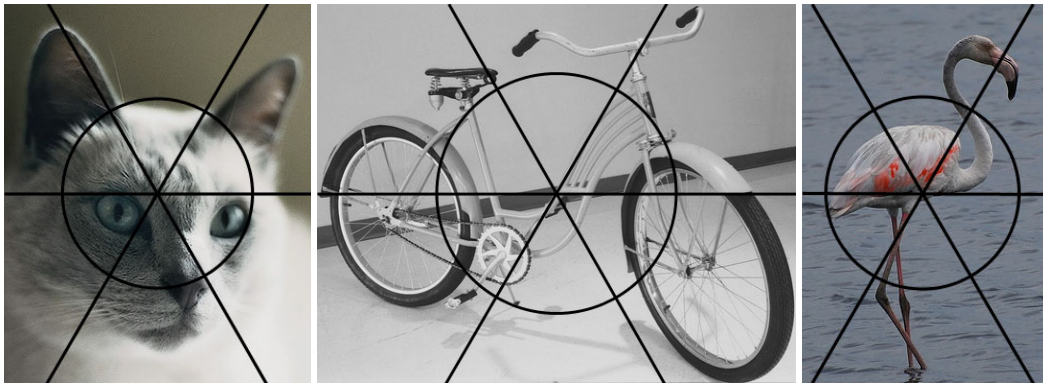
Baze slik, ki smo jih uporabili za testiranje, so se v nekaterih primerih izkazale za zelo pristranske, tako da je za nekatere kategorije objektov obstajalo veliko učnih primerov, za druge pa manj. Podoben problem je tudi pri učnih podatkih za semantične atribute, kjer je običajno razmerje med pozitivnimi in negativnimi primeri lahko tudi $1:9$. V ta namen je treba algoritme strojnega učenja dodatno obvestiti o takem razmerju, zato je v naslednjih poglavjih pri nekaterih algoritmih mogoče opaziti, da je testiranje potekalo tako s pristranskostjo kot tudi z odpravljeno pristranskostjo. V primerih, kjer se je odprava pristranskosti izkazala za uspešno, se je te algoritme uporabljalo tudi v naslednjih testiranjih.

2.3.4 Krožno vzorčenje

Namesto pravokotnega vzorčenja za lokalizacijo se uvede krožno vzorčenje, ki se je v našem primeru bolje izkazalo pri hierarhičnih delih. V nasprotju s pravokotnim vzorčenjem, kjer se nad sliko položi mrežo, se pri krožnem vzorčenju sliko razdeli v kroge, ki se vijejo iz sredine. Med dvema krogoma tako dobimo kolobarje, ki predstavljajo eno skupino celic. Vsako skupino pa se lahko še naprej razdeli na enakomerne dele, katere označujemo kot celica $L_{i,j}$. Za vsak slikovni element $x \in \mathcal{S}$ to pomeni, da se izračuna njegovo razdaljo od centra slike $r = |x - c_{\mathcal{S}}|$, ter to dolžino kvantizira na 2 razdalji r_q (glede na število krogov). Nato se tudi pogleda, pod kakšnim kotom $\theta = \arctan(x + c_{\mathcal{S}})$ se slikovni element x nahaja glede na center slike $c_{\mathcal{S}}$, ter se njegovo orientacijo še dodatno kvantizira na 6 delov θ_q. Glede na različne kombinacije obeh kvantizacij iz tega dobimo $2 \times 3 = 12$ različnih celic $L_{i,j}$ plus dodatno celico za celotno sliko $L_{\mathcal{S}}$.

2.3.5 Izboljšani diskriminativni atributi

Obstoječi diskriminativni atributi se v naših testiranjih niso povsem dobro odrezali. Možni razlog za to je lahko, da se je algoritem preveč prilagodil



Slika 2.11: Primer slik, razdeljenih po krožnem vzorčenju v šest orinetacij ter dve razdalji.

učnim podatkom. To bi sicer lahko rešili z učenjem na ločeni podmnožici slik, vendar pa to ne reši še dveh dodatnih problemov. Prvi je sama hitrost učenja atributov, ki je zaradi naključnosti razdelitve lahko relativno majhna. Drugi problem pa je, da so ti atributi predvsem prilagojeni obstoječim kategorijam. Torej so v primeru, da se želimo naučiti nove kategorije, so obstoječi atributi povsem neprimerni, saj bi bilo vse diskriminativne atribute treba ponovno izračunati skupaj z obstoječimi in novimi kategorijami. Poleg tega se pri računanju obstoječih diskriminativnih atributov ne uporablja nobene informacije o tem, med katerimi kategorijami algoritem strojnega učenja ni sposoben razlikovati. Obstoječi algoritem se zanaša le na verjetnost, da se bo z velikim številom naključnih razdelitev med temi našel tudi tak atribut, ki bo rešil problem. Zato je lahko iskanje teh atributov na koncu neučinkovito ter počasno.

Naš predlagani novi algoritem za izračun diskriminativnih atributov se zanaša na ugotovitev, da je možno z učenjem le na obstoječih atributih dobiti dodatno informacijo o kategorijah, ki jih s strojnim učenjem ni možno razlikovati. Iz teh informacij je možno dobiti seznam parov kategorij (r_a, r_b, k), med katerimi se klasifikator velikokrat zmoti. Te pare je mogoče uporabiti za razdelitev učne množice na dva dela $\mathcal{U}_a, \mathcal{U}_b$ ter se to razdelitev eksplicitno naučiti na obstoječih baznih značilkah z L1-reg. L2-izguben SVC iz libLinear s privzetimi nastavitvami (C = 1, epsilon = 0,01). Pri tem ni bilo uporabljenega nobenega dodatnega odpravljanja pristranskosti zaradi nesorazmernega števila učnih primerov. Iz rezultata tega dodatnega učenja $m_{novi.disc.atri}$ lahko sedaj za vsako sliko (oz. bazno značilko slike) dobimo ustrezno napoved, ali slika pripada razredu r_a ali razredu r_b, ki ju sedaj preslikamo v ustrezen učni ali testni vektor $\vec{u}$ oz. $\vec{t}$ kot dve dodatni dimenziji vektorja. Ena dimenzija identificira kategorijo r_a, druga pa r_b. Iz kombinacije obeh atributov lahko diskriminativno

strojno učenje še vedno najde tudi primere, ki ne spadajo ne v eno ne v drugo skupino. Na koncu se kategorije nauči skupaj z obstoječimi atributi (lahko so semantični in diskriminativni atributi iz učenja drugih kategorij) ter novimi diskriminativnimi atributi. S tem algoritmom se naučimo le tiste diskriminativne atribute, za katere vemo, da bodo pripomogli h klasifikaciji kategorij, ki jih semantični atributi niso sposobni razlikovati. To prinese manjše število dodatnih atributov, hitrejši izračun, ter posledično je dodajanje novih kategorij relativno enostavno ter hitro.

Pri implementaciji diskriminativnih atributov sta pomembni dve podrobnosti. Prva je način ovrednotenja vrednosti k, ki določa, s kakšno gotovostjo bo klasifikator zamenjeval kategoriji r_a in r_b. Ta podatek je možno dobiti pri nekaterih algoritmih strojnega učenja, ki za vsako kategorijo interno zgradijo ločen model. Interna klasifikacija nato poteka tako, da se za vsak učni vektor slike $\vec{u}$ klasificira za vsako kategorijo $r_i \in \mathcal{K}$ ločeno, iz česar nato za en primer dobimo $n = |\mathcal{K}|$ glasov $[g_{r_1}, g_{r_2}, ..., g_{r_n}]$, ki se jih nato uporabi za glasovanje končne kategorije. Te vmesne glasove je možno uporabiti za ugotavljanje gotovosti k. Potem, ko kategorije naučimo z obstoječimi atributi, lahko celotno učno množico klasificiramo z dobljenim modelom strojnega učenja $m_{kategorije}$ ter iz rezultatov izluščimo ustrezna interna glasovanja $[g_{r_1}, g_{r_2}, ..., g_{r_n}]$ za vsak primer. Za vsako kategorijo r_i lahko nato zberemo vsa glasovanja učnih primerov, ki spadajo v to kategorijo, ter jih povprečimo in normaliziramo v $[g_{r_1}, g_{r_2}, ..., g_{r_n}]_{r_i}$. Iz tega lahko nato enostavno sestavimo matriko vseh glasovanj za vse kategorije skupaj velikosti $n \times n$:

$$\begin{matrix} [0, g_{r_2}, ..., g_{r_n}]_{r_1} \\ [g_{r_1}, 0, ..., g_{r_n}]_{r_2} \\ \vdots \\ [g_{r_1}, g_{r_2}, ..., 0]_{r_n} \end{matrix} \tag{2.34}$$

Iz te matrike je zelo enostavno ugotoviti med, katerimi kategorijami obstajajo problemi, ter iz nje izluščiti pare, ki se jih je treba naučiti. Vse simetrične pare se obravnava ločeno, saj je interpretacija para (r_a, r_b, k) lahko različna od (r_b, r_a, h). Če imamo par $(»mačekˮ, »pesˮ, k)$, to pomeni, da imamo mačka, ki ga klasifikator zamenja za psa z določeno gotovostjo k. Če zamenjamo kategoriji v par $(»pesˮ, »mačekˮ, h)$, pa interpretiramo kot kategorijo »pesˮ, pri čemer je zamejena z mačkom z neko drugo gotovostjo h. Čeprav bodo na koncu nekateri binarni atributi podvojeni, lahko vsaj iz teh atributov ugotovimo, za katero interpretacijo so bili ustvarjeni. Pri identifikaciji problematičnih primerov je zelo pomembno, kakšen prag za gotovost k uporabimo. Če vzamemo prevelik prag, se lahko zgodi, da ne dobimo nobenih parov, kar bi povsem

izničilo uporabo novih atributov, vendar je po drugi strani to lahko povsem upravičeno, če dejansko ne obstajajo problemi med kategorijami. V primeru, da je prag postavljen prenizko, pa bo klasifikacija sicer lahko boljša, vendar dobimo ogromno število nepotrebnih parov, za katere je treba ustvariti nov atribut, kar je lahko na koncu zelo potratno. V naši implementaciji smo za prag uporabili vrednost 0,85, ki se je v naših primerih najbolje obnesla.

Druga pomembna podrobnost pri implementaciji je uporabljen algoritem strojnega učenja za razlikovanje med problematičnimi pari kategorij. Ker je lahko teh parov relativno veliko, mora biti algoritem sposoben hitrega učenja in klasifikacije, saj se lahko v nasprotnem primeru čas učenja in klasifikacije zviša na povsem neuporabno raven.

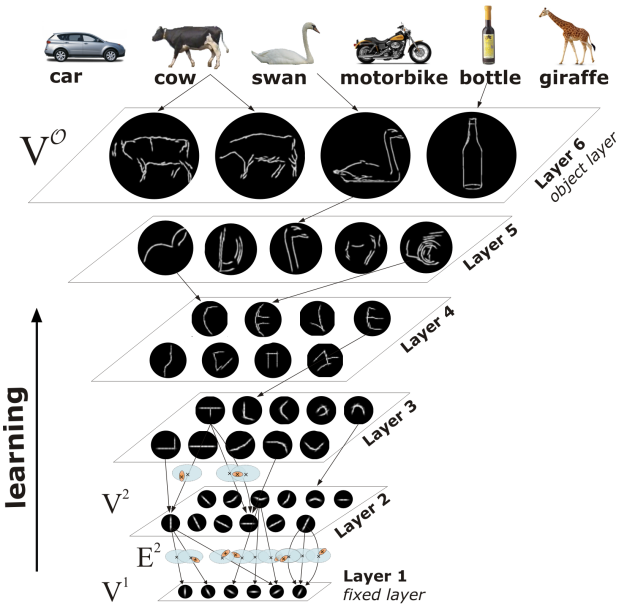
Poglavje 3

Uporaba atributnega učenja z modelom LHOP

3.1 Koncept modela LHOP

Model naučene hierarhije delov ali LHOP (ang. *Learned Hierarchy of Parts*) predstavlja hierarhičen način učenja oblike za različne vizualne kategorije po principu od spodaj navzgor. Vsako kategorijo predstavimo s specifično obliko, le-ta pa je modelirana iz hierarhije manjših in enostavnejših delov, kot je prikazano na sliki (3.1). Koncept sloni na štirih glavnih lastnostih, ki jih mora model vsebovati. To so možnost skaliranja modelov na večjo možico kategorij, hitro statistično in inkerementalno učenje ter robustna detekcija. Celotno osnovo za učenje predstavljajo množica odzivov na enostavne filtre. V praksi se uporablja različne orientacije Gaborjevih filtrov, katerih odziv predstavlja enostavne dele, kot sta horizontalna in vertikalna črta. Ti deli nato predstavljajo najnižji del hierarhije in se jih postopoma združuje v vse bolj kompleksne dele. Namen hierarhičnega načina učenja je, da se lahko iz slik izlušči dele, ki so skupni vsem kategorijam, te skupne dele pa se nato kombinira v bolj kompleksne dele, specifične za določene kategorije. S tem je možno v model integrirati veliko število različnih kategorij ter enostavno dodajati nove kategorije. To pa sta tudi dve glavni lastnosti, ki ju mora model izpolnjevati.

Učenje modela LHOP se prične z odzivi različnih Gaborjevih filtrov, ki predstavljajo enostavne dele na prvem (najnižjem) nivoju. Te dele se iz slik vseh kategorij uporabi za učenje bolj kompleksnih delov, ki so na drugem nivoju. Glavnega pomena pri tem je način učenja novih delov. Učenje je predvsem statistično usmerjeno ter je prilagojeno veliki kompleksnosti, ki jo prinaša množica različnih kombinacij načina sestave kompleksnega dela iz bolj



Slika 3.1: Predstavitev hierarhičnega modeliranja oblike posamezne kategorije z enostavnejšimi deli. V spodnjih nivojih je možno opaziti zelo enostavne oblike (npr. različne orientacije črt, enostavni sklepi itd.), v višjih nivojih, pa se le-ti združijo v bolj kompleksne oblike, primerne za posamezne kategorije, kot so avto, krava, labod itd. Učenje hierarhije poteka od spodnjih nivojev proti zgornjim. Slika je povzeta iz [57].

enostavnega dela. Učenje na vseh slikah se običajno ponovi do tretjega oz. četrtega nivoja, nato pa se vse nadaljnje dele uči ločeno za vsako kategorijo. Ta način učenja omogoča, da si kategorije med sabo delijo določene dele, vendar to običajno ni dovolj za uspešno kategorizacijo, zato se višje nivoje nauči ločeno za vsako kategorijo. S tem je omogočeno tudi enostavno dodajanje novih kategorij, saj je treba nove dele dodati le na višjih nivojih, medtem ko so spodnji nivoji lahko skupni vsem kategorijam.

3.2 Model LHOP in učenje z atributi

Za metodo učenja kategorij z atributi ter modelom LHOP je mogoče opaziti, da imata določene skupne točke. Oba omogočata učenje večje množice kategorij ter pri učenju uporabljata skupne lastnosti kategorij za hitrejše in učinkovitejše učenje. Pri učenjem z atributi te skupne značilnosti predstavljajo semantični atributi, medtem ko so pri modelu LHOP to deli do 3. nivoja, ki se jih uči na vseh kategorijah. Vsaka metoda pa ima tudi svoje prednosti. Po eni strani učenje z atributi omogoča analizo atributov ter učenje in klasifikacijo objektov

brez slik, medtem ko model LHOP te sposobnosti nima. Po drugi strani pa je model LHOP zelo dober pri učenju kompleksnejših oblik kategorij, saj se opira na model HMAX, ki izvira iz preučenaja biloških vizualnih sistemov. S tega vidika bi bilo smiselno obe metodi združiti ter pridobiti prednosti obeh. V ta namen smo v naslednjem poglavju predstavili eno izmed možnosti, kako bi lahko model LHOP vpeljali v sistem atributnega učenja.

3.3 Integracija značilk LHOP v atributno učenje

Model LHOP lahko v učenje z atributi vstopa kot del bazne značilke $\mathcal{B}$, ki se uporabi za učenje semantičnih in diskriminativnih atributov. LHOP je po naravi omejen le na uporabo vizualnih delov oz. oblike, in ne uporablja nobene informacije o barvi ali teksturi. Posledično modela samega ni možno direktno uporabiti za učenje atributov, saj se atributi zelo razlikujejo in je veliko tudi takih, ki jih ni mogoče uporabiti brez barve ali teksture. To pomeni, da je, če želimo model LHOP uporabiti za učenje semantičnih atributov, treba v bazno značilko poleg značilk LHOP vključiti tudi značilki za barvo in teksturo. Efektivno lahko tako nadomestimo le značilko HOG in značilko za robove.

Bazna značilka se sedaj nadgradi tako, da se odstrani značilki za HOG $\mathcal{H}$ in robove $\mathcal{D}$ ter se doda značilko za LHOP $\mathcal{L}$.

$$\mathcal{B} = \left[\begin{array}{cccc} \mathcal{C}, & \mathcal{Q}, & \mathcal{L} \end{array} \right] \tag{3.1}$$

3.3.1 Značilka LHOP

Značilko označimo z $\mathcal{L}$ ter predstavlja histogram vseh delov na 3. in/ali 2. nivoju, ki se jih model LHOP nauči. Dele le do tretjega nivoja uporabljamo zato, ker se bazna značilka uporabi le za učenje atributov, ti pa morajo biti skupni različnim kategorijam. To se ujema s prvim delom učenja hierarhije delov, kjer se do 3. nivoja nauči dele, ki so skupni vsem kategorijam.

Knjižnica hierarhičnih delov je bila naučena na popolnoma neodvisni množici slik, in sicer na takih primerih, ki so najbolje primerni za naravne slike. Naučena knjižica vsebuje dele le do 3. nivoja, kjer je 10 delov na drugem nivoju ter 914 delov na tretjem nivoju. Vsem slikam, na katerih iščemo dele, predhodno spremenimo velikost, tako da se največjo stranico zmanjša oz. poveča na 300 slikovnih elementov, drugo stranico pa se spremeni tako, da se ohrani originalno razmerje stranic. Dele poiščemo tudi na različnih skalah z uporabo prostorske piramide, pri čemer se skala spreminja s faktorjem $\sqrt{2}/2$. Glede na

velikost vsake slike in faktorja skale se na koncu dele za vsako sliko poišče na štirih različnih skalah. Izmed vseh delov, najdenih na vsaki sliki, ignoriramo tiste, ki imajo obtežitev manjšo od 0,1. Za vse ostale dele pa nato naredimo histogram oz. seznam pogostosti pojavitev določenega tipa dela. Histogram naredimo za 2. in 3. nivo ločeno, nato pa ju združimo v skupno značilko oz. uporabimo vsakega ločeno; odvisno od tega, ali testiramo 2. nivo, 3. nivo ali oba skupaj. Značilko na koncu še normaliziramo z normo ℓ^2.

Poglavje 4

Testi in rezultati

4.1 Uporabljene baze slik

Za učenje in testiranje uporabljamo popolnoma iste množice slik kot v [23], ter še eno dodatno:

- aPascal (PASCAL VOC 2008)
- aYahoo
- Caltech 101 [54]

4.1.1 Baza aPascal

Baza aPascal sloni na bazi PASCAL VOC 2008. Vsebuje 20 kategorij, ki jih lahko smiselno razdelimo v skupine »živali«, »vozila« in »stvari«: ljudje, ptič, mačka, krava, pes, konj, ovca, letalo, kolo, čoln, avtobus, avto, motorno kolo, vlak, steklenica, stol, miza, lončnice (roža), kavč in TV/monitor. Vsi primeri objektov so vzeti iz naravnega okolja, zato načeloma za vsak objekt nimamo vedno slike iz vsake poze, osvetlitve in orientacije. Vendar pa se z dovolj velikim številom primerov za vsako kategorijo poskuša zajeti čim več različnih kombinacij orinetacij in osvetlitev, kot se pojavljajo v realnem okolju. Pri tem se število primerov na kategorijo močno razlikuje, saj se gliblje med 150 in 1000 oz. tudi do 5000 primerov za kategorijo ljudje.

Slike iz baze PASCAL VOC 2008 še niso povsem primerne za učenje atributov, saj za vsak objekt na sliki potrebujemo ustrezne oznake semantičnih atributov. Te označbe so ročno naredili avtorji [23], pri tem pa še dodatno uprabili storitev Amazon's Mechanical Turk. Oznake avtorjev in označevalcev

iz Amazon Turk so združili skupaj, pri čemer je prišlo do razlikovanja v le manj kot 20 % primerov. Za vsak objekt s slike so s tem dobili seznam, katere izmed 64 semantičnih atributov objekt vsebuje.

kategorija	**Učna množica**	**Testa množica**	**Skupaj**
avtobus	73	77	*150*
avto	533	489	*1022*
čoln	232	221	*453*
kavč	121	132	*253*
kolo	161	146	*307*
konj	153	153	*306*
krava	103	94	*197*
letalo	184	186	*370*
ljudje	2500	2571	*5071*
mačka	187	193	*380*
miza	113	111	224

kategorija	**Učna množica**	**Testa množica**	**Skupaj**
motorno kolo	150	147	297
ovca	123	111	234
pes	244	240	484
ptič	254	295	549
lončnice (roža)	225	221	446
steklenica	297	279	576
stol	447	464	911
TV/monitor	150	149	299
vlak	90	86	176
Skupaj	***6340***	***6365***	***12705***

Tabela 4.1: Število primerov po kategorijah v bazi aPascal glede na učno in testno množico.

Baza aPascal je še dodatno razdeljena na učno množico $\mathcal{U}$ in testno množico $\mathcal{T}$. Učna množica $\mathcal{U}$ ustreza slikam iz učne podmnožice PASCAL VOC 2008, medtem ko testna množica $\mathcal{T}$ utreza validacijski podmnožici iz PASCAL VOC 2008. V učni množici se nahaja 6340 primerov iz 2113 slik, v testni množici pa 6365 primerov iz 2227 slik. Podrobno število slik na kategorijo je podano v tabeli (4.1).

Popolno bazo je tako možno dobiti na domači strani članka [23] `http://vision.cs.uiuc.edu/attributes`, same slike pa tudi na dimači strani PASCAL VOC 2008 tekmovanja `http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2008`.

4.1.2 Baza aYahoo

Druga baza, ki jo uporabljamo, je množica slik, poiskanih na spletni strani Yahoo. To bazo so ustvarili v [23] in vsebuje 12 kategorij, ki jih ni v bazi aPascal. Namen teh novih kategorij je preiskusiti, kako dobro se obnesejo semantični atributi naučeni na kategorijah aPascal, torej, kakšen je problem generalizacije. Nove kategorije so izbrane tako, da imajo podobne atribute kot razredi

v množici aPascal, tako da se semantični atributi lahko enostavno prenesejo na novo bazo. Te kategorije so: jetski, kentaver, kip, kočija, koza, opica, osel, skodelica, stavba, torbica, volk in zebra. Objekti na slikah se prav tako nahajajo v naravnih okoljih, tako da se poskuša s čim večjim številom primerov na kategorijo zagotoviti objekte v različnih orinetacijah in osvetlitvah. Vseh objektov skupaj je 2644 na 2238 različnih slikah. Vse slike so prav tako ročno označene za ustrezne kategorije in semantične atribute s strani avtorjev ter Amazon's Mechanical Turk.

Kdar uporabimo množico tudi za učenje, takrat za učno in testno bazo uporabljamo naključno razdelitev v 30 primerov na kategorijo za učno in 30 primerov za testno bazo oz. v primeru nekaterih kategorij lahko uporabimo tudi manj primerov za testno množico, če nimamo dovolj primerov. V teh primerih za večjoo natančnost večino testov ponovimo vsaj 3-krat, kadar se testi lahko izvedejo hitreje, pa tudi 5-krat.

Popolno bazo je prav tako možno dobiti na domači strani članka `http://vision.cs.uiuc.edu/attributes`.

kategorija	**Število primerov**
jetski	399
kentaver	51
kip	207
kočija	152
koza	163
opica	186

kategorija	**Število primerov**
osel	139
skodelica	226
stavba	323
torbica	349
volk	205
zebra	244
Skupaj	***2644***

Tabela 4.2: Število primerov po kategorijah v bazi aYahoo.

4.1.3 Baza Caltech 101

V naših testih smo uporabili še dodatno množico slik iz baze Caltech 101, ki vsebuje 101 različnih kategorij. Za vsako kategorijo baza vsebuje od 40 do 800 primerov, večina kategorij pa vsebuje okoli 50 primerov. Objekti se tudi v tem primeru večinoma nahajajo v realnih okoljih z različno osvetlitvijo in orientacijo. Ker ima baza označene le kategorije, in nima semantičnih atributov, načeloma ni uporabna za učenje semantičnih atributov, vendar jo še vedno lahko uporabimo za klasifikacijo objektov, saj lahko atribute še vedno enostavno napovemo z naučenimi modeli semantičnih atributov. Ker se kategorije

te baze tudi razlikujejo od kategorij, s katerimi smo učili semantične atribute, je ta baza dober pokazatelj, kako smo se naučili semantične atribute oz. kako dobro lahko posplošimo atribute na druge kategorije.

Za učno in testno množico smo pri testiranju uporabili naključno razdelitev v 30 primerov na kategorijo za obe množici (če ni bilo dovolj primerov, se je v testni lahko uporabilo tudi manj primerov); nato teste ponovimo vsaj 3-krat, lahko pa tudi 5-krat, kjer je zaradi hitrosti to izvedljivo.

Celotna baza je javno dostopna na spletni strani `http://www.vision.caltech.edu/Image_Datasets/Caltech101/`.

Pri tej bazi je pomembno izpostaviti še, da se običajno testiranje izvaja na celotni sliki, in ne na segmentiranih objektih (tj., lokacija objekta je določena s pravokotnikom oz. ang. *bounding box*), kot je to pri bazi aPascal in aYahoo. Ker v naših testih nisomo uporabili nobene dodatne metode za lokalizacijo oz. določanje objekta na sliki, smo v nekaterih primerih naredili teste s tremi različnimi kombinacijami učenja. Pri prvi smo učili kategorije na segmentiranih objektih ter testirali na celotni, pri drugi smo učili in testirali na celotni sliki, pri tretji pa smo učili in testirali na segmentiranih objektih. Slednja predstavlja idealno situacijo, v kateri uspemo lokalizirati objekt s 100% natančnostjo.

4.2 Uporabljene meritve

Za primerjavo rezultatov smo uporabili dvoje meritev. Pri učenju semantičnih atributov uporabljamo aproksimacijo AUC iz ene točke ROC [30]. Vse rezultate klasifikacije zapišemo kot matriko napak M_e.

$$M_e = \begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix} \tag{4.1}$$

Iz te matrike lahko dobimo le eno točko ROC, saj imamo diskretne klasifikatorje, ki vrnejo le binarno vrednost, in ne verjetnosti. Najprej izračunamo $fp_{rate} = \frac{FP}{FP+TN}$ in $tp_{rate} = \frac{TP}{TP+FN}$, nato pa te vrednosti uporabimo za točko $ROC = (fp_{rate}, tp_{rate})$. Iz te točke v prostoru ROC lahko sedaj izračunamo aproksimacijo vrednosti AUC, tako da enostavno seštejemo vrednosti trikotnikov pod krivuljo.

$$AUC_{aproks} = \frac{fp_{rate} + tp_{rate}}{2} + (1 - fp_{rate}) \times tp_{rate} + \frac{(1 - fp_{rate}) \times (1 - tp_{rate})}{2} \tag{4.2}$$

Pri učenju kategorij pa običajno uporabljamo klasifikacijsko točnost *ca*. Pri klasifikacijski točnosti poročamo tako *skupno točnost* kot tudi *povprečno točnost* preko vseh kategorij. Prvo izračunamo tako, da za vse testne primere seštejemo pravilne klasifikacije $T = TP + TN$ in napačne klasifikacije $F = FP + FN$, ter dobimo skupno klasifikacijsko točnost:

$$ca_{skupna} = \frac{T}{T + F} \tag{4.3}$$

Povprečno klasifikacijsko točnost po razredih pa dobimo tako, da zgornjo enačbo (4.3) izračunamo ločeno za vsak razred $r_i \in \mathcal{K}$, ter nato izračunamo povprečno vrednost iz klasifikacijskih točnosti vseh kategorij:

$$ca_{povprečna} = \frac{\sum ca_{r_i}}{|\mathcal{K}|} \tag{4.4}$$

Povprečno točnost uporabljamo, ker se lahko število primerov na razred močno razlikuje, ter imajo razredi z veliko primeri po enačbi (4.3) prevelik vpliv na končni rezultat.

4.3 Učenje z atributi

4.3.1 Vrste testov

V splošnem teste razdelimo na dve skupini. V prvi je učenje in klasifikacija semantičnih atributov, v drugi pa je učenje in klasifikacija kategorij. Načeloma uporabljamo bazo aPascal v obeh skupinah, predvsem se učno množico aPascal uporablja za učenje atributov, le-ti pa se nato uporabijo za učenje kategorij v drugi skupini testov. Množico slik aYahoo pa uporabljamo pri učenju in klasifikaciji novih kategorij, torej kadar že imamo klasifikatorje atributov naučene na učni množici aPascal ter se želimo naučiti kategorije, ki niso bile v učni množici aPascal.

1. Učenje in klasifikacija semantičnih atributov:

Pri klasifikaciji atributov za vsak semantični atribut zgradimo ustrezen klasifikator na učni množici aPascal ter naredimo ustrezen test na ločeni bazi slik. Glede na testno bazo lahko teste razdelimo v dve vrsti:

a) Testni podatki imajo iste kategorije kot učni podatki (ang. *within category*). V tem primeru se testira na testni množici aPascal.

b) Testni podatki imajo povsem ločene kategorije (ang. *across category*). Testna množica je v tem primeru baza aYahoo.

V tej skupini testov testiramo tudi razliko med uporabo *vseh značilk* proti uporabi le *izbranih značilk* za učenje klasifikatorjev (ang. *feature selection*).

Za učni algoritem uporabljamo klasifikator OpenCV SVM ter L1-regulariziran L2-izguben SVC (libLinear). Slednji se je izkazal za boljšega in hitrejšega, zato so v večini testiranj druge skupine testov uporabljeni semantični atributi, naučeni s tem algoritmom.

2. Učenje in klasifikacija kategorij:

Pri učenju in klasifikaciji kategorij lahko teste v grobem delimo na:

a) Uporaba le baznih značilk (učenje brez atributov). Iz tega testa je razvidno, kakšno klasifikacijsko točnost dobimo brez atributov.

b) Uporaba le *'gorund truth'* semantičnih atributov za testiranje. Ta test nam pove, koliko bi v idealnih pogojih lahko dobili, če bi bilo možno semantične atribute naučiti se/napovedati s 100% točnostjo.

c) Uporaba le napovedanih semantičnih atributov za testiranje.

d) Uporaba semantičnih in diskriminativnih atributov.

V splošnem vse teste, ki vključujejo semantične atribute, naredimo dvakrat. Enkrat pri učenju atributov uporabimo vse značilke, drugič pa le pomembne značilke (ang. *feature selection*).

Nekatere izboljšave, ki se nanašajo na učenje kategorij, kot na primer učenje razredov na napovedih atributov in odprava pristranskosti, testirmo nad vsemi zgornjimi testi oz. nad tistimi testi, kjer je to smiselno. Druge izboljšave (ločeno učenje za vsako lokalizacijo) pa testiramo le nad določenimi testi in v kombinaciji z nekaterimi drugimi izboljšavami.

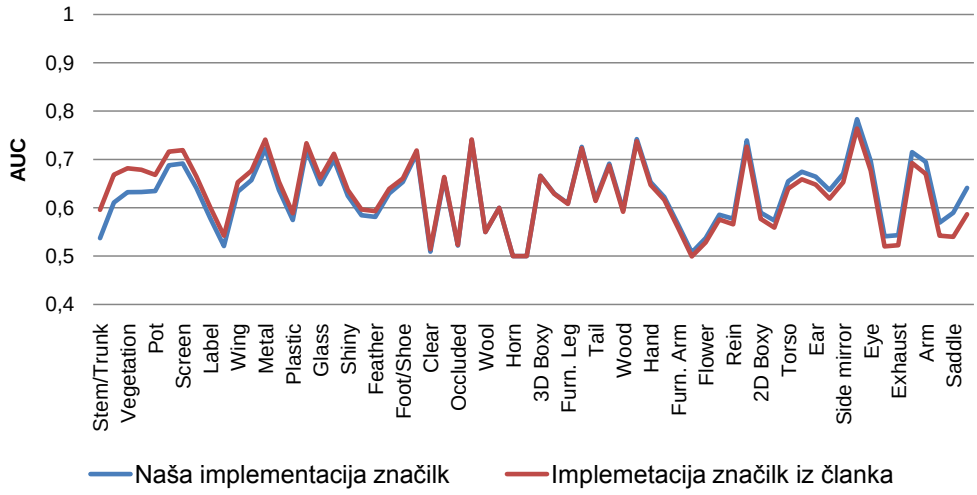
Zaradi ustrezne primerjave z rezultati iz članka v teh testih ne uporabljamo krožnega vzorčenja ter novih diskriminativnih atributov. Le-ti so uporabljeni šele v naslednjem podpoglavju.

Nekatere izmed zgornjih testov primerjamo tudi z atributi, izračunanimi na originalnih značilkah, ki so bile izračunane s pomočjo kode MATLAB iz članka [23]. Ti testi so v rezultatih označeni kot 'implementacija značilk iz članka'. S tem lahko primerjamo pravilnost izračuna naših baznih značilk z implementacijo izračuna značilk iz članka.

4.3.2 Rezultati učenja semantičnih atributov

4.3.2.1 Z aPascal za učno množico $\mathcal{U}$ in testno množico $\mathcal{T}$

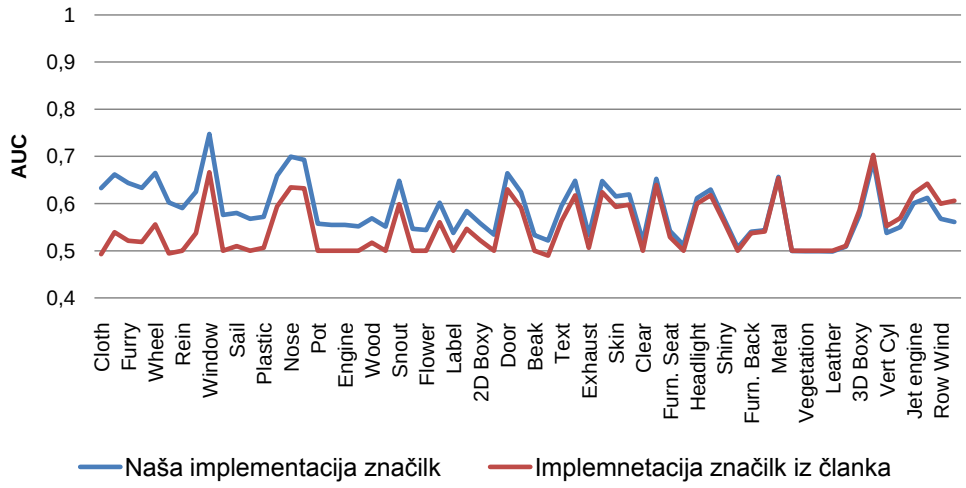
Pri učenju semantičnih atributov z algoritmom OpenCV SVM lahko dosežemo povprečen AUC 0,63, kadar vzamemo vse značilke. V primerjavi z značilkami iz članka pa dobimo povprečen AUC 0,63. Razlika med njima je minimalna (za 0,002), in iz slike (4.1) je mogoče opaziti, da se krivulji skoraj povsem prekrivata, razen v nekaterih primerih, kjer so značilke iz članka boljše pri atributih, kot so »reaktivni motor«, »zaslon« in »steblo«, slabše pa se obnesejo pri atributih »pedal«, »roka«, »jadro« itd.



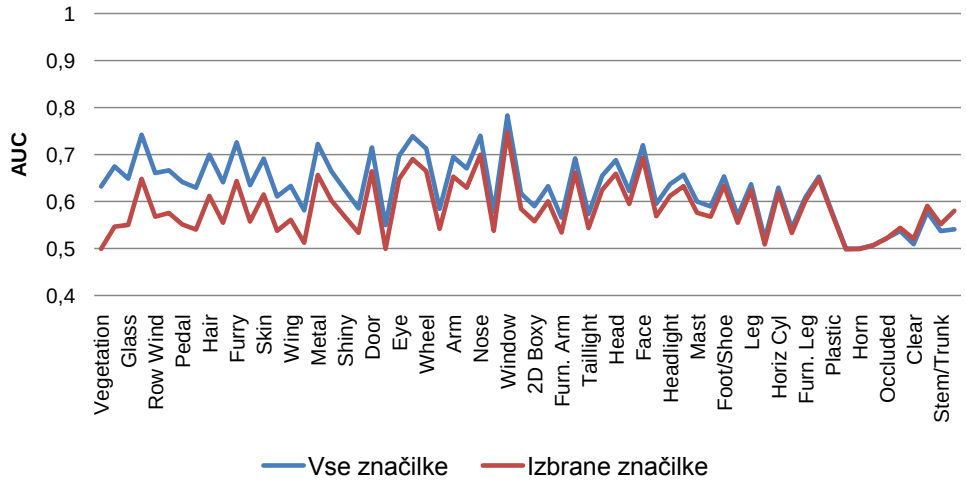
Slika 4.1: Primerjava klasifikacije semantičnih atributov z značilkami iz članka. Pri strojnem učenju smo uporabili vse značilke. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

V primeru, da učimo atribute le na izbranih značilkah, se povprečen AUC zniža na 0,59, pri značilkah iz članka pa na 0,55. Razliko med posameznimi atributi je mogoče opaziti na sliki (4.2). Izkaže se, da je učenje na značilkah iz članka boljše le pri atributih, kot so »letalsko krilo«, »lasje« in »steklo«, vendar pa je veliko slabše pri atributih »zaslon«, »čevelj«, »telo« itd.

Če se omejimo le na našo implementacijo in naredimo primerjavo semantičnih atributov naučenih, na vseh značilkah, in atributov, naučenih le na izbranih značilkah, pridemo do zaključka, da je AUC pri uporabi izbranih značilk slabši za 0,04 točke. Iz slike (4.3) je mogoče opaziti, da je največja sprememba pri atributih, kot so »rastlinje«, »steklo«, »lasje« itd. Podobne rezultate poročajo tudi v [23], vzrok za to pa je mogoče iskati v uporabljeni



Slika 4.2: Primerjava klasifikacije semantičnih atributov z značilkami iz članka. Pri strojnem učenju smo uporabljali le *izbrane značilke*. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

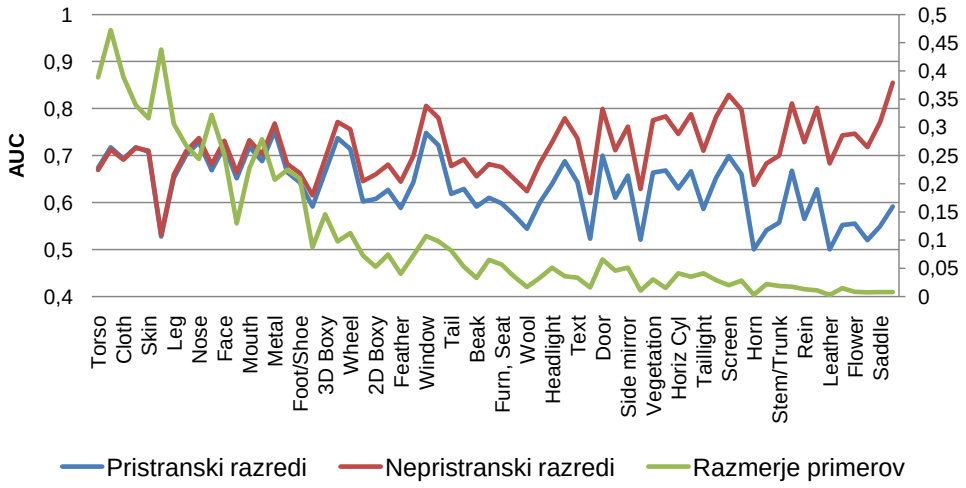


Slika 4.3: Primerjava klasifikatorjev semantičnih atributov, naučenih na vseh značilkah, in atributov, naučenih le na *izbranih značilkah*. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

testni množici, kjer se nahajajo iste kategorije kot v učni množici. V obeh primerih sta uporabljeni množici aPascal z enakimi kategorijami, ter se zato semantični atributi lahko pojavljajo skupaj v učni in testni množici. Pri tem pa smo zaradi postopka izbiranja značilk dejansko prekinili povezavo med atributi, ki se pojavljajo skupaj, zato jih v tem primeru v testni množici težje zaznamo.

Vsi zgornji rezultati uporabljajo za algoritem strojnega učenja OpenCV SVM, ki pa se je v primerjavi z L1-regulariziranim L2-izgubnim SVC-jem izkazal za slabšega. Pri slednjem dobimo v primeru učenja na vseh značilkah povprečen AUC 0,64, pri učenju le z izbranimi značilkami pa 0,61. Izkaže se tudi za veliko hitrejšega in manj potratnega glede na velikost modelov strojnega učenja, zato se večinoma uporablja SVC. Poleg tega se ob odpravi pristranskosti klasifikatorjev AUC izboljša še za med 0,05 in 0,10 točke.

Rezultati naših dodatnih izboljšav: Izboljšavi, ki vplivata tudi na semantične atribute, sta odprava pristranskosti ter uporaba ločene lokalizacije. Za odpravo pristranskosti je povsem logično, da je pomembna pri učenju semantičnih atributov, za ločeno lokalizacijo pa to ni tako očitno. Razlika pri uporabi ločene lokalizacije je, da se semantične atribute uči le nad celotno sliko, kar pomeni, da je uporabljena značilka 1393-dimenzionalna v nasprotju z 9751-dimenzionalno značilko pri uporabi skupne lokalizacije. Rezultati izboljšav se nanašajo na učenje z L1-regulariziranim L2-izgubnim algoritmom SVC, ki se je predhodno izkazal za uspešnejšega.

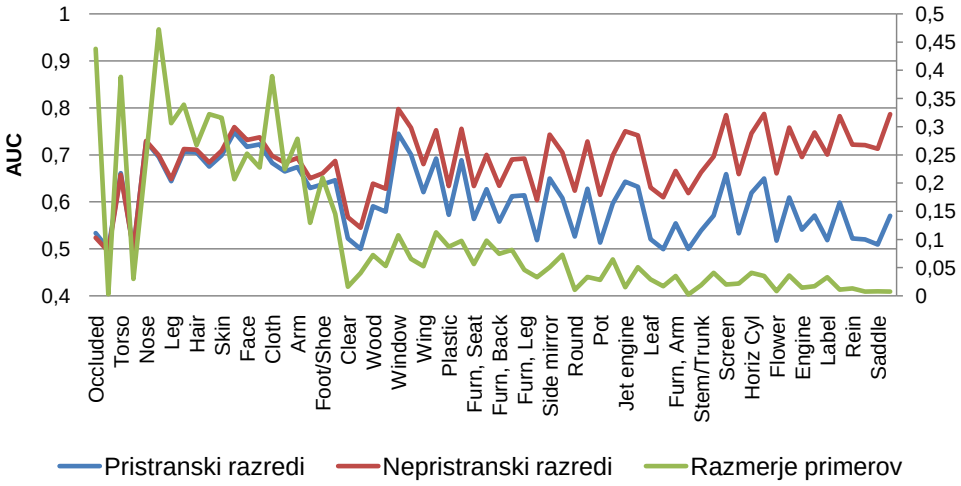


Slika 4.4: Primerjava med učenjem atributov s pristranskimi in nepristranskimi razredi v primeru učenja na vseh značilkah. Primeri so razvrščeni glede na razliko med pristranskimi in nepristranskimi razredi. Naključna vrednost je 0,5. Graf prikazuje tudi razmerje primerov za vsak semantičen atribut.

Odprava pristranskosti močno pripomore k boljši klasifikaciji, saj je povprečen AUC pri učenju z vsemi značilkami 0,71 ter 0,68 pri učenju le z izbranimi značilkami. Podrobne razlike med posameznimi atributi je možno opaziti na

sliki (4.4) in (4.5). Pri tem se leva skala (AUC) nanša na modre in rdeče podatke, desna skala pa na zelene podatke, ki prikazujejo razmerje pozitivnih proti vsem učnim primerom za vsak semantični atribut.

Največja prednost pri odpravi pristranskosti se pokaže pri atributih, kot so »sedlo«, »jadro«, »roža« itd., medtem ko je pri atributih, kot so »telo«, »noga« in »nos«, razlika praktično zanemarljiva. Iz obeh grafov je tudi zelo očitna korelacija med izboljšavo klasifikacije ter razmerjem primerov. Povsem očito je, da se uspešnost odprave pristranskosti izboljšuje z manjšim številom negativnih primerov. To je povsem skladno z namenom vpeljave te izboljšave.



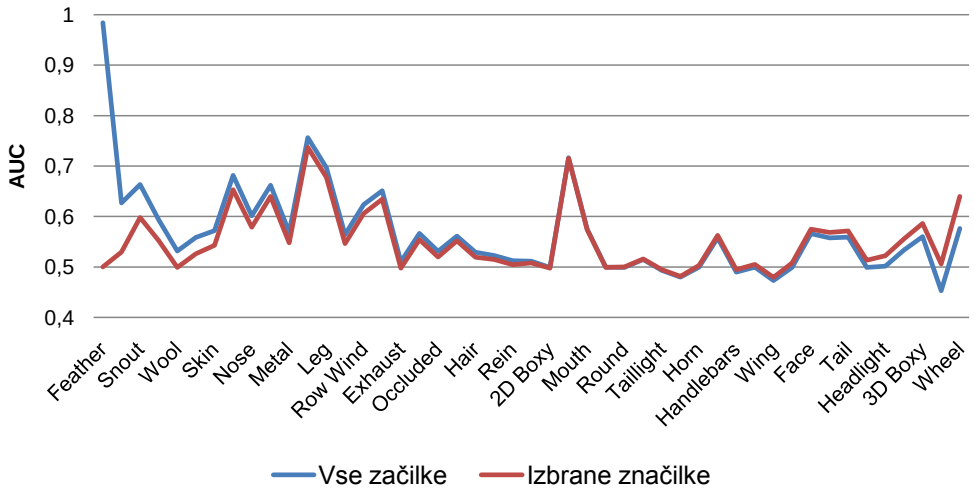
Slika 4.5: Primerjava med učenjem atributov s pristranskimi in nepristranskimi razredi v primeru učenja le na *izbranih značilkah*. Primeri so razvrščeni glede na razliko med pristranskimi in nepristranskimi razredi. Naključna vrednost je 0,5. Graf prikazuje tudi razmerje primerov za vsak semantičen atribut.

Kot omenjeno, je poleg odprave pristranskosti pomembna tudi ločena lokalizacija, katere test pa smo izvedli skupaj z odpravo pristranskosti in L1-reg. L2-izg. SVC učenjem atributov. Z ločeno lokalizacijo sicer dobimo manjšo značilko, vendar to ne vpliva močno na rezultate. Povprečen AUC se celo dvigne za kakšen odstotek z 0,71 na 0,72 pri uporabi vseh značilk in z 0,68 na 0,69 pri uporabi le izbranih značilk.

4.3.2.2 Z aPascal za učno množico $\mathcal{U}$ in aYahoo za testno množico $\mathcal{T}$

Pri naslednjem testu se preverja, kako uspešna je klasifikacija semantičnih atributov na množicah z različnimi kategorijami. Efektivno preverjamo kako velik

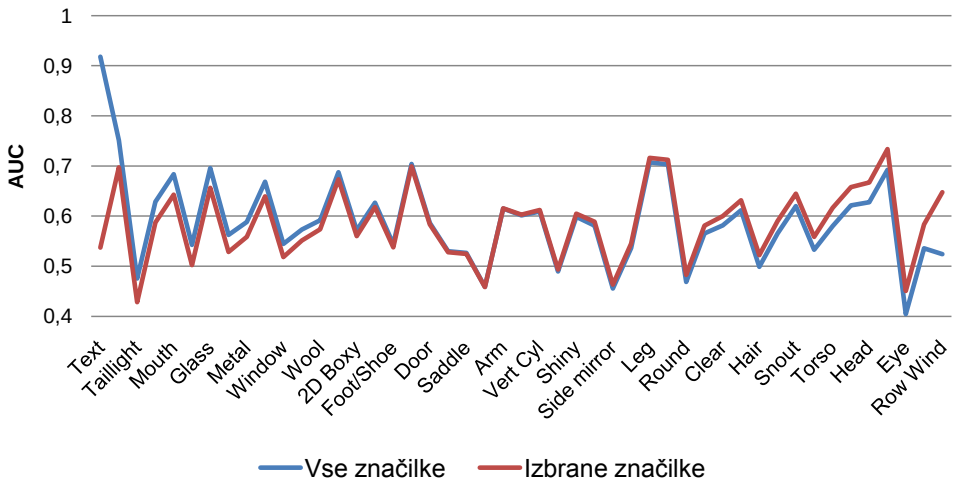
je problem generalizacije. Vse semantične atribute smo naučili na učni množici aPascal z L1-regulariziranim L2-izgubnim SVC strojnim učenjem, nato pa smo jih preverjali na celotni množici aYahoo, kjer imajo primeri različne kategorije. V teh testih smo se omejili le na uporabo naše implementacije. Povprečen AUC pri učenju z vsemi značilkami je 0,57, v primeru uporabe izbranih atributov pa 0,55. Primerjavo klasifikatorjev je možno videti na sliki (4.6), kjer lahko opazimo, da je razlika med atributi, naučenimi z vsemi značilkami, in atributi, naučenimi na izbranih značilkah, relativno majhna, razen v nekaterih izjemah, kot so atributi »perje«, »koža«, »volna«, »kolo« itd.



Slika 4.6: Primerjava klasifikatorjev semantičnih atributov pri testni množici z različnimi kategorijami. Primerjava poteka med modeli naučenih z vsemi značilkami ter primeri naučenih na izbranih značilkah. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

Razlika med obema se sedaj močno zmanjša, iz česar lahko sklepamo, da je postopek izbiranja značilk veliko bolj vpliven pri kategorijah, ki niso v učni množici. To dejstvo se povsem ujema s prvotno predpostavko, da je izbira značilk pomembna pri novih kategorijah.

Rezultati dodatnih izboljšav: Tudi v primeru testiranja na množici z različnimi kategorijami sta pomembni le izboljšavi odprava pristranskosti ter ločena lokalizacija. Odprava pristranskosti prav tako izboljša rezultate, vendar ne tako izrazito kot v primeru testiranja na istih kategorijah. Povprečen AUC se v tem primeru z 0,57 poviša na 0,59 pri učenju z vsemi značilkami in iz 0,55 na 0,59 pri učenju le na izbranih značilkah. Tako iz povprečnih AUC



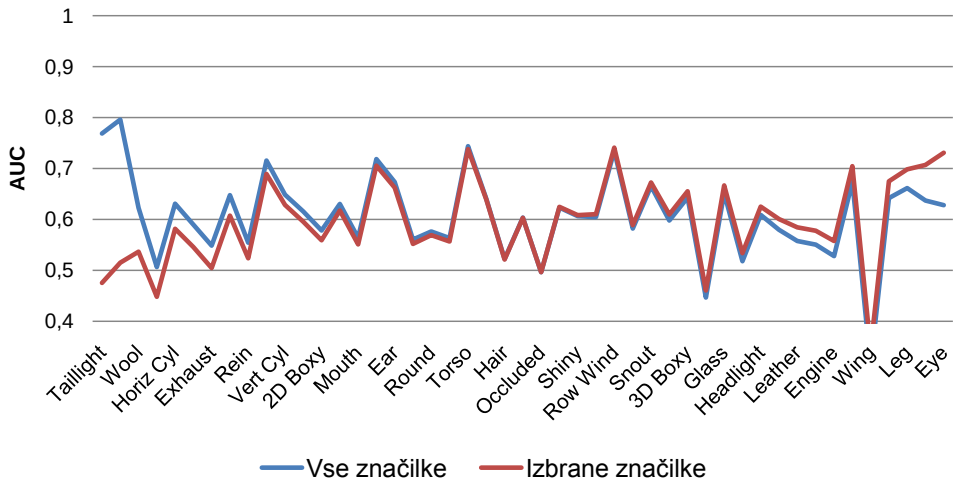
Slika 4.7: Primerjava klasifikatorjev semantičnih atributov pri testni množici z različnimi kategorijami po odpravi pristranskosti razredov. Primerjava poteka med modeli naučenih z vsemi značilkami ter primeri naučenih na izbranih značilkah. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

kot tudi iz slike (4.7) je mogoče opaziti, da se razlika med vsemi značilkami in izbranimi značilkami pri odpravi pristranskosti močno zmanjša, razen pri nekaterih atributih, kot so »perje«, »volna«, »letalsko krilo«, »kolo«, itd.

Z ločeno lokalizacijo se tako kot pri testiranju z obstoječimi kategorijami tudi v tem primeru uspešnost izboljša, vendar le za minimalno. Povprečen AUC je tako pri učenju z vsemi značilkami 0,61 ter 0,60 pri učenju le z izbranimi značilkami. Ti rezultati so bili dobljeni skupaj z odpravo pristranskosti ter L1-regulariziranim L2-izgubnim algoritmom SVC. Razlika med vsemi značilkami in izbranimi značilkami po odpravljeni pristranskosti ter ločeni lokalizaciji je prikazana na sliki (4.8), kjer je možno opaziti tudi atribute, za katere so izboljšave celo slabše (npr. »letalsko krilo«).

4.3.3 Rezultati učenja objektov

Vsi testi v tem sklopu uporabljajo množico slik aPascal. Kjer se uporabljajo tudi semantični in diskriminativni atributi, so le-ti naučeni na učni množici aPascal, in prav tako se učenje kategorij izvaja na isti učni množici. Testiranje kategorij nato poteka na testni množici aPascal. Rezultati teh testov so podani v skupni klasifikacijski točnosti ter v povprečni klasifikacijski točnosti. Slednja



Slika 4.8: Primerjava klasifikatorjev semantičnih atributov pri testni množici z različnimi kategorijami po odpravi pristranskosti razredov ter z ločeno lokalizacijo. Efektivno so se semantični atributi učili na 1393-dimenzionalni značilki. Primerjava poteka med modeli naučenih z vsemi značilkami ter primeri naučenih na izbranih značilkah. Primeri so razvrščeni glede na razliko med obema. Naključna vrednost je 0,5.

je v večini primerov veliko bolj merodajna, saj je lahko zaradi različnega števila primerov v posameznih kategorijah skupna klasifikacijska točnost zavajajoča.

Prva dva opravljena testa pri učenju kategorij predstavljata osnovo za ugotavljanje uspešnosti atributnega učenja. Prvi test ne uporablja ne semantičnih ne diskriminativnih atributov, ampak le bazne značilke, ki predstavljajo učne atribute za učenje kategorij, za učni algoritem pa se uporablja OpenCV implementacijo algoritma SVM (brez vseh izboljšav). Rezultati tega testa so podobni kot v [23], in sicer ima učenje na značilkah iz članka skupno klasifikacijsko točnost 0,60 ter povprečno klasifikacijsko točnost 0,40. Skupna klasifikacijska točnost učenja na naši implementaciji značilk pa je 0,59 ter povprečna točnost 0,39.

Z drugim osnovnim testom pa je mogoče ugotoviti teoretični limit za učenje kategorij z uporabo atributov. Pri teh testih se ne uporablja klasifikatorjev semantičnih atributov, ampak se za učne atribute uporabi *'ground truth'* podatke semantičnih atributov. Ti podatki so uporabljeni tako pri učenju kot tudi pri testiranju kategorij. S tem testom je mogoče preveriti, kako dobro bi delovalo učenje z atributi, če bi vse semantične atribute napovedali s 100% točnostjo. Za teste se je uporabljalo različne algoritme strojnega učenja ter v enem primeru tudi odpravo pristranskosti. Ostale izboljšave niso bile uporabljene.

Klasifikator objektov		Učenje in testiranje na *'ground truth'* podatkih	
		Semantični in diskriminativni atributi	Semantični atributi
OpenCV SVM	pristranski razredi	56,67 (30,06)	87,00 (76,57)
OpenCV SVM	nepristranski razredi	55,18 (28,12)	78,91 (73,29)
$SVM^{multiclass}$	pristranski razredi	44,40 (11,88)	49,94 (18,44)
Log. Regresija		56,58 (29,05)	82,32 (75,78)

Tabela 4.3: Rezultati učenj in testiranj kategorij na bazi aPascal. Informacije o semantičnih atributih za učenje in za testiranje so vzete iz '*ground truth*' oznak slik. Klasifikatorji semantičnih atributov v tem primeru niso bili uporabljeni. Podatki prestavljajo skupno klasifikacijsko točnost ter bolj merodajno povprečno klasifikacijsko točnost, ki je podana v oklepaju.

Iz tabele (4.3) je mogoče opaziti, da v večini primerov uporaba semantičnih atributov prinese več kot 30-odstotno izboljšavo v primerjavi z učenjem samo na baznih značilkah. Vendar pa uporaba dodatnih diskriminativnih atributov rezultat celo poslabša, kar je predvsem nenavadno. V nasprotju s semantičnimi atributi, kjer dobimo povprečno klasifikacijsko točnost 0,76, se ob uporabi semantičnih in diskriminativnih atributov povprečna točnost zmanjša na 0,30. Možen razlog, ki pojasnjuje tako nenavaden rezultat, je, da se diskriminativni atributi preveč prilagajajo učnim podatkom, in zato algoritem strojnega učenja upošteva le diskriminativne atribute, medtem ko semantične atribute povsem izpusti. Kljub temu nam podatki iz te tabele nakazujejo, da lahko učenje z atributi prinesejo poleg novih zmogljivosti tudi še boljšo klasifikacijo. Iz tabele je mogoče opaziti še, da se bo po vsej verjetnosti najbolje obnesla implementacija OpenCV algoritma SVM s pristranskimi razredi.

Rezultati glavnih testov so predstavljeni v tabeli (4.4), kjer je mogoče primerjati različne algoritme strojnega učenja, uporabo izbranih značilk ter uporabo diskriminativnih značilk. Nekateri izmed testov uporabljajo tudi odpravo pristranskosti, medtem ko ostale izboljšave niso bile uporabljene.

Podatki iz tabele (4.4) ne razkrijejo takoj najboljšega algoritma, saj so nekateri algoritmi boljši pri le izbranih značilkah, drugi so boljši pri vseh značilkah, tretji pa so boljši pri uporabi diskriminativnih atributov.

Kljub temu je mogoče hitro ugotoviti, da se $SVM^{multiclass}$ izkaže za najslabšega. Čeprav je povprečna klasifikacijska točnost med 0,42 in 0,44 relativno dobra, podroben pregled podatkov razkrije dejstvo, da je večino primerov klasificiral kot »oseba«, in ta razred ima skoraj 10-krat več primerov kot ostali, zato je veliko pravilnih klasifikacij. Šele podatek o povprečni točnosti razkrije slabost tega klasifikatorja, kjer se vrednost ne povzpe na več kot 0,11. Tudi

Klasifikator objektov	*Klasifikator atributov*	Vse značilke		Izbrane značilke	
		Semantični in diskriminativni atributi	Semantični atributi	Semantični in diskriminativni atributi	Semantični atributi
OpenCV SVM (pristranski razredi)	OpenCV— SVM (pristranski razredi)	54,00 (26,98)	45,71 (25,07)	53,34 (25,70)	30,41 (15,08)
	SVM^{perf} (pristranski razredi)	53,83 (26,76)	51,15 (20,36)	52,26 (24,54)	32,69 (5,11)
	L1-reg. L1-izg. SVC (nepristranski razredi)	53,73 (26.63)	38,53 (29,27)	53.65 (26,49)	11.37 (17,90)
	L1-reg. L1-izg. SVC (pristranski razredi)	53,54 (26,46)	45,75 (25,56)	53,53 (26,02)	43,17 (20,97)
OpenCV SVM (nepristranski razredi)	OpenCV SVM (pristranski razredi)	\	45,19 (24,17)	\	\
	L1-reg. L1-izg. SVC (pristranski razredi)	\	41,44 (28,01)	\	\
$SVM^{multiclass}$ (pristranski razredi)	OpenCV SVM (pristranski razredi)	42,61 (10,78)	43,83 (8,15)	42,70 (10,49)	41,93 (6,06)
Logistična regresija (pristranski razredi)	OpenCV SVM (pristranski razredi)	53,28 (25,39)	50,55 (24,97)	51,95 (22,83)	42,15 (14,30)

Tabela 4.4: Primerjava različnih algoritmov strojnega učenja za učenje kategorij ter učenje atributov pri kombinacijah uporabe vseh značilk in le izbranih ter semantičnih in diskriminativnih atributih. V primerjavo je vključena tudi odprava pristranskosti pri nekaterih algoritmih. Podatki predstavljajo skupno klasifikacijsko točnost, ter bolj merodajno povprečno klasifikacijsko točnost, ki je podana v oklepaju. Obe vrednosti sta podani v odstotkih.

OpenCV SVM z odpravljeno pristranskostjo (tj. nepristranski razredi) ni nič boljši kot brez odpravljene pristranskosti. Če gledamo skupno klasifikacijsko točnost, se je relativno dobro obnesla tudi logistična regresija z okoli 0,50 in 0,42, vendar je povprečna točnost slabša od nekaterih drugih primerov (SVC in OpenCV SVM). Zanimiv je tudi podatek, da se klasifikacijska točnost pri uporabi diskriminativnih atributov ne spreminja veliko, kar je konsistentno s podatki iz tabele (4.3) in kar ponovno kaže na dejstvo, da strojno učenje ignorira semantične atribute ter uporabi le diskriminativne atribute.

4.3.3.1 Učenje na napovedanih semantičnih atributih

Pri učenju kategorij na napovedanih semantičnih atributih so bili testi narejeni le z algoritmi, ki so se predhodno izkazali za boljše. Uporabljena sta bila

predvsem OpenCV SVM in L1-regulariziran L1-izguben SVC. Rezultati teh testov so podani v tabeli (4.5).

Klasifikator objektov	*Klasifikator atributov*	Vse značilke		Izbrane značilke	
		Semantični in diskriminativni atributi	Semantični atributi	Semantični in diskriminativni atributi	Semantični atributi
OpenCV SVM (pristranski razredi)	OpenCV SVM (pristranski razredi)	53,83 (26,96)	49,56 (27,20)	53,42 (26,32)	47,20 (20,44)
	L1-reg. L1-izg. SVC (nepristranski razredi)	**54,57 (28,08)**	**51,62 (30,64)**	**53,91 (27,04)**	**49,40 (26,33)**
OpenCV SVM (nepristranski razredi)	OpenCV SVM (pristranski razredi)	53,83 (26,96)	47,38 (26,01)	53,42 (26,32)	43,77 (17,21)
	L1-reg. L1-izg. SVC (nepristranski razredi)	54,57 (28,08)	49,36 (28,43)	53,91 (27,04)	45,09 (21,51)

Tabela 4.5: Primerjava različnih algoritmov pri učenju z napovedanimi semantičnimi atributi. V primerjavo je vključena tudi odprava pristranskosti pri nekaterih algoritmih. Podatki prestavljajo skupno klasifikacijsko točnost, ter bolj merodajno povprečno klasifikacijsko točnost, ki je podana v oklepaju. Obe vrednosti sta podani v odstotkih.

V primerjavi s tabelo (4.4) je pri uporabi napovedi atributov za učenje kategorij mogoče opaziti izrazito izboljšanje, predvsem pri izbranih značilkah. Pred tem so se vrednosti povprečne klasifikacijske točnosti gibale pod 0,20, medtem ko se jih sedaj večina nahaja nad 0,20. V tem primeru se je tako pri uporabi vseh značilk kot tudi pri izbranih značilkah za najboljšo izkazala kombinacija OpenCV SVM brez odprave pristranskosti za klasifikator objektov ter L1-regulariziran L1-izguben SVC z odpravo pristranskosti za klasifikator atributov. Prav tako kot v vseh predhodnih rezultatih je tudi v tem primeru mogoče opaziti nenavaden vzorec pri uporabi diskriminativnih atributov, saj se splošna klasifikacijska točnost v vseh primerih giblje okoli 0,54.

4.3.3.2 Odprava pristranskosti

Rezultati odprave pristranskosti so predstavljeni že v tabelah (4.3), (4.4) in (4.5), zato se jih v tem podpoglavju ne izpostavlja posebno. Omeniti velja le, da je odprava pristranskosti pozitivno vplivala na klasifikacijo atributov z algoritmom SVC, medtem ko pa je negativno vplivala na klasifikacijo objektov z algoritmom OpenCV SVM.

4.3.3.3 Uporaba ločene lokalizacije

Pri ločeni lokalizaciji se za algoritme strojnega učenja uporabi le kombinacija, ki se je predhodno izkazala za najboljšo; OpenCV SVM za klasifikator objektov ter L1-regulariziran L1-izguben SVC za klasifikacijo atributov. Za vse teste v tem sklopu se uči kategorije na napovedanih semantičnih atributih. Odprava pristranskosti se uporabi le pri klasifikatorju atributov. Rezultati teh testov so podani v tabeli (4.6).

Klasifikator objektov	*Klasifikator atributov*	Vse značilke		Izbrane značilke	
		Semantični in diskriminativni atributi	Semantični atributi	Semantični in diskriminativni atributi	Semantični atributi
OpenCV SVM (pristranski razredi)	L1-reg. L1-izg. SVC (nepristranski razredi)	52,85 (25,06)	48,87 (29,86)	51,86 (23,63)	47,49 (29,24)

Tabela 4.6: Rezultati učenja kategorij na napovedanih atributih ter z ločeno lokalizacijo. Podatki prestavljajo skupno klasifikacijsko točnost, ter bolj merodajno povprečno klasifikacijsko točnost, ki je podana v oklepaju. Obe vrednosti sta podani v odstotkih.

V primerjavi z rezultati iz tabele (4.5) se klasifikacijska točnost sicer malce poslabša, vendar pa se pri uporabi izbranih značilk in le semantičnih atributov povprečna klasifikacijska točnost dejansko izboljša za 0,03 točke.

4.3.3.4 Učenje in kategorizacija novih kategorij

Pri učenju novih kategorij uporabimo množico slik, ki niso bile zajete pri učenju atributov, zato za učenje atributov uporabimo učno množico aPascal, za učenje in testiranje kategorij pa množico aYahoo. Za učenje in testiranje se naključno izbere 30 različnih primerov iz vsake kategorije. Vsi testi so zaradi naključnosti izvedeni 3-krat ter nato poročamo povprečje teh testov. Testi v tem sklopu niso uporabljali diskriminativnih atributov, saj so le-ti po naravi prilagojeni na stare kategorije. Za algoritme strojnega učenja so bili uporabljeni algoritmi, ki so se predhodno izkazali za uspešne, torej OpenCV SVM za učenje kategorij (brez odprave pristranskosti) ter L1-reg. L2-izgubni SVC (z odpravo pristranskosti) za učenje atributov.

Pri prvem testu se za učenje in testiranje kategorij uporablja *'ground truth'* podatke in tako predstavlja okvir ter teoretično najboljši rezultat, ki ga je možno dobiti z uporabo semantičnih atributov. V tem primeru dobimo splošno klasifikacijsko točnost 0,75 ter povprečno točnost 0,76.

Nadaljnji testi so narejeni sprva brez vseh izboljšav, nato z učenjem na napovedanih semantičnih atributih, nazadnje pa še skupaj z ločeno lokalizacijo. Rezultati vseh testov so prikazani v tabeli (4.7).

Izboljšave	Vse značilke	Izbrane značilke
	Semantični atributi	
Brez izboljšav*	20,89 (20,64)	19,39 (18,98)
Učenje na napovedanih semantičnih atributih	30,73 (30,46)	27,16 (26,83)
Učenje na napovedanih semantičnih atributih in ločena lokalizacija	44,28 (43,75)	40,87 (40,31)

Tabela 4.7: Rezultati učenja kategorij na različnih izboljšavah za vse značilke in le izbrane značilke na bazi aYahoo. Podatki prestavljajo skupno klasifikacijsko točnost, ter bolj merodajno povprečno klasifikacijsko točnost, ki je podana v oklepaju. *Primer brez izboljšave dejansko vsebuje odpravo pristranskosti vendar le pri klasifikatorju atributov, medtem ko pri klasifikatorju kategorij odprava pristranskosti ni bila uporabljena. Obe vrednosti sta podani v odstotkih.

Iz zgornjih podatkov je takoj mogoče opaziti, da vsaka izboljšava močno izboljša klasifikacijsko točnost. Če primerjamo teste brez izboljšav, vidimo, da se s povprečne klasifikacijske točnosti okoli 0,20 vrednost pri vseh izboljšavah izboljša za skoraj 100 % na okoli 0,40. Toda če primerjamo vse značilke proti izbranim značilkam, lahko ugotovimo, da so izbrane značilke malenkost slabše — za okoli 0,03 točke.

4.3.4 Povzetek rezultatov

Primerjava zgornjih rezultatov s teoretičnim limitom, predstavljenim v tabeli (4.3), lahko pokaže, da je še vedno veliko prostora za izboljšave. Direktna uporaba značilk se s povprečno klasifikacijsko točnostjo 0,39 sicer izkaže za malenkost boljšo kot učenje z atributi pri 0,30, vendar sta še vedno daleč pod teoretičnim limitom 0,76. Iz teh podatkov lahko sklepamo, da bi lahko z boljšo klasifikacijo semantičnih atributov dosegli veliko boljše rezultate kot samo z učenjem na značilkah.

Izbiranje pomembnih značilk za učenje semantičnih atributov se sprva ne izkaže za zelo uporabno. Iz slike (4.3) ter tabel (4.4) in (4.5) ni mogoče opaziti večjih izboljšav pri izbiri značilk, dejansko se uspešnost celo poslabša. Razlog za to je, da so bile pri učenju in testiranju uporabljene iste kategorije, pri čemer pa je izbiranje značilk relativno nepomembno. Šele pri dodatnih izboljšavah se razlika med izbranimi in vsemi značilkami močno zmanjša, vendar so v

povprečju vse značilke še vedno boljše. Izbiranje značilk se izkaže za uporabno šele pri učenju novih kategorij. Tako je v slikah (4.6), (4.7) in (4.8) takoj možno opaziti manjšo razliko, v nekaterih primerih pa je izbiranje značilk celo boljše. Pri novih kategorijah je izbiranje bolj uspešno zato, ker pride do večjega problema generalizacije, izbiranje značilk pa odpravlja ravno probleme pri generalizaciji.

Tudi vse dodatne izboljšave so se izkazale za uporabne. V skoraj vseh primerih se je rezultat izboljšal, razen v primeru ločene lokalizacije, kjer je iz tabele (4.6) mogoče opaziti, da se povprečna klasifikacijska točnost vseh značilk malenkost poslabša. Vendar se zato povprečna točnost pri izbranih značilkah izboljša za skoraj 0,03 točke. Najbolj se opazi izboljšava pri učenju novih kategorij. Iz tabele (4.7) je mogoče videti, da se pri uporabi vseh izboljšav rezultat izboljša za skoraj 100 %.

Če primerjamo vse zgornje rezultate z rezultati iz [23], lahko opazimo močno povezavo. Pri skoraj vseh testih dobimo podobne rezultate, razlike pa so v večini primerov le za 0,05 do 0,10 točke. Te razlike gre pripisati različni implementaciji samega izvlečeka baznih značilk, uporabi različne implementacije algoritmov strojnega učenja ter možnim manjšim spremembam v implementaciji učenja in klasifikacije. Pri uporabi diskriminativnih atributov sicer ne opazimo takšnih izboljšav kot v članku, vendar je to mogoče pripisati problemu prevelikega prilagajanja učnim podatkom v naši implementaciji. Če odmislimo diskriminativne atribute, se najboljši primer kategorizacije z vsemi značilkami iz članka razlikuje le za 0,04 točke, medtem ko se najboljši primer kategorizacije z izbranimi značilkami razlikuje le za 0,02 točke.

4.4 Nadgradnja atributnega učenja z modelom LHOP

4.4.1 Vrste testov

V tem podpoglavju preverjamo, kako dobro se obnese integracija značilke LHOP v učenje z atributi. Pri tem zato poročamo rezultate o uporabi nove bazne značilke, kjer smo za značilko LHOP preverili tako uporabo 3. nivoja delov, kot tudi 3. in 2. nivoja delov skupaj. Obenem za ustrezno primerjavo poročamo tudi rezultate o uporabi originalne značilke.

Vrste testov lahko razdelimo v isti dve skupini kot v prejšnjem poglavju. V prvi so vsi testi, ki se nanašajo na učenje semantičnih atributov, v drugi skupini pa so testi iz učenja kategorij.

1. Učenje in klasifikacija semantičnih atributov

Testi v tej skupini potekajo popolnoma isto kot v prejšnjem podpoglavju, kjer smo semantične atribute naučili na učni množici aPascal ter naredili ustrezen test bodisi na testni množici aPascal (ang. *within category*) bodisi na bazi aYahoo (ang. *across category*). Prav tako testiramo, kako dobro se obnesejo izbrane značilke proti vsem značilkam (ang. *feature selection*).

2. Učenje in klasifikacija kategorij

Pri učenju in klasifikaciji kategorij lahko teste v grobem delimo podobno kot v testih prejšnjega poglavja, le da v tem primeru ni treba testirati idealne situacije, saj bi bili rezultati popolnoma enaki. Testiranje tako poteka pri uporabi le baznih značilk (učenje brez atributov), uporabi semantičnih atributov ter uporabi semantičnih in novih diskriminativnih atributov.

Za ustrezno primerjavo novih in starih značilk teste druge skupine naredimo na množici aPascal. Na tej bazi so predstavljeni primeri, kjer uporabljamo iste kategorije za učenje semantičnih atributov ter učenje kategorij. Za primerjavo klasifikacije novih kategorij med starimi in novimi značilkami,pa teste naredimo na bazi Caltech.

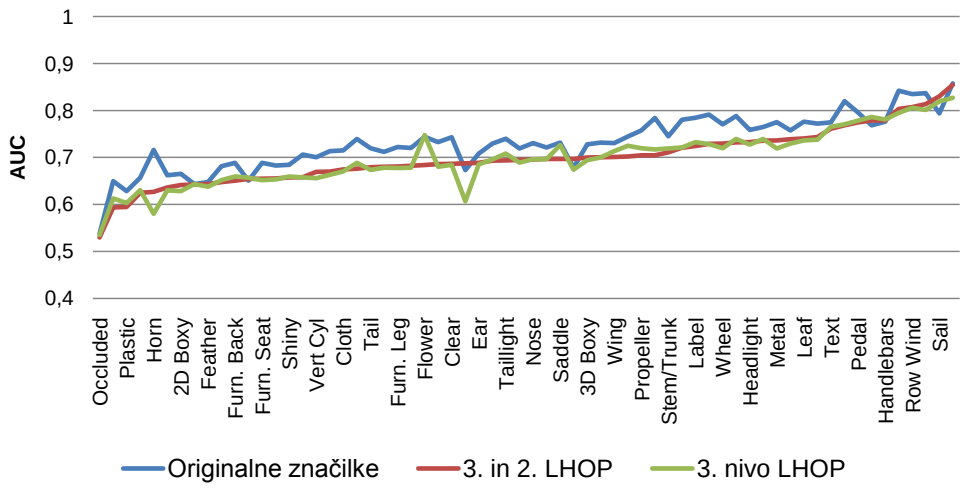
Pri vseh testih uporabljamo tudi vse izboljšave, vključno s krožno lokalizacijo. Zaradi slednje je treba teste z originalnimi značilkami ponoviti, tako da bo primerjava med novimi značilkami povsem pravilna. Te teste ponovimo tako pri prvi skupini testov, kjer se uči in klasificira semantične atribute, kot tudi pri drugi skupini, kjer poteka kategorizacija. Druge uporabljene izboljšave so tudi učenje kategorij na napovedanih atributih, uporaba ločene lokalizacije, odprava pristranskosti, kjer se je to izkazalo za uspešno, ter tudi novi izbojšani diskriminativni atributi.

Uporabljene baze slik: Bazi aPascal in aYahoo uporabljamo predvsem za ustrezno primerjavo z rezultati iz prejšnjega poglavja, medtem ko je večina testov v tem poglavju usmerjena v uporabo množice slik iz baze Caltech, ki pa jo zaradi pomankanja oznak semantičnih atributov lahko uporabimo le za učenje kategorij, ne pa tudi pri učenju semantičnih atributov. Pri tem je seveda treba semantične atribute naučiti na drugi bazi, zato za učenje semantičnih atributov uporabljamo učno množico aPascal.

4.4.2 Rezultati učenja semantičnih atributov

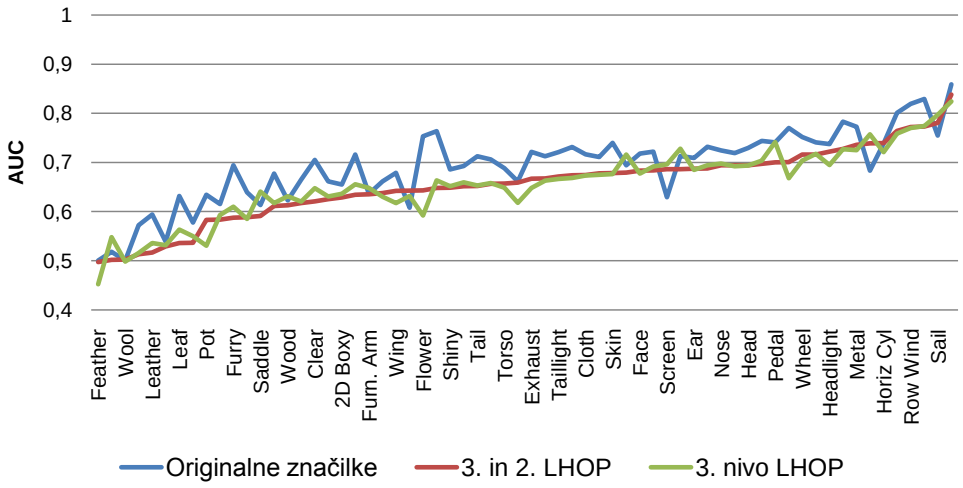
4.4.2.1 Z aPascal za učno in testno množico

Semantične atribute se je mogoče z originalnimi značilkami in krožnim vzorčenjem naučiti s povprečnim AUC 0,73 pri uporabi vseh značilk in 0,69 pri uporabi le izbranih značilk. V primerjavi z novimi značilkami LHOP se slednje obnesejo slabše, in sicer z uporabo vseh značilk dobimo povprečen AUC 0,70 ter 0,65 pri le izbranih značilkah. Ta rezultat je podan za značilko LHOP s 3. nivoja, medtem ko se rezultat pri uporabi 3. in 2. nivoja skorajda ne razlikuje od značilke s 3. nivoja. Podrobne razlike pri posameznih atributih je mogoče videti na sliki (4.9) in (4.10).



Slika 4.9: Primerjava originalnih značilk, značilk LHOP s 3. nivoja ter značilk LHOP s 3. in 2. nivoja skupaj, kjer so bili vsi semantični atributi naučeni na vseh značilkah. Atributi so bili naučeni na učni množici aPascal ter testirani na testni množici aPascal. Vrednosti predstavljajo AUC ter so razporejeni glede na naraščajočo vrednost značilk LHOP s 3. in 2. nivoja. Naključna verjetnost je 0,5.

Oba grafa prikazujeta posamezne vrednosti AUC za originalne značilke, značilke LHOP 3. nivoja ter značilke LHOP 3. in 2. nivoja skupaj. Prvi prikazuje atribute naučene, na vseh značilkah, medtem ko drugi prikazuje semantične atribute, naučene le na izbranih značilkah. Iz obeh je mogoče opaziti, da so pri večini atributov boljše originalne značilke, vendar kljub temu obstajajo tudi primeri, kot so »okroglo«, »zaslon« in »jadro«, kjer se značilka LHOP obnese malenkost bolje. Te razlike so sicer veliko bolj izrazite pri izbranih



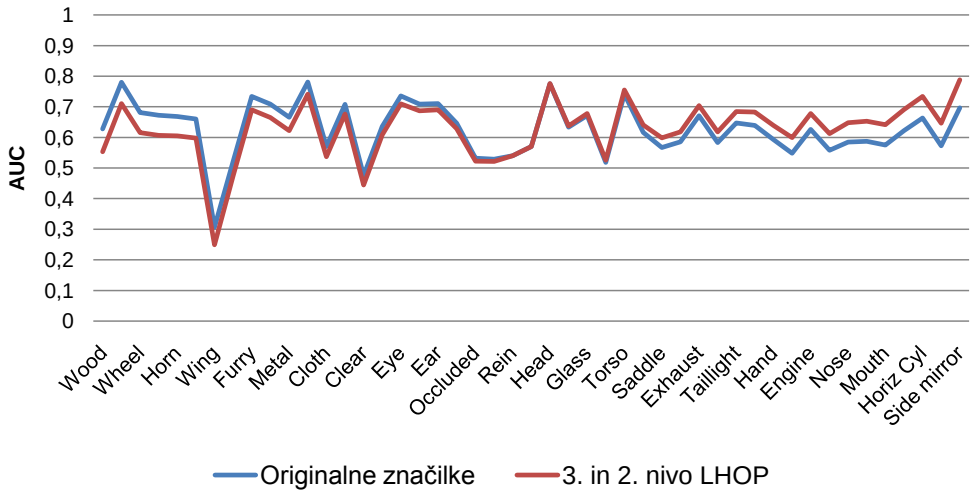
Slika 4.10: Primerjava originalnih značilk, značilk LHOP s 3. nivoja ter značilk LHOP s 3. in 2. nivoja skupaj, kjer so bili vsi semantični atributi naučeni le na *izbranih značilkah*. Atributi so bili naučeni na učni množici aPascal ter testirani na testni množici aPascal. Vrednosti predstavljajo AUC ter so razporejeni glede na naraščajočo vrednost značilk LHOP s 3. in 2. nivoja. Naključna verjetnost je 0,5.

značilkah, vendar obstajajo tudi pri vseh značilkah. Glede na primere, kjer je značilka LHOP boljša, bi lahko sklepali, da se slednja bolje obnese pri semantičnih atributih, kjer je enostavna oblika zelo izrazita. Tako »okroglo« kot tudi »zaslon« in »jadro« imajo zelo enostavno, vendar izrazito obliko.

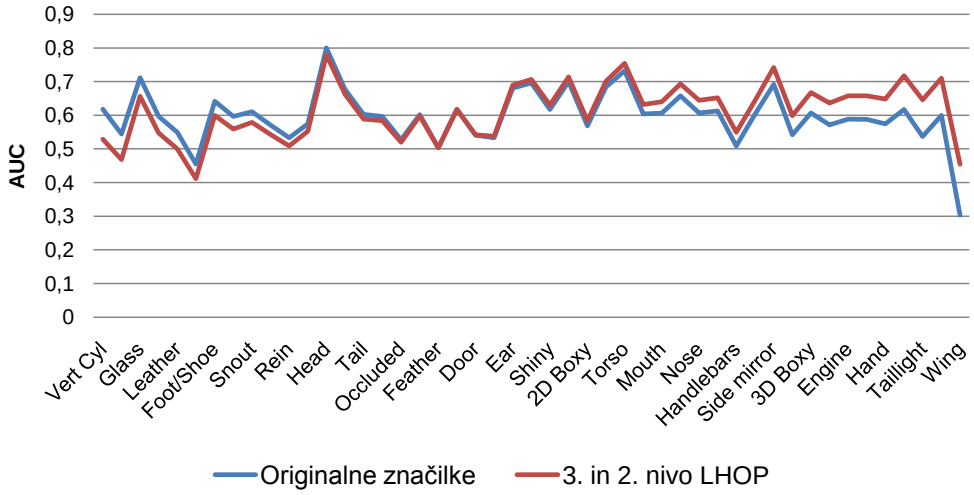
Razlika med uporabo le 3. nivoja in uporabo 3. in 2. nivoja skupaj je v povprečju povsem neopazna, medtem ko oba grafa razkrijeta, da so vrednosti sicer močno skupaj, ampak še vedno obstaja nekaj primerov, kjer se eden ali drugi izkaže za boljšega. Iz slike (4.10) je mogoče opaziti, da se le 3. nivo pri izbranih značilkah izkaže boljši pri atributih, kot so »sedlo«, »les«, »pedal« itd.

4.4.2.2 Z aPascal za učno množico in aYahoo za testno množico

Pri testiranju semantičnih atributov na bazi aYahoo so se značilke LHOP izkazale za malenkost boljše in je razlika med nekaterimi posameznimi primeri dobro opazna. Povprečen AUC pri originalnih značilkah in pri značilkah LHOP je 0,63 pri uporabi vseh značilk, medtem ko je AUC pri uporabi le izbranih značilk 0,60 za originalne značilke ter 0,61 za značilke LHOP. Pri značilkah



Slika 4.11: Primerjava originalnih značilk, značilk LHOP s 3. in 2. nivoja skupaj, kjer so bili vsi semantični atributi naučeni na vseh značilkah. Atributi so bili naučeni na učni množici aPascal ter testirani na testni množici aYahoo. Vrednosti predstavljajo AUC ter so razporejene glede na razliko med obema. Naključna vrednost je 0,5.

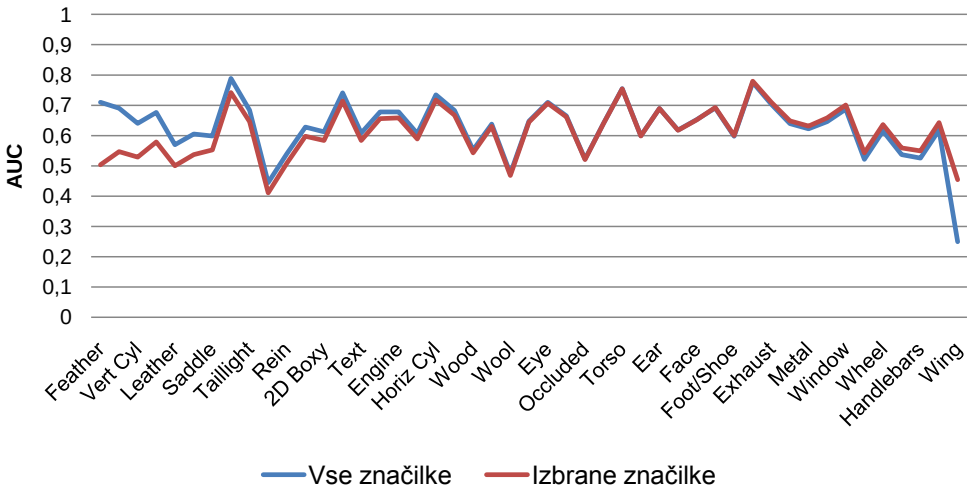


Slika 4.12: Primerjava originalnih značilk ter značilk LHOP s 3. in 2. nivoja skupaj, kjer so bili vsi semantični atributi naučeni na *izbranih značilkah*. Atributi so bili naučeni na učni množici aPascal ter testirani na testni množici aYahoo. Vrednosti predstavljajo AUC ter so razporejene glede na razliko med obema. Naključna vrednost je 0,5.

LHOP se rezultat med uporabo 3. nivoja ter uporabo 3. in 2. nivoja razlikuje za manj kot 0,01 točke. Podrobni rezultati posameznih atributov so podani na sliki (4.11) za učenje na vseh značilkah ter na sliki (4.12) za učenje na le izbranih značilkah.

Iz slike (4.11) je mogoče opaziti, da se originalne značilke pri učenju na vseh značilkah bolje obnesejo pri atributih, kot so »les«, »kolo«, »rep«, »kovinsko« itd., medtem ko se značilke LHOP obnesejo bolje pri atributih »stransko ogledalo«, »usta«, »nos«, »motor vozila«, »roka« itd.

Podobno je tudi na sliki (4.12), kjer je možno opaziti razliko med originalnimi značilkami in značilkami LHOP pri učenju na izbranih značilkah. Zanimivo je, da se v tem primeru nahaja ravno atribut »letalsko krilo« na drugi strani seznama skupaj z »roka«, »zadnja motorna luč«, »motor vozila« in »stransko ogledalo«, kjer so atributi, ki so se izkazali za boljše pri značilki LHOP. Atributi, ki se izkažejo za slabše pri značilki LHOP, pa so predvsem »steklo«, »usnje«, »čevelj/noga« itd. Iz vseh primerov je mogoče opaziti, da se značilke LHOP bolje obnesejo pri atributih, ki imajo zelo izrazito obliko, kot na primer »usta«, »zadnja luč«, »motor vozila«, medtem ko se pri atributih, ki nimajo izrazite oblike, kot je »les«, »steklo«, »kovinsko«, obnesejo slabše. Navkljub nekaj izjemam (»kolo«, »rep«, »čevelj/noga«) so ti rezultati povsem



Slika 4.13: Primerjava med vsemi značilkami in izbranimi značilkami pri uporabi značilke LHOP s 3. in 2. nivoja. Atributi so bili naučeni na učni množici aPascal ter testirani na testni množici aYahoo. Vrednosti predstavljajo *AUC* ter so razporejene glede na razliko med obema. Naključna vrednost je 0,5.

konsinstentni s pričakovanji, saj je značilka LHOP veliko primernejša za bolj specifično obliko.

Zanimivi so tudi podatki iz slike (4.13), kjer je mogoče primerjati značilke LHOP (3. in 2. nivoja skupaj) pri uporabi vseh značilk proti uporabi le izbranih značilk. Izbiranje značilk močno pripomore pri atributu »letalsko krilo«, medtem ko pa se pri atributih »perje« in »dlaka« močno poslabša. Iz tega bi lahko sklepali, da se izbiranje značilk še bolje obnese v primerih, kjer je oblika bolj izrazita, medtem ko v primerih, kjer je izrazita tekstura in barva, izbiranje ni tako dobro.

4.4.3 Rezultati učenja objektov

4.4.3.1 Učenje in kategorizacija na množici aPascal

Rezultati v tem podpoglavju predstavljajo teste kategorizacije iz množice slik aPascal. Kategorije so bile naučene na učni množici aPascal ter nato testirane na testni množici aPascal. Pri tem so bili tudi semantičnimi atributi naučeni na učni množici aPascal. Primerjava poteka med tremi različnimi značilkami; originalnimi značilkami, značilkami LHOP s 3. nivoja ter značilkami LHOP iz 3. in 2. nivoja skupaj. Rezultati so podani v tabeli (4.8), kjer za vsako vrsto značilk testiramo učenje brez atributov s celimi baznimi značilkami ter učenje z atributi pri uporabi vseh značilk ter uporabi le izbranih značilk.

Vrsta značilk	Bazne značilke (brez atributov)	Atributno učenje	
		Vse značilke	Izbrane značilke
Originalne značilke	62,06 (41,02)	52,32 (33,07)	41,02 (23,18)
3. & 2. nivo LHOP	56,90 (36,74)	47,23 (27,02)	45,90 (27,35)
3. nivo LHOP	57,40 (36,74)	46,73 (26,38)	45,27 (25,83)

Tabela 4.8: Primerjava kategorizacije med originalnimi značilkami ter značilkami LHOP na bazi aPascal. Testira se bazne značilke (brez atributov) ter učenje z atributi bodisi z vsemi značilkami bodisi le z izbranimi. Rezultati so podani v skupni klasifikacijski točnosti ter v povprečni klasifikacijski točnosti, ki je bolj merodajna in je podana v oklepajih. Obe vrednosti sta podani v odstotkih.

Iz tabele (4.8) je mogoče opaziti, da se je najbolje obnesla originalna značilka z učenjem na celotni bazni značilki (brez atributov), medtem ko se učenje z atributi obnese za skoraj 0,05 do 0,10 točk slabše. Če naredimo primerjavo med značilko LHOP s 3. nivoja ter s 3. in 2. nivojem skupaj,

lahko sicer opazimo, da se 3. nivo ob uporabi baznih značilk obnese malenkost bolje, medtem ko se 3. in 2. nivo skupaj obneseta bolje pri atributnem učenju. Razlika med obema je skoraj zanemarljiva. Omeniti velja tudi, da se značilke LHOP obnesejo bolje od originalnih značilk pri atributnem učenju z izbranimi značilkami, medtem ko se slednje izkažejo za boljše pri učenju atributov na vseh značilkah.

Vsi zgornji testi za lokalizacijo uporabljajo krožno vzorčenje, zato lahko rezultate z originalnimi značilkami primerjamo z rezultati iz prejšnjega poglavja, kjer smo uporabljali pravokotno vzorčenje. Pri pravokotnem vzorčenju dobimo skupno klasifikacijsko točnost za učenje brez atributov 0,60 ter povprečno točnost 0,40, medtem ko je pri krožnem vzorčenju rezultat boljši za kakšen odstotek ali dva. Pri učenju z atributi se tudi izboljša za nekaj odstotkov vendar le pri uporabi vseh značilk, medtem ko se pri uporabi izbranih značilk obe klasifikacijski točnosti dejansko poslabšata za med 0,05 do 0,10 točk.

4.4.3.2 Učenje in kategorizacija na množici Caltech 101

Za testiranje klasifikacije novih kategorij (tj. klasifikacija kategorij, ki niso bile uporabljene pri učenju semantičnih atributov) se uporablja množico slik iz baze Caltech. Primerjava poteka podobno kot pri bazi aPascal med originalnimi značilkami ter dvema vrstama LHOP značilk. Podobno tudi testiramo za vsako vrsto značilk učenje brez atributov ter učenje z atributi; bodisi z vsemi značilkami bodisi le z izbranimi. Pri tem se testira tudi nove diskriminativne atribute, ki so povsem primerni za kategorizacijo novih kategorij. Rezultati vseh teh testiranj so podani v tabeli (4.9).

V nasprotju z testi aPascal se v tem primeru najbolje izkaže značilka LHOP s 3. nivoja brez atributnega učenja, torej z uporabo le bazne značilke. Prednost pred originalno značilko je zelo majhna, in sicer originalna značilka zaostaja za kakšen odstotek, medtem ko značilka LHOP s 3. in 2. nivoja zaostaja za okoli 0,03 do 0,04 točke.

Če primerjamo učenje s celo bazno značilko ter učenje z atributi, lahko opazimo, da je bazna značilka še vedno veliko boljša. Z uporabo novih diskriminativnih atributov lahko to razliko močno zmanjšamo, vendar učenja brez atributov še vedno ne prekosimo. Uporaba novih diskriminativnih atributov izboljša učenje z atributi za skoraj 0,05 do 0,10 točk, ter je najbolj očitna pri originalnih značilkah z učenjem atributov na vseh značilkah. V tem primeru dobimo splošno klasifikacijsko točnost 0,57 ter povprečno točnost 0,56, ter se sedaj od baznih značilk razlikuje le za dobrih 0,05 točk.

Pri učenju z atributi se najbolje izkažejo ravno originalne značilke, ki so od

Vrsta značilk	Bazne značilke (brez atributov)	Vse značilke		Izbrane značilke	
		Semantični in novi diskriminativni atributi	Semantični atributi	Semantični in novi diskriminativni atributi	Semantični atributi
Originalne značilke	63,13 (61,64)	57,49 (55,58)	49,29 (47,93)	53,40 (51,51)	43,88 (42,15)
3. & 2. nivo LHOP	61,26 (59,70)	52,61 (50,54)	45,92 (44,39)	54,13 (52,27)	46,45 (44,73)
3. nivo LHOP	64,58 (62,86)	50,26 (48,49)	45,04 (43,55)	50,38 (48,64)	46,56 (44,87)

Tabela 4.9: Primerjava kategorizacije med originalnimi ter LHOP značilkami na bazi Caltech 101. Testira se bazne značilke brez atributov, ter učenje z atributi. Slednje se testira tako pri učenju atributov na vseh značilkah, kot tudi pri učenju na le izbranih značilkah. Prav tako se testira tudi ločeno samo semantične atribute ter semantične atribute v kombinaciji z novimi diskriminativnimi atributi. Za učne in testne primere se je uporabilo celo sliko iz baze Caltech. Rezultati so podani v skupni klasifikacijski točnosti ter v povprečni klasifikacijski točnosti, ki je bolj merodajna in je podana v oklepajih. Obe vrednosti sta podani v odstotkih. Pri učenju in testiranju je bilo za vsakega uporabljenih naključnih 30 primerov. Zaradi naključnosti so bili testi z uporabo le semantičnih atributov ponovljeni 5-krat in ostali testi 3-krat.

LHOP boljše za okoli 0,05 točk. Če primerjamo obe značilki LHOP lahko ugotovimo, da je pri učenju z atributi boljša značilka s 3. in 2. nivojem, medtem ko je razlika v primerjavi s samo 3. nivojem manj kot odstotek. Zanimivo je tudi opažanje, da se značilka LHOP obnese malenkost bolje pri učenju z atributi, naučenimi na le izbranih značilkah. Značilke LHOP se obnesejo bolje pri obeh vrstah značilk, kadar uporabimo samo semantične atribute, medtem ko je pri dodatni vpeljavi diskriminativnih atributov boljša značilka s 3. in 2. nivojem skupaj.

V nasprotju z bazo aPascal, kjer je učenje in testiranje potekalo na slikah, segmentiranih s pravokotnim oknom, se v zgornjih testih ni uporabilo nobene segmentacije. Za učne in testne primere se je vzelo celotno sliko. Večina primerov v bazi Caltech je takih, da se objekt nahaja relativno na sredini in je okolica relativno majhna, zato uporaba celotne slike za učenje ne predstavlja velikega problema. S tem se tudi izognemo problemu iskanja objekta v sliki.

V naslednjih testih se primerja uspešnost kategorizacije pri učenju in testiranju na različno segmentiranih slikah. Primerjava poteka na značilkah LHOP s 3. in 2. nivoja. Primerja se učenje in testiranje na segmentirani sliki, učenje na segmentirani in testiranje na celotni sliki ter učenje in testiranje na celotni sliki. Testi potekajo na učenju brez atributov (bazna značilka) ter učenju z

diskriminativnimi in semantičnimi atributi, naučenimi bodisi na vseh značilkah bodisi le na izbranih značilkah. Rezultati teh testiranj so zbrani v tabeli (4.10).

Vrsta učne in testne slike	Bazne značilke (brez atributov)	Vse značilke		Izbrane značilke	
		Semantični in novi diskriminativni atributi	Semantični atributi	Semantični in novi diskriminativni atributi	Semantični atributi
Učenje in testiranje na segmentirani sliki	71,08 (69,74)	62,68 (61,22)	52,34 (51,12)	62,74 (61,17)	51,96 (50,63)
Učenje na segmentirani, testiranje na celotni sliki	55,41 (54,49)	48,30 (46,99)	38,96 (37,80)	47,75 (46,63)	39,27 (38,10)
Učenje in testiranje na celotni sliki	61,26 (59,70)	53,45 (51,63)	46,09 (44,31)	52,51 (50,68)	46,34 (44,55)

Tabela 4.10: Primerjava med uporabo različnih kombinacij učenja in testiranja bodisi na celotni sliki bodisi na sliki segmentirani s pravokotnim oknom, na bazi Caltech 101. Vsi testi uporabljajo značilko LHOP s 3. in 2. nivoja skupaj. Testira se uporabo le baznih značilk (brez atributov) ter uporabo semantičnih in diskriminativnih atributov, naučenih bodisi na vseh značilkah bodisi na izbranih značilkah. Rezultati so podani v skupni klasifikacijski točnosti ter v povprečni klasifikacijski točnosti, ki je bolj merodajna in je podana v oklepajih. Obe vrednosti sta podani v odstotkih. Pri učenju in testiranju je bilo za vsakega uporabljenih naključnih 30 primerov. Zaradi naključnosti so bili testi z uporabo le semantičnih atributov ponovljeni 5-krat in ostali testi 3-krat. Rezultati zadnje vrstice v tej tabeli bi se načeloma morali ujemati z rezultati 2. vrstice iz tabele (4.9), vendar lahko zaradi naključnega izbiranja primerov pride do manjšega odstopanja.

Iz tabele (4.10) je mogoče opaziti, da so najboljši rezultati pri učenju in testiranju na segmentirani sliki. V tem primeru velja predpostavka, da je v času učenja in testiranja podana natančna lokacija in velikost objekta na sliki. Vendar v realnih pogojih pri postopku testiranja ta predpostavka ne drži. Posledično to pomeni, da rezultati v zgornji tabeli za učenje in testiranje na segmentirani sliki predstavljajo idealne pogoje, v katerih lahko v času testiranja določimo lokacijo objekta s 100% natančnostjo. Te rezultate lahko uporabimo kot okvirno napoved o izboljšavi klasifikacije v primeru, da bi uporabili dodatne algoritme za iskanje lokacije objekta.

V primeru, da bi za učenje uporabili segmentirano sliko ter celotno sliko za testiranje, lahko opazimo, da se rezultat v vseh primerih poslabša za skoraj 0,15 ali več točk. Veliko boljšo natančnost dobimo, kadar se tudi učimo na celotni sliki. V tem primeru se klasifikacijska točnost zmanjša le za 0,05 do 0,10 točk.

Zanimivo je tudi opažanje, da lahko z uporabo diskriminativnih atributov v primeru učenja in testiranja na celotni sliki dosežemo skoraj isti ali celo boljši

rezultat kot pri uporabi samih semantičnih atributov v idealnem primeru, če bi poznali lokacijo objekta. Prav tako bi lahko ob uporabi diskriminativnih atributov z dodatnimi algoritmi segmentacije dosegli skoraj boljši rezultat od uporabe samih baznih značilk brez dodatne segmentacije.

4.4.4 Povzetek rezultatov

Iz vseh zgornjih rezultatov je možno sklepati, da se v večini primerov originalne značilke še vedno obnesejo bolje od značilk LHOP. To potrjujeta tako sliki (4.9) in (4.10), kjer je možno opaziti, da je AUC pri učenju večine semantičnih atributov slabši pri obeh vrstah značilk LHOP, kot tudi tabeli (4.8) in (4.9), kjer so originalne značilke boljše pri skoraj vseh testih. Še vedno pa obstajajo primeri, kjer se je značilka LHOP izkazala za boljšo. Tako je pri testiranju semantičnih atributov na bazi aYahoo možno videti, da ima značilka LHOP malenkost boljši povprečen AUC. O tem pričata tudi sliki (4.11) in (4.12), kjer je videti, da obstaja kar nekaj semantičnih atributov, pri katerih je značilka LHOP dejansko boljša od originalne značilke. Iz tabel (4.9) in (4.10) je možno ugotoviti tudi, da se značilka LHOP veliko bolje obnese pri uporabi izbranih značilk za učenje semantičnih atributov. Možni razlog za slabšo klasifikacijsko točnost bi lahko iskali v knjižici delov, uporabljeni za izračun značilke LHOP. Ta knjižica je bila naučena na povsem neodvisni množici slik, in iz zgornjih rezultatov je očitno, da sicer lahko dobimo soliden rezultat, vendar še vedno slabšega kot z originalnimi značilkami.

Če naredimo primerjavo med obema značilkama LHOP, ugotovimo, da se v večini primerov ne razlikujeta močno. Iz slik (4.9) in (4.10) je mogoče opaziti, da obstajajo atributi, kjer se eden obnese bolje od drugega, vendar v povprečju ni velike razlike med njima. Tudi iz tabel (4.8) in (4.9) je mogoče videti, da je razlika med njima zelo majhna, vendar lahko v tem primeru opazimo, da se značilka s 3. nivojem obnese malenkost bolje pri uporabi le bazne značilke, medtem ko se značilka s 3. in 2. nivojem skupaj obnese bolje pri učenju z atributi.

Zelo dobro so se obnesli tudi novi diskriminativni atribuiu. Poleg tega, da lahko te atribute sedaj uporabimo tudi pri novih kategorijah, je iz tabel (4.9) in (4.10) mogoče opaziti, da se vsi rezultati pri uporabi diskriminativnih atributov močno izboljšajo. Seveda je to izboljšavo treba po drugi strani tudi plačati, tako da se čas učenja in testiranja dodatno poveča, vendar je še vedno v mejah sprejemljivega.

Zanimivi so tudi rezultati testiranj z bazo Caltech 101. Glede na to, da se pri teh testih uporablja iste semantične atribute kot pri bazi aPascal in

aYahoo ter se ne uporabi nobenih novih atributov, ki bi bili primerni za bazo Caltech, dobimo rezultat, ki se lahko dobro primerja tudi z nekaterimi drugimi metodami. Z dodatnimi semantičnimi atributi bi se lahko po vsej verjetnosti rezultat še dodatno izboljšal. Prav tako rezultati iz tabele (4.10) nakazujejo, da je mogoče rezultate še močno izboljšati z dodatnimi algoritmi za iskanje objekta na sliki.

Poglavje 5

Zaključek

5.1 Sklepi in ugotovitve

V diplomski nalogi smo natančneje opisali metodo učenja vizualnih kategorij z uporabo atributov ter preverili njeno uspešnost. Pri tem smo pokazali, da se naša implementacija obnese podobno dobro kot originalna implementacija, ki so jo naredili Ali Farhadi idr. Pokazali smo, da dobimo podobne rezultate tako za učenje in kalsifikacijo semantičnih atributov kot tudi za učenje in klasifikacijo kategorij. V metodo smo vpeljali tudi izboljšave, kot so odprava pristranskosti klasifikatorjev, učenje na napovedanih atributih, ločena lokalizacija, izboljšani diskriminativni atributi ter krožno sempliranje. Za vse smo tudi pokazali, da se v večini primerov izkažejo za uporabne. Metodo smo tudi testirali na množici slik Caltech 101, za katero se je učenje z atributi izkazalo relativno uspešno, pri tem pa je treba upoštevati, da nismo uporabili nobenih dodatnih atributov, ki bi bili bolj primerni za kategorije iz baze Caltech. Čeprav je skozi vse teste možno opaziti, da se učenje brez atributov sicer obnese bolje, pa se mu učenje z atributi po vseh izboljšavah zelo dobro približa. Skupaj z dodatnimi zmožnostmi, kot so učenje brez slik, učenje na majhnem številu primerov, poročanje nenavadnih lastnosti itd., postane majhna razlika v slabši kategorizaciji skoraj zanemarljiva.

V diplomski nalogi smo poskušali pokazati tudi, da je mogoče originalne bazne značilke delno zamenjati z značilkami LHOP, ter dobiti enak ali boljši rezultat. Tega iz rezultatov sicer ni mogoče zanesljivo trditi, saj se je v večini primerov originalna značilka izkazala za boljšo, vendar pa še vedno obstajajo primeri, ki kažejo na dejstvo, da se ponekod izkaže za boljšo značilka LHOP. Predvsem se ta bolje izkaže v primerih, kjer je zelo pomembna oblika, ter tudi pri uporabi semantičnih atributov, naučenih na izbranih značilkah. Čeprav je

LHOP v drugih primerih slabši, pa je izbiranje značilk pomembna izboljšava, saj glede na podatke iz [23] močno izboljša dodatne zmožnosti atributnega učenja, kot so poročanje nenavadnih lastnosti in učenje brez slik.

5.2 Nadaljnje delo

Navkljub relativno dobrim rezultatom pa še vedno obstaja veliko prostora za izboljšave. Boljše rezultate bi lahko dobili z izboljšavo posameznih delov sistema. Prva možnost je izboljšava učenja samih atributov, ki bi lahko glede na nekatere zgornje rezultate drastično izboljšala končno uspešnost kategorizacije. Prav to bi lahko bil tudi predmet nadaljnjih raziskav, kjer bi lahko preverjali, kako se uspešnost kategorizacije izboljšuje s postopno izboljšavo natančnosti atributov. Druga možnost za izboljšave bi lahko bilo povečanje števila semantičnih atributov. Pri tem bi lahko tudi preverjali, kako se kategorizacija spreminja z dodajanjem oz. odstranjevanjem semantičnih atributov ter ali obstaja kakšna točka, proti kateri s tem kategorizacija konvergira. Za izboljšano zaznavanje objektov na slikah, katerih lokacije ne poznamo, bi bilo smisleno vpeljati še metodo premikajočih se oken (ang. *sliding window*), saj bi lahko s tem natančneje zajeli le dejanske lastnosti objekta.

Smiselno bi bilo tudi vpeljati sistem, ki bi z dodajanjem novih kategorij in ustrezno analizo zmogel sam poiskati nove pomembne atribute ter ob določeni človeški pomoči te atribute tudi enostavno označiti z besedo. Te atribute bi se lahko učil tudi nenadzarovano preko interneta, kjer bi lahko s pomočjo internetnih slovarjev, kot je WordNet [55], poiskal semantične atribute, nato pa bi zanje v internetnih iskalnikih slik, kot sta Google in Yahoo, samodejno poiskal učne primere teh atributov, ter se jih tako naučil brez nadzora uporabnika.

Zanimivo bi bilo tudi poskusiti združiti atributno učenje objektov z modelom LHOP, ne le kot del značilke do 3. nivoja, ampak povsem združiti v nov model, ki bi združeval prednosti obeh. Po eni strani bi lahko bil sposoben učenja oblike in lastnosti glede na geometrijske značilnosti ter enostavnega dodajanja novih kategorij, podobno kot to omogoča model LHOP. Po drugi strani pa bi lahko uporabljal vizualne lastnosti, npr. teksturo in barvo, kot jih ima atributni model, ter imel dodatne zmožnosti, kot so ugotavljanje nenavadnih lastnosti objekta ter učenje brez slik, če bi vpeljali koncept semantičnih atributov.

Glede na vse dosedanje rezultate in ideje o možnih izboljšavah je mogoče sklepati, da ima učenje kategorij s pomočjo atributov še velik potencial, in bi lahko predstavljalo možno pot do boljše splošne kategorizacije objektov.

Slike

Tabele

Literatura

[1] C. Harris, M.J. Stephens., "A combined corner and edge detector," *Alvey Vision Conference*, 1988, str. 147–152.

[2] K. Mikolajczyk. C. Schmid, "Scale and affine invariant interest point detectors," *IJCV*, št. 1, zv 60, str. 63-86, 2004.

[3] J.Matas, O. Chum, M. Urban, T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," *BMVC*, str. 384-393, 2002.

[4] T. Kadir, A. Zisserman, M. Brady, "An affine invariant salient region detector," *ECCV*, str. 404-416, 2004.

[5] D. Lowe, "Distinctive image features from scale invariant keypoints," *IJCV*, št. 2, zv. 60, str. 91-110, 2004.

[6] S. Dickinson, "The Evolution of Object Categorization and the Challenge of Image Abstraction," S. Dickinson, A. Leonardis, B. Schiele, and M. Tarr, (eds), *Object Categorization: Computer and Human Vision Perspectives*, Cambridge University Press, str. 1-37, 2009.

[7] A. Pinz, "Object Categorization," *Foundations and Trends® in Computer Graphics and Vision*, št. 4. zv. 1, 2006.

[8] M. Weber, M. Welling, P. Perona, "Unsupervised Learning of Models for Recognition," *ECCV*, 2000.

[9] R. Fergus, P. Perona, A. Zisserman, "Object Class Recognition by Unsupervised Scale-Invariant Learning," *CVPR*, 2003.

[10] M. Fischler, R. Elschlager, "The Representation and Matching of Pictoral Structures," *IEEE Transactions on Computers*, 1973.

[11] G. Csurka, C. Bray, C. Dance, L. Fan, "Visual categorization with bags of keypoints," *Proc. of the 8th ECCV*, Prague, May 2004.

[12] H. Rowley, S. Baluja, T. Kanade, "Neural network-based face detection," *IEEE PAMI*, št. 1, zv. 20, str, 23–38, jan 1998.

[13] A.B.J. Teoh, D.C.L Ngo, A. Goh, "BioHashing: two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition*, zv. 37, str. 2245-2255, 2004.

[14] N. Dalal, B.Triggs, "Histograms of oriented gradients for human detection," *CVPR*, 2005.

[15] K. Grauman, T. Darrell, "The Pyramid Match Kernel:Discriminative Classification with Sets of Image," *ICCV*, 2005.

[16] M. Riesenhuber, T. Poggio, "Hierarchical Models of Object Recognition in Cortex," *Nature Neuroscience 2*, str. 1019-1025, 1999.

[17] N. Logothetis, J. Pauls, T. Poggio, "Shape representation in the inferior temporal cortex of monkeys," *Curr. Biol. 5*, str 552–563, 1995.

[18] J. Mutch, D. G. Lowe, "Multiclass Object Recognition with Sparse, Localized Features," *CVPR 2006, IEEE Computer Society Press*, New York, junij 2006, str. 11-18.

[19] R. Caruana, "Multitask learning," *Machine Learning*, št. 1, zv. 28, str 41–75, 1997.

[20] A. Torralba, K.P. Murphy, W. T. Freeman, "Sharing features: efficient boosting procedures for multiclass object detection," *CVPR*, 2004.

[21] A. Opelt, A, Pinz, A. Zisserman, "Learning an alphabet of shape and appearance for multi-class object detection," *International Journal of Computer Vision*, zv. 80, str. 16–44, 2008.

[22] V. Ferrari, A. Zisserman. "Learning visual attributes," *NIPS*, 2007.

[23] A. Farhadi, I. Endres, D. Hoiem, D.A. Forsyth, "Describing Objects by their Attributes," *CVPR*, 2009.

[24] C. H. Lampert, H, Nickisch, S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," *CVPR*, 2009.

[25] S. Fidler, A. Leonardis, "Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts," *CVPR*, 2007.

[26] M. Varma, A. Zisserman, "A statistical approach to texture classification from single images," *Int. J. Comput. Vision*, zv. 62, str. 61–81, 2005.

[27] J. Canny, "A Computational Approach To Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, zv. 8, str. 679–714, 1986.

[28] P. Felzenszwalb, D. McAllester, D. Ramanan, "A Discriminatively Trained, Multiscale, Deformable Part Model," *Proceedings of the IEEE CVPR*, 2008.

[29] Andrew Y. Ng, "Feature selection, L1 vs. L2 regularization, and rotational invariance," *Proceedings of the twenty-first international conference on Machine learning*, str. 78, julij, 2004.

[30] Tom Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers," *Kluwer Academic Publisher*, 2004.

[31] J. Chen, Y. Tong, W. Gray, Q. Ji, "A robust 3D eye gaze tracking system using noise reduction," *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*, 2008.

[32] J. Orozco, F. Xavier Roca, J. Gonzalez, "Real-time gaze tracking with appearance-based models," *Machine Vision and Applications*, 2008.

[33] J.M. Aguilera, A. Cipriano, M. Eraña, I. Lillo, D. Mery, A. Soto, "Computer Vision for Quality Control in Latin American Food Industry," A Case Study, *ICCV*, 2007 .

[34] J. Blasco, N. Aleixos, S. Cubero, J. Gomez-Sanchis, E. Molto, "Automatic sorting of satsuma (Citrus unshiu) segments using computer vision and morphological features," *Computers and Electronics in Agriculture*, št. 1, zv. 66, str. 1-8, 2009.

[35] P. Peer, B. Batagelj, F. Solina, "Using Computer Vision in Security Applications," *International Symposium on Electronics in Traffic ISEP'02*, Eds. M. Anžek, S. Petelin, P. Verlič, str. V10 6 strani, Ljubljana, Slovenia, October 2002.

[36] B. Coifman, D. Beymer,P. McLauchlan, J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, št. 4, zv. 6, str. 271-288, 1998.

[37] Microsoft Project Natal. Dostopno na: http://en.wikipedia.org/wiki/ProjectNatal

[38] Omek Interactive, SHADOW SDK. Dostopno na: http://www.omekinteractive.com/

[39] M.F. Duarte, Y.H Hu, "Vehicle classification in distributed sensor networks," *Journal of Parallel and Distributed Computing*, št. 7, zv. 64, str. 826-838, 2004.

[40] G. Zhang, R.P. Avery, Y. Wang, "A Video-based Vehicle Detection and Classification System for Real-time Traffic Data Collection Using Uncalibrated Video Cameras," *Transportation Research Record: Journal of the Transportation Research Board*, No. 1993, TRB, National Research Council,Washington, D.C., str. 138-147, 2007.

[41] S. Gupte, O. Masoud, R.F.K. Martin, and N.P. Papanikolopoulos, "Detection and Classification of Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, št. 1, zv. 3, str. 37-47, 2002.

[42] K. Nevatia, T.O. Binford, "Structured descriptions of complex objects," *Proceedings of the 3rd international Joint Conference on Artificial intelligence*, str. 641-647, avgust, 1973,

[43] G.J. Agin, T.O. Binford, "Computer Description of Curved Objects," *Proceedings of Third International Joint Conference on Artificial Intelligence*, Stanford, California, str. 629-635, avgust. 1973.

[44] D. Marr, H. Nishihara, "Representation and recognition of the spatial organization of three dimensional shapes," *Proc. Royal Soc.,* London B 200, str. 269–294, 1978.

[45] D. Lowe, "Perceptual Organization and Visual Recognition," *Kluwer Academic*, 1985.

[46] D. Huttenlocher, S. Ullman, "Recognizing Solid Objects by Alignment with an Image," *Int'l J. Computer Vision*, št. 2, zv. 5, str. 195-212, 1990.

[47] N. Ayache, O.D. Faugeras, "HYPER: A new approach for the recognition and positioning of two-dimensional objects," *IEEE Trans. Patt. Anal. Mach. Intell*, št 1, zv 8, str. 44-54, 1986.

[48] R.A. Brooks, "Symbolic reasoning around 3-D models and 2-D images," *Artificial Intelligence J. 17*: str 285-348, 1981.

[49] R. A. Brooks, R. Creiner, T. O. Binford, "The ACRONYM model-based vision system," *International Joint Conference On Artificial Intelligence*, str. 105-113, 1979.

[50] M. Muja, D.G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," *International Conference on Computer Vision Theory and Applications (VISAPP'09)*, INSTICC Press, 2009, str. 331-340.

[51] S.P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, zv. 28, str. 129-137, 1982.

[52] C. Schmid, "Constructing models for content-based image retrieval," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, zv. 2, str. 39-45, 2001.

[53] T. Leung, J. Malik, "Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons," *International Journal of Computer Vision*, št. 1, zv. 43, str. 29-44, 2001.

[54] L. Fei-Fei, R. Fergus, P. Perona, "One-Shot learning of object categories," *IEEE Trans. Pattern Recognition and Machine Intelligence*, v tisku.

[55] George A. Miller, "WordNet - About Us," *WordNet, Princeton University*, 2009. Dostopno na: `http://wordnet.princeton.edu`

[56] M. Turk, A. Pentland, "Face recognition using eigenfaces," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991.

[57] S. Fidler, M. Boben, A. Leonardis, "Learning a Hierarchical Compositional Shape Vocabulary for Multi-class Object Representation". Dostopno na: `http://civs.stat.ucla.edu/sig09/leonardis_sig09_slides.pdf`