SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1546

Slabo nadzirana semantička segmentacija rukom pisanih jednadžbi

Josip Milić

Zagreb, lipanj 2017.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA DIPLOMSKI RAD PROFILA

Zagreb, 15. ožujka 2017.

DIPLOMSKI ZADATAK br. 1546

Pristupnik: **Josip Milić (0036456339)**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Slabo nadzirana semantička segmentacija rukom pisanih jednadžbi**

Opis zadatka:

Semantička segmentacija je važan zadatak računalnog vida. Cilj tog zadatka je odrediti semantički razred svakog pojedinog piksela slike strojno naučenim modelom. Kao i u drugim problemima računalnog vida, najbolji rezultati postižu se strogo nadziranim učenjem. Međutim, velik nedostatak tog pristupa je potreba za ručnim označavanjem svakog pojedinog piksela u skupu slika za učenje. Kako bismo smanjili potrebu za tim skupim procesom, razmatramo mogućnost slabo nadziranog učenja segmentacijskih modela. U ovom pristupu, označavanje skupa za učenje ne provodi se na razini piksela, nego je za svaku sliku poznato samo nalazi li se u njoj objekt željenog razreda ili ne.
U okviru rada, potrebno je proučiti i ukratko opisati postojeće slabo nadzirane segmentacijske pristupe utemeljene na dubokom učenju. Naučiti plitki klasifikacijski model nad agregiranim konvolucijskim značajkama. Agregaciju provesti usrednjavanjem značajki ili njihovih Fisherovih reprezentacija. Naučeni klasifikacijski model primijeniti konvolucijski nad Fisherovim vektorima slikovnih okana. Uhodati postupke učenja i validiranja hiperparametara. Primijeniti naučene modele na zbirci slika matematičkih jednadžbi tvrtke Photomath. Prikazati i ocijeniti ostvarene rezultate. Predložiti pravce budućeg razvoja.
Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Zadatak uručen pristupniku: 10. ožujka 2017.
Rok za predaju rada: 29. lipnja 2017.

Mentor:

Izv. prof. dr. sc. Siniša Šegvić

Djelovođa:

Doc. dr. sc. Tomislav Hrkać

Predsjednik odbora za diplomski rad profila:

Prof. dr. sc. Siniša Srbljić

Zahvaljujem se svom mentoru prof. dr. sc. Šiniši Šegviću na savjetima i pomoći tokom izrade ovog rada.

Zahvaljujem se također i Jurici Cerovcu na ustupljenim anotiranim slikama tvrtke Photomath.

SADRŽAJ

1. Uvod

Klasifikacijom slike možemo dobiti informaciju o tome što ona predstavlja no u nekim slučajevima je potreban finiji postupak koji bi odredio gdje se točno nalazi područje slike koje pripada određenoj kategoriji odnosno koji pikseli slike odgovaraju nekoj zadanoj kategoriji. Taj postupak naziva se semantička segmentacija (engl. *semantic segmentation*). Njome se omogućava razdvajanje slike na smislene cjeline koje se mogu različito interpretirati u nekom mogućem daljnjem postupku ili potpuno odbaciti ako nisu od interesa.

Tako se na primjer može razdvojiti slika scene prometa na različite dijelove i predati ih specijaliziranim klasifikatorima. Dio slike koji predstavlja prometni znak može interpretirati klasifikator prometnih znakova, dio slike koji predstavlja kolnik može interpretirati klasifikator vrste prometnica, dio slike koji predstavlja nebo se može odbaciti i tako dalje. Interpretacijom pojedinih dijelova slike dolazi se do konačne informacije koja ta slika daje, na primjer o prometnoj situaciji i pravilima kojih bi se vozač ili autonomno vozilo trebalo pridržavati.

S obzirom na to da je očekivani izlaz segmentacije skup nezavisnih područja, postupak učenja također zahtijeva definirani skup dijelova slike što znači da je potrebno na svakoj slici, kojom učimo model, označiti koji piksel pripada kojoj kategoriji (ili u slučaju binarne klasifikacije pripadnost kategoriji). To je dugotrajan ručni postupak koji obično zahtijeva angažiranje anotatora (engl. annotators) koji bi to radili i s obzirom da se radi o anotaciji na razini piksela, provjera ispravnosti postupka nije jednostavna. Postoje različiti skupovi anotiranih slika kao što je to skup KITTI [8] no problem se događa kad je potrebno izvršiti semantičku segmentaciju slika koje su specifične za neki problem. Tako se u ovom radu obrađuje problem slabo nadzirane binarne semantičke segmentacije nad slikama koje sadrže rukom napisane jednadžbe. Cilj je za svaku sliku odrediti koji piksel pripada jednadžbi. Time se omogućava ekstrakcija samo relevantnih piksela koji se zatim mogu predati klasifikatoru prepoznavanja znakova (OCR, engl. *Optical Character Recognition*) koji bi prepoznao znakove i znamenke napisane jednadžbe.

2. Reprezentacija primjeraka Fisherovim vektorima

U kontekstu opisa postupaka korištene slike za učenje i ispitivanje nazivamo primjercima (engl. *samples*). Svaka slika sastoji se od piksela koji koji daju informaciju o razinama svjetline točke u dvodimenzionalnom prostoru. Tako se primjerice, u slučaju RGB slike (engl. *Red Green Blue image*) svaki piksel sastoji od tri razine koje zajedno definiraju neku boju, a u slučaju crno-bijele slike koristi se samo jedna razina. Za opis pojedinih razina obično se koriste cijeli brojevi od 0 do 255. Slika se tako zapisuje kao matrica razina čije su širina i dužina jednake dimenzijama slike, a dubini odgovara broj razina koje opisuju pojedini piksel. Takav način zapisa je prilagođen za prikaz ljudskom vidu. Iako su se kroz povijest koristili i takvi zapisi u postupcima računalnog vida, pokazalo se svrsishodnije koristiti neku reprezentaciju koja je prilagođena računalu. Tako se obično najprije iz slike izluče značajke (engl. *features*). Značajke još nazivamo opisnicima jer sadrže neku korisnu informaciju o slici ili ili dijelu slike koja se koristi u daljnjem postupku. U ovom radu obavlja se ekstrakcija značajki i zatim njihova pretvorba u Fisherove vektore. Time se dobiva reprezentacija primjeraka koja je prikladna za korišten način segmentacije slike.

Slijedi opis nekih opisnika i način njihovog dobivanja. Zatim se opisuje generativni model koji služi za stvaranje Fisherovih vektora, i konačno opis Fisherovih vektora koji su poslužili kao reprezentacija korištenih primjeraka.

2.1. Lokalni opisnici

Ekstrakcijom opisnika iz primjeraka dobivaju se vektori značajki koji se mogu potom koristiti u algoritmima strojnog učenja koji rezultiraju primjerice lokalizacijom objekata. Razlikujemo opisnike pojedinih dijelova slike (slikovna okna) i nazivamo ih opisnicima niske razine ili lokalnim opisnicima. U kontekstu ovog rada važno je

da su vektori slikovnih okana neosjetljivi u odnosu na lokalne pomake i fotometrijske promjene na slici.

Razlikujemo dvije glavne vrste lokalnih opisnika [29]:

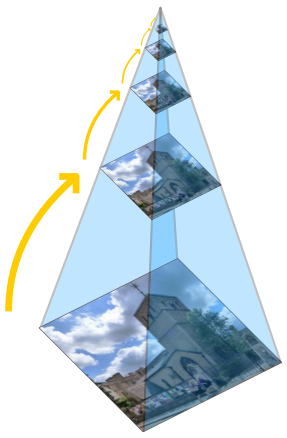
– opisnici koji su prilagođeni za korišteni algoritam (engl. *hand-crafted descriptors*)

– opisnici koji su dobiveni učenjem (engl. *learned local descriptors*)

2.1.1. Opisnici prilagođeni korištenom algoritmu

Jedni od najpopularnijih opisnika prilagođenih korištenom algoritmu su SIFT (engl. *Scale Invariant Feature Transform*) opisnici. Prije njihove pojave postojale su značajke nastale detekcijom kutova (primjerice Harrisovim detektorom [10]) koji su bile neosjetljive na rotaciju, ali ne i na skaliranje slike. Zbog toga je razvijen algoritam SIFT [20] čiji rezultat su značajke koje su invarijantne na skaliranje.

Kako bi se detektirali kutovi različitih veličina potrebno je koristiti i prozore (engl. *windows*) različitih veličina. Za sliku se pronalazi Laplasijan [1] Gaussovog filtra (LoG, *Laplacian of Gaussian*) za različite σ i služi kao detektor zaobljenja (engl. *blob detector*). Ovisno o manjoj ili većoj σ Gaussova jezgra daje veću ili manju vrijednost za različite veličine kutova. Među različitim (x, y, σ) pronalazi se lokalni maksimum, a time i potencijalna značajka slike.



Slika 2.1: Primjer Gaussove piramide. Svaka razina predstavlja zamućenje i smanjenje rezolucije. Slika je preuzeta iz [5]

[1] Laplasijan - diferencijalni operator koji daje divergenciju gradijenta skalarne funkcije

Opisani način je računalno zahtijevan proces kojeg SIFT mijenja s razlikom Gaussovih filtara (DoG, *Difference of Gaussians*). Razlika Gaussijana je razlika Gaussovih zamućenja slike s različitim $k\sigma$, $k = \mathbb{R}$. DoG se obavlja za različite oktave slike Gaussove piramide (Slika 2.1). Zatim se za svaki piksel unutar DoG-a promatra njegovo susjedstvo (8 okolnih piksela) i uspoređuje sa 9 piksela na sljedećoj i prethodnoj skali. Ako piksel predstavlja lokalni ekstrem, smatra se potencijalno važnim dijelom slike.

Osim SIFT značajki kao opisnici prilagođeni korištenom algoritmu koriste se SURF (engl. *Speeded Up Robust Features*), LBP (engl. *Local Binary Patterns*) te HOG (engl. *Histogram of Oriented Gradients*) značajke. [29]

2.1.2. Opisnici dobiveni učenjem

Najbolji primjer za opisnike dobivene učenjem su konvolucijske značajke dobivene konvolucijskom neuronskom mrežom.

Konvolucijska neuronska mreža

Konvolucijska neuronska mreža (CNN, engl. *Convolutional Neural Network*) je vrsta neuronske mreže koja je posebno prilagođena slikama. Omogućava prepoznavanje scene i klasifikaciju, i općenito predstavlja vrlo moćan alat računalnog vida. Početak razvoja i korištenja KNN-a pa tako i područja dubokog učenja (engl. *deep learning*) smatra se početak 90-ih godina prošlog stoljeća. U to vrijeme računalna moć je bila ograničena, procesori su bili komputacijski slabi, a grafička jedinica se nije ni mogla koristiti u tu svrhu. Potrebno je bilo pronaći metodu koja bi ograničila broj korištenih parametara i potrebu za velikim računalnim resursima. Uobičajena praksa je bila postaviti sliku na ulaz višeslojnog perceptrona (engl. *multi-layer neural network*) pri čemu se svaki piksel smatrao nezavisnim ulazom. Yann LeCun je 1994. razvio arhitekturu KNN-a LeNet5 [18] koja iskorištava prostornu korelaciju piksela na slici.

Struktura te arhitekture čini jezgru današnjih arhitektura KNN-a [2]:

- koristi se tri vrste slojeva: sloj konvolucije, sloj sažimanja (engl. *pooling*) i sloj nelinearnosti
- konvolucijom se iz slike izvlače prostorne značajke
- značajke se poduzorkuju (engl. *subsampling*) koristeći prostorne mape
- aktivacijskim funkcijama uvodi se nelinearnost (primjer su tangens hiperbolni i sigmoida)

– slojevi su međusobno nepotpuno povezani kako bi se izbjegla (pre)velika količina parametara

– višeslojni perceptron (MLP) se koristi kao konačni klasifikator

Razvoj i izum novi arhitektura konvolucijskih neuronskih mreža je stalno u tijeku, naročito od 2010. na dalje zbog dostupnosti snažnije računalne moći po pristupačnoj cijeni i novim mogućnostima poput učenja KNN-a pomoću grafičkih jedinica (engl. *GPU neural net*) [4] što je praktički postao standard. Konvolucijske neuronske mreže i općenito duboko učenje je vrlo popularno područje računarske znanosti, a budućnost razvoja je zajamčena zbog razvoja tehnologija koje koriste neki oblik umjetne inteligencije poput primjerice autonomnih vozila (engl. *autonomous vehicles*). Razvoj hardvera donosi moćnije resurse za manju cijenu čime alati postaju još dostupniji široj javnosti.

Neke od istaknutijih arhitektura KNN-a:

– AlexNet [17] je uvela zglobnicu kao aktivaciju (ReLU, engl. *Rectified Linear Units*) i metodu izbacivanja pojedinih neurona (engl. *dropout*) prilikom učenja čime se smanjuje prenaučenost modela. Ostvarila je značajne rezultate nad skupom ImageNet.

– VGG [25] je koristila manje (3×3) filtre i veliki broj slojeva, od 11 u A inačici sve do 19 u E inačici (prikazana na Slici 4.5). To rezulira velikim brojem parametara, do čak 144 milijuna, no ujedno i postiže veliku moć učenja.

– GoogLeNet (Inception) [27] je smanjila broj operacija koje su potrebe za učenje KNN-a tako što je uvela paralelne konvolucijske filtre i tzv. usko grlo (engl. *bottleneck*) čime se smanjuje broj značajki.

– ResNet [11] je imao jednostavnu, ali vrlo važnu ideju: s izlazima dva uzastopna sloja zbraja se ulaz. Time se smanjuje broj značajki za svaki sloj i omogućava korištenje velike količine (> 100) slojeva.

Konvolucijske značajke

Konvolucijske značajke se dobivaju pomoću naučene konvolucijske neuronske mreže, obično, na velikom skupu podataka. Na ulaz mreže postavlja se slika, a na izlazu pojedinog konvolucijskog sloja dobivaju se konvolucijske značajke. One se mogu koristiti kao reprezentacija slike i biti ulaz nekog klasifikatora. Pokazalo se da odzivi pojedinih kovolucijskih slojeva odgovaraju strukturama više razina u slici (Slika 4.7),

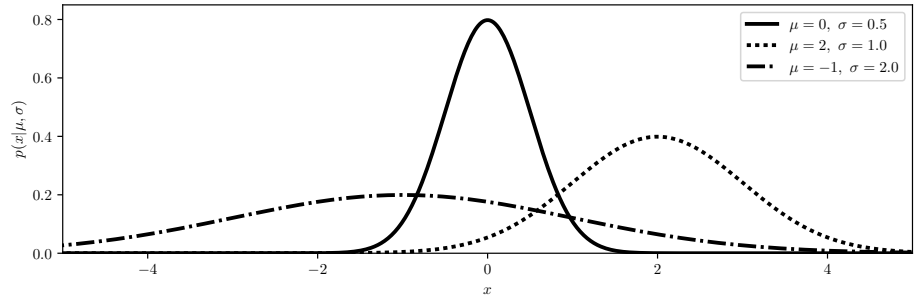
bez obzira što slika možda nije srodna korištenom skupu podataka za učenje mreže [9]. KNN se u tom slučaju koristi kao autoenkoder (engl. *autoencoder*) i obično se ne uzimaju sve konvolucijske značajke, već samo one dobivene određenim slojem, ovisno o željenim detaljima.

2.2. Mješavina Gaussovih razdiobi (GMM)

Prije opisa GMM-a potrebno je razumjeti njegove sastavne dijelove.

2.2.1. Gaussova razdioba

Normalna razdioba (poznata i pod imenom Gaussova razdioba, engl. *Gaussian distribution*) ima vrlo važnu ulogu u polju statistike. Analitički je relativno lako prilagodljiva, ima poznati oblik zvona zbog čega je uobičajena prilikom izrade modela populacije i može se koristiti za aproksimaciju velikog broja različitih razdiobi u velikom skupu uzoraka što je formalizirano Središnjim graničnim teoremom [2] [3].



Slika 2.2: Normalna (Gaussova) razdioba s različitim parametrima μ, σ^2

Normalna razdioba ima dva parametra: srednju vrijednost (očekivanje) μ i varijancu (kvadrat standardne devijacije) σ^2. Funkcija gustoće vjerojatnosti (PDF, engl. *probability density function*) normalne razdiobe dana je izrazom (2.1) i njezin oblik se, u ovisnosti o različitim parametrima, može pregledati na Slici 2.2.

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{(x-\mu)^2}{2\sigma^2}\} \tag{2.1}$$

[2] Središnji granični teorem (engl. *Central limit theorem*) tvrdi da razdioba sume ili prosjeka velikog broja nezavisnih varijabli uzorkovanih iz iste zajedničke razdiobe (iid, engl. *independent and identically distributed*) teži prema normalnoj razdiobi, bez obzira na razdiobu uzoraka [30]

Srednja vrijednost $\mu \in \mathbb{R}$ predstavlja tendenciju centra razdiobe, a varijanca $\sigma^2 \in (0, \infty)$ jačinu odstupanja od srednje vrijednosti.

U literaturi [19] može se pronaći efikasniji način evaluacije PDF-a normalne razdiobe koristeći zamjenski parametar $\beta \in (0, \infty)$:

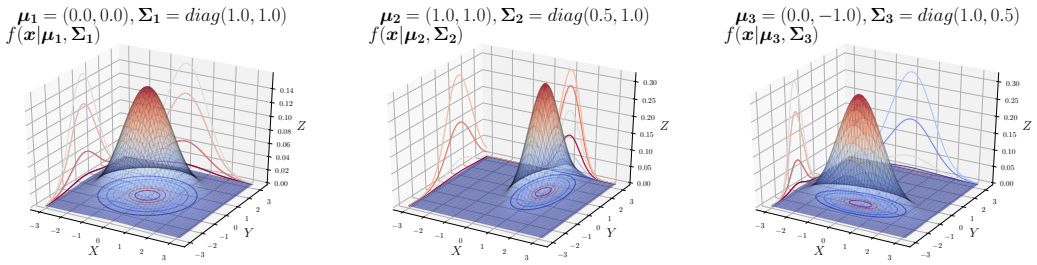
$$f(x|\mu, \beta^{-1}) = \sqrt{\frac{\beta}{2\pi}} \exp\{-\frac{1}{2}\beta(x-\mu)^2\} \tag{2.2}$$

2.2.2. Multivarijantna Gaussova razdioba

Poopćenjem Gaussove razdiobe na više (n) dimenzija dobiva se multivarijantna Gaussova razdioba (engl. *multivariate Gaussian distribution*):

$$f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\} \tag{2.3}$$

U ovom slučaju parametri su n-dimenzijski vektor srednje vrijednosti $\boldsymbol{\mu}$ i kovarijacijska matrica $\boldsymbol{\Sigma}$ dimenzija $n \times n$. Da bi vrijedio izraz (2.3) kovarijacijska matrica $\boldsymbol{\Sigma}$ mora biti simetrična i pozitivno definitna[3] kako bi mogla imati inverz i često se zadaje kao dijagonalna matrica.



Slika 2.3: Bivarijantna normalna (Gaussova) razdioba s različitim parametrima $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$

Na Slici 2.3 prikazan je izgled funkcije gustoće vjerojatnosti bivarijantne normalne razdiobe ($n = 2$) s različitim parametrima $\boldsymbol{\mu}$ i $\boldsymbol{\Sigma}$.

Kao i u slučaju univarijantne normalne razdiobe postoji efikasniji način evaluacije PDF-a koristeći zamjenski parametar $\boldsymbol{\beta} \in {\mathbb{R}_{0+}}^{n \times n}$ (još poznat kao matrica preciznosti, engl. *precision matrix*) umjesto računanja inverza kovarijacijske matrice $\boldsymbol{\Sigma}$ [3]:

$$f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\beta}^{-1}) = \sqrt{\frac{det(\beta)}{(2\pi)^n}} \exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\beta}(\mathbf{x}-\boldsymbol{\mu})\} \tag{2.4}$$

[3]Matrica $\boldsymbol{A}$ je pozitivno definitna akko $\mathbf{x}^T\boldsymbol{A}\mathbf{x} > 0$ za svaki ne-nul vektor $\mathbf{x}$

2.2.3. Mješavina razdiobi

Kombinacijom jednostavnijih razdiobi dobiva se mješavina razdiobi. Ona se sastoji od više komponenti od kojih svaka predstavlja jednu razdiobu. Uzorkovanjem identiteta komponente $P(c)$ iz kategoričke razdiobe (engl. *Multinoulli distribution*) saznaje se koja komponenta generira uzorak:

$$P(x) = \sum_i P(c = i)P(x|c = i) \tag{2.5}$$

Model mješavine omogućuje nam kratki uvid u važan koncept koji se naziva latentna varijabla. Latentna varijabla (engl. *latent variable*) je slučajna varijabla koju ne možemo izravno pratiti. U izrazu (2.5) latentna varijabla je identitet komponente mješavine c. Povezanost latentne varijable s x je preko zajedničke razdiobe: $P(x, c) = P(x|c)P(c)$. Razdioba $P(c)$ koja ovisi o latentnoj varijabli i uvjetna razdioba $P(x|c)$ opisuju oblik razdiobe $P(x)$ iako ju je moguće opisati i bez latentne varijable [19].

Vrlo moćan i popularan model mješavine razdiobi je mješavina normalnih (Gaussovih) razdiobi odnosno GMM (engl. Gaussian mixture model). Svaka njegova komponenta $p(\mathbf{x}|c = k)$ predstavlja jednu Gaussovu razdiobu koja ima svoje parametre, vektor srednje vrijednosti $\boldsymbol{\mu}^{(k)}$ i matricu kovarijacije $\boldsymbol{\Sigma}^{(k)}$. GMM sadrži još i apriori vjerojatnosti $w_k = P(c = k)$ za svaku komponentu k. Mješavina Gaussovih razdiobi smatra se univerzalnim aproksimatorom gustoća vjerojatnosti. Svaka glatka gustoća vjerojatnosti može se aproksimirati uz dovoljnu količinu komponenti [19].

Izraz (2.6) formalno opisuje funkciju gustoće vjerojatnosti (PDF) mješavine Gaussovih razdiobi, a izraz (2.7) PDF pojedine komponente[29]:

$$p(\mathbf{x}; \boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \cdot p(\mathbf{x}; \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \tag{2.6}$$

$$p(\mathbf{x}; \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma_k}|^{1/2}} \exp\Big\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma_k^{-1}}(\mathbf{x} - \boldsymbol{\mu_k})\Big\} \tag{2.7}$$

gdje su:

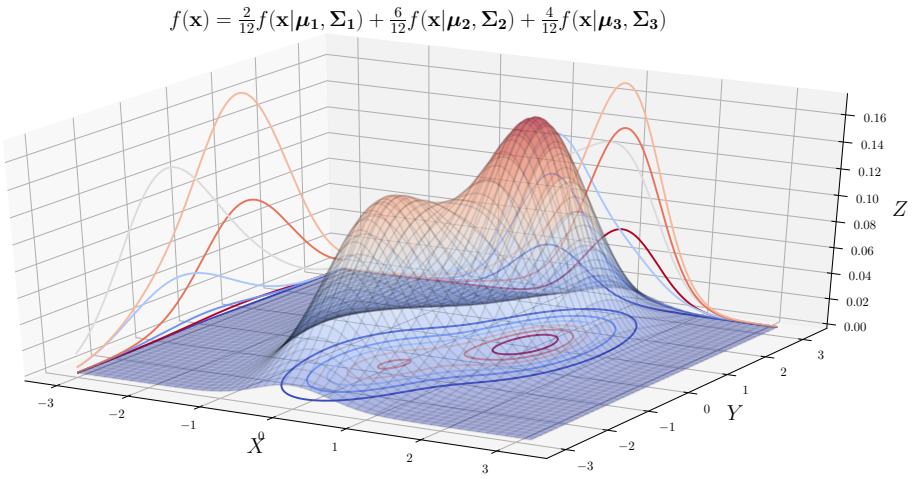
$\boldsymbol{\theta} = \{w_k, \boldsymbol{\mu_k}, \boldsymbol{\sigma_k}\}_{k=1}^{K}$,

$w_k \in \mathbb{R}$,

$\boldsymbol{\mu_k} \in \mathbb{R}^D$,

$$\boldsymbol{\sigma_k^2} \in \mathbb{R}^D => \boldsymbol{\Sigma_k} \in \mathbb{R}^{D \times D}$$

Na Slici 2.4 može se vidjeti prikaz PDF-a mješavine Gaussovih razdiobi koja je nastala miješanjem razdiobi prikazanih na Slici 2.3 i množenjem s različitim parametrima miješanja w_k.



Slika 2.4: Mješavina Gaussovih razdiobi (GMM) s parametrima $\boldsymbol{\mu_i}$, $\boldsymbol{\Sigma_i}$ iz Slike 2.3

Model GMM-a može se naučiti algoritmom maksimizacije očekivanja (EM, engl. *Expectation-Maximization*). To je iterativni algoritam koji počinje od neke početne procjene Φ, obično nasumično odabrane, i zatim ju iterativno osvježava sve dok ne postigne zadovoljavajuću konvergenciju. Svaka iteracija sastoji se od sljedećih koraka:

– E-korak (engl. *E-step*): izračunati težine pripadnosti w_k za sve točke ulaza x_i, $1 \leq i \leq N$ i za sve komponente mješavine $1 \leq k \leq K$. Zbroj težina pripadnosti mora biti $\sum_{k=1}^{K} = 1$ i to za svaki x_i što znači da se dobiva matrica težina pripadnosti dimenzija $N \times K$ gdje zbroj svakog retka jednak 1.

– M-korak (engl. *M-step*): koristeći izračunate težine pripadnosti osvježiti vrijednosti parametara $\boldsymbol{\mu}$ i $\boldsymbol{\Sigma}$.

Detaljan opis EM algoritma može se pronaći u [26]

2.3. Fisherova jezgra

Problemi računalnog vida, i općenito područja računarske znanosti poput prepoznavanja govora i analize teksta, mogu biti posebno kompleksni pri čemu uobičajeni alati statistike ne mogu pomoći. Primjer je lokalizacija objekata na slici koji mogu imati različite pozicije u sceni što uključuje manji ili veću udio u slici ovisno o udaljenosti od promatrača. Potrebno je pretvoriti korištene primjerke u vektorski oblik koji je pogodan za linearnu klasifikaciju . Jedan od načina kojim je to moguće ostvariti je učenje generativnog modela poput mješavine Gaussovih razdiobi (GMM). Koristeći naučeni GMM i jezgrene funkcije moguće je izvući diskriminativne značajke.

Jezgrene funkcije

Za primjer možemo imati skup podataka za učenje koji se sastoji od primjeraka X_i i koji sadrži odgovarajuće binarne oznake $Y_i \in \{-1, 1\}$ koje označavaju pripadnost razredu. Želimo klasificirati novi primjerak X. U slučaju jezgrenih metoda (engl. *kernel methods*) predikcija se ne računa izravno, već se uzima u obzir težinska suma oznaka primjeraka iz skupa podataka za učenje. Svakom primjerku X_i pridaje se odgovarajuća važnost u obliku parametra λ_i i računa se sličnost između X_i i X u obliku jezgrene funkcije (engl. *kernel function*) $K(X_i, X)$. Predviđena oznaka $\hat{Y}$ novog primjerka računa se kao:

$$\hat{Y} = sign\left(\sum_i Y_i \lambda_i K(X_i, X)\right) \tag{2.8}$$

Potrebno je riješiti problem optimizacije koji će pronaći prikladan koeficijent λ_i i potrebno je izabrati prikladnu jezgrenu funkciju. Jezgrene metode mogu predstavljati generalizirani linearni modeli kao što je to primjerice model logističke regresije. U njemu se računa vjerojatnost oznake Y za primjer X pomoću logističke funkcije (sigmoida, engl. *logistic function*) $f_\sigma(z) = (1 + e^{-z})^{-1}$ s parametrima θ:

$$P(Y|X, \theta) = f_\sigma(Y\theta^T X) \tag{2.9}$$

Maksimalna a posteriori estimacija (MAP, engl. *Maximum A posteriori Probability*) parametara θ nad skupom primjeraka za učenje pronalazi se maksimizacijom log-izglednosti (engl. *log-likelihood*) uz penalizaciju (2.10). Kao a priori razdioba $P(\sigma)$ uzima se normalna razdioba s matricom kovarijacije Σ. Konstanta c je nezavisna od parametara θ.

$$\sum_i logP(Y_i|X_i,\theta) + logP(\theta) = \sum_i logf_\sigma(Y_i\theta^T X_i)\frac{1}{2}\theta^T\Sigma^{-1}\theta + c \tag{2.10}$$

Rješenje problema može se zapisati kao:

$$\hat{\theta} = \sum_i Y_i\lambda_i\Sigma X_i \tag{2.11}$$

gdje je:

$\lambda_i = \frac{\partial}{\partial z}logf_\sigma(z) \mid_{z=Y_i\hat{\theta}^T X_i} = f_\sigma(-Y\hat{\theta}^T X_I)$

Koristeći (2.11) u (2.9) dobiva se uvjetna vjerojatnost oznake Y:

$$P(Y|X,\hat{\theta}) = f_\sigma(Y\sum_i Y_i\lambda_i(X_i^T\Sigma X)) \tag{2.12}$$

Uz određenje da je oznaka Y s najvećom vjerojatnosti ona koja ima predznak jednak sumi u argumentu i $K(X_i, X) = X_i^T\Sigma X$, dobiva se decizijska funkcija (engl. *decision function*) iz izraza (2.8).

Kako bi jezgrena funkcija bila ispravna, mora biti pozitivno semidefinitna. Prema Mercerovom teoremu, svaka takva jezgrena funkcija je produkt vektora značajki koje su dobivene nekim mapiranjem $X \rightarrow \phi_X$.

$$K(X_i, X_j) = \phi_{X_i}^T\phi_{X_j} \tag{2.13}$$

Tim produktom u prostoru značajki definira se Euklidski prostor. Euklidska udaljenost između dva vektora značajki može se dobiti izravno koristeći jezgrene funkcije:

$$\|\phi_{X_i} - \phi_{X_j}\|^2 = K(X_i, X_j) - 2K(X_i, X_j) + K(X_j, X_i) \tag{2.14}$$

Prikladna jezgrena funkcija ne može se dobiti nekim algoritmom i njezin odabir nije jednostavan, pogotovo ako se koriste primjerci različitih veličina.

Fisherova jezgra

Prilikom definicije jezgrene funkcije automatski se zaključilo o metričkim odnosima (engl. *metric relations*) između primjeraka no oni su se mogli dobiti pomoću generativnog modela $P(X|\theta)$. U slučaju klasifikacije u tom modelu može biti uključena i oznaka klasifikacije Y kao latentna varijabla. Za primjer klasifikacije s više razreda

(engl. *class*), koristimo skup podataka $S = \{(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \ldots, (\mathbf{X}_N, y_N)\}$ koji se sastoji od strukturiranih objekata $X_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, ..., \mathbf{x}_{nD}\}$ i pripadajućih oznaka $y_n \in \{1, 2, ..., C\}$. D predstavlja dimenziju primjerka, a C broj razreda. Svaki X_n može se prikazati kao graf $G_n = (V_n, E_n)$ u kojem i-ti vrh (V, engl. *Vertex*) odgovara $\mathbf{x}_{ni}$ i čiji rub (E, engl. *Edge*) (i, j) odgovara relaciji između $\mathbf{x}_{ni}$ i $\mathbf{x}_{nj}$. Za modeliranje razdiobe $p(\mathbf{X})$ moguće je koristiti Markovo nasumično polje (MRF, engl. *Markov Random Field*) nad latentnim varijablama $\mathbf{z} = \{z_1, z_2, \ldots, z_D\}$ s proizvoljnim razdiobama emisije (engl. *emission distributions*):

$$p(\mathbf{X}) \propto \sum_{\mathbf{z}} \left[\prod_{i \in V} p_\omega(\mathbf{x}_i | z_i) \right] \exp \left[\sum_{(i,j) \in E} A_{z_i z_j} \right] \tag{2.15}$$

gdje je $\mathbf{A}$ matrica log-tranzicija vjerojatnosti, a razdioba emisije $p_\omega(\mathbf{x}_i|z_i)$ može biti Gaussova ili neka multinomna razdioba s parametrima ω. Parametri MRF-a $\Theta = \{\omega, \mathbf{A}\}$ uče se na način koji maksimizira log-izglednost $L(S) = \sum_{n=1}^{N} \log p(\mathbf{X}_n)$. Učenje se može sprovesti algoritmom maksimizacije očekivanja ili gradijentnog uzlaza (Δ , engl. *gradient ascent*).

Fisherova jezgra podrazumijeva da dva slična objekta $\mathbf{X}_n$ i $\mathbf{X}_m$ imaju slične parcijalne derivacije $\frac{\partial L(\mathbf{X}_n)}{\partial \theta}$ i $\frac{\partial L(\mathbf{X}_m)}{\partial \theta}$ za sve $\theta \in \Theta$ po generativnom modelu [13]. Za pojednostavljenje zapisa, gradijent log-izglednosti $L(\mathbf{X}_n)$ jednog strukturiranog objekta $\mathbf{X}_n$ po parametrima modela zapisuje se kao $\mathbf{g}_n = \left[\forall \theta \in \Theta : \frac{\partial L(\mathbf{X}_n)}{\partial \theta} \right]$. Naziva se još i Fisherovim odzivom (engl. *Fisher score*). Koristeći odzive $\mathbf{g}_n$ kao značajke objekta $\mathbf{X}_n$, Fisherova jezgrena funkcija κ zapisuje se kao:

$$\kappa(\mathbf{X}_i, \mathbf{X}_j) = \mathbf{g}_i^T \mathbf{U}^{-1} \mathbf{g}_j \tag{2.16}$$

gdje je $\mathbf{U}$ Fisherova informacijska matrica (engl. *Fisher information matrix*) koja predstavlja lokalnu metriku Riemannovskog manifolda (engl. *Riemannian manifold*) M_Θ:

$$\mathbf{U} = E \left[\left(\frac{\partial L(\mathbf{X})}{\partial \Theta} \right)^T \left(\frac{\partial L(\mathbf{X})}{\partial \Theta} \right) \right]_{p(\mathbf{X})} \tag{2.17}$$

Fisherova informacijska matrica $\mathbf{U}$ predstavlja kovarijacijsku matricu Fisherovih odziva i pomoću nje Fisherova jezgra postaje neosjetljiva na reparametrizaciju modela θ [29].

Njezin izračun je računalno zahtijevan i umjesto nje se u praksi koriste različite aproksimacije ili se čak zanemaruje [13], odnosno postavlja se kao jedinična matrica

I. Takva Fisherova jezgra jednostavno množi gradijente $\mathbf{g}_n$ kao značajke, bez dodatnog skaliranja ili normalizacije.

2.4. Fisherovi vektori

Fisherova informacijska matrica je pozitivno definitna zbog čega vrijedi:

$$\alpha^T \mathbf{U} \alpha = E_{\mathbf{X}\ p(\mathbf{x};\theta)} \left[(g(\mathbf{x};\theta)^T \alpha)^2 \right] > 0 \tag{2.18}$$

uz uvjet $\alpha \neq \mathbf{0}$.

To znači da se može obaviti dekompozicija njezina inverza $\mathbf{U}^{-1} = \mathbf{L}^T \mathbf{L}$ i posljedično prikazati Fisherovu jezgru kao skalarni produkt funkcija preslikavanja [29]

$$\kappa(\mathbf{X}_i, \mathbf{X}_j) = \phi_\theta(\mathbf{X}_i)^T \phi_\theta(\mathbf{X}_j) \tag{2.19}$$

Funkcija preslikavanja $\phi_\theta(\mathbf{X})$ naziva se Fisherov vektor (engl. *Fisher vector*):

$$\phi_\theta(\mathbf{X}) = \mathbf{L} \cdot \mathbf{g} \tag{2.20}$$

U kontekstu slika skup vektora čini $\mathbf{X} = \{\mathbf{x}_t\}$, gdje su $\mathbf{x_t} \in \mathbb{R}^D$, $t = 1 \dots T$, lokalni opisnici dobiveni uzorkovanjem primjeraka. Primjer su SIFT opisnici i konvolucijske značajke. Pretpostavka je da su korišteni lokalni opisnici nezavisni i jednoliko raspodijeljeni (iid, engl. *independent and identically distributed*).

Za opis skupa $\mathbf{X}$ koristi se generativni model. To može biti mješavina Gaussovih razdiobi s parametrima $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k \boldsymbol{\sigma}_k\}_{k=1}^K, \omega_k \in \mathbb{R}, \boldsymbol{\mu}_k \in \mathbb{R}^D, \boldsymbol{\sigma}_k \in \mathbb{R}^D$, gdje je K broj komponenti. Vjerojatnost generiranja opisnika $\boldsymbol{x}$ može se izračunati po komponenti GMM-a [29]:

$$p(k|\mathbf{x}) = \frac{p(k) \cdot p(\mathbf{x};k)}{p(\mathbf{x})} = \frac{w_k \cdot p(\mathbf{x};\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)}{p(\mathbf{x},\boldsymbol{\theta})} = \frac{w_k \cdot p(\mathbf{x};\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)}{\sum_{i=1}^K w_i \cdot p(\mathbf{x};\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)} \tag{2.21}$$

gdje je w_k težina miješanja.

Gradijenti logaritma funkcije izglednosti $p(\mathbf{x};\boldsymbol{\theta})$ u odnosu na parametre α_k, $\boldsymbol{\mu}_k$ i $\boldsymbol{\sigma}_k$ definirani su sljedećim izrazima [23]:

$$\phi_{\alpha_k}(\mathbf{x}) = \frac{p(k|\mathbf{x}) - w_k}{\sqrt{w_k}} \tag{2.22}$$

$$\phi_{\boldsymbol{\mu}_k}(\mathbf{x}) = \frac{p(k|\mathbf{x})}{\sqrt{w_k}} \cdot \frac{\mathbf{x} - \boldsymbol{\mu}_k}{\boldsymbol{\sigma_k}} \tag{2.23}$$

$$\phi_{\boldsymbol{\sigma}_k}(\mathbf{x}) = \frac{p(k|\mathbf{x})}{\sqrt{2w_k}} \cdot \left[\frac{(\mathbf{x} - \boldsymbol{\mu}_k)^2}{\boldsymbol{\sigma}_k^2} - 1 \right] \tag{2.24}$$

Konačna reprezentacija u obliku Fisherovog vektora dobiva se konkatenacijom navedenih gradijenata za sve komponente $k = 1 \dots K$ u jedinstven vektor $\phi_{\boldsymbol{\theta}}(\mathbf{x})$ čija je dimenzionalnost posljedično jednaka $K(2D + 1)$.

Izglednost skupa nezavisnih jednoliko raspodijeljenih podataka $\mathbf{X} = \{\mathbf{x}_t, t = 1 \dots T\}$ odgovara umnošku $p(\mathbf{X}; \boldsymbol{\theta}) = \prod_{t=1}^{T} p(\mathbf{x}_t; \boldsymbol{\theta})$. U praksi se zbog jednostavnosti koristi logaritam izglednosti $\ln p(\mathbf{X}; \boldsymbol{\theta}) = \sum_{t=1}^{T} \ln p(\mathbf{x}_t; \boldsymbol{\theta})$. S obzirom na to da Fisherova jezgra predstavlja gradijent logaritma izglednosti, Fisherov vektor slike ili okna može se prikazati kao prosječna vrijednost Fisherovih vektora njezinih lokalnih opisnika $\mathbf{x}_t$ [23]:

$$\boldsymbol{\Phi}_{\boldsymbol{\theta}}(\mathbf{X}) = \frac{1}{T} \sum_{t=1}^{T} \phi_{\boldsymbol{\theta}}(\mathbf{x}_t) \tag{2.25}$$

Opisano svojstvo Fisherovog vektora naziva se aditivnost i ima važnu ulogu prilikom lokalizacije. Osim aditivnosti Fisherov vektor ima sljedeća važna svojstva [29]

– očekivanje Fisherovog vektora jednako je nul-vektoru:

$$\mathbf{E}_{\mathbf{X}\ p(\mathbf{x};\boldsymbol{\theta})}[\phi(\mathbf{x}; \boldsymbol{\theta})] = \overrightarrow{0} \tag{2.26}$$

– očekivanje kovarijance je jednako jediničnoj matrici $\mathbf{I}$:

$$\mathbf{E}_{\mathbf{X}\ p(\mathbf{x};\boldsymbol{\theta})}[\phi(\mathbf{x}; \boldsymbol{\theta})\phi(\mathbf{x}; \boldsymbol{\theta})^T] = \mathbf{I} \tag{2.27}$$

i dva vrlo važna svojstva u kontekstu ovog rada:

– kodiranjem Fisherovim vektorom poništava se utjecaj pozadinske informacije neovisne o slici

– kodiranjem Fisherovom jezgrom obavlja se nelinearna transformacija zbog čega su Fisherovi vektori pogodni za linearnu klasifikaciju

2.5. Poboljšanje reprezentacije Fisherovim vektorima

U literaturi se mogu pronaći načini poboljšanja reprezentacije Fisherovim vektorima. Normalizacijom potenciranjem umanjuje se utjecaj opisnika dodijeljenih karakterističnim slikovim riječima koji se natprosječno često pojavljuju u slici (engl. *bursty visual features*) [22]. Normalizacija potenciranjem obavlja se za svaki element Fisherovog

vektora X_d, $d \in \{1 \dots K(2D+1)\}$, gdje je D dimenzija lokalnog opisnika, a K broj komponenti mješavine Gaussovih razdiobi:

$$s(X_d) = sign(X_d)|X_d|^\rho \tag{2.28}$$

gdje je $0 < \rho < 1$

Obično se kao parametar ρ uzima vrijednost 0.5 čime izraz (2.28) postaje predznačeno korjenovanje (engl. *signed square routing*). Normalizacija potenciranjem pokušava smanjiti utjecaj moguće zavisnosti lokalnih opisnika. U kontekstu Fisherovih vektora njenom primjenom povećava se apsolutna vrijednost pojedinog elementa vektora X_d. Time se postiže manji stupanj rijetkosti Fisherovog vektora čineći ga pogodnijim za uspoređivanje skalarnim produktom u okviru jezgrene funkcije
$K(\mathbf{X}, \mathbf{X}') =< s(\mathbf{X}, s(\mathbf{X}') >$.

Fisherovim vektorom poništava se utjecaj pozadine no slike mogu sadržavati objekt u različitim mjerilima i neće imati jednaki Fisherov potpis [29]. Razlika između reprezentacija primjeraka koje sadrže objekt u različitim veličinama umanjuje se metričkom normalizacijom. Globalna metrička normalizacija projicira Fisherov vektor na jediničnu sferu dijeleći ga s $\sqrt{n(\mathbf{X})}$, gdje je $n(\mathbf{X})$:

$$n(\mathbf{X}) = s(\mathbf{X})^T s(\mathbf{X}) = \sum_{d=1}^{K(2D+1)} s(X_d)^p \tag{2.29}$$

Obično se kao parametar p uzima vrijednost 2 čime izraz (2.29) postaje l_2 norma izraza (2.28).

Osim globalne metričke normalizacije postoji i metrička normalizacija na razini GMM komponente. Pokazano je da komponente Fisherovog vektora $\mathbf{X}_k \in \mathbb{R}^{2D+1}$ čine gradijenti (2.22, 2.23, 2.24). Normalizacija po komponenti slična je normalizaciji cijelog vektora. Metrička normalizacija se tako dobiva projiciranjem pojedine komponente na jediničnu sferu:

$$n(\mathbf{X}_k) = \sum_{d=(k-1)(2D+1)}^{k(2D+1)} s(X_d)^2 \tag{2.30}$$

Po komponentama l_2 normalizirani Fisherov vektor je konkatenacija normalizacija njegovih komponenti. Konačno se vektor podijeli s korijenom broja komponenti kako bi se osigurala jedinična norma cijelog vektora.

Normalizacijom po komponentama umanjuje se mogućnost dominacije pojedinih komponenti koje daju natprosječno velik doprinos Fisherovom vektoru. Takav slučaj naziva se efektom eksplozije slikovne riječi (engl. *burstiness effect*) [16].

3. Slabo nadzirana semantička segmentacija

U ovom poglavlju opisuje se model lokalizacije objekta koji koristi Fisherove vektore slikovnih okana primjeraka.

3.1. Uzorkovanje slikovnih okana

Prije učenja lokalizacijskog modela potrebno je odabrati skup slikovnih okana koji predstavljaju relevantnu informaciju o slici. U literaturi se najčešće spominju sljedeći tipovi uzorkovanja: uzorkovanje interesnim točkama i gusto uzorkovanje. Uzorkovanjem interesnim točkama nastoji se pronaći skup lokacija koje su invarijantne na skaliranje, translaciju i rotaciju odnosno promjenu perspektive i na osvjetljenje u slici. Gustim uzorkovanjem nastoji se obradom svih piksela izdvojiti razmjerno malena i međusobno preklapajuća okna na različitim mjerilima (engl. *multi-scale dense grid*). U literaturi je pokazano da se gustim uzorkovanjem dobivaju bolji rezultati klasifikacije slika i lokalizacije objekata no kako bi se identificirala slična okna u različitim slikama, potrebno je primijeniti razmjerno veliku frekvenciju uzorkovanja što povećava računalnu složenost procesa.

U [29] navedeni su neki primjeri algoritama uzorkovanja interesnim točkama:

- Harrisov algoritam neosjetljiv na promjene u mjerilu i na afine transformacije: proširenje originalnog Harrisovog algoritma detekcije kutova koji se temelji na autokorelacijskoj matrici i tzv. mjeri kutnosti značajke (engl. *cornerness*)
- algoritam pronalaženja lokalnih ekstrema u piramidi razlika Gaussovih filtara (DoG): pronalazi interesne točke neosjetljive na promjene u mjerilu
- algoritam pronalaženja regija najveće stabilnosti (MSER, engl. Maximally Stable Extremal Regions): pronalazi povezane komponente nad sivom (engl. *grayscale*) slikom čije su vrijednosti određene nekim pragom (engl. *threshold*)

U ovom radu koristili su se primjerci koji su anotirani na razini pravokutnih okvira (engl. *bounding box*) znamenki i slova (detalji se mogu pronaći u poglavlju 4). Svaka slika primjerka podijelila se na sadržaj koji se nalazi unutar okvira i na sadržaj koji se nalazi izvan okvira. Tako su se za svaki primjerak dobila dva slikovna okna:

- pozitivno slikovno okno: veliku većinu čine pikseli rukom napisane jednadžbe, a ostatak pikseli pozadine
- negativno slikovno okno: čine ga pikseli pozadine

3.2. Učenje lokalizacijskog modela

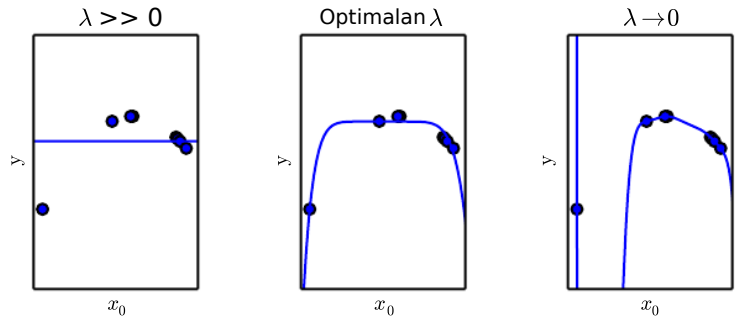
U prethodnim poglavljima objašnjeno je kako izvući lokalne opisnike primjeraka (2.1) i pomoću njih stvoriti generativni model kojeg čini mješavina Gaussovih razdiobi (GMM, poglavlje 2.2). Pomoću tog modela lokalni opisnici slikovnih okana kodiraju se u odgovarajuće Fisherove vektore. U prethodnom potpoglavlju objašnjeno je da se svaki primjerak sastoji od pozitivnog i negativnog slikovnog okna. Vektori pojedinog okna sažimaju se usrednjavanjem (engl. *sum pooling*) prema izrazu (2.25) u jedinstven vektor. Zbog toga se svaki primjerak sastoji od pozitivnog $y_i = 1$ i negativnog $y_i = -1$ Fisherovog vektora. Dobiveni Fisherovi vektori ulazi su algoritma učenja linearnog lokalizacijskog modela. U potpogavlju 2.5 opisana su neka moguća poboljšanja Fisherovog vektora kao reprezentacije primjerka. Kako bi se poboljšala generalizacija modela odnosno smanjila prenaučenost moguće je koristiti regularizaciju. Slika 3.3 prikazuje postupak učenja s korištenjem $K = 4$ komponenti.

3.2.1. Regularizacija

Odabirom algoritma učenja modela biraju se tipovi mogućih hipoteza. Tako na primjer ako se koristi algoritam linearne regresije prostor hipoteza (engl. *hypothesis space*) čine linearne funkcije. Povećanje ili smanjivanje kapaciteta modela može se postići dodavanjem ili micanjem funkcija iz prostora hipoteza koje algoritam učenja može koristiti. Moguće je uvesti preferiranje onih hipoteza koje bolje odgovaraju skupu za učenje odnosno njihovim korištenjem dobiva se manja pogreška klasifikacije na skupu za učenje. Na primjeru linearne regresije, u funkciju gubitka može se uvesti propadanje težina (engl. *weight decay*):

$$J(\mathbf{w}) = MSE_{train} + \lambda \mathbf{w}^T \mathbf{w} \tag{3.1}$$

gdje je MSE_{train} pogreška srednjih kvadrata (MSE, engl. *Mean Squared Error*) na skupu za učenje, a λ parametar kojim se regulira jača ili manja preferencija.



Slika 3.1: Primjer efekta korištenja različitih vrijednosti parametra λ. Slika je preuzeta iz [19]

U ovom primjeru preferiraju se težine koje imaju manju l_2 normu. Minimiziranjem $J(\mathbf{w})$ odabiru se težine koje su prikladne za skup podataka za učenje, a usto su i male. Time se postiže umjerena prilagodba skupu učenja stavljajući naglasak na manji broj značajki. Primjer korištenja manjeg ili većeg parametra λ može se vidjeti na modelu regresije s polinomnom funkcijom (Slika 3.1). Vrlo veliki λ uzrokuje nepostojeću prilagodbu modela. Naučena funkcija je konstanta i model je podnaučen (engl. *underfitted model*). S malim $\lambda \rightarrow 0$ model se previše prilagođava skupu podataka za učenje odnosno mala je vjerojatnost dobre generalizacije. Takav model nazivamo prenaučenim modelom (engl. *overfitted model*). S optimalnim λ funkcija regresije se prilagodila skupu za učenje i sadrži malu varijancu (engl. *variance*). Iako model ima kapacitet za korištenje kompleksnijih funkcija, zbog preferencija se odabrala jednostavnija funkcija od koje se očekuje bolja generalizacija.

Uvođenje preferencije za manjim težinama jedan je od brojnih primjera regularizacije. Regularizacija (engl. *regularization*) je svaka modifikacija algoritma učenja koja ima za cilj smanjiti pogrešku generalizacije. U kontekstu regularizacije parametar λ nazivamo regularizacijskim faktorom i smatramo ga hiperparametrom. Odabir prikladne regularizacijske funkcije i regularizacijskog faktora ovisi o pojedinom problemu i općenito se smatra jednim od središnjih problema područja strojnog učenja [19]. U dubokom učenju obično se uzima neka općenita regularizacijska funkcija uz fino prilagođavanje regularizacijskog faktora nekim postupkom pronalaska hiperparametra kao što je to unakrsna validacija (engl. *cross-validation*). U tom postupku se manji dio skupa podataka za učenje izuzima i služi za validaciju korištenih parametara. Prona-

đeni optimalni parametri zatim se koriste na skupu za ispitivanje.

U kontekstu ovog rada potrebno je koristiti regularizaciju kojom se inducira rijetkost modela. Slijedi pregled nekih regularizacijkih funkcija.

l_2 regularizacija

l_2 regularizacija, poznata još kao Tikhonova regularizacija, definirana je sljedećim izrazom:

$$l_2(\mathbf{w}) = \|\mathbf{w}\|_2^2 = \sum_i w_i^2 = \mathbf{w}^T\mathbf{w} \tag{3.2}$$

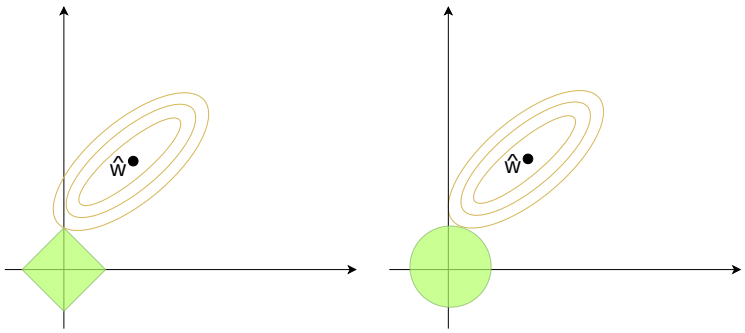
Minimizacijom izraza (3.2) kažnjava se suma veličina koeficijenata vektora težina $\mathbf{w}$ te se time dobiva model koji preferira manje iznose koeficijenata.

l_1 regularizacija

l_1 regularizacija, poznata još kao operator apsolutnog smanjivanja i odabira (LASSO, engl. *Least Absolute Shrinkage and Selection Operator*), definirana je sljedećim izrazom:

$$l_1(\mathbf{w}) = \|\mathbf{w}\|_1 = \sum_i |w_i| \tag{3.3}$$

Razlika između l_1 i l_2 regularizacije može se vidjeti na Slici 3.2.



Slika 3.2: Prikaz l_1 regularizacije (lijevo) i l_2 regularizacije (desno)

Problem minimizacije regularizirane funkcije gubitka $\min_{\mathbf{w}} J(\mathbf{w}) + \lambda\|\mathbf{w}\|_1$ može se zapisati u obliku optimizacijske funkcije s linearnim ograničenjima $\min_{\mathbf{w}} J(\mathbf{w})$ tako da vrijedi $\|\mathbf{w}\|_1 < B$, gdje B označava gornju vrijednost koeficijenata [29]. Analogno

vrijedi i za l_2 regularizaciju $\|\mathbf{w}\|_2 < B$. Optimalno rješenje navedenih problema dobiva se u prvoj točki gdje se plohe funkcije gubitka i funkcije ograničenja dodiruju. Kutovi plohe ograničenja odgovaraju rješenjima na koordinatnim osima gdje neka od značajki poprima vrijednost ništice. U slučaju l_1 regularizacije kutovi plohe ograničenja su točke presjecišta. U slučaju l_2 regularizacije ploha ograničenja odgovara kružnici koja može dodirnuti plohu optimizacijske funkcije u bilo kojoj točki čime se uklanja prednost rijetkih rješenja kao u slučaju l_1 regularizacije.

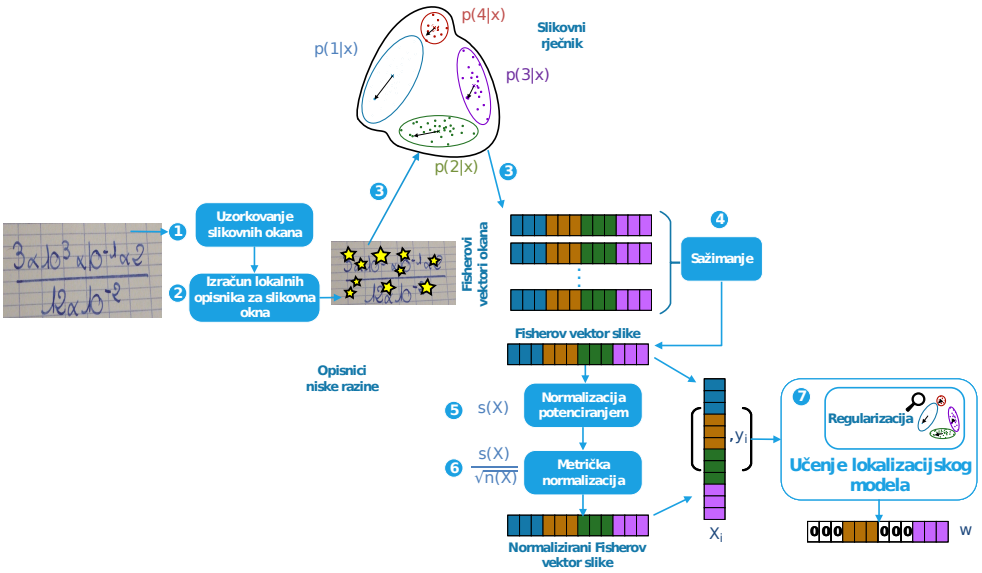
$l_{2,1}$ regularizacija

Rijetka $l_{2,1}$ regularizacija podrazumijeva strukturu podataka koji se sastoji od nezavisnih dijelova odnosno komponenti što je posebno bitno u kontekstu ovog rada i Fisherovih vektora. Definira se na sljedeći način:

$$l_{2,1}(\mathbf{w}) = \sum_{k=1}^{K} \|\mathbf{w}_k\|_2 \tag{3.4}$$

gdje je $\mathbf{w}_k$ dio vektora težina koji odgovara komponenti $k \in \{1 \dots K\}$.

Primjenom l_2 regularizacije unutar doprinosa pojedine komponente $\mathbf{w}_k$ postiže se bolja generalizacija, a l_1 regularizacijom između komponenti postiže se postavljanje koeficijenata nediskriminativnih komponenti $\mathbf{w}_k$ u ništicu [29].



Slika 3.3: Postupak učenja modela. Slika je preuzeta iz [29] i zatim prilagođena ovom radu.

3.3. Lokalizacija objekata

Naučeni linearni model $\mathbf{w}$ koristi se za lokalizaciju objekata. Kao i u postupku učenja moguće je uzorkovati slikovna okna primjeraka. Lokalni opisnici pojedinih slikovnih okana kodiraju se u odgovarajuće Fisherove vektore pomoću GMM-a naučenog prilikom faze učenja. Kako bi se izračunao odziv pojedinog okna obavlja se operacija skalarnog produkta u odnosu na model $\mathbf{w}$ ili gradijent odziva normalizirane slike $\nabla_{\mathbf{x}} f(\mathbf{X})$ (opisan u sljedećem potpoglavlju). Pozitivna okna $f(\mathbf{x}_i) > 0$ čine područje interesa odnosno sadrže traženi objekt koji je u ovom radu rukom napisana jednadžba. Slika 3.4 prikazuje postupak lokalizacije objekta u kojem se koriste $K = 4$ komponenti.

3.4. Optimizacija postupka

Jedan od načina kako postupak ispitivanja učiniti efikasnijim je proračun vjerojatnosti $p(k|\mathbf{x})$ (2.21) kojom je opisnik pojedinog slikovnog okna $\mathbf{x}$ dodijeljen GMM komponenti k. Slikovna okna $\mathbf{x}$ koja imaju malu vjerojatnost pridruživanja u odnosu na diskriminativne slikovne komponente smatraju se nebitnim i za njih se ne računaju Fisherovi vektori niti njihov odziv. U praksi se koristi uvjet $p(k|\mathbf{x}) > 1/K$ gdje K označava broj slikovnih riječi.

U [29] navodi se efikasan proračun odziv slikovnih okana primjenom rijetkog modela. Notacija $\mathbf{X}$ koristi se za Fisherov vektor slike, a $\mathbf{x}$ za Fisherov vektor slikovnog okna.

U ovom radu razmatra se slučaj binarne klasifikacije što znači da $f(\mathbf{x})$ označava prisutnost traženog objekta u slikovnom oknu. Efikasan proračun ovisi o činjenici je li model učen normaliziranim ili nenormaliziranim Fisherovim vektorima.

Slučaj nenormaliziranih vektora je jednostavan jer za izračun odziva potrebno je samo izračunati skalarni produkt vektora težina modela $\mathbf{w}$ i Fisherovog vektora:

$$f_{lin}(\mathbf{X}) = \mathbf{w}^T \cdot \mathbf{X} \tag{3.5}$$

$$f_{lin}(\mathbf{x}) = \mathbf{w}^T \cdot \mathbf{x} \tag{3.6}$$

Pri čemu se klasifikacijski odziv slike može prikazati i kao suma odziva okana $\mathbf{x}_i$.

Slučaj normaliziranih vektora je nešto složeniji. Klasifikacijski odziv slike $\mathbf{X}$

odgovara skalarnom produktu težina modela $\mathbf{w}$ i nelinearne normalizacije:

$$f_{norm}(\mathbf{X}) = \mathbf{w}^T \cdot \frac{s(\mathbf{X})}{\sqrt{n(\mathbf{X})}} \tag{3.7}$$

Zbog navedene nelinearne normalizacije svojstvo aditivnosti više ne vrijedi, odnosno $\mathbf{w}^T \cdot s(\mathbf{X})/\sqrt{n(\mathbf{X})} \neq \sum_i \mathbf{w}^T \cdot s(\mathbf{x}_i)/\sqrt{n(\mathbf{x}_i)}$. Problemu se može pristupiti sljedećim izrazom:

$$f_{norm}(\mathbf{x}) = f_{norm}(\mathbf{X}) - f_{norm}(\mathbf{X} - \mathbf{x}) \tag{3.8}$$

no takvo rješenje je prilično neefikasno jer potrebno je oduzeti Fisherov vektor slikovnog okna od Fisherovog vektora slike, primijeniti normalizaciju i zatim izračunati skalarni produkt s modelom, a slika može sadržavati potencijalno veliki broj slikovnih okana. U [29] navodi se efikasnije rješenje pomoću aproksimacije prvog reda.

Gradijent odziva normalizirane slike

Odziv slikovnog okna može se prikazati aproksimacijom prvog reda razvoja funkcije f_{norm} u Taylorov red oko vektora $\mathbf{X}$:

$$f_{norm}(\mathbf{X} + \mathbf{x}) \approx f_{norm}(\mathbf{X}) + \nabla_{\mathbf{x}} f_{norm}(\mathbf{X})^T \cdot \mathbf{x} \tag{3.9}$$

gdje $\nabla_{\mathbf{x}} f_{norm}(\mathbf{X})$ predstavlja gradijent odziva normalizirane slike $f_{norm}(\mathbf{X})$ u odnosu na nenormalizirani Fisherov vektor slikovnog okna $\mathbf{x}$. Doprinos samog okna $\mathbf{x}$ može se prikazati kao:

$$f_{grad}(\mathbf{x}) \approx f_{norm}(\mathbf{X} + \mathbf{x}) - f_{norm}(\mathbf{X}) \approx \nabla_{\mathbf{x}} f_{norm}(\mathbf{X})^T \cdot \mathbf{x} \tag{3.10}$$

Vektor gradijenta za pojedinu dimenziju d računa se sljedećim izrazom:

$$\frac{\partial f_{norm}\mathbf{X}}{\partial x_d} = \frac{\partial f_{norm}(\mathbf{X})}{\partial \mathbf{X}} \cdot \frac{\partial \mathbf{X}}{\partial x_d} \tag{3.11}$$

S obzirom da je derivacija nenormaliziranog Fisherovog vektora slike $\mathbf{X}$ jednaka ništici, osim elemenata na d-toj poziciji koji su jednaki jedinici, gradijent se može zapisati kao:

$$\frac{\partial f_{norm}\mathbf{X}}{\partial x_d} = \frac{\partial f_{norm}(\mathbf{X})}{\partial X_d} \tag{3.12}$$

Za izvod derivacije $\frac{\partial f_{norm}\mathbf{X}}{\partial x_d}$ potrebno je izračunati derivaciju normalizacije potenciranjem i metričke normalizacije:

$$\frac{\partial s(\mathbf{X})}{\partial X_d} = [0 \ldots \rho |X_d|^{\rho-1} \ldots 0] \tag{3.13}$$

$$\frac{\partial n(\mathbf{X})}{\partial X_d} = 2s(\mathbf{X})\frac{\partial s(\mathbf{X})}{\partial X_d} = 2s(X_d)\rho|X_d|^{\rho-1} \tag{3.14}$$

Izvod normalizacije po komponentama analogan je izrazima za globalnu normalizaciju, gdje se izraz $n(\mathbf{X})$ supstituira sa $n(\mathbf{X}_k)$, a izraz $s(\mathbf{X})$ sa $s(\mathbf{X}_k)$ gdje $\mathbf{X}_k$ odgovara dijelu Fisherovog vektora dobivenog k-tom komponentom GMM-a. Derivacija $\frac{\partial f(\mathbf{X})}{\partial X_d}$ može se izvesti na sljedeći način:

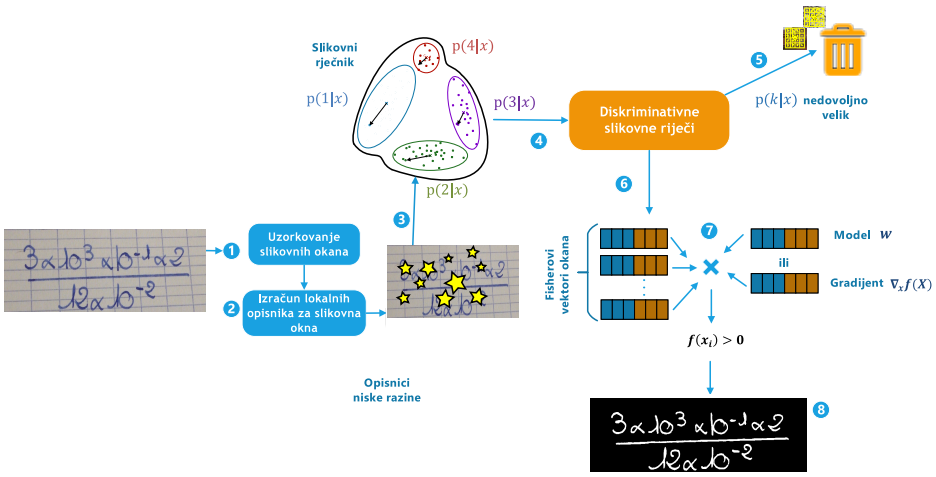
$$\begin{aligned}
\frac{\partial f(\mathbf{X})}{\partial X_d} &= \frac{\partial \mathbf{w}^T s(\mathbf{X})/\sqrt{n(\mathbf{X})}}{\partial X_d} \\
&= \frac{1}{\sqrt{n(\mathbf{X})}}\frac{\partial \sum_d w_d \cdot s(X_d)}{\partial X_d} + \mathbf{w}^T s(\mathbf{X})\frac{\partial [n(\mathbf{X})]^{-0.5}}{\partial X_d} \\
&= \frac{w_d}{\sqrt{n(\mathbf{X})}}\frac{\partial s(X_d)}{\partial X_d} - 0.5 \cdot \frac{\mathbf{w}^T s(\mathbf{X})}{[n(\mathbf{X})]^{1.5}}\frac{\partial n(\mathbf{X})}{\partial X_d}
\end{aligned} \tag{3.15}$$

Primjenom izraza (3.13) i (3.14) u (3.15) dobiva se sljedeći izvod:

$$\begin{aligned}
\frac{\partial f(\mathbf{X})}{\partial X_d} &= \frac{w_d}{\sqrt{n(\mathbf{X})}}\rho|X_d|^{\rho-1} - 0.5 \cdot \frac{\mathbf{w}^T s(\mathbf{X})}{[n(\mathbf{X})]^{1.5}} \cdot 2s(X_d) \cdot \rho|X_d|^{\rho-1} \\
&= \frac{\rho|X_d|^{\rho-1}}{\sqrt{n(\mathbf{X})}}\left(w_d - \frac{\mathbf{w}^T s(\mathbf{X})}{n(\mathbf{X})} \cdot s(X_d)\right) \\
&= \frac{\rho|X_d|^{\rho-1}}{\sqrt{n(\mathbf{X})}}\left(w_d - \frac{f(\mathbf{X})s(X_d)}{\sqrt{n(\mathbf{X})}}\right)
\end{aligned} \tag{3.16}$$

Potrebno je pripaziti na slučaj kad je element vektora $X_d = 0$, zbog $|X_d|^{\rho-1} = \frac{1}{\sqrt{|X_d|}}$ (uz pretpostavku da je $\rho = 0.5$) izraz (3.16) postaje nedefiniran. To je izniman slučaj jer smatra se da su Fisherovi vektori gusti jer su nastali sažimanjem operacijom usrednjavanja. U takvim slučajevima na poziciji d u vektoru gradijenta postavlja se vrijednost ništice. U slučaju normalizacije po komponentama, odziv se računa po komponentama i zatim zbraja:

$$f_{norm}(\mathbf{X}) = \sum_k f_{norm}(\mathbf{X}_k) \tag{3.17}$$



Slika 3.4: Postupak lokalizacije objekta. Slika je preuzeta iz [29] i zatim prilagođena ovom radu.

4. Skup podataka za učenje i ispitivanje

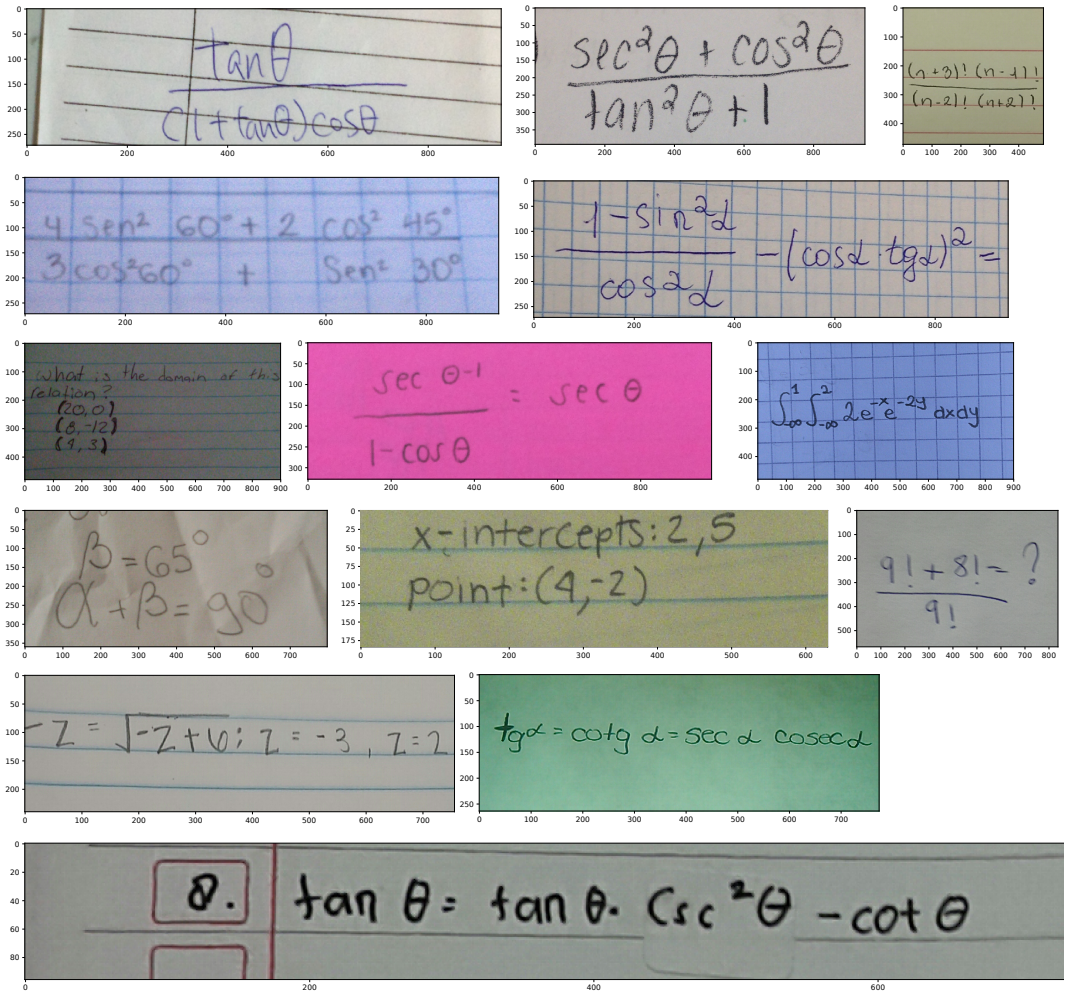
Skup podataka čine primjerci (engl. *samples*). Svaki primjerak sastoji se od originalne slike, lokalnih opisnika dobivenih konvolucijskom neuronskom mrežom i anotacije na razini okvira ili piksela. U ovom radu dostupno je bilo ukupno 1099 primjeraka od čega njih 99 je sadržavalo anotacije na razini piksela. Slijedi kratak opis dijelova primjeraka.

4.1. Slike primjeraka

Slike primjeraka su bile pohranjene u obliku JPEG [1] datoteka. Veliku većinu slika činile su RGB [2] slike. Slike su bile različitih rezolucija (od 96×486 do 728×1528) i sadržavale različite pozadine, boje i stilove rukopisa. Na Slici 4.1 mogu se vidjeti neki primjeri slika korištenih primjeraka.

[1] JPEG (engl. *Joint Photographic Experts Group*) je popularan slikovni format koji koristi algoritam kompresije kako bi dobio zadovoljavajuću veličinu sliku s minimalnim gubicima.

[2] RGB model (engl. *Red Green Blue model*) je model boja u kojem se miješaju 3 razine (od 0 do 255) crvene, zelene i plave boje u različitom omjeru kako bi se postigao širok raspon boje.

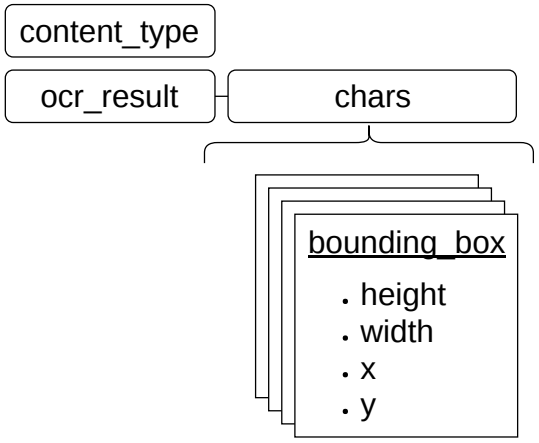


Slika 4.1: Primjeri slika iz skupa podataka i njihove dimenzije

4.2. Anotacije primjeraka

Anotacije primjeraka na razini okvira

Svaka korištena slika ima i odgovarajuću datoteku anotacije spremljenu u JSON [3] formatu. Na Slici 4.2 se može vidjeti koncept relevantnih dijelova i primjer anotacije.

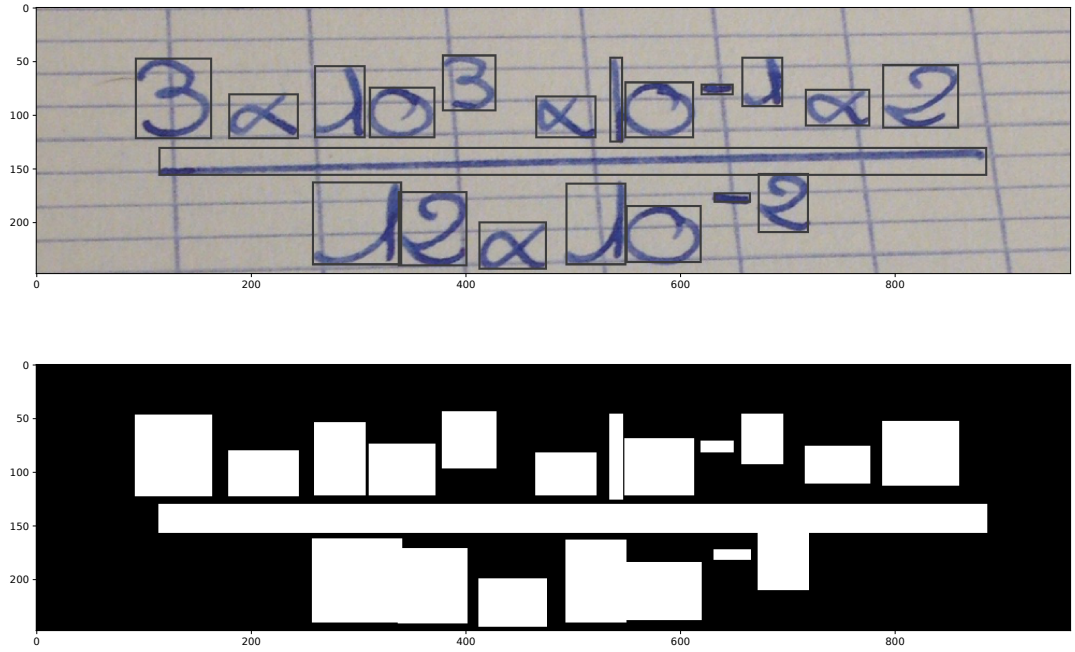



```
content_type: equation
ocr_result
    chars
        bounding_box
            height: 32.43
            width: 31.0
            x: 50.0
            y: 44.57
        bounding_box
            height: 21.0
            width: 33.0
            x: 88.0
            y: 57.0
        ...
```

Slika 4.2: Koncept anotacije i dio primjera anotacije jednog primjerka (prikazani su samo relevantni detalji)

Svaka anotacija sadrži koordinate vrha, dužinu i širinu pravokutnog okvira (engl. *bounding box*) dijela jednadžbe. Sve što je unutar okvira smatra se pozitivnim, a izvan njega negativnim dijelom slike što se može vidjeti na Slici 4.3. U programskoj implementaciji koristi se u obliku matrice istinitosti (engl. *boolean matrix*).

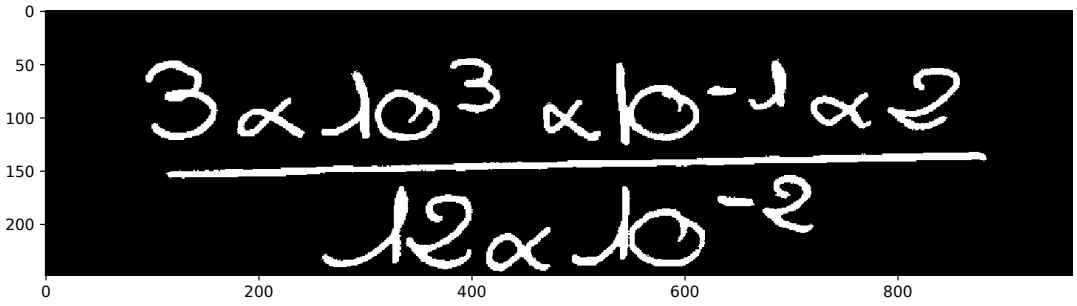
[3] JSON (engl. *JavaScript Object Notation*) je popularan format zapisa podataka u obliku ključ-vrijednost. Vrijednost može biti broj, niz slova, istinitost ili polje vrijednosti)



Slika 4.3: Primjer anotirane slike na razini okvira i podjela slike na pozitivni (bijela boja) i negativni dio (crna boja)

Anotacije primjeraka na razini piksela

Za manji dio primjeraka dostupna je anotacija primjeraka na razini piksela koja je poslužila za konačnu evaluaciju postupka koji je opisan u poglavlju 6. Anotacije su dobivene kao rezultat postupka opisanog u [1] i zatim manje ručne obrade. Pohranjena je kao binarna slika koja ima dvije moguće razine piksela: 0 (minimalna razina) i 255 (maksimalna razina). U programskoj implementaciji koristi se u obliku matrice istinitosti na način da se prisustvom piksela jednadžbe smatra piksel maksimalne razine, a pozadine piksel minimalne razine.



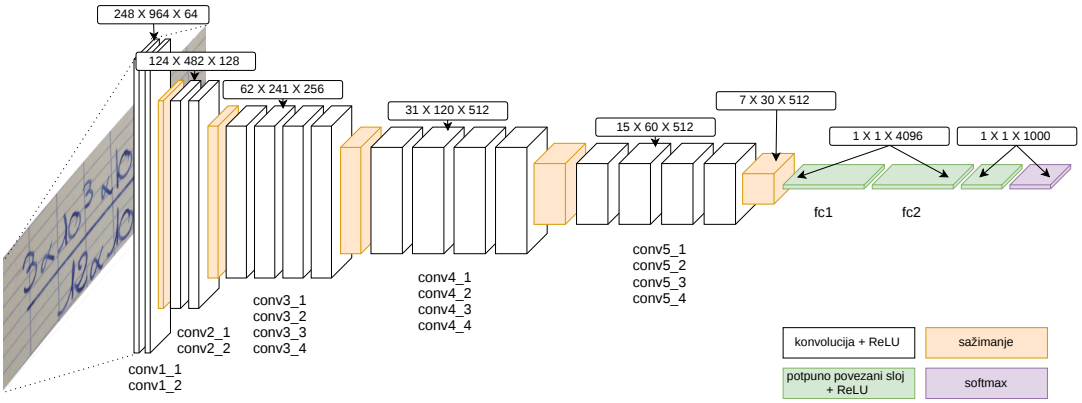
Slika 4.4: Primjer anotirane slike na razini piksela i podjela slike na pozitivni (bijela boja) i negativni dio (crna boja)

4.3. Konvolucijske značajke primjeraka

Kao lokalni opisnici primjeraka koriste se konvolucijske značajke dobivene predtreniranom konvolucijskom neuronskom mrežom (KNN). U ovom radu za tu svrhu su se koristila dva KNN-a. Slijedi njihov opis.

4.3.1. Značajke dobivene VGG19 KNN-om

U ovom slučaju koriste se lokalni opisnici primjeraka dobivenih pomoću VGG19 predtrenirane konvolucijske neuronske mreže (CNN, *Convolutional Neural Network*) nad skupom podataka ImageNet [4] [6]. VGG19, još poznata kao VGG-VD (engl. *very deep*) mreža, je inačica VGG [5] arhitekture mreže. Broj 19 se odnosi na broj njezinih slojeva koji su prikazani na Slici 4.5 zajedno s njihovim dimenzijama za primjer sa Slike 4.5.



Slika 4.5: Arhitektura konvolucijske neuronske mreže VGG19 s dimenzijama svakog sloja za primjer sa Slike 4.3

Za potrebe dobivanja konvolucijskih značajki koristili su se slojevi do i uključujući sloja `conv2_2`. Time se za svaki primjerak dobio tenzor dimenzija $(\frac{H}{2} \times \frac{W}{2} \times 128)$, gdje je H visina, a W širina slike primjerka. Svaki od 128 slojeva se može i prikazati čime se dobiva predodžba o reakciji sloja na korištenu sliku. Na Slici 4.7 mogu se

[4] ImageNet je skup podataka slika koji je organiziran prema WordNet hijerarhiji. Svaki koncept u WordNet-u koji je opisan s više riječi ili fraza naziva se skupom sinonima (engl. *synset*, *synonim set*). U ImageNetu se nalazi preko 100 000 takvih skupova od čega su većina njih imenice. Cilj je imati u prosjeku 1000 slika po svakom skupu sinonima odnosno ostvariti vrlo veliki skup podataka. Svaka slika je ručno provjerena i anotirana.

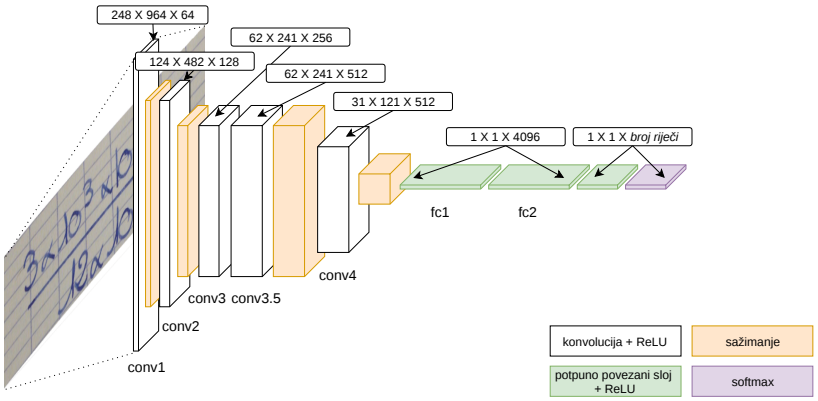
[5] VGG je akronim grupe *Visual Geometry Group* sa sveučilišta Oxford koja je razvila duboku arhitekturu konvolucijske neuronske mreže i naučila je skupom podataka ImageNet.

vidjeti nasumično odabrani slojevi tenzora dobivenog konvolucijom primjerka iz Slike 4.3. Svaki od 128 slojeva daje određenu informaciju i zajedno se koriste kao reprezentacija primjerka. Konvolucijske značajke su vektori $\mathbf{x_{ij}} \in \mathbb{R}^{128}$ koji su nastali konkatenacijom vrijednosti slojeva na poziciji (i, j), gdje su $i = 1, 2, \ldots, \frac{H}{2}$, $j = 1, 2, \ldots, \frac{W}{2}$. Dobiveni vektori značajki se centriraju oduzimanjem tzv. VGG19 srednjeg piksela koji se sastoji od vrijednosti $[103.939, 116.779, 123.68]$. VGG19 srednji piksel se u literaturi navodi kao srednja vrijednost korištenog skupa podataka za učenje VGG mreže.

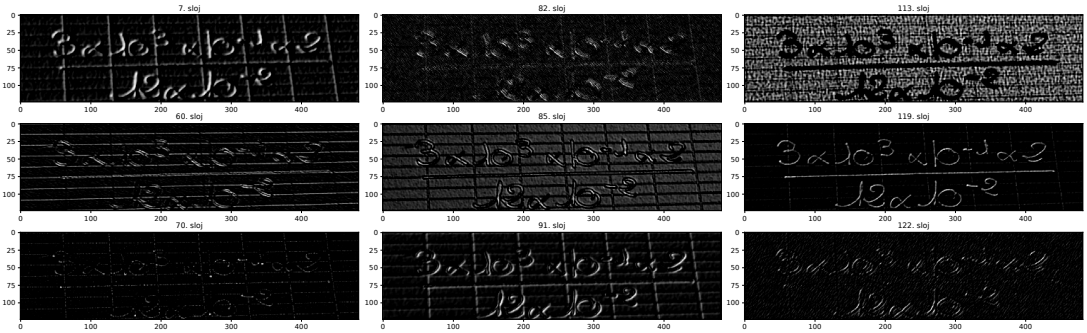
4.3.2. Značajke dobivene CharNet KNN-om

U ovom slučaju koriste se lokalni opisnici primjeraka dobivenih pomoću CharNet predtrenirane konvolucijske neuronske mreže nad skupom podataka MJSynth [6] Arhitektura CharNet (Slika 4.6) je slična arhitekturi VGG uz neke bitne razlike kao što je korištenje jednog umjesto više konvolucijskih blokova po konvoluciji uz iznimku u trećem konvolucijskom nizu koji se sastoji od dva različita kovolucijska bloka. Zbog izostanka sloja sažimanja između ta dva bloka, autori [15] ga smatraju prijelaznim blokom (`conv3.5`). CharNet mreža je naučena na skupu podataka koji se sastoji od sivih (engl. *grayscale*) slika. Kao ulaz u mrežu koristile su se slike primjeraka pretvorene u njihovu sivu verziju. Za potrebe rada koristile su se konvolucijske značajke dobivene slojem `conv2`. Dobiveni vektori se centriraju oduzimanjem srednje vrijednosti ulazne slike i dijeljenjem razlike sa standardnom devijacijom ulazne slike.

[6] MJSynth je skup podataka koji se sastoji od 9 milijuna umjetno stvorenih sivih (engl. *grayscale*) slika znakova. Slike su se stvarale na način da se najprije izradi slika računalno pisanih znakova. Tim se znakovima dodaju boje i sjene. Nakon toga se nad njima izvršava neka transformacija u drugu perspektivu uz dodavanje izobličenja. Konačno se slika stapa s nekim slikama pozadine kako bi se dobila realistična slika [14].



Slika 4.6: Arhitektura konvolucijske neuronske mreže CharNet s dimenzijama svakog sloja za primjer sa Slike 4.3



Slika 4.7: Nekoliko nasumično odabranih slojeva VGG19 konvolucijskih značajki primjerka iz Slike 4.3

5. Programska implementacija i korištene biblioteke

Programski jezik implementacije je Python (verzija `2.7.13` se koristila za implementaciju opisanih postupaka, a verzija `3.6.1` za implementaciju ekstrakcije konvolucijskih značajki). Python je popularan interpretirani, objektno-orijentirani, skriptni jezik visoke razine. Omogućava brzo razvijanje programskog okruženja i spajanje postojećih komponenti napisanih u drugim jezicima (obično niže razine koji omogućavaju mnogo brže izvođenje). Sintaksa Pythona je jednostavna s naglaskom na čitljivost i sažetost programskog koda. Sadrži veliko mnoštvo standardnih biblioteka i podržava uključivanje vanjskih biblioteka.

U nastavku slijede opisi korištenih vanjskih biblioteka, a zatim i opis programske implementacije.

5.1. Vanjske biblioteke

5.1.1. NumPy

NumPy (engl. *Numerical Python*) [28] je Python biblioteka otvorenog koda koja omogućava modeliranje multidimenzionalnih polja i matrica. Sadrži mnoštvo ugrađenih optimiziranih operacija za uporabu s NumPy poljima (engl. *arrays*) i matricama

(engl. *matrices*): matematička i logička manipulacija, mijenjanje oblika, sortiranje, odabir i postavljanje vrijednosti, Fourierove transformacije, nasumična simulacija i mnoge druge.

Glavne razlike između standardnih Python nizova i NumPy polja [24]:

- NumPy polja imaju fiksnu veličinu prilikom stvaranja za razliku od standardnih dinamičkih lista. Mijenjanje veličina polja iziskuje stvaranje novog i brisanje originalnog polja.
- Svi tipovi elemenata NumPy polja moraju biti jednakog tipa čime se zauzima jednako mjesta u memoriji. Iznimku čine polja s objektom (Python ili NumPy) kao tipom podataka koji mogu biti različitih veličina.
- NumPy polja su prilagođena za rad s velikom količinom podataka i naprednim matematičkim operacijama koje se efikasno izvode.
- Mnoge druge vanjske biblioteke sa znanstvenom i matematičkom namjenom koriste NumPy polja kao osnovne objekte.

Verzija korištene NumPy biblioteke: `1.12.1`.

5.1.2. TensorFlow

TensorFlow je Python biblioteka otvorenog koda koja omogućava izgradnju neuronskih mreža velikih dubina i njihovo pokretanje pomoću procesora ili grafičke jedinice. Nastao je u Googleu (grupa Google Brain) na temelju njihove interne biblioteke DistBelief. Prva službena verzija pod otvorenom licencom je postala dostupna u studenom 2015. godine. U ovom radu koristila se za modeliranje konvolucijske neuronske mreže s korištenim predtreniranim parametrima i dobivanje konvolucijskih značajki. Verzija korištene TensorFlow biblioteke: `1.1.0`.

5.1.3. Matplotlib

Matplotlib (engl. *Mathematical plotting library*) [12] je Python biblioteka koja omogućava iscrtavanje polja (obično NumPy polja). Prvotna namjena joj je bila emulacija MATLAB [1] mogućnosti iscrtavanja no kasnije se potpuno prilagodila Python načinu djelovanja. U ovom radu koristilo se Matplotlib sučelje `pylab` za iscrtavanje slika, 2D i 3D grafova u vektorskom obliku. Verzija korištene Matplotlib biblioteke: `2.0.0`.

5.1.4. Yael

Yael [7] je biblioteka koja podržava računalno zahtjevne funkcije za vađenje informacije iz velike količine podataka slika. Razvijena je na institutu INRIA [2]. Omogućava efikasno izvođenje algoritama kao što su pretraživanje susjeda (engl. *neighbor search*) i grupiranje (engl. *clustering*). U ovom radu koristi se za efikasno dobivanje mješavine Gaussovih razdiobi (GMM-a) i Fisherovih vektora pomoću konvolucijskih značajki. Verzija korištene Yael biblioteke: `v438`.

5.1.5. SPAMS

SPAMS (engl. *SPArse Modeling Software*) [21] je biblioteka posebno optimizirana za rješavanje problema rijetkih modela. Razvijena je na institutu INRIA. Omogućava efikasnu matričnu faktorizaciju i dekompoziciju. U ovom radu koristi se za učenje

[1] MATLAB (engl. *matrix laboratory*) je numerički skriptni programski jezik namijenjen manipulacijama matricama, iscrtavanju funkcija i podataka, izradi korisničkog sučelja i spajanju programskog koda napisanog u drugim jezicima.

[2] INRIA (fran. *Institut national de recherche en informatique et en automatique*) je francuski institut za računarsku znanost i primijenjenu matematiku.

rijetkog modela s različitim postavkama regularizacije. Verzija korištene SPAMS biblioteke: `2.6`.

5.2. Struktura programske implementacije

Programska implementacija opisanih postupaka ovog rada izvedena je pomoću Pythona i opisanih njegovih vanjskih biblioteka. Za ekstrakciju konvolucijskih značajki iz slika koristili su se moduli `util.py`, `vgg19.py` i `charnet.py` i skripta `extract_features.py`.

Za modeliranje primjeraka, stvaranje generativnog modela, kodiranje u odgovarajuće Fisherove vektore, učenje i ispitivanje modela koristili su se moduli: `gmm.py`, `worker_manager.py`, `sample.py`, `model.py`, `dataset.py` i `tools.py` i programske skripte: `experiment.py` i `segment.py`.

`worker_manager.py`

Modul `worker_manager.py` sadrži razred `WorkerManager` koji omogućava pokretanje željeni broj nezavisnih procesa (radnika, engl. *workers*) koji izvršavaju neki željeni algoritam (posao, engl. *work*). Za komunikaciju se koriste dva reda poruka (engl. *queue*): za slanje prema radnicima i za primanje rezultata od radnika. Korištenjem više paralelnih procesa (engl. *multiprocessing*) postupak učenja i ispitivanja se značajno ubrzava.

`tools.py`

Modul `tools.py` sadrži razrede `ShowWrapper` i `Tools`. Razred `ShowWrapper` omogućava prikazivanje i spremanja `numpy` matrice u obliku slike s različitim opcionalnim postavkama: naslovljavanje slike, prisustvo grafa dimenzija, mijenjanje rezolucije i prikaz u boji. Razred `Tools` sadrži statičke metode kojima se vrše različiti matematički izračuni poput normalizacije Fisherovih vektora i izračuna gradijenta slike.

`sample.py`

Modul `sample.py` sadrži razred `Sample` kojim se modelira primjerak. Njegove instance sadrže putanje do pojedinih dijelova primjeraka: slike, odgovarajućih lokalnih opisnika i anotacije na razini okvira i/ili piksela. Razred sadrži metode kojim

se pojedini dio može učitati u memoriju kao `numpy` matrica.

Metodom `as_img_obj` dohvaća se matrica slike koja se sastoji, ukoliko se radi o slici u boji, od cjelobrojnih vrijednosti koje su raspoređene u tri sloja (svaki za pojedini kanal boje) dimenzija jednakih dimenzijama slike. Methodom `as_conv_obj` dohvaća se matrica lokalnih opisnika odnosno konvolucijskih značajki koja se sastoji od realnih vrijednosti i ima onoliko slojeva koliko i konvolucijski sloj pomoću kojih su dobivene. Metodama `as_annotation_obj` i `as_pixel_obj` dohvaća se matrica anotacija na razini okvira odnosno piksela koja se sastoji od vrijednosti istinosti (engl. *boolean*).

Osim navedenih metoda, valja istaknuti nasumično uzorkovanje lokalnih opisnika primjerka metodom `sample_features` i ekstrakciju lokalnih opisnika koji odgovaraju pozitivnim ili negativnim slikovnim oknima metodom `extract_features`.

`gmm.py`

Modul `gmm.py` sadrži razred `GMM` kojim se uči generativni model u obliku mješavine Gaussovih razdiobi (GMM) pomoću biblioteke `yael`. Rezultat učenja je lista s tri `numpy` matrice: faktori miješanja w_k i parametri $\boldsymbol{\mu}$ i $\boldsymbol{\Sigma}$. Dobivena lista se automatski pohranjuje kao datoteka u obliku serijaliziranog *pickle* objekta. Ukoliko već postoji datoteka GMM-a sa željenim parametrima, metoda učenja `learn` preskače učenje i vraća pohranjenu listu.

`dataset.py`

Modul `dataset.py` sadrži razred `Dataset` kojim se stvara skup podataka kojeg čine Fisherovi vektori slikovnih okana primjeraka dobivenih pomoću biblioteke `yael` i odgovarajuće vrijednosti oznaka. Metoda `create` prima naučeni GMM i objekte primjeraka, i vraća `numpy` matricu Fisherovih vektora $\mathbf{x}_i$ i listu oznaka $y_i \in \{-1, 1\}$. Metoda omogućava normalizaciju Fisherovih vektora. Dobiveni skup podataka se automatski pohranjuje kao datoteka u obliku serijaliziranog *pickle* objekta. Ukoliko već postoji datoteka skupa podataka sa željenim parametrima, metoda preskače stvaranje i vraća pohranjen skup podataka.

`model.py`

Modul `model.py` sadrži razred `Model` koji pomoću metode `learn` uči rijetki model $\mathbf{w}$ na temelju skupa podataka pomoću biblioteke `spams`. Omogućava korištenje različitih regularizacijskih funkcija (l_1,, l_2, $l_{2,1}$) i regularizacijskih faktora, s glo-

balnim izračunom ili izračunom po komponentama. Učenje modela provodi shodno činjenici koristi li normalizirane Fisherove vektore. Naučene težine se automatski pohranjuju u obliku serijaliziranog *pickle* objekta. Ukoliko već postoji datoteka težina sa željenim parametrima, metoda preskače učenje i vraća pohranjene težine. Razred `Model` sadrži još i metode `predict` i `segment` pomoću kojih omogućava izračun odziva slikovnih okana primjerka i pomoću dobivenih odziva segmentira sliku primjerka na način da uklanja sve piksele za koje se pretpostavlja da ne sadrže piksele rukom pisane jednadžbe.

`experiment.py`

Programska skripta `experiment.py` poslužila je provođenje eksperimenata s različitim parametrima. Više detalja o eksperimentima i načinu njihovog provođenja može se pronaći u sljedećem poglavlju.

`segment.py`

Programska skripta `segment.py` služi za segmentiranje primjerka prema željenim parametrima. Omogućava evaluaciju u obliku teksta i slike na kojoj su obojani pikselu u ovisnosti o ispravnoj ili neispravnoj segmentaciji.

6. Eksperimentalni rezultati

Korišteni skup podataka sastojao se od ukupno 1099 primjeraka:

- 850 primjeraka se koristilo za učenje modela
- 150 primjeraka se koristilo za validaciju
- 99 primjeraka se koristilo za ispitivanje

Od navedenog samo su primjerci korišteni za ispitivanje sadržavali osim anotacija na razini okvira i anotacije na razini piksela.

Kao mjera lokalizacijske točnosti koristila se srednja prosječna preciznost (MAP, engl. *Mean Average Precision*) koja je nastala usrednjavanjem prosječnih preciznosti (AP, engl. *Average Precision*) svih primjeraka skupa za validaciju. Prosječna preciznost označava površinu ispod krivulje odnosa između odziva (engl. *recall*) i preciznosti (engl. *precision*). Razlikujemo:

- pozitivno klasificirane primjere koji su pozitivni (TP, engl. *True Positive*)
- pozitivno klasificirane primjere koji su negativni (FP, engl. *False Positive*)
- negativno klasificirane primjere koji su pozitivni (FN, engl. *False Negative*)
- negativno klasificirane primjere koji su negativni (TN, engl. *True Negative*)

Odziv $R = T_p/(T_p + F_n)$ označava udio ispravno klasificiranih pozitivnih primjera u odnosu na ukupan broj pozitivnih primjera. Preciznost $P = T_p/(T_p + F_p)$ označava udio ispravno klasificiranih pozitivnih primjera u odnosu na sve primjere pozitivnog ishoda klasifikacije. Prosječna preciznost balansira te dvije mjere i često se koristi kao mjera točnosti u problemima segmentacije.

Željeni ulazi i parametri postupka učenja odnosno ispitivanja zadali su se konfiguracijskom datotekom. Slijedi objašnjenje pojedinog dijela korištene datoteke.

```
[Sample load]
dir = /home/jmilic/photomath_dataset_2000/fer/handwritten/
suffix_img = .jpg
suffix_conv = _vgg19_conv22.bin
suffix_annotation = .annotation.json
suffix_pixel = .pixel.png
train_perc = 0.85
```

Ovim dijelom postavljaju se filtri pomoću kojih se određuje koji će se datoteke koristiti za izradu primjeraka. Na primjer ako se žele koristiti *CharNet* značajke umjesto *VGG19* značajki, varijabla `suffix_conv` bi se izmijenila u '`_charnet_conv2.bin`' jer taj direktorij sadrži datoteke značajki s tim sufiksom. Varijablom `train_perc` navodi se udio primjeraka koji se koriste za učenje modela, ostatak se koristi za validaciju.

```
[GMM]
dir = /home/jmilic/project/semantic_segmentation/gmm
num_features = 2000000
num_components = 1, 2, 4, 8, 64
fname_pattern = gmm_vgg19_nf%s_k%s_t%s.pkl
num_workers = 24
force_learn = False
```

Ovime se definira gdje će se pohraniti i otkud će se pokušati učitati datoteka GMM-a. Definira se i ime datoteke čime se raspoznaje postoji li već naučen GMM s tim parametrima (nf označava broj uzoraka, k broj komponenti i t broj primjeraka koji su se koristili za učenje). Varijablom `num_components` navodi se koliko će naučeni GMM sadržavati komponenti. Ukoliko je to niz kao u navedenom primjeru onda će se naučiti GMM za svaki željeni broj komponenti. Brojem radnika `num_workers` navodi se koliko će se paralelnih procesa koristiti za učenje GMM-a. Podrazumijeva se korištenje primjeraka iz prethodnog dijela konfiguracijske datoteke `[Sample load]`.

```
[Dataset]
dir = /home/jmilic/project/semantic_segmentation/dataset
normalize = False, True
fname_pattern = dataset_vgg19_k%s_s%s%s.pkl
```

num_workers = 24
force_create = False

Stvaranje skupa podataka, kojeg čine Fisherovi vektori i njihove oznake, je definirano slično kao u dijelu za stvaranje GMM-a. Navodi se direktorij u koji će se pohraniti skup podataka i kako će se datoteka skupa nazvati. Varijablom `normalize` navodi se želi li se normalizirati Fisherove vektore. U slučaju obje vrijednosti, stvara se skup podataka s normaliziranim vektorima i s nenormaliziranim vektorima. Brojem radnika `num_workers` navodi se koliko će se paralelnih procesa koristiti za stvaranje skupa podataka. Podrazumijeva se korištenje primjeraka iz `[Sample load]` i GMM-a iz `[GMM]`.

[Model]
dir = /home/jmilic/project/semantic_segmentation/weights
fname_pattern = weights_vgg19_k%s_reg%s_r%s%s%s.pkl
regularization = l1, l2, l21
reg_factor = 0, 1e−05, 1e−4, 5e−4, 5e−3, 5e−2, 5e−1
intra_component = False, True
num_workers = 24
force_create = False

Za stvaranje modela moguće je koristiti različite parametre. Varijablom `regularization` navodi se koja će se regularizacijska funkcija koristiti sa regularizacijskim faktorom definiranim varijablom `reg_factor`.
Varijablom `intra_component` navodi se želi se izračun odziva sprovesti globalno (engl. *global*) ili po komponenti (engl. *intra-component*) Fisherovog vektora. U slučaju niza brojeva odnosno vrijednosti istinosti, izračun će se sprovesti za svaku kombinaciju vrijednosti varijabli. Podrazumijeva se korištenje primjeraka iz `[Sample load]`, GMM-a iz `[GMM]` i skupa podataka iz `[Dataset]`.

6.1. Pregled rezultata

Konfiguracijskom datotekom definira se kombinacija sljedećih parametara:

- broj komponenti GMM-a
- korištenje (ne)normaliziranih Fisherovih vektora
- regularizacijska funkcija
- regularizacijski faktor
- izračun odziva slikovnog okna globalno ili po komponenti

Postupak s korištenom kombinacijom parametara izvršio se za svaki primjerak skupa za validaciju (150 primjera) čime se dobila prosječna preciznost (AP) za svaki primjerak. Usrednjavanjem rezultata dobila se srednja prosječna preciznost (MAP) za svaku korištenu kombinaciju parametara.

Slijedi pregled najboljih 10 kombinacija parametara koji su izračunati koristeći najprije VGG19, a zatim CharNet konvolucijske značajke. Za globalan izračun odziva koristi se skraćenica 'GLOB', a za izračun po komponenti 'INTRA' (u slučaju jedne komponente $K = 1$ globalan izračun se smatra jednakim izračunu po komponenti). Napomena: prosječna preciznost se računala pomoću anotacije na razini okvira. Svi pikseli unutar okvira se smatraju pozitivnim pikselima, a izvan okvira negativnim pikselima. Zbog toga se pikseli koji su unutar okvira, a nisu dio rukom pisane jednadžbe mogu smatrati u slučaju negativne klasifikacije pogrešno negativnim (FN, engl. *False Negative*). To može uzrokovati nešto manju prosječnu preciznost.

VGG19					
K	Norm. FV	f_{reg}	λ	Odziv	$MAP\ \%$
1	Da	$l_{2,1}$	0.05	GLOB	71.29
1	Da	$l_{2,1}$	0.005	GLOB	71.02
1	Da	l_1	0.05	GLOB	70.64
1	Da	l_1	0.005	GLOB	70.11
4	Da	$l_{2,1}$	0.005	INTRA	69.93
4	Da	$l_{2,1}$	0.005	GLOB	69.91
1	Da	$l_{2,1}$	0.0005	GLOB	67.28
1	Da	l_1	0.0005	GLOB	67.02
4	Da	$l_{2,1}$	0.0005	INTRA	66.48
4	Da	$l_{2,1}$	0.0005	GLOB	66.37

CharNet					
K	Norm. FV	f_{reg}	λ	Odziv	$MAP\ \%$
1	Da	$l_{2,1}$	0.05	GLOB	73.17
1	Da	$l_{2,1}$	0.005	GLOB	73.12
1	Da	l_1	0.05	GLOB	72.23
1	Da	l_1	0.005	GLOB	72.01
2	Da	$l_{2,1}$	0.05	INTRA	71.78
2	Da	$l_{2,1}$	0.05	GLOB	71.63
1	Da	l_1	0.0005	GLOB	69.44
4	Da	$l_{2,1}$	0.005	INTRA	68.94
4	Da	$l_{2,1}$	0.005	GLOB	68.81
2	Da	$l_{2,1}$	0.005	GLOB	67.67

Korištenjem CharNet konvolucijskih značajki postigli su se malo bolji rezultati. U oba slučaja najbolja srednja prosječna preciznost (MAP) na skupu za validaciju postiže se korištenjem GMM-a s jednom komponentom i korištenjem normaliziranih Fisherovih vektora. Kao najpogodnija regularizacija pokazala se regularizacijska funkcija $l_{2,1}$ s faktorom regularizacije $\lambda = 0.05$. Iznenađujuće je da su se povećanjem broja komponenti rezultati pogoršali. Tako npr. u slučaju VGG19 značajki i korištenja 64 komponenti srednja prosječna preciznost (MAP), uz korištenje najboljih ostalih parametara, iznosi 62.07.

Na skupu za validaciju pronašla se najpogodnija kombinacija parametara. Također, pokazano je da bolje rezultate postižu CharNet konvolucijske značajke. Navedeno se koristilo na skupu za ispitivanje koji za razliku od skupa za validaciju sadrži i anotacije na razini piksela. Time se preciznije izračunala prosječna preciznost za pojedini primjerak. Srednja prosječna preciznost (MAP) na skupu za ispitivanje (99 primjera) iznosi **74.28**.

6.2. Pregled segmentiranih slika primjeraka

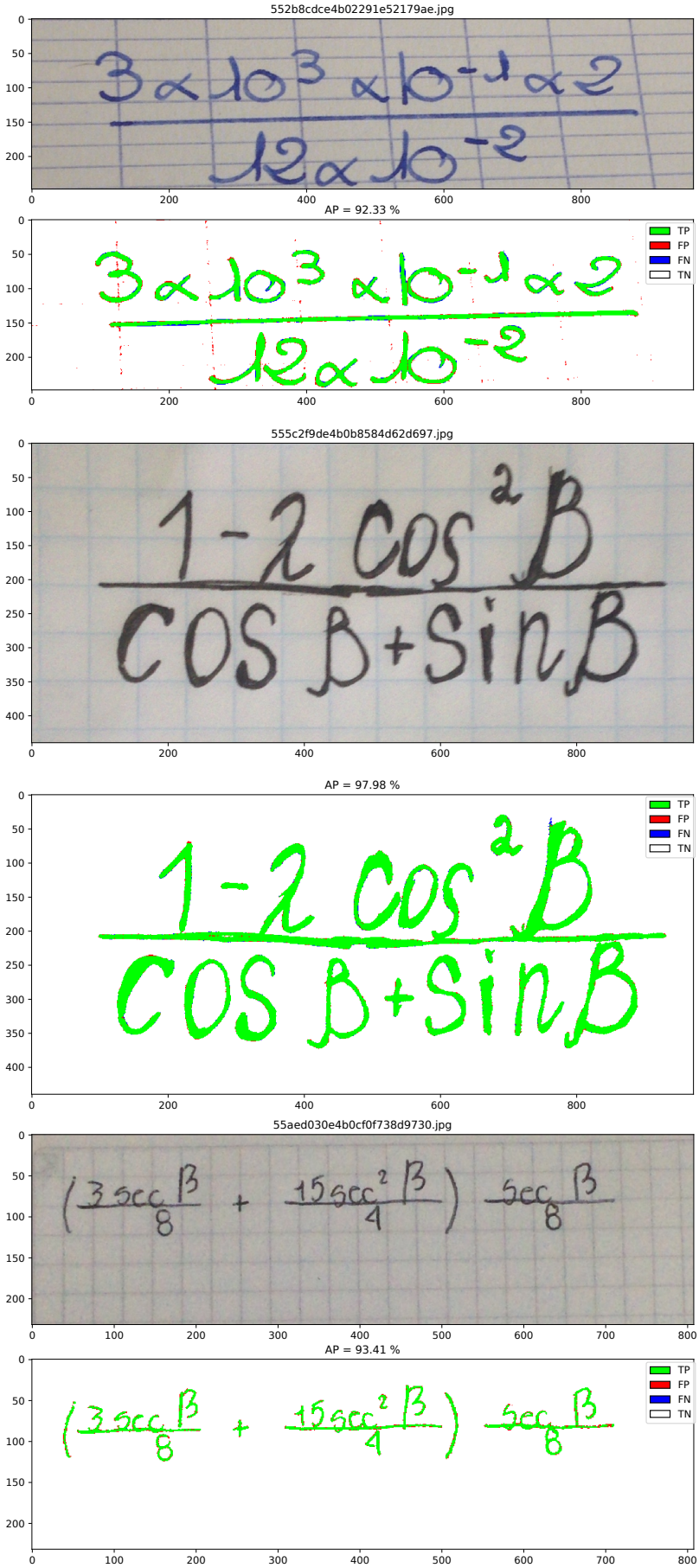
Kao slikovna okna moguće je okna veličine piksela prilikom ispitivanja. Tako se za svaki piksel može dobiti odgovarajući Fisherov vektor, pomoću modela izračunati odziv i zatim odrediti radi li se o pozitivnom ili negativnom pikselu. Time se dobiva matrica segmentacija pomoću koje se mogu izvući pikseli od interesa na slikama pri-

mjeraka. U sljedećim primjerima evaluacija rezultata se prikazuje bojanjem piksela slike. Svi pozitivni pikseli koji su klasificirani kao pozitivni (TP, engl. *True Positive*) prikazani su zelenom bojom. Svi pozitivni pikseli koji su klasificirani kao negativni (FN, engl. *False Negative*) prikazani su plavom bojom. Svi negativni pikseli koji su klasificirani kao pozitivni (FP, engl. *False Positive*) prikazani su crvenom bojom. Svi negativni pikseli koji su klasificirani kao negativni (TN, engl. *True Negative*) prikazani su bijelom bojom. Uz sliku evaluacije prikazana je i originalna slika kao referenca.

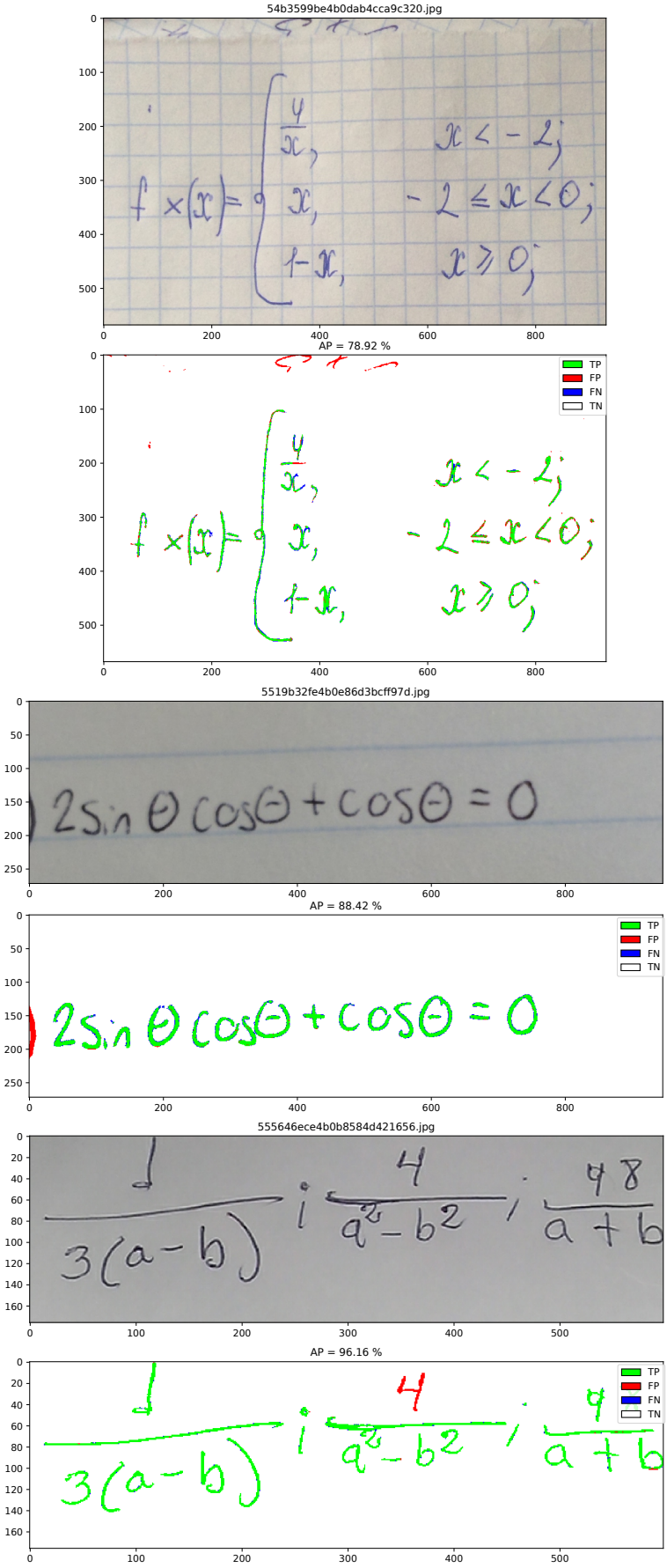
Slika 6.1 prikazuje neke primjere u kojima su se pikseli pozadine uspješno klasificirali kao negativni, a pikseli jednadžbe kao pozitivni.

Slika 6.2 prikazuje zanimljive primjere u kojima su se uspješno prepoznali pikseli rukom pisane jednadžbe, ali nisu bili anotirani.

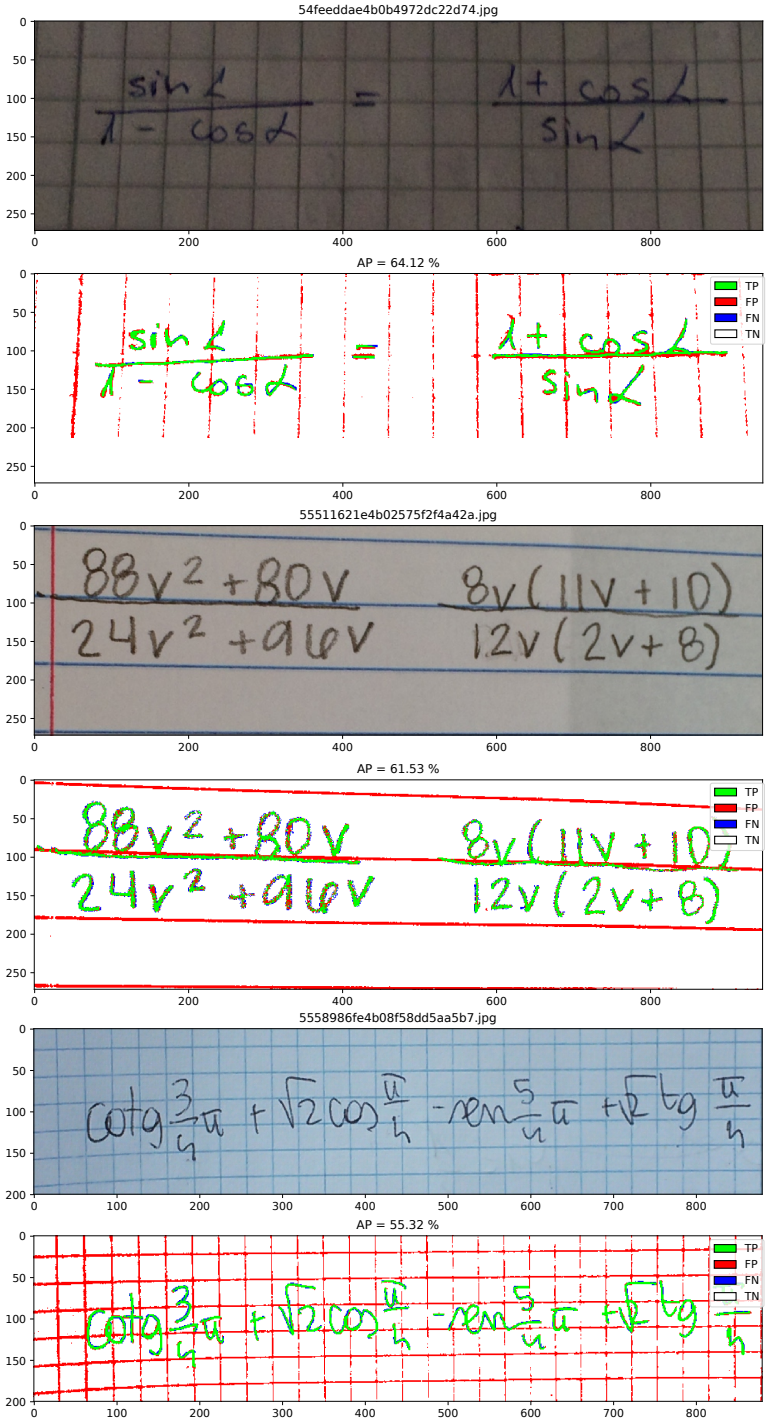
Slika 6.3 prikazuje primjere koji sadrže lošu segmentaciju. Valja istaknuti kako su pikseli rukom pisane jednadžbe velikom većinom ispravno klasificirani.



Slika 6.1: Primjeri uspješno segmentiranih primjeraka



Slika 6.2: Primjeri zanimljivih segmentacija



Slika 6.3: Primjeri loših segmentacija

7. Zaključak

U ovom radu opisuje se postupak slabo nadzirane semantičke segmentacije pomoću Fisherovih vektora. Lokalnim opisnicima pokušava se dohvatiti neodređena struktura koja je reprezentirana na više razina. Pomoću njih se uči generativni model u obliku mješavine Gaussovih razdiobi (GMM) kojim se pokušava pronaći njihova razdioba. Pomoću GMM-a lokalni opisnici kodiraju se u Fisherov vektor koji služi kao reprezentacija slike odnosno slikovnog okna. Ideja postupka je u stvaranju slikovnih riječi koje odgovaraju pojedinim komponentama Fisherovog vektora i koje sadrže diskriminativnu informaciju koja služi za razlučivanje željene strukture. Pomoću Fisherovih vektora i oznaka, kojima se označava sadrži li dio slike rukom pisanu jednadžbu, uči se rijetki model. Model se može koristiti za izračun odziva slikovnog okna odnosno za prepoznavanje sadrži li željeni objekt. Efikasnost i točnost lokalizacije se može poboljšati uporabom normaliziranih Fisherovih vektora i gradijenta odziva slike.

Opisan je zanimljiv postupak čija je implementacija imala neočekivane eksperimentalne rezultate. Pokazalo se da se najbolji rezultati postižu uporabom jedne komponente GMM-a. Očekivano je da je jedna komponenta premala količina za pronalazak razdiobe opisnika. U pojedinim primjerima segmentacija je bila vrlo uspješna, ali općenito, rezultati nisu sjajni. Moguće je da bi se popravili korištenjem većeg skupa podataka za učenje jer pokazano je da se korišteni skup sastoji od slika koje sadrže rukom pisane jednadžbe u mnogo različitih oblika, rukopisa, boje, veličine i oštrine samog prikaza. Pozadine su također vrlo raznolike. Jedan od mogućih problema je i što se radi o finim detaljima na slici koji se slabo razlikuju od objekata koji ne pripadaju rukom pisanim jednadžbama. Tako se na primjer jednadžbe obično pišu na rešetkama (popularno nazvanih 'kvadratićima') matematičkih bilježnica pa sadrže mnogo horizontalnih i vertikalnih linija. S obzirom da se za učenje koristila anotacija na razini okvira, često se unutar okvira nalazila i pozadina dijela rešetke čime se otežava postupak pronalaska diskriminativne informacije. Unatoč navedenome, postupak je demonstrirao moć Fisherovih vektora i pokazao potencijal daljnjeg razvoja.

LITERATURA

[1] Katarina Blažić. Interaktivna semantička segmentacija rukom pisanih znakova. 2017.

[2] Alfredo Canziani, Adam Paszke, i Eugenio Culurciello. An analysis of deep neural network models for practical applications. *CoRR*, abs/1605.07678, 2016. URL `http://arxiv.org/abs/1605.07678`.

[3] George Casella i Roger Berger. R. 2001, statistical inference. *Duxbury Press*.

[4] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, i Jürgen Schmidhuber. Deep big simple neural nets excel on handwritten digit recognition. *CoRR*, abs/1003.0358, 2010. URL `http://arxiv.org/abs/1003.0358`.

[5] Cmglee. Wikipedia. `https://en.wikipedia.org/wiki/Pyramid_(image_processing)`.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, i L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. U *CVPR09*, 2009.

[7] Matthijs Douze i Hervé Jégou. The yael library. U *Proceedings of the 22nd ACM international conference on Multimedia*, stranice 687–690. ACM, 2014.

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, i Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, i Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 580–587, 2014.

[10] Chris Harris i Mike Stephens. A combined corner and edge detector. U *Alvey vision conference*, svezak 15, stranice 10–5244. Manchester, UK, 1988.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

[12] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95, May 2007. ISSN 1521-9615. doi: 10.1109/MCSE.2007.55.

[13] Tommi Jaakkola i David Haussler. Exploiting generative models in discriminative classifiers. U *Advances in neural information processing systems*, stranice 487–493, 1999.

[14] M. Jaderberg, A. Vedaldi, i A. Zisserman. Deep features for text spotting. U *European Conference on Computer Vision*, 2014.

[15] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, i Andrew Zisserman. Reading text in the wild with convolutional neural networks. *CoRR*, abs/1412.1842, 2014. URL `http://arxiv.org/abs/1412.1842`.

[16] Hervé Jégou, Matthijs Douze, i Cordelia Schmid. On the burstiness of visual elements. U *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, stranice 1169–1176. IEEE, 2009.

[17] Alex Krizhevsky, Ilya Sutskever, i Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. U *Advances in neural information processing systems*, stranice 1097–1105, 2012.

[18] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10): 1995, 1995.

[19] Yann LeCun, Yoshua Bengio, i Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[20] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. ISSN 1573-1405.

[21] Julien Mairal, Francis Bach, Jean Ponce, i Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11(Jan):19–60, 2010.

[22] Florent Perronnin, Jorge Sánchez, i Thomas Mensink. Improving the fisher kernel for large-scale image classification. *Computer Vision–ECCV 2010*, stranice 143–156, 2010.

[23] Jorge Sánchez, Florent Perronnin, Thomas Mensink, i Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

[24] SciPy. What is numpy? `https://docs.scipy.org/doc/numpy-1.12.0/user/whatisnumpy.html`.

[25] Karen Simonyan i Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL `http://arxiv.org/abs/1409.1556`.

[26] Padhraic Smyth. The EM algorithm for Gaussian mixtures. `http://www.ics.uci.edu/~smyth/courses/cs274/notes/EMnotes.pdf`.

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, i Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL `http://arxiv.org/abs/1409.4842`.

[28] S. van der Walt, S. C. Colbert, i G. Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, 13(2): 22–30, March 2011. ISSN 1521-9615. doi: 10.1109/MCSE.2011.37.

[29] Valentina Zadrija. Lokalizacija objekata odozdo prema gore primjenom fisherovih vektora. 2017.

[30] Eric R Ziegel i John Rice. Mathematical statistics and data analysis, 1995.

Slabo nadzirana semantička segmentacija rukom pisanih jednadžbi

Sažetak

Semantička segmentacija je važan zadatak računalnog vida. Cilj tog zadatka je odrediti semantički razred svakog pojedinog piksela slike strojno naučenim modelom. Kao i u drugim problemima računalnog vida, najbolji rezultati postižu se strogo nadziranim učenjem. Međutim, veliki nedostatak tog pristupa je potreba za ručnim označavanjem svakog pojedinog piksela u skupu slika za učenje. Kako bismo smanjili potrebu za tim skupim procesom, razmatramo mogućnost slabo nadziranog učenja segmentacijskih modela. U ovom pristupu na temelju anotacije na razini okvira u skupu podataka za učenje dobiva se anotacija na razini piksela znakova rukom pisanih jednadžbi u skupu podataka za ispitivanje.

Ključne riječi: slabo, nadzirana, semantička, segmentacija, rukom, pisana, jednadžba, računalni, vid

Weakly-supervised semantic segmentation of hand-written equations

Abstract

Semantic segmentation is an important task of computer vision. The goal of this task is to determine the semantic class of each individual pixel in a machine-learned model. As in other computer vision problems, the best results are achieved by strictly controlled learning. However, the major disadvantage of this approach is the need to manually mark each pixel in a learning image set. To reduce the need for this expensive process, we consider the possibility of poorly supervised learning of segmentation models. In this approach, based on the bounding box level annotation in the train dataset, annotation is obtained at the level of character pixels of handwritten equations in the test dataset.

Keywords: weakly, supervised, semantic, segmentation, handwritten, equation, computer, vision