Segmentare de imagini

Alexandru Agape

24 Aprilie 2012

1 Introducere

În această temă veţi lua contact cu două domenii ale *computer science* aflate pe val, şi anume *computer vision* şi *machine learning*. Mulţi dintre algoritmii învăţaţi de voi la Proiectarea Algoritmilor sunt folosiţi în rezolvarea problemelor ce ţin de computer vision (sau mai general, a problemelor de machine learning aplicate şi în computer vision). Inferenţa în modele probabilistice graf-reprezentabile [5] admite rezolvări prin metode greedy, programare dinamică, drumuri minime şi flux maxim, şi are numeroase aplicaţii. Una dintre acestea este segmentarea imaginilor, ce face subiectul temei. Chiar dacă vor fi menţionate mai multe concepte specifice celor două domenii, rezolvarea temei nu este condiţionată de înţelegerea acestora şi nu ar trebui să vă sperie. Teoretic, secţiunea 4 reprezintă un enunţ suficient al temei, restul documentului oferind doar explicaţii suplimentare şi o motivare succintă a formulelor de calcul. Veţi constata însă că seminarele de matematică din anul I nu au fost o pierdere de timp.

2 Segmentarea imaginilor

Aceasta presupune împărţirea unei imagini în multiple segmente după nişte criterii. Sunt posibile numeroase abordări şi interpretări ale problemei [13] (*clustering*-ul fiind adesea întâlnit). Noi vom modela însă problema folosind probabilităţi şi distribuţii de tip Gibbs [12].

Problema segmentării nu este o problemă bine definită (ill-posed problem): nu exista (decât rareori) un răspuns corect, doi oameni construind soluţii diferite pentru o imagine dată. O primă cauză ar fi faptul că o imagine poate fi segmentată atât în trei-patru regiuni mari cât şi în o sută de segmente foarte mici, în funcţie de răbdarea şi raţionamentul celui care adnotează. Pentru a face lucrurile mai clare ne vom concentra atenţia pe extragerea unui singur segment reprezentând obiectul/obiectele din prim-plan, restul imaginii considerându-l a fi fundal. Acest caz particular se numeşte segmentare binară sau foreground-background.

3 Modelarea problemei

Extragerea segmentului principal din imagine poate fi văzută ca a asigna fiecărui pixel din imagine o valoare **true** sau **false** după cum aparţine sau nu segmentului respectiv. Vom defini deci pentru fiecare pixel o variabilă aleatoare $X_i = x_i \in \{0, 1\}$. Notând cu N numărul total de pixeli din imagine vom avea deci 2^N posibilităţi de a asigna valori variabilelor (numite în continuare configuraţii), şi deci tot 2^N posibile segmentări ale imaginii. Dintre toate acestea, trebuie aleasă însă cea mai bună, după nişte criterii date.

3.1 Modelul probabilistic

Vom construi o distribuţie de probabilităţi peste spaţiul exponenţial de soluţii. Vom folosi modelul Gibbs $\widetilde{p}(x) = e^{-E(x)}$ pentru a modela probabilitatea ca o anumită configuraţie $x = x_1x_2...x_N$ să reprezinte segmentul principal (foreground). Vom considera drept soluţie configuraţia cu cea mai mare probabilitate sub modelul ales. Datorită monotoneicităţii funcţiei e^{-x}, probabilitatea maximă echivalează cu energia minimă. Să vedem cum putem modela funcţia de energie pentru a ne asigura că într-adevăr configuraţia cu energie minimă reprezintă o segmentare binară bună a imaginii.

3.1.1 Model independent

În primă fază vom considera un model în care cele N variabile aleatoare sunt independente. Altfel spus, pentru fiecare pixel, decizia dacă este sau nu din segmentul principal nu este influenţată de decizia luată pentru vecinii săi. Vom construi atât pentru fundal cât şi pentru prim-plan câte un model probabilistic simplu, si pe baza acestora vom construi funcţia de energie. Considerand că $p_f(x_i)$ modelează probabilitatea unui pixel de a fi în prim-plan, ne interesează energia asociată cu aceasta într-un model de tip Gibbs, ce are valoarea $-\log p_f(x_i)$.

$$E_u(x) = \sum_{i=1}^{N} \{x_i \cdot (-\log p_f^i) + (1 - x_i)(-\log p_b^i)\} \tag{1}$$

$$E(x) = E_u(x) \tag{2}$$

Pentru a minimiza această energie, este de ajuns a alege pentru fiecare pixel acea variantă (foreground/background) ce are probabilitate mai mare.

3.1.2 Modelul complet

În funcţie de modelul ales pentru p_f^i şi p_b^i, modelul anterior poate da rezultate mulţumitoare, dar în continuare este loc de mai bine. Se doreşte renunţarea la presupunerea de independenţă. Daca un pixel este fundal, atunci e foarte probabil ca un pixel alăturat, mai ales dacă este foarte asemănător, să fie şi el tot fundal. Dorim deci să penalizăm aceste inconsistenţe între pixeli vecini, lucru realizabil prin adăugarea unor noi termeni la energia configuraţiei. Termenii noi sunt ponderaţi cu o constantă λ.

$$E_p(x) = \sum_{(i,j) \in \mathcal{N}} f_p^{ij}(x_i, x_j) \tag{3}$$

$$E(x) = E_u(x) + \lambda E_p(x) \tag{4}$$

3.2 Particularizarea modelului

Deşi modelul a fost deja stabilit, numeroase alegeri ce influenţează calitatea solutiilor au rămas, şi anume forma concretă a probabilităţilor p_f^i şi p_b^i şi a funcţiei f_p^{ij}.

3.2.1 Costul individual

Probabilităţile de apartenenţă la prim-plan şi fundal se modelează similar: fiind daţi nişte pixeli marcaţi de utilizator ca fiind sigur din clasa respectivă putem construi un model de culoare al clasei de tip Gaussian [10]. Un astfel de model are doar doi parametri: μ - media valorilor şi σ - deviaţia standard. Cei doi parametri pot fi calculaţi folosind formulele 6 respectiv 7. Probabilităţile sunt apoi calculate folosing o distribuţie normală ca în ecuaţia 5, unde **argumentul x reprezintă culoarea pixelului** în cauză.

$$\mathcal{N}(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{5}$$

$$\mu = \frac{\sum_{i \in Mask} Image(i)}{|Mask|} \tag{6}$$

$$\sigma = \sqrt{\frac{\sum_{i \in Mask}(\mu - Image(i))^2}{|Mask|}} \tag{7}$$

Acesta este un model mult simplificat. Rezultate mult mai bune se obţin utilizând o combinaţie de distribuţii gausiene (*Gaussian Mixture Model* [11]) ce acoperă cazurile în care obiectul/fundalul nu este omogen, ci alcătuit din mai multe părţi de culori diferite. Stabilirea parametrilor μ şi σ pentru fiecare din componentele modelului este însă mult mai dificilă aşa că ne vom mulţumi cu modelul simplificat. Dacă doriţi însă să experimentaţi puteţi simula un model GMM folosind marcaje indiviuale (pixeli selectati din imagine) pentru fiecare componenta a mixturii. Nu se cere şi nu va fi punctată drept corectă o astfel de abordare.

Întrucât modelul este slab, pentru unii pixeli se poate întâmpla ca valoarea uneia din probabilităţi să fie foarte mică ducând la o valoare mare a logaritmului. De aceea aceste costuri se truncheazǎa la o valoare UMAX=10. De asemenea, pentru simplificare e garantat ca în toate testele valoarea lui σ nu va fi 0.

3.2.2 Costul perechilor

Cel mai simplu mod de a defini funcţia de penalizare este de a nu ţine cont de culoarea celor doi vecini, şi a penaliza egal orice neconcordanţă (Ecuatia 8). Energia astfel obţinută este numită energie Ising (sau mai general Potts) iar modelul nostru probabilistic complet este un *Markov Random Field* [14]. Un model probabilistic mai puternic (cunoscut drept *Conditional Random Field*) se obţine condiţionând penalizarea de observaţie (culoarea pixelilor), prin folosirea unei funcţii de tipul celei din ecuaţia 9. O vom folosi pe aceasta (prima variantă va fi doar un caz particular, pentru un *treshold* foarte mare). Vom considera ca vecini oricare doi pixeli învecinaţi pe verticală sau orizontală (sunt maxim 4 vecini pentru un pixel). Fiecare pereche de pixeli este considerată o singură dată.

$$f_p(x_i, x_j) = 1 - \delta(x_i, x_j) = \begin{cases} 1 & \text{daca } x_i \neq x_j \\ 0 & \text{altfel} \end{cases} \quad (8)$$

$$f_p^{ij}(x_i, x_j) = \begin{cases} 1 & \text{dacă } x_i \neq x_j \text{ şi } |\text{Imagine(i) - Imagine(j)}| \leq \textit{treshold} \\ 0 & \text{altfel} \end{cases} \quad (9)$$

Deoarece funcţia din ecuaţia 8 defineşte o metrică iar cea din ecuaţia 9 o pseudometrică, configuraţia ce minimizează energia din ecuaţia 4 poate fi calculată exact (nu doar o aproximaţie), cu toate că spaţiul de căutare este unul exponenţial.

4 Cerinţe

4.1 Date de intrare

Programul vostru va citi datele de intrare din mai multe fişiere detaliate în continuare:

- **imagine.pgm**
  - fişier de tip pgm [1] conţinând imaginea ce trebuie segmentată
- **mask_fg.pgm**
  - fişier de tip pgm conţinând pixelii marcaţi de utilizator ca facând parte din obiect (valori nenule)
  - pixelii respectivi vor fi folosiţi pentru a calcula parametrii μ si σ pentru modelul obiectului (se folosesc valorile din imagine a pixelilor marcaţi în mască)
- **mask_bg.pgm**
  - fişier de tip pgm conţinând pixelii marcaţi de utilizator ca facând parte din fundal (valori nenule)
  - pixelii respectivi vor fi folosiţi pentru a calcula parametrii μ si σ pentru modelul fundalului (se folosesc valorile din imagine a pixelilor marcaţi în mască)

- **parametri.txt**
  - fişier text conţinând diverşi parametri de tip întreg, câte unul pe linie
  - parametri incluşi: λ (vezi ecuaţia 4), *treshold* (vezi ecuaţia 9).
- observaţii fişiere pgm
  - toate fişierele sunt de tip *plain* (magic number $P2$)
  - fişierele nu conţin comentarii
  - valoarea lui $Maxval$ este întotdeauna 255
  - în Windows pot fi deschise cu *gimp*

4.2 Ieşire

Programul vostru va crea fişierul **segment.pgm** valid, fără comentarii, având $Maxval = 255$ şi dimensiunile imaginii egale cu cele ale imaginii de intrare. Pentru fiecare pixel considerat a fi fundal se va afişa valoarea 0, iar pentru pixelii obiect valoarea 255. Vom verifica dacă segmentarea dată de voi produce o configuraţie pentru care energia este minimă, cu o precizie de 2 zecimale (în caz că există mai multe soluţii oricare este bună).

4.3 Detalii de implementare

Trebuie să scrieţi un program care calculează configuraţia $x = x_1 x_2 ... x_N$ ce minimizează funcţia din ecuaţia 10.

$$E(x) = \sum_{i=1}^{N} f_u^i(x_i) + \lambda \sum_{(i,j)\in\mathcal{N}} f_p^{ij}(x_i, x_j) \tag{10}$$

În continuare descriem variabilele ce apar în ecuaţie. Pentru prima sumă avem (deduse din ecuaţiile anterioare):

$$\widetilde{f_u^i}(x) = x \cdot (\frac{1}{2}(\frac{Image(i) - \mu_f}{\sigma_f})^2 + \log \sqrt{2\pi\sigma_f^2}) +$$
$$(1-x) \cdot (\frac{1}{2}(\frac{Image(i) - \mu_b}{\sigma_b})^2 + \log \sqrt{2\pi\sigma_b^2}) \quad (11)$$
$$f_u^i(x) = \min(\widetilde{f_u^i}(x), 10) \quad (12)$$
$$\mu_f = \frac{\sum_{i \in Mask_f} Image(i)}{|Mask_f|} \quad (13)$$
$$\sigma_f = \sqrt{\frac{\sum_{i \in Mask_f} (\mu - Image(i))^2}{|Mask_f|}} \quad (14)$$
$$\mu_b = \frac{\sum_{i \in Mask_b} Image(i)}{|Mask_b|} \quad (15)$$
$$\sigma_b = \sqrt{\frac{\sum_{i \in Mask_b} (\mu - Image(i))^2}{|Mask_b|}} \quad (16)$$
$$(17)$$

Logaritmii care apar în aceste ecuaţii precum şi în restul documentului sunt logaritmi naturali - au baza e (sunt derivate din logaritmarea unor exponenţiale). În $C++$ şi $Java$ puteţi deci folosi funcţia log din $math.h$ ţespectiv $java.lang.Math$ fără alte scalări.

Pentru ce-a dea doua sumă, deduse din ecuaţiile anterioare, avem relaţiile:

$$f_p^{ij}(x_1, x_2) = \begin{cases} 1 & \text{dacă } x_1 \neq x_2 \text{ şi } |\text{Imagine(i) - Imagine(j)}| \leq treshold \\ 0 & \text{altfel} \end{cases} \quad (18)$$

$$\mathcal{N} = \{(i,j) | 1 \leq i < j \leq N \ si \ i \ adiacent \ j\} \quad (19)$$

4.3.1 Hint 1

Deşi nu sunteţi constrânşi în a proceda astfel, puteţi afla configuraţia minimă aplicând und algoritm de flux maxim (tăietură minimă).

4.3.2 Hint 2

Graful se poate construi astfel: fiecare pixel este un nod, există muchii doar între pixeli vecini ($\mathcal{N}$ devine mulţimea muchiilor) şi acestea au capacităţi proporţionale cu valoarea lui $f_p^{ij}(0,1) = f_p^{ij}(1,0)$ pentru pixelii respectivi. Suplimentar, apar două noi noduri f şi g de care se leagă toate celelalte noduri prin muchii ale căror capacităţi modelează costurile unare $f_u^i(1)$ respectiv $f_u^i(0)$. Graful este neorientat. O tăietură minimă în acest graf este echivalentă cu o configuraţie ce minimizează funcţia de energie: dacă o muchie $f - i$ face parte din tăietură atunci muchia $i - g$ nu este în tăietura minimă iar pixelul corespunzător nodului i este din segmentul principal. Fluxul maxim (şi capacitatea tăieturii) este egal cu valoarea minimă a energiei: pentru orice pixel i din prim-plan, muchia $f - i$ face parte din tăietură iar la energie se adaugă termenul $f_u^i(1)$ dar nu şi $f_u^i(0)$, şi similar pentru pixelii din fundal; pentru doi pixeli vecini ce primesc aceaşi valoare (ori fundal ori prim-plan) nu se taie nicio muchie, dar nici la energie nu se adună nimic ($f_p^{ij}(x,x) = 0, \forall x \in \{0,1\}$); dacă cei doi vecini primesc valori diferite, de exemplu $x_i = 1$ şi $x_j = 0$ atunci muchia $i - j$ având capacitatea $f_p^{ij}(1,0)$ face parte din tăietură.

4.4 Punctare

Pentru 5 puncte din 8 programul vostru trebuie să dea rezultate corecte pe imagini având un număr total de pixeli $N \leq 2100$. Pentru toate cele 8 puncte trebuie ca programul să ruleze pentru $N \leq 50000$ (atenţie la memorie, nu e permis să utilizaţi mai mult de 64MB). Vor apărea ulterior informaţii suplimentare despre limitele de timp în funcţie de limbajul de programare.

5 Aprecieri

Dacă vreţi să aflaţi mai multe despre cele două domenii introduse — *computer vision* şi *machine learning*, vă recomandăm să urmaţi cursurile *Coursera* predate de doi profesori renumiţi ce se desfăşoară în această perioadă ([7] şi [6]).

Pentru mai multe informaţii despre segmentarea imaginilor în general puteţi consulta articolele ştiinţifice [2], [9], [4], [3] sau [8].

6 Revizii ale documentului

- **24.04.2012** Versiunea iniţială
- **27.04.2012** Corectare cost individual (cand σ este foarte mic).

- **1.05.2012** Precizare baza logaritm.
- **2.05.2012** Corectie ecuatia 11; probabilitatea ca un pixel sa fie foreground depinde de culoarea pixelului.

Cuprins

Referinţe

[1] Pgm file type. http://netpbm.sourceforge.net/doc/pgm.html.

[2] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient n-d image segmentation.

[3] Yuri Boykov and Marie-Pierre Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *ICCV*, pages 105–112, 2001.

[4] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.

[5] Daphne Koller. Probabilistic graphical models. https://www.coursera.org/course/pgm.

[6] Jitendra Malik. Computer vision: The fundamentals. https://www.coursera.org/course/vision.

[7] Andrew Ng. Machine learning. https://www.coursera.org/course/ml.

[8] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. ”grabcut”: interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[9] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall F. Tappen, and Carsten Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):1068–1080, 2008.

[10] Wikipedia. Gaussian distribution. http://en.wikipedia.org/wiki/Normal_distribution.

[11] Wikipedia. Gaussian mixture model. http://en.wikipedia.org/wiki/Mixture_model.

[12] Wikipedia. Gibbs distribution. http://en.wikipedia.org/wiki/Boltzmann_distribution.

[13] Wikipedia. Image segmentation. http://en.wikipedia.org/wiki/Segmentation_(image_processing).

[14] Wikipedia. Markov random field. http://en.wikipedia.org/wiki/Markov_random_field.