

Análise de Imagens (MO445/MC940)

Prof. Alexandre Xavier Falcão

25 de setembro de 2008

1 Classificação supervisionada por florestas de caminhos ótimos

No caso supervisionado, o rótulo $\lambda(s) \in \{1, 2, \dots, c\}$ de todas as amostras $s \in Z_1$ é conhecido. Esta informação pode ser usada para encontrar protótipos para S em todas as classes w_i , $i = 1, 2, \dots, c$. A função de conexidade deve ser pensada para que as amostras de uma dada classe sejam mais fortemente conexas a protótipos desta classe do que a protótipos de qualquer outra classe.

Uma idéia é definir protótipos entre as amostras mais próximas de classes distintas. O exemplo da aula anterior ilustra este caso, onde o peso dos arcos $w(s, t)$ pode ser a distância $d(s, t)$ entre os vetores de características $\vec{v}(s)$ e $\vec{v}(t)$. A função de conexidade fica então:

$$\begin{aligned} f_1(\langle t \rangle) &= \begin{cases} 0 & \text{if } t \in S, \\ +\infty & \text{no caso contrário.} \end{cases} \\ f_1(\pi_s \cdot \langle s, t \rangle) &= \max\{f_1(\pi_s), d(s, t)\}. \end{aligned} \quad (1)$$

A minimização de f_1 com protótipos na fronteira faz com que os protótipos evitem que amostras de sua classe sejam conquistadas por protótipos de outras classes.

Os protótipos podem ser encontrados como as amostras de classes distintas, as quais compartilham um arco na árvore de peso mínimo obtida do grafo. A árvore de peso mínimo é aquela cuja a soma dos pesos dos arcos é mínima. As seções seguintes descrevem o método (ver código em www.ic.unicamp.br/~afalcao/libopf).

1.1 Treinamento em Z_1

Dado o grafo completo (Z_1, A_1) , onde $t \in A_1(s)$ se $t \neq s$ para todo $s, t \in Z_1$, e função de conexidade f_1 , o treinamento consiste de encontrar os protótipos para S^* através da árvore de peso mínimo e a floresta de caminhos ótimos com raízes em S^* . Assumindo que conhecemos S^* , o algoritmo FCO propaga os rótulos dos protótipos para todas as amostras de suas respectivas árvores de caminhos ótimos, formando um mapa de rótulos $L(s) \in \{1, 2, \dots, c\}$:

Algorithm 1 – ALGORITMO FCO

INPUT: Conjunto Z_1 , protótipos $S^* \subset Z_1$ e o par (v, d) para extração de características e cálculo de distâncias.
 OUTPUT: Floresta de caminhos ótimos P , mapa de valores ótimos V e mapa de rótulos L .
 AUXILIARY: Fila Q de prioridades e variável tmp .

1. Para cada $s \in Z_1 \setminus S^*$, faça $V(s) \leftarrow +\infty$.
2. Para cada $s \in S^*$, faça
3. \perp $V(s) \leftarrow 0$, $P(s) \leftarrow nil$, $L(s) \leftarrow \lambda(s)$, e insira s em Q .
4. Enquanto Q não for vazia, faça
5. $\left[$ Remova de Q uma amostra s tal que $V(s)$ é mínimo.
6. $\left[$ Para cada $t \in Z_1$ tal que $t \neq s$ e $V(t) > V(s)$, faça
7. $\left[$ $\left[$ Calcule $tmp \leftarrow \max\{V(s), d(s, t)\}$.
8. $\left[$ $\left[$ Se $tmp < V(t)$, então
9. $\left[$ $\left[$ $\left[$ Se $V(t) \neq +\infty$, então remova t de Q .
10. $\left[$ $\left[$ $\left[$ $\left[$ $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$ e $V(t) \leftarrow tmp$.
11. $\left[$ $\left[$ $\left[$ $\left[$ $\left[$ Insira t em Q .

As linhas 1 – 3 inicializam mapas e inserem protótipos em Q . O laço principal calcula caminhos ótimos de S^* para toda amostra s , seguindo uma ordem não-decrescente de valores (linhas 4 – 11). A cada iteração, um caminho π_s de valor ótimo $V(s)$ é obtido em P , quando removemos seu último nó s de Q (linha 5). Empates são resolvidos com política *first-in-first-out* em Q . Isto é, quando dois caminhos ótimos encontram uma mesma amostra s , esta amostra é associada ao primeiro caminho que a encontrou. Note que $V(t) > V(s)$ na linha 6 é falso quando s não pode modificar os atributos de t e que $V(t) \neq +\infty$ na linha 9 é verdadeiro apenas quando $t \in Q$. As demais linhas avaliam se o caminho $\pi_s \cdot \langle s, t \rangle$ é melhor que o caminho atual $\pi_t \in P$ e, se for, atualizam Q , $V(t)$, $L(t)$ e $P(t)$ de acordo (i.e., $\pi_t \leftarrow \pi_s \cdot \langle s, t \rangle$).

O Algoritmo 1 também pode ser modificado para calcular uma árvore de peso mínimo (MST - *minimum spanning tree*) e selecionar protótipos para S^* . Neste caso, a função de conectividade f_{msf} não é suave, mas o algoritmo se degenera no Algoritmo de Prim.

$$\begin{aligned}
 f_{msf}(\langle t \rangle) &= \begin{cases} 0 & \text{para um nó arbitrário } t = t_0 \\ +\infty & \text{no caso contrário.} \end{cases} \\
 f_{msf}(\pi_s \cdot \langle s, t \rangle) &= d(s, t).
 \end{aligned} \tag{2}$$

A condição $V(t) > V(s)$ na linha 6 não é mais válida. Nós precisamos manter o *status* de cada nó em relação à fila Q , tal que $status(t) = 0$, se t nunca foi inserido em Q , $status(t) = 1$ se $t \in Q$, e $status(t) = 2$ se t já foi removido de Q na linha 5. O mapa L também não é necessário e os protótipos em S^* (inicialmente vazio e agora como saída do algoritmo) podem ser identificados entre as linhas 5 e 6.

- Se $P(s) \neq nil$ então
- Se $\lambda(s) \neq \lambda(P(s))$ então

- $S^* \leftarrow S^* \cup \{s, P(s)\}$.

Note que a união evita a inserção duplicada de nós em S . O algoritmo fica.

Algorithm 2 – ALGORITMO MST

INPUT: Conjunto Z_1 , par (v, d) para extração de características e cálculo de distâncias.

OUTPUT: Árvore de peso mínimo P , mapa de valores V e protótipos S^* .

AUXILIARY: Fila Q de prioridades, $status()$ e variável tmp .

1. Para cada $s \in Z_1 \setminus t_0$, faça $V(s) \leftarrow +\infty$ e $status(s) \leftarrow 0$.
2. Faça $V(t_0) \leftarrow 0$, $P(t_0) \leftarrow nil$, $status(t_0) \leftarrow 1$, e insira t_0 em Q .
3. Enquanto Q não for vazia, faça
4. Remova de Q uma amostra s tal que $V(s)$ é mínimo e faça $status(s) \leftarrow 2$.
5. Se $P(s) \neq nil$, então
6. | Se $\lambda(s) \neq \lambda(P(s))$, então
7. | | $S^* \leftarrow S^* \cup \{s, P(s)\}$.
8. Para cada $t \in Z_1$ tal que $t \neq s$ e $status(t) \neq 2$, faça
9. | Calcule $tmp \leftarrow d(s, t)$.
10. | Se $tmp < V(t)$, então
11. | | Se $status(t) = 1$, então remova t de Q .
12. | | $P(t) \leftarrow s$ e $V(t) \leftarrow tmp$.
13. | | Insira t em Q e faça $status(t) \leftarrow 1$.

As Figura 1a- 1c ilustram um exemplo.

1.2 Classificação

A classificação de uma nova amostra t considera todas as conexões entre t e amostras s de treinamento, como se t fizesse parte do grafo original. Seja S^* o conjunto de protótipos encontrado pela MST, o caminho ótimo $P^*(t)$ de S^* até t é encontrado incrementalmente pela avaliação do valor ótimo $V(t)$ (Figura 1d).

$$V(t) = \min\{\max\{V(s), d(s, t)\}\}, \forall s \in Z_1. \quad (3)$$

Seja $s^* \in Z_1$ o nó que satisfaz a equação (i.e., $P(t) = s^*$). Dado que $L(s^*) = \lambda(R(t))$, a classificação simplesmente associa $L(s^*)$ como sendo a classe de t (Figuras 1e e 1f). Um erro ocorre quando $L(s^*) \neq \lambda(t)$.

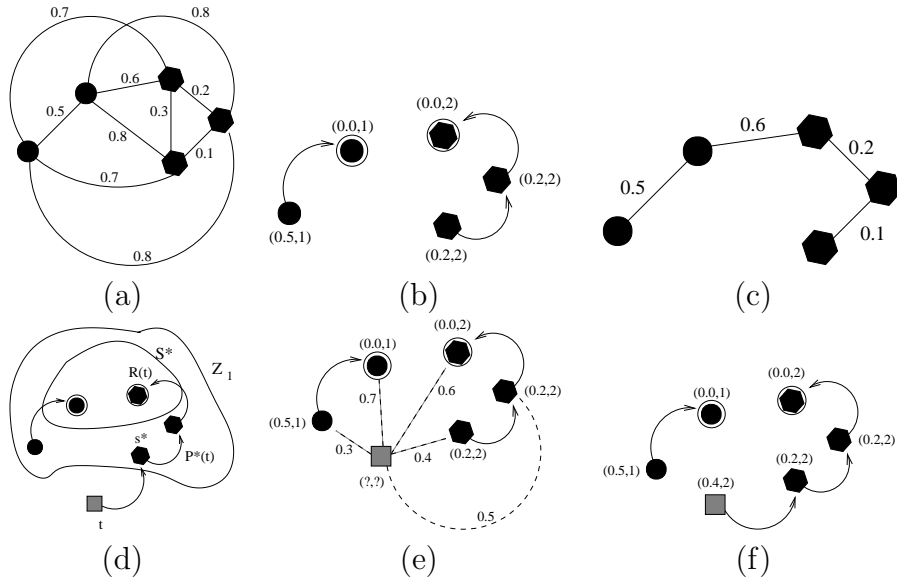


Figura 1: (a) Grafo completo com as distâncias nos arcos e dois protótipos. (b) Floresta de caminhos ótimos onde os protótipos estão circulados. As entradas (x, y) indicam valor ótimo e rótulo das amostras. (c) MST usada para selecionar protótipos. (d) A classificação de uma nova amostra (quadrado cinza) t associa o caminho ótimo $P^*(t)$ de $R(t) \in S^*$ até t passando por s^* . (e) Exemplo da classificação de t onde as linhas tracejadas indicam conexões candidatas. (f) Caminho ótimo com valor 0.4 e rótulo 2 atribuído a t . Mesmo t estando mais próxima de uma amostra circular, ela é classificada na classe hexagonal.