Fakulta Aplikovaných věd

-

Zpracování digitálního obrazu
(seminární práce)

Jakub Nedvěd (A13N0141P)

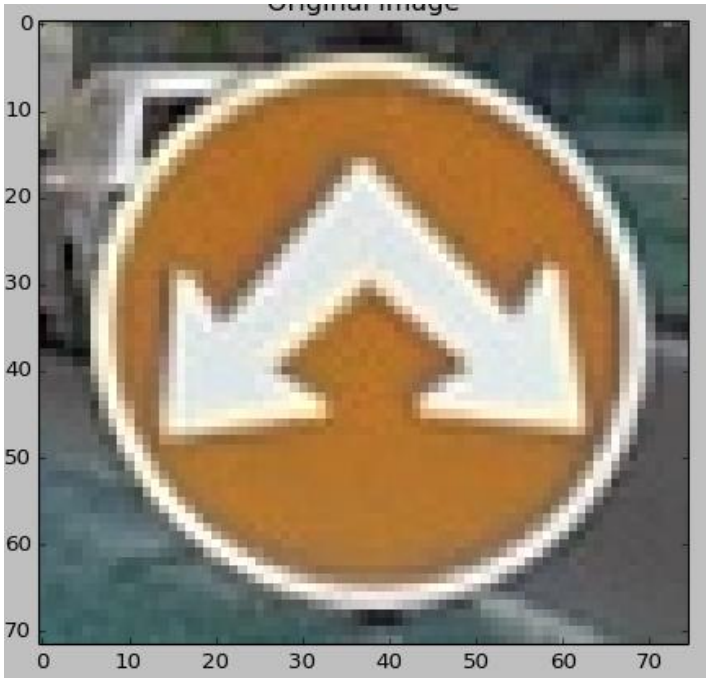
Rozpoznávání značek

nedvedj@students.zcu.cz

Plzeň 2014

Tvorba vektoru příznaků

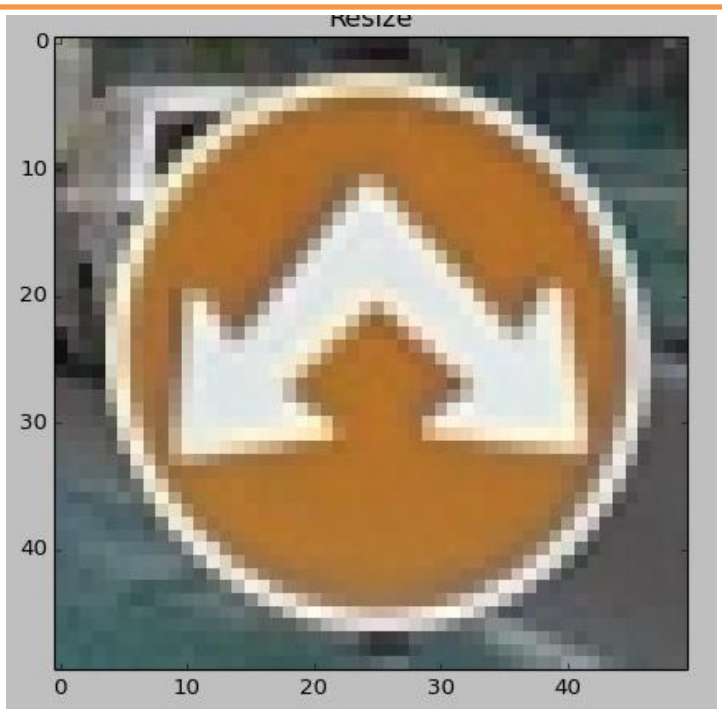
Pro tvorbu vektoru příznaků jsem využil 2 metody. Nejdříve však před samotném zpracovávání obrazu byla provedena změna velikosti, čímž je zaručeno, že se metoda může použít na jakkoli velký vstupní obrázek. Zvolil jsem rozměry 50x50, které jsou pro reprezentaci obrazu dostatečné a nedochází k příliš velkému zkreslení při zvětšování velmi malých značek.



Obr. 1: Vstupní obraz

Pro změnu rozměrů je vhodné použít funkci transform

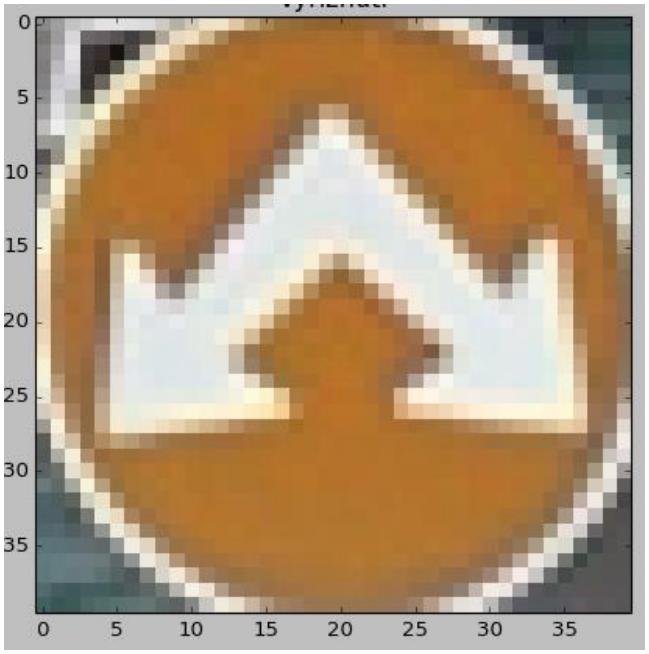
```
skimage.transform.resize(im, [50, 50])
```



Obr. 2: Změna rozměrů obrazu

První metoda vyřízne oblast kolem středu obrazu, díky čemuž z obrázku zmizí nežádoucí objekty pozadí, jako např. obloha, stromy, auta,…. Oblast kolem středu má rozměry 40x40, což se sice nezdá jako velký rozdíl oproti 50x50, ale musíme myslet, že při konstrukci vektoru příznaků to bude hrát velikou roli, jelikož 50x50 = 2 500 obrazových bodů a 40x40 = 1 600 bodů – rozdíl 900 je již patrný a pro trénování klasifikátoru se může jevit jako rozhodující.

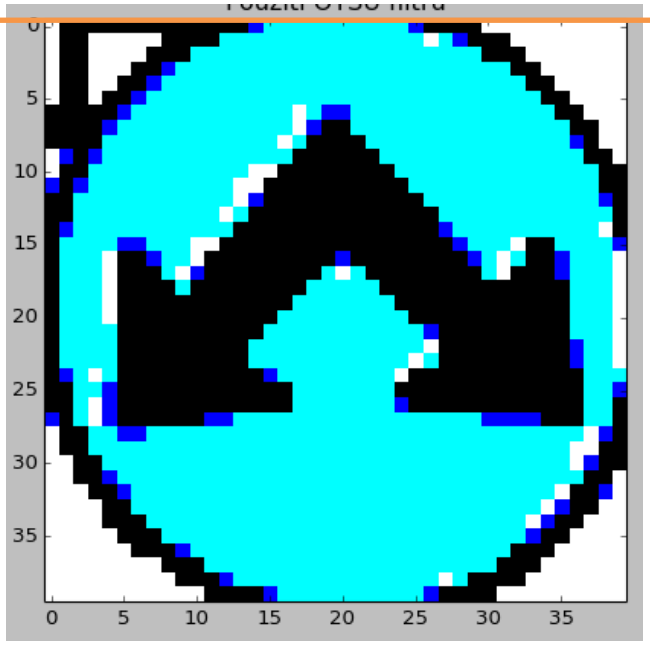
```
im [int(im.shape[0]/2)-20:int(im.shape[0]/2)+20, int(im.shape[1]/2)-20:int(im.shape[1]/2)+20]
```



Obr. 3: Vyříznutí čtvercové plochy kolem středu obrázku

Druhou metodou je využití Otsuova filtru, který najde optimální mez pro daný obraz pro segmentaci.

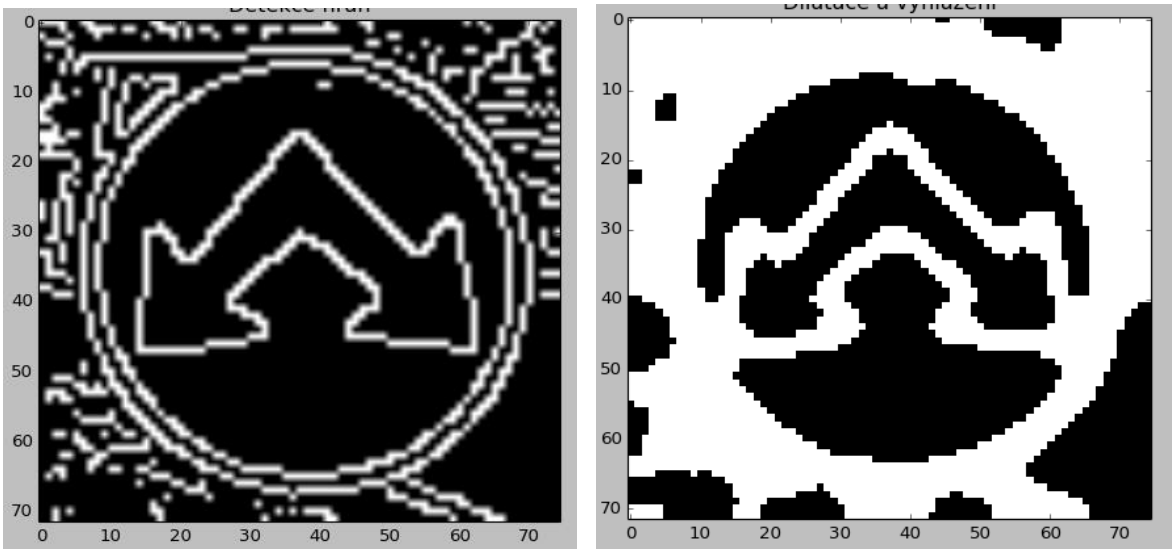
```
threshold = filter.threshold_otsu(im)
im = im < threshold
```



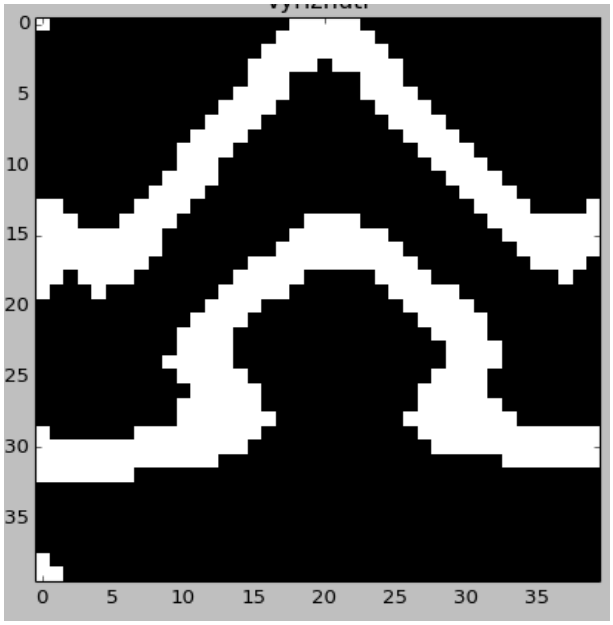
Obr. 4: Použití Otsu filtru

Při zpracování jsem chtěl také použít detekci hran, bohužel Bayesův klasifikátor mi nechtěl brát příznaky z černobílého obrázku. Při využití této metody, jsem na obrázek využil dilataci a dále filtroval pomocí Gausovo filtru, díky čemuž se z obrázku odstranili nežádoucí objekty.

```
im = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
im = cv2.Canny(im,60,60)
kernel_big = skimage.morphology.diamond(1)
im = skimage.morphology.binary_dilation(im, kernel_big)
im = cv2.GaussianBlur(im,(5,5), 5)
```



Obr. 5: Detekce hran a následná dilatace + vyhlazení přes Gausův filtr



Obr. 6: Vyříznutí kolem středu

Trénování klasifikátoru

Jelikož program musí běžet maximálně vteřinu pro správnou klasifikaci, je důležité mít klasifikátor, který správně a rychle rozhodne.

Původně jsem zamýšlel užití metody křížové validace, která rozděluje data na několik setů a dále s nimi pracuje.

Křížová validace (cross validation) – rozdělení dat

Data se rozdělí do k-shluků, v našem případě 5, 4 se využijí na trénování klasifikátoru a poslední se využije pro testování.

Toto testování by však potřebovalo rovnoměrné rozložení dat, kdy bychom mohli snadno určit práh, od kterého se budou brát data jako trénovací a od kterého jako testovací. Bohužel zde by volba prahu byla složitější a potřebovala by předzpracování dat.

Místo této metody jsem se tedy rozhodl využít vhodnějšího postupu a to sestavení korpusu dat pouze z každého n-tého obrázku a následné otestování na podobně vytvořených datových korpusech. Postupným zvyšováním kroku a tedy snižováním velikosti korpusu zjistíme úspěšnost pro jednotlivé velikosti.

Trénování na korpusu training3, testování na training3

Trénování	Testování			
n-tý obraz	**12-tý**	**16 -tý**	**20-tý**	**Průměr**
5	61,13%	60,36%	84,93%	68,81%
10	52,93%	53,01%	92,47%	66,14%
12	93,58%	56,39%	56,32%	68,76%
15	45,27%	35,30%	53,16%	44,57%
17	39,21%	37,59%	36,89%	37,89%
20	44,35%	43,73%	97,74%	61,94%

Trénování na korpusu training3, testování na training

Trénování	Testování			
n-tý obraz	**12-tý**	**16 -tý**	**20-tý**	**Průměr**
5	53,86%	53,23%	53,81%	53,63%
10	42,62%	42,74%	40,80%	42,05%
12	39,73%	36,37%	38,45%	38,18%
15	30,31%	30,46%	31,42%	30,73%
17	31,36%	31,49%	31,88%	31,58%
20	30,80%	29,43%	28,73%	29,65%

Z tabulek s výsledky je patrné, že klasifikátor dosahuje mnohem lepší výsledky na korpusu, kde probíhalo trénování, než na zcela odlišných datech. Za povšimnutí stojí, že při trénování a testování na naprosto totožných souborech dosahuje klasifikátor úspěšnosti maximálně 97,74% a to z důvodu, že se při trénování a testování bere v potaz pouze každý 20 soubor a datový korpus je tedy znatelně menší, než např. při každém 12 souboru.

Z první tabulky je zřejmé, že nejlepších výsledků bylo dosaženo při trénování po 5,12 a 20 souborech a klasifikátor si vedl celkem dobře i v druhém testovacím korpusu. Nyní zkusíme zjistit, zda budou lepší výsledky při trénování z dat training a testování na training a training 3 pro tyto opakování.

Trénování na korpusu training, testování na training

Trénování	Testování			
n-tý obraz	**12-tý**	**16 -tý**	**20-tý**	**Průměr**
5	56,42%	54,31%	62,36%	57,70%
12	90,72%	52,58%	54,28%	62,86%
20	41,98%	40,11%	96,60%	59,56%

Trénování na korpusu training, testování na training3

Trénování	Testování			
n-tý obraz	**12-tý**	**16 -tý**	**20-tý**	**Průměr**
5	49,68%	49,27%	49,39%	49,47%
12	48,27%	44,94%	46,08%	46,43%
20	37,40%	38,19%	36,60%	37,33%

Z naměřených hodnot můžeme vidět, že nejlepší je natrénovat klasifikátor z datového korpusu **training3**, při trénování **po 5 souborech**, za předpokladu, že nám nezávisí na velikosti dat. Pokud budeme brát v potaz co nejmenší velikost klasifikátoru, využili bychom datový korpus training, při trénování po 12 souborech.