Toward Retail Product Recognition on Grocery Shelves

Gül Varol
gul.varol@boun.edu.tr

Boğaziçi University, İstanbul, Turkey
İdea Teknoloji Çözümleri, İstanbul, Turkey

Rıdvan S. Kuzu
ridvan.salih@boun.edu.tr

Boğaziçi University, İstanbul, Turkey
İdea Teknoloji Çözümleri, İstanbul, Turkey

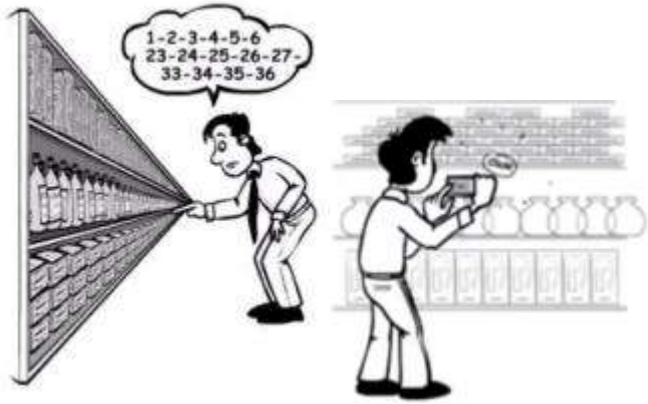
Outline

- Introduction
- Dataset
- Methodology
- Experiments
- Conclusions

Introduction

- What is a planogram?
  - Visual description of the retail products' placement on shelves
- Our goal
  - Automatic planogram compliance check in groceries
  - Recognition of product positions on shelf images

betterretailing.com



mathecsys.com/automatic-planogram-generation

Related work

- Planogram compliance from images through SIFT descriptors, OCR etc.
  - Opalach et al., “Planogram extraction based on image processing,” Patent US 8 189 855, May 29, 2012.
- Real-time product detection system from video
  - Winlock et al., “Toward real-time grocery detection for the visually impaired,” in Computer Vision and Pattern Recognition Workshops (CVPRW), 2010.
- Macthing database images on an input image using SIFT vectors
  - Auclair et al., “How to use SIFT vectors to analyze an image with database templates.” in Adaptive Multimedia Retrieval, ser. Lecture Notes in Computer Science, Springer, 2007.

Dataset* overview

- Tobacco images from ~40 groceries
- 4 different cameras
- Different number of shelves

Number of shelves	2	3	4	5	6	7	Total
Number of images	92	137	83	34	6	2	**354**

- Over 13,000 cropped product images on 354 shelf images
- 10 brand classes
- ~300 training images for each class
- Full annotation

*available upon request

Samples from the dataset

A sample rack of 4 shelves photographed as a whole (top-left) and as in pieces (the rest)

Various images of one brand

Overview of brand classes

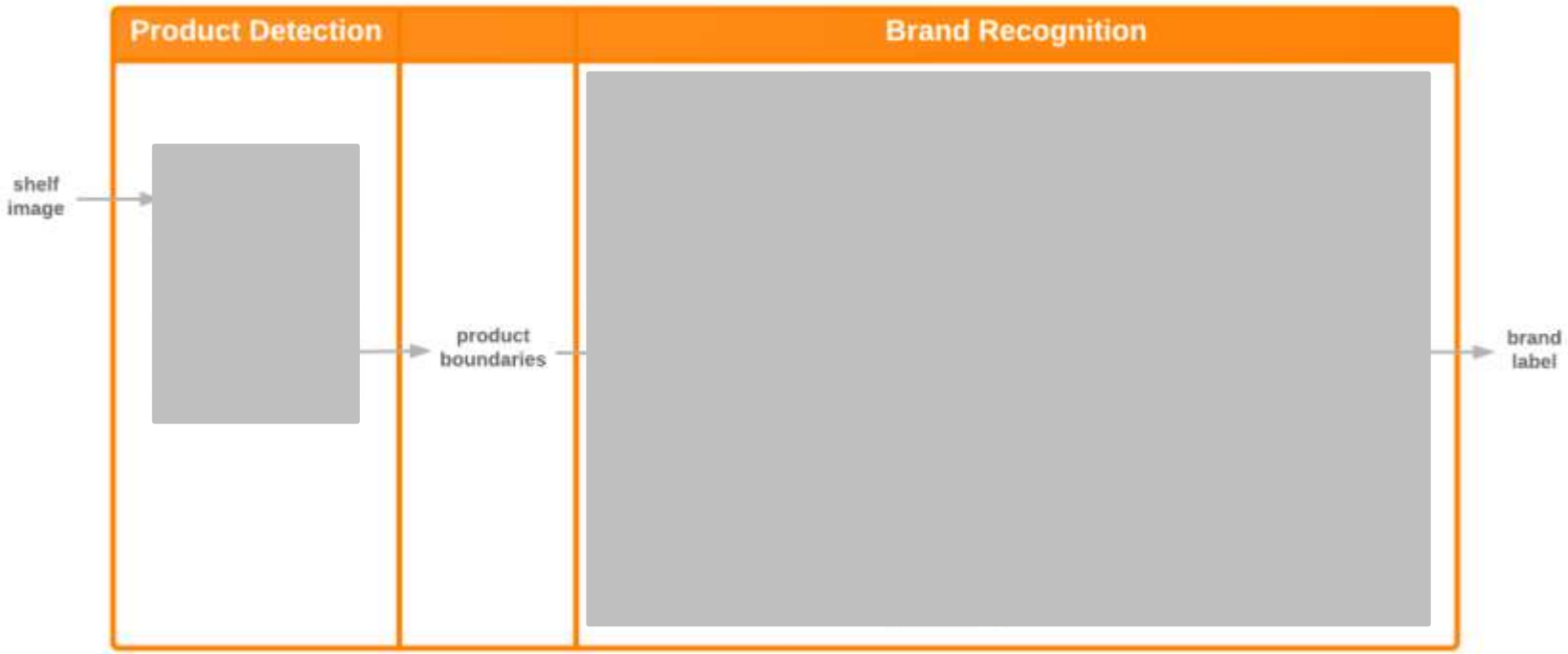
Annotation

automatic cropping
(training images)

manual cropping
(test images)

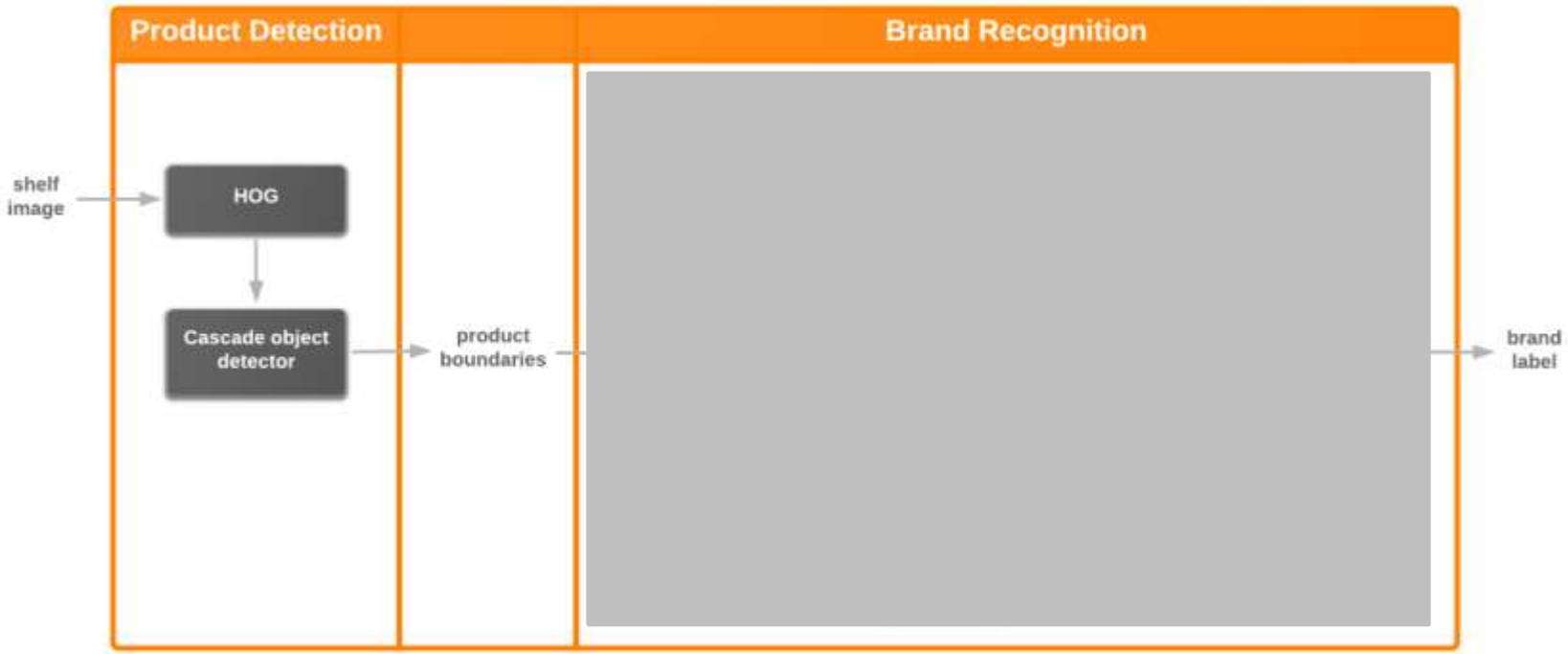
Pipeline

- Segmentation
  - Shelf detection (optional)
  - Product detection
- Brand recognition

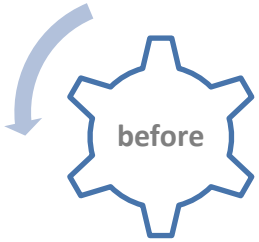


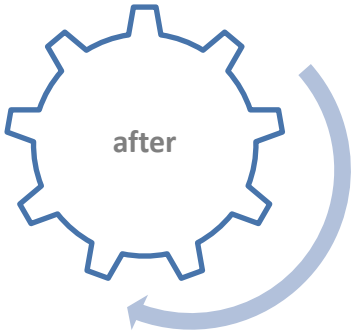
Product detection

- Viola-Jones object detection framework (**cascade object detection** with AdaBoost learning algorithm)
- Histogram of Oriented Gradients (**HOG**)
- Constraints for eliminating false alarms

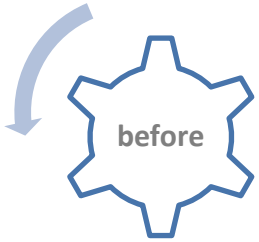


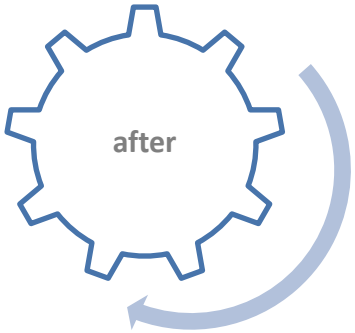
Improvement through constraints



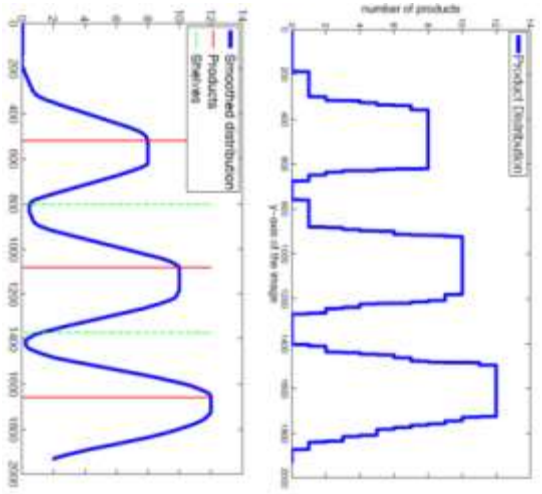


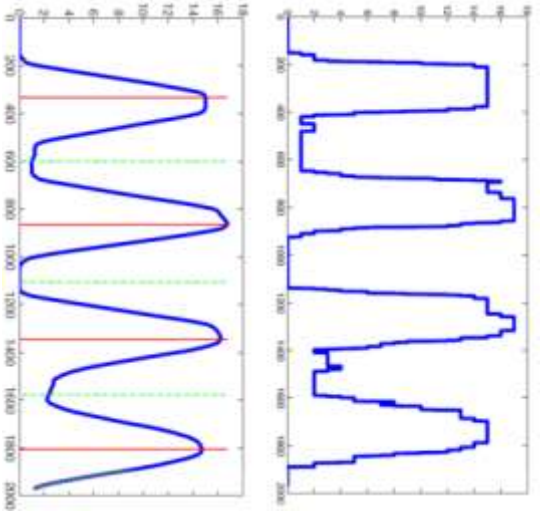
Improvement through constraints





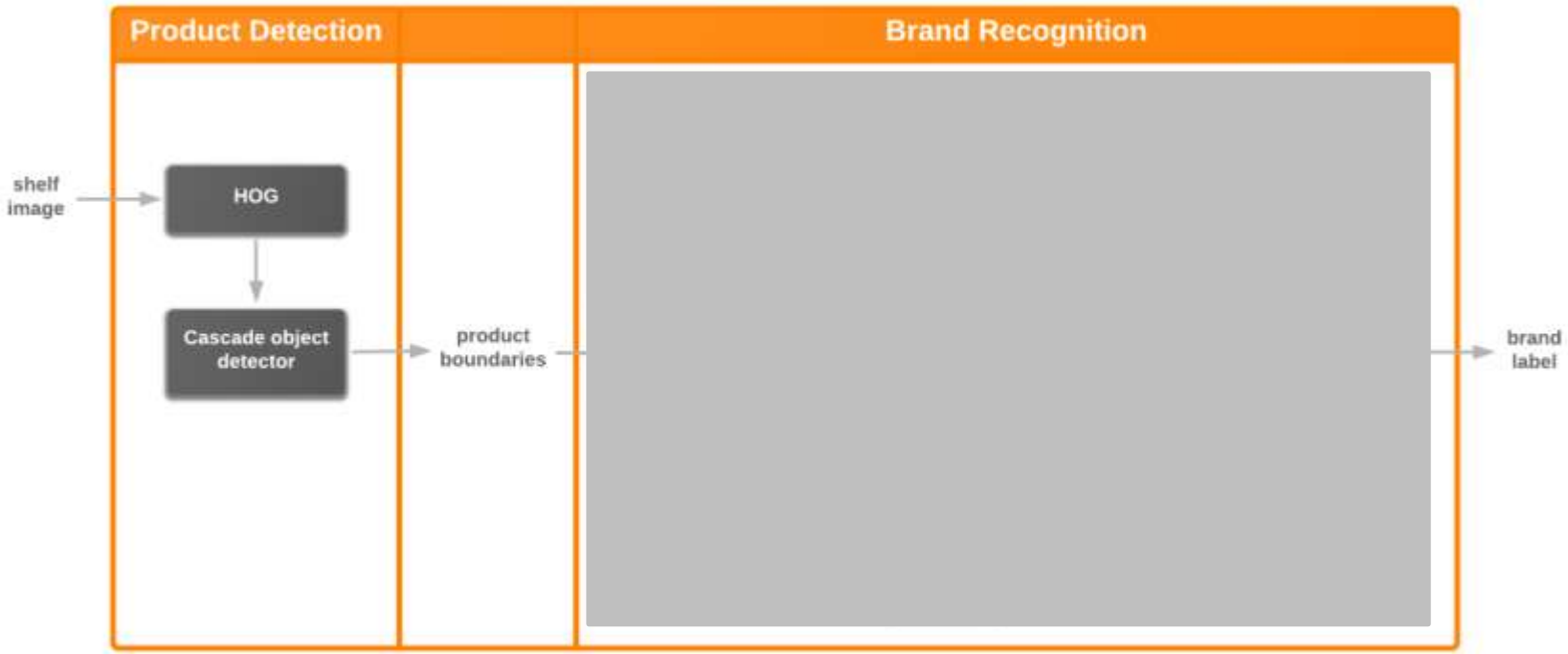
Shelf detection





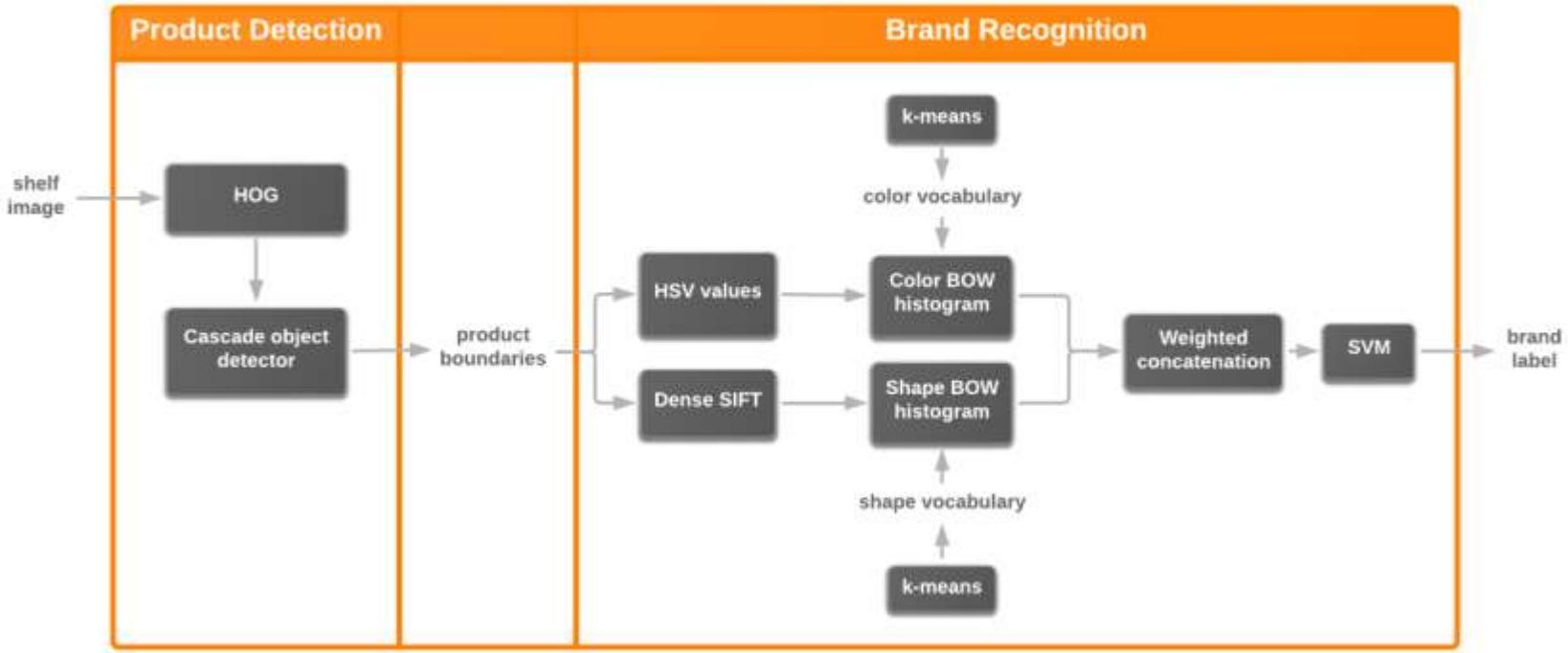
Brand recognition

- Bag of Words (BOW)
  - 2 separate dictionaries with k-means clustering
- Weighted concatenation of color & shape histograms
- Support Vector Machine (SVM)



Brand recognition

- Bag of Words (BOW)
  - 2 separate dictionaries with k-means clustering
- Weighted concatenation of color & shape histograms
- Support Vector Machine (SVM)



Experiments

- Product detection performance
- Brand recognition performance

Product detection performance

sample positive instances

sample negative instances

- Training (375 positive, 195 negative instances)
- Test (over 13,000 positive instances)

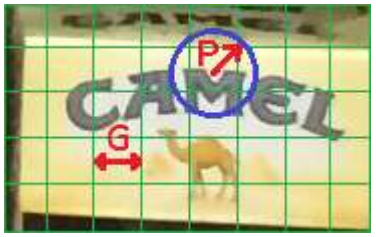
	Recall (%)	Precision (%)
no constraint	89	69
min-max height	94	75
width-height deviation	94	81

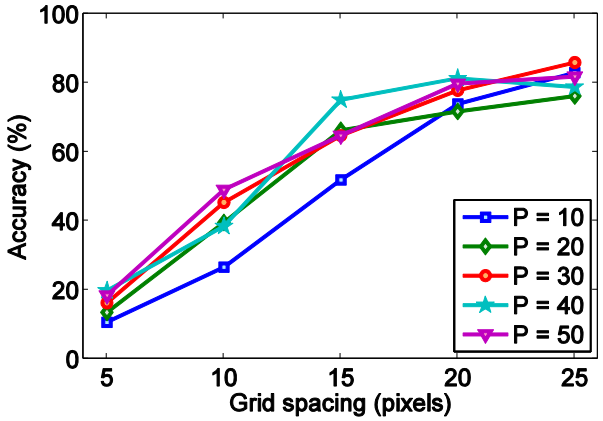
Brand recognition experiments

- The classification results on ~1000 images

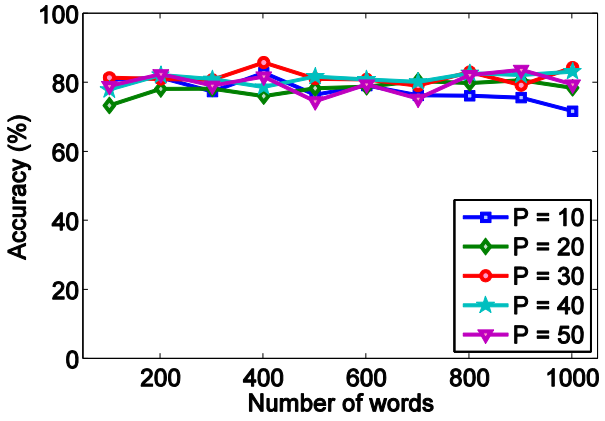
Shape features

- Dense SIFT parameters
  - Grid spacing (G)
  - Patch size (P)
- Number of words (V_s)

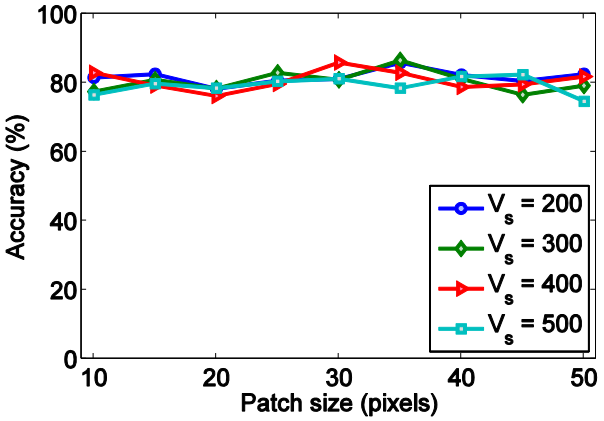




V_s=400



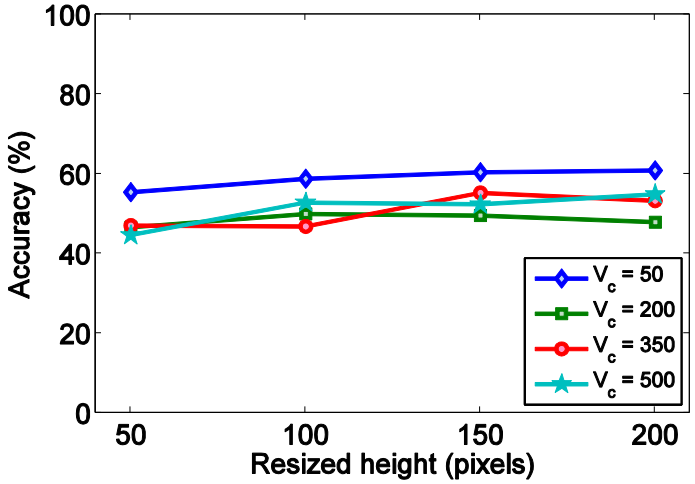
G=25

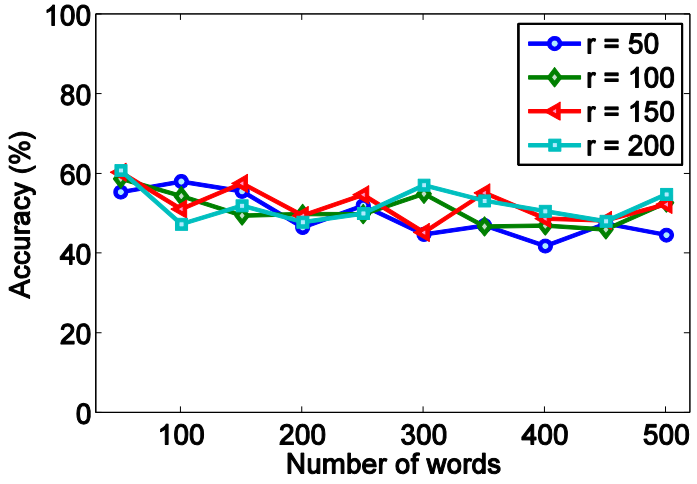


G=25

Color features

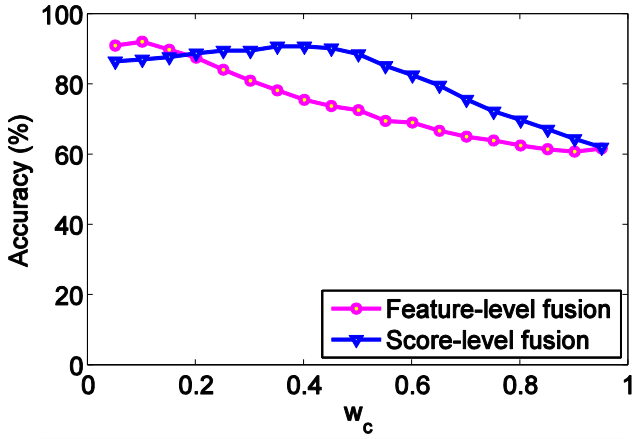
- Image resizing height (r)
- Number of words (V_c)





Combined model

- Results for brand recognition (G=25, P=30, Vs=400, Vc=50)



	Color (HSV)	Shape (SIFT)	Combined
Accuracy	60.5 %	85.9 %	92.3 %

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
C_1	96	0	0	1	0	0	0	0	0	3
C_2	0	95	0	1	0	2	0	0	0	2
C_3	0	1	65	0	0	0	0	0	0	1
C_4	1	7	0	83	0	0	3	1	1	4
C_5	0	0	0	0	93	2	1	0	1	3
C_6	1	0	0	0	0	95	1	0	2	1
C_7	0	0	0	0	0	0	100	0	0	0
C_8	0	0	0	0	1	0	0	98	0	1
C_9	0	0	0	0	0	0	0	0	62	13
C_{10}	0	0	0	0	6	3	1	0	6	59

confusion matrix

Conclusions

- Optical character recognition
- Handling *null* class
- Preprocessing (e.g. noise reduction, perspective correction)

Questions?