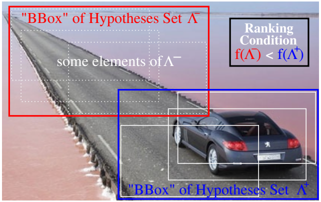
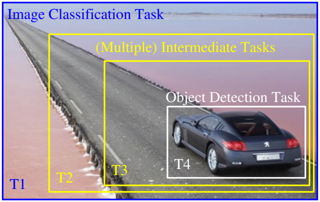




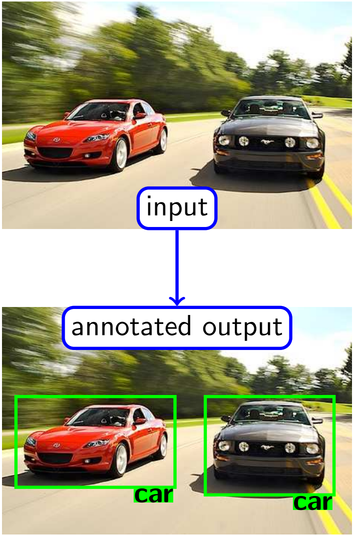
Branch&Rank: Efficient, Non-Linear Object Detection

Alain D. Lehmann,
Peter V. Gehler, and Luc Van Gool

Computer Vision Laboratory, ETH Zurich, Switzerland

30 August, 2011

Detection means to **localise and categorise** objects

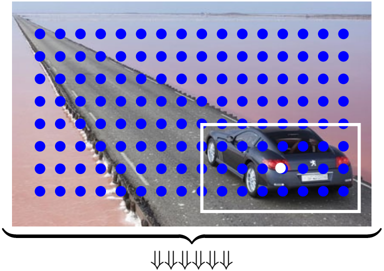


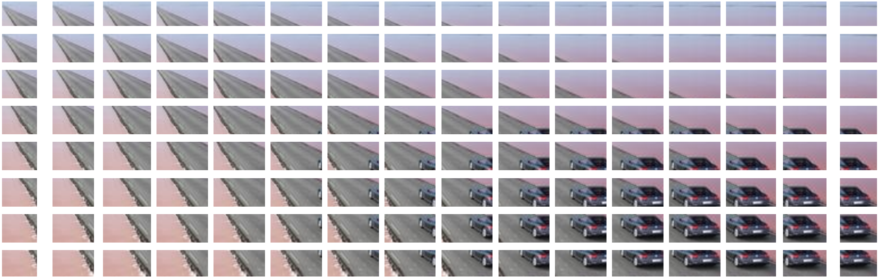
Appearance variations make it difficult

- intra-class variations
- different views/poses
- illumination changes
- occlusions, *etc.*

sophisticated **expensive** models

Localise objects among thousands of hypotheses





Search space size

- $>10'000$ locations
- $>1'000$ classes

avoid exhaustive enumeration

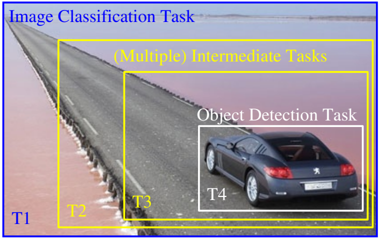
Efficient detection by ranking sub-images

Runtime = (classifier cost) × (#calls)

- reduce cost: cascades [Viola *et al.* 04, Vedaldi *et al.* 09]
  - *exhaustive search →not scalable*
- reduce calls: branch&bound [Lampert *et al.* 08, Lehmann *et al.* 09]
  - *bounds not tight enough →not effective*

Ranking: "learn the bound"

- branch, but not bound
- often <100 classifier calls →non-linear SVMs
- classification for detection



Outline

1. Detection: *best-first search*
2. Training: *ranking hypothesis sets*
3. Multi-tasks aspects
4. Results and conclusion

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

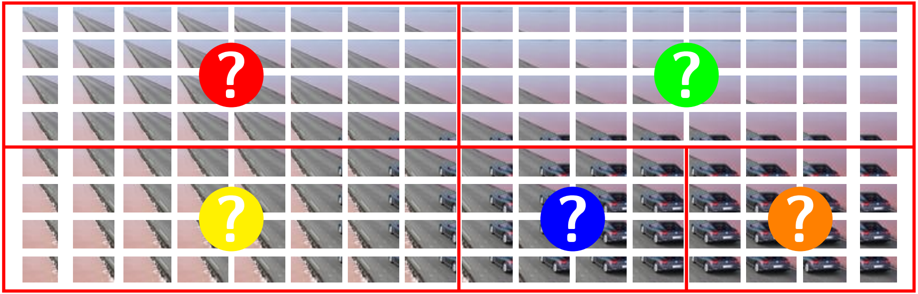
- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning

Sets of hypothesis

- exploit correlations
- split promising sets

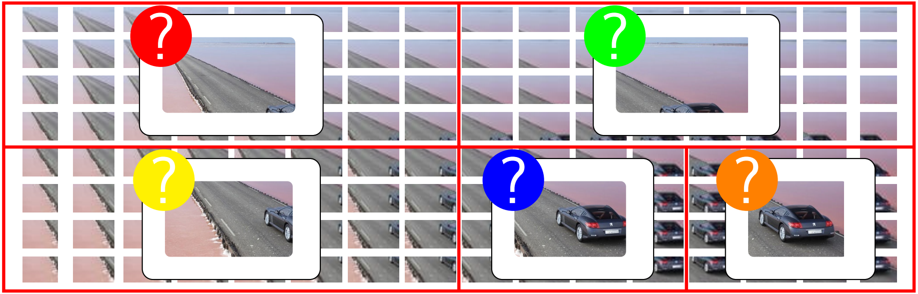
Efficiency by means of adaptive partitioning



Sets of hypothesis

- exploit correlations
- split promising sets

Efficiency by means of adaptive partitioning



Sets of hypothesis

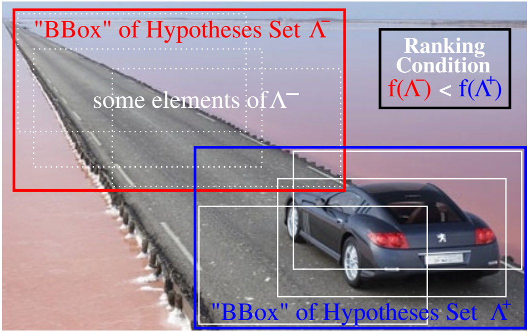
- exploit correlations
- split promising sets
- correspond to subimages

Ranking function f prioritises

$$f\Big(\underbrace{\qquad}_{\textit{contains obj.}}\Big) > f\Big(\underbrace{\qquad}_{\textit{no object}}\Big)$$

- supersedes upper bounds

Training with sets for increased efficiency



Structured SVM ranking [Tsochantaridis *et al.* 04, Blaschko *et al.* 08]

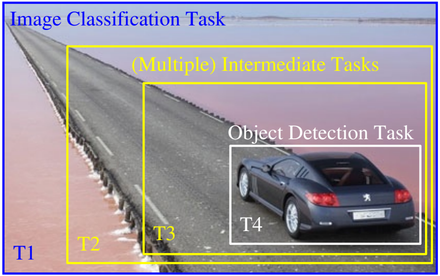
$$\min_{w,\xi_i \geq 0} \|w\|^2 + C \sum_i \xi_i$$

$$f(\Lambda_i^+) - f(\Lambda^-) \geq \Delta(\Lambda^-) - \xi_i$$

with $f(\Lambda) = \langle w, \phi(\Lambda) \rangle$

- bag-of-words descriptor $\phi(\Lambda)$
- kernelize with RBF-χ^2 kernel
- Λ^+: generate with oracle
- Λ^-: delayed constraint generation

From image classification to object categorisation



Large sets

- object somewhere
- image classification

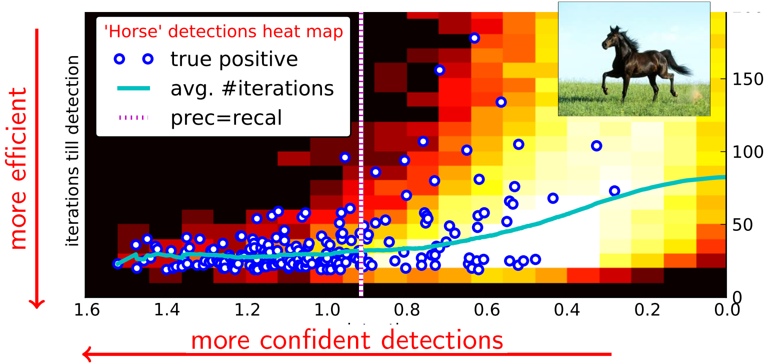
Small sets

- object centred
- object categorisation

Task-adapted ranking $f(\Lambda) = \langle w_{q(\Lambda)}, \phi(\Lambda) \rangle$

- task mapping $q(\Lambda)$
- leverage set information
- exploit context
- improved AP by $\approx 10\%$

Branch&rank detects in often <50 iterations



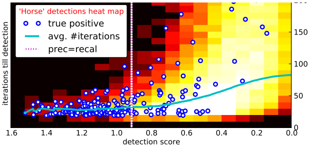
Dataset: PASCAL VOC 2007 (Horses) [Everingham *et al.*, 2007]

- non-linear RBF-χ^2 SVMs
- no cascade approximations

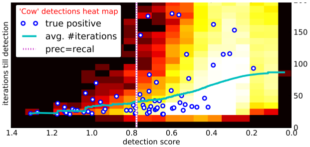
costly classifier feasible

More results

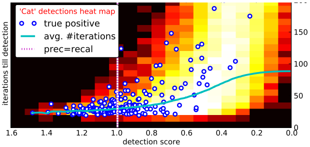
Horse



Cow



Cat



	branch&rank [Lehmann *et al.* 2011]	part-based detector [Felzenszwalb *et al.* 2008]	best in challenge [Everingham *et al.* 2007]	
Horse	**36.8%**	30.1%	33.5%	better
Cow	10.8%	**16.5%**	14.0%	worse
Cat	17.6%	11.0%	**24.0%**	in-between

- Future work
  - *combine multiple features*
  - *use task-adapted features*

Conclusion

- **Branch&rank is efficient**
  - *less than 100 classifier calls*
  - *non-linear SVMs feasible*
- **Process hypothesis sets**
  - *during detection and training*
  - *"learn the bound"*
- **Multiple task**
  - *combine classification and detection*