Accelerated Training of Linear Object Detectors

Charles Dubout
François Fleuret

Idiap Research Institute
EPFL

June 24, 2013

Plan

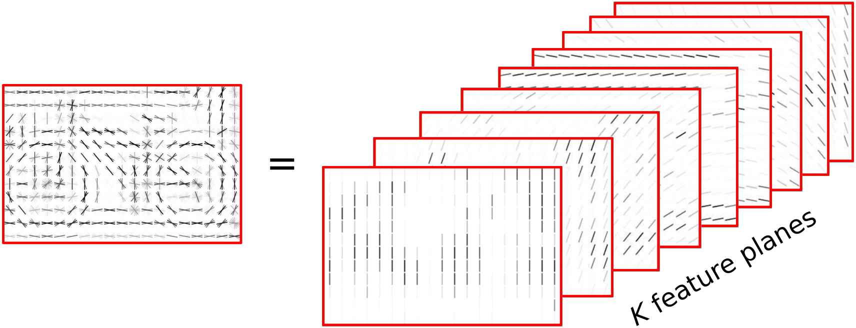
Exact Acceleration of Linear Object Detectors (ECCV 2012)

- Feature planes
- Standard convolution process
- Fourier convolution processes

Accelerated Training of Linear Object Detectors

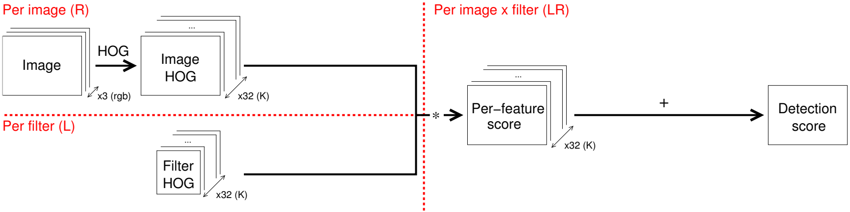
- Linear object detector
- Fourier based gradient computation process
- Results

Feature planes

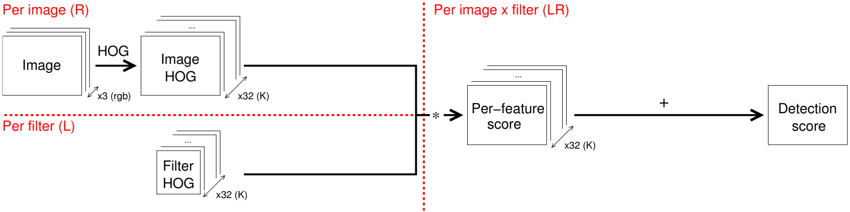


The image features can be seen as organized in planes, containing distinct features from each grid cell.

Standard convolution process



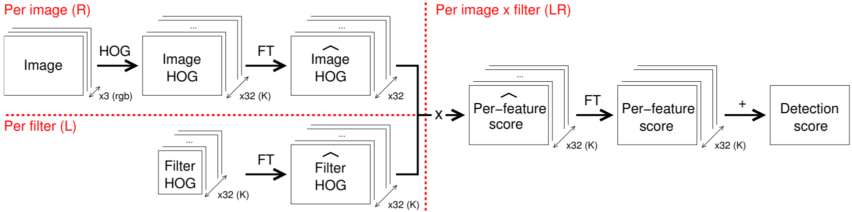
Standard convolution process



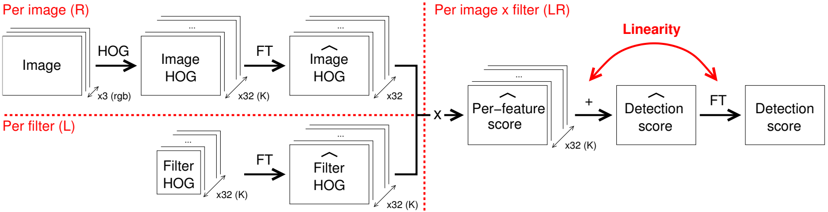
The computational cost to convolve a HOG image of size $M \times N$ with L filters of size $P \times Q$ across K features is:

$$C_{\text{std}} = \mathcal{O}(KLMNPQ)$$

Fourier based convolutions



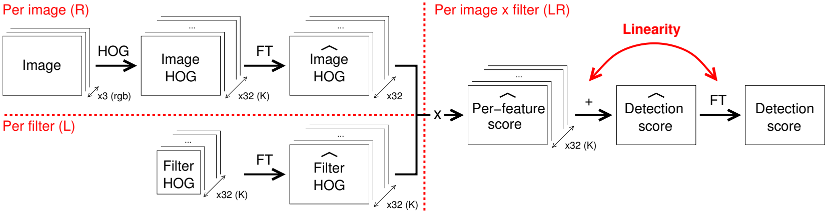
Fourier based convolutions



The computational cost to convolve a HOG image of size $M \times N$ with L filters of size $P \times Q$ across K features is:

$$C_{\text{FFT}} = \underbrace{\mathcal{O}(KMN \log MN)}_{\text{Forward FFTs}} + \underbrace{\mathcal{O}(KLMN)}_{\text{Multiplications}} + \underbrace{\mathcal{O}(KLMN \log MN)}_{\text{Inverse FFTs}}$$

Fourier based convolutions



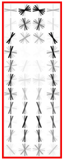
The computational cost to convolve a HOG image of size $M \times N$ with L filters of size $P \times Q$ across K features is:

$$C_{\text{opt}} = \underbrace{\mathcal{O}(KMN \log MN)}_{\text{Forward FFTs}} + \underbrace{\mathcal{O}(KLMN)}_{\text{Multiplications}} + \underbrace{\mathcal{O}(\cancel{K}LMN \log MN)}_{\text{Inverse FFTs}}$$

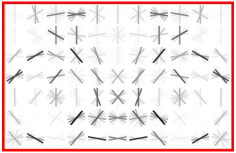
$$\approx \mathcal{O}(KLMN)$$

Linear object detector

Pedestrian template



Bicycle template



- The simplest form of the detection score is:

$$f_r(i,j) = \sum_{a,b} w(a,b)\, \Phi_r(i+a, j+b)$$

where $\Phi_r(i,j)$ is the feature at location (i,j) and $\mathbf{w}$ are the model weights.

- Can be extended to more complex models: mixtures, DPMs, etc.

Per sample loss

- We consider the data-driven term of training loss to be of the form

$$L(\mathbf{w}) = \sum_r \sum_{i,j} l(y_r(i,j) f_r(i,j))$$

  where $y_r(i,j) \in \{-1, 1\}$ is the label of the sub-window located at (i,j) in image r, and l is the per sample loss.

- Typical per sample losses are: hinge loss, logistic loss, exponential loss, etc.

Gradient of the loss

$$\nabla L(a,b) = \frac{\partial L(\mathbf{w})}{\partial w(a,b)} \tag{1}$$

$$= \sum_r \sum_{i,j} \frac{\partial l(y_r(i,j) f_r(i,j))}{\partial w(a,b)} \tag{2}$$

$$= \sum_r \sum_{i,j} y_r(i,j)\, l'(y_r(i,j) f_r(i,j))\, \frac{\partial f_r(i,j)}{\partial w(a,b)} \tag{3}$$

$$= \sum_r \sum_{i,j} \overline{(\mathbf{y}_r \cdot l'(\mathbf{y}_r \cdot \mathbf{f}_r))}(i,j)\ \Phi_r(a-i, b-j) \tag{4}$$

$$= \left(\sum_r \overline{\mathbf{y}_r \cdot l'(\mathbf{y}_r \cdot \mathbf{f}_r)} * \mathbf{\Phi}_r \right)(a,b) \tag{5}$$

Gradient of the loss

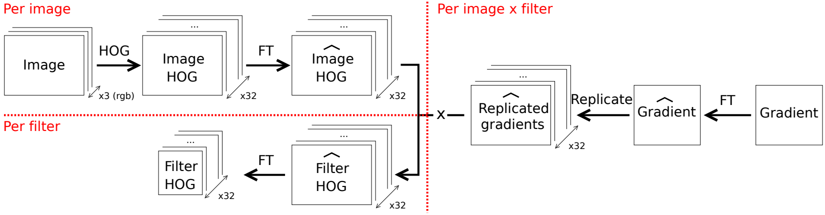
- Hence:

$$\nabla L = \sum_r \overline{\mathbf{y}_r \cdot l'(\mathbf{y}_r \cdot \mathbf{f}_r)} * \mathbf{\Phi}_r$$

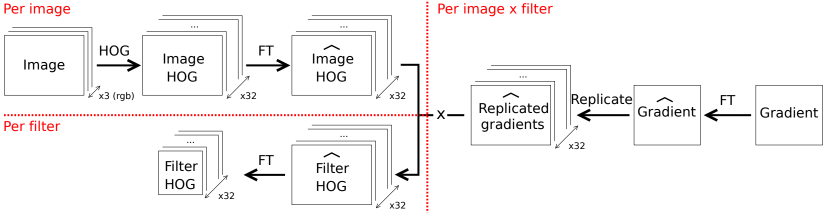
  where $\overline{\alpha}(i,j) = \alpha(-i,-j)$ and the operator $\cdot$ stands for the pointwise multiplication.

- The gradient can thus also be computed using the Fourier transform.

Fourier based gradient computation process



Fourier based gradient computation process



The computational cost to compute the gradient of L filters of size $P \times Q$ over R HOG images of size $M \times N$ with K features is:

$$C_{\text{opt}} = \underbrace{\mathcal{O}(RLMN\log(MN))}_{\text{Forward FFTs of the derivatives}} + \underbrace{\mathcal{O}(KLRMN)}_{\text{Multiplications}} + \underbrace{\mathcal{O}(KLRMN\log(MN))}_{\text{Inverse FFTs of the derivatives}}$$

$$\approx \mathcal{O}(KLRMN)$$

Results

We trained a linear detector on the Pascal VOC 2007 dataset using the *hinge* loss, and report the gradient computation time.

	1 scene per mini-batch		10 scenes per mini-batch	
	Normal filters	**Large filters**	**Normal filters**	**Large filters**
Standard (ms)	41.3	70.9	390	699
Ours (ms)	7.2	7.4	33.1	33.1
Std. sparse (ms)	1.1	1.3	6.6	8.1

Table : Average time to compute the gradient of the loss for one stochastic gradient descent iteration.

Conclusion

+ Leverages the FT to make the overall training computational cost independent of the filters' sizes
+ Relieves all the constraints inherent to sparse and approximate methods
- Contrary to our previous contribution, not very useful when using the hinge loss/bootstrapping

Thank you for your attention!

Questions?

Contact me at `charles.dubout@idiap.ch`