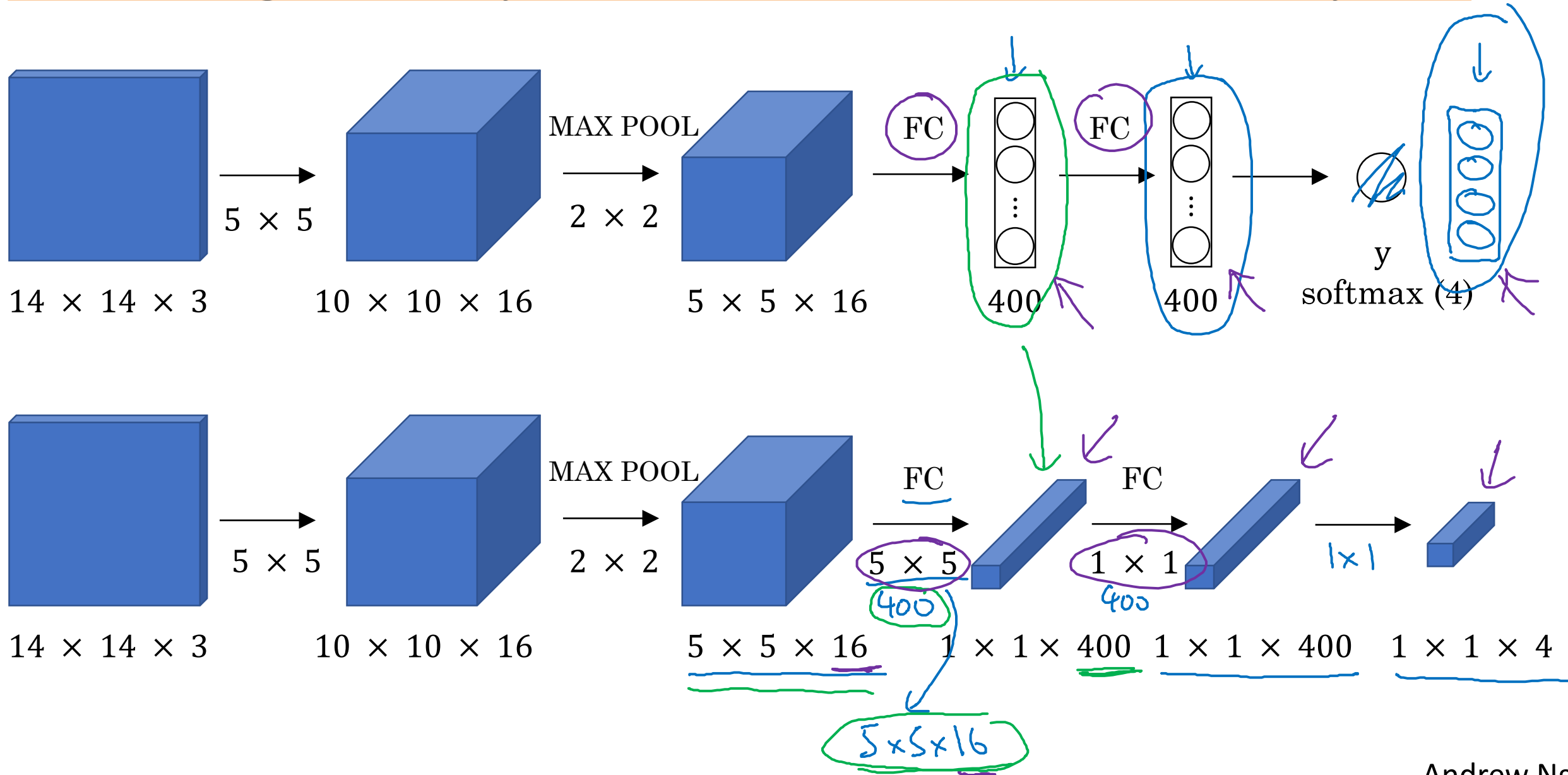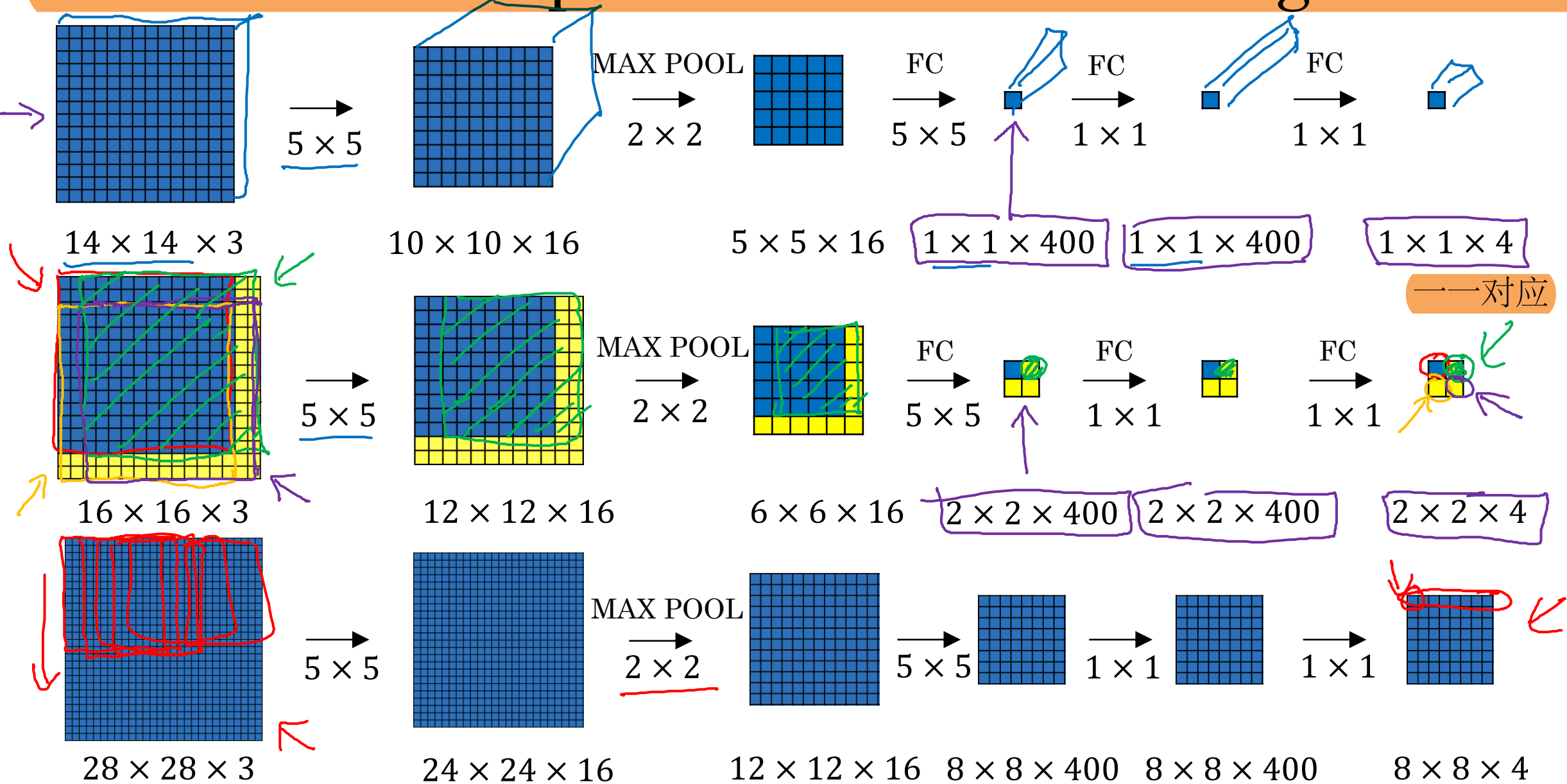deeplearning.ai

Object Detection

Convolutional implementation of sliding windows

Turning FC layer into convolutional layers

14 × 14 × 3   10 × 10 × 16   5 × 5 × 16   400   400   softmax (4)

5 × 5   MAX POOL 2 × 2   FC   FC   y

14 × 14 × 3   10 × 10 × 16   5 × 5 × 16   1 × 1 × 400   1 × 1 × 400   1 × 1 × 4

5 × 5   MAX POOL 2 × 2   FC 5 × 5 (400)   FC 1 × 1 (400)   1×1

5 × 5 × 16

Andrew Ng

$14 \times 14 \times 3$ → 5×5 → $10 \times 10 \times 16$ MAX POOL 2×2 → $5 \times 5 \times 16$ FC 5×5 → $1 \times 1 \times 400$ FC 1×1 → $1 \times 1 \times 400$ FC 1×1 → $1 \times 1 \times 4$

$16 \times 16 \times 3$ → 5×5 → $12 \times 12 \times 16$ MAX POOL 2×2 → $6 \times 6 \times 16$ FC 5×5 → $2 \times 2 \times 400$ FC 1×1 → $2 \times 2 \times 400$ FC 1×1 → $2 \times 2 \times 4$

一一对应

$28 \times 28 \times 3$ → 5×5 → $24 \times 24 \times 16$ MAX POOL 2×2 → $12 \times 12 \times 16$ → 5×5 → $8 \times 8 \times 400$ → 1×1 → $8 \times 8 \times 400$ → 1×1 → $8 \times 8 \times 4$

[Sermanet et al., 2014, OverFeat: Integrated recognition, localization and detection using convolutional networks]

Andrew Ng

Convolution implementation of sliding windows

$$28 \times 28 \quad \xrightarrow{5 \times 5} \quad 16 \times 16 \quad \text{MAX POOL} \xrightarrow{2 \times 2} \quad 12 \times 12 \quad \xrightarrow{5 \times 5} \quad 8 \times 8 \times 400 \quad \xrightarrow{1 \times 1} \quad 8 \times 8 \times 400 \quad \xrightarrow{1 \times 1} \quad 8 \times 8 \times 4$$
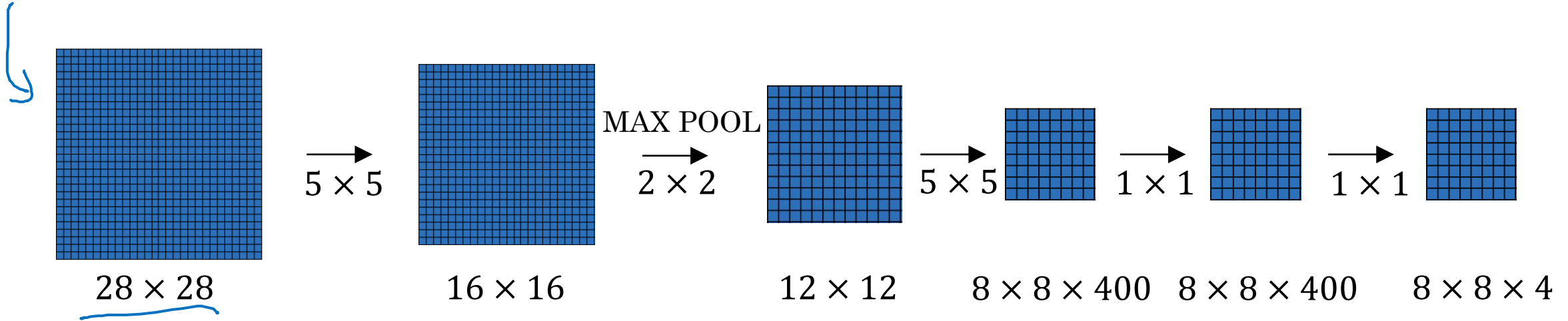
Doing the Sliding Windows algorithms sequentially will be computationally very expensive
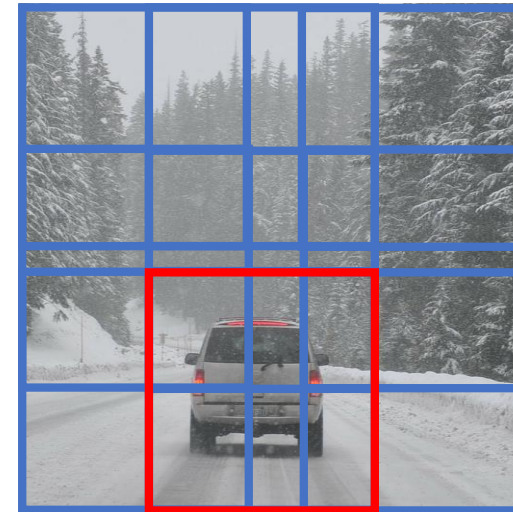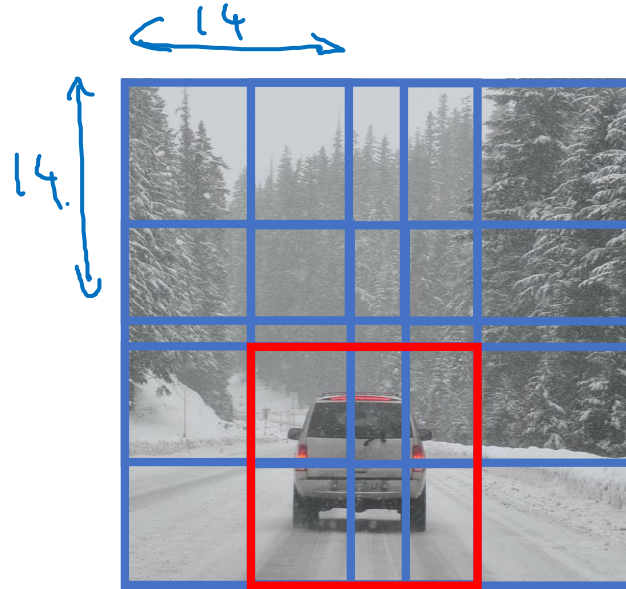
Implementing the Sliding Windows Algorithms convolutionally will solve the problem and speed up the computation.

All done at the same time, sharing same computations, very efficient.

Andrew Ng