

Applied Machine Learning

Convolution for Images

Convolution for Images

- Convolution operator
- Pattern detection
- Invalid convolutions
- Strides
- Properties of convolution

Image Classification

- Small local patterns in objects
 - edges
 - corners
 - dominant colors
 - textures
- Composition of patterns
 - small figures or shapes
- More composition
 - larger objects



Convolutional Layer - Pattern Detection

- Domain Knowledge from Image Processing
- Feature maps
 - Convolutional layers
- Stacks of convolutional layers
 - Learn patterns from patterns learned at previous layers
 - More complex patterns learned at each successive layer

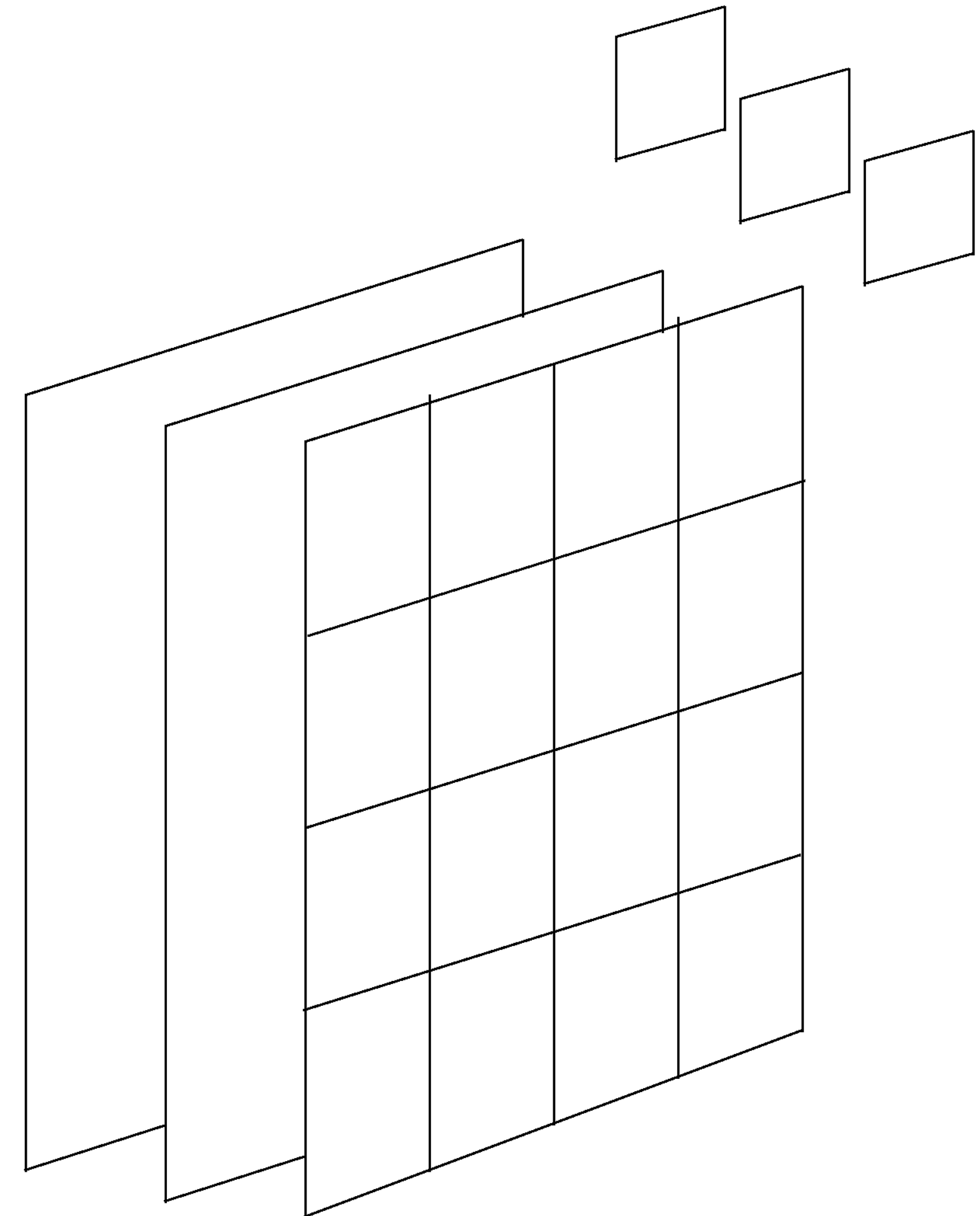
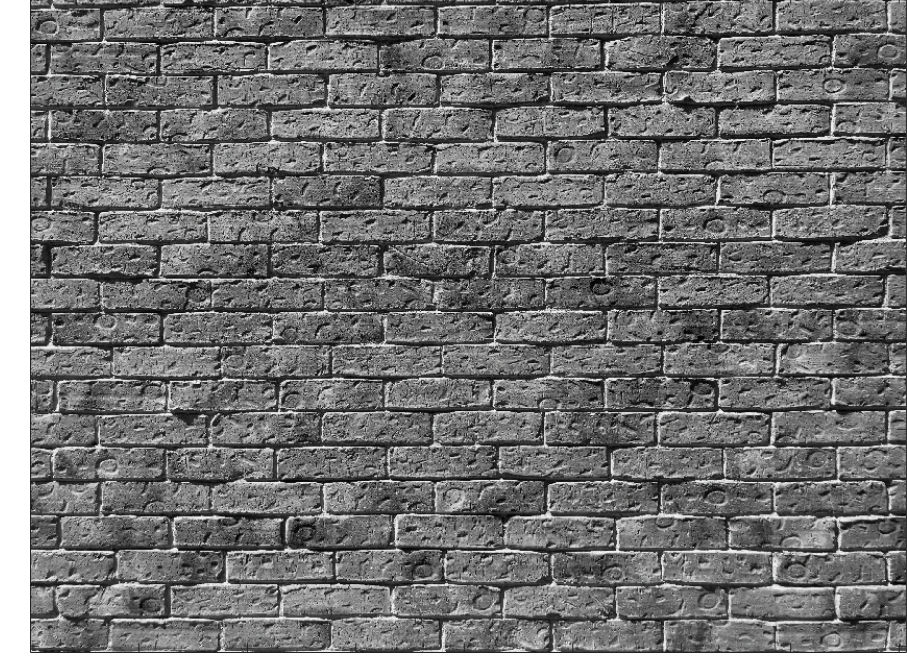


Image Processing



- Image: matrix I

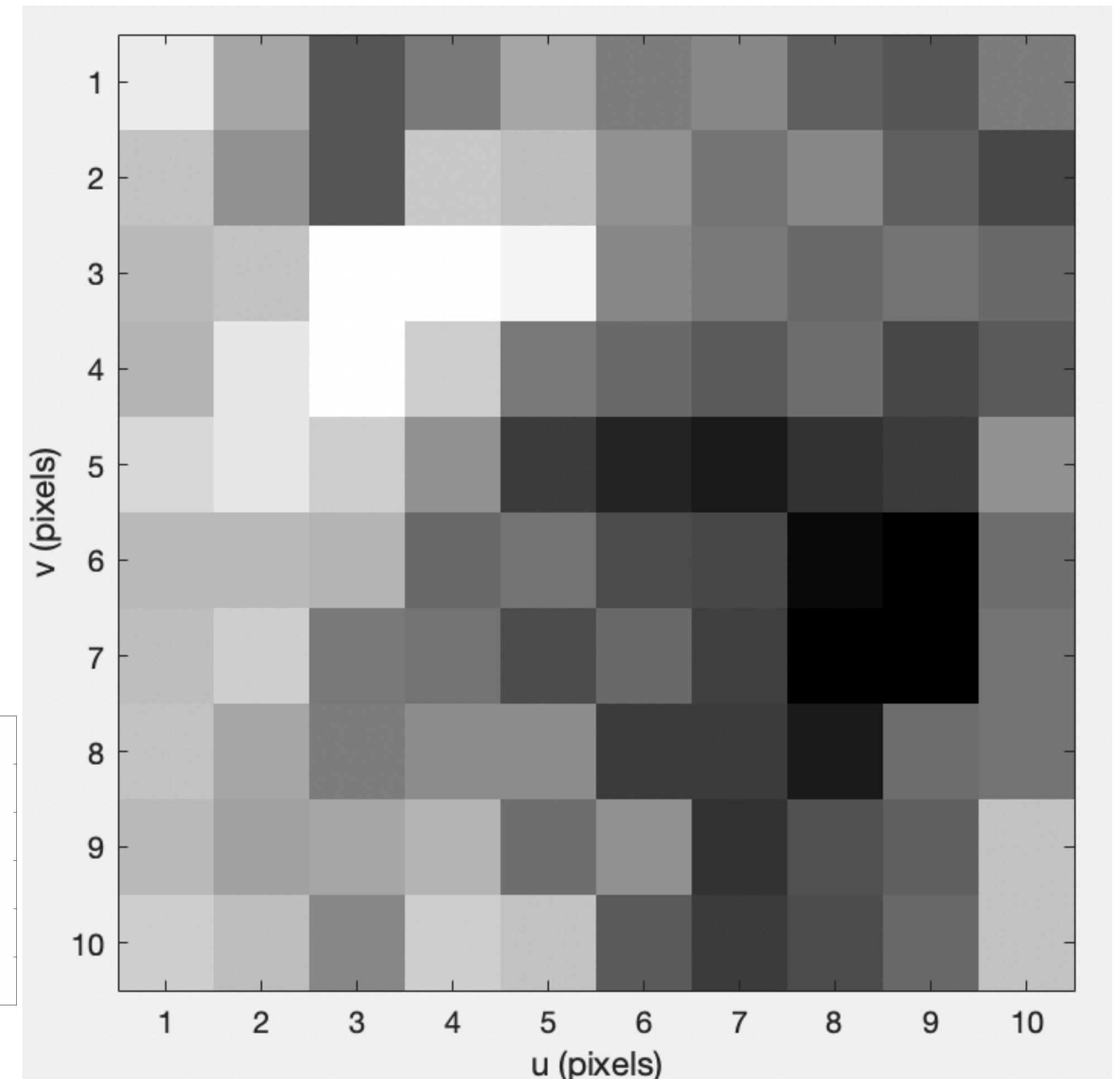
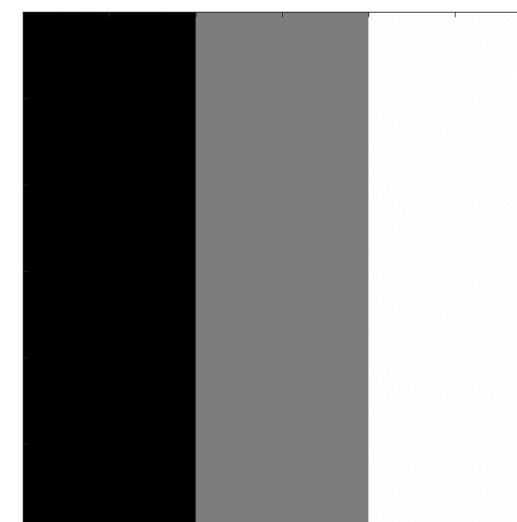
65	50	33	40	50	41	44	35	33	41
56	46	33	58	55	46	39	44	35	30
54	56	69	69	67	44	40	37	39	37
53	64	69	59	40	37	34	38	30	34
61	64	59	46	27	22	20	25	27	46
54	54	53	37	39	31	30	17	14	38
55	59	40	39	31	37	28	15	14	39
56	50	41	45	45	27	27	20	38	39
54	49	50	53	38	46	25	32	35	56
59	55	44	59	56	34	27	31	37	56

- Pixels: $I_{u,v}$

- Pattern: Kernel: matrix K

0.5000	0	-0.5000
0.5000	0	-0.5000
0.5000	0	-0.5000

- Weights: $K_{u,v}$



Convolution

- Convolution

- Slide kernel K over the image I

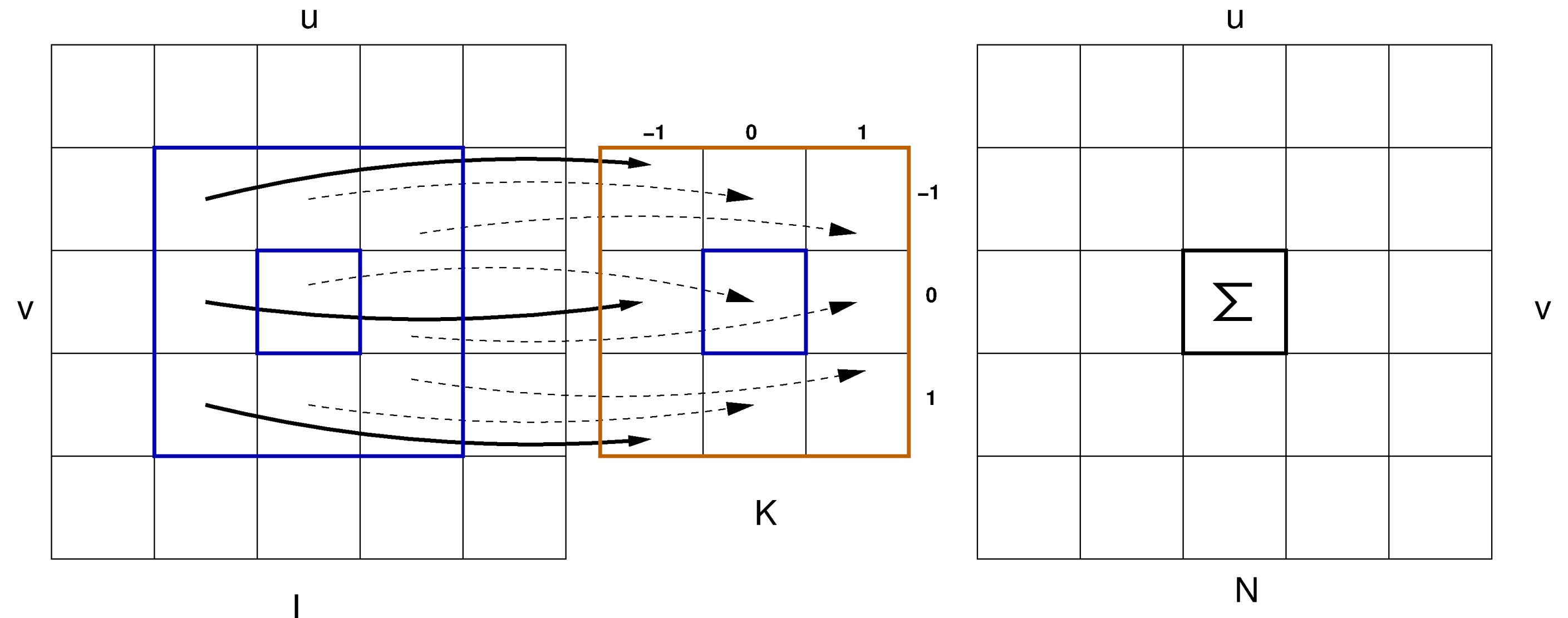
- $$N_{u,v} = \sum_{(i,j) \in K} I_{u+i,v+j} K_{i,j}$$

- compute for every pixel $\forall (u, v) \in I$
- Common representations of convolution:

- $$N = K \otimes I$$

- $$N = \text{conv}(K, I)$$

- Pixels at the output image encode information around them as defined by the weights in the kernel



Pattern Detection through Convolution

- Convolution

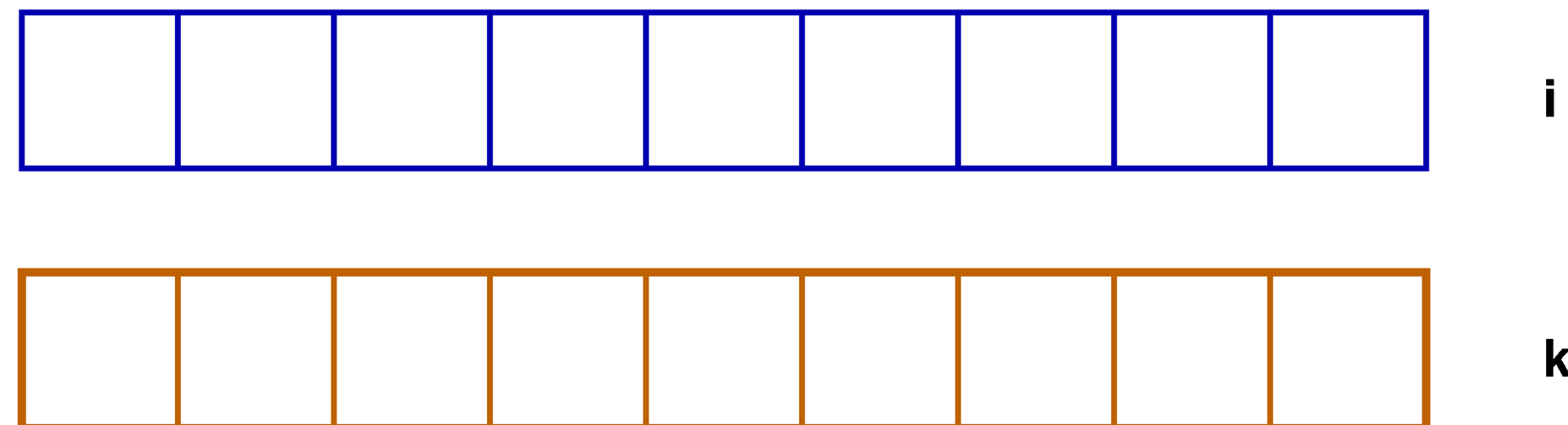
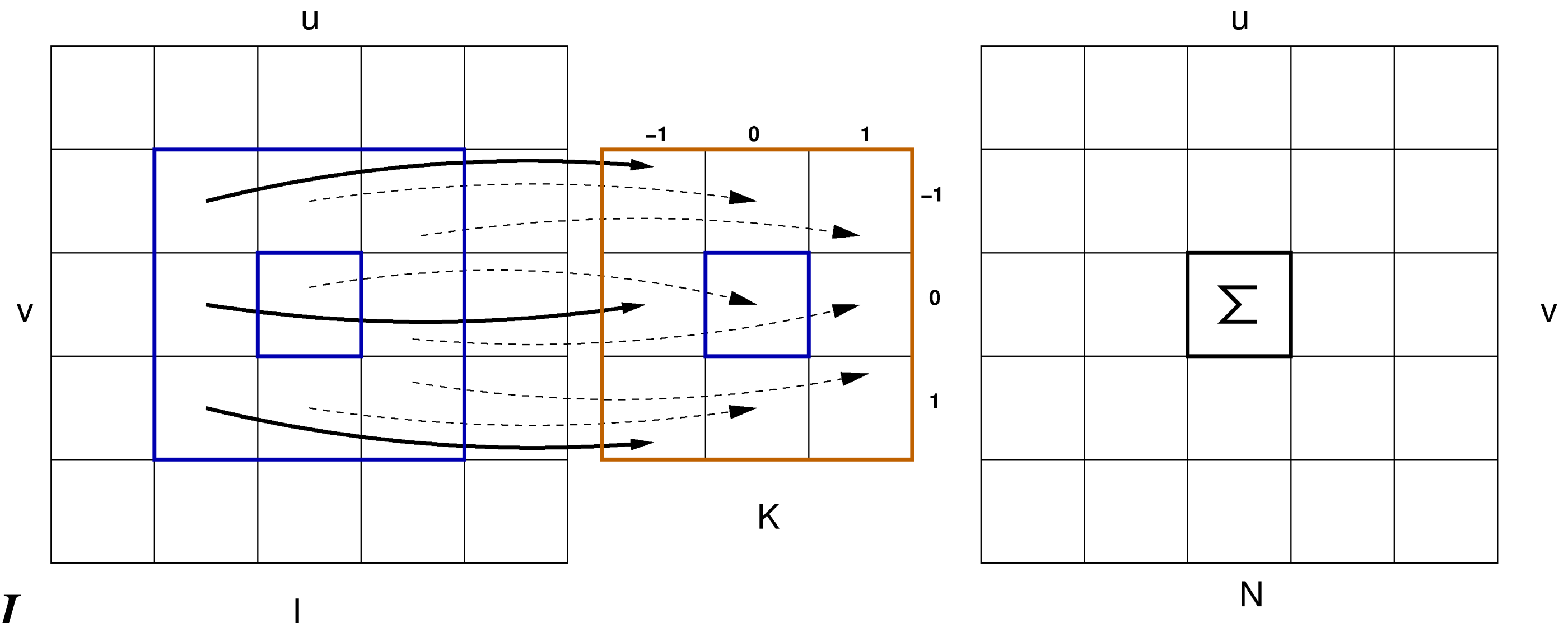
- Slide kernel K over the image I

- $$N_{u,v} = \sum_{(i,j) \in K} I_{u+i,v+j} K_{i,j}$$

- compute for every pixel $\forall (u, v) \in I$

- $$N_{u,v} = \mathbf{i} \cdot \mathbf{k}$$

- $$\begin{cases} \mathbf{i} = \mathbf{k} & N_{u,v} > 0 \\ \mathbf{i} = -\mathbf{k} & N_{u,v} < 0 \end{cases}$$



Pattern Detection through Convolution

- Convolution
 - Slide kernel K over the image I
 - $$N_{u,v} = \sum_{(i,j) \in K} I_{u+i,v+j} K_{i,j}$$
 - compute for every pixel $\forall (u, v) \in I$
- Kernel Templates
 - Edges, Points, Corners, Others
 - Filtering
- Learning Kernels
 - Weights in convolutional Layers

-0.5000	0	0.5000
-0.5000	0	0.5000
-0.5000	0	0.5000



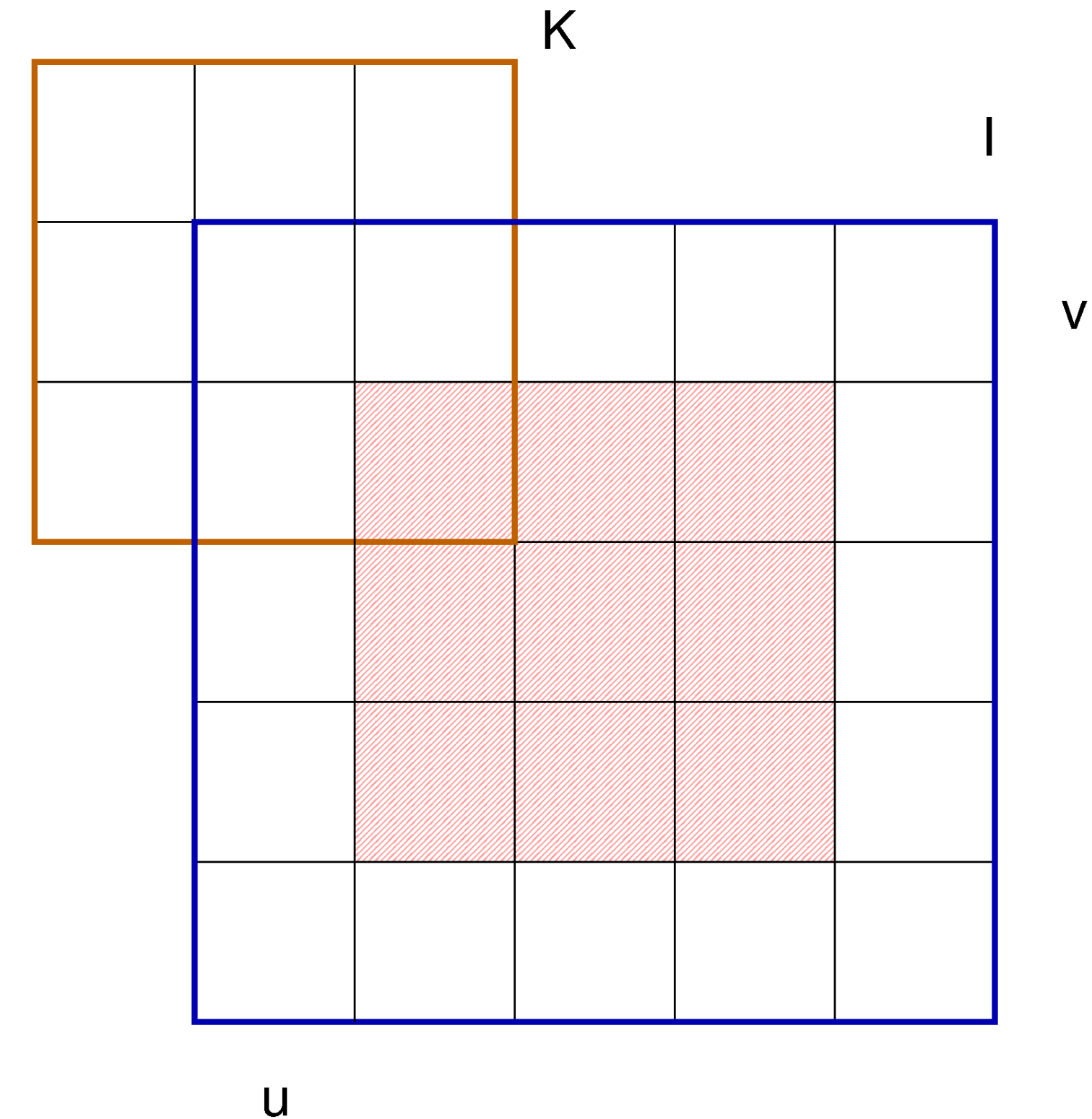
Kernel:
Vertical
Edge



Bricks on a Wall

Kernel Outside the Image

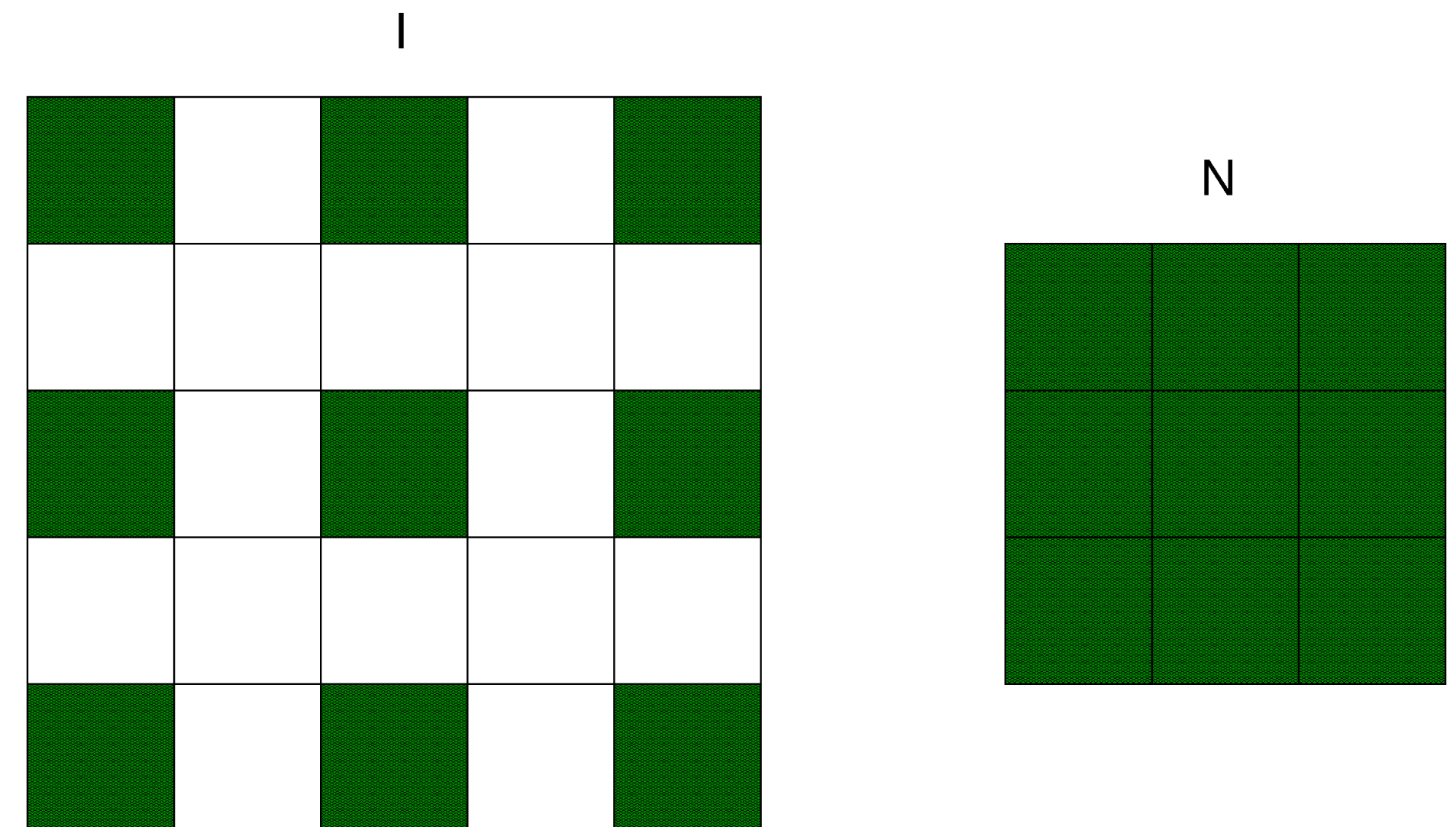
- Pixels with invalid convolution
 - kernel of size $\{2k + 1\} \times \{2k + 1\}$
 - k rows and columns closest to the borders: invalid convolution
- Handling invalid convolution
 - Valid convolution
 - Input I of size $\{i\} \times \{j\}$
 - Output N of size $\{i - 2k\} \times \{j - 2k\}$
 - Padding
 - expand I on each side by k pixels
 - assign value of 0 to new rows and columns
 - Input I of size $\{i\} \times \{j\}$
 - Output N of size $\{i\} \times \{j\}$



Invalid

Reducing Redundancy - Strides

- Applying convolution to every pixel leads to some redundancy in the output
 - downsampling reduces the size of the output
 - compute convolution to a subset of the input
- Stride size
 - number of pixels to move the kernel over the input image I
 - Stride of 1: compute convolution for every pixel
 - Stride of 2: move kernel by two pixels, skipping one in each dimension
 - Stride of s : move kernel by s pixels in each dimension
 - Input I of size $\{i\} \times \{j\}$
 - Output N of size $\{\lceil \frac{i}{s} \rceil\} \times \{\lceil \frac{j}{s} \rceil\}$



Properties of Convolution

- Commutative: $A \otimes B = B \otimes A$
- Associative: $A \otimes (B \otimes C) = (A \otimes B) \otimes C$
- Distributive: $A \otimes (B + C) = A \otimes B + A \otimes C$
- Linear: $A \otimes (\alpha B) = \alpha(A \otimes B)$
- Invariant to spatial shift $S(\cdot)$: $A \otimes S(B) = S(A \otimes B)$

Convolution for Images

- Convolution operator
- Pattern detection
- Invalid convolutions
- Strides
- Properties of convolution

Applied Machine Learning

Convolution for Images