Convolutional Neural Networks

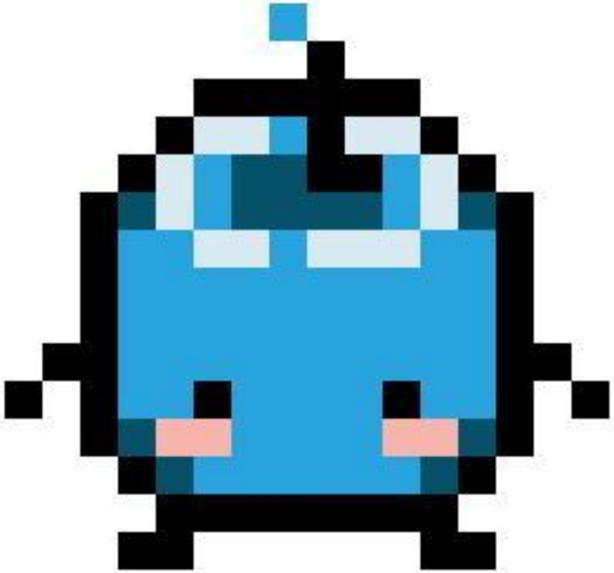
Dr. Chelsea Parlett-Pelleriti

Outline

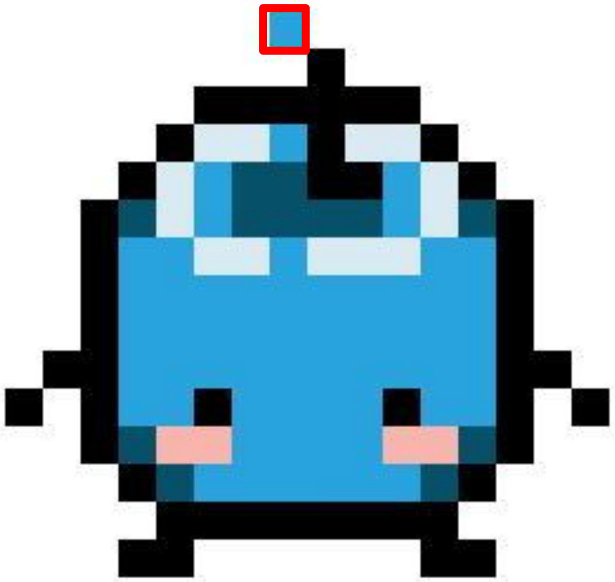
- Spatial Data
- Image Filtering
- Convolution
- Pooling

Spatial Data

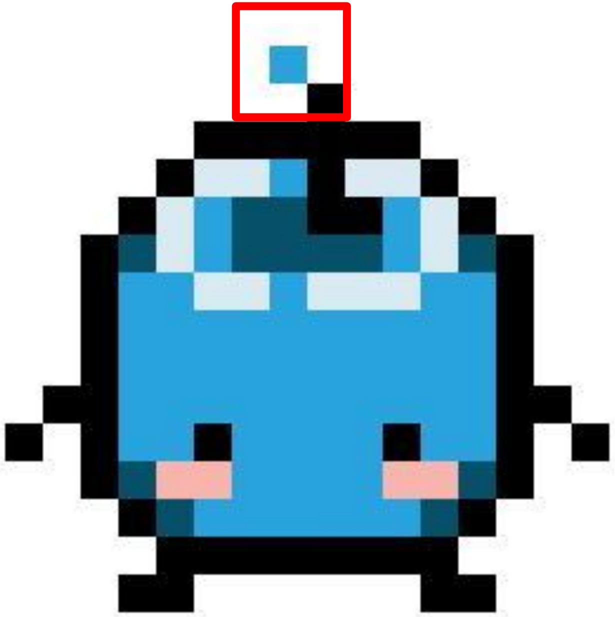
Spatial Data



Spatial Data



Spatial Data



Spatial Data

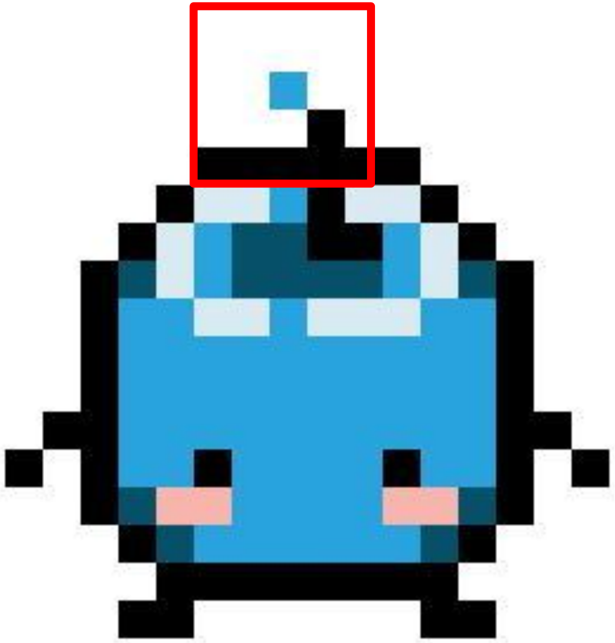
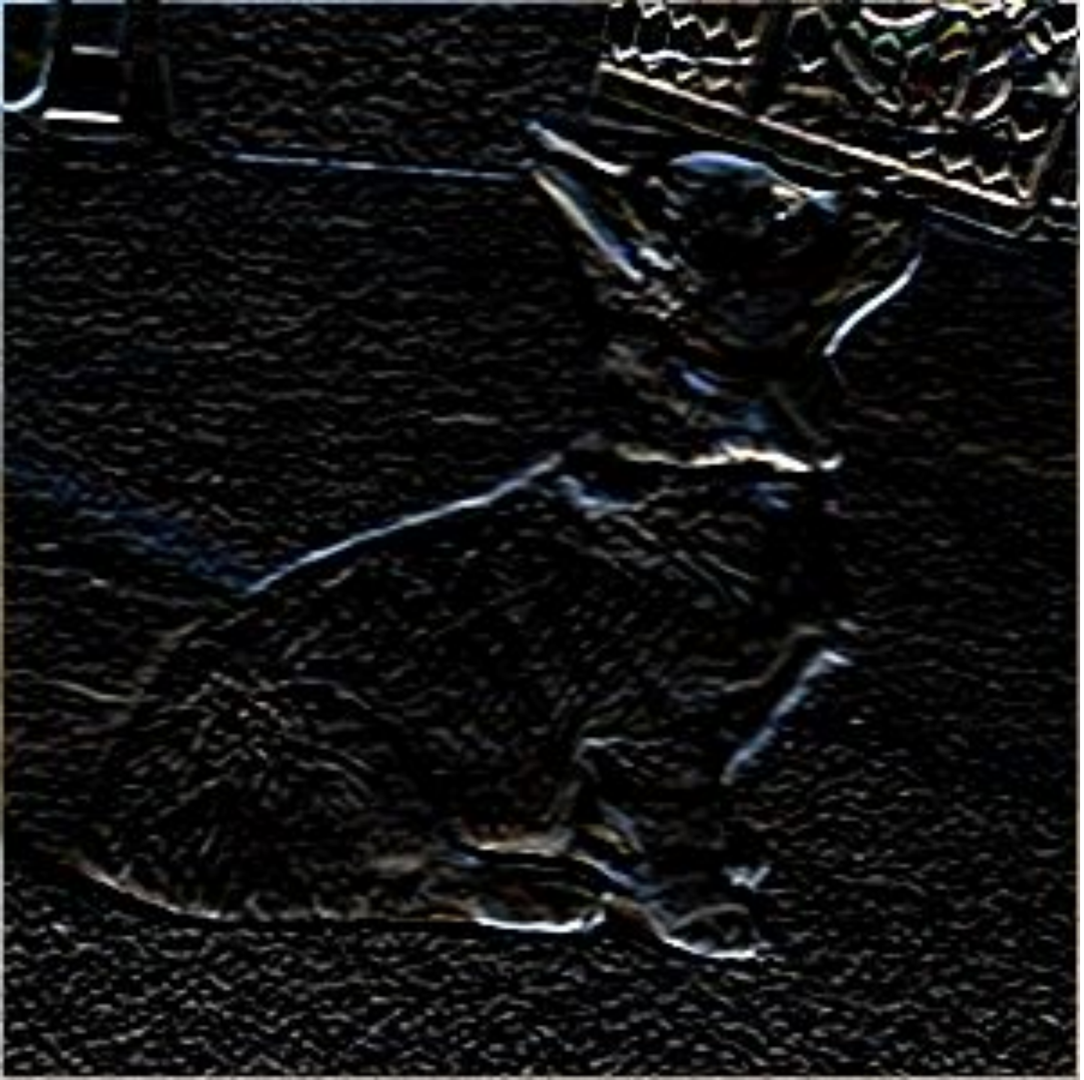


Image Filtering

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

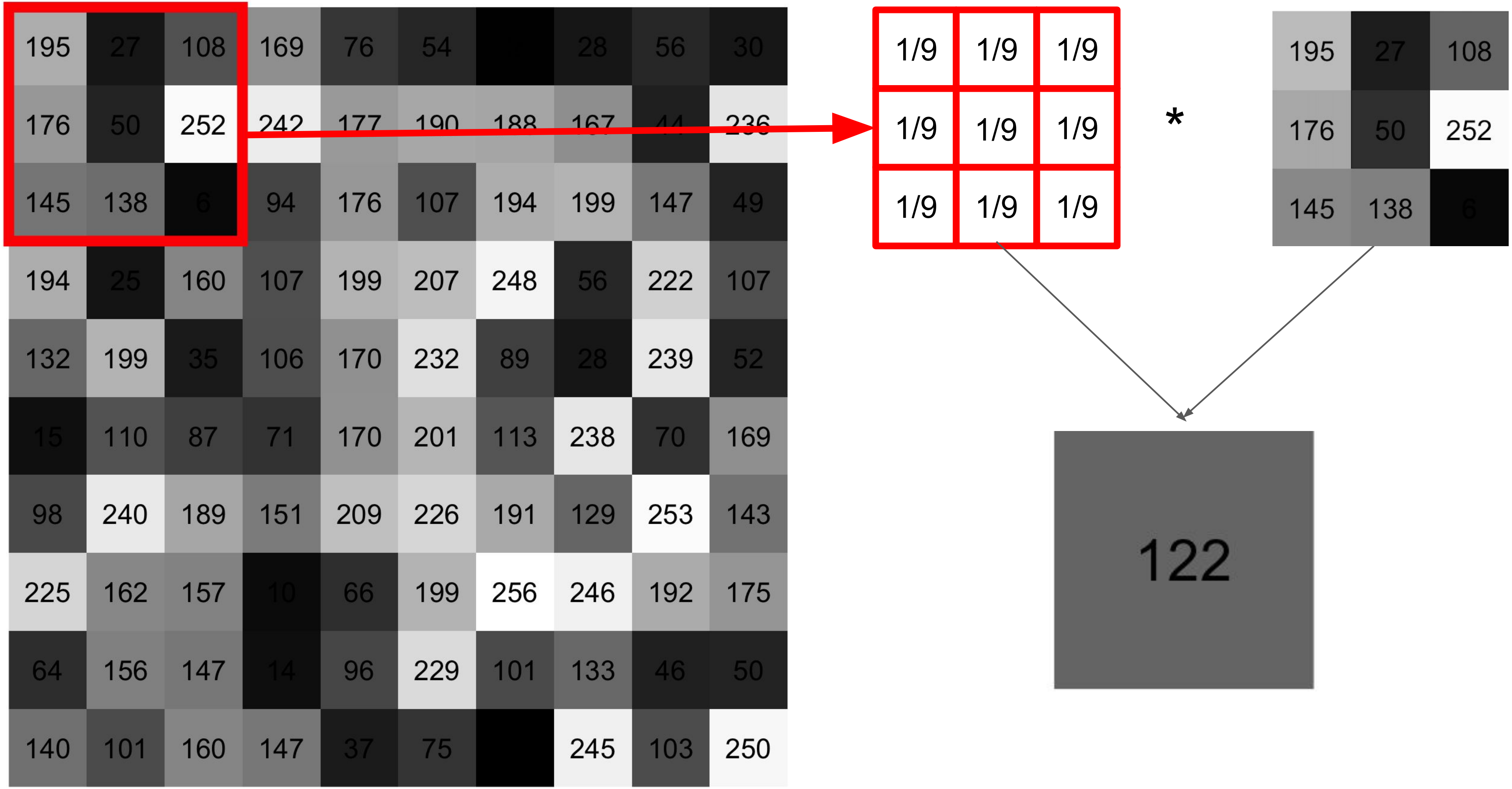


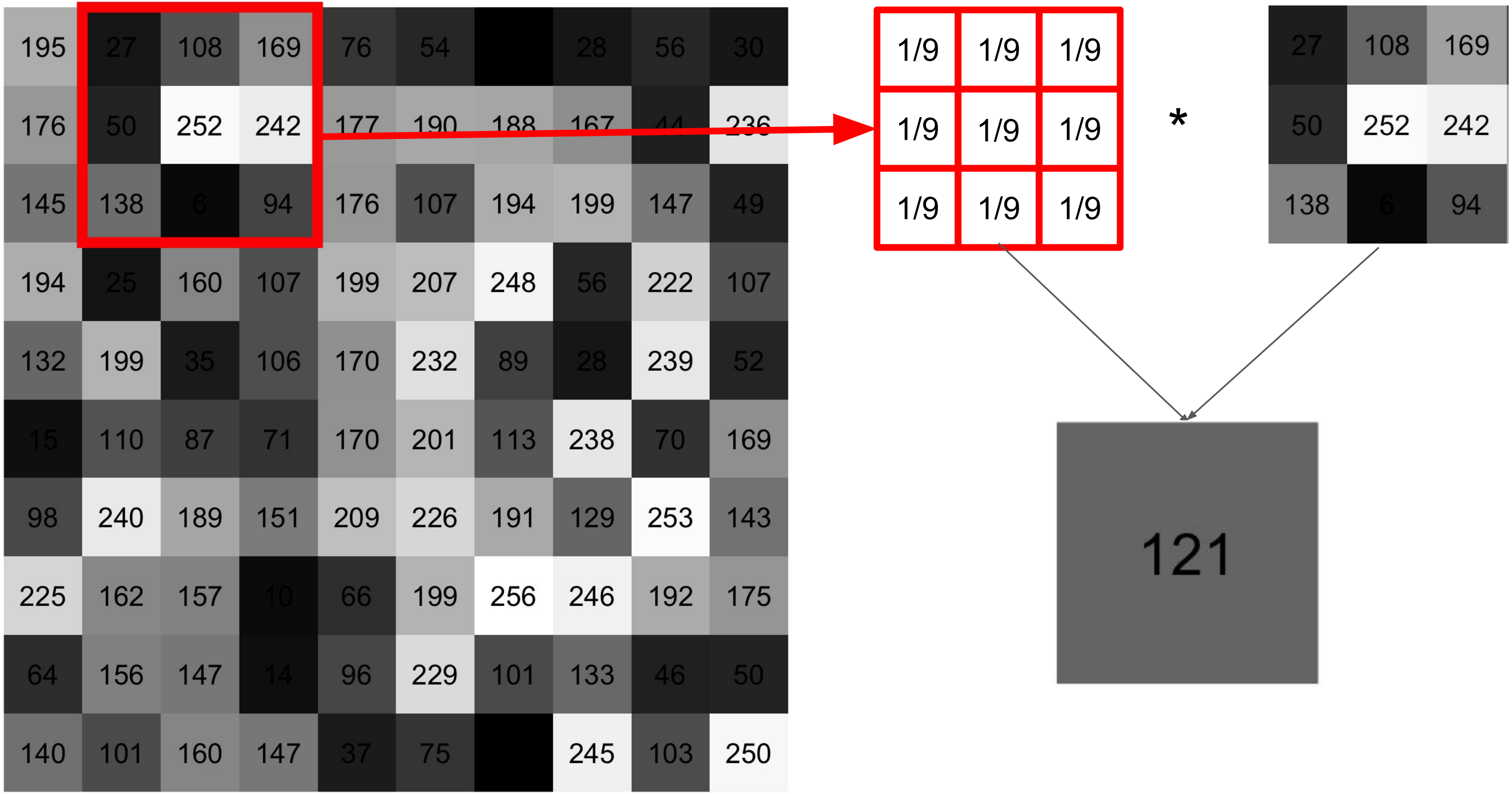
$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Convolution

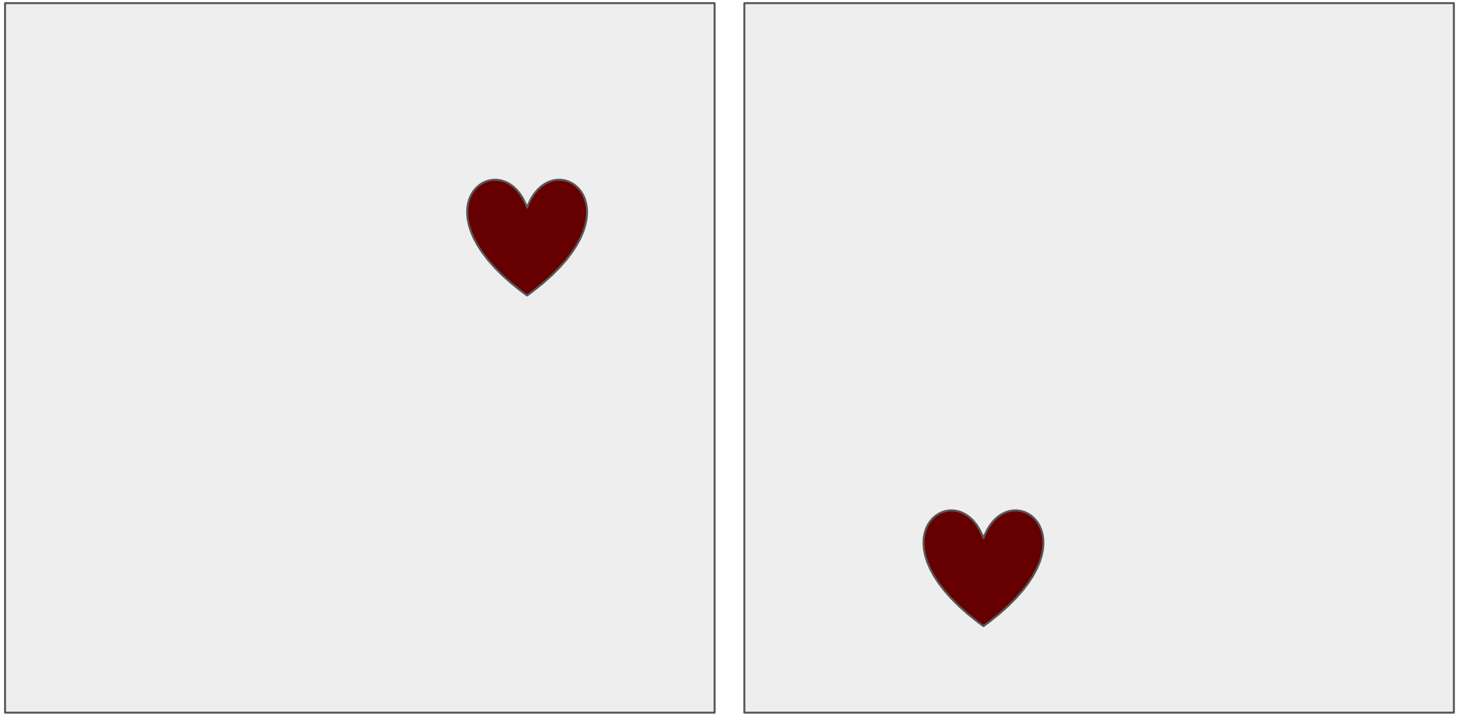
195	27	108	169	76	54		28	56	30
176	50	252	242	177	190	188	167	44	236
145	138		94	176	107	194	199	147	49
194	25	160	107	199	207	248	56	222	107
132	199	35	106	170	232	89	28	239	52
15	110	87	71	170	201	113	238	70	169
98	240	189	151	209	226	191	129	253	143
225	162	157	10	66	199	256	246	192	175
64	156	147	14	96	229	101	133	46	50
140	101	160	147	37	75		245	103	250


195 27 108 169 76 54 28 56 30
176 50 252 242 177 190 188 167 44 236
145 138 6 94 176 107 194 199 147 49
194 25 160 107 199 207 248 56 222 107
132 199 35 106 170 232 89 28 239 52
15 110 87 71 170 201 113 238 70 169
98 240 189 151 209 226 191 129 253 143
225 162 157 10 66 199 256 246 192 175
64 156 147 14 96 229 101 133 46 50
140 101 160 147 37 75 245 103 250
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
*
195 27 108
176 50 252
145 138 6
122



195 27 108 169 76 54 28 56 30
176 50 252 242 177 190 188 167 44 236
145 138 6 94 176 107 194 199 147 49
194 25 160 107 199 207 248 56 222 107
132 199 35 106 170 232 89 28 239 52
15 110 87 71 170 201 113 238 70 169
98 240 189 151 209 226 191 129 253 143
225 162 157 10 66 199 256 246 192 175
64 156 147 14 96 229 101 133 46 50
140 101 160 147 37 75 245 103 250
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
*
27 108 169
50 252 242
138 6 94
121

Weight Sharing



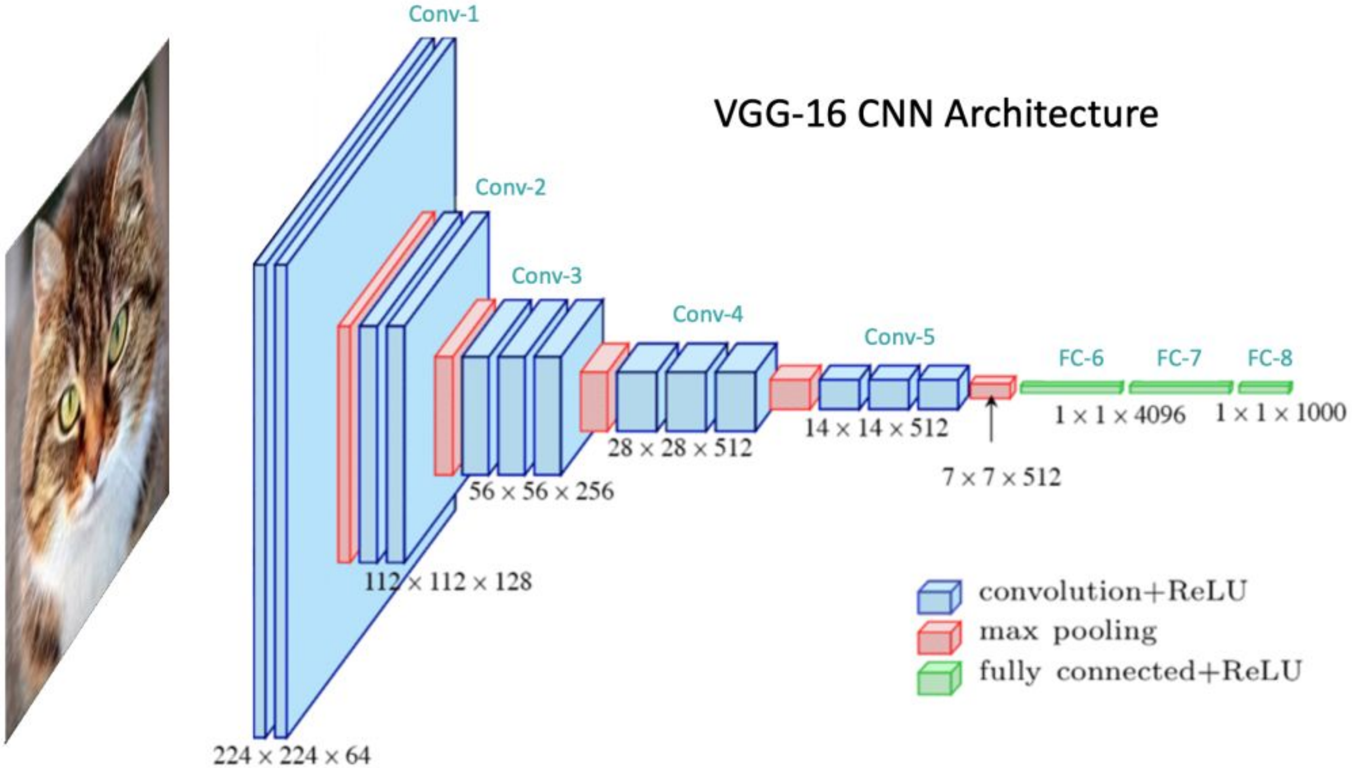

weight
sharing
gym bros
conv nets
imgflip.com

Convolutional Neural Networks

But choosing filters is a lot of work

What if we could learn filters through Gradient Descent/Backprop?

Convolutional Neural Networks



Hierarchical Feature Detection

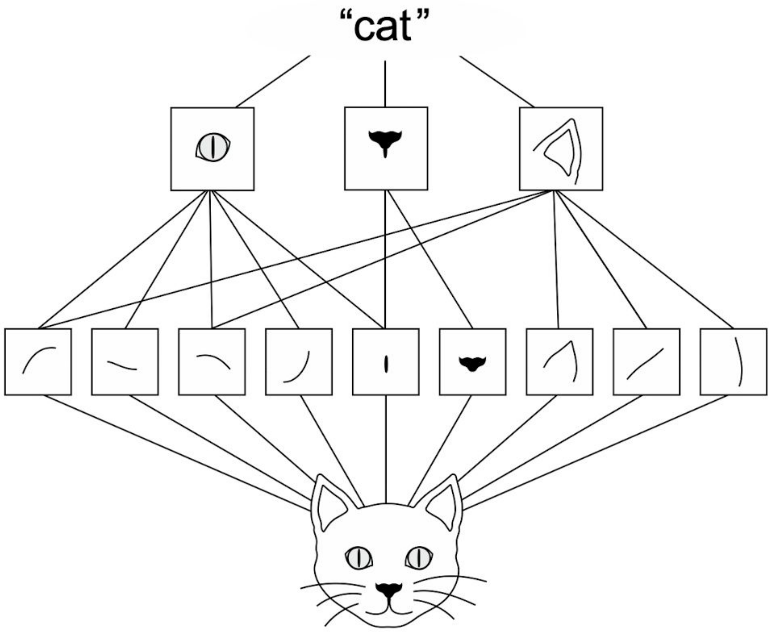


Figure 8.2 The visual world forms a spatial hierarchy of visual modules: elementary lines or textures combine into simple objects such as eyes or ears, which combine into high-level concepts such as "cat."

Hierarchical Feature Detection

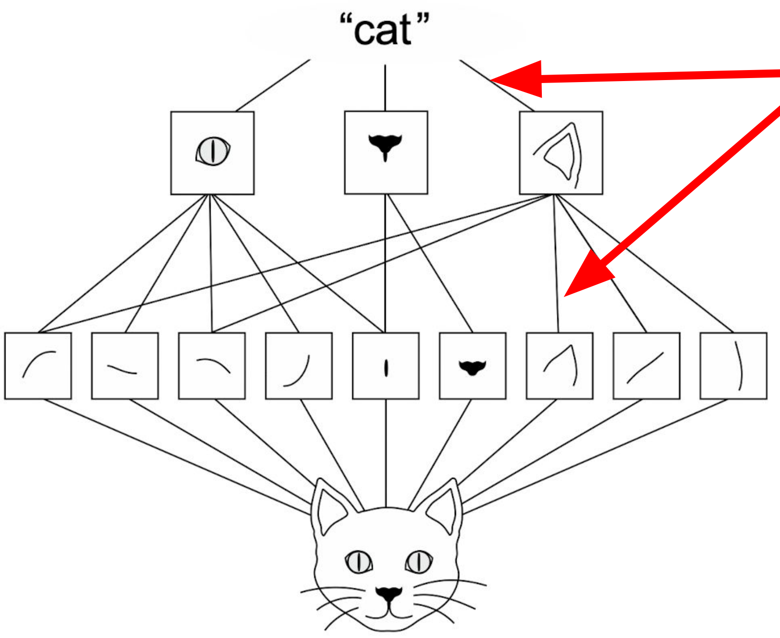
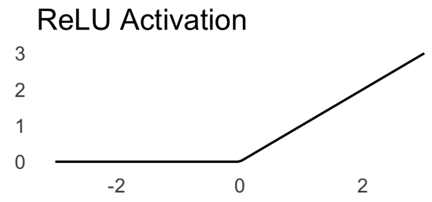
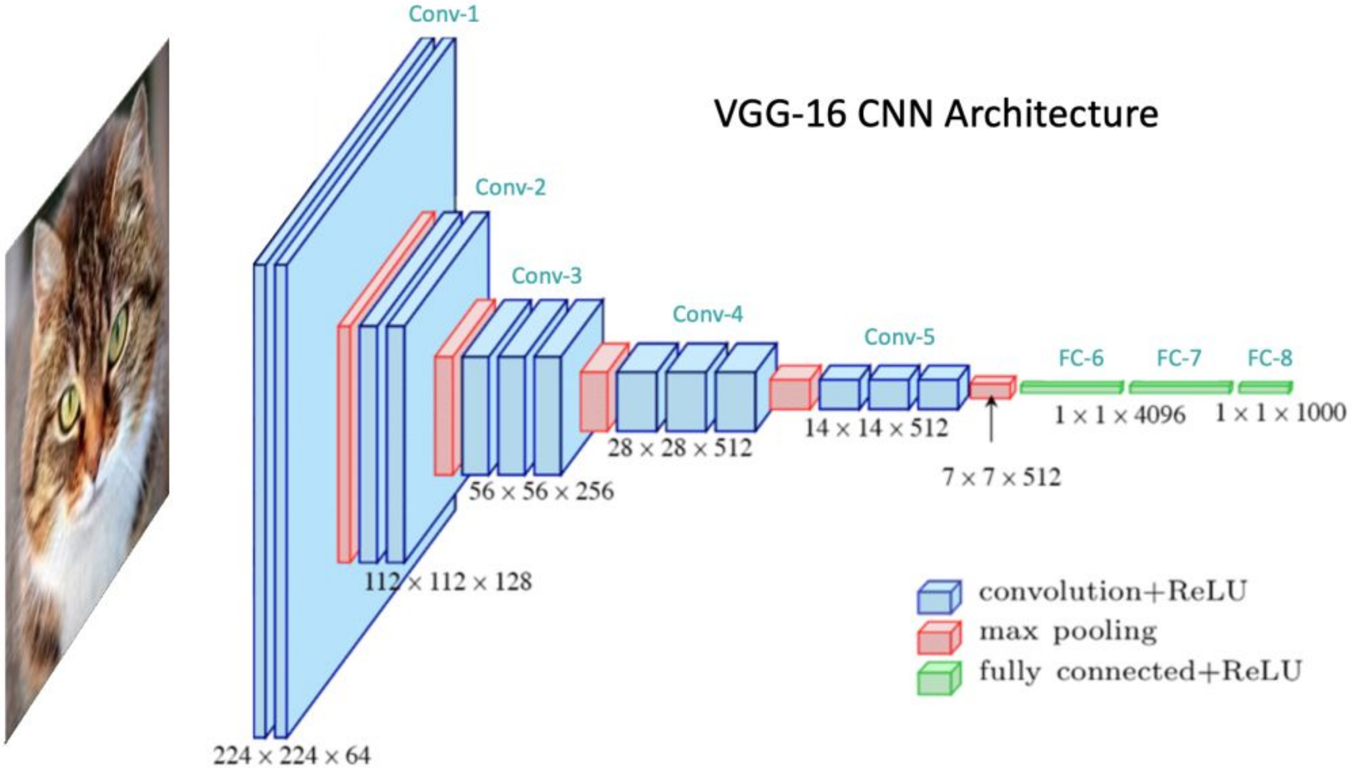


Figure 8.2 The visual world forms a spatial hierarchy of visual modules: elementary lines or textures combine into simple objects such as eyes or ears, which combine into high-level concepts such as "cat."



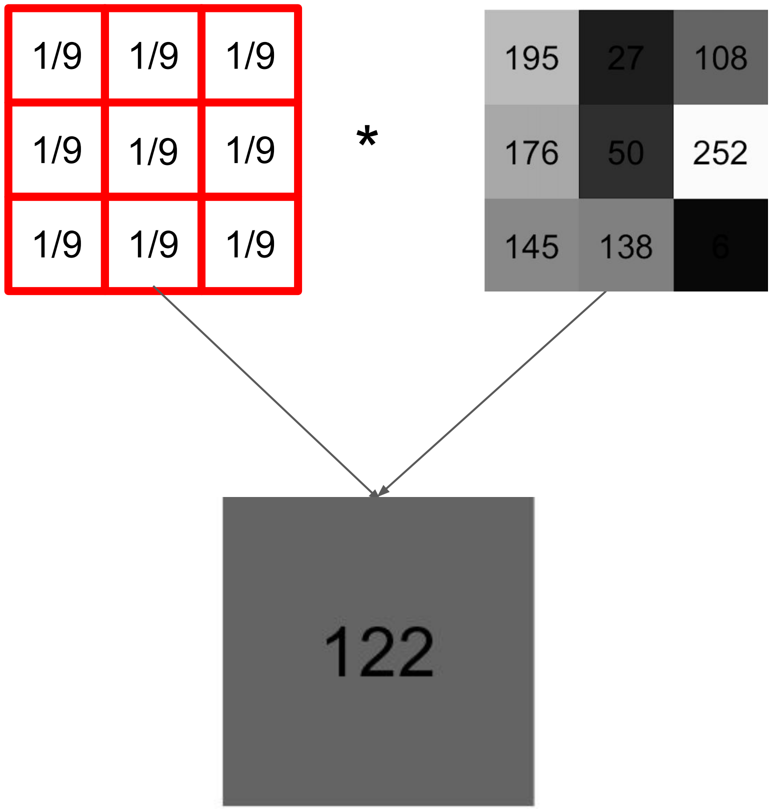
Convolutional Neural Networks



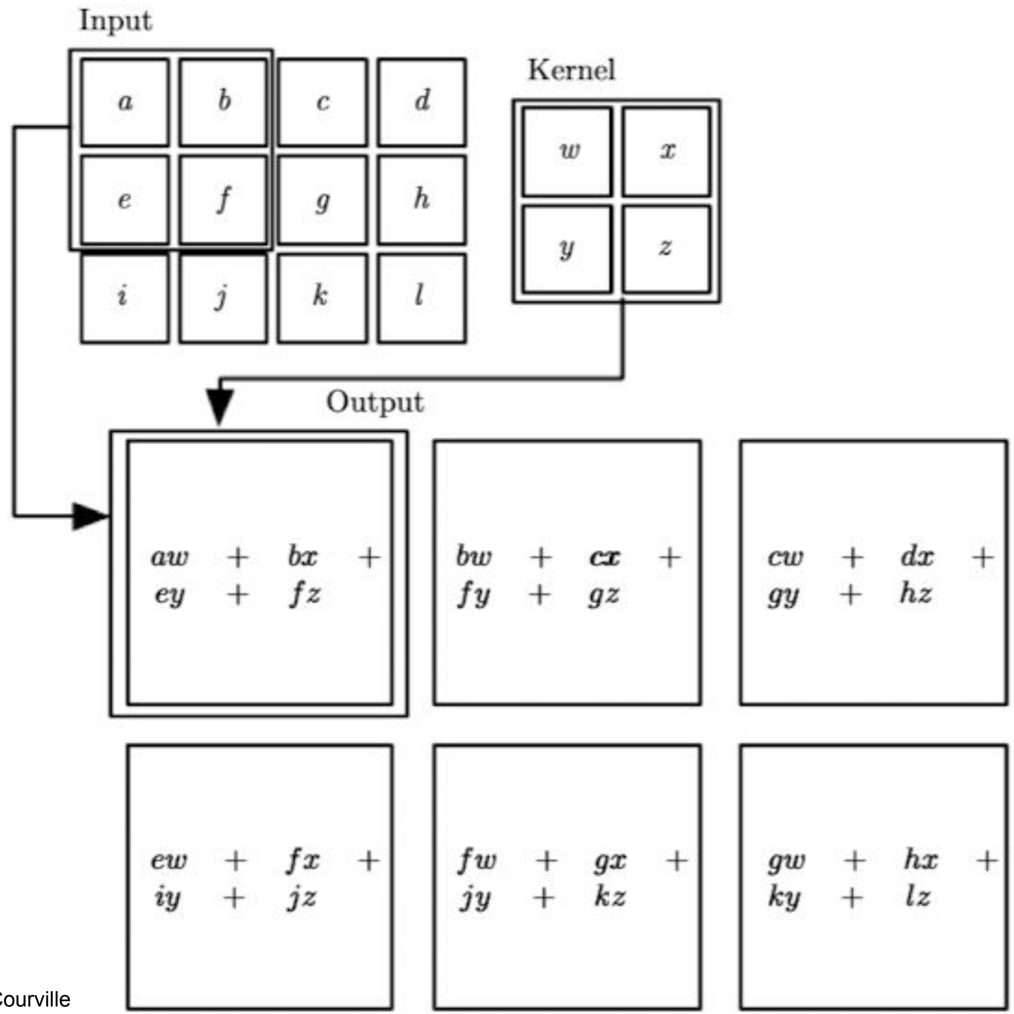
Convolution (Math)

195	27	108	169	76	54		28	56	30
176	50	252	242	177	190	188	167	44	236
145	138	6	94	176	107	194	199	147	49
194	25	160	107	199	207	248	56	222	107
132	199	35	106	170	232	89	28	239	52
15	110	87	71	170	201	113	238	70	169
98	240	189	151	209	226	191	129	253	143
225	162	157	10	66	199	256	246	192	175
64	156	147	14	96	229	101	133	46	50
140	101	160	147	37	75		245	103	250

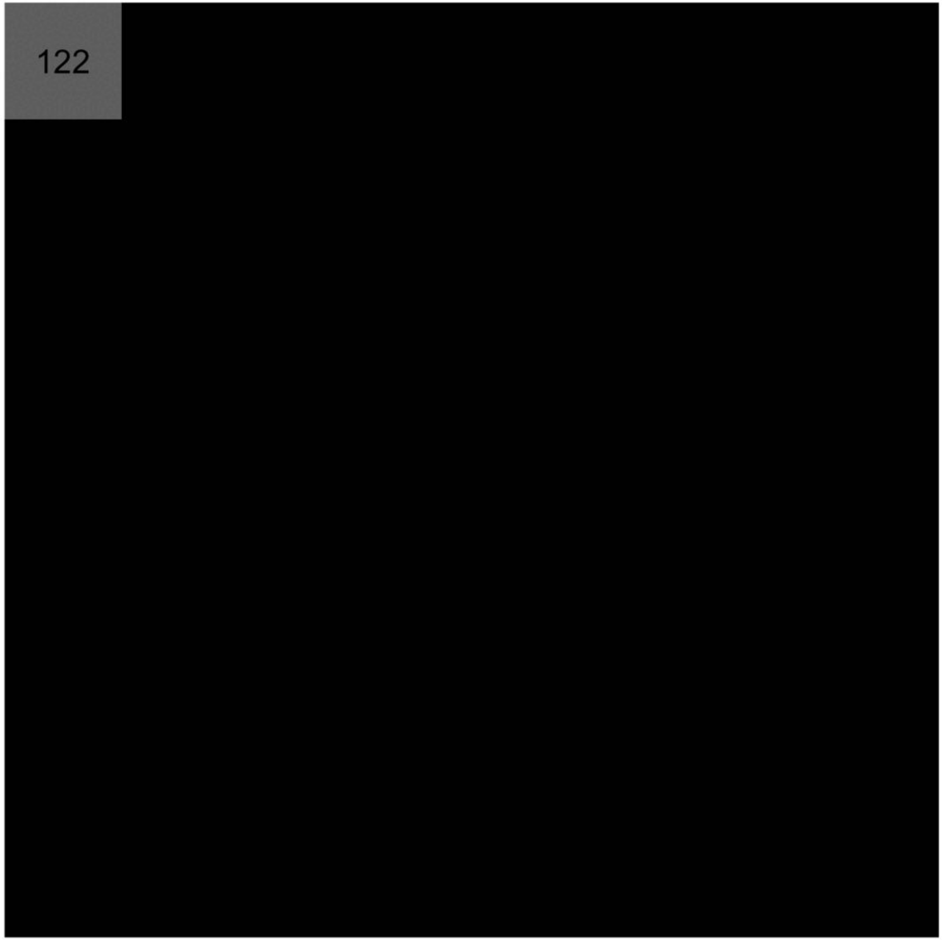
Convolution (Math)



Convolution



From Deep Learning by Goodfellow, Bengio and Courville

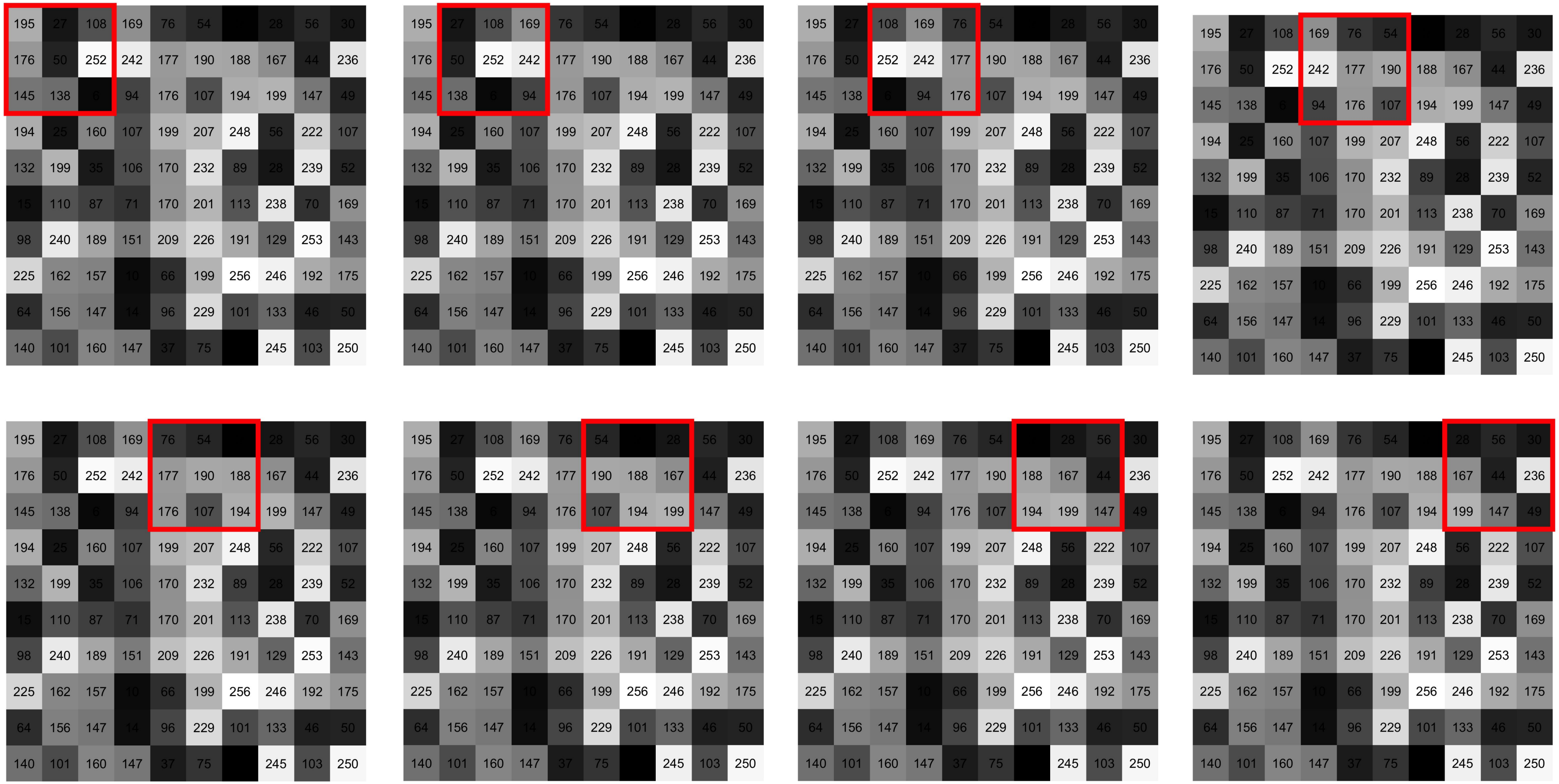


122	121	144	143	129	125	114	106
127	119	157	167	187	173	163	136
115	97	117	155	180	151	158	122
106	100	123	163	181	157	145	131
123	132	132	171	178	161	150	147
143	131	123	145	181	200	188	179
160	136	115	133	175	190	172	152
146	117	93	97	118	165	147	160

10x10 to 8x8 with a 3x3 filter

122	121	144	143	129	125	114	106
127	119	157	167	187	173	163	136
115	97	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
106	100	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
123	132	132	171	178	161	[illegible]	[illegible]
143	131	123	145	181	200	[illegible]	[illegible]
160	136	115	133	175	190	172	152
146	117	93	97	118	165	147	160

The Output Matrix is Smaller than the Input Matrix


195 27 108 169 76 54 28 56 30
176 50 252 242 177 190 188 167 44 236
145 138 6 94 176 107 194 199 147 49
194 25 160 107 199 207 248 56 222 107
132 199 35 106 170 232 89 28 239 52
15 110 87 71 170 201 113 238 70 169
98 240 189 151 209 226 191 129 253 143
225 162 157 10 66 199 256 246 192 175
64 156 147 14 96 229 101 133 46 50
140 101 160 147 37 75 245 103 250

Padding

195	27	108	169	76	54	[illegible]	28	56	30
176	50	252	242	177	190	188	167	44	236
145	138	6	94	176	107	194	199	147	49
194	25	160	107	199	207	248	56	222	107
132	199	35	106	170	232	89	28	239	52
15	110	87	71	170	201	113	238	70	169
98	240	189	151	209	226	191	129	253	143
225	162	157	10	66	199	256	246	192	175
64	156	147	14	96	229	101	133	46	50
140	101	160	147	37	75	[illegible]	245	103	250

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

Padding

What size will the output be now?

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

Padding

What size will the output be now? (size is reduced by **n**-1 for an **nxn** filter)

	195	27	108	169	76	54		28	56	30	
	176	50	252	242	177	190	188	167	44	236	
	145	138	6	94	176	107	194	199	147	49	
	194	25	160	107	199	207	248	56	222	107	
	132	199	35	106	170	232	89	28	239	52	
	15	110	87	71	170	201	113	238	70	169	
	98	240	189	151	209	226	191	129	253	143	
	225	162	157	10	66	199	256	246	192	175	
	64	156	147	14	96	229	101	133	46	50	
	140	101	160	147	37	75		245	103	250	

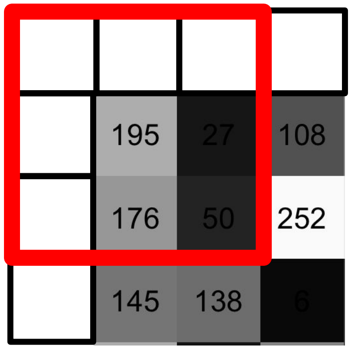
Padding

Valid

195	27	108
176	50	252
145	138	6

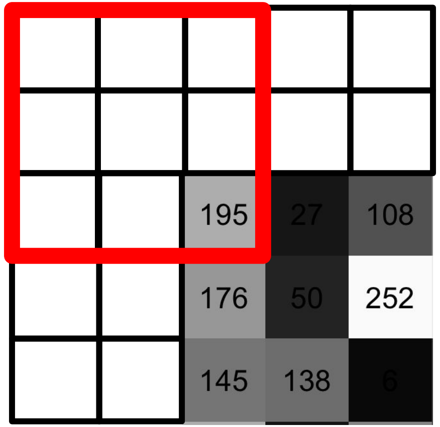
Padding

Same



Padding

Full



Quick Detour: RGB Convolution

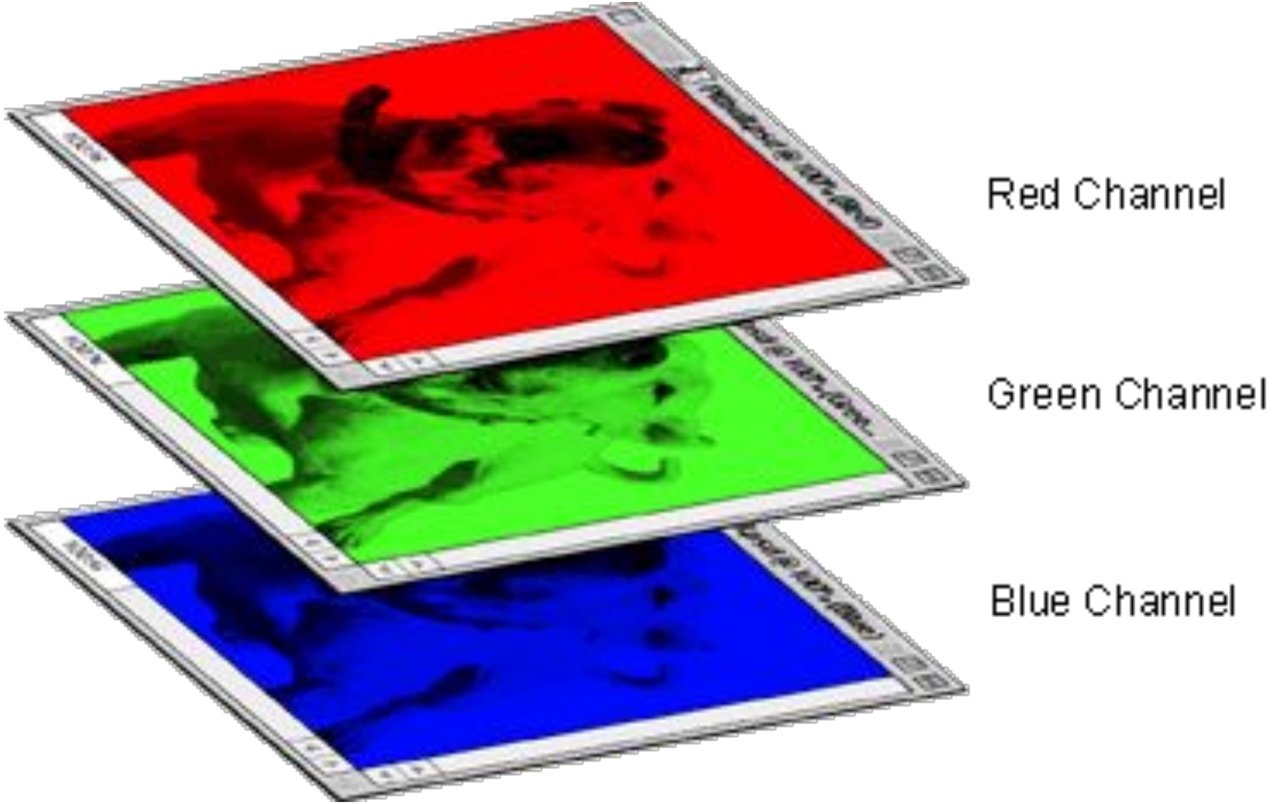


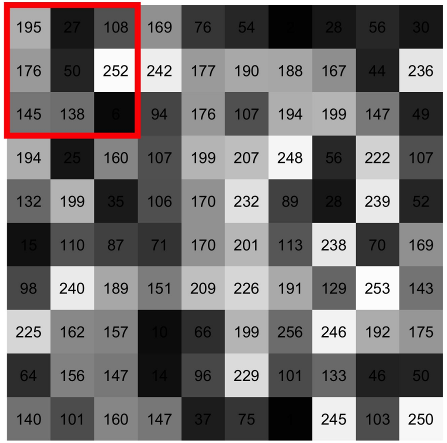
Image from: https://www.sketchpad.net/channels1.htm

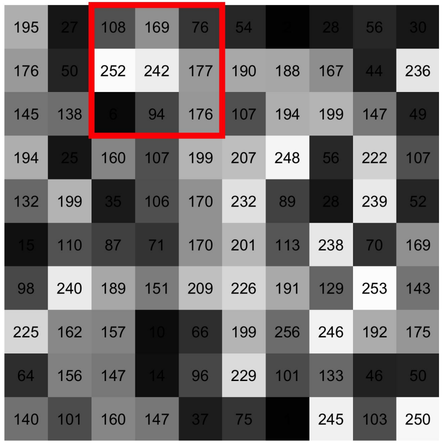
Terminology

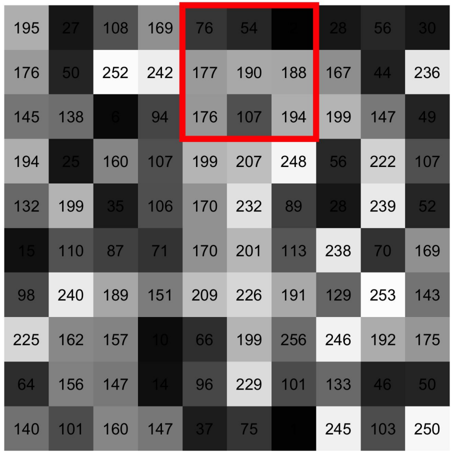
- Input
- Filters/Kernels
- Output/Featuremap

Strides

Strides







What will happen to the size of our input when we use stride > 1?

Pooling

Max Pooling

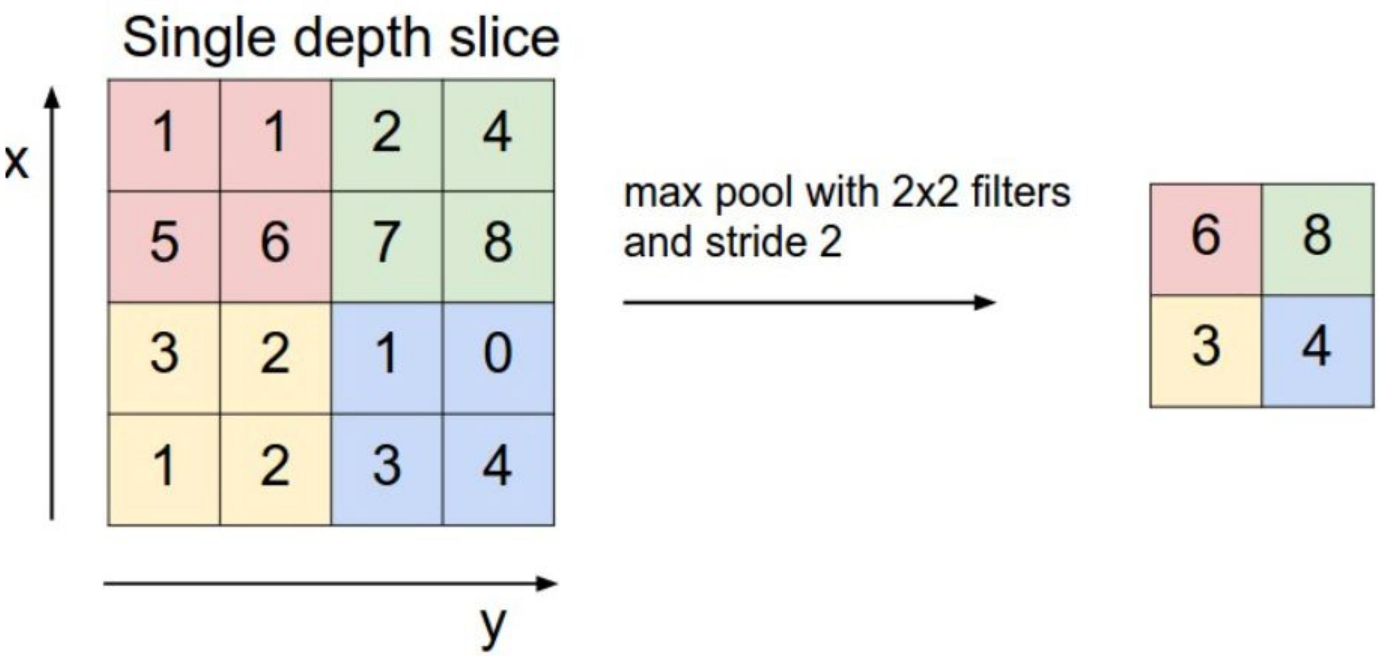
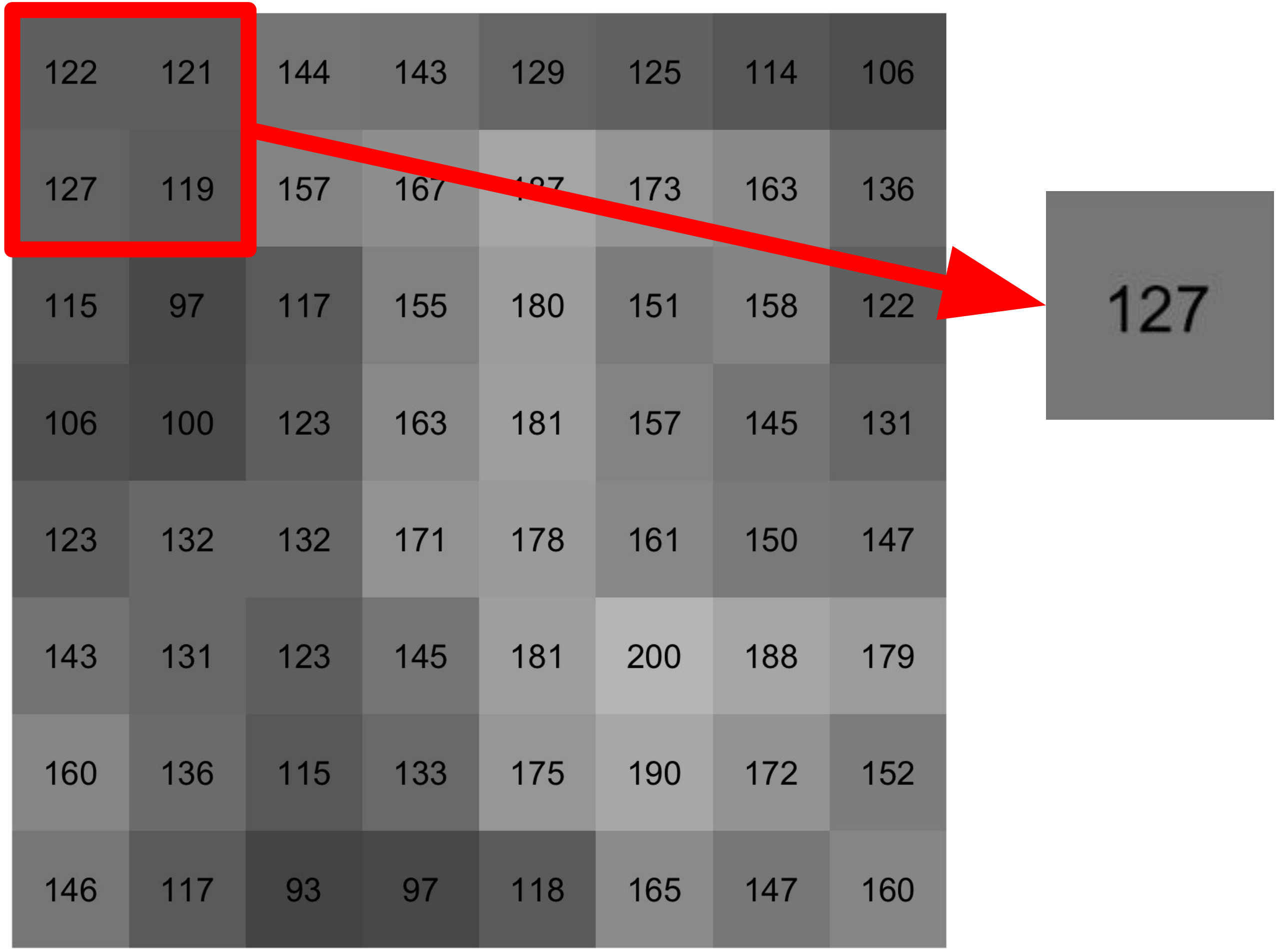
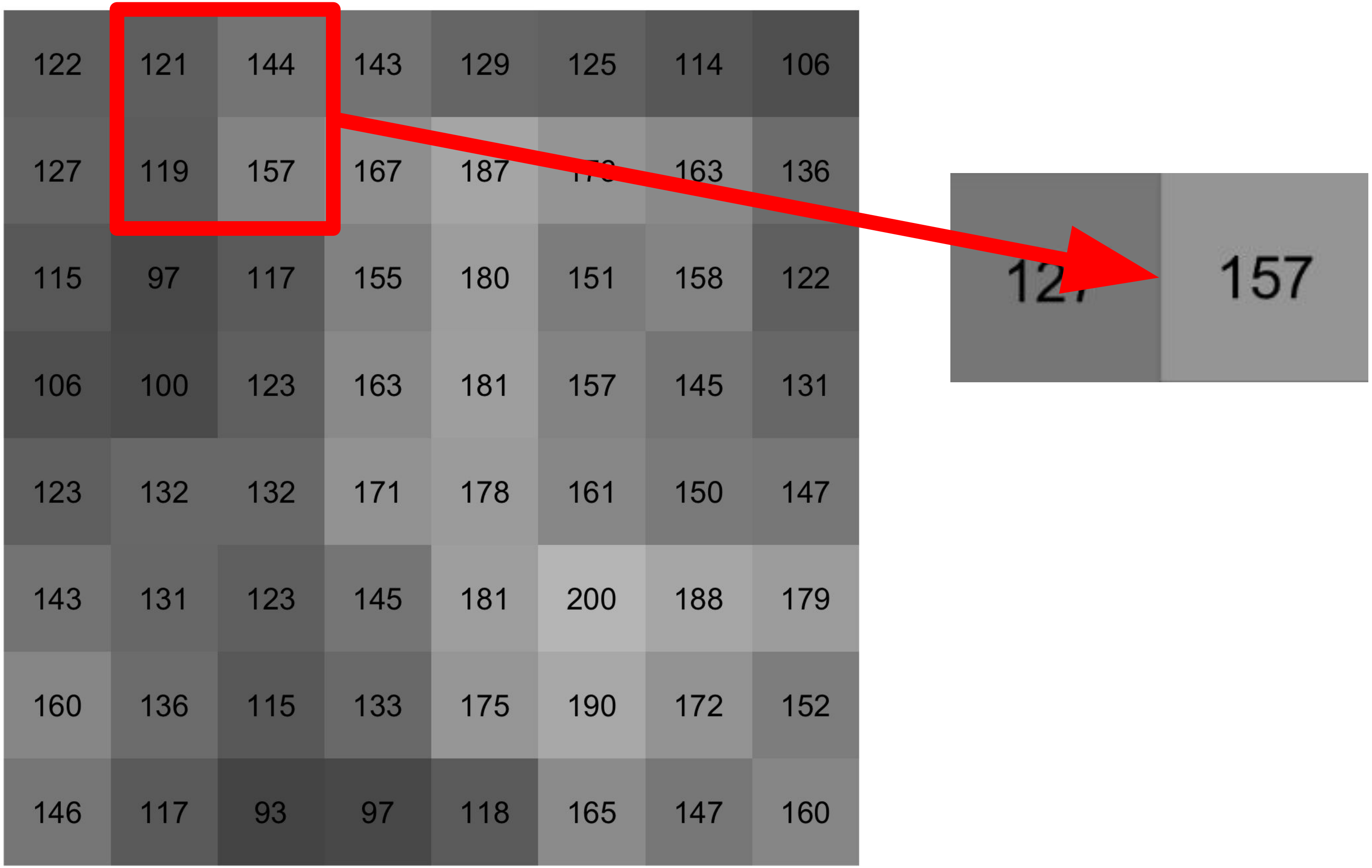
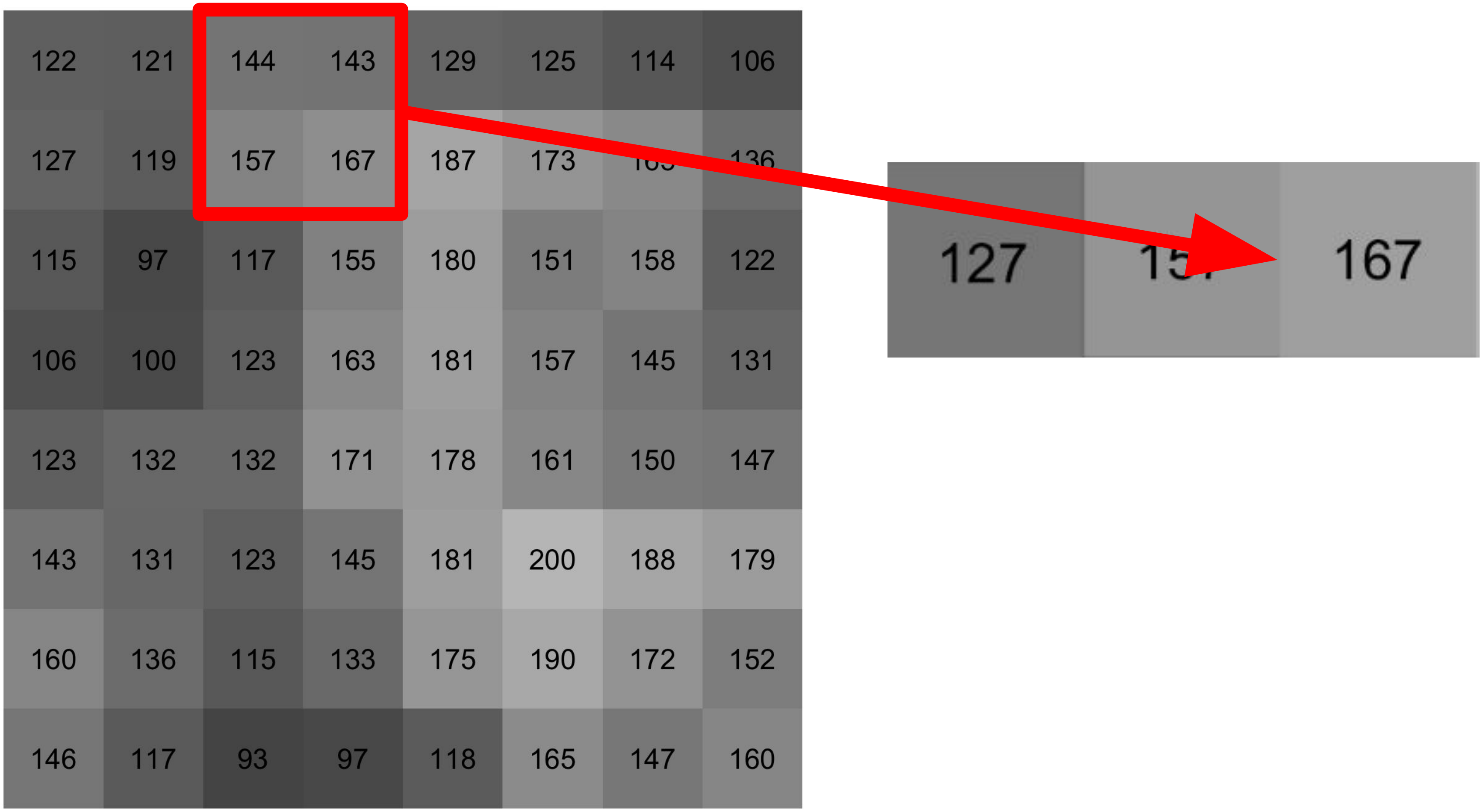


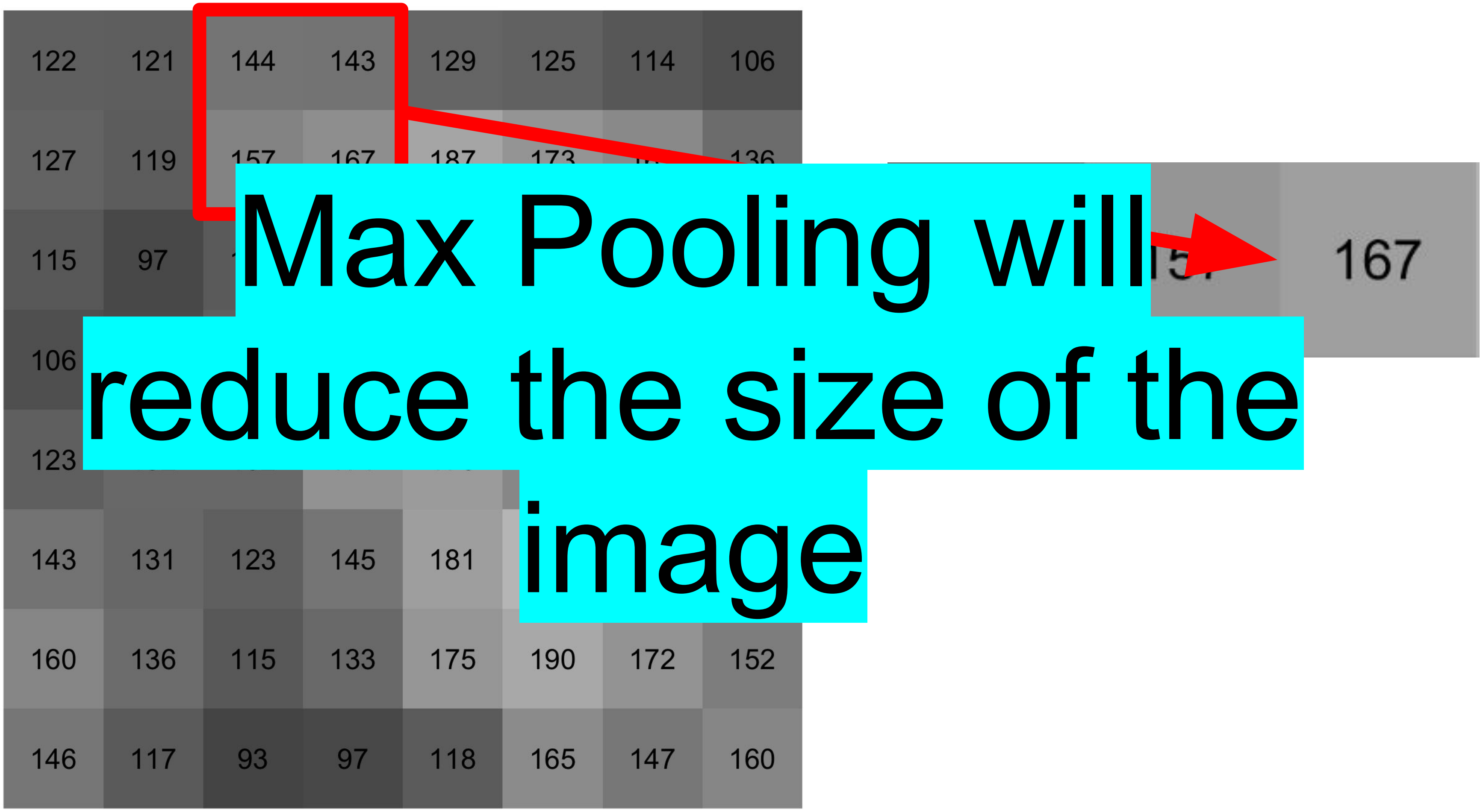
Image from: https://cs231n.github.io/convolutional-networks/


122 121 144 143 129 125 114 106
127 119 157 167 173 163 136
115 97 117 155 180 151 158 122
106 100 123 163 181 157 145 131
123 132 132 171 178 161 150 147
143 131 123 145 181 200 188 179
160 136 115 133 175 190 172 152
146 117 93 97 118 165 147 160
127

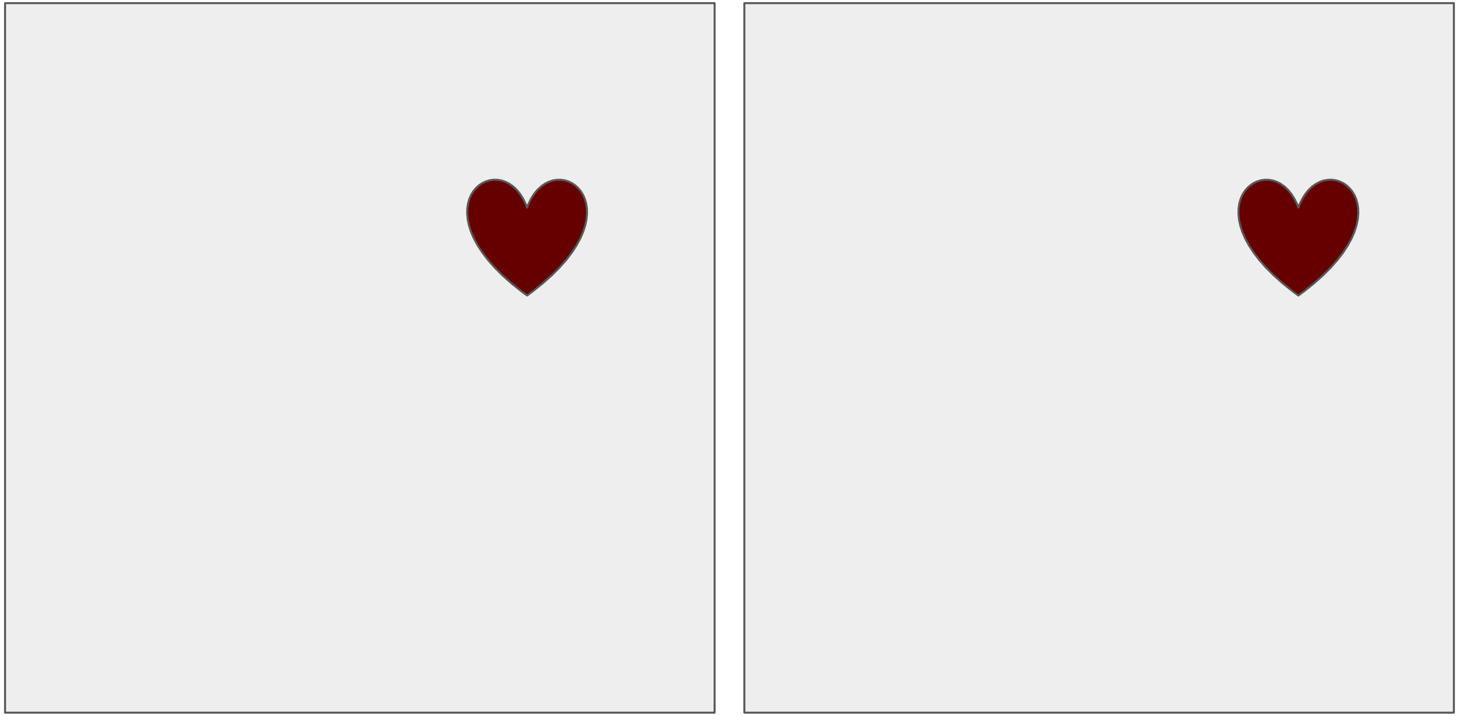

122 121 144 143 129 125 114 106
127 119 157 167 187 163 136
115 97 117 155 180 151 158 122
106 100 123 163 181 157 145 131
123 132 132 171 178 161 150 147
143 131 123 145 181 200 188 179
160 136 115 133 175 190 172 152
146 117 93 97 118 165 147 160
157



122 121 144 143 129 125 114 106
127 119 157 167 187 173 136
115 97 117 155 180 151 158 122
106 100 123 163 181 157 145 131
123 132 132 171 178 161 150 147
143 131 123 145 181 200 188 179
160 136 115 133 175 190 172 152
146 117 93 97 118 165 147 160
127 167


Max Pooling will reduce the size of the image
122 121 144 143 129 125 114 106
127 119
115 97
106
123
143 131 123 145 181
160 136 115 133 175 190 172 152
146 117 93 97 118 165 147 160
167

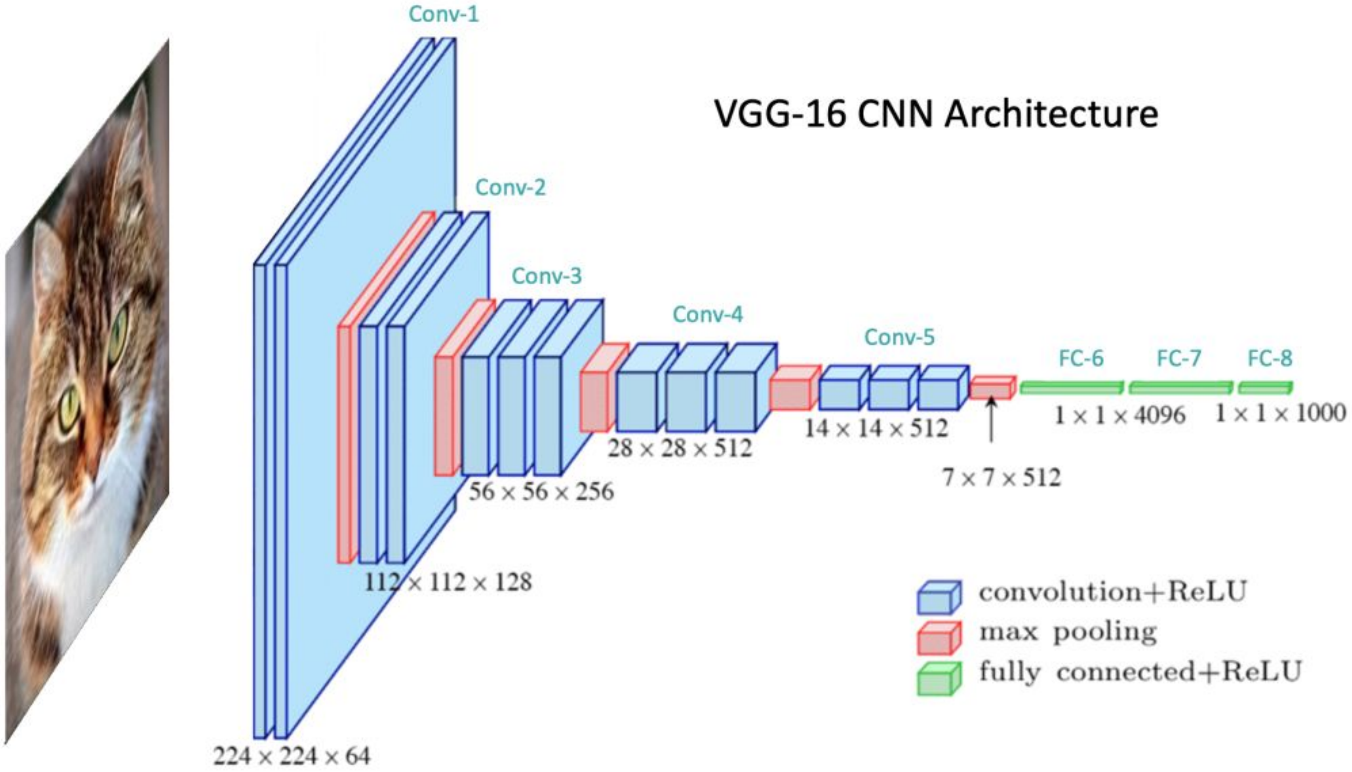
Translational Invariance and Weight Sharing



Pooling

- Downsample feature maps
- Max pooling works best because when looking for a feature it is better to look at the maximal presence rather than the average presence
- It’s best practice to do un-strided convolutions then downsample with maxpooling rather than using strides to downsample

Convolutional Neural Networks



Convolutional Neural Networks