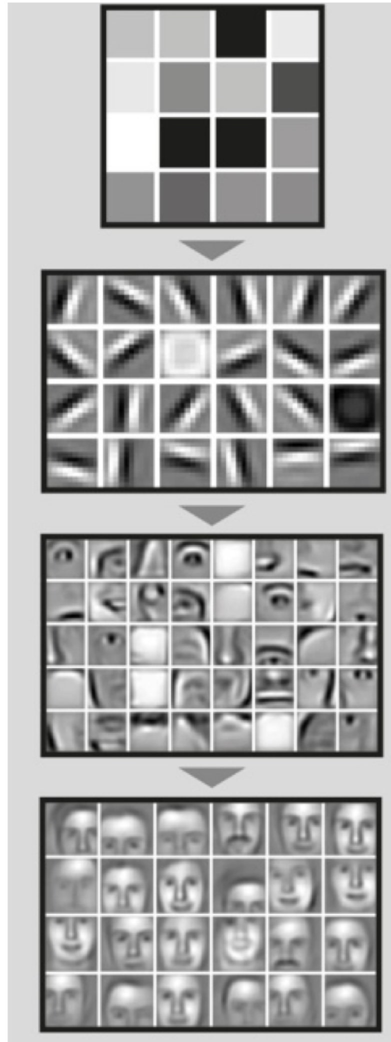
Machine Learning Workshop:
Deep Convolutional Neural Networks

Why Deep Learning?

- Deep learning enables machine learning to not only learn how features can be used for classification, but also learn the features themselves from the most basic representations
- Deep learning is a process of learning simple patterns in the data and building off of those patterns to learn more complex ones

Deep Learning for Facial Recognition



(Andrew Ng)

Input Layer:
Different Intensities of Pixels

Layer 1: Learn simple shapes

Layer 2: Learn More Complex Shapes and objects

Layer 3: Learn compositions of complex shapes that define a face

Global vs Local Feature Learning

A typical classifier would look at each word in this sentence independently and try to identify a relationship between the frequency of words with the class label associated with it

Is identical to:

it A relationship look classifier at associated each in this try to a between sentence and the would frequency of with typical the class label with words it independently identify word

For most encodings and machine learning classifiers.
(e.g., word frequency)

Global vs Local Feature Learning

A typical classifier would look at each word in this sentence independently and try to identify a relationship between the frequency of words with the class label associated with it

a	it	try	identify	…
2	1	1	1	…

it A relationship look classifier at associated each in this try to a between sentence and the would frequency of with typical the class label with words it independently identify word

a	it	try	identify	…
2	1	1	1	…

Global vs Local Feature Learning

But when we read, we look at neighbors to understand the context and meaning:

A typical classifier would look at each word in this sentence independently and try to identify a relationship between the frequency of words with the class label associated with it

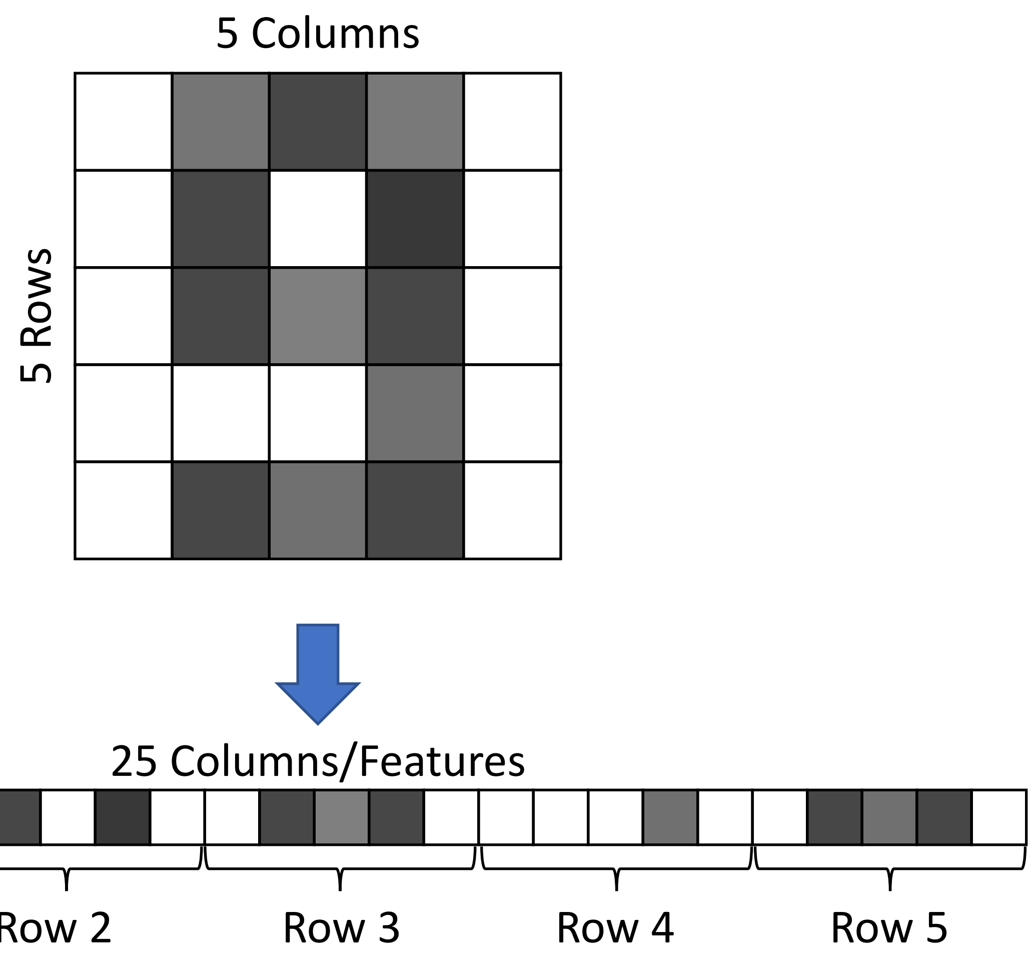
The position of the words with respect to one another is just as important as the words themselves!

Classifiers that can learn local contexts:

Deep Convolutional Neural Networks

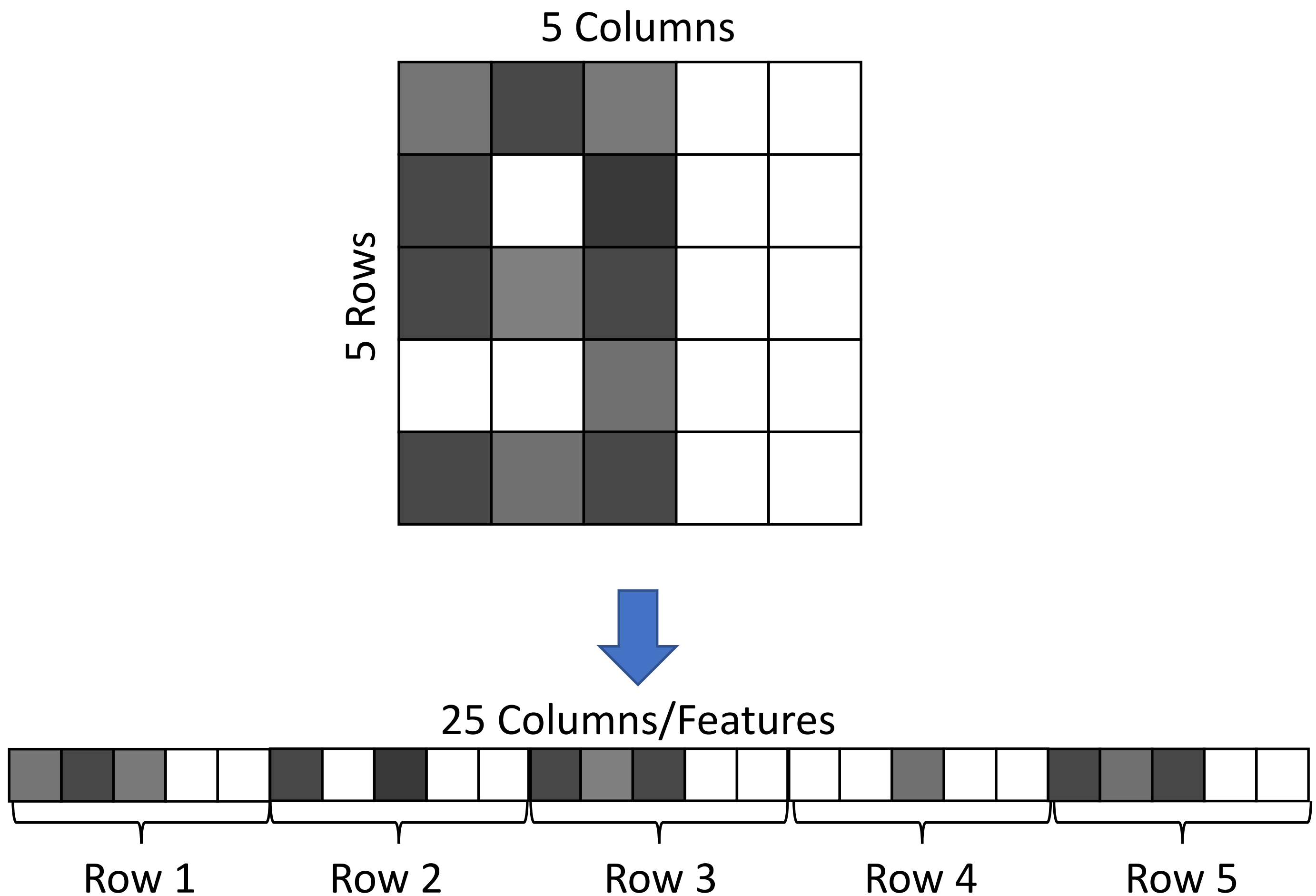
Recurrent Neural Networks

Revisiting our Number Encoding:

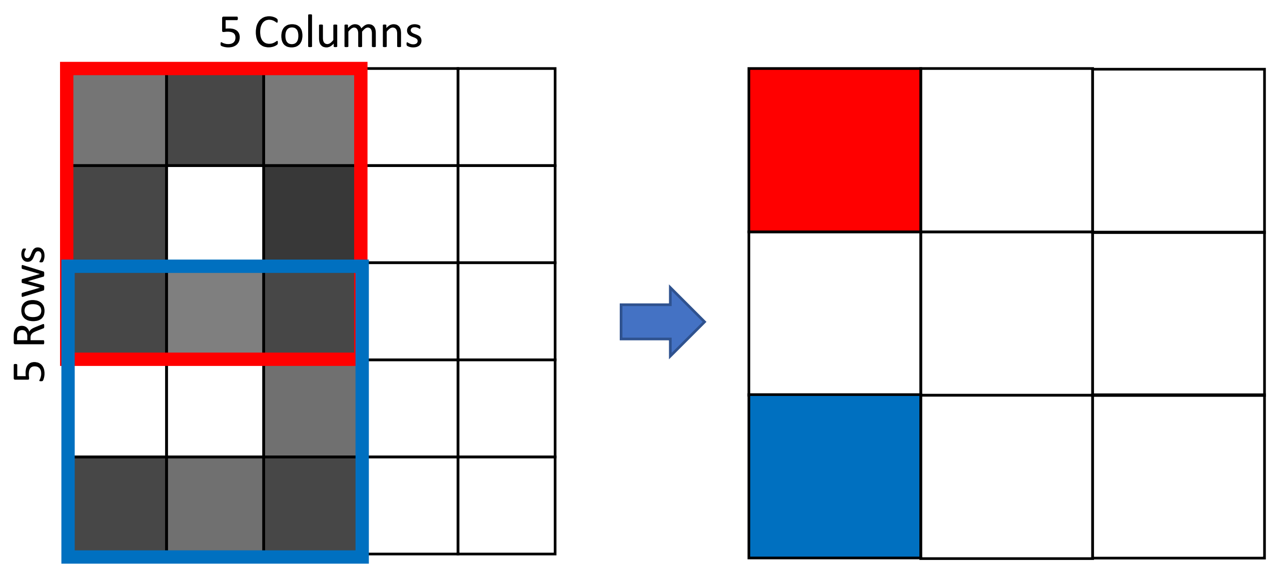


This is known as “**Flattening**”

Problem: Each number can be shifted or represented slightly differently, increasing both model complexity and training data to capture all variations



Problem: Each number can be shifted or represented slightly differently, increasing both <u>model complexity</u> and <u>training data</u> to capture all variations



We can start to deconstruct our number into components, and then use these components to predict the number being represented by the image!

The Convolution Function

The convolution function transforms the data by learning patterns from local neighborhoods in the data and generating a new representation of the data using patterns observed

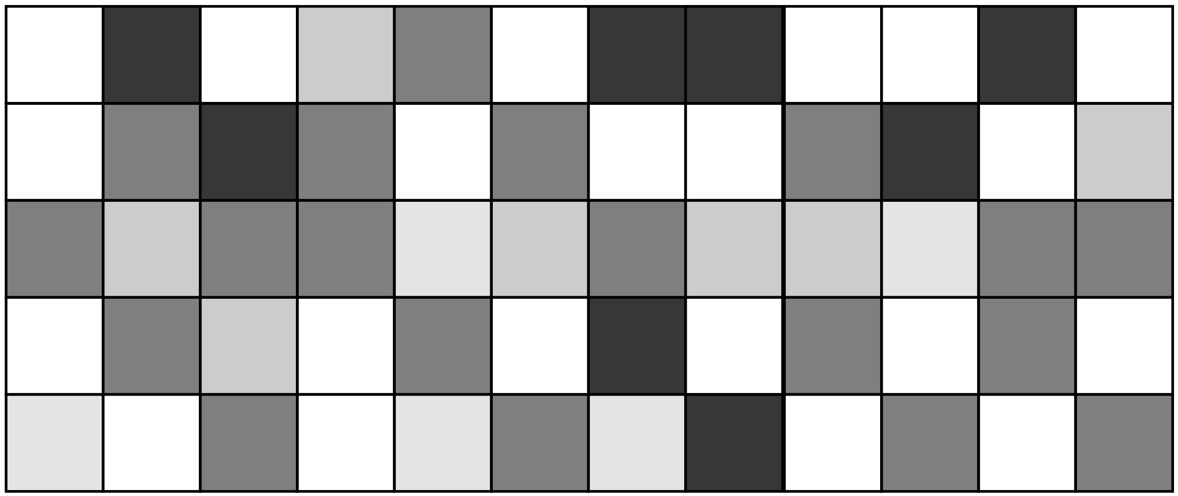
Kernel dimensions: The size of the window/tensor used to examine data points in close proximity

 # Filters: The number of filters to train on the data

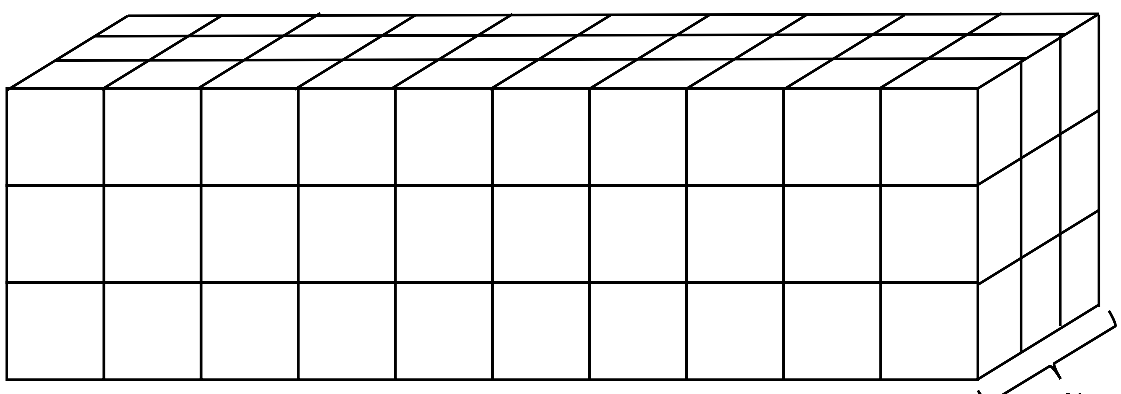
Strides: How the window will translate/slide across the data to generate the new output

Convolution Example

Input:



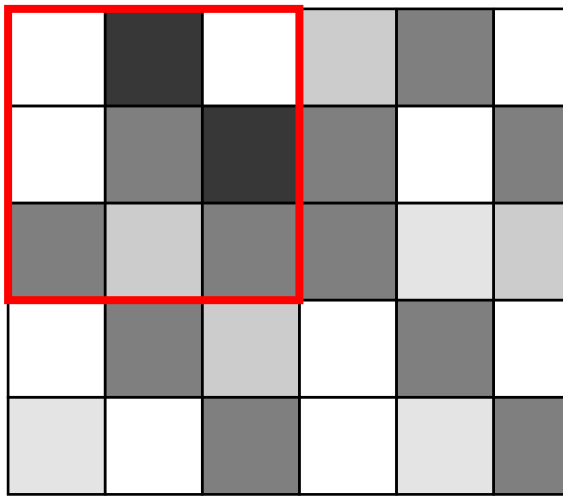
Output:



Convolution Example

Input:

Kernel Window/ Tensor

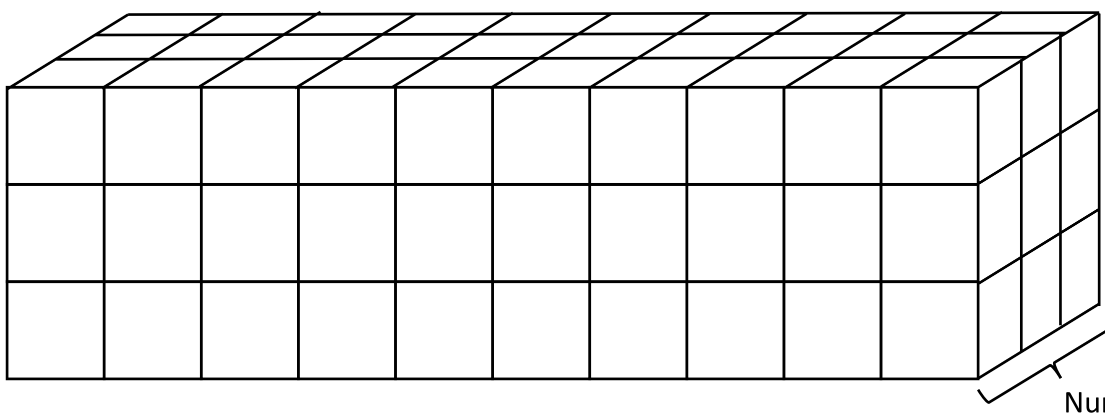


$$\begin{pmatrix} 0.0, & 1.0, & 0.0 \\ 0.0, & 0.5, & 1.0 \\ 0.5, & 0.3, & 0.5 \end{pmatrix} * \begin{pmatrix} w1, & w2, & w3 \\ w4, & w5, & w6 \\ w7, & w8, & w9 \end{pmatrix}$$

Observed　　　　Filter 1

0.0*w1+1.0*w2+0.0*w3+
0.0*w4+0.5*w5+1.0*w6+
0.5*w7+0.3*w8+0.5*w9

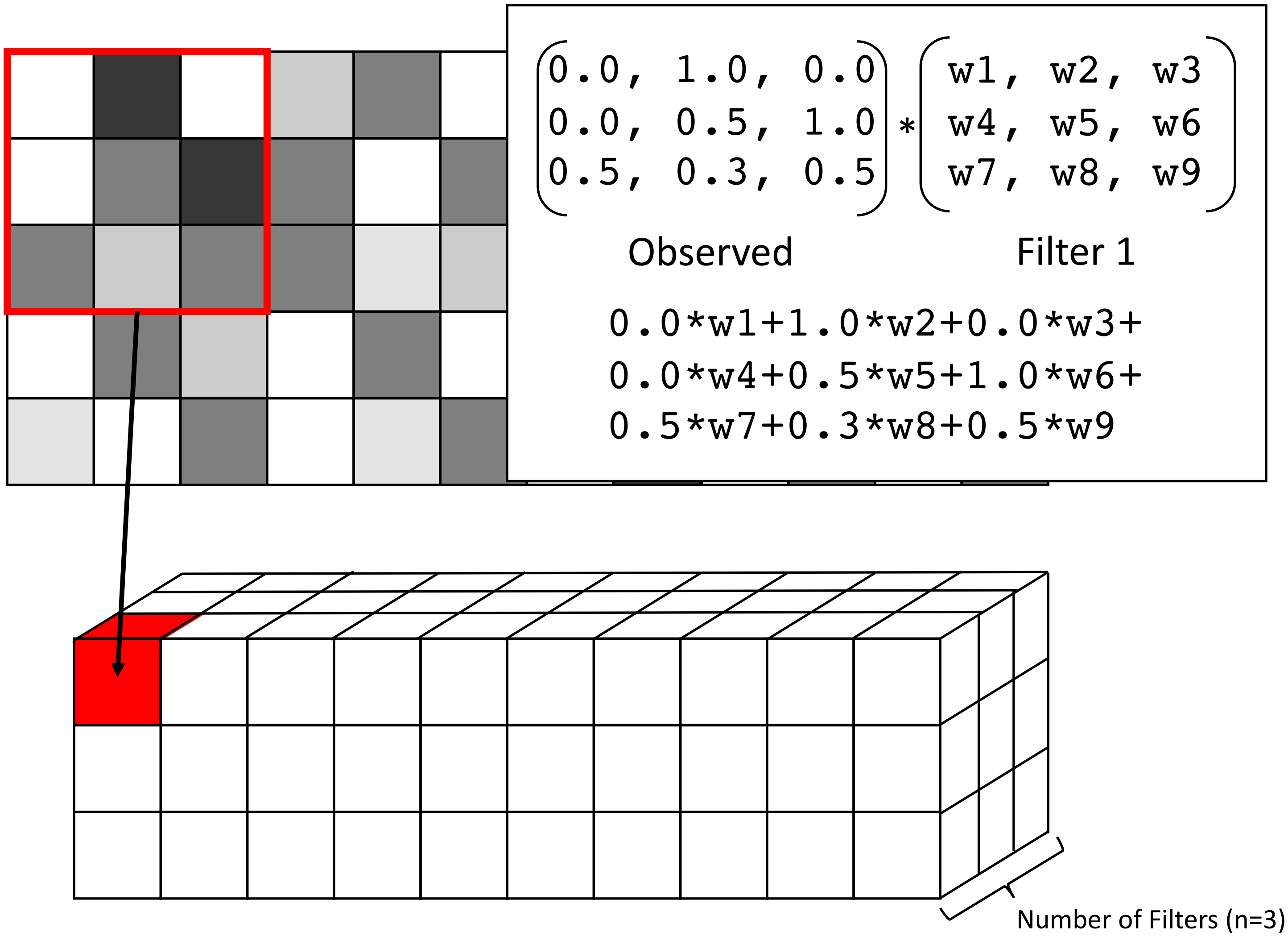
Output:



Convolution Example

Input:

Output:

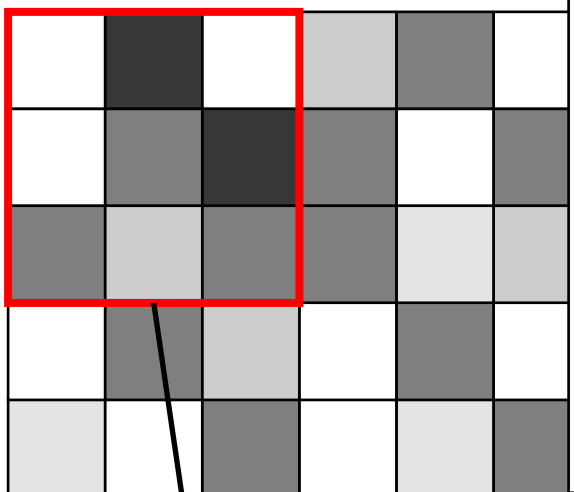


$$\begin{pmatrix} 0.0, & 1.0, & 0.0 \\ 0.0, & 0.5, & 1.0 \\ 0.5, & 0.3, & 0.5 \end{pmatrix} * \begin{pmatrix} w1, & w2, & w3 \\ w4, & w5, & w6 \\ w7, & w8, & w9 \end{pmatrix}$$

$$0.0*w1+1.0*w2+0.0*w3+$$
$$0.0*w4+0.5*w5+1.0*w6+$$
$$0.5*w7+0.3*w8+0.5*w9$$

Convolution Example

Input:

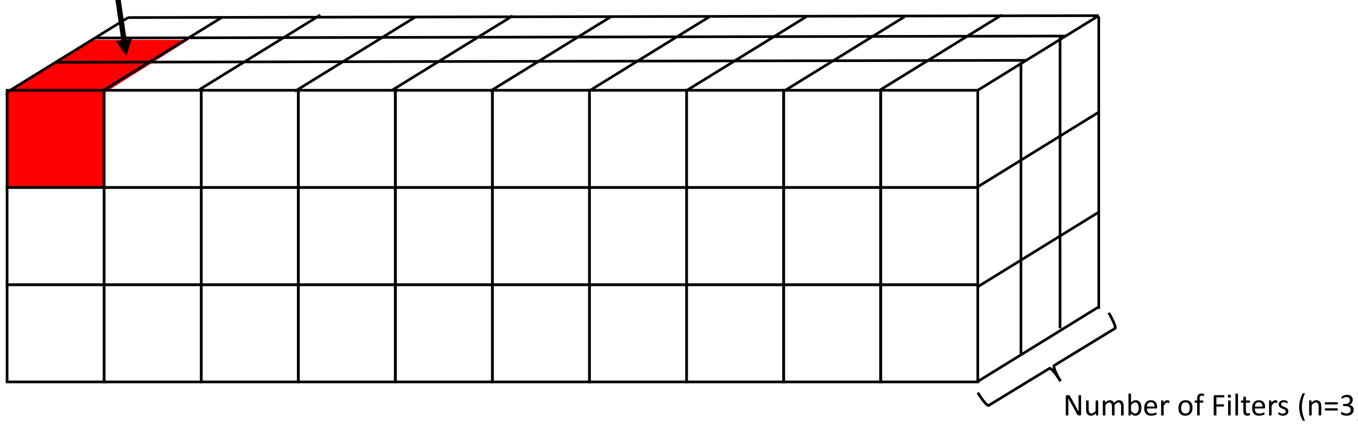


$$\begin{pmatrix} 0.0, & 1.0, & 0.0 \\ 0.0, & 0.5, & 1.0 \\ 0.5, & 0.3, & 0.5 \end{pmatrix} * \begin{pmatrix} w1, & w2, & w3 \\ w4, & w5, & w6 \\ w7, & w8, & w9 \end{pmatrix}$$

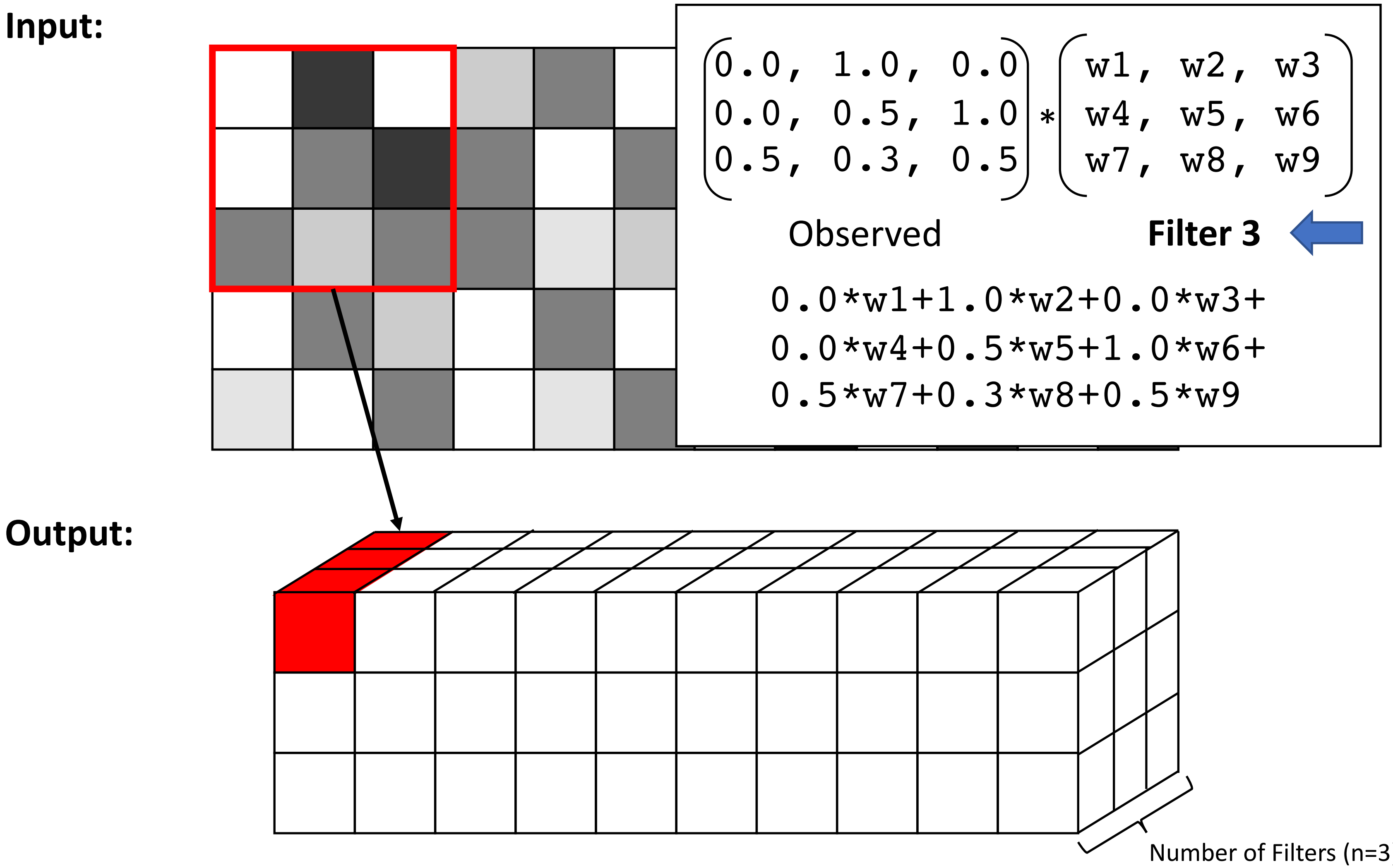
Observed **Filter 2**

0.0*w1+1.0*w2+0.0*w3+
0.0*w4+0.5*w5+1.0*w6+
0.5*w7+0.3*w8+0.5*w9

Output:

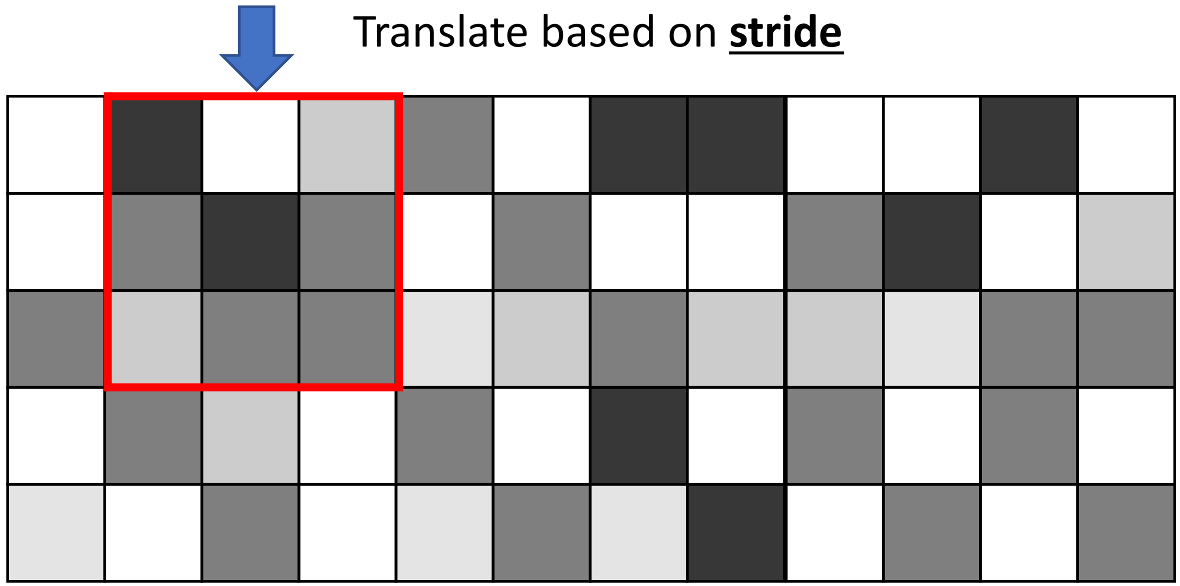


Convolution Example

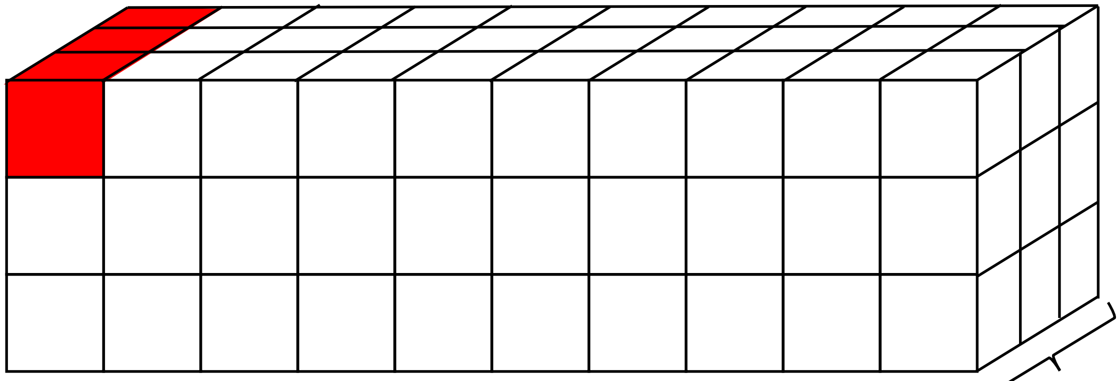


Convolution Example

Input:

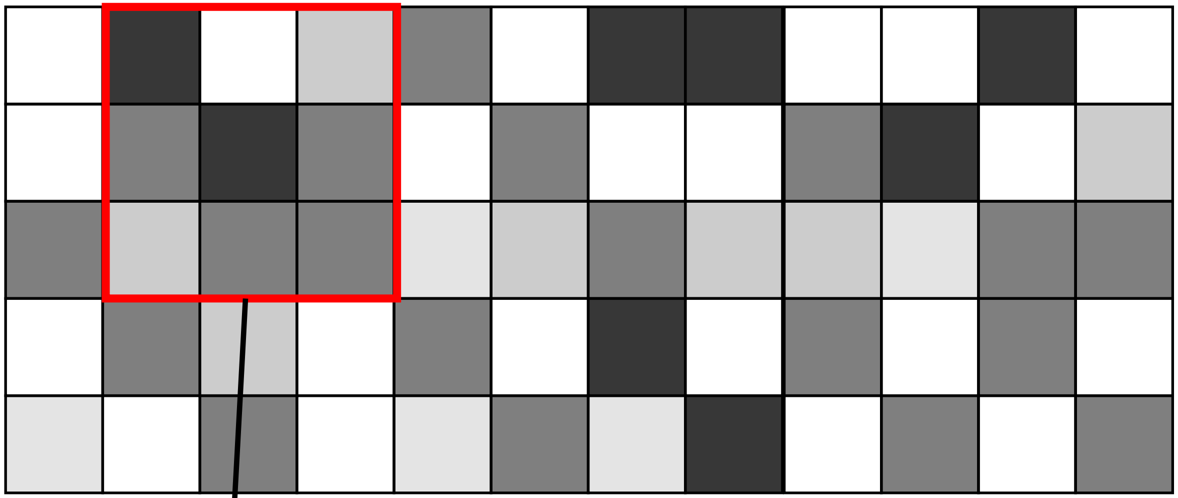


Output:

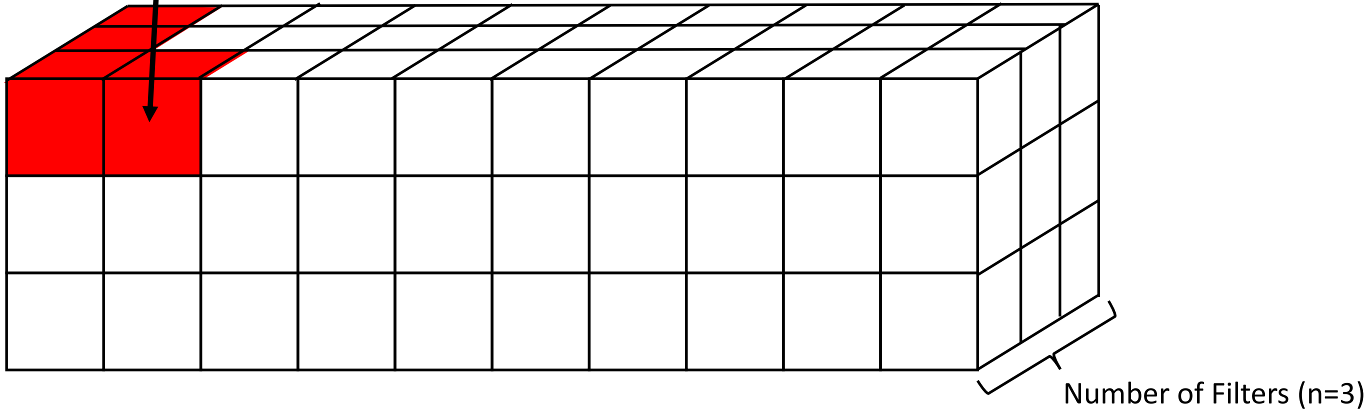


Convolution Example

Input:

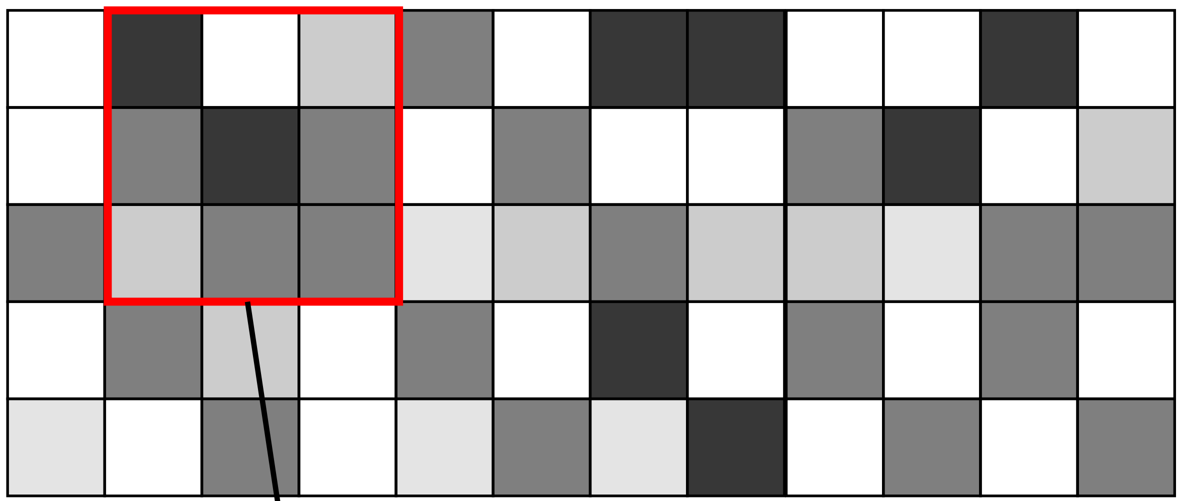


Output:

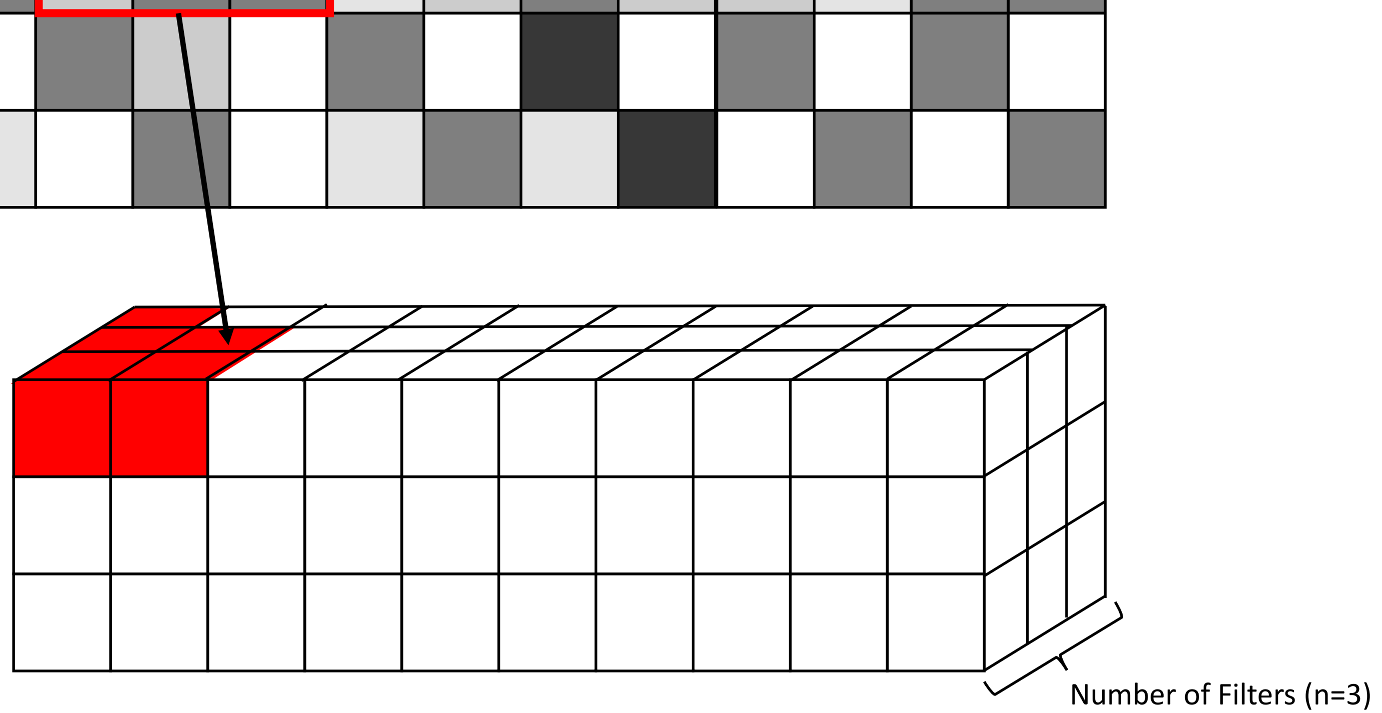


Convolution Example

Input:



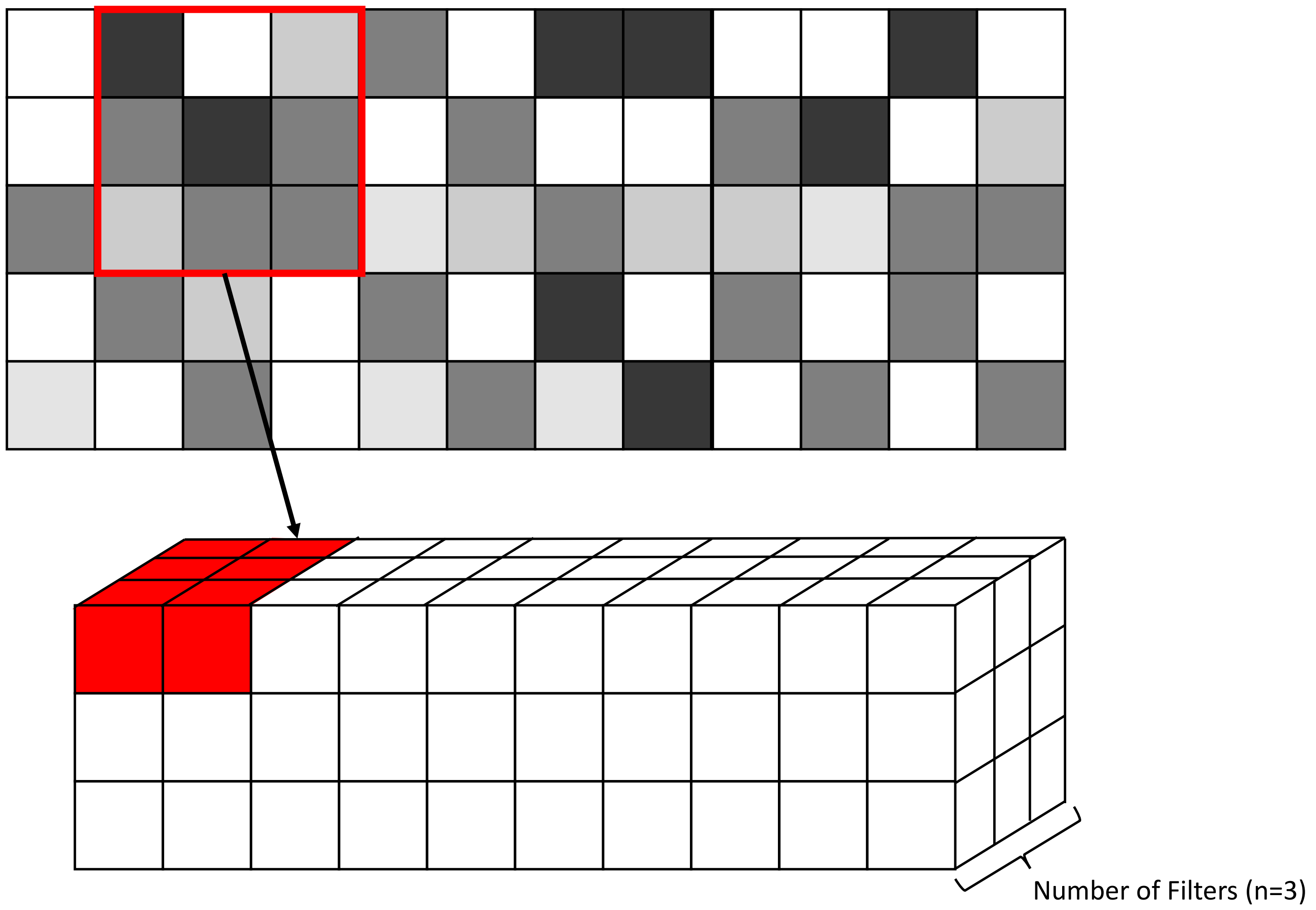
Output:



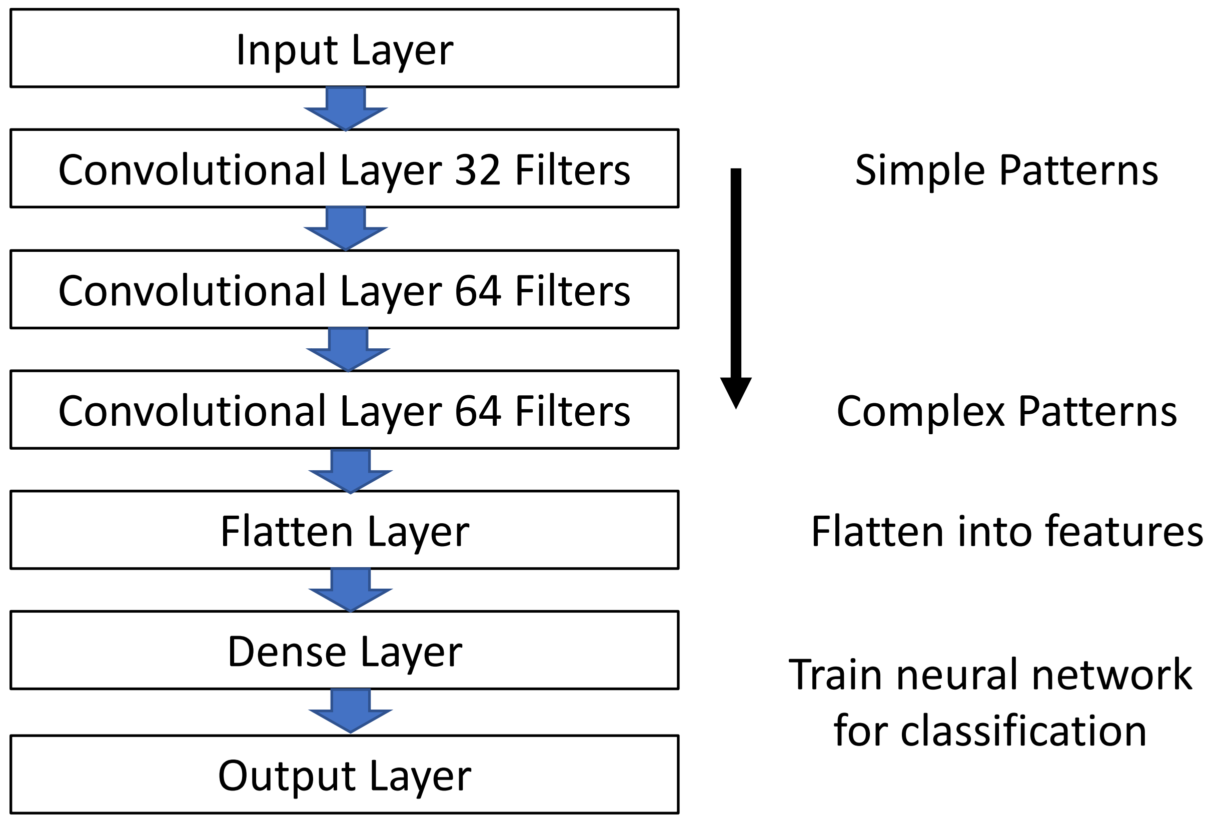
Convolution Example

Input:

Output:

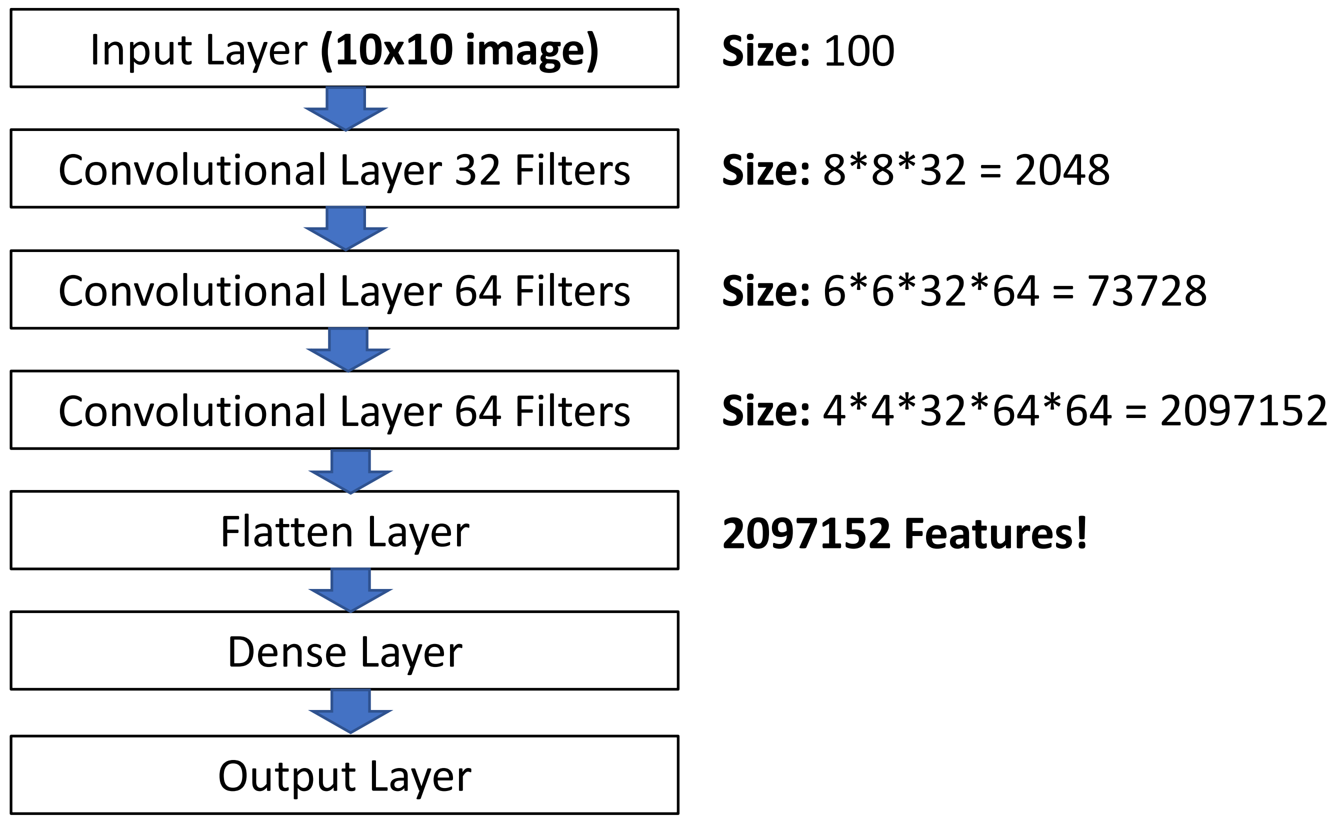


A Typical Deep CNN Architecture



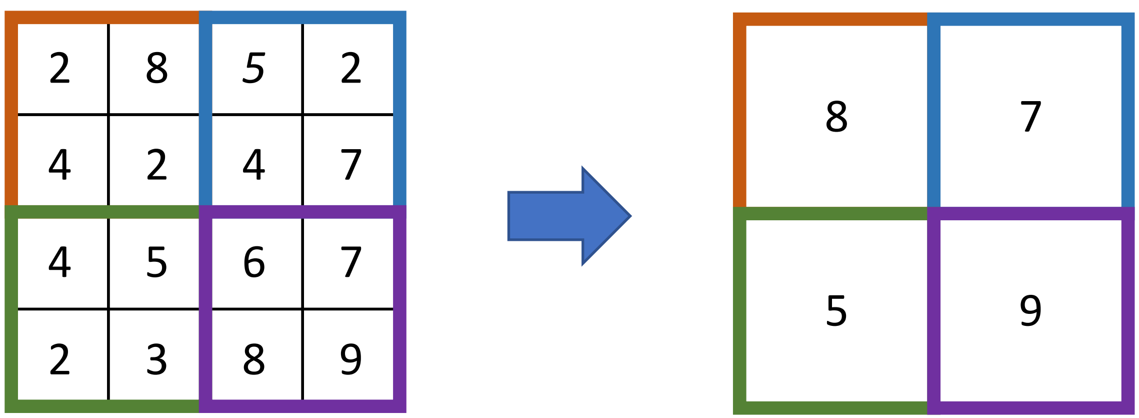
Problem with convolution: The number of features grows as we go deeper in the network

Assuming 3x3 kernels for each convolutional layer



Max Pooling

Basic Idea: Reduce the the dimensions of the data by taking the maximum value within a specified window



Problem with convolution: The number of features grows as we go deeper in the network

	Input Layer **(10x10 image)**	**Size:** 100
Assume a 3x3 convolutional layers	Convolutional Layer 32 Filters	**Size:** 8*8*32 = 2048
2x2 Maxpooling	Max Pooling	**Size:** 4x4*32 = 512
	Convolutional Layer 64 Filters	**Size:** 2*2*32*64 = 8192

We actually reach our limit on convolutions sooner and are forced to stop here and move to flattening into features and training a dense neural network

Overcoming Overfitting with Dropout

Basic idea: randomly set a certain percentage of features to 0, forcing the machine learning algorithm to learn consistent patterns in the data

"Soon after that I went to my bank. The tellers kept changing and I asked one of them why. He said he didn't know but they got moved around a lot. I figured it must be because it would require cooperation between employees to successfully defraud the bank. This made me realize that randomly removing a different subset of neurons on each example would prevent conspiracies and thus reduce overfitting."

– Geoff Hinton

Dropout Layer Example

2	8	5	2
4	2	4	7
4	5	6	7
2	3	8	9

50% Drop Out

2	8	5	0
0	0	0	7
4	5	0	0
2	0	8	0

Drop out layers add random noise to the data, making it harder to overfit

Keras Exercise

Three Key Questions to Ask for Machine Learning

1. What is the input data? What is the ground truth?

2. What are the features/data? How is the data encoded?

3. What model will be used? What will be the architecture?