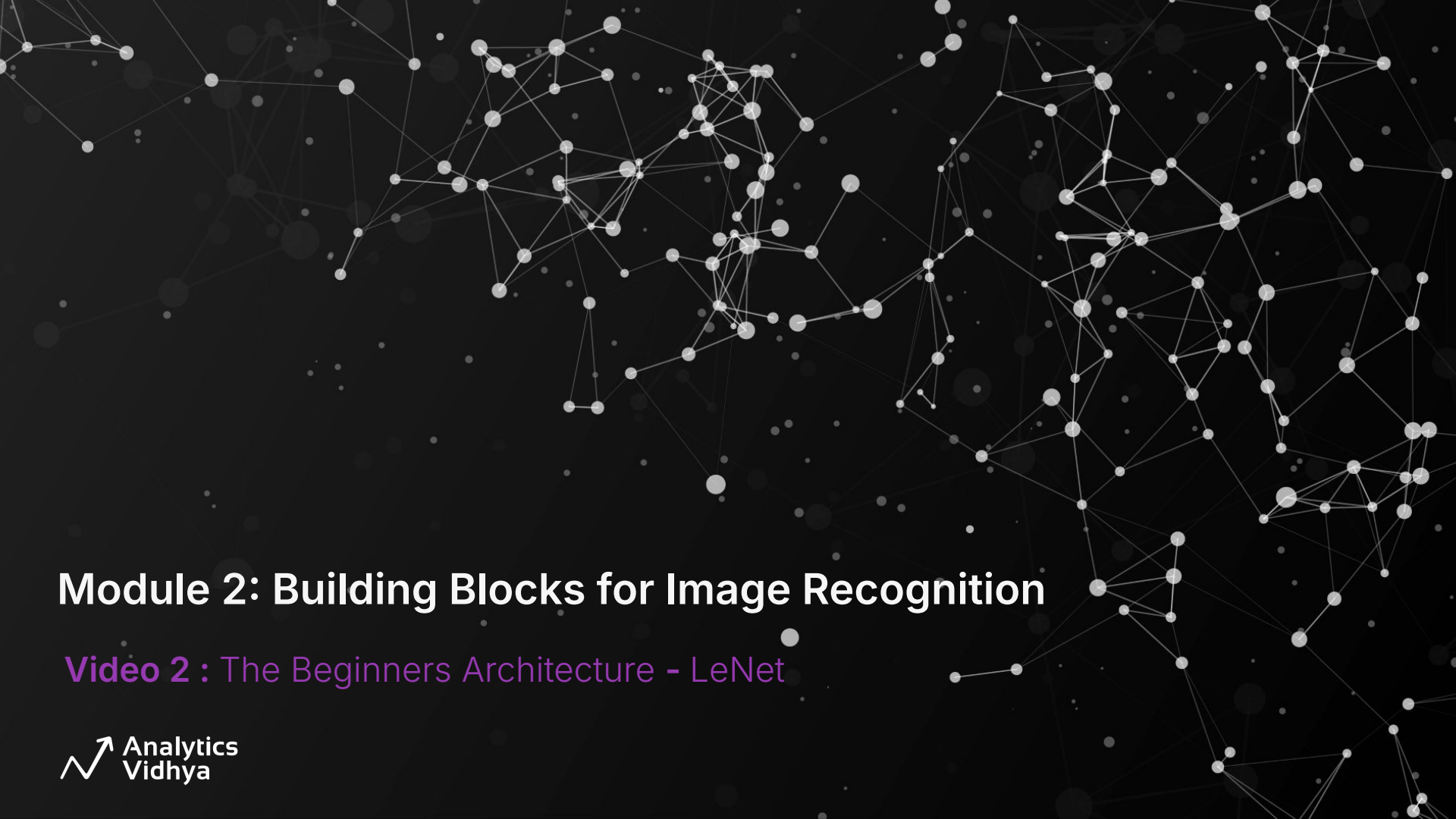


Module 2: Building Blocks for Image Recognition

Video 2 : The Beginners Architecture - LeNet

Analytics Vidhya

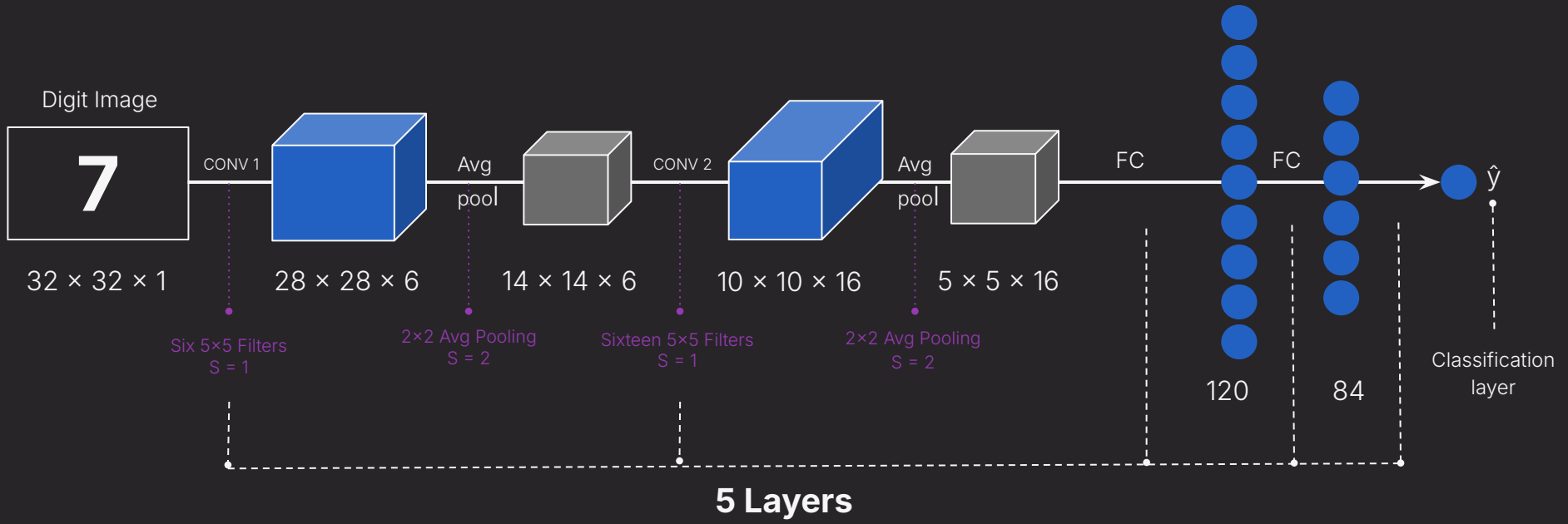
LeNet

- Introduced by **Yann LeCun** in **1998**

Yann LeCun

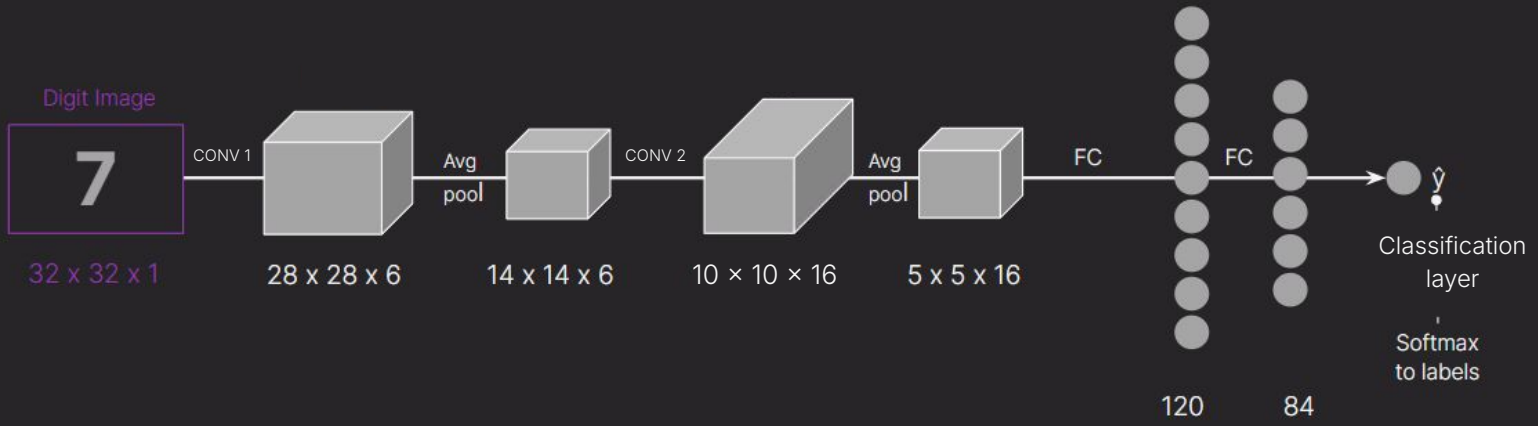
Analytics Vidhya

LeNet



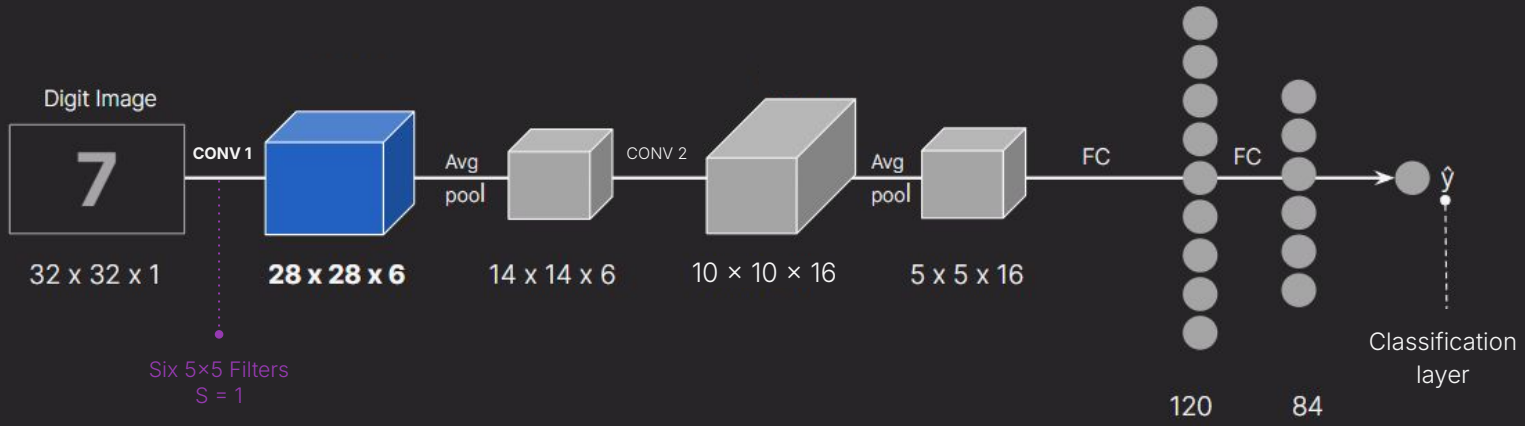
LeNet: The Input Layer

- **The Input Layer:** Accepts a 32×32×1 grayscale image, suitable for processing simple black-and-white visuals.

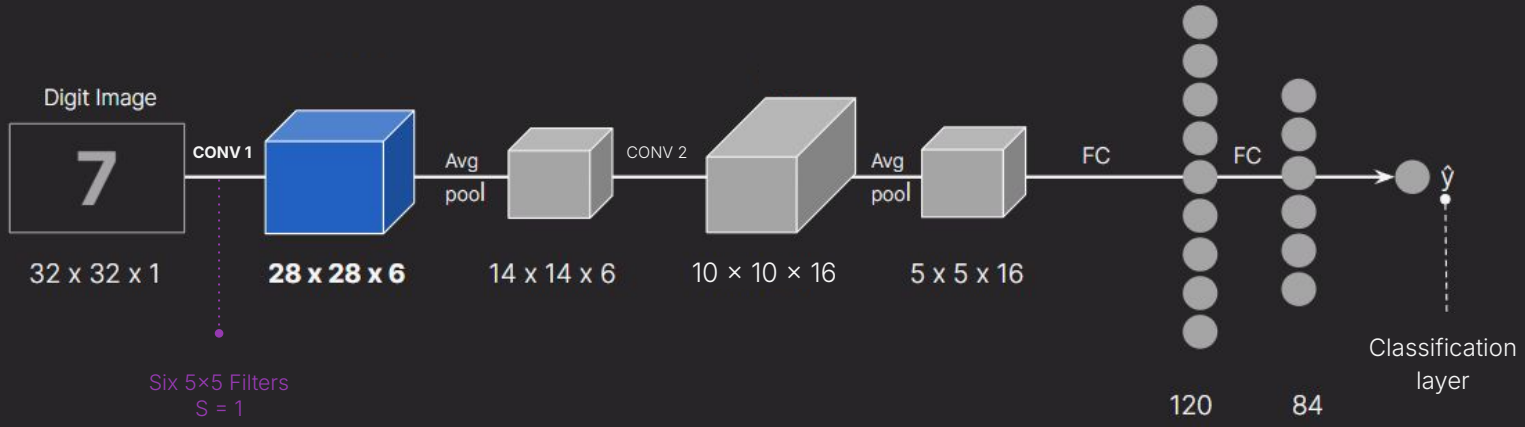


LeNet: The Convolution Layer 1

- **Convolutional layer:** Applies six 5×5 filters to the 32×32 input image (p=0) to give an output of six 28×28 channels.



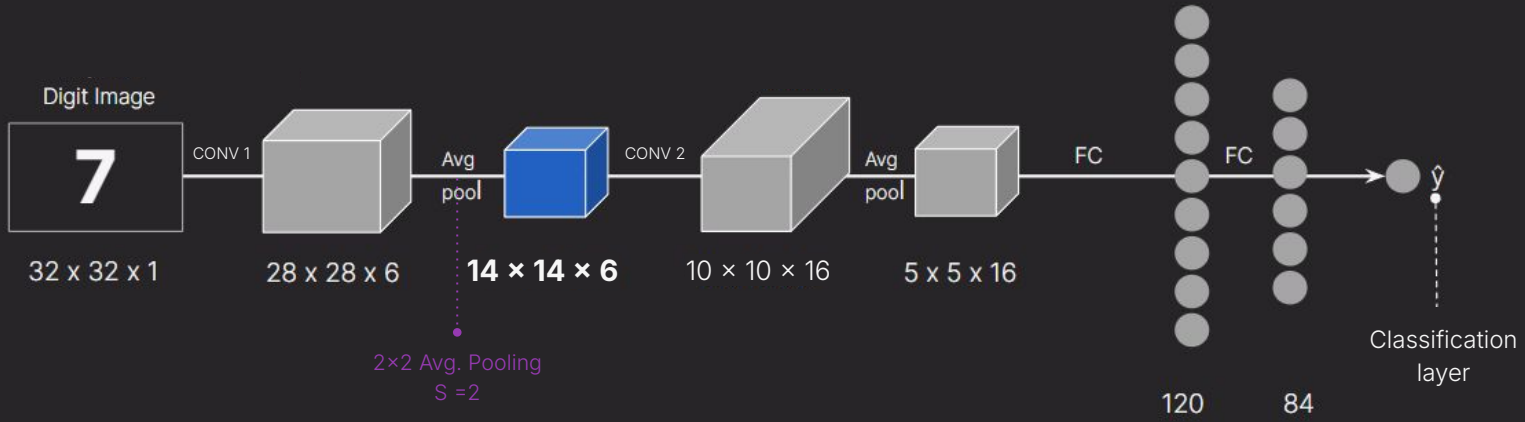
LeNet: The Convolution Layer 1



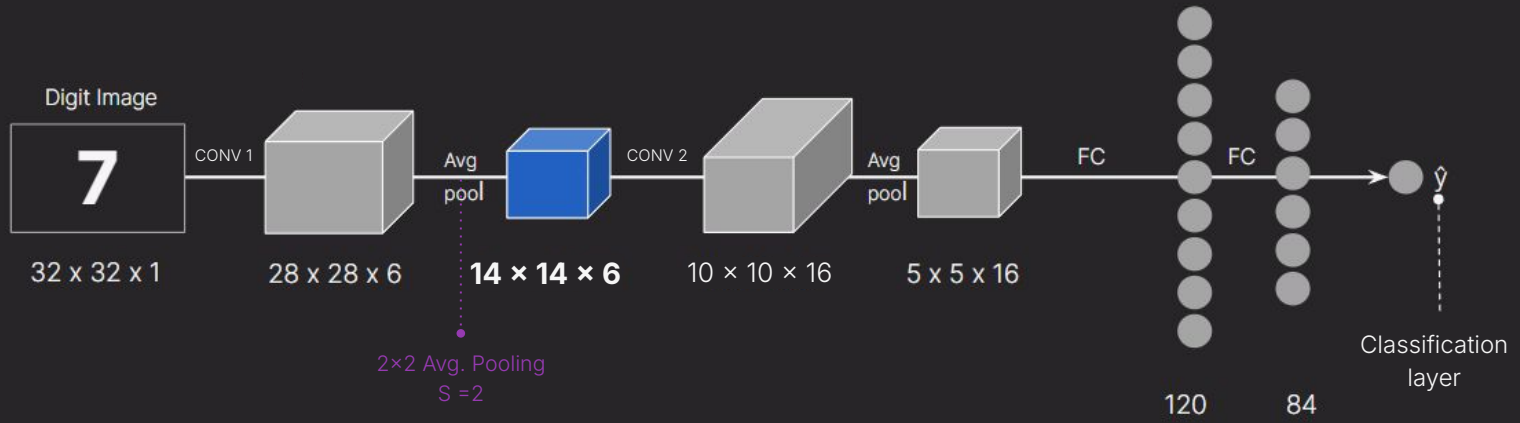
$$\textbf{Output size} = \frac{\text{Input size - Filter size + 2 * Padding}}{\text{Stride}} + 1 = \frac{32 - 5 + 2 \times 0}{1} + 1 = \mathbf{28}$$

LeNet: The Pooling Layer 1

- **Pooling Process:** Applies a 2×2 average pooling filter with a stride of 2, downsizing six 28×28 feature maps to 14×14.
- **Output Dimensions:** Maintains a depth of 6, resulting in an output of 14×14×6.



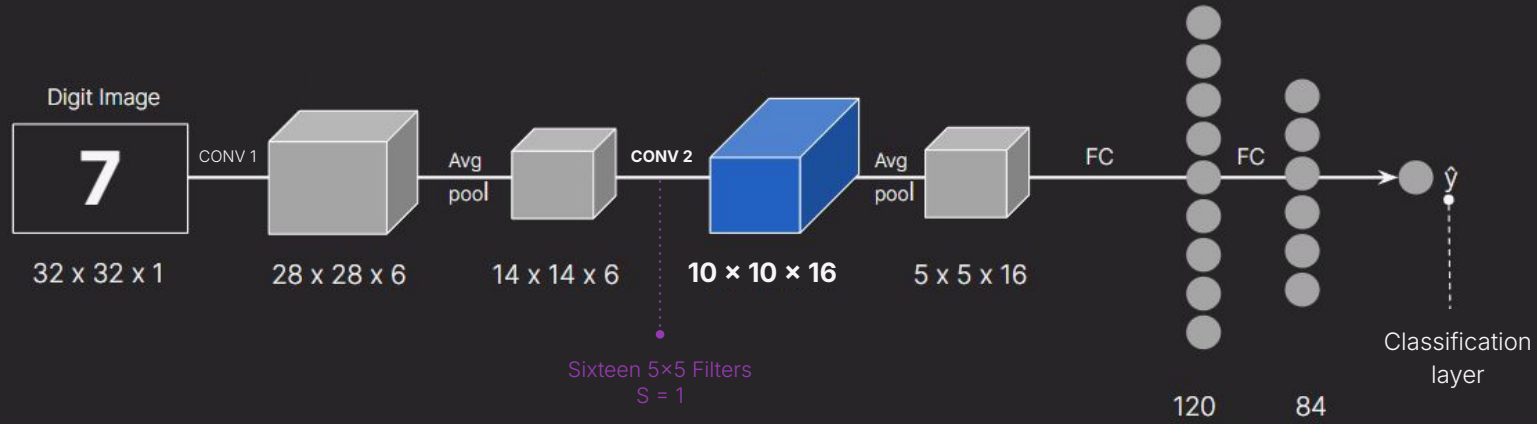
LeNet: The Pooling Layer 1



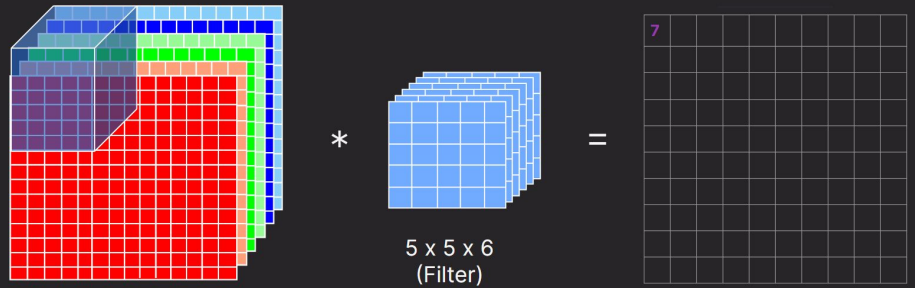
$$\textbf{Output size} = \frac{\text{Input size - Filter size + 2 * Padding}}{\text{Stride}} + 1 = \frac{28 - 2}{2} + 1 = \mathbf{14}$$

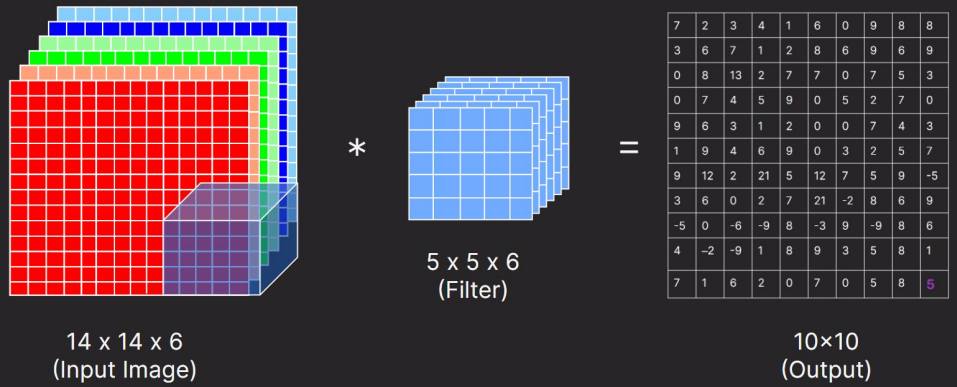
LeNet: The Convolution Layer 2

- **Convolution Layer:** Uses sixteen 5×5 filters of depth 6 to process the pooled feature maps from the previous layer.
- **Increased Depth:** With 16 filters, the depth of the output increases, resulting in feature maps sized 10×10×16.

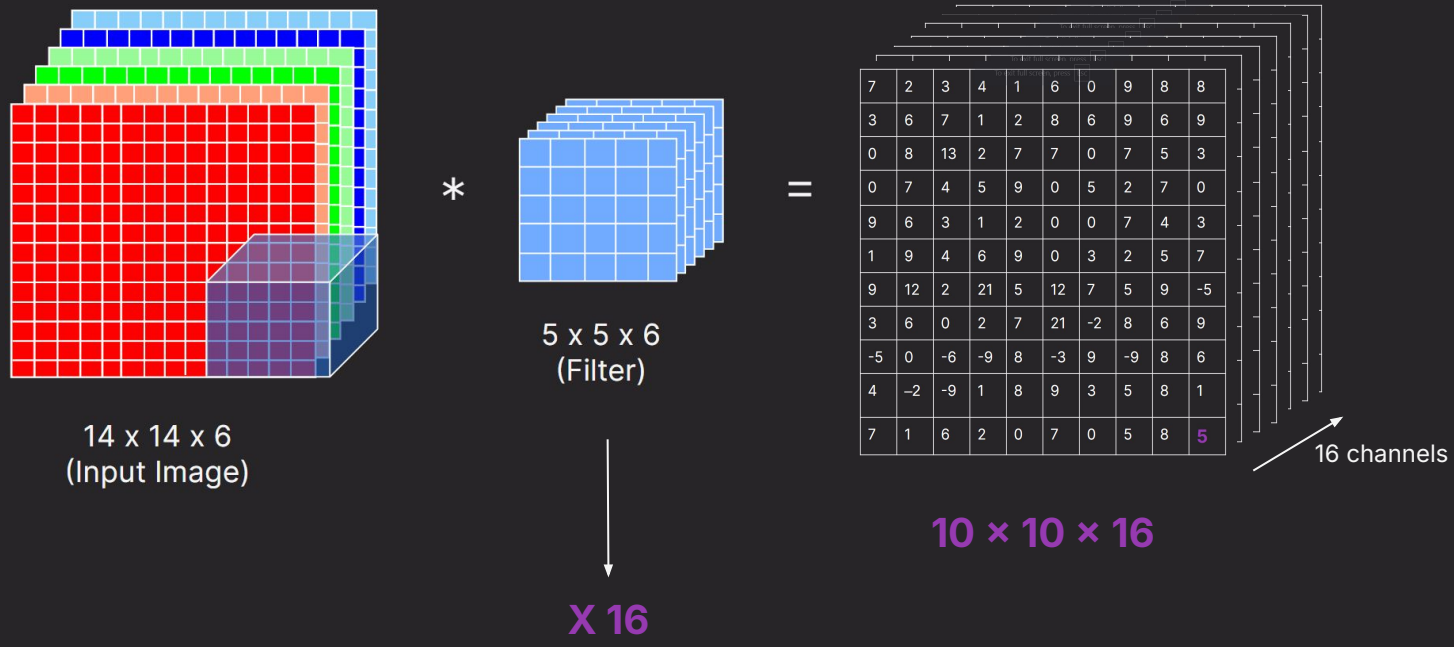


LeNet: Convolution on an RGB Image



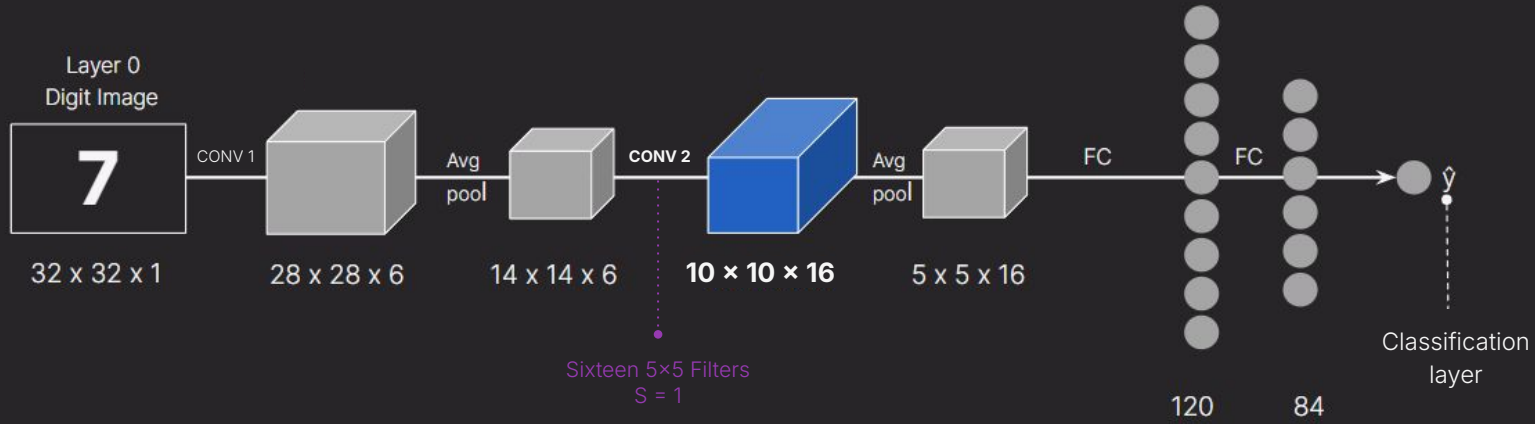


LeNet: Convolution on an RGB Image



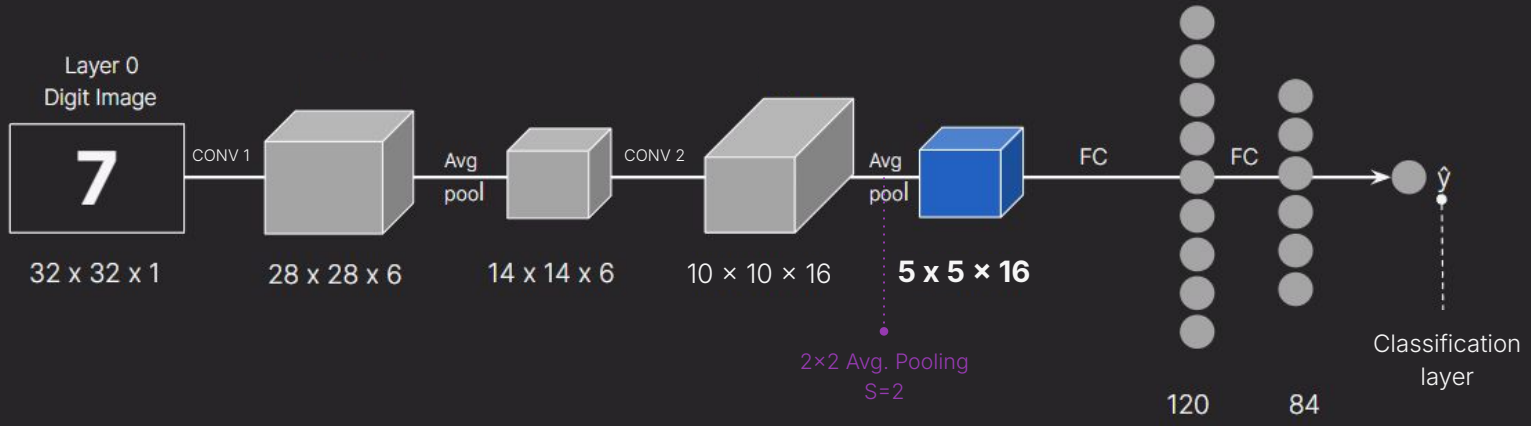
LeNet: The Convolution Layer 2

- **Convolution Layer:** Uses sixteen 5×5 filters of depth 6 to process the pooled feature maps from the previous layer.
- **Increased Depth:** With 16 filters, the depth of the output increases, resulting in feature maps sized 10×10×16.



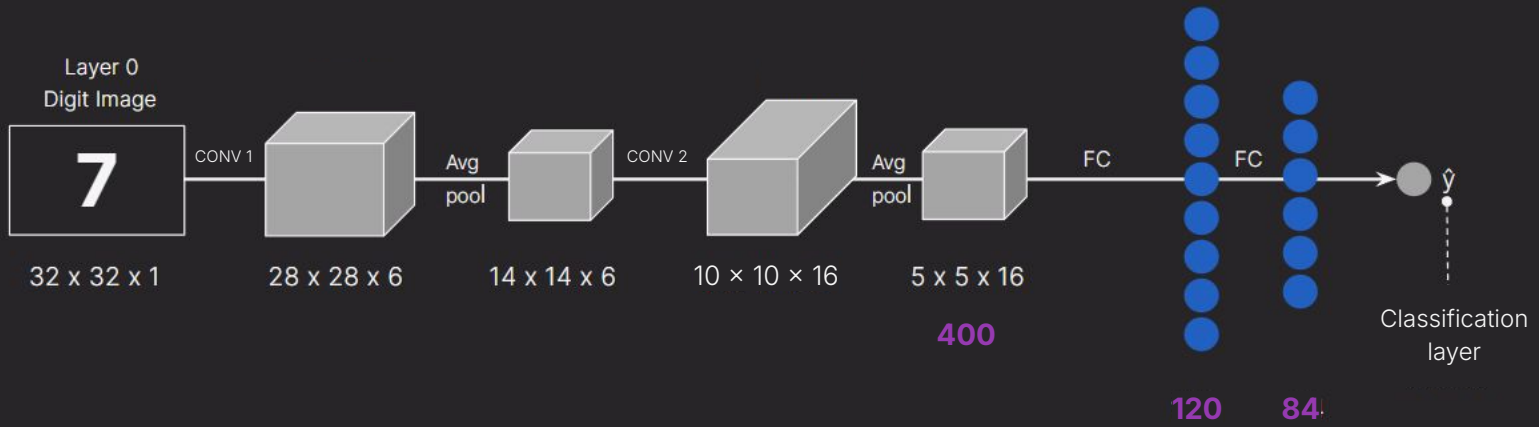
LeNet: The Pooling Layer 2

- **Pooling Process:** Applies a 2×2 average pooling filter with a stride of 2, downsizing sixteen 10×10 feature maps to 5×5.
- **Output Dimensions:** Maintains a depth of 16, resulting in an output of 5×5×16.



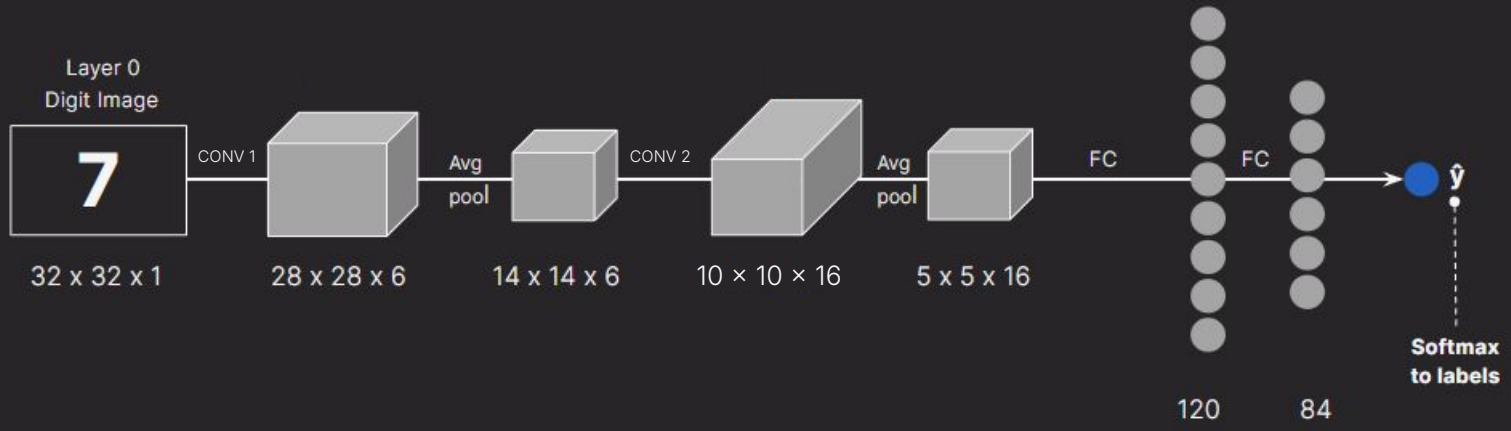
LeNet: The Fully Connected Layers

- Flattened into a **400-element vector**, and serves as input to a fully connected layer with 120 neurons.
- Further the 120-neuron layer is connected to another fully connected layer with 84 neurons.



LeNet: The Output Layer

- **Output layer:** 10 neurons, each for a digit class from 0 to 9.
- **Probability calculation:** Softmax function converts neuron scores to probabilities, predicting the output based on the highest probability neuron.



Analytics Vidhya

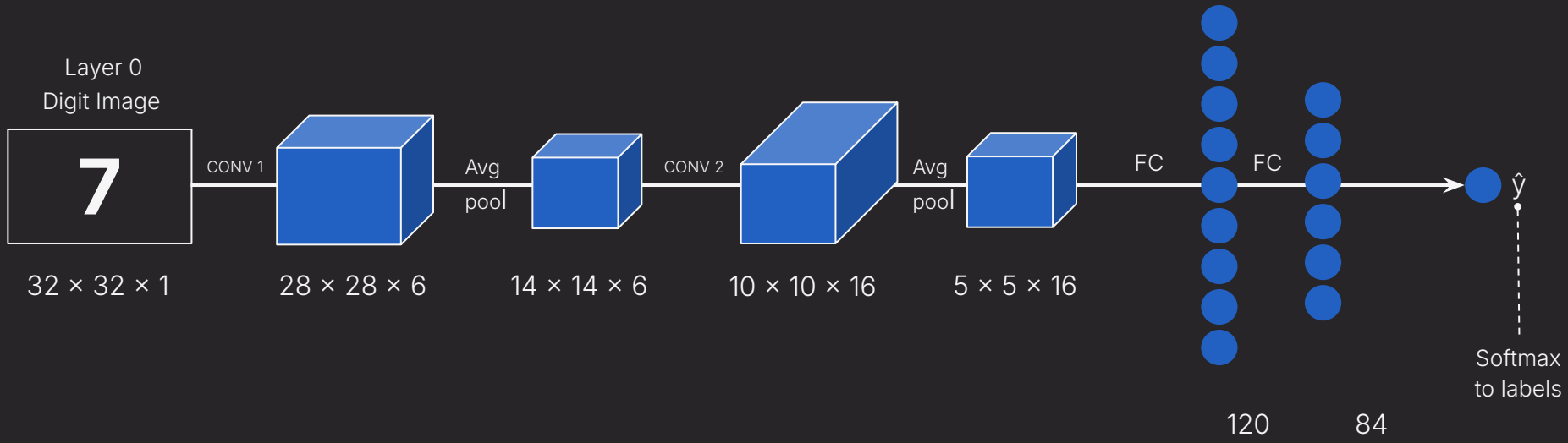
LeNet: Deviations from Modern CNNs

Activation Functions: LeNet used sigmoid or tanh; modern CNNs often use ReLU.

Operation Sequence: LeNet applied activations after pooling; modern CNNs usually apply activations before pooling.

Softmax Layer: LeNet did not use Softmax for classification.

LeNet


Layer 0
Digit Image
7
CONV 1
Avg
pool
CONV 2
Avg
pool
FC
FC
ŷ
Softmax
to labels
32 × 32 × 1
28 × 28 × 6
14 × 14 × 6
10 × 10 × 16
5 × 5 × 16
120
84

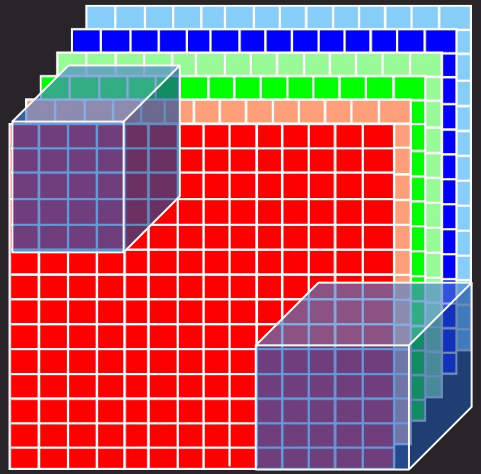
Analytics Vidhya

UpNext: Building an Image Classification Model using LeNet

Analytics Vidhya

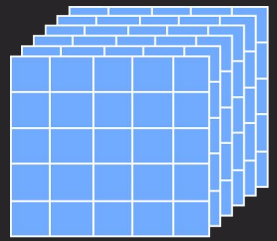
7	2	3	4	1	6	0	9	8	8
3	6	7	1	2	8	6	9	6	9
0	8	13	2	7	7	0	7	5	3
0	7	4	5	9	0	5	2	7	0
9	6	3	1	2	0	0	7	4	3
1	9	4	6	9	0	3	2	5	7
9	12	2	21	5	12	7	5	9	-5
3	6	0	2	7	21	-2	8	6	9
-5	0	-6	-9	8	-3	9	-9	8	6
4	–2	-9	1	8	9	3	5	8	1
7	1	6	2	0	7	0	5	8	**5**

LeNet: Convolution on an RGB Image



14 × 14 × 6
(Input Image)

*



5 × 5 × 6
(Filter)

=

10×10
(Output)