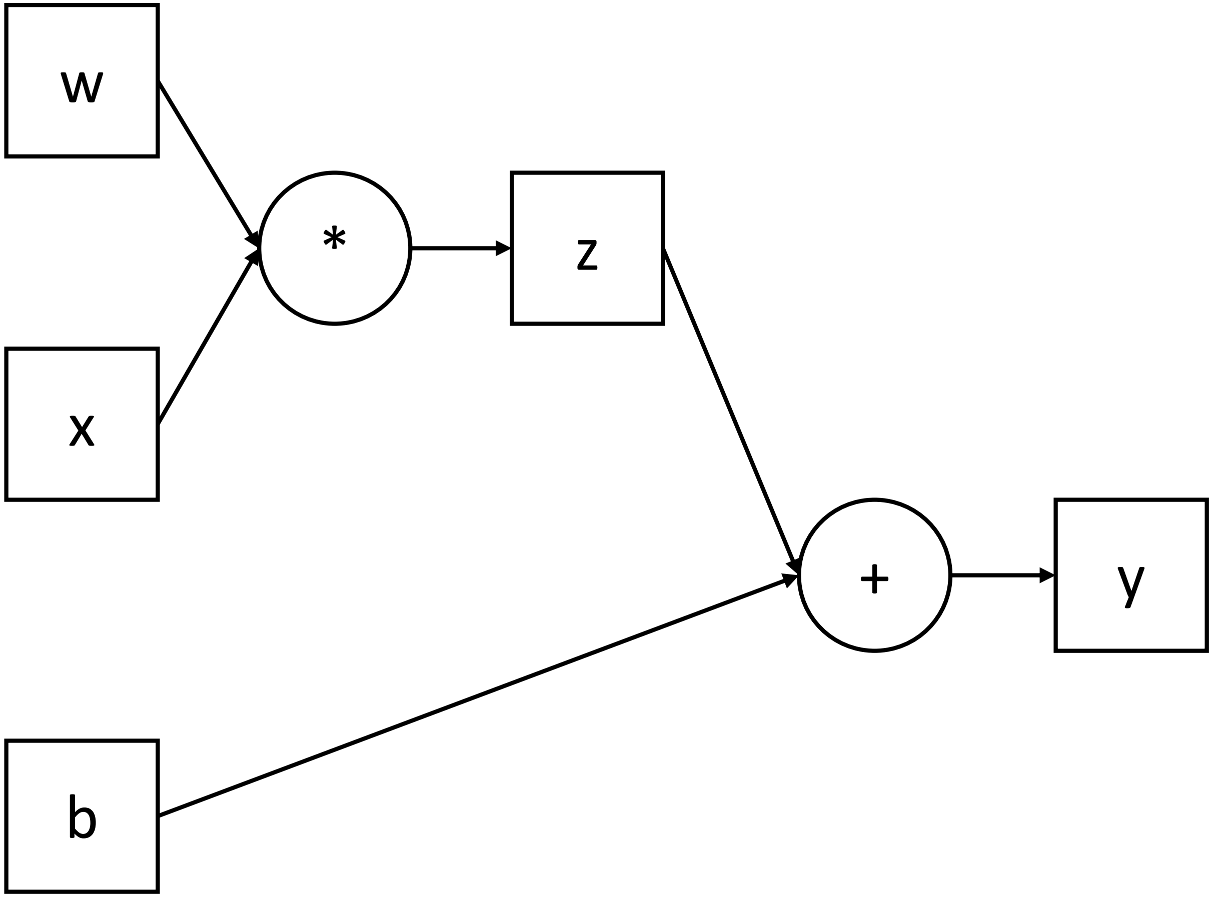
Convolutional Neural Network using TensorFlow

Youngjae Yu (yj.yu@vision.snu.ac.kr)

SEOUL NATIONAL UNIV.
VISION & LEARNING

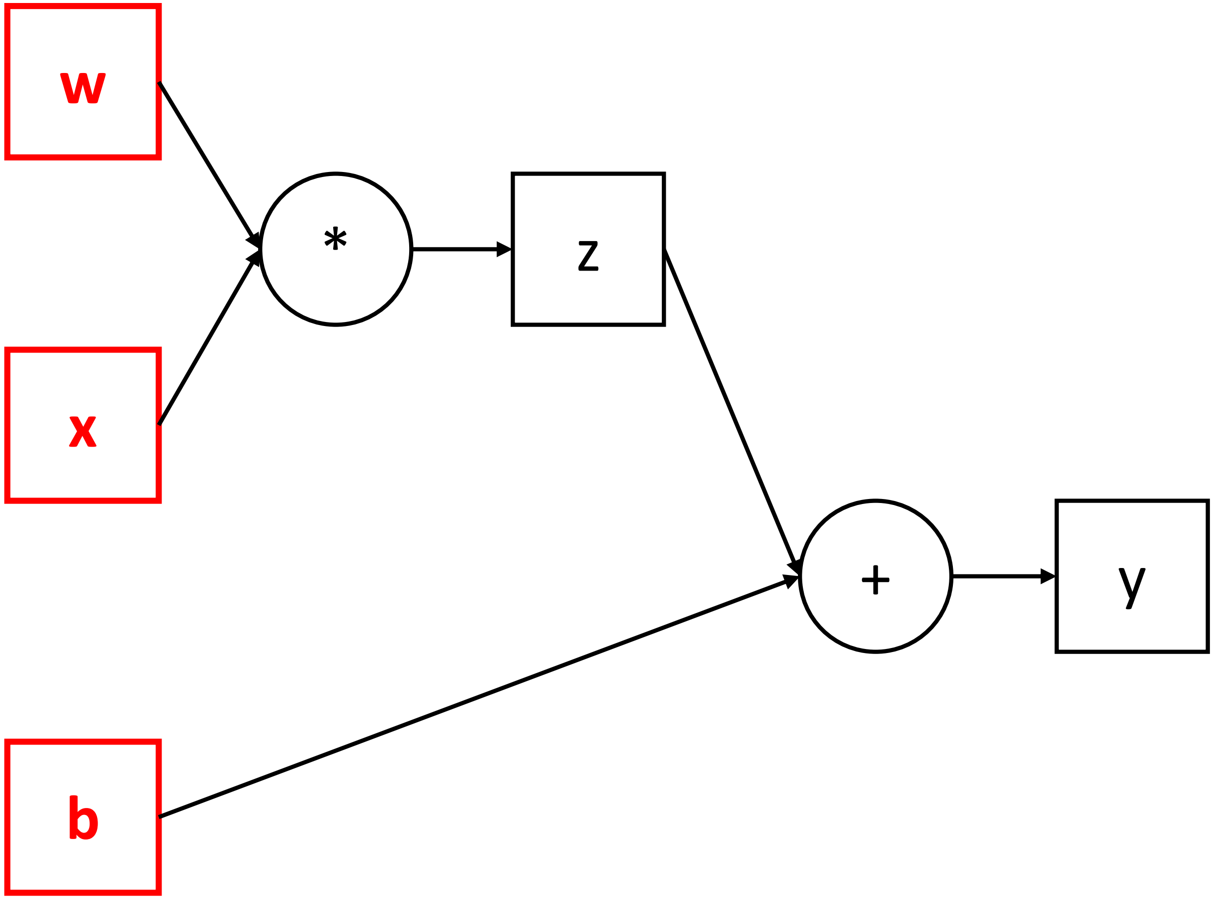
TensorFlow Basics

- Graph example



TensorFlow Basics

- Graph example

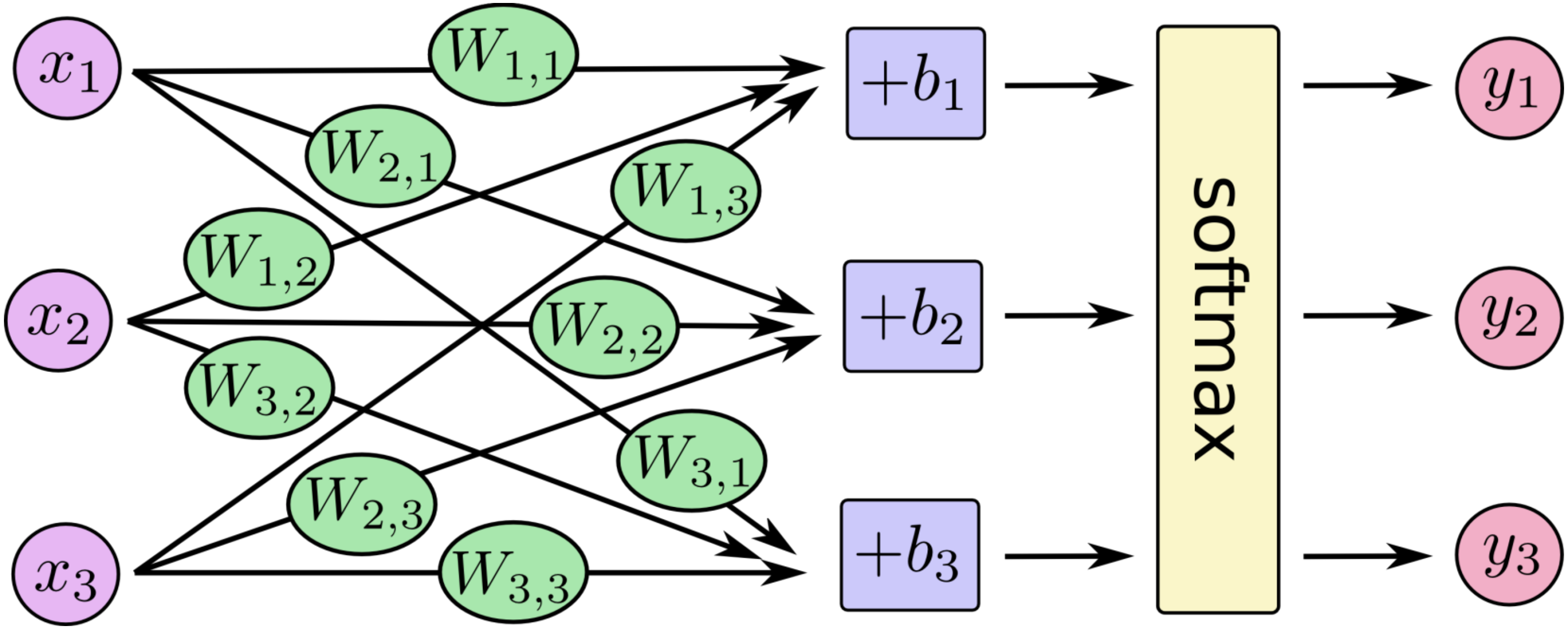


TensorFlow Basics

- Constant
  - `tf.constant(value, dtype, …)`
  - **e.g.** `tf.constant([1, 2, 3])`
- Variable
  - `tf.Variable(initial-value, …)`
  - **e.g.** `tf.Variable(tf.zeros(shape=(2,2)))`
- Placeholder
  - `tf.placeholder(dtype, shape, ...)`
  - **e.g.** `tf.placeholder(tf.float32, shape=(10, 10))`

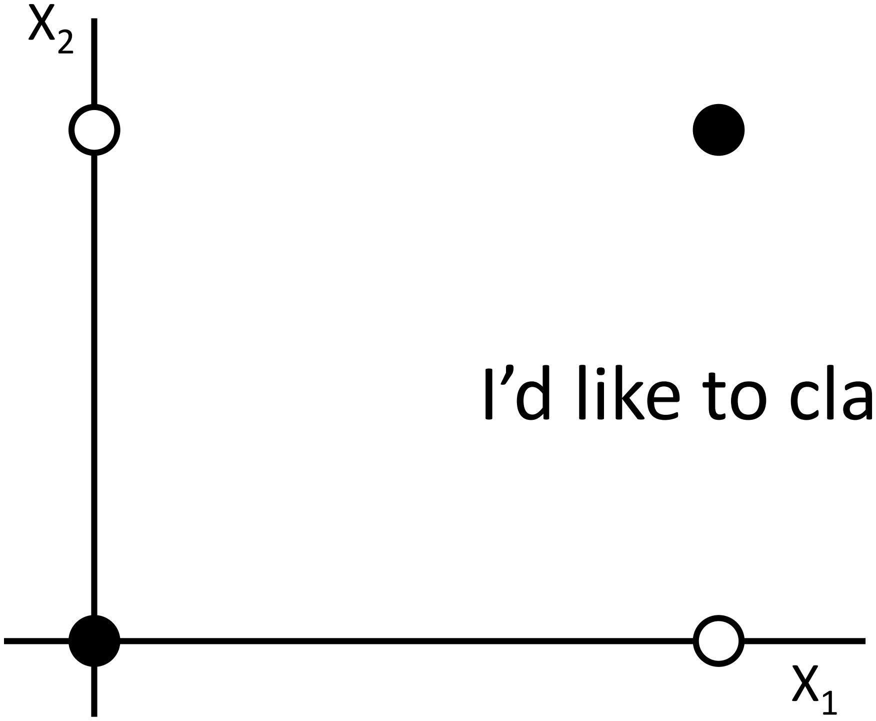
Softmax Regression

- Multinomial logistic regression



Neural Network

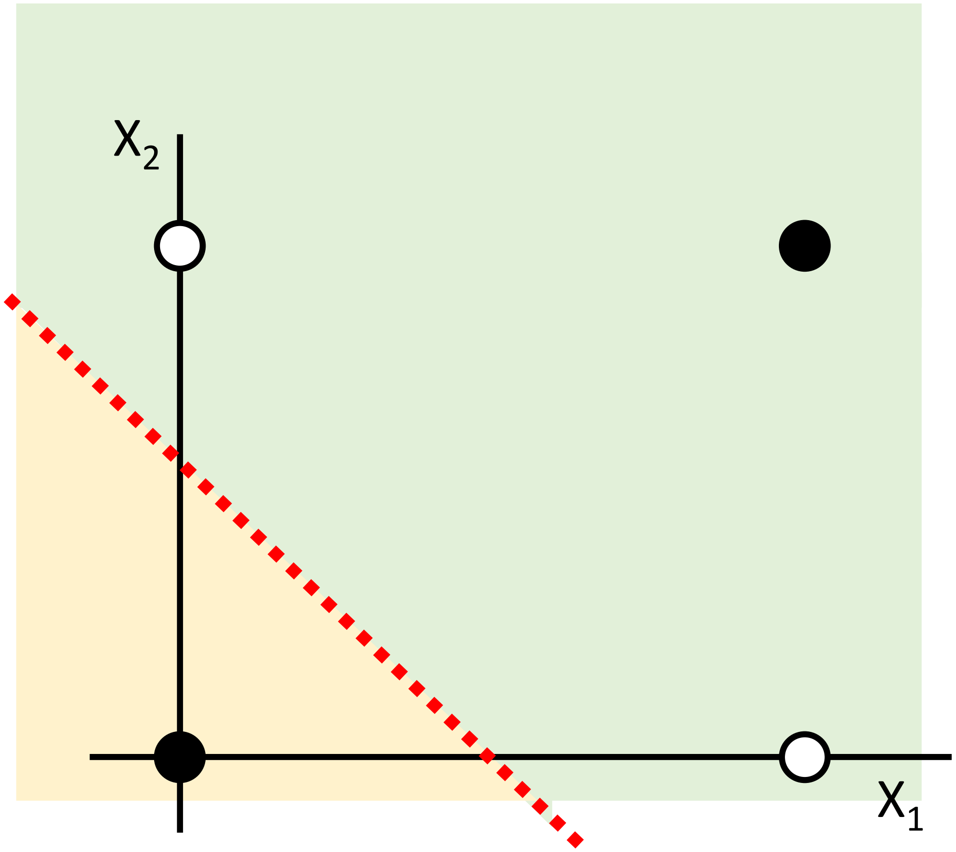
- XOR Problem

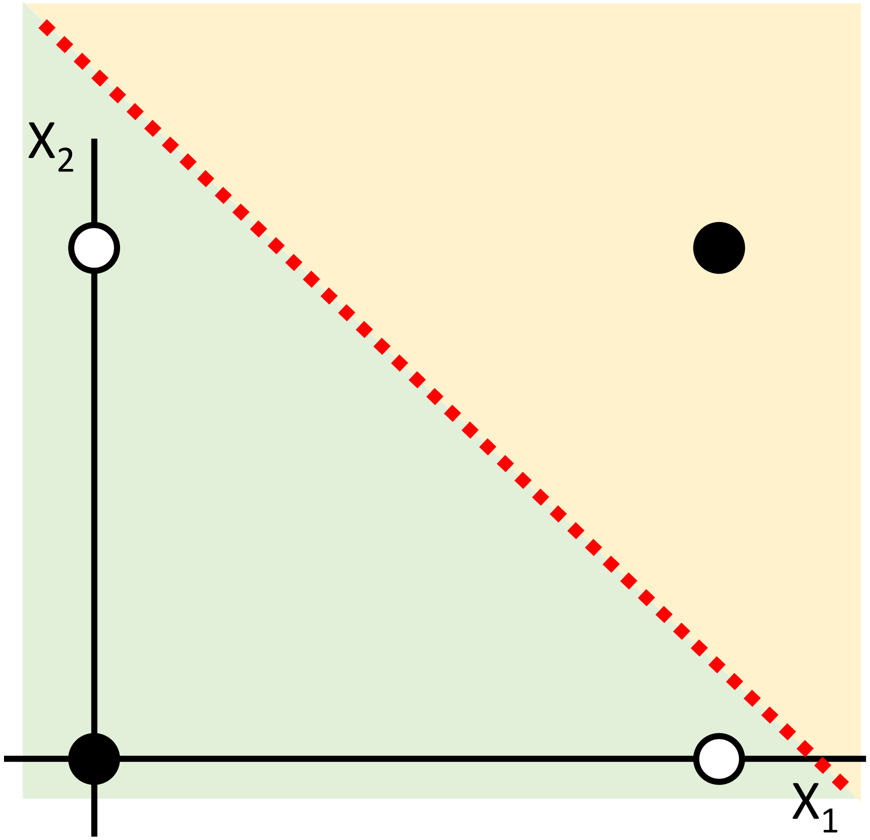


I’d like to classify 0(●) and 1(○).

Neural Network

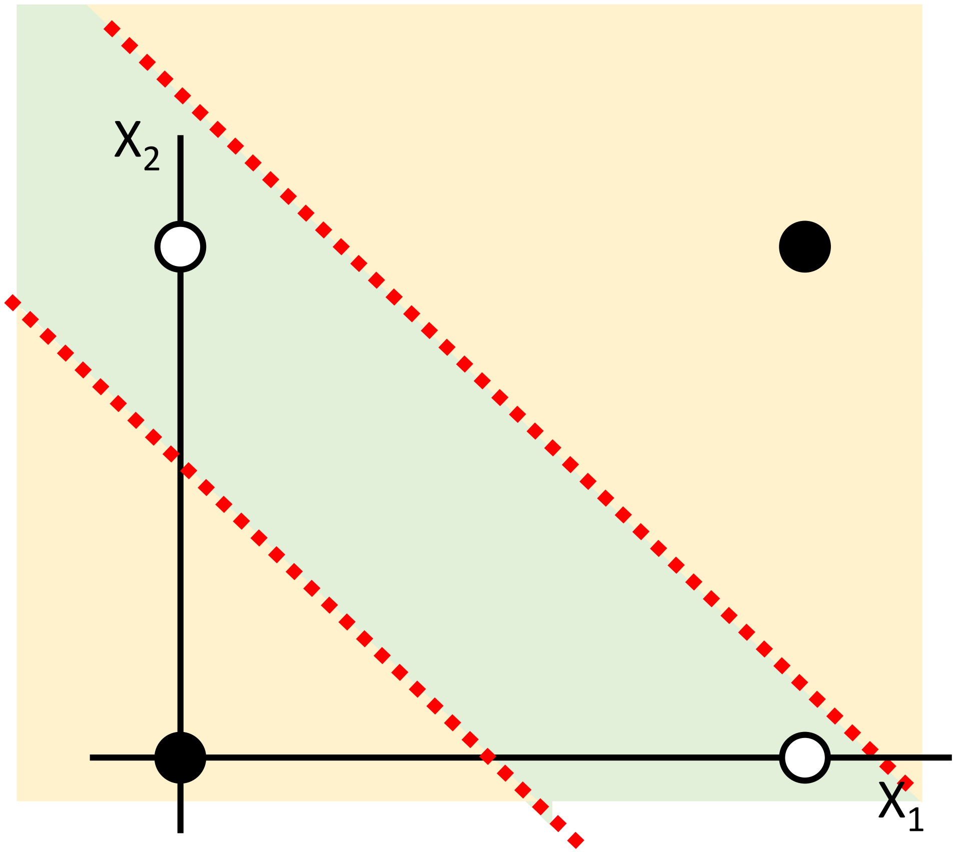
- If only with linear classifier (e.g. logistic regression model), …





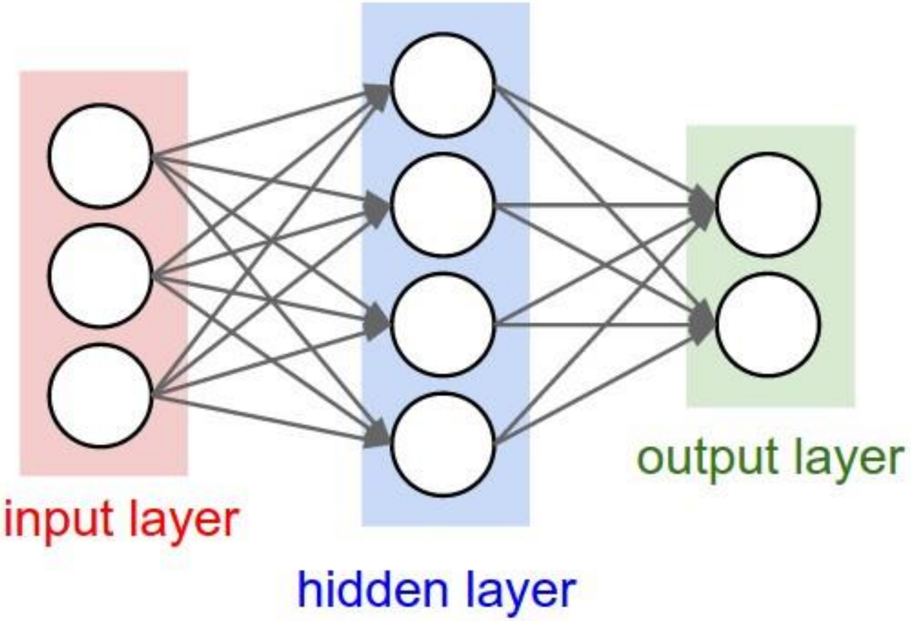
Neural Network

- But if we combine them, …



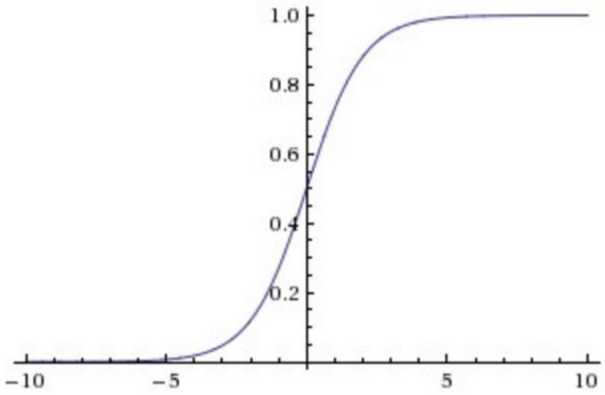
Neural Network

- Put hidden layer between input and output layer!
- Consists of fully-connected layers in which neurons between two adjacent layers are fully pairwise connected.
- Neurons within a single layer share no connections.

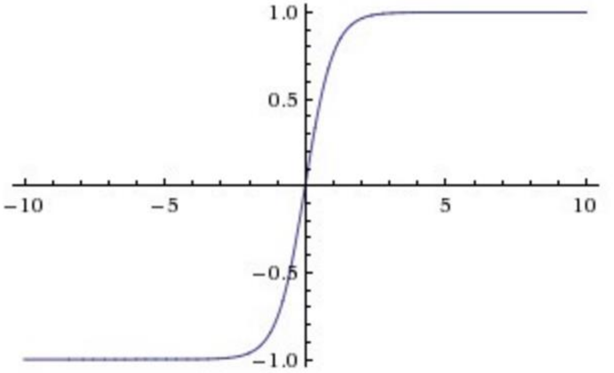


Neural Network

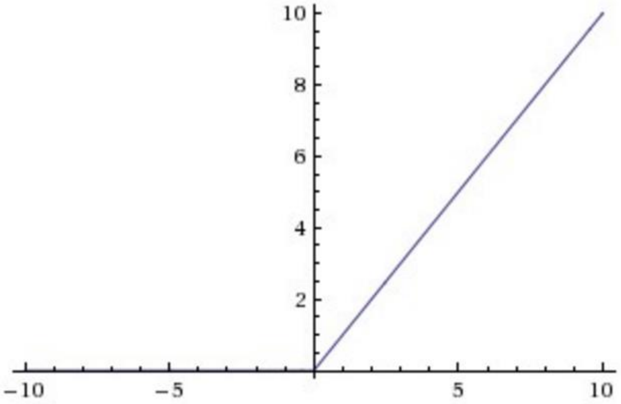
- Activation Functions



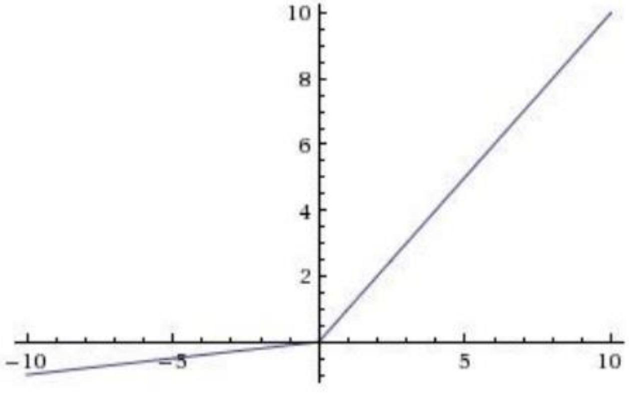
Sigmoid



tanh



ReLU

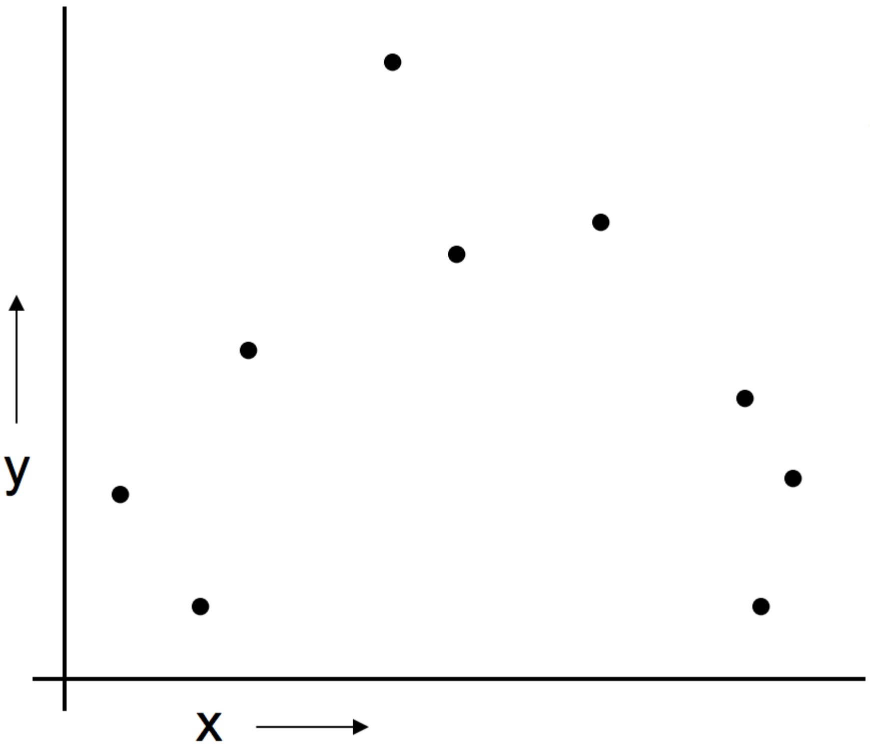


Leaky ReLU

Regularization

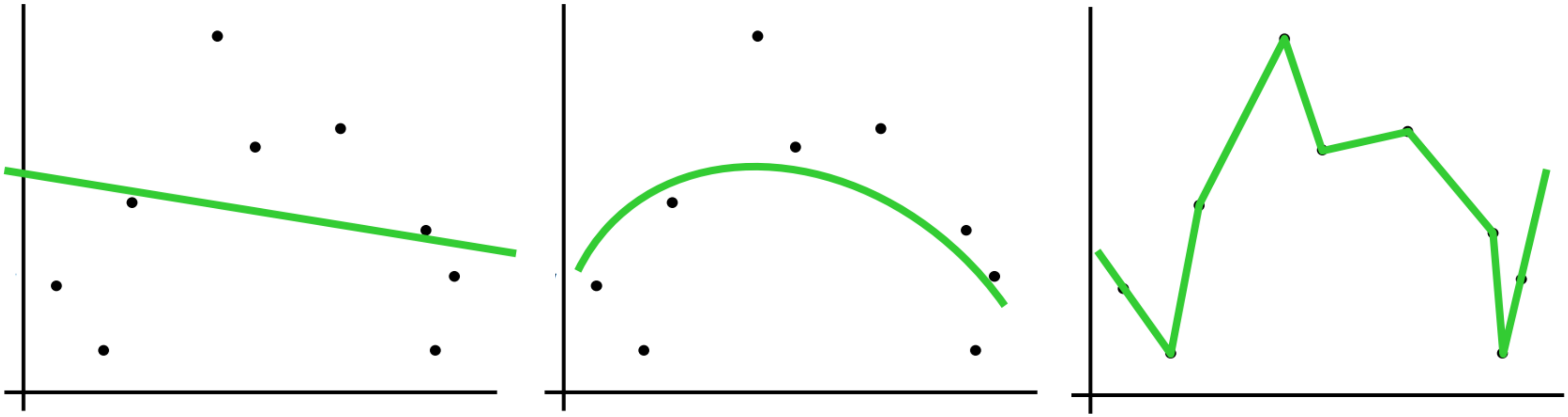
Regularization

- Let’s make the model which explains the data below.
- $y = f(x) + \epsilon$



Regularization

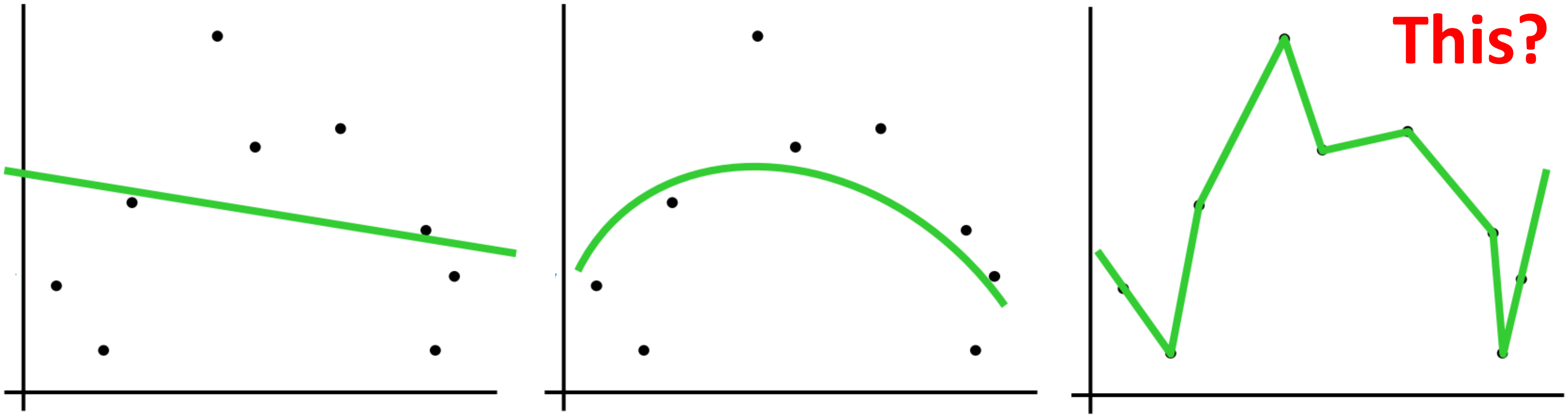
- Three candidates



Need to choose one of them… Which one is the best?

Regularization

- Three candidates

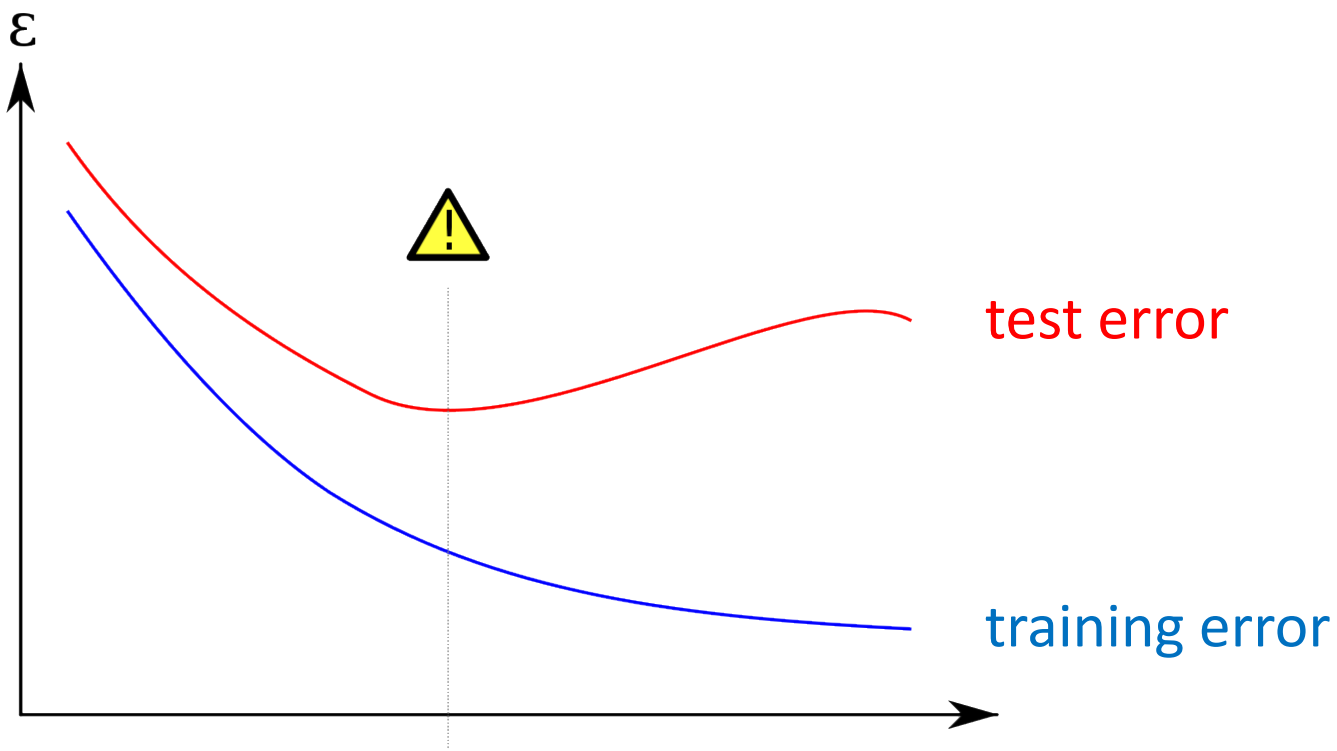


This?

Need to choose one of them... Which one is the best?

Regularization

- Overfitting problem



Regularization

- Several methods to avoid overfitting problem
  - L2 regularization
  - Dropout
  - …

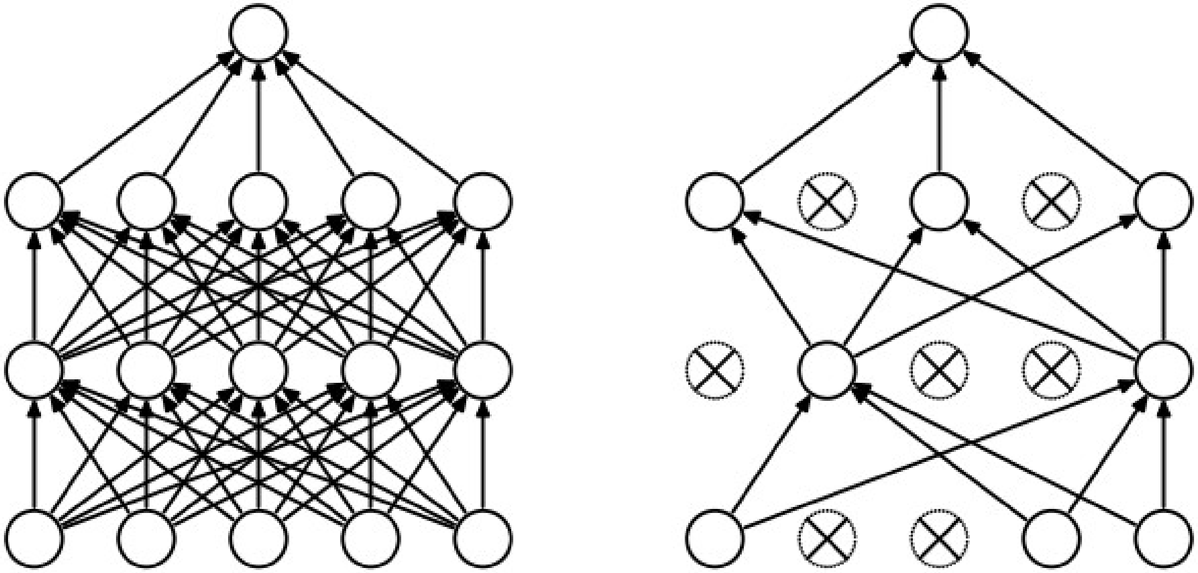
Regularization

- L2 regularization
  - Add L2 penalty (λw^2) to cost function
  - `tf.nn.l2_loss(t, …)`

```
with graph.as_default():
  ...

  loss = tf.reduce_mean(
    tf.nn.softmax_cross_entropy_with_logits(logits,
    + l2_lambda * tf.nn.l2_loss(weights)

  ...
```

Regularization

- Dropout
  - Sampling a Neural Network within the full Neural Network, and only updating the parameters of the sampled network based on the input data

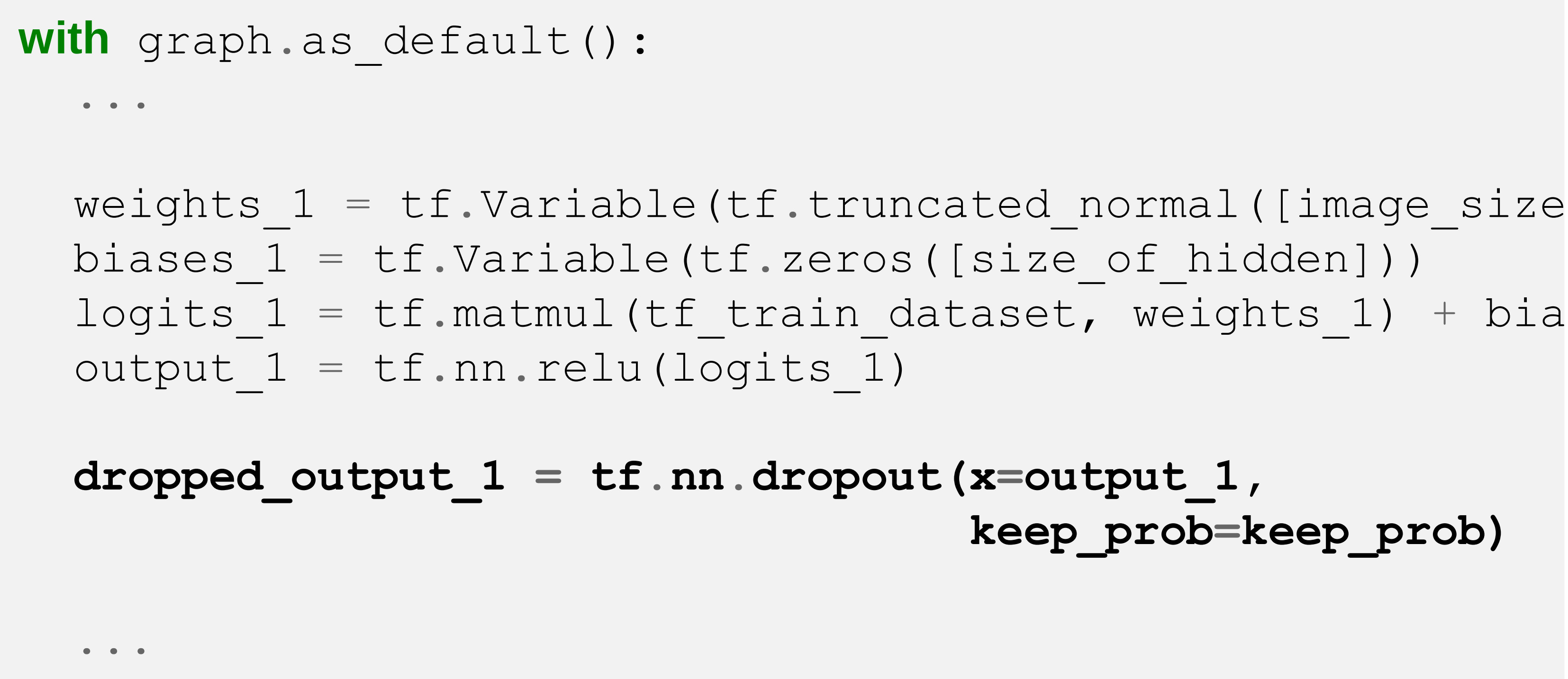


(a) Standard Neural Net

(b) After applying dropout.

Regularization

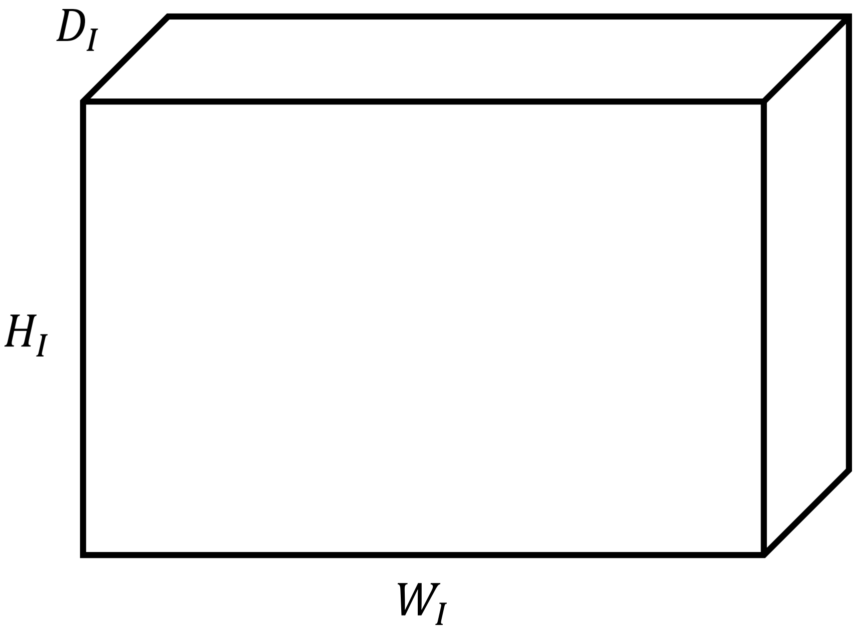
- Dropout
    - `tf.nn.dropout(x, keep_prob, …)`
    - `keep_prob`: The probability that each element is kept



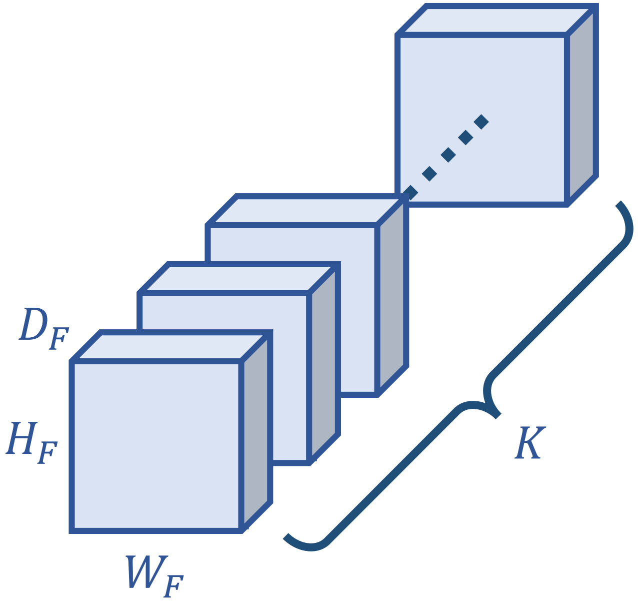
```
with graph.as_default():
  ...

  weights_1 = tf.Variable(tf.truncated_normal([image_size
  biases_1 = tf.Variable(tf.zeros([size_of_hidden]))
  logits_1 = tf.matmul(tf_train_dataset, weights_1) + bia
  output_1 = tf.nn.relu(logits_1)

  dropped_output_1 = tf.nn.dropout(x=output_1,
                                   keep_prob=keep_prob)


  ...
```

Convolutional Neural Network

Convolution

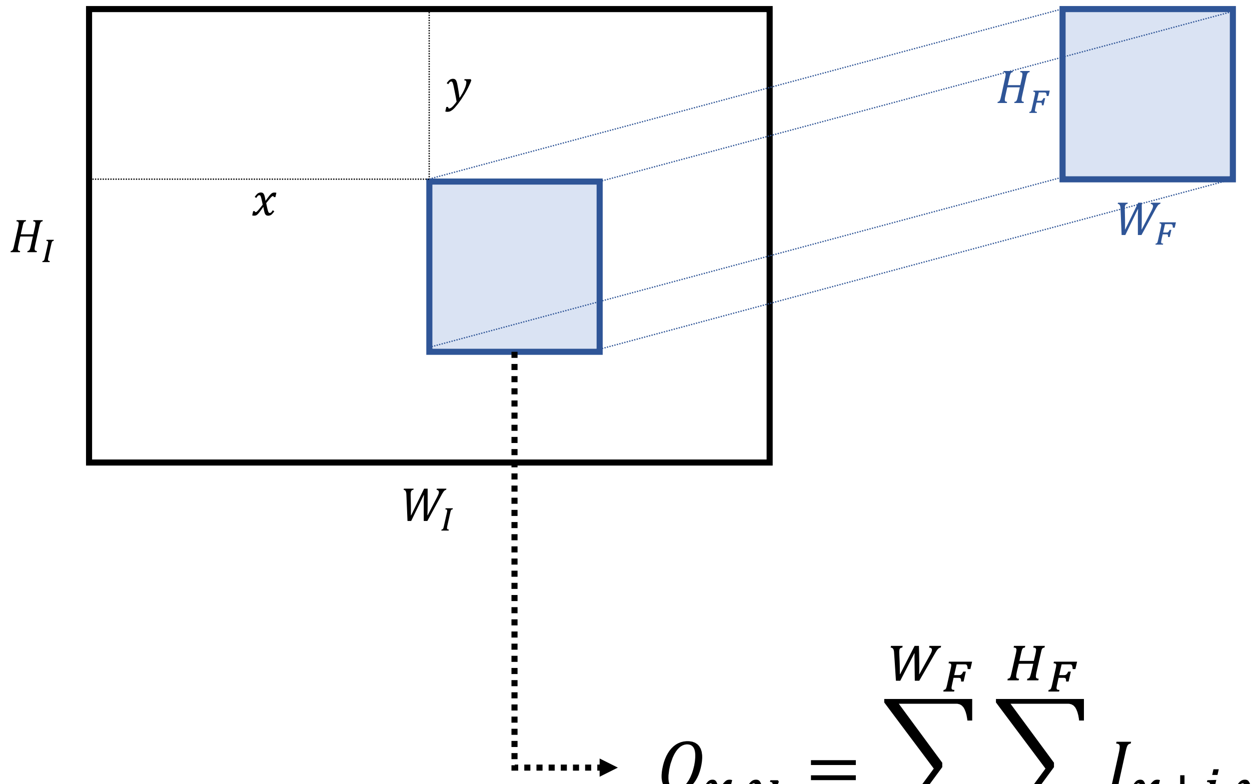


Input I: $[W_I, H_I, D_I] \in \mathbb{R}^3$



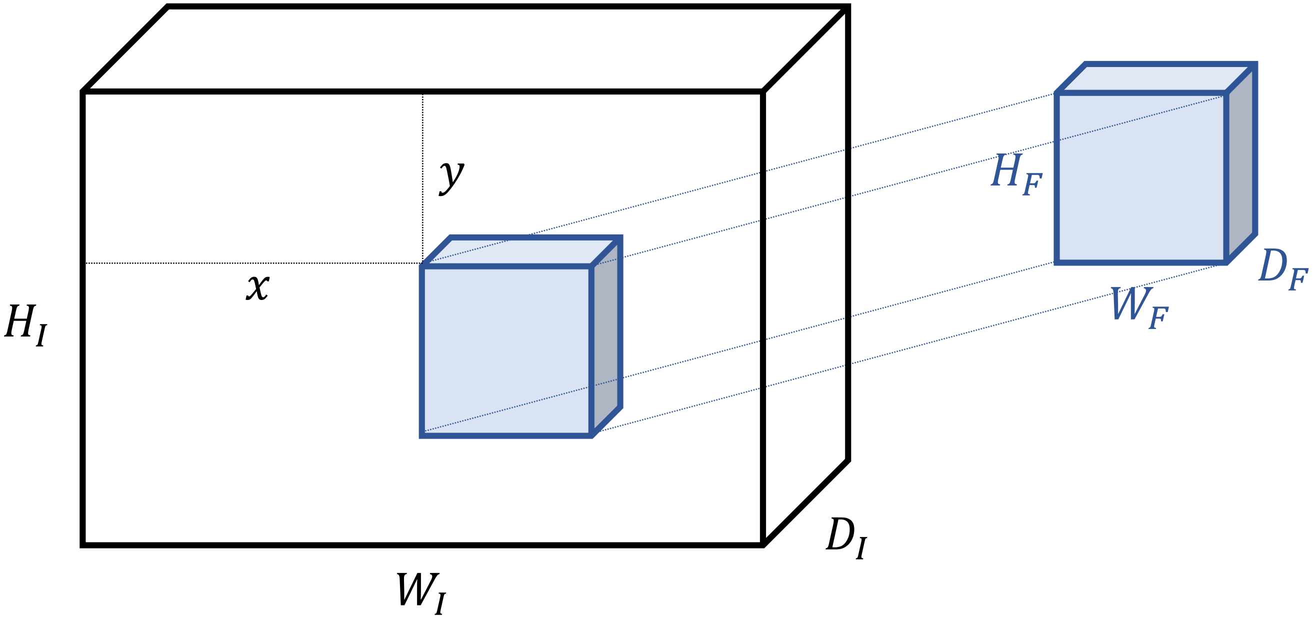
Filter F: $[W_F, H_F, D_F, K] \in \mathbb{R}^4$

Convolution



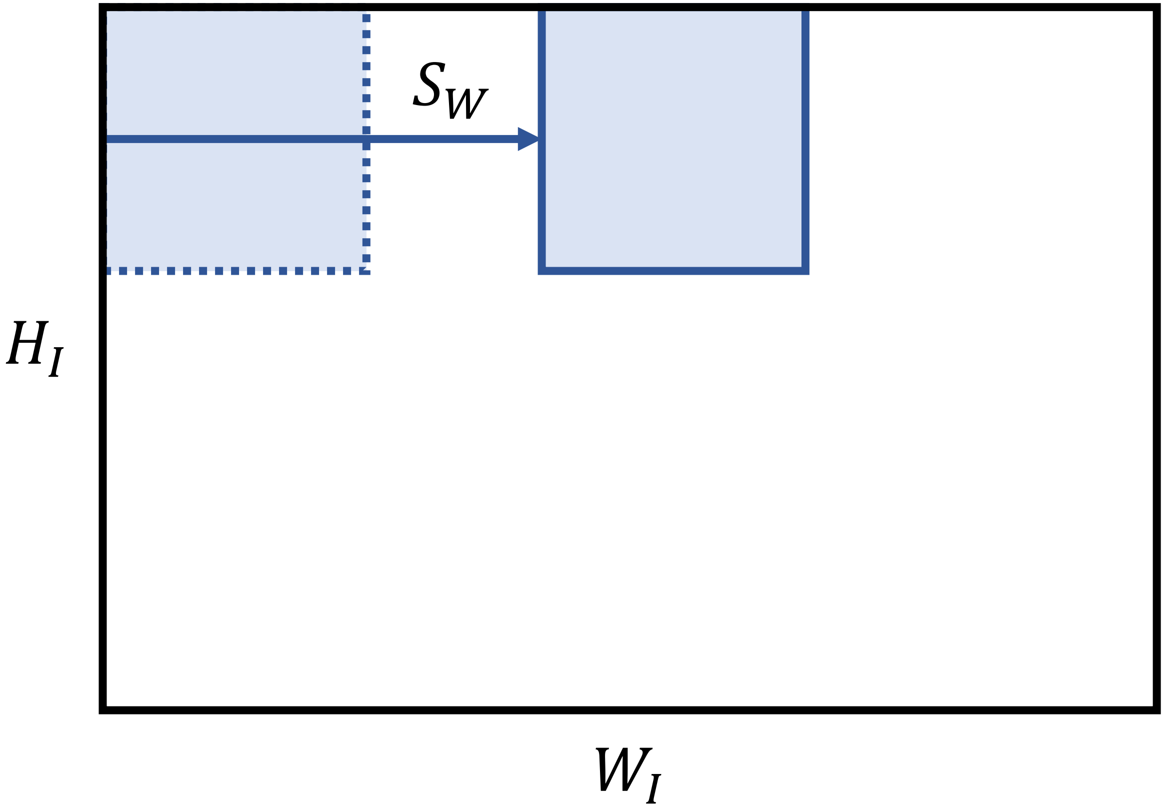
$$O_{x,y} = \sum_{i}^{W_F} \sum_{j}^{H_F} I_{x+i,y+j} \times F_{i,j}$$

Convolution



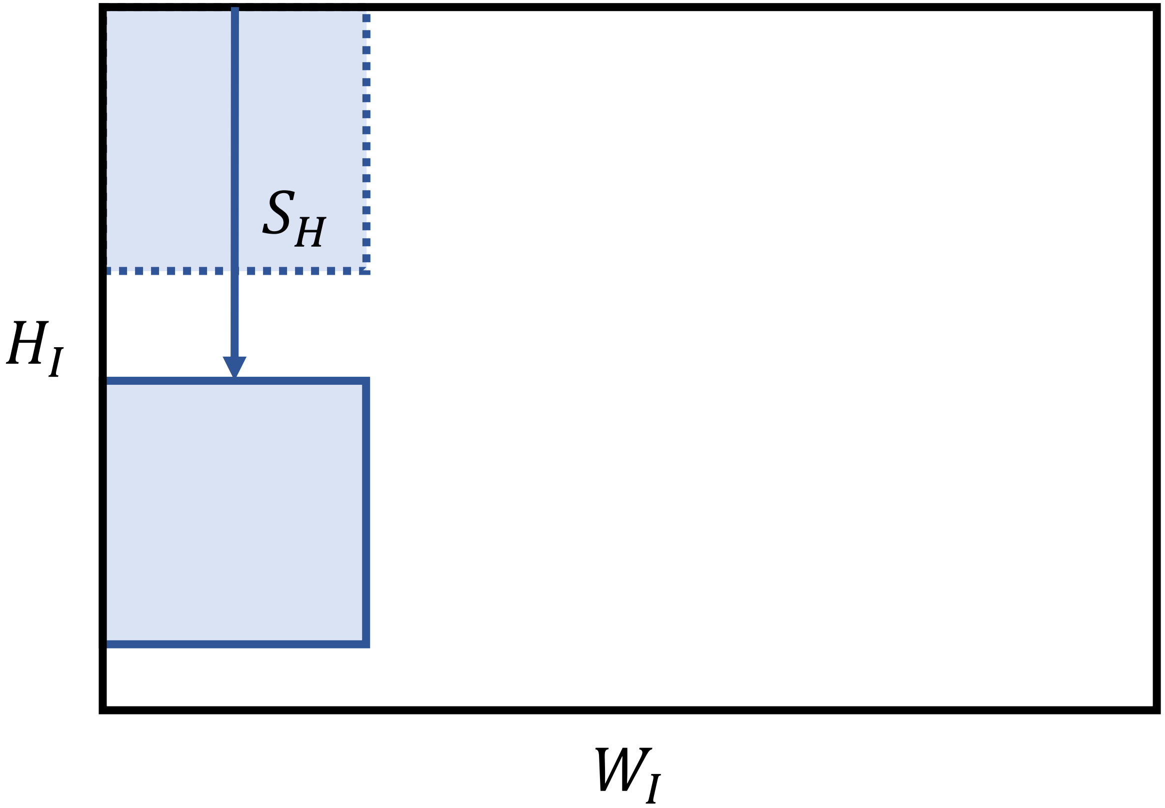
Convolution

- Stride



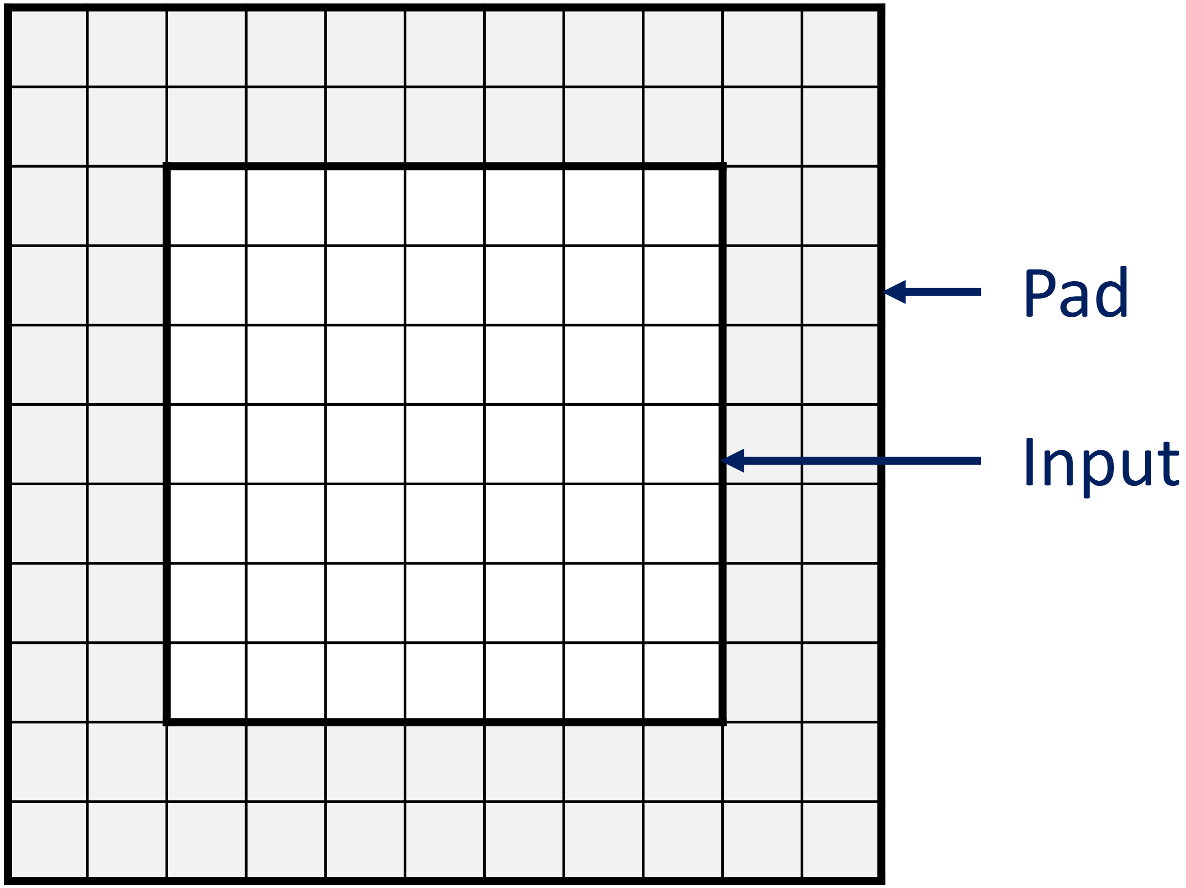
Convolution

- Stride



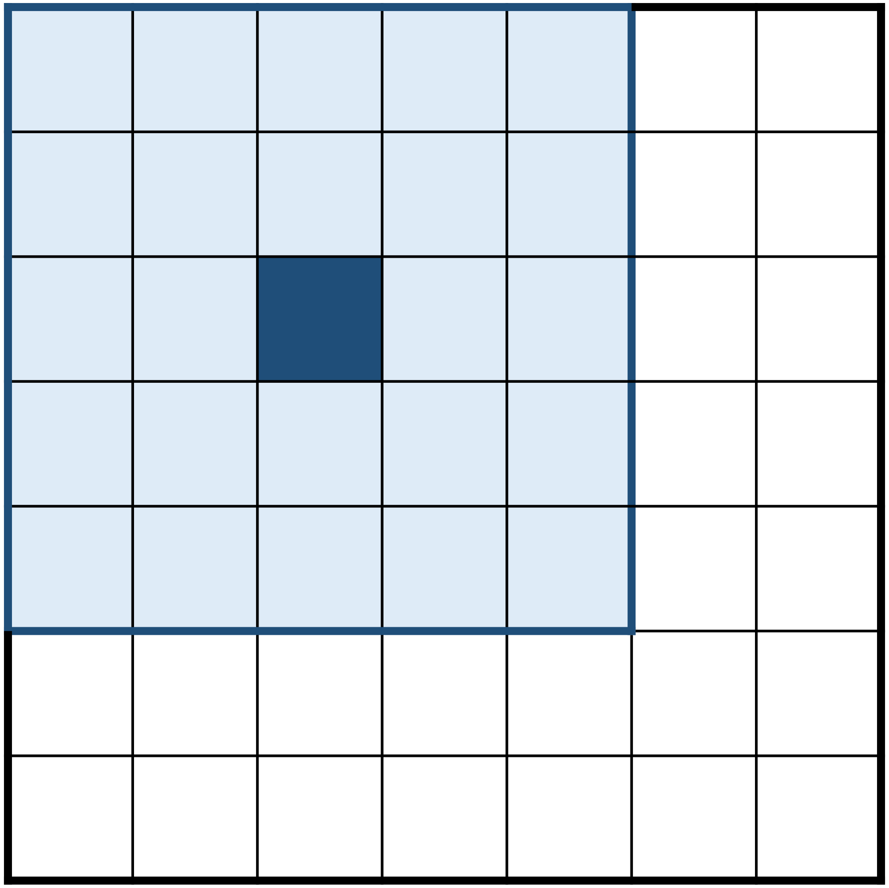
Convolution

- Padding



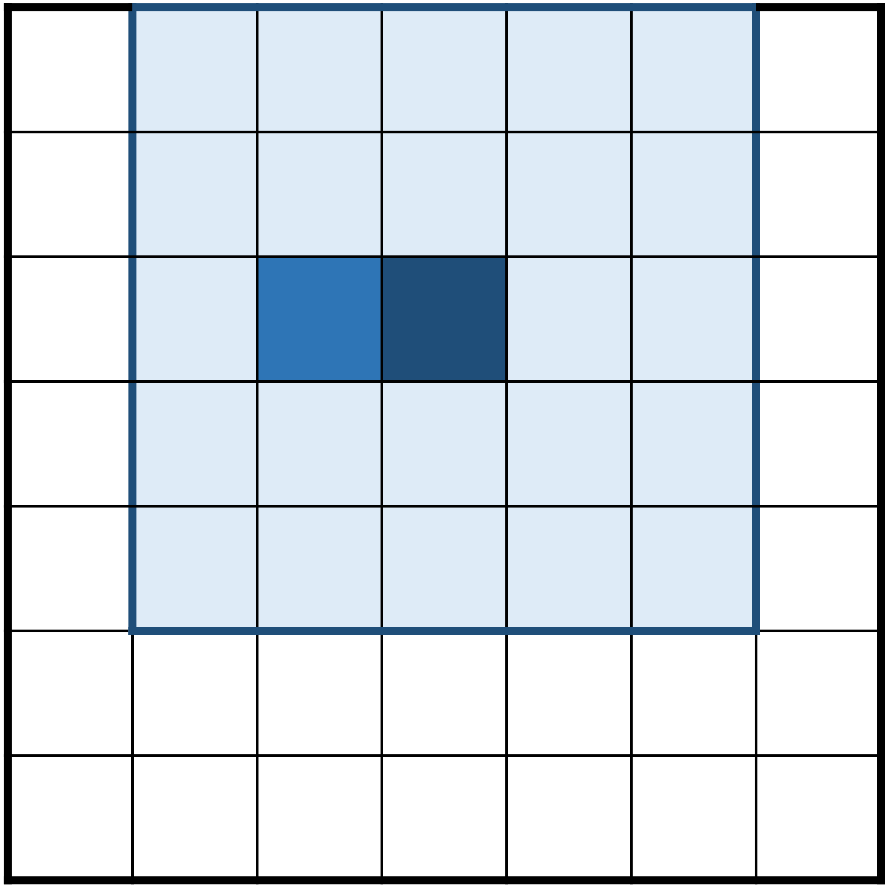
Convolution

- Padding



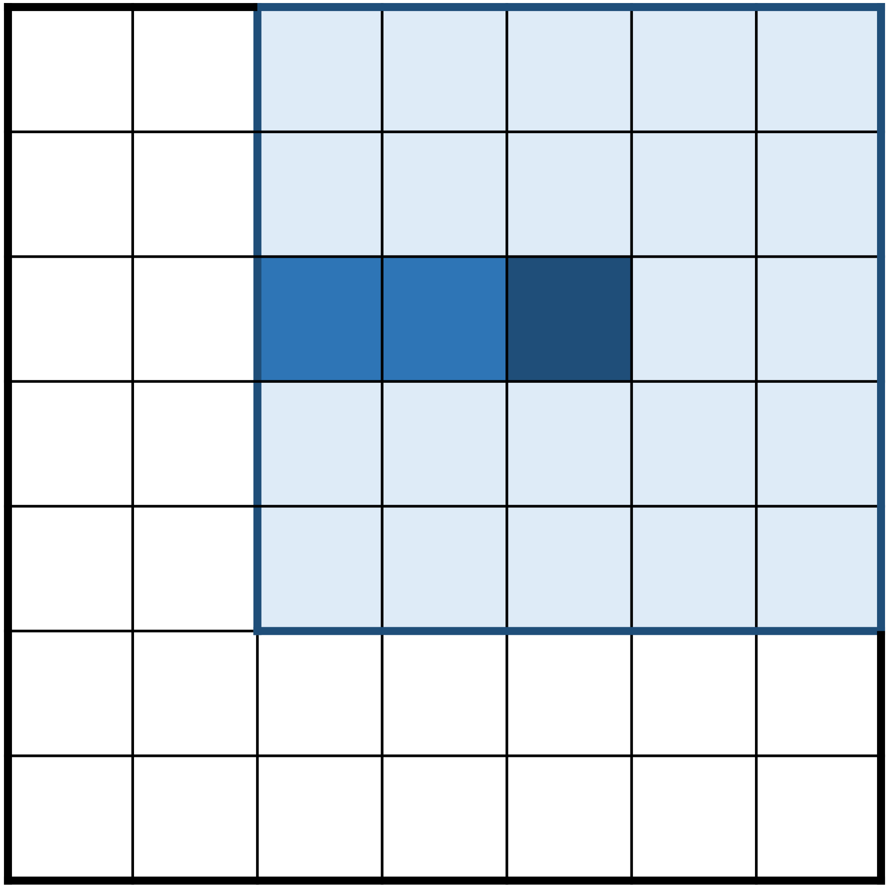
Convolution

- Padding



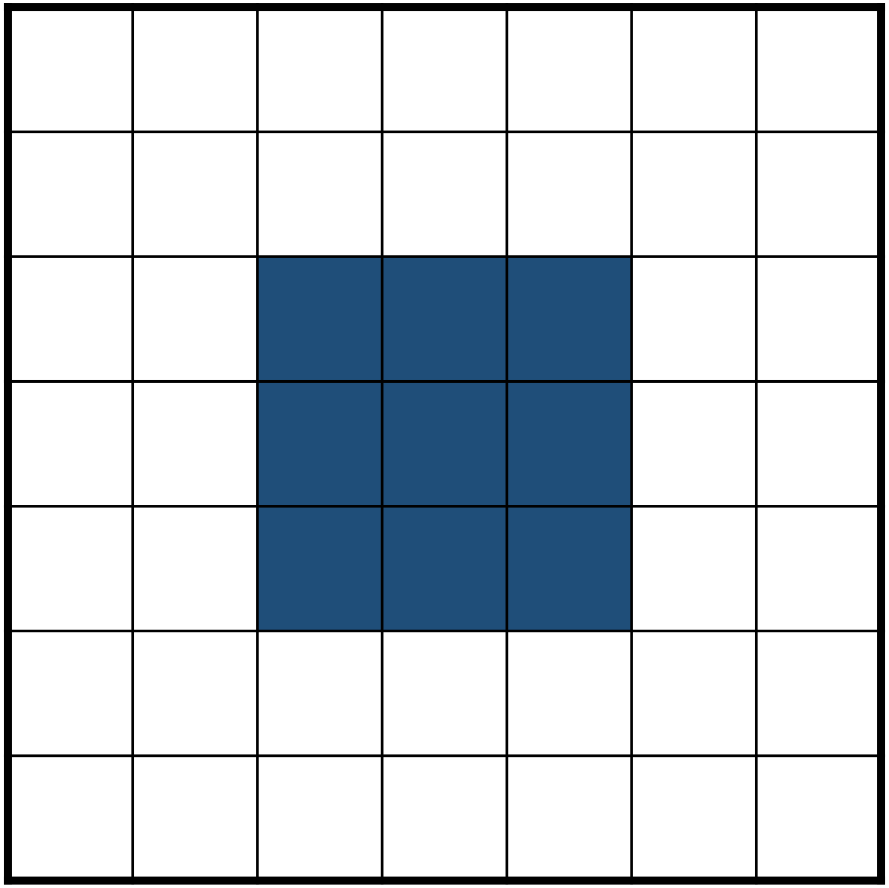
Convolution

- Padding



Convolution

- Padding



Convolution

- Padding

0	0	0	0
0	0	0	0
0	1	4	0
0	2	5	0
0	3	6	0
0	0	0	0
0	0	0	0

Zero

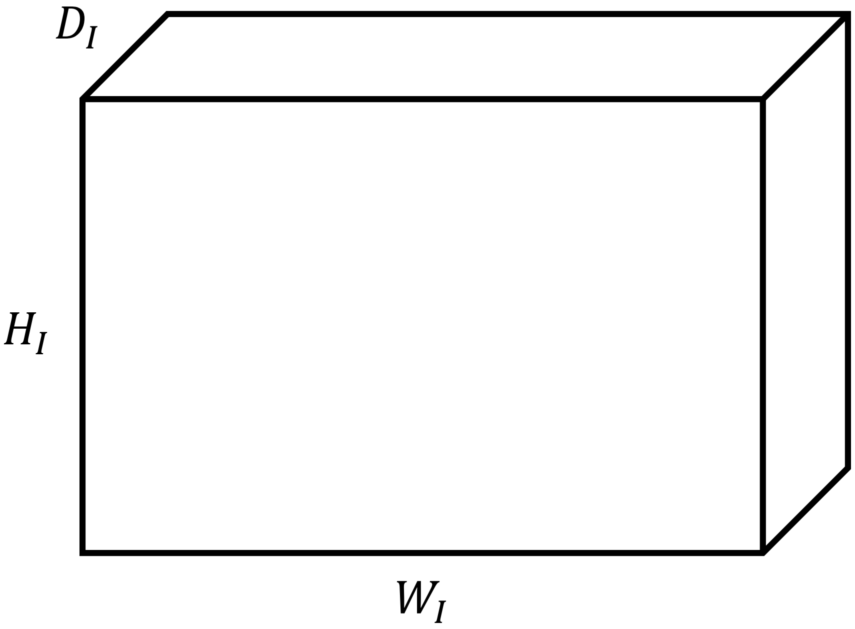
6	3	6	3
5	2	5	2
4	1	4	1
5	2	5	2
6	3	6	3
5	2	5	2
4	1	4	1

Reflect

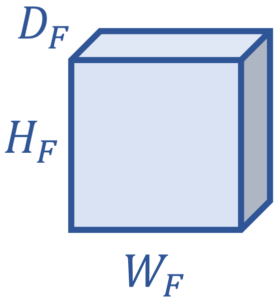
2	2	5	5
1	1	4	4
1	1	4	4
2	2	5	5
3	3	6	6
3	3	6	6
2	2	5	5

Symmetric

Pooling

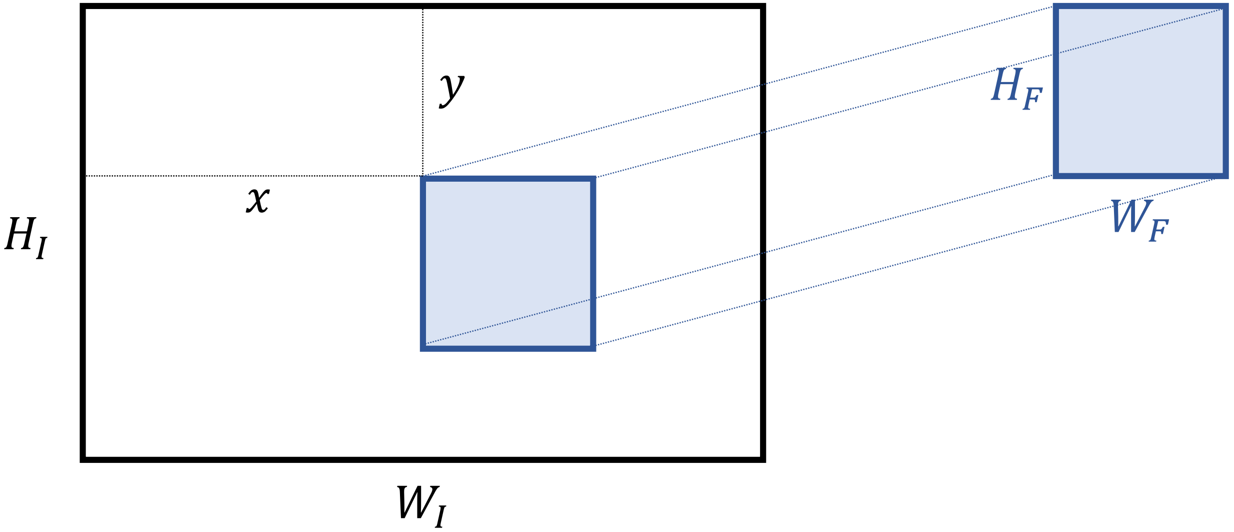


Input I: $[W_I, H_I, D_I] \in \mathbb{R}^3$



Filter F: $[W_F, H_F, D_F] \in \mathbb{R}^3$

Pooling



Max: $$O_{x,y} = \max_{i,j} I_{x+i,y+j}$$

Average: $$O_{x,y} = \frac{1}{W_F \times H_F} \sum_{i}^{W_F} \sum_{j}^{H_F} I_{x+i,y+j}$$

Reference

- TensorFlow official webpage

  https://www.tensorflow.org/

- Stanford CS class - CS231n

  http://cs231n.github.io/

- Udacity - Deep learning course

  https://www.udacity.com/course/deep-learning--ud730