

**Predicting Financial Time Series using Deep Learning**

# Module2. Convolutional Neural Network

Jongho Kim

NICE Pricing & Information Inc.

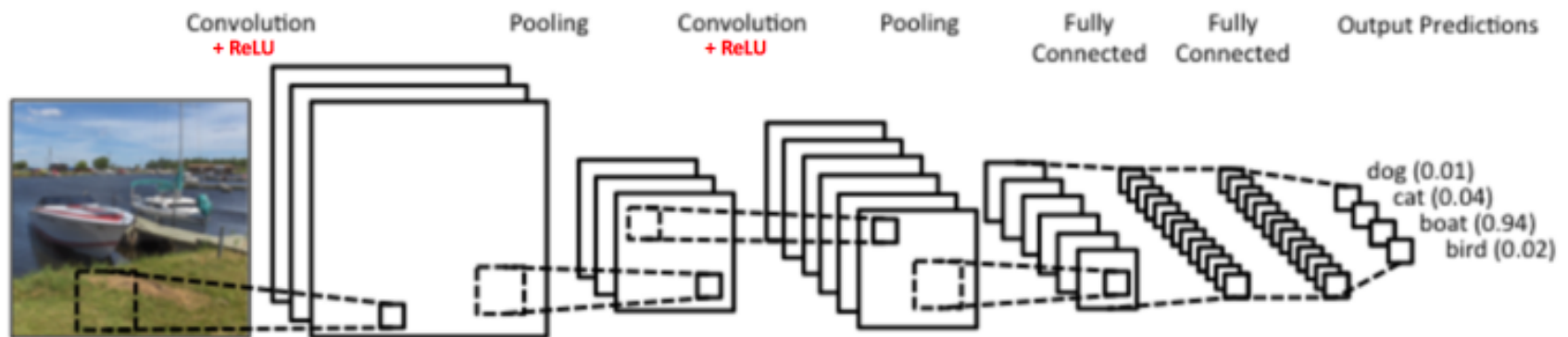
Fall, 2018

Note. This content mainly refers the summer session of KAIST organized by Jiyong Park(2018)

# Convolutional Neural Network

# Review: What is CNN?

- A typical example of CNN



- Inputs (2-dimension × channels) → [Convolution -> ReLu -> Pooling] → ...  
→ [Convolution -> ReLu -> Pooling] → Fully connected layer  
→ Output prediction (Multi-class classification)

# Review: What is CNN?

- Convolution

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image data

1	0	1
0	1	0
1	0	1

Convolution filter

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image

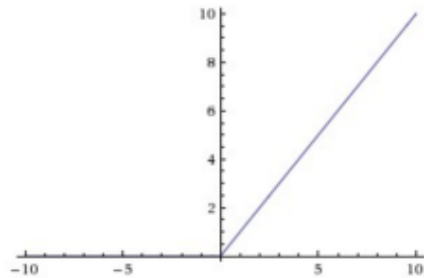
4		

Convolved  
Feature

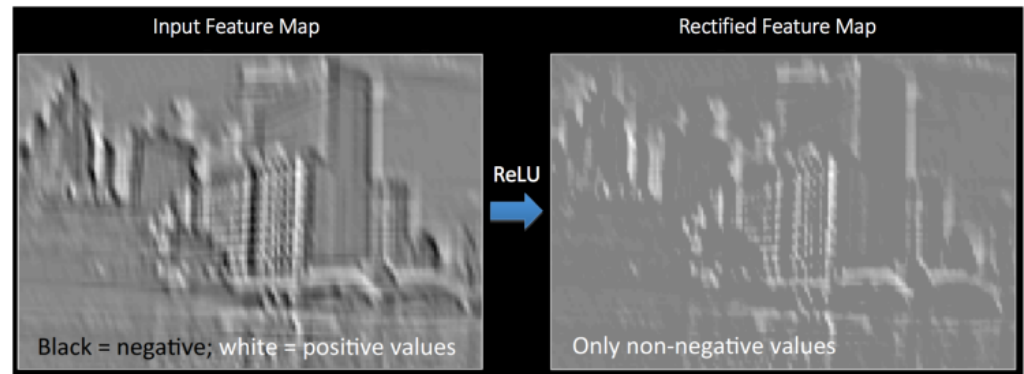


# Review: What is CNN?

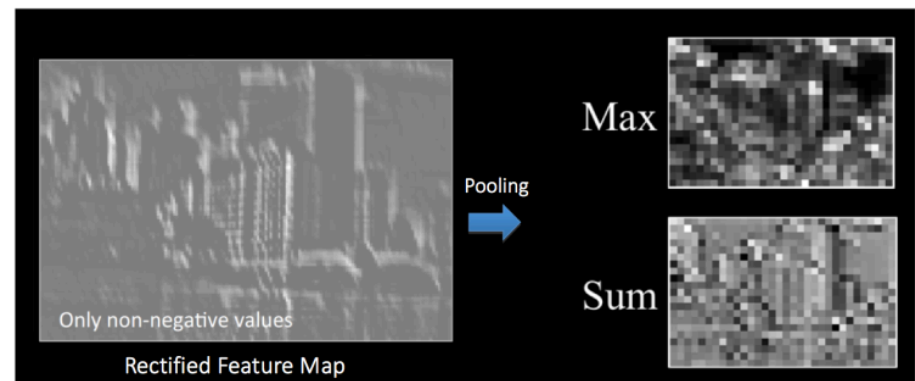
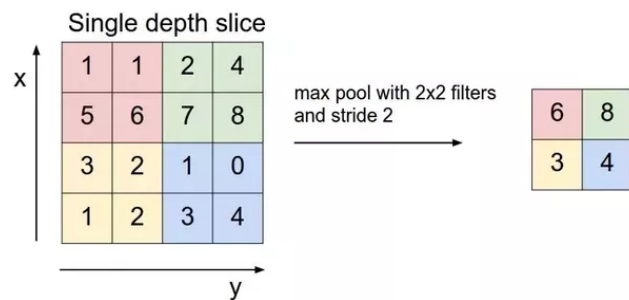
- ReLU



$$\text{Output} = \text{Max}(\text{zero}, \text{Input})$$

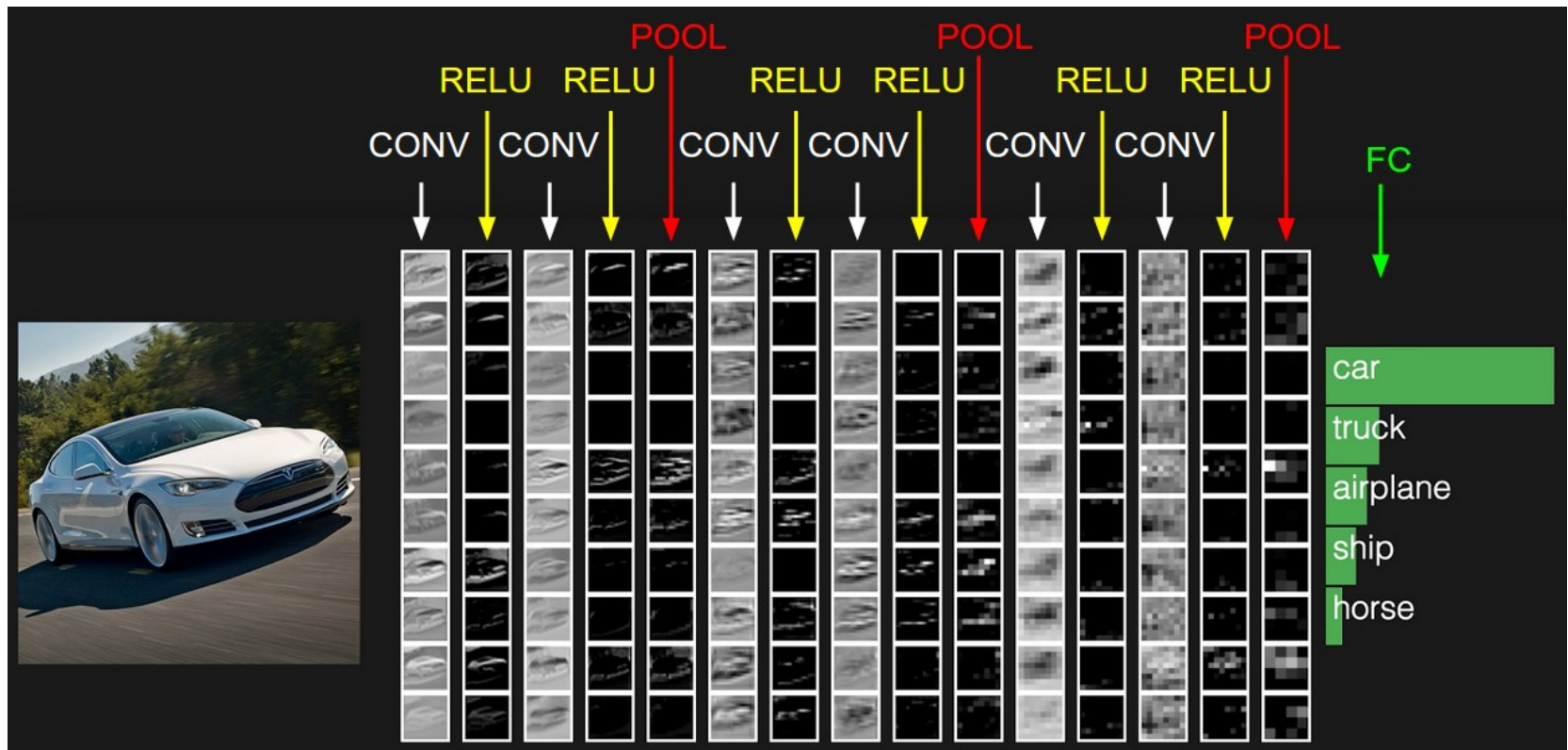


- Max pooling



# Review: What is CNN?

- Hierarchical feature representation (multiple convolution layers)



```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D

model = Sequential()

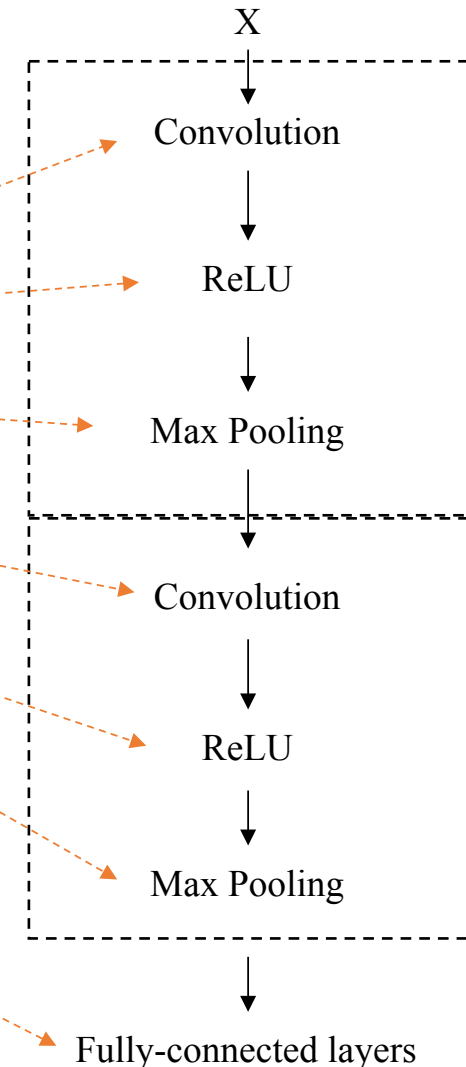
model.add(Conv2D(256, (3, 3), input_shape=X.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(256, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten()) # this converts our 3D feature maps to 1D feature vectors
model.add(Dense(64))
model.add(Dense(1))
model.add(Activation('sigmoid'))

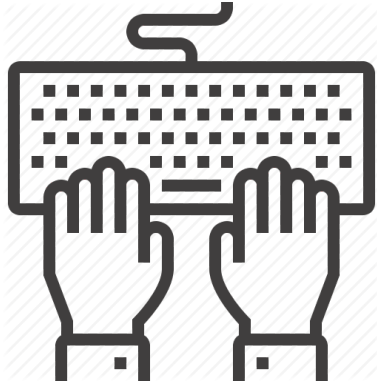
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model.fit(X_train, y_train, batch_size=32, epochs=30, validation_split=0.3)
```



# Hands-on-Labs Convolutional Neural Network





# Dogs & Cats Classifier using CNN

*Lab1\_Convolutional\_Neural\_Network.ipynb*

<https://colab.research.google.com/drive/1qKSNVWgBkqi7S4KL-pTnbBawmv7hT2fP>

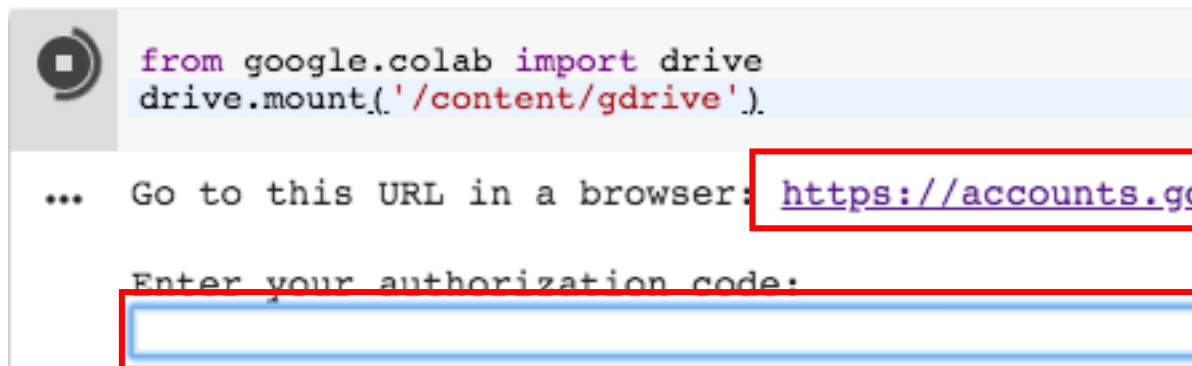
- First Step: Upload your data on Google Drive

<http://drive.google.com>

Data Download at:

<https://drive.google.com/open?id=1Dje13qjGZwtaVhhjp-ZJVzeyzJrG6oc1>

- Second Step: Enter Authentication Code on Google Colab



The screenshot shows a Google Colab notebook cell with the following code:

```
from google.colab import drive
drive.mount('/content/gdrive')
```

Below the code, there is a message: "... Go to this URL in a browser: <https://accounts.google.com/ConnectApp?app=google-colab>". The URL is highlighted with a red box.

Below the URL, there is a prompt: "Enter your authorization code:". Below this prompt is a text input field, which is also highlighted with a red box.

• Go to Url

• Type Code

- `!ls "/content/gdrive/My Drive/"`

```
[ ] !ls "/content/gdrive/My Drive/Lecture/StudyPie/Data"  
[> crypto_data.zip kagglecatsanddogs_3367a.zip PetImages.zip
```

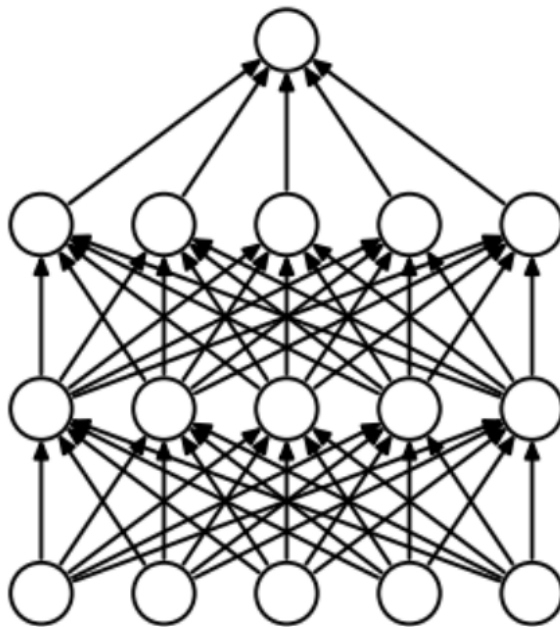
- Set data path on the code according to the your local environment
  - In my case:

```
[3] DATA_PATH = "/content/gdrive/My Drive/Lecture/StudyPie/Data/"
```

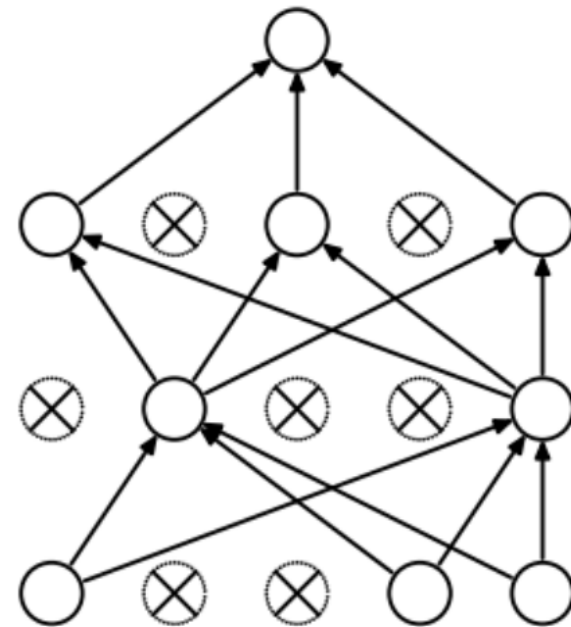
# Improving Performance on CNN

# Improving Performance (1) Dropout

- Dropout avoids overfitting of neural networks



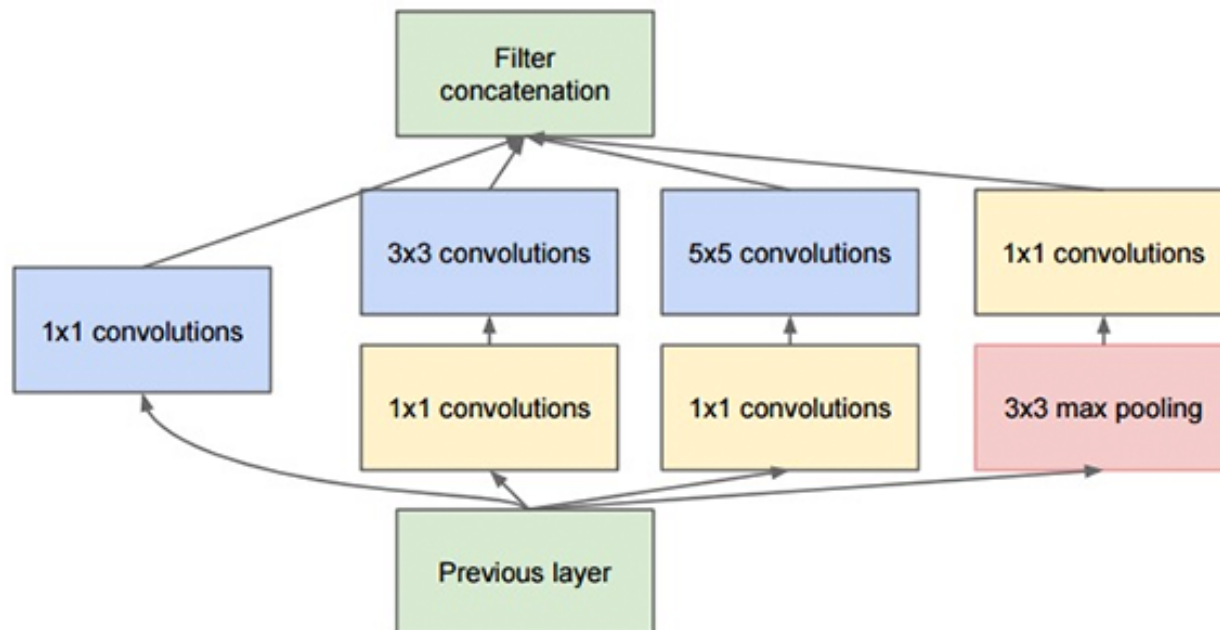
(a) Standard Neural Net



(b) After applying dropout.

# Improving Performance (2) Inception

- CNN can have filters with multiple sizes operating on the same level.
  - CNN with inception tends to be more efficient and has better performance (Szegedy et al. 2015)



Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going Deeper with Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

# 1D Convolution

- How can we apply convolution on time series data?
- 1D Convolution of discrete time signals
- Example

- Kernel:

-1	1
----	---

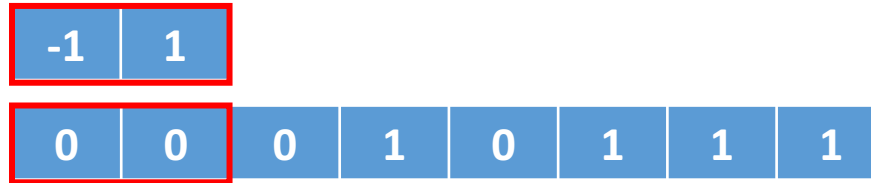
- Time series data:

0	0	0	1	0	1	1	1
---	---	---	---	---	---	---	---

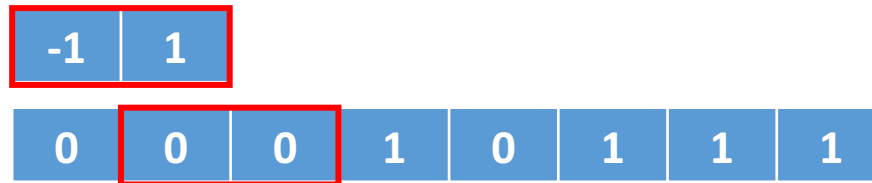


- Example

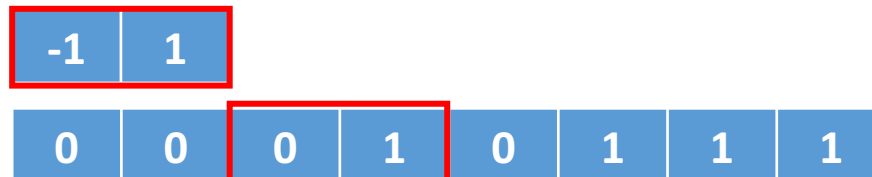
- $N = 0: -1 * 0 + 1 * 0 = 0$



- $N = 1: -1 * 0 + 1 * 0 = 0$



- $N = 2: -1 * 0 + 1 * 1 = 1$



Thank you ☺

Contact Info: [quantic.jh@gmail.com](mailto:quantic.jh@gmail.com)

# References

---

- Jiyong Park (2018), KAIST Summer Session, Retrieved from <https://sites.google.com/view/kaist-mis-session2018/overview?authuser=0>