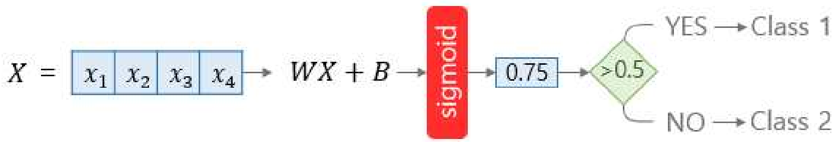
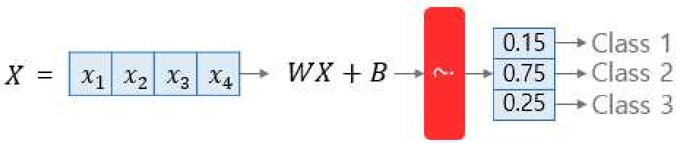
딥러닝 활용 TSR(Traffic Sign Recognition)

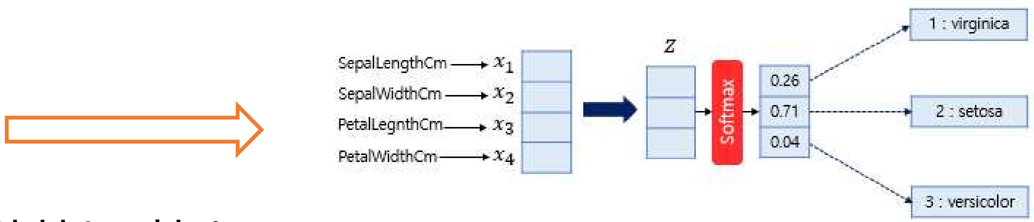
Base Knowledge: Softmax Regression

- Classifier(분류기)
- Multinomial Classification
  - N개 이상의 선택지 중에서 1개를 고르는 문제 (vs. Binary Classification: 타이타닉, etc.)
  - MNIST: Handwrite된 0~9 숫자 중에 하나를 고르는 문제
  - Logistic vs. Softmax



가설 : $H(X) = sigmoid(WX + B)$





가설 : $H(X) = softmax(WX + B)$ 확률 값으로 출력하는 함수

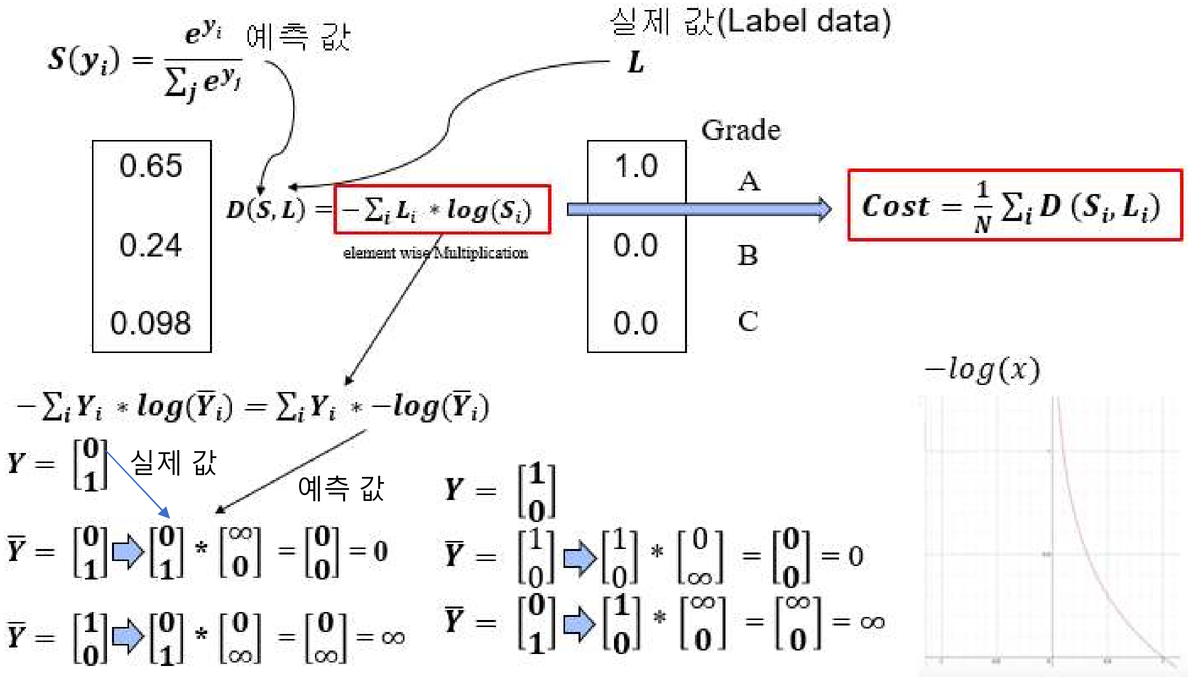
  - **TSR: 여러가지 Traffic Sign 중에 가장 확률이 높은 하나를 선택하는 Application**

Base Knowledge: Softmax Regression

- Multinomial Classification
  - 가설함수(Softmax) 정의: 확률 값으로 출력 (0~1사이의 값)

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}} \quad \text{(yi = wx + b)}$$

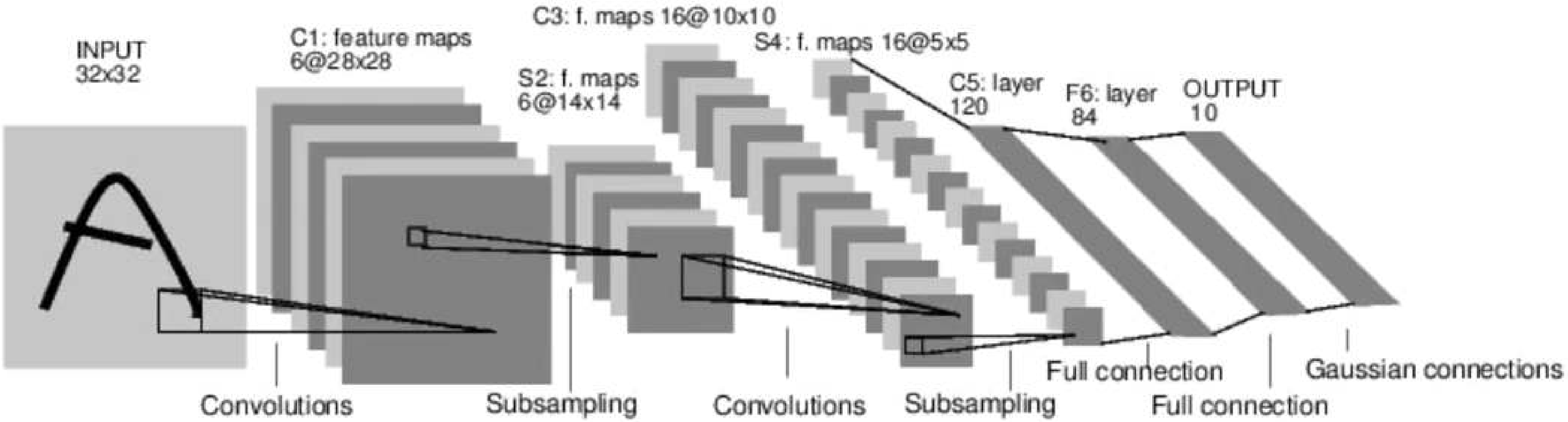
  - **Cost Function: Cross-Entropy**



Base Knowledge: CNN (Convolution Neural Network)

- CNN Case Study
  - LeNet-5

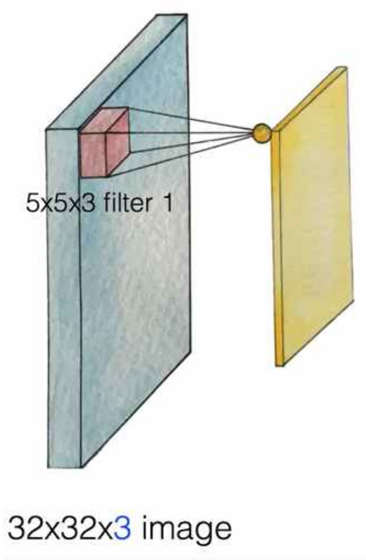
[LeCun et al., 1998]

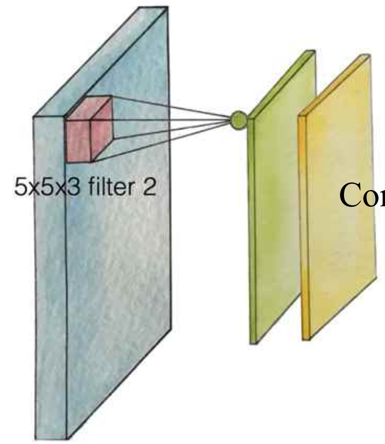


Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-CONV-FC]

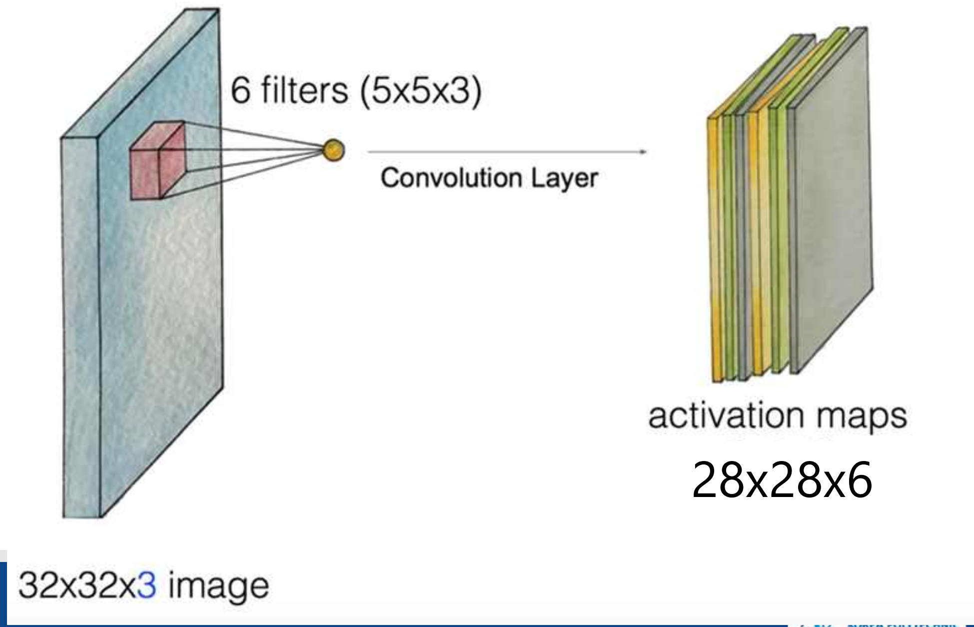
Base Knowledge: CNN (Convolution Neural Network)

- CNN
  - Conv Layer

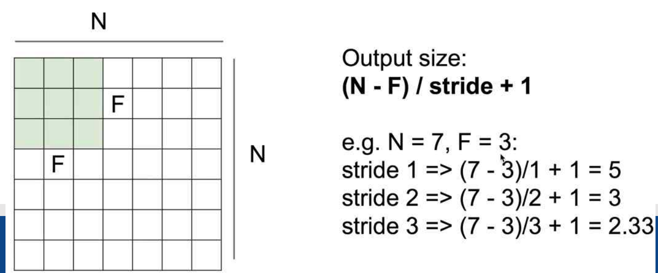




Come from filter 1

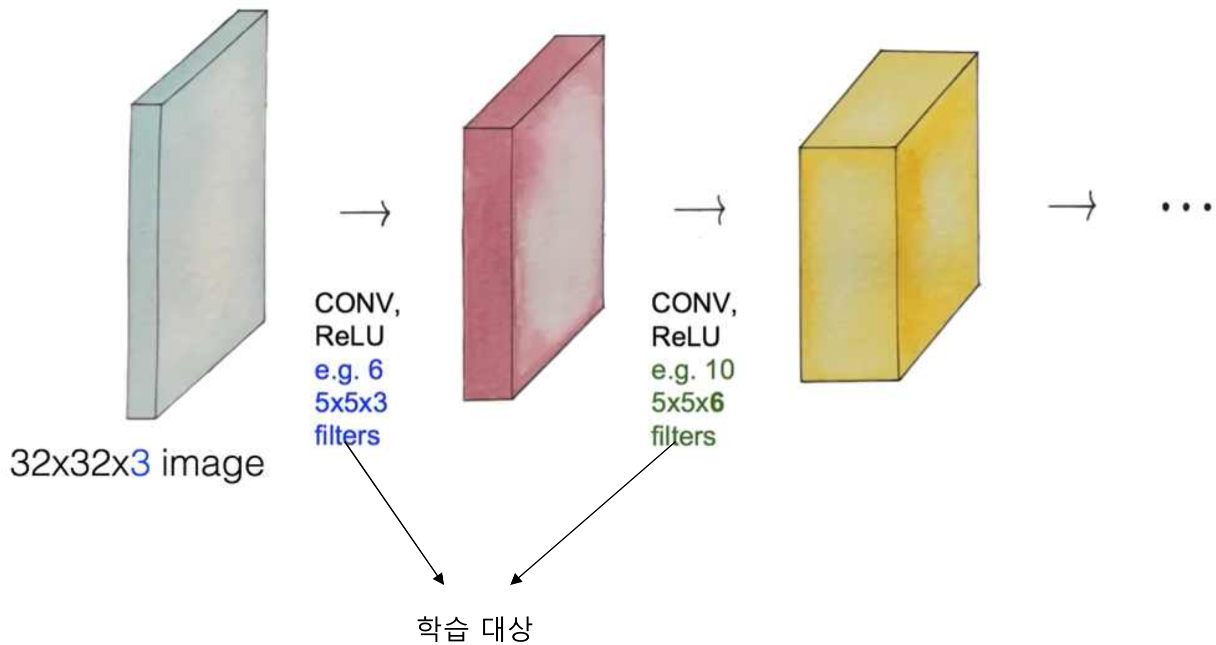


28x28x6



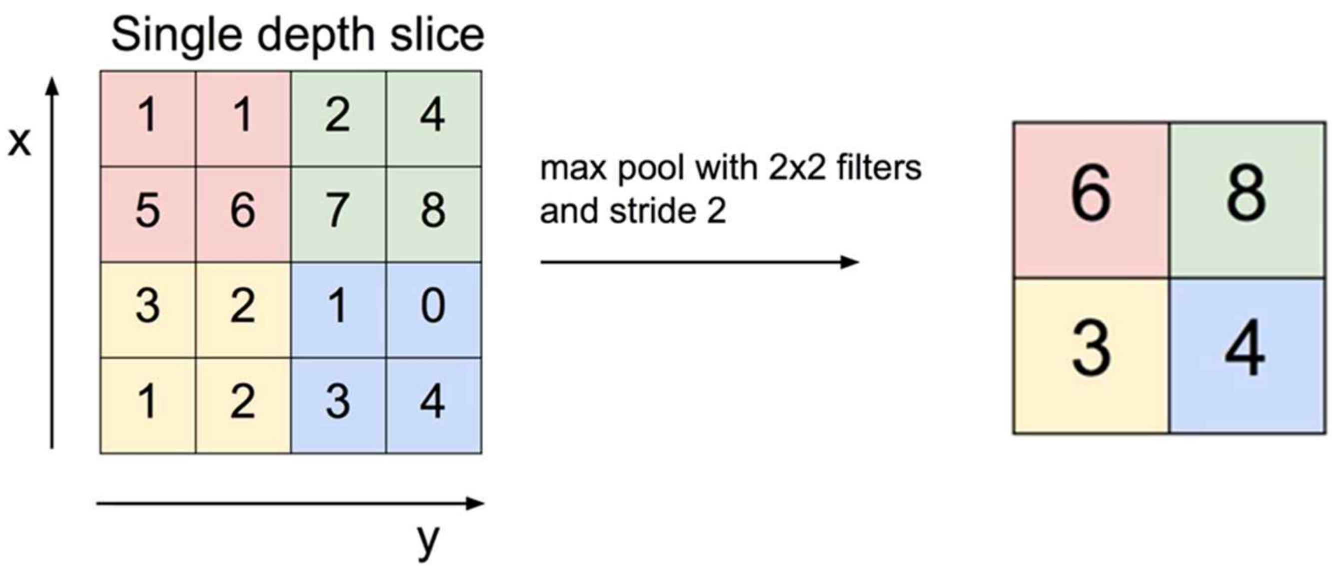
Base Knowledge: CNN (Convolution Neural Network)

- CNN
  - Conv Layer



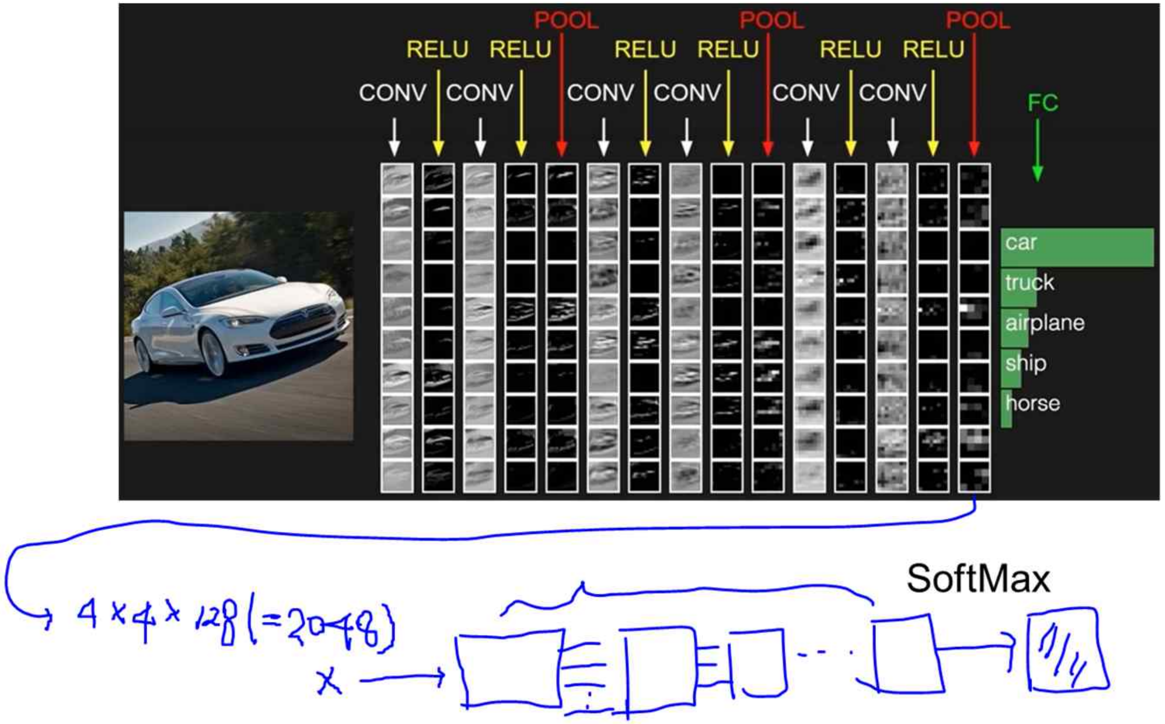
Base Knowledge: CNN (Convolution Neural Network)

- CNN
  - Pooling Layer
    - Max Pooling (avg Pooling, etc.)



Base Knowledge: CNN (Convolution Neural Network)

- CNN
  - FC (Fully Connected) Layer



인공 신경망 Deep Neural Network

Project #1: TSR(Traffic Sign Recognition) using CNN

TSR using CNN

- **Build a Traffic Sign Recognition Project**
  - Load the data set (German Traffic Sign: http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset)
    - Using provided "pickle" files

```
# Load pickled data
import pickle

# TODO: Fill this in based on where you saved the training and testing data

training_file = 'traffic-signs-data/train.p'
validation_file = 'traffic-signs-data/valid.p'
testing_file = 'traffic-signs-data/test.p'

with open(training_file, mode='rb') as f:
    train = pickle.load(f)
with open(validation_file, mode='rb') as f:
    valid = pickle.load(f)
with open(testing_file, mode='rb') as f:
    test = pickle.load(f)

X_train, y_train = train['features'], train['labels']
X_valid, y_valid = valid['features'], valid['labels']
X_test, y_test = test['features'], test['labels']
```

Containing raw pixel data of the traffic sign images

  - Explore, Summarize and visualize the data set
  - Design, Train and Test a CNN Model architecture
  - Use the model to make predictions on new images
  - Analyze the softmax probabilities of the new images

Project #1: TSR(Traffic Sign Recognition) using CNN

- **TSR using CNN**
  - **Build a Traffic Sign Recognition Project**
    - Load the data set
    - Explore, Summarize and visualize the data set
      - The size of training/validation/test set is 34799/4410/12630.
      - The shape of a traffic sign images is (32, 32, 3).
      - The number of unique classes/labels in the data set is 43.

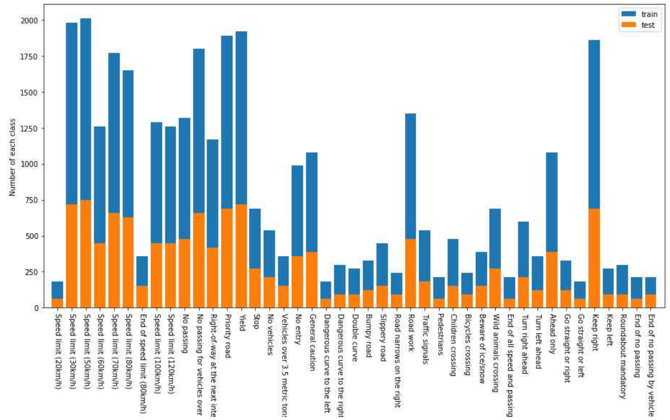
```
image_shape = X_train[0].shape
```

```
n_classes = len(np.unique(y_train))
```

```
Number of training examples = 34799
Number of validation examples = 4410
Number of testing examples = 12630
Image data shape = (32, 32, 3)
Number of classes = 43
```

 # TODO: make code

 # TODO: Reference the provided code



    - Design, Train and Test a CNN Model architecture
    - Use the model to make predictions on new images
    - Analyze the softmax probabilities of the new images

Project #1: TSR(Traffic Sign Recognition) using CNN

- **TSR using CNN**
  - **Build a Traffic Sign Recognition Project**
    - Load the data set
    - Explore, Summarize and visualize the data set

 # TODO: Reference the provided code

(1)

```
### Data exploration visualization code goes here.
### Feel free to use as many code cells as needed.
import matplotlib.pyplot as plt
import pandas as pd
import random as rnd
import cv2
# Visualizations will be shown in the notebook.
%matplotlib inline

readfile = pd.read_csv('signnames.csv')
sign_name = readfile['SignName'].values

train_classes, train_class_cnt = np.unique(y_train, return_counts = True)
test_classes, test_class_cnt = np.unique(y_test, return_counts = True)

fig, axis = plt.subplots(2,4, figsize=(15,6))
fig.subplots_adjust(hspace=0.2, wspace=0.2)
axis = axis.ravel()
for i in range(8):
    idx = rnd.randint(0, n_train)
    img = X_train[idx]
    axis[i].axis('off')
    axis[i].set_title(sign_name[y_train[idx]])
    axis[i].imshow(img)
```

```
fig0 = plt.figure(figsize=(13,10))
plt.bar(np.arange(n_classes), train_class_cnt, align='center', label='train')
plt.bar(np.arange(n_classes), test_class_cnt, align='center', label='test')
plt.xlabel('Class: Name of Traffic sign')
plt.ylabel('Number of each class')
plt.xlim([-1, n_classes])
plt.xticks(np.arange(n_classes), sign_name, rotation=270)
plt.legend()
plt.tight_layout()

plt.show()
```

(2)

    - Design, Train and Test a CNN Model architecture
    - Use the model to make predictions on new images
    - Analyze the softmax probabilities of the new images

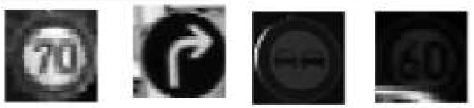
Project #1: TSR(Traffic Sign Recognition) using CNN

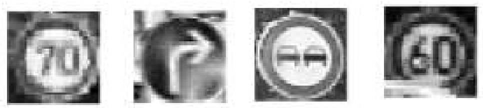
- **TSR using CNN**
  - **Build a Traffic Sign Recognition Project**
    - » Load the data set
    - » Explore, Summarize and visualize the data set
    - » Design, Train and Test a CNN Model architecture
      - Pre-processing image data
        - ✓Grayscale images & normalize

 # TODO: Apply CLAHE (Contrast Limited Adaptive Histogram Equalization) to Grayscale using provided code and Plotting CLAHE images

```
X_train_gray = []
X_train_CLAHE = []
X_valid_gray = []
X_valid_CLAHE = []
X_test_gray = []
X_test_CLAHE = []

clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(4,4))
for i in range(n_train):
    X_train_gray.append(cv2.cvtColor(X_train[i], cv2.COLOR_RGB2GRAY))
    X_train_CLAHE.append(clahe.apply(X_train_gray[i]))
for i in range(n_validation):
    X_valid_gray.append(cv2.cvtColor(X_valid[i], cv2.COLOR_RGB2GRAY))
    X_valid_CLAHE.append(clahe.apply(X_valid_gray[i]))
for i in range(n_test):
    X_test_gray.append(cv2.cvtColor(X_test[i], cv2.COLOR_RGB2GRAY))
    X_test_CLAHE.append(clahe.apply(X_test_gray[i]))

fig, axis = plt.subplots(2,4, figsize=(15,6))
fig.subplots_adjust(hspace=0.2, wspace=0.2)
axis = axis.ravel()
```

```
for i in range(8):
    idx = rnd.randint(0, n_train)
    img = X_train_CLAHE[idx]
    axis[i].axis('off')
    axis[i].set_title(sign_name[y_train[idx]])
    axis[i].imshow(img, 'gray')

X_train_arr = np.array(X_train_CLAHE)
X_valid_arr = np.array(X_valid_CLAHE)
X_test_arr = np.array(X_test_CLAHE)
X_train_arr = X_train_arr.reshape(X_train_arr.shape + (1,))
X_valid_arr = X_valid_arr.reshape(X_valid_arr.shape + (1,))
X_test_arr = X_test_arr.reshape(X_test_arr.shape + (1,))
X_train = norm(X_train_arr)
X_valid = norm(X_valid_arr)
X_test = norm(X_test_arr)
```





    - » Use the model to make predictions on new images
    - » Analyze the softmax probabilities of the new images

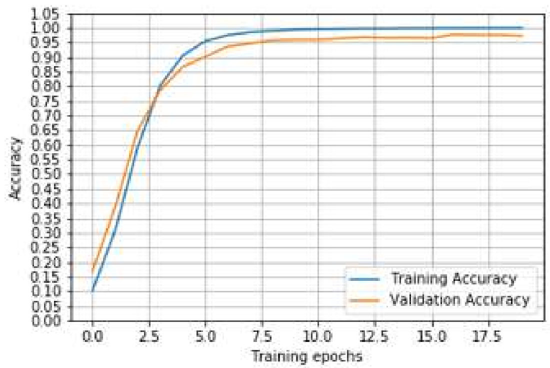
Project #1: TSR(Traffic Sign Recognition) using CNN

- **TSR using CNN**
  - **Build a Traffic Sign Recognition Project**
    - Load the data set
    - Explore, Summarize and visualize the data set
    - Design, Train and Test a CNN Model architecture
      - Pre-processing image data
        - ✓Grayscale images & normalize

 # TODO: Make CNN Architecture and plot the accuracy

Layer	Description
Input	32x32x1 (CLAHE & Normalize)
Convolution 3x3	1x1 stride, same padding, outputs 32x32x96
RELU	
Max pooling	2x2 stride, outputs 16x16x96
Convolution 4x4	1x1 stride, same padding, outputs 16x16x128
RELU	
Max pooling	2x2 stride, outputs 8x8x128
Convolution 3x3	1x1 stride, same padding, outputs 8x8x256
RELU	
Max pooling	2x2 stride, outputs 4x4x256
Convolution 4x4	1x1 stride, same padding, outputs 4x4x256
RELU	
Dropout	0.5
Flatten	4x4x256 = 4096
Fully connected	(4096, 1024)
Dropout	0.5
Fully connected	(1024, 256)
Dropout	0.5
Fully connected	(256, 43)

- Training set accuracy of 99%
- Validation set accuracy of 97.3%
- Test set accuracy of 95.6%



- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images

Thank you

Q&A

www.kopo.ac.kr
jsshin7@kopo.ac.kr

