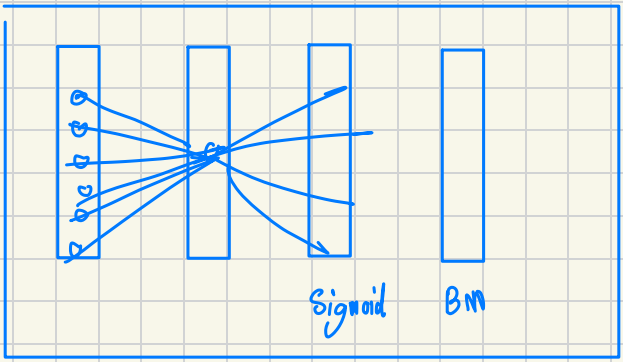
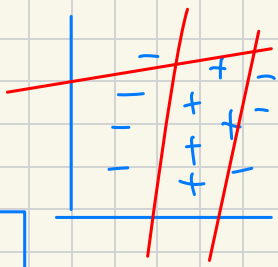
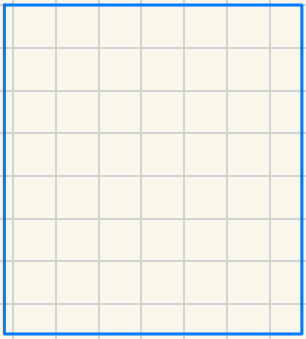
CNN : Convolution Neural Network



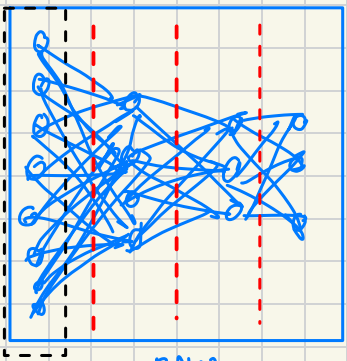
BNN





128×128

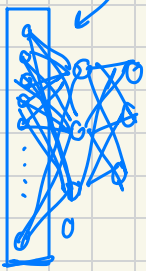
Convolution



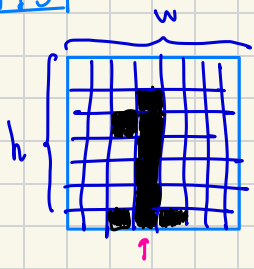
feature images

BNN

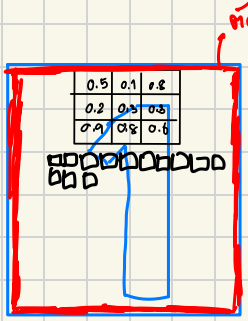
=> 28×28



MNIST

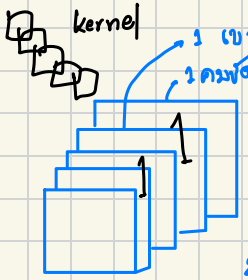


1 1 1 4 4 4 0 4 5 4 4 9 4 5 → sequence vector ตามลำดับการเขียนเลข



filter mask

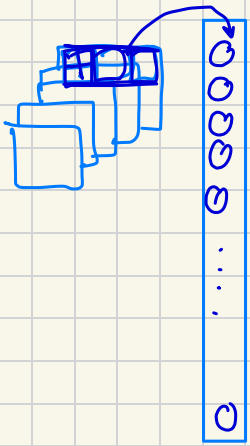
0.5	0.1	0.8
0.2	0.3	0.3
0.9	0.8	0.6



$26 \times 26 \times 5 = 3380$

$28 \times 28 \rightarrow 26 \times 26$ (ตัดขอบบนล่างซ้ายขวา)

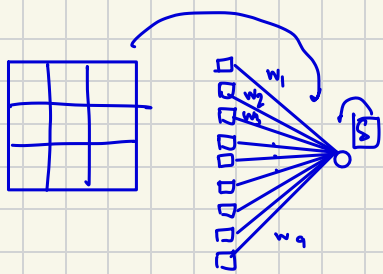
Greyscale 1 (7 ช่อง)

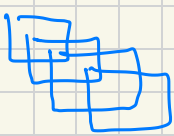


$\longrightarrow$ $13 \times 13 \times 5 = 169 \times 5 = 845$

pooling รวม information

เอา n Pixel มาตัวแทน



ImageNet $\longrightarrow$  $\rightarrow$ pooling $\rightarrow$  $\rightarrow$ pooling

VGG16

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 1: Different configurations of VGG. Source

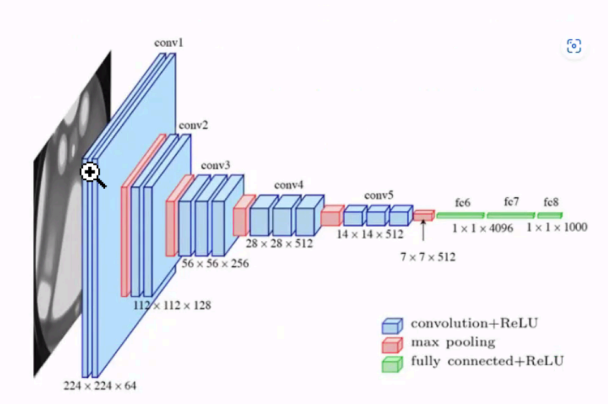


Figure 1: The architecture of VGG16. Source: Researchgate.net

[1] 2024-04-07 02:53:44 (1.24 MB/s) - 'Koala.-GettyImages-1266039701-4

```python
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import preprocess_i
import numpy as np

model = ResNet50(weights='imagenet')

img_path = 'image1.jpg'
img = image.load_img(img_path, target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

preds = model.predict(x)
# decode the results into a list of tuples (class, description,
# (one such list for each sample in the batch)
print('Predicted:', decode_predictions(preds, top=3)[0])
```

j?crop=0.444xw:1.00xh puppy.jpg

Transfer learning source code

```python
LEARNING_RATE = 1e-2

# Define paths
train_data_dir = 'fgvc-aircraft-2013b/data/
val_data_dir = 'fgvc-aircraft-2013b/data/i
test_data_dir = 'fgvc-aircraft-2013b/data/

# Load and preprocess dataset
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

val_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='categorical'
)
```

```python

# Load pre-trained MobileNetV2 model
base_model = MobileNetV2(
    weights='imagenet',
    include_top=False,
    input_tensor=Input(shape=(224, 224, 3))
)

# Construct the head of the model that will be placed on top of the base model
head_model = base_model.output เอา mobileNet ต่อท้ายเพื่อสร้าง CNN ต่อ
head_model = AveragePooling2D(pool_size=(7, 7))(head_model)
head_model = Flatten(name='flatten')(head_model)
head_model = Dense(128, activation='relu')(head_model)
head_model = Dropout(0.5)(head_model)
head_model = Dense(len(train_generator.class_indices), activation='softmax')(head_mode

# Place the head FC model on top of the base model
model = Model(inputs=base_model.input, outputs=head_model)

# Freeze the base model layers so they are not updated during training
for layer in base_model.layers:
    layer.trainable = False
```

ไม่ train ให้ train ต่อนหลัง

```
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```

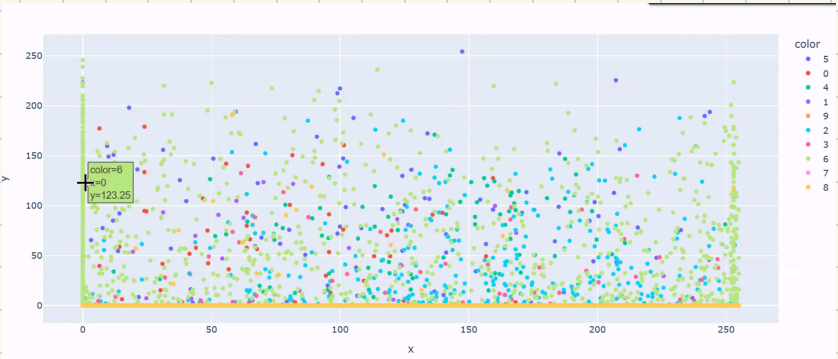
+ Code

```
y_train[:3,:]
```

```
rray([[0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
      [1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
      [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.]], dtype=float32)
```

เปลี่ยนเป็น one-hot

batch_size = 500 → เวลา train 60,000 ตัว ก็ train 108 รอบ / batch
รอบละ 500 รูป
เท่าไม่ได้ train ทีละ 1 รูป



scatter คุณกระจายตัวของรูปภาพ

precision recall support

support คือจำนวนภาพที่ตกไป

	precision	recall	f1-score	support
A300				
0.59	0.70	0.64		33
Boeing 707				
0.70	0.42	0.53		33
C-130				
0.89	0.94	0.91		33
DC-10				
0.57	0.70	0.63		33
F-16				
1.00	0.94	0.97		33
accuracy			0.74	165
macro avg	0.75	0.74	0.74	165
weighted avg	0.75	0.74	0.74	165

Boeing

```
Confusion Matrix:
[[23  2  1  7  0]
 [ 9 14  0 10  0]
 [ 2  0 31  0  0]
 [ 5  4  1 23  0]
 [ 0  0  2  0 31]]
```

คุณทายว่าเป็น DC-10