파이썬 딥러닝

이성주

seongjoo@codebasic.io

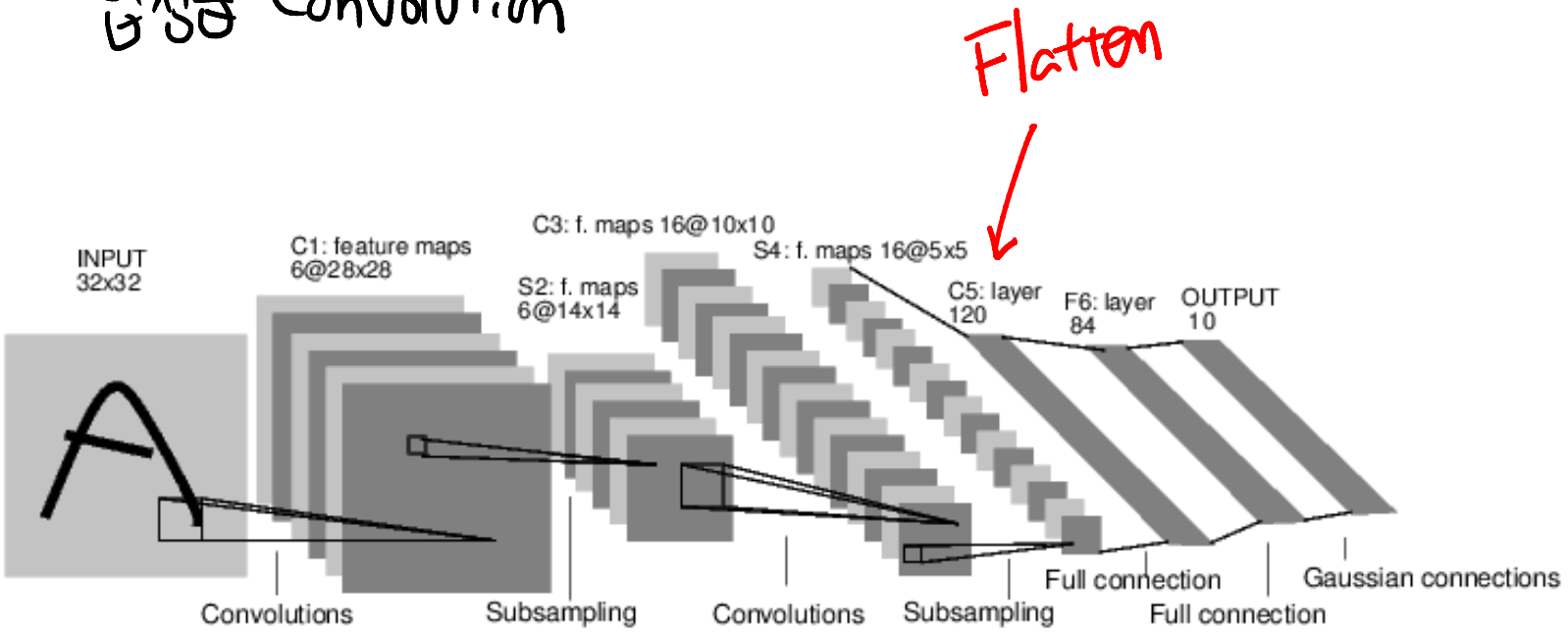
CNN

Convolutional Neural Network

합성곱 신경망

1998 LeNet ~ CNN

합성곱 Convolution



합성곱 연산

$$
\begin{aligned}
&1\times2+2\times0+3\times1\\
+\;&0\times0+1\times1+2\times2\\
+\;&3\times1+0\times0+1\times2
\end{aligned}
$$

x

1	2	3	0
0	1	2	3
3	0	1	2
2	3	0	1

⊗

filter 또는 kernel ← "weight"

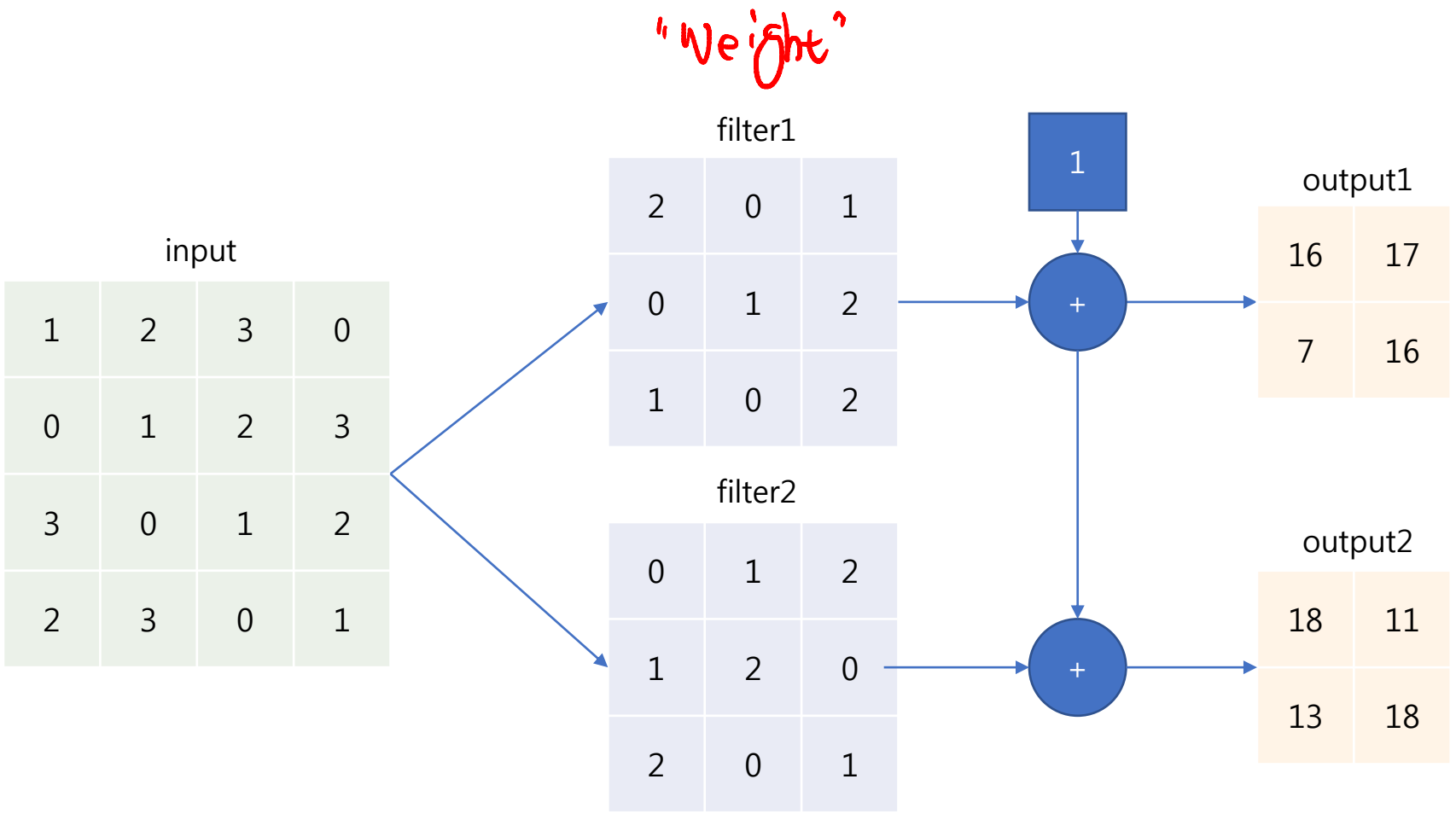
2	0	1
0	1	2
1	0	2

3x3

→

15	16
6	15

2D 합성곱 층


"Weight"
input
1 2 3 0
0 1 2 3
3 0 1 2
2 3 0 1
filter1
2 0 1
0 1 2
1 0 2
filter2
0 1 2
1 2 0
2 0 1
1
+
+
output1
16 17
7 16
output2
18 11
13 18

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

Out[4]: (60000, 28, 28)

In [5]:
```
X_test.shape
```

$\Delta w := \boxed{\text{ERROR} \times x_i}$

Out[5]: (10000, 28, 28)

$$\frac{100 + \Delta w}{1 + \Delta w}$$

In [6]:
```
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
```

In [8]:
```
X_train /= 255
X_test /= 255
```

In [ ]:

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

In [11]:
```
Y_train = np_utils.to_categorical(y_train)
Y_test = np_utils.to_categorical(y_test)
```

In [12]:
```
model = Sequential()
```

In [13]:
```
from keras.layers.convolutional import Conv2D
```

channel

In [14]:
```
model.add(Conv2D(
    20, kernel_size=5, padding='same', input_shape=(28, 28, 1)))
```

In [ ]:

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

In [17]:
```
X_train = np.expand_dims(X_train, -1)
X_train.shape
```

마지막에 축(차원) 확장

Out[17]: (60000, 28, 28, 1)

In [9]:
```
y_train[:10]
```

Out[9]: array([5, 0, 4, 1, 9, 2, 1, 3, 1, 4], dtype=uint8)

In [11]:
```
Y_train = np_utils.to_categorical(y_train)
Y_test = np_utils.to_categorical(y_test)
```

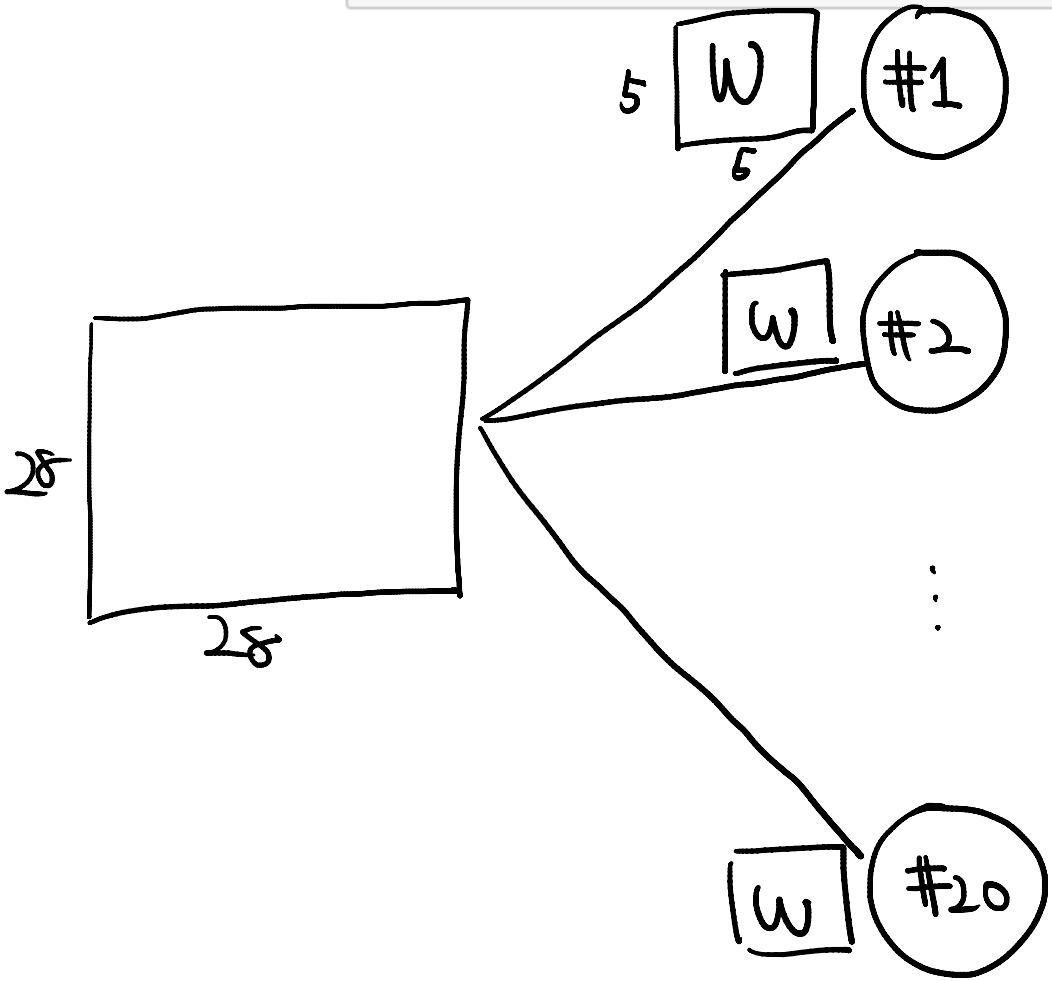
In [12]:
```
model = Sequential()
```

In [13]:
```
from keras.layers.convolutional import Conv2D
```

In [14]:
```
model.add(Conv2D(
    20, kernel_size=5, padding='same', input_shape=(28, 28, 1)))
```

In [ ]:

In [21]:
```
model.add(Conv2D(
    20, kernel_size=5, padding='same', input_shape=(28, 28, 1),
    activation='relu'
))
```



합성곱 연산

padding = "same"

4x4

1	2	3	0
0	1	2	3
3	0	1	2
2	3	0	1

⊗

Filter 3x3

2	0	1
0	1	2
1	0	2

→

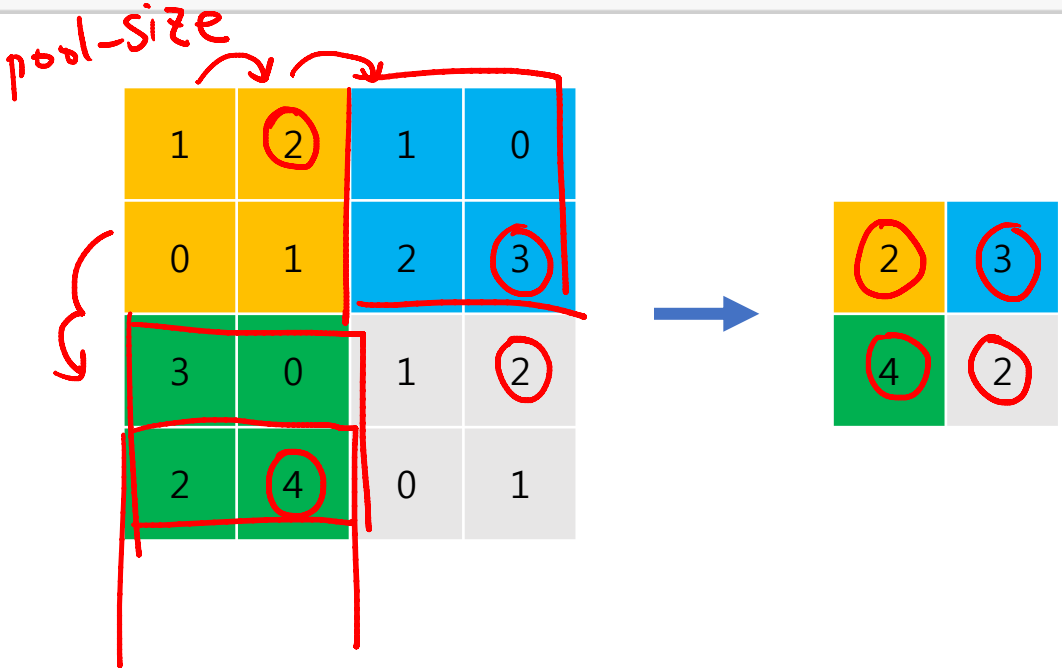
4 x 4

0			0
	15	16	
	6	15	
0			0

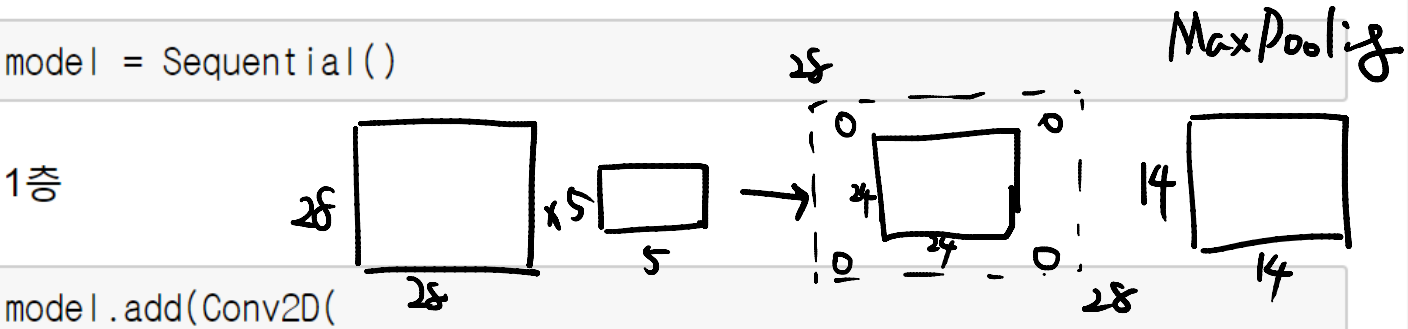
Max Pooling

"step"

```
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
```

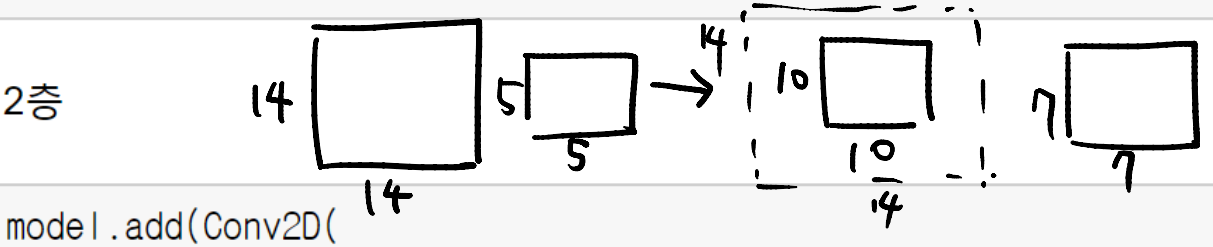


In [20]:
```
model = Sequential()
```



In [21]:
```
model.add(Conv2D(
    20, kernel_size=5, padding='same', input_shape=(28, 28, 1),
    activation='relu'
))
```

In [23]:
```
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
```



In [24]:
```
model.add(Conv2D(
    50, kernel_size=5, padding='same', activation='relu'))
```

In [25]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

출력 준비층

In [26]:
```
from keras.layers.core import Flatten
```

In [27]:
```
model.add(Flatten())
```

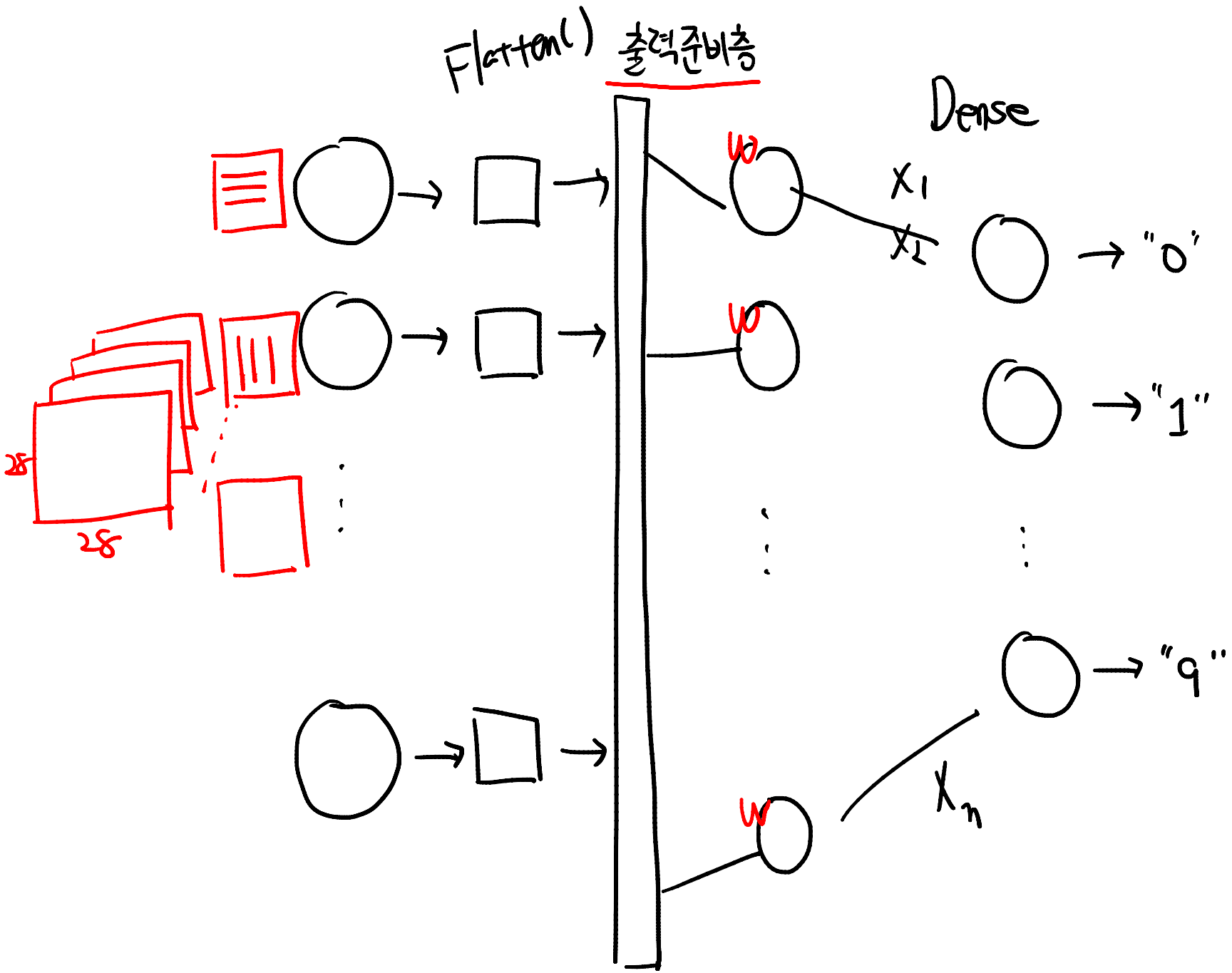
In [28]:
```
model.add(Dense(500, activation='relu'))
```

출력층

In [29]:
```
model.add(Dense(10, activation='softmax'))
```

In [ ]:
```

```


Flatten()
출력준비층
Dense
w
w
w
x1
xn
"0"
"1"
"9"
28
28

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

In [59]:
```
model = Sequential()
```

In [60]:
```
model.add(Conv2D(
    32, kernel_size=3, padding='same',
    input_shape=(32, 32, 3), activation='relu'))
```

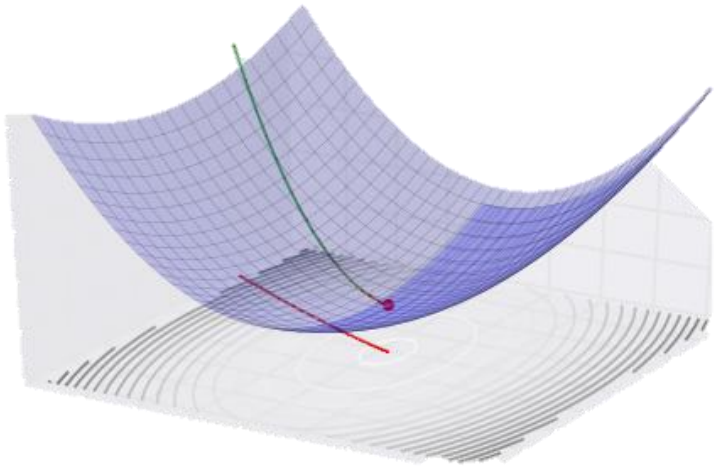
In [61]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

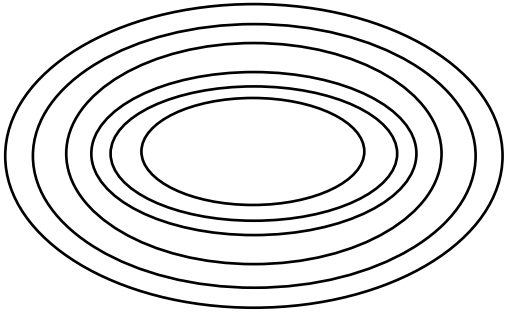
In [62]:
```
model.add(Flatten())
model.add(Dense(512, activation='relu'))
```

In [63]:
```
model.add(Dense(10, activation='softmax'))
```

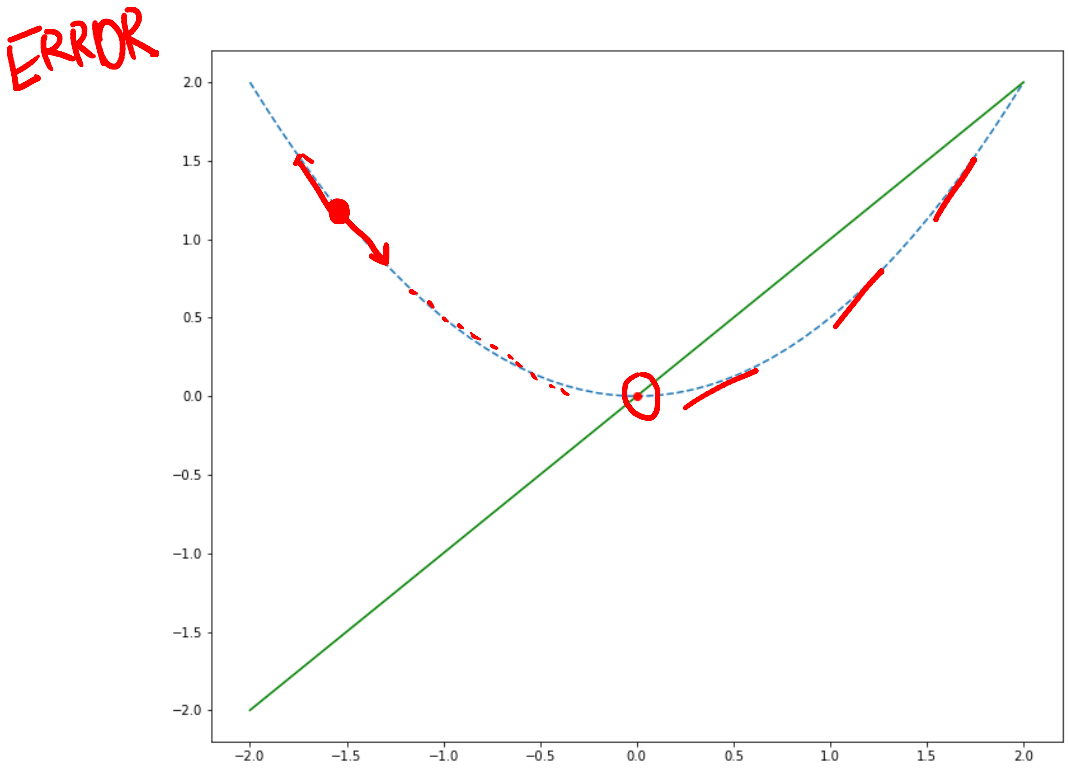
In [64]:
```
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

최적화





경사 하강법 (Gradient Descent)



Optimize = 'sgd'

SGD

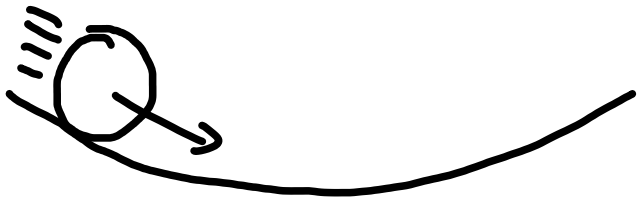
Batch

확률적 경사 하강
Stochastic Gradient Descent

$$w \leftarrow w - \eta \frac{\partial L}{\partial w}$$

학습률

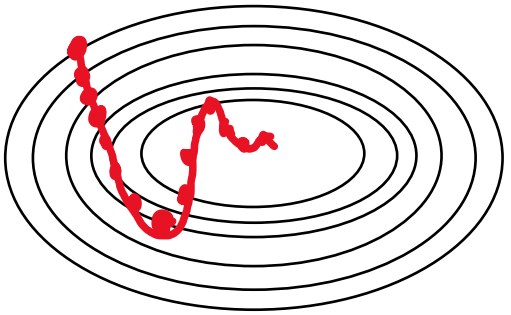
모멘텀



$$w \leftarrow w + v$$

$$v \leftarrow \alpha v - \eta \frac{\partial L}{\partial w}$$

"또르르"



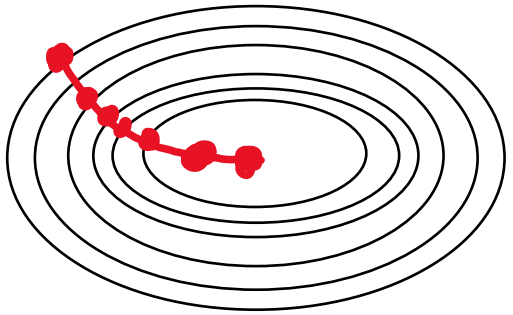
AdaGrad

Optimizer = 'RMSProp'

적응형 학습률

$$W \leftarrow W - \eta \frac{1}{\sqrt{h}} \frac{\partial L}{\partial W}$$

$$h \leftarrow h + \frac{\partial L}{\partial W} \odot \frac{\partial L}{\partial W}$$



*RMSProp으로 개선

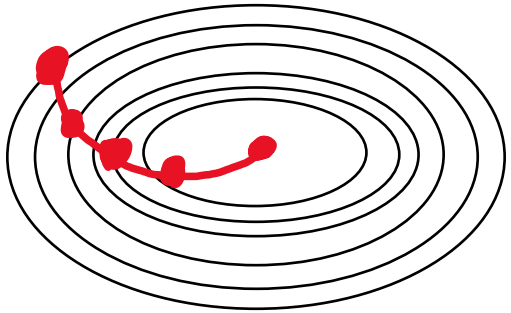
Adam

optimizer = "adam"

" Momentum + AdaGrad "

구르는 공 　　적응형 속도

2015!



jupyter Day 5

Logout

Trusted | Python 3

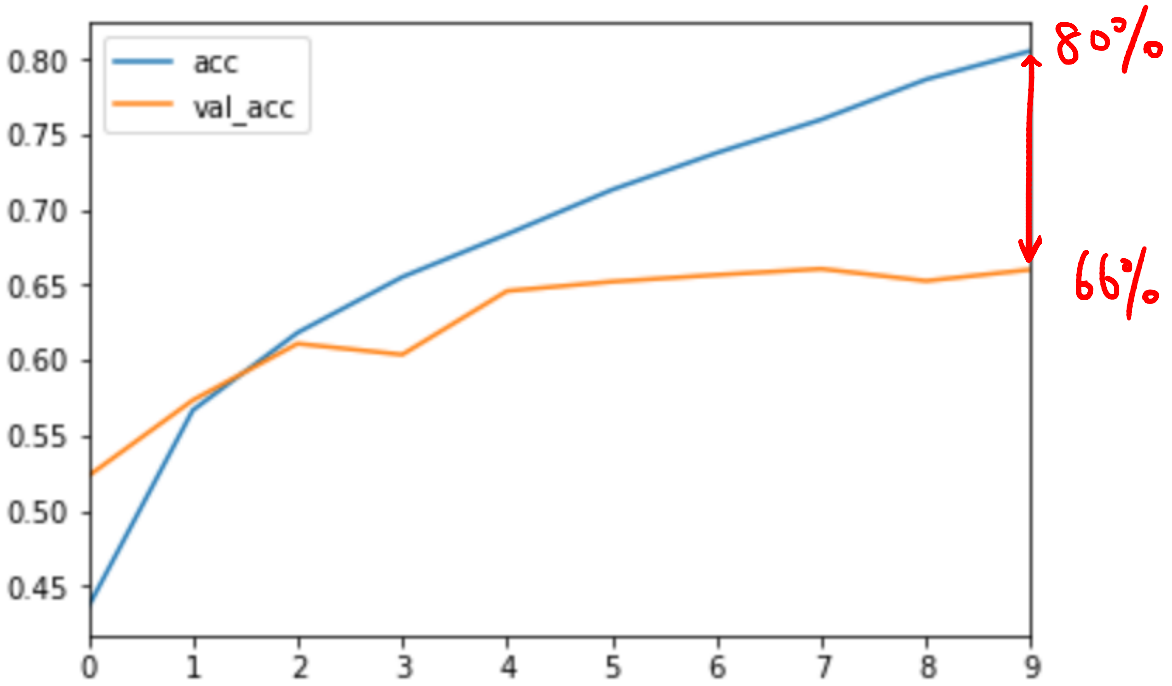
File Edit View Insert Cell Kernel Widgets Help

Code

In [66]:
```
train_results = DataFrame(history.history)
```

In [68]:
```
train_results[['acc', 'val_acc']].plot()
```

Out[68]: <matplotlib.axes._subplots.AxesSubplot at 0xea40470>



jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

Dropout(0.5)

In [70]:
```
model = Sequential()
```

In [71]:
```
model.add(Conv2D(
    32, kernel_size=3, padding='same',
    input_shape=(32, 32, 3), activation='relu'))
```

W=6

In [72]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

In [73]:
```
model.add(Dropout(0.25))
```

25%

In [74]:
```
model.add(Flatten())
model.add(Dense(512, activation='relu'))
```

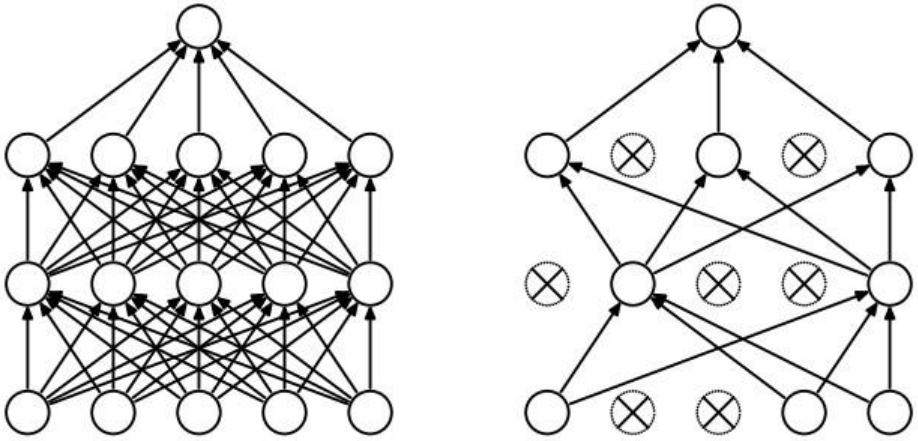
In [75]:
```
model.add(Dropout(0.5))
```

50%

In [76]:
```
model.add(Dense(10, activation='softmax'))
```

In [*]:
```
model.compile(
```

Dropout



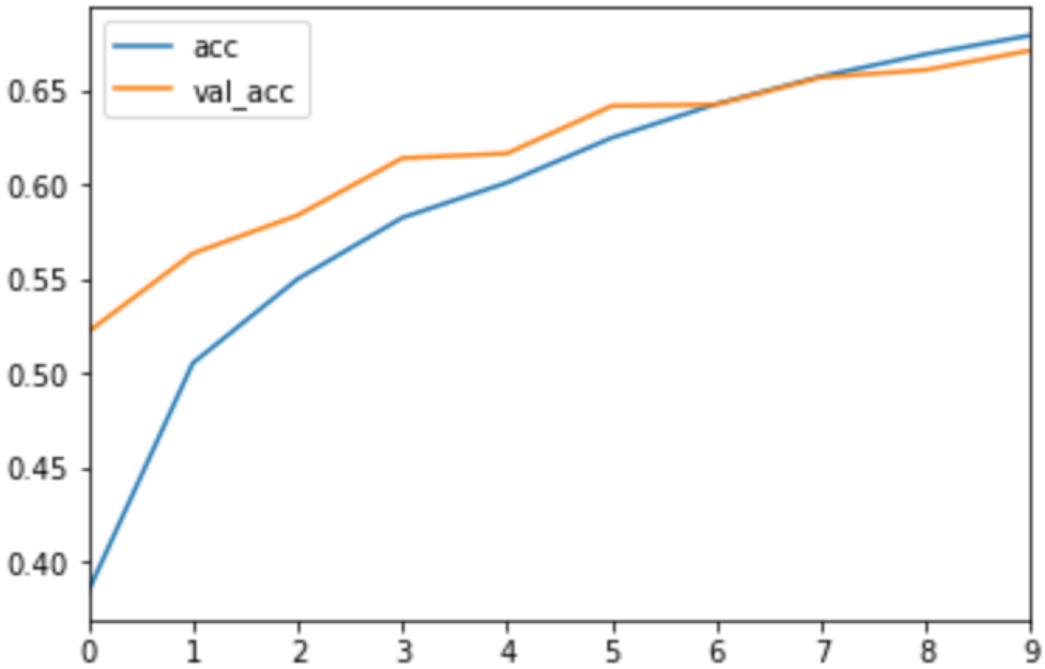
(a) Standard Neural Net

(b) After applying dropout.

In [79]:
```
train_results = DataFrame(history.history)
```

In [80]:
```
train_results[['acc', 'val_acc']].plot()
```

Out[80]: <matplotlib.axes._subplots.AxesSubplot at 0xf5d9588>



jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

보다 깊은 층을 구성해 성능 개선

conv+conv+maxpool+dropout+conv+conv+maxpool+dropout
dense+dropout+dense

In [81]:
```
model = Sequential()
```

In [82]:
conv

```
model.add(Conv2D(
    32, kernel_size=3, padding='same', activation='relu',
    input_shape=(32, 32, 3)
))
```

conv
In [ ]:
```
model.add(Conv2D(
    32, kernel_size=3, padding='same', activation='relu'))
```

maxpool
In [84]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

dropout
In [85]:
```
model.add(Dropout(0.25))
```

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

Code

In [86]:
```
model.add(Conv2D(
    64, kernel_size=3, padding='same', activation='relu'))
```

Conv

In [87]:
```
model.add(Conv2D(
    64, kernel_size=3, padding='same', activation='relu'))
```

Conv

In [84]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

Maxpool

In [88]:
```
model.add(Dropout(0.25))
```

Dropout

In [89]:
```
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
```

In [90]:
```
model.add(Dense(10, activation='softmax'))
```

In [91]:
```
model.compile(
    loss='categorical_crossentropy', optimizer='rmsprop',
    metrics=['accuracy'])
```

jupyter Day 5

Logout

Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

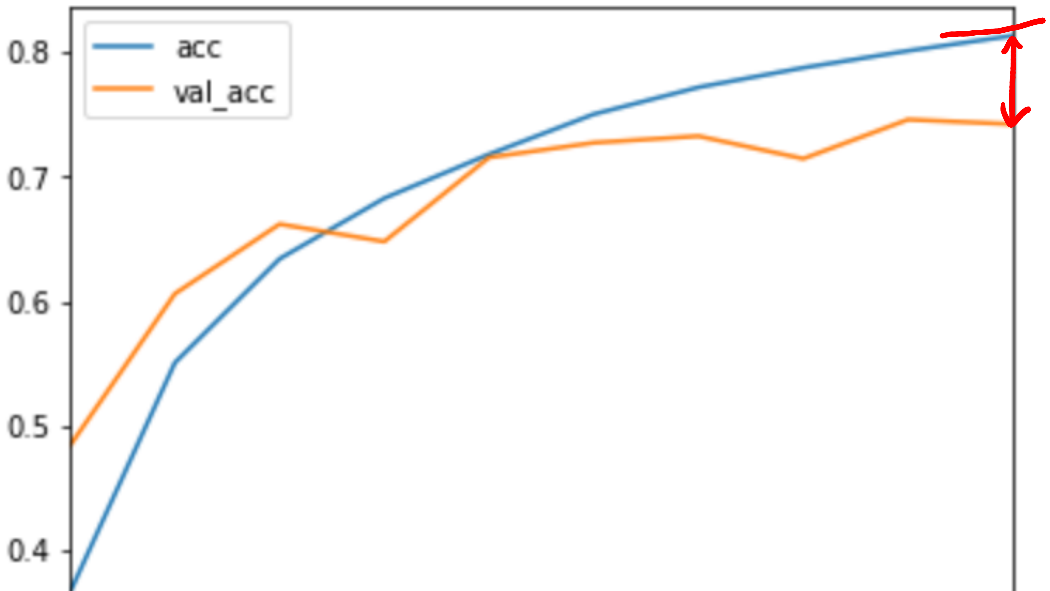
Code

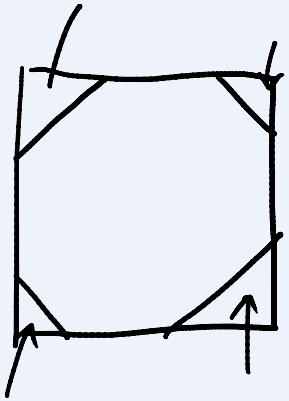
```
40000/40000 [==============================] - 174s 4ms/step - los
s: 0.5636 - acc: 0.8130 - val_loss: 0.8833 - val_acc: 0.7420
```

In [93]:
```
train_results = DataFrame(history.history)
```

In [94]:
```
train_results[['acc', 'val_acc']].plot()
```

Out[94]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x143bf3c8>
```



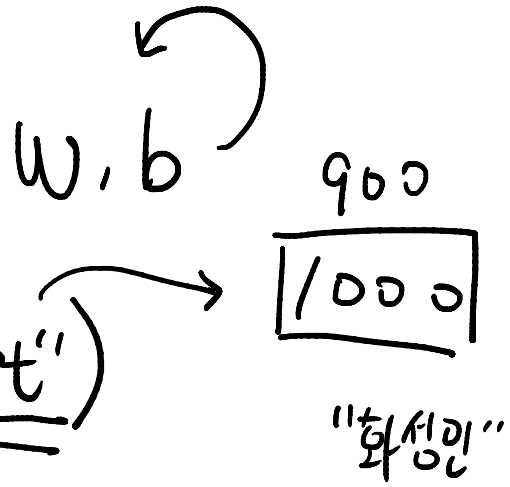


Conv2d(96, kernel_size=3, input_shape=(32,32,3), activation='relu')

Model

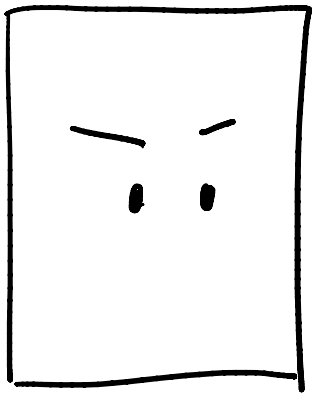
Strided-CNN-C	ConvPool-CNN-C	All-CNN-C
Input 32×32 RGB image		
3×3 conv. 96 ReLU 3×3 conv. 96 ReLU with stride $r = 2$	3×3 conv. 96 ReLU 3×3 conv. 96 ReLU 3×3 conv. 96 ReLU	3×3 conv. 96 ReLU 3×3 conv. 96 ReLU
	3×3 max-pooling stride 2	3×3 conv. 96 ReLU with stride $r = 2$
3×3 conv. 192 ReLU 3×3 conv. 192 ReLU with stride $r = 2$	3×3 conv. 192 ReLU 3×3 conv. 192 ReLU 3×3 conv. 192 ReLU	3×3 conv. 192 ReLU 3×3 conv. 192 ReLU
	3×3 max-pooling stride 2	3×3 conv. 192 ReLU with stride $r = 2$

모델 갱신

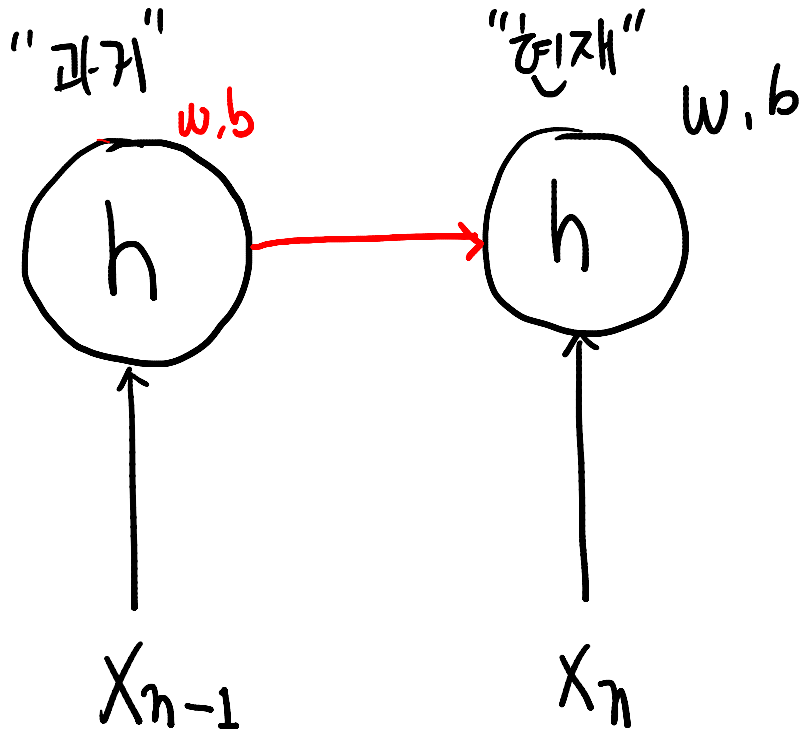


VGG16(weights="imagenet")

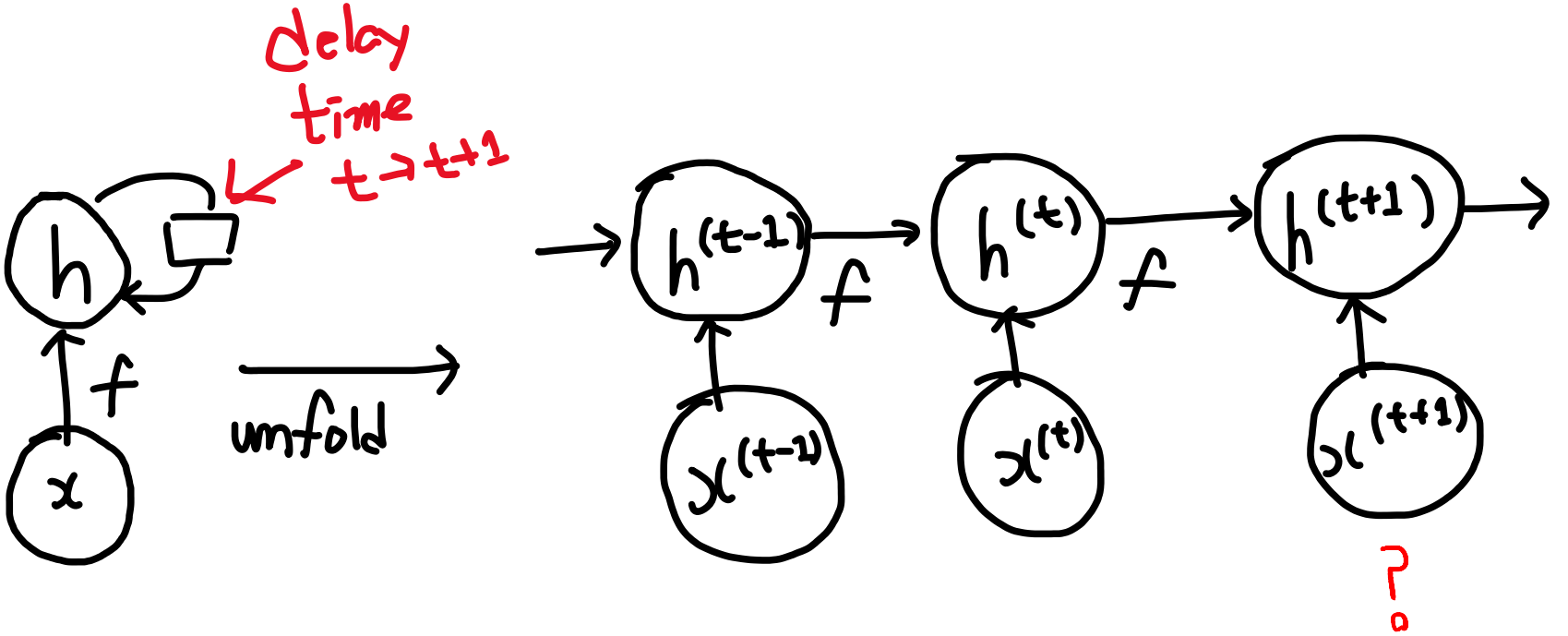
model.fit(X_new_train, y_new_train)



$$y = \begin{cases} 1 & \text{"사용자"} \\ 0 & \text{"사용자 X"} \end{cases}$$


"과거"
w,b
h
"현재"
w,b
h
X_{n-1}
X_n

Recurrent Neural Network



Hidden state

EXTERNAL SIGNAL

$$h^{(t)} = f\left(h^{(t-1)}, x^{(t)}; \theta\right)$$

Unfolding과 매개변수 공유

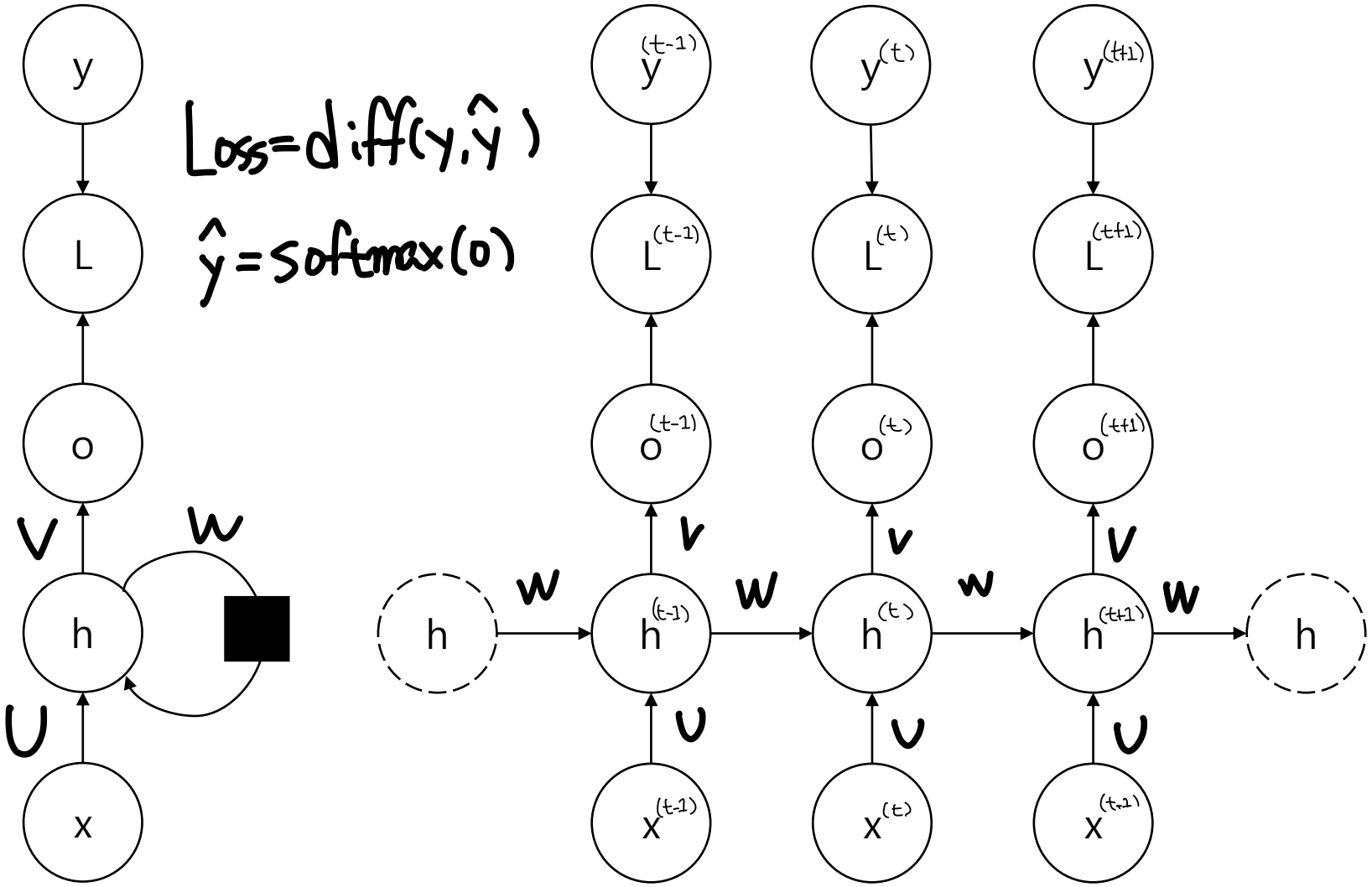
$$h^{(t)} = g^{(t)}\left(x^{(t)}, x^{(t-1)}, x^{(t-2)}, \cdots, x^{(2)}, x^{(1)}\right)$$

"현재까지의 모든 입력"

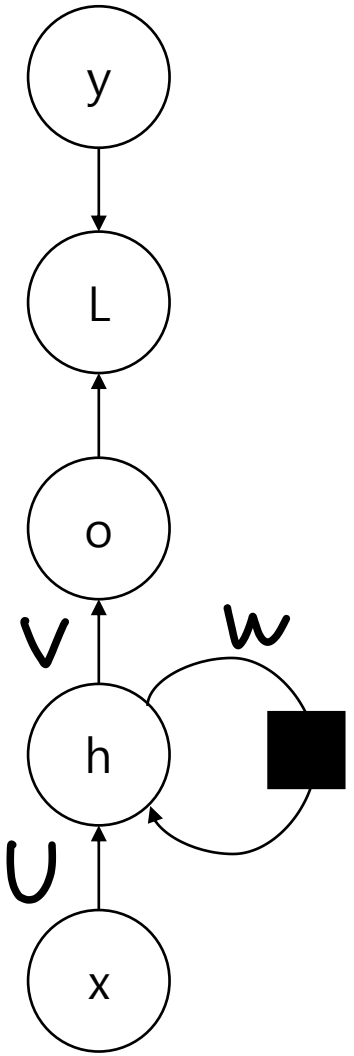
$$= f\left(h^{(t-1)}, x^{(t)}; \theta\right)$$

θ ← W, b

RNN 패턴 1


y
L
o
V
W
h
U
x
Loss=diff(y,ŷ)
ŷ=softmax(o)
y(t-1)
y(t)
y(t+1)
L(t-1)
L(t)
L(t+1)
o(t-1)
o(t)
o(t+1)
V
V
V
W
W
W
W
h
h(t-1)
h(t)
h(t+1)
h
U
U
U
x(t-1)
x(t)
x(t+1)

RNN 패턴 1



$$a^{(t)} = Ux^{(t)} + Wh^{(t-1)} + b$$

$$h^{(t)} = \text{Activation}(a^{(t)})$$

bias

$$o^{(t)} = Vh^{(t)} + c$$

$$\hat{y}^{(t)} = \text{softmax}(o^{(t)})$$

$$\text{Loss} = \text{diff}(y, \hat{y})$$

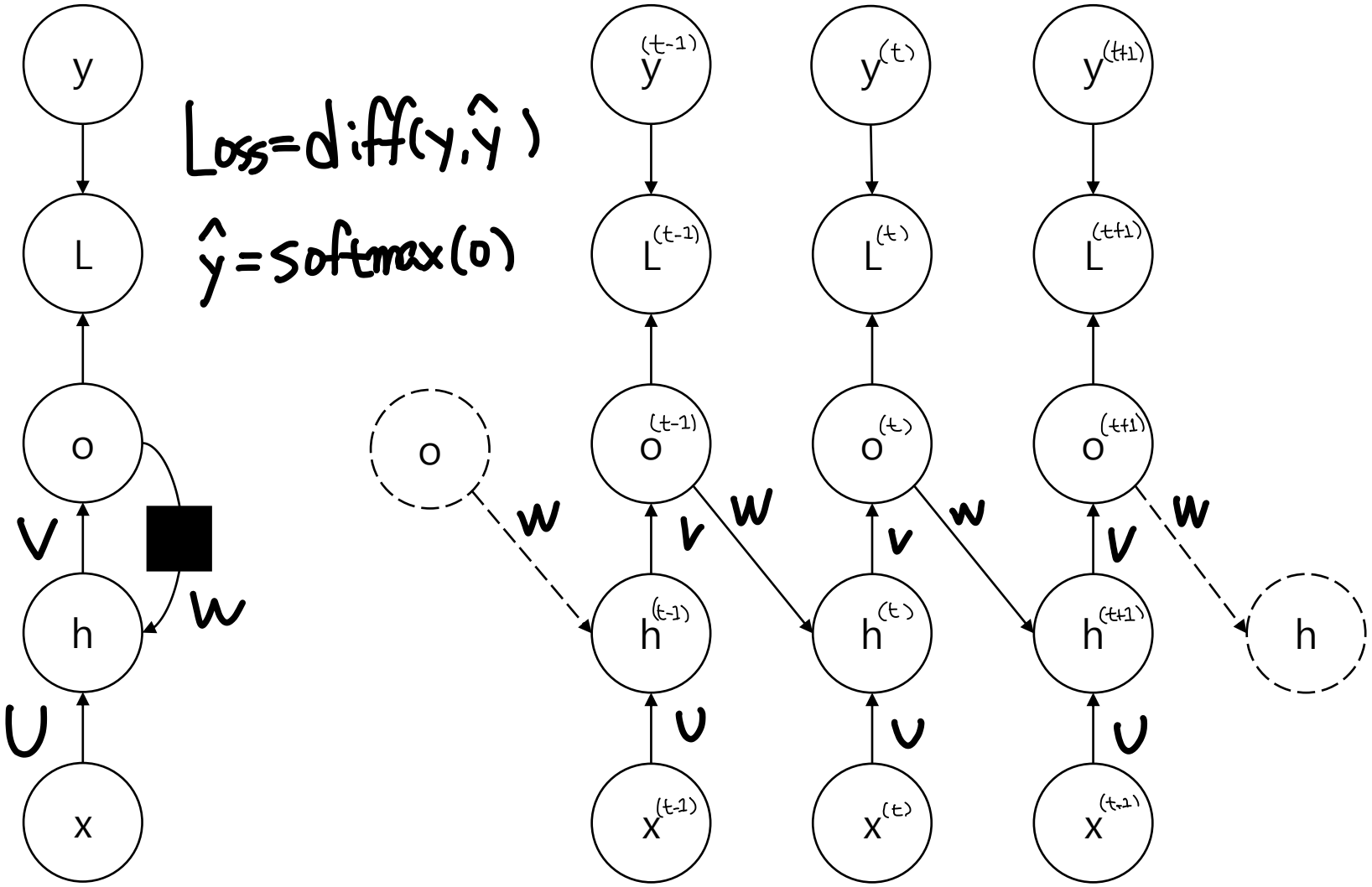
$$L(\{x^{(1)}, \cdots, x^{(\tau)}\}, \{y^{(1)}, \cdots, y^{(\tau)}\})$$

$$= \sum_t L^{(t)}$$

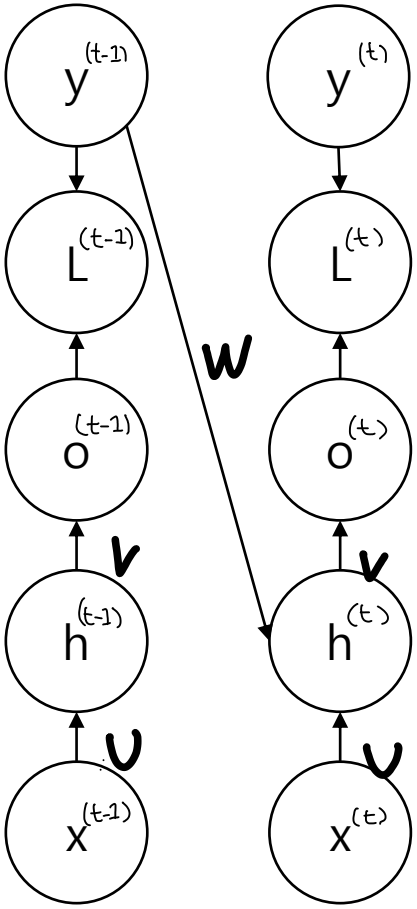
BPTT

Back-prop. through time

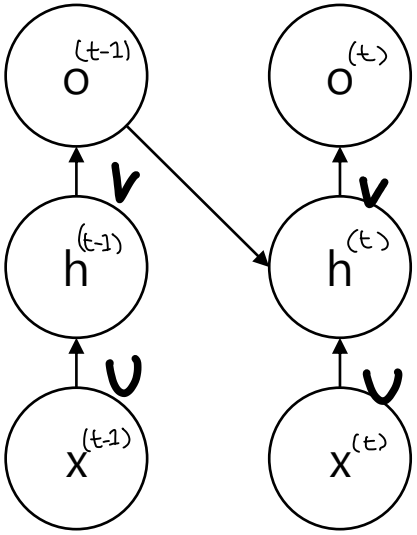
RNN 패턴 2


y
L
o
V
W
h
U
x
Loss=diff(y,ŷ)
ŷ=softmax(o)
y(t-1)
y(t)
y(t+1)
L(t-1)
L(t)
L(t+1)
o
o(t-1)
o(t)
o(t+1)
W
V
W
V
W
V
W
h(t-1)
h(t)
h(t+1)
h
U
U
U
U
x(t-1)
x(t)
x(t+1)

Teacher Forcing

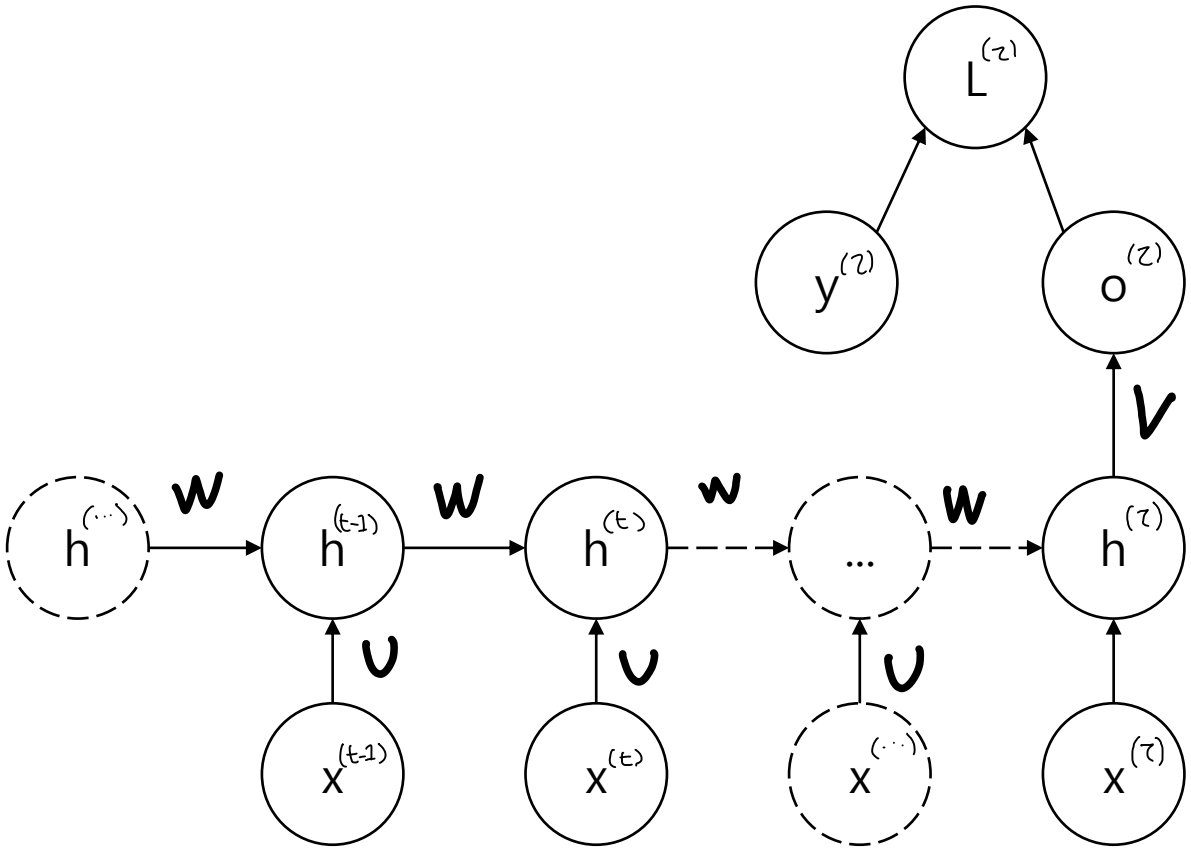


Train time



Test time

RNN 패턴 3



Bidirectional RNN