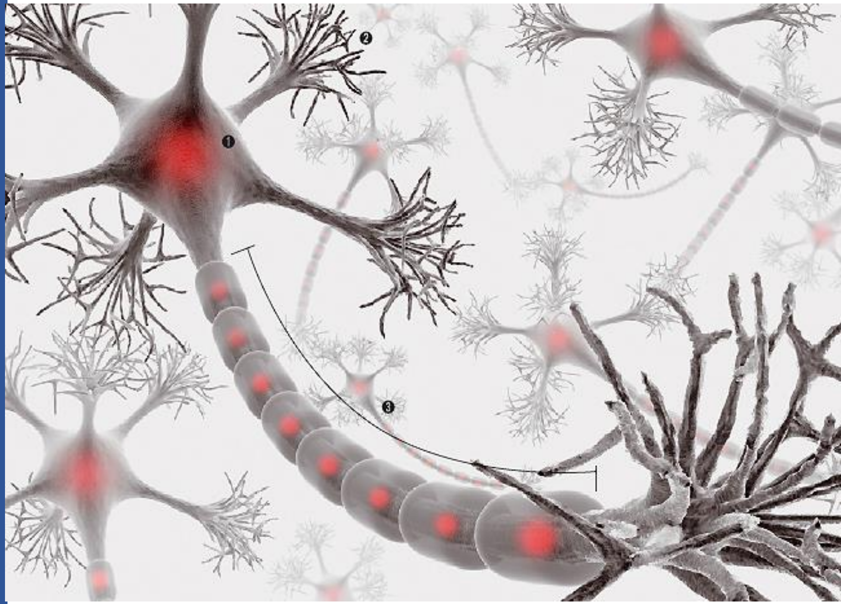
(RNN) 옷 이미지 분류

학습 목표

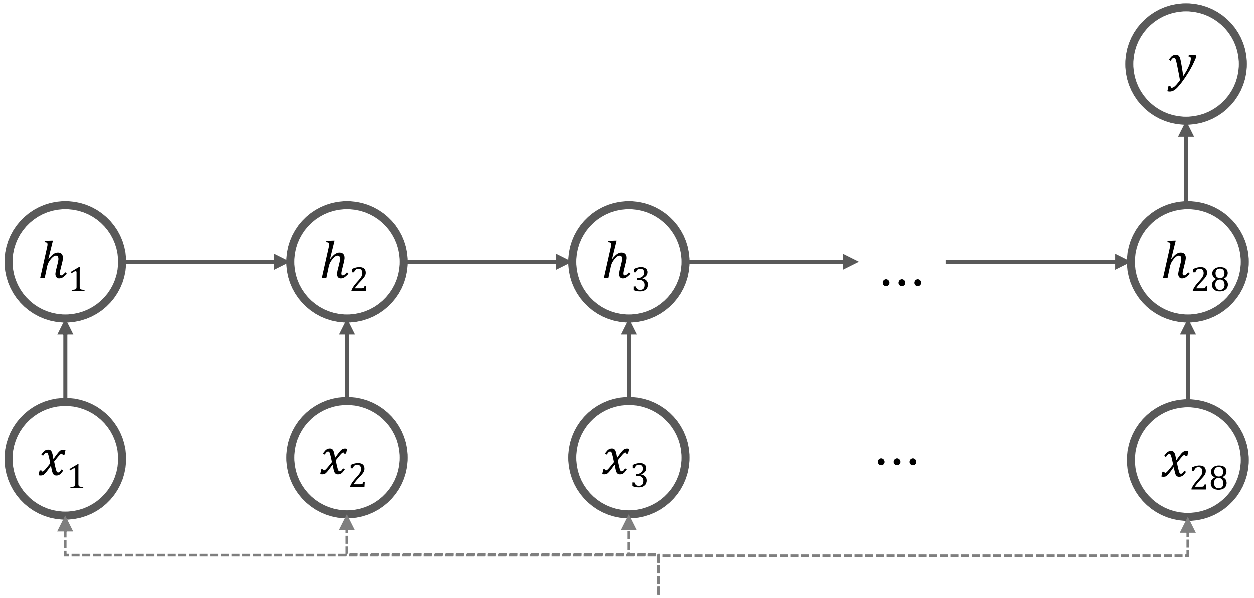
- 운동화나 셔츠 같은 옷 이미지를 분류하는 RNN 신경망 모델을 만들어 본다.



문제

CNN으로 개발된 Fashion MNIST 모델을 다음의 RNN 모델을 바꿔보자!

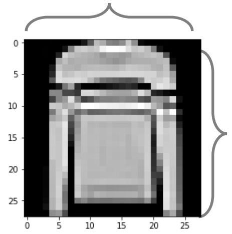
$y = (y_1, y_2, y_3, \ldots, y_{10})$ 0에서 9까지의 클래스에 속할 확률



- Hidden Layer 64
- Time Steps 28

(time steps, input dimension)

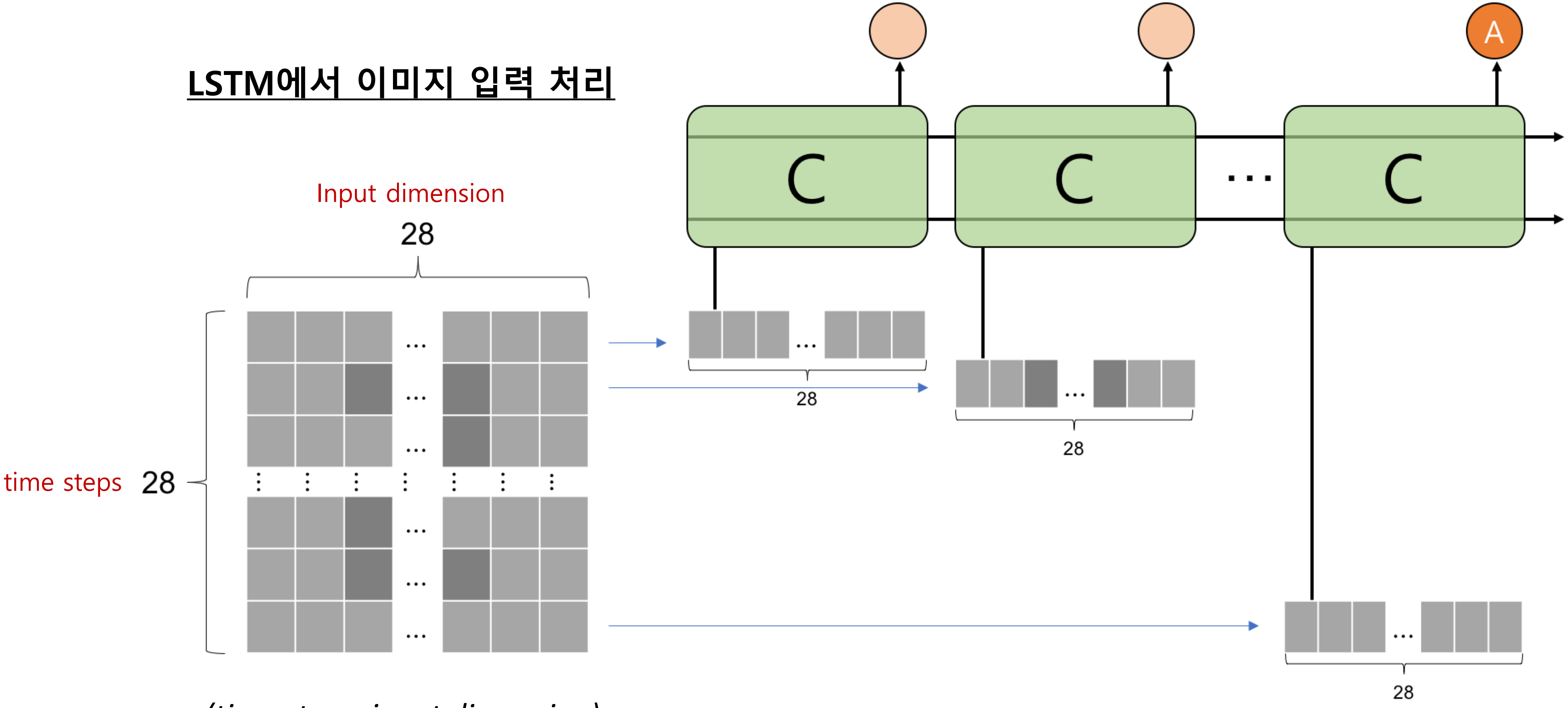
28 input dimension



28 time steps

참고 Keras LSTM Input

LSTM에서 이미지 입력 처리

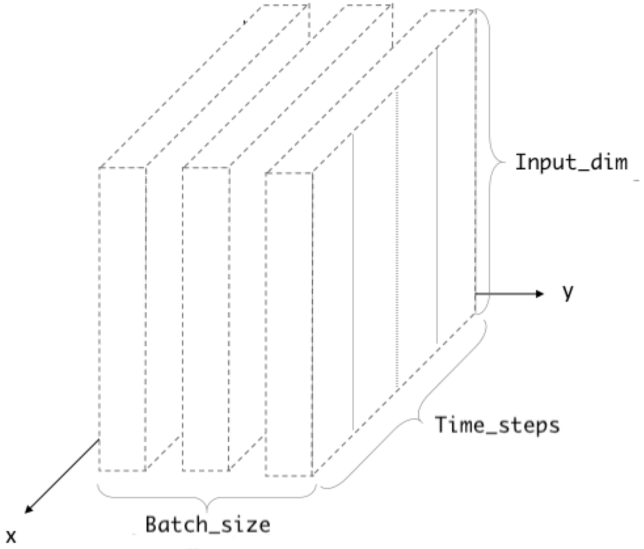


(time steps, input dimension)

https://pozalabs.github.io/lstm/

참고 Keras LSTM Input

LSTM 입력 형태



(batch size, time steps, input dimension)

- **LSTM의 input_shape에는 (*time steps, input dimension*)만 기입**

```
keras.layers.LSTM(units=64, input_shape(28, 28))
```

- Batch size를 매번 다르게 줄 수 있음

- **고정된 Batch size를 명시하려면 batch_input_shape을 사용**

```
keras.layers.LSTM(units=64, batch_input_shape(32, 28, 28))
```

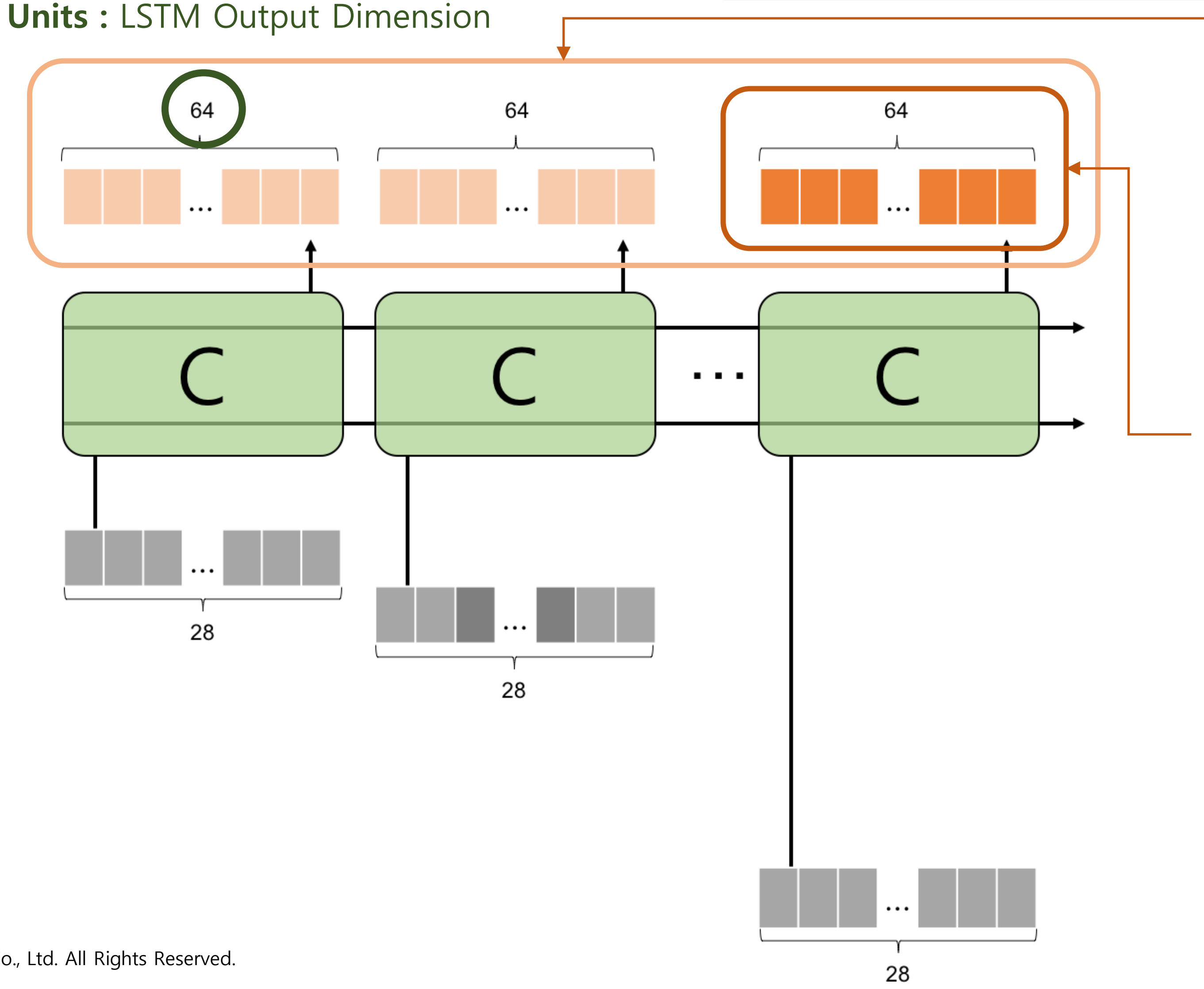
- Batch size를 다르게 주면 오류가 생김

https://medium.com/@shivajbd/understanding-input-and-output-shape-in-lstm-keras-c501ee95c65e

참고 Keras LSTM Output

```
keras.layers.LSTM(units=64, input_shape(28, 28), return_sequence=True)
```

Units : LSTM Output Dimension



return_sequence = True :

전체 time step의 출력 return

(batch size, time steps, output dimension)

return_sequence = False :

최종 time step의 출력만 return

(batch size, output dimension)

https://pozalabs.github.io/lstm/

모델 정의 (문제)

LSTM을 이용해서 RNN 모델을 정의
(Hint : tf.keras.layers.LSTM, tf.keras.layers.BatchNormalization, tf.keras.layers.Dense 사용)

```
# Each MNIST image batch is a tensor of shape (batch_size, 28, 28).
# Each input sequence will be of size (28, 28) (height is treated like time).
input_dim = 28
timesteps = 28
units = 64
output_size = 10  # labels are from 0 to 9

# Build the RNN model (tf.keras.layers.LSTM)
# Batch Normalization 적용
def build_model():
  model = tf.keras.models.Sequential(#your code)
  return model
```

참고 tf.keras.layers.LSTM

```
tf.keras.layers.LSTM(
    units, activation='tanh', recurrent_activation='sigmoid', use_bias=True,
    kernel_initializer='glorot_uniform', recurrent_initializer='orthogonal',
    bias_initializer='zeros', unit_forget_bias=True, kernel_regularizer=None,
    recurrent_regularizer=None, bias_regularizer=None, activity_regularizer=None,
    kernel_constraint=None, recurrent_constraint=None, bias_constraint=None,
    dropout=0.0, recurrent_dropout=0.0, implementation=2, return_sequences=False,
    return_state=False, go_backwards=False, stateful=False, time_major=False,
    unroll=False, **kwargs
)
```

- **units**: Positive integer, dimensionality of the output space.
- **return_sequences**: Boolean. Whether to return the last output. in the output sequence, or the full sequence. Default: False.
- **stateful**: Boolean (default False). If True, the last state for each sample at index i in a batch will be used as initial state for the sample of index i in the following batch.

https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

참고 tf.keras.layers.LSTM

```
inputs = np.random.random([32, 10, 8]).astype(np.float32)
lstm = tf.keras.layers.LSTM(4)

output = lstm(inputs)  # The output has shape `[32, 4]`.

lstm = tf.keras.layers.LSTM(4, return_sequences=True, return_state=True)

# whole_sequence_output has shape `[32, 10, 4]`.
# final_memory_state and final_carry_state both have shape `[32, 4]`.
whole_sequence_output, final_memory_state, final_carry_state = lstm(inputs)
```

https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

모델 훈련

```
model = build_model()

model.compile(loss='sparse_categorical_crossentropy',
                         optimizer='adam',
                         metrics=['accuracy’])
```

```
batch_size = 64
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=10)
model.fit(train_images, train_labels,
          batch_size=batch_size,
          validation_split = 0.2,
          epochs=20,
          callbacks=[early_stop])
```

Thank you!