9. RNN/LSTM(2)

AILab
Hanyang Univ.

Reference : Cs231n / hunkim/DeepLearningZeroToAll

오늘 실습 내용

- Review RNN
- LSTM Basic
- 과제 : Training Long sequence

Review RNN

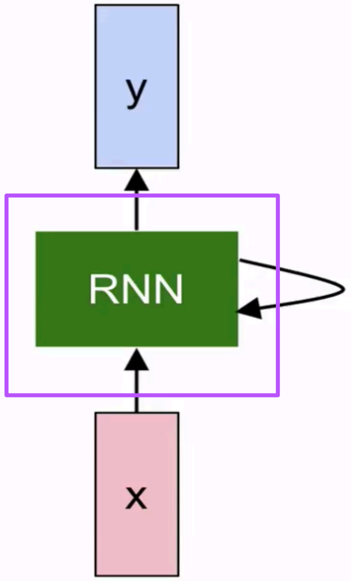
$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

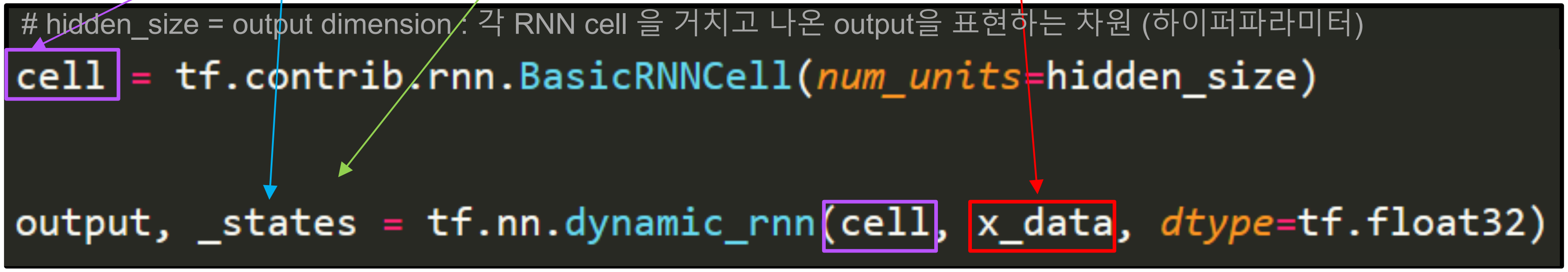
new state

some function with parameters W

old state

input vector at some time step



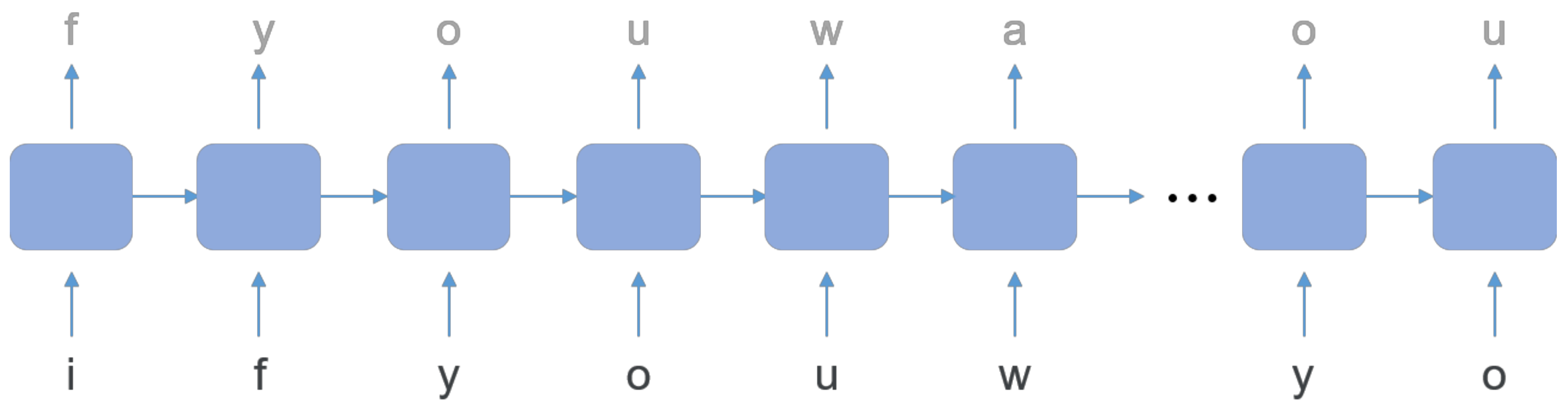


```
# hidden_size = output dimension : 각 RNN cell 을 거치고 나온 output을 표현하는 차원 (하이퍼파라미터)
cell = tf.contrib.rnn.BasicRNNCell(num_units=hidden_size)

output, _states = tf.nn.dynamic_rnn(cell, x_data, dtype=tf.float32)
```

Review RNN

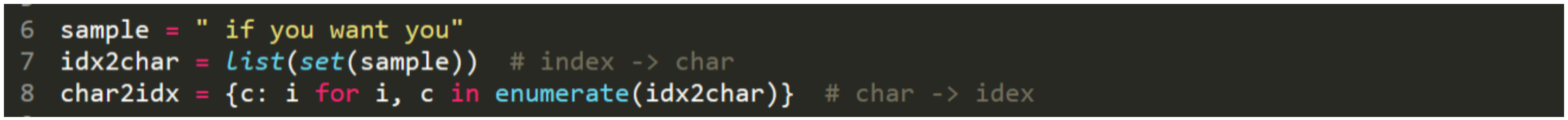
- **Character 단위의 RNN 실습**

각 character의 다음 character 예측하기

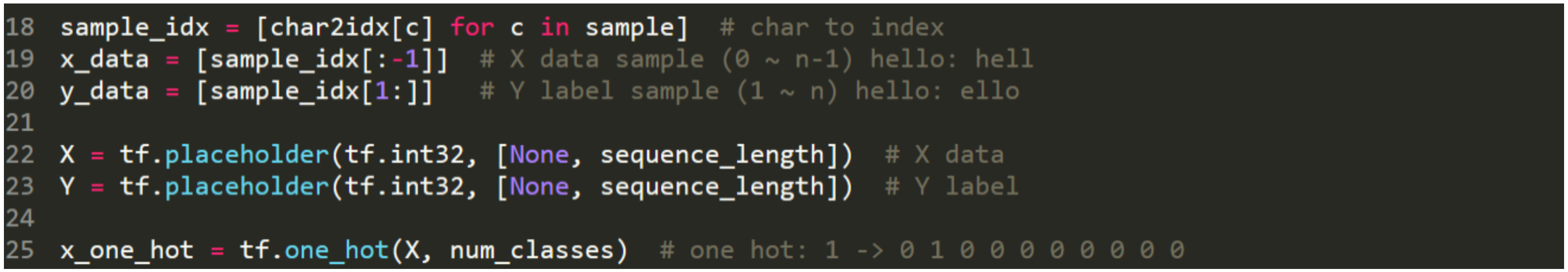


Review RNN : character sequence RNN

- 데이터 생성



```python
sample = " if you want you"
idx2char = list(set(sample))  # index -> char
char2idx = {c: i for i, c in enumerate(idx2char)}  # char -> idex
```



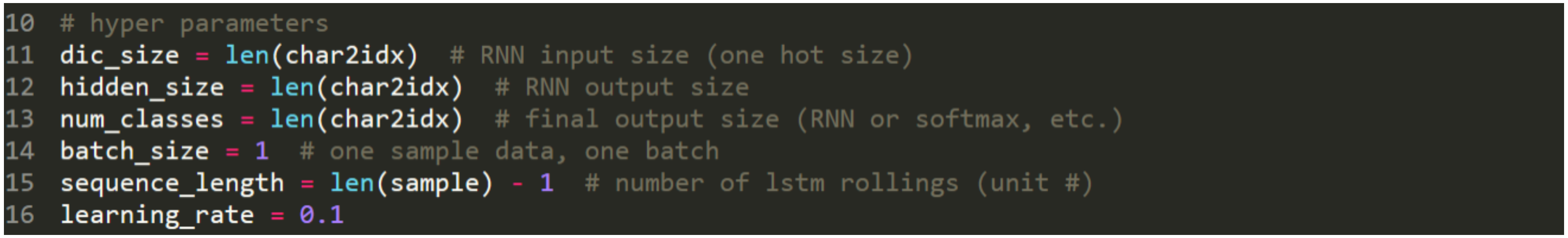
```python
sample_idx = [char2idx[c] for c in sample]  # char to index
x_data = [sample_idx[:-1]]  # X data sample (0 ~ n-1) hello: hell
y_data = [sample_idx[1:]]   # Y label sample (1 ~ n) hello: ello

X = tf.placeholder(tf.int32, [None, sequence_length])  # X data
Y = tf.placeholder(tf.int32, [None, sequence_length])  # Y label

x_one_hot = tf.one_hot(X, num_classes)  # one hot: 1 -> 0 1 0 0 0 0 0 0 0 0
```

Example) ‘ hihello ‘ 학습

```python
idx2char = ['h', 'i', 'e', 'l', 'o'] # h=0, i=1, e=2, l=3, o=4
x_data = [[0, 1, 0, 2, 3, 3]]        # hihell
x_one_hot = [[[1, 0, 0, 0, 0],       # h 0
              [0, 1, 0, 0, 0],       # i 1
              [1, 0, 0, 0, 0],       # h 0
              [0, 0, 1, 0, 0],       # e 2
              [0, 0, 0, 1, 0],       # l 3
              [0, 0, 0, 1, 0]]]      # l 3

y_data = [[1, 0, 2, 3, 3, 4]]        # ihello
X = tf.placeholder(tf.float32,
        [None, sequence_length, input_dim]) # X one-hot
Y = tf.placeholder(tf.int32, [None, sequence_length])  # Y label
```

Review RNN : character sequence RNN

- 하이퍼파라미터 설정

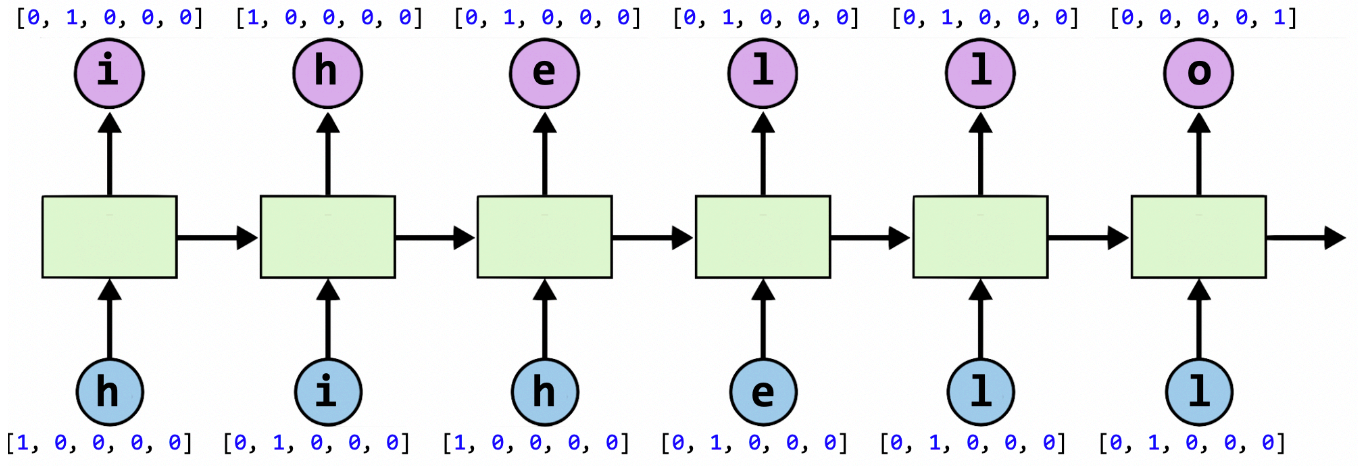


```python
# hyper parameters
dic_size = len(char2idx)  # RNN input size (one hot size)
hidden_size = len(char2idx)  # RNN output size
num_classes = len(char2idx)  # final output size (RNN or softmax, etc.)
batch_size = 1  # one sample data, one batch
sequence_length = len(sample) - 1  # number of lstm rollings (unit #)
learning_rate = 0.1
```

Example) ‘ hihello ‘ 학습

```
[1, 0, 0, 0, 0],    # h 0
[0, 1, 0, 0, 0],    # i 1
[0, 0, 1, 0, 0],    # e 2
[0, 0, 0, 1, 0],    # l 3
[0, 0, 0, 0, 1],    # o 4
```

Output size (hidden size) = 5

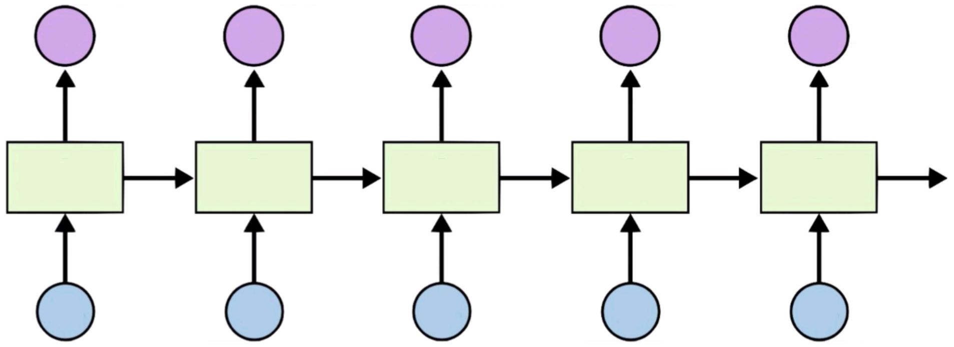


One-hot vector size(input dimension) = 5

Review RNN : Batching input

```
Hidden_size=2
sequence_length=5
batch_size=3
```

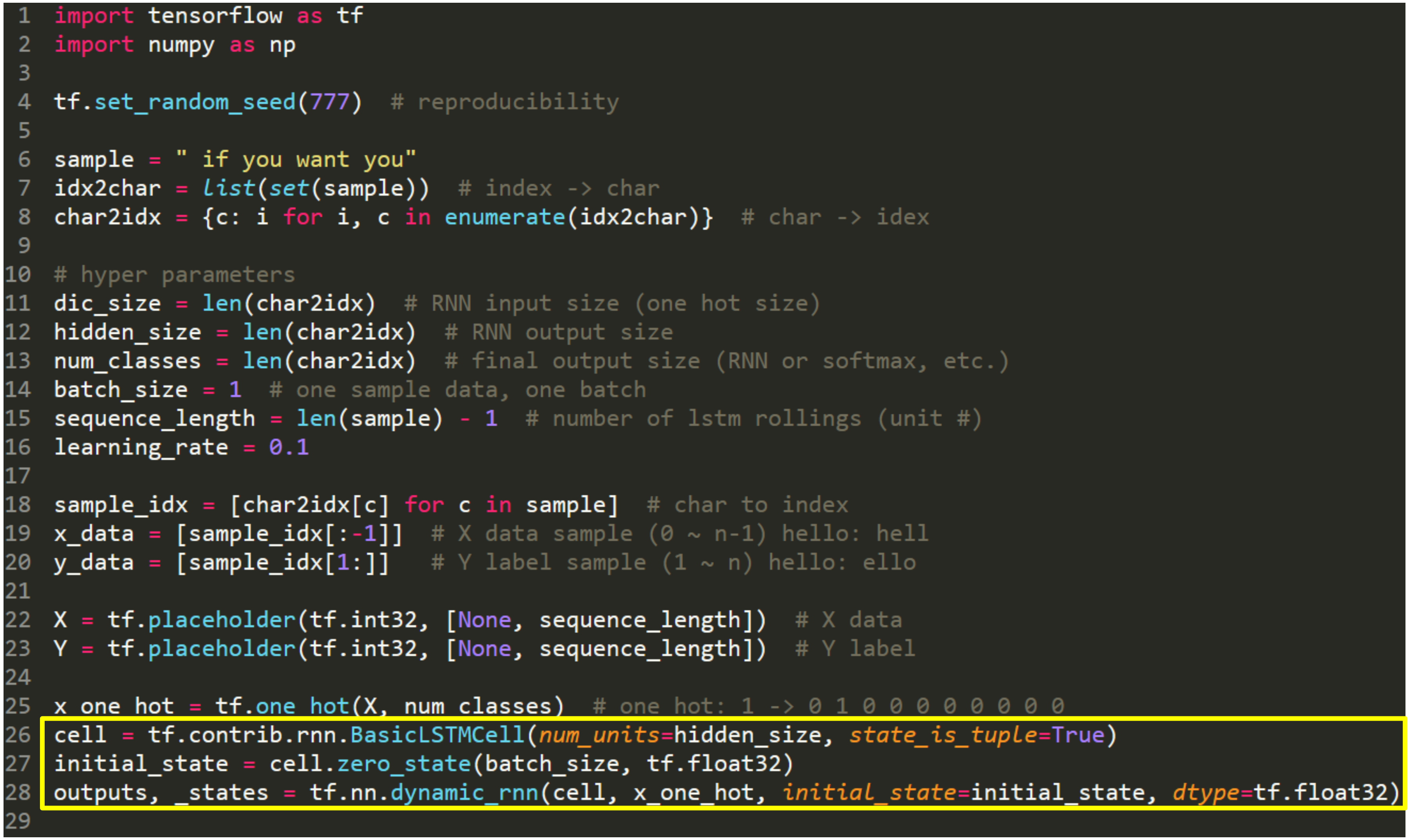
```
shape=(3,5,2): [[[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]],
                [[x,x], [x,x], [x,x], [x,x], [x,x]]]
```



```
shape=(3,5,4): [[[1,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,1,0], [0,0,0,1]], # hello
                [[0,1,0,0], [0,0,0,1], [0,0,1,0], [0,0,1,0], [0,0,1,0]]  # eolll
                [[0,0,1,0], [0,0,1,0], [0,1,0,0], [0,1,0,0], [0,0,1,0]]] # lleel
```

Review RNN : character sequence RNN

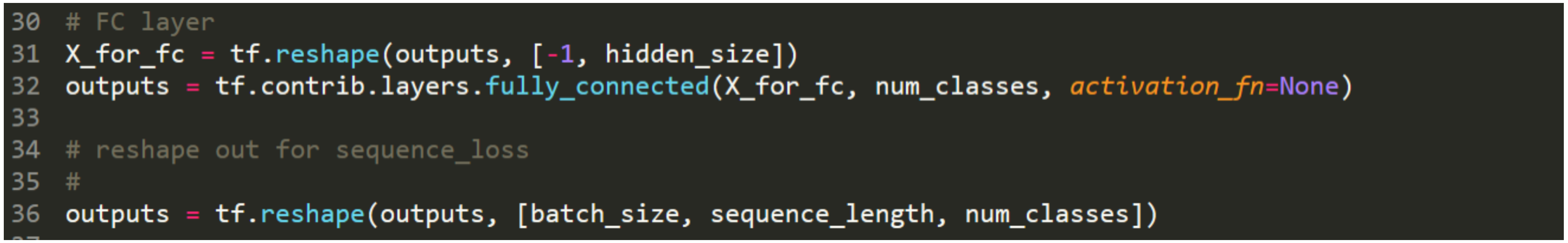
- 전체 코드(1)



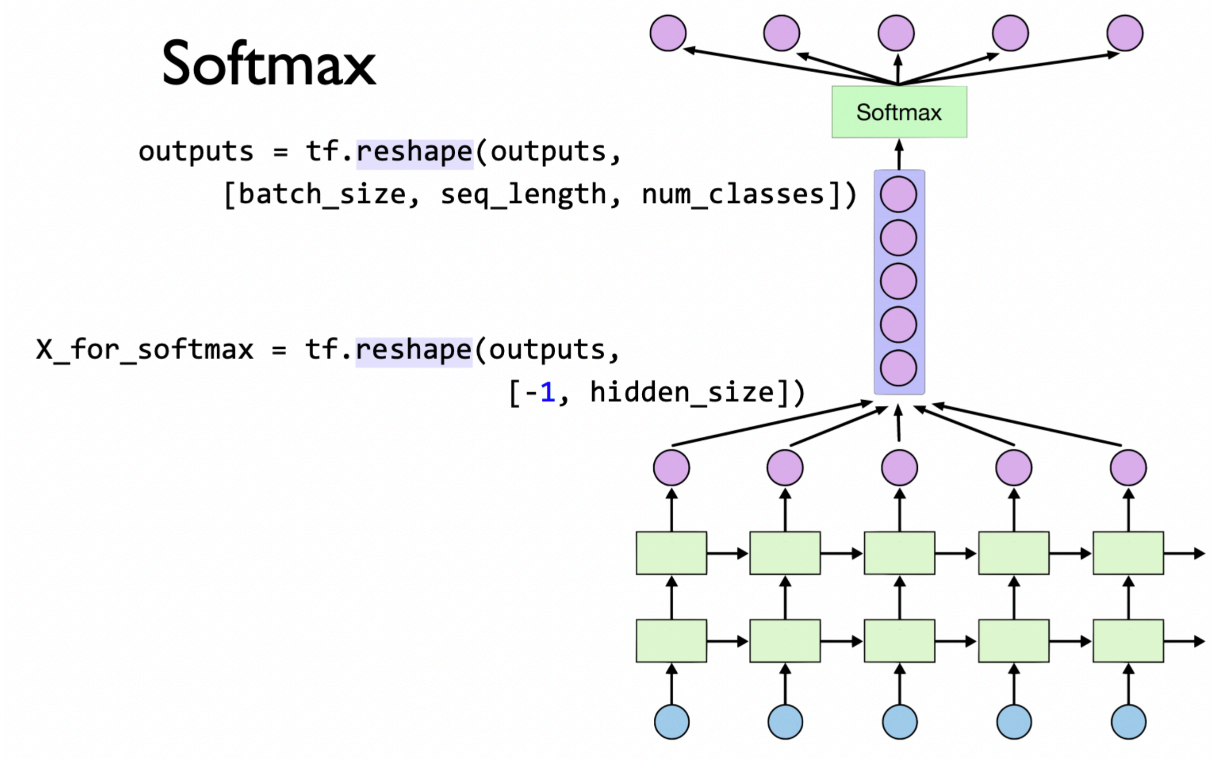
```python
import tensorflow as tf
import numpy as np

tf.set_random_seed(777)  # reproducibility

sample = " if you want you"
idx2char = list(set(sample))  # index -> char
char2idx = {c: i for i, c in enumerate(idx2char)}  # char -> idex

# hyper parameters
dic_size = len(char2idx)  # RNN input size (one hot size)
hidden_size = len(char2idx)  # RNN output size
num_classes = len(char2idx)  # final output size (RNN or softmax, etc.)
batch_size = 1  # one sample data, one batch
sequence_length = len(sample) - 1  # number of lstm rollings (unit #)
learning_rate = 0.1

sample_idx = [char2idx[c] for c in sample]  # char to index
x_data = [sample_idx[:-1]]  # X data sample (0 ~ n-1) hello: hell
y_data = [sample_idx[1:]]   # Y label sample (1 ~ n) hello: ello

X = tf.placeholder(tf.int32, [None, sequence_length])  # X data
Y = tf.placeholder(tf.int32, [None, sequence_length])  # Y label

x_one_hot = tf.one_hot(X, num_classes)  # one hot: 1 -> 0 1 0 0 0 0 0 0 0 0
cell = tf.contrib.rnn.BasicLSTMCell(num_units=hidden_size, state_is_tuple=True)
initial_state = cell.zero_state(batch_size, tf.float32)
outputs, _states = tf.nn.dynamic_rnn(cell, x_one_hot, initial_state=initial_state, dtype=tf.float32)
```

Review RNN : character sequence RNN

- Output layer



```
# FC layer
X_for_fc = tf.reshape(outputs, [-1, hidden_size])
outputs = tf.contrib.layers.fully_connected(X_for_fc, num_classes, activation_fn=None)

# reshape out for sequence_loss
#
outputs = tf.reshape(outputs, [batch_size, sequence_length, num_classes])
```



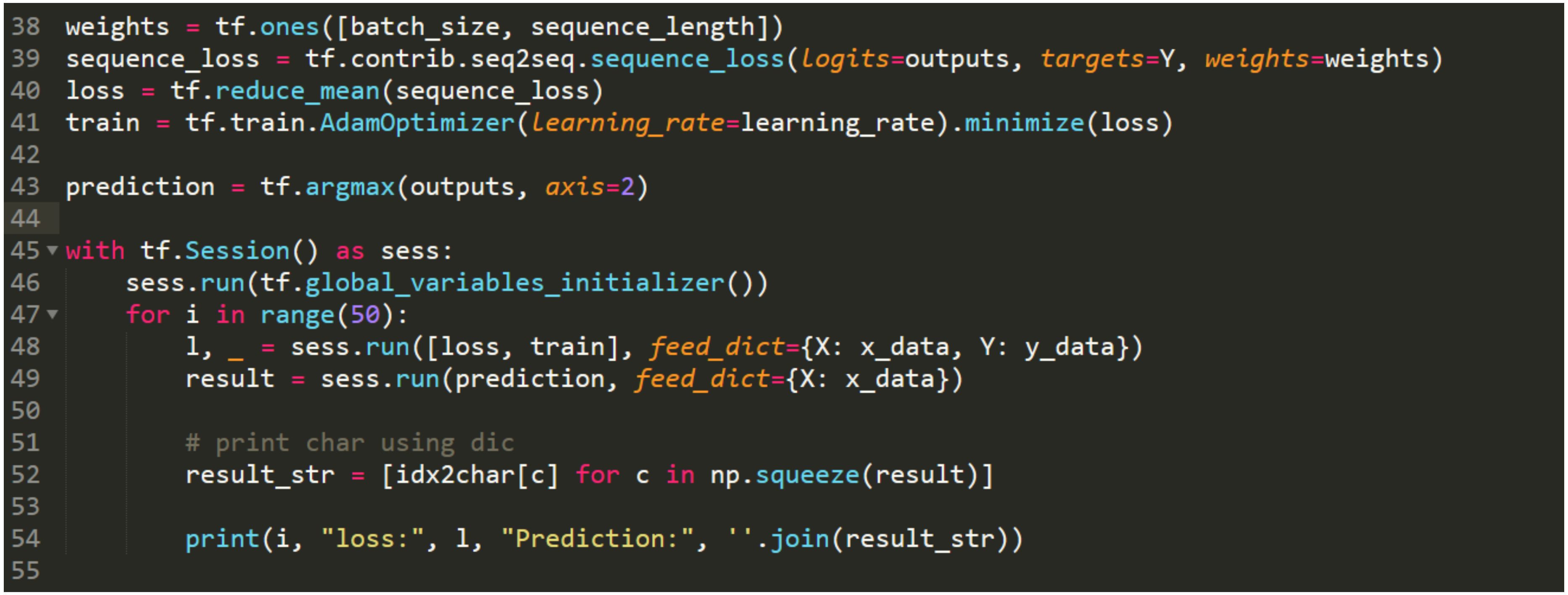
Softmax

```
outputs = tf.reshape(outputs,
     [batch_size, seq_length, num_classes])
```

```
X_for_softmax = tf.reshape(outputs,
                          [-1, hidden_size])
```

Review RNN : character sequence RNN

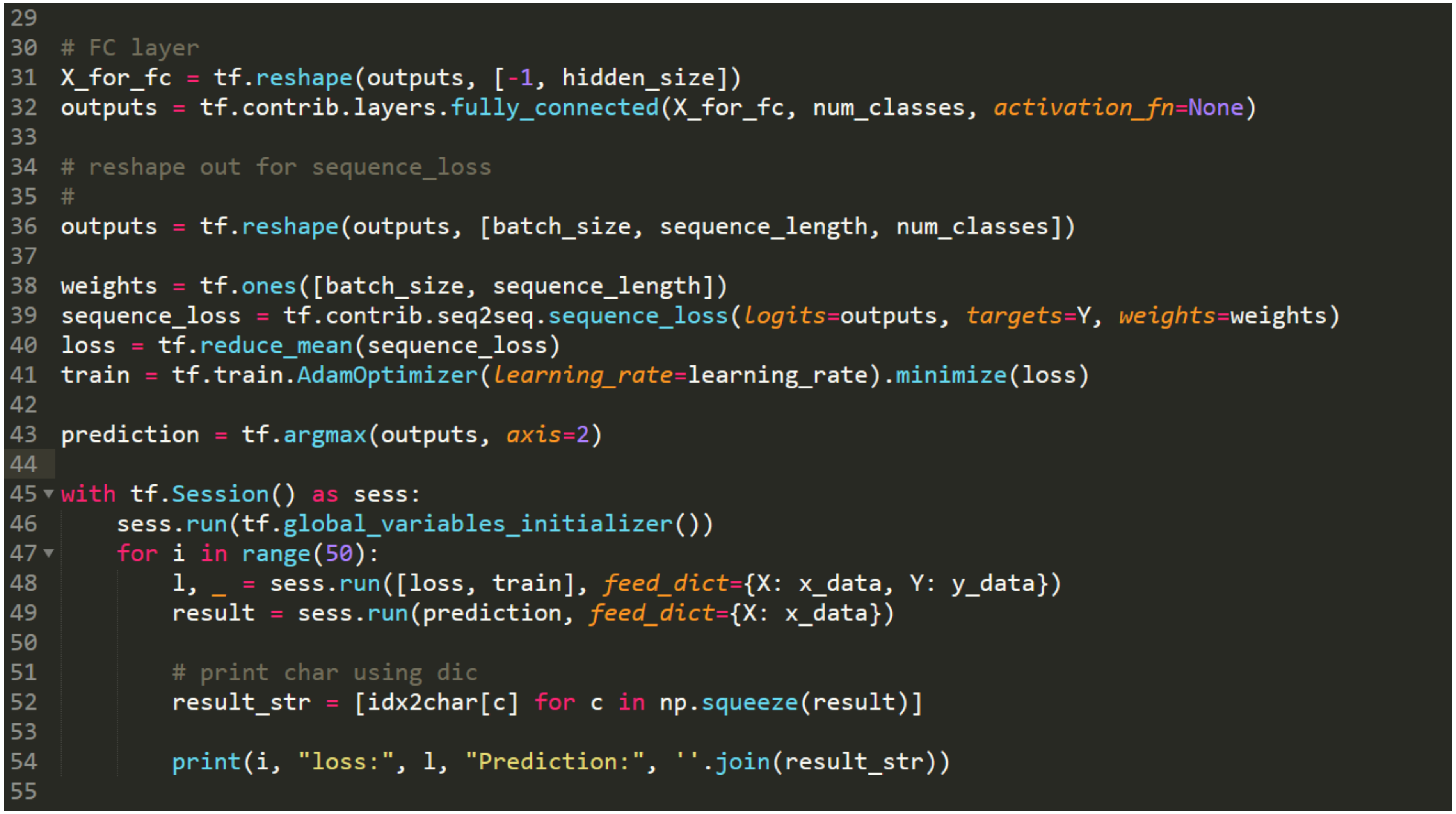
- Sequence loss 설정 및 학습



```python
weights = tf.ones([batch_size, sequence_length])
sequence_loss = tf.contrib.seq2seq.sequence_loss(logits=outputs, targets=Y, weights=weights)
loss = tf.reduce_mean(sequence_loss)
train = tf.train.AdamOptimizer(learning_rate=learning_rate).minimize(loss)

prediction = tf.argmax(outputs, axis=2)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(50):
        l, _ = sess.run([loss, train], feed_dict={X: x_data, Y: y_data})
        result = sess.run(prediction, feed_dict={X: x_data})

        # print char using dic
        result_str = [idx2char[c] for c in np.squeeze(result)]

        print(i, "loss:", l, "Prediction:", ''.join(result_str))

```

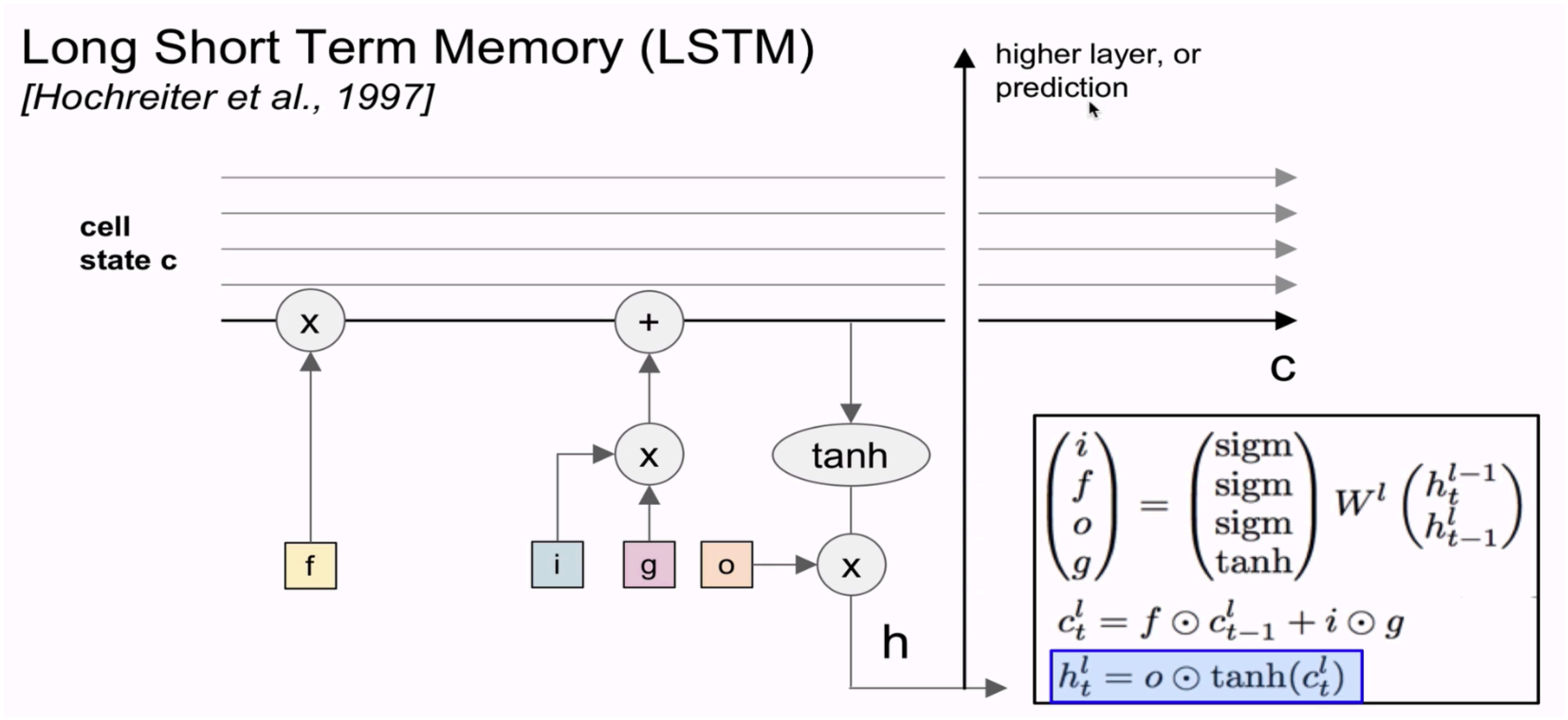
Review RNN : character sequence RNN

- 전체 코드(2)



```python

# FC layer
X_for_fc = tf.reshape(outputs, [-1, hidden_size])
outputs = tf.contrib.layers.fully_connected(X_for_fc, num_classes, activation_fn=None)

# reshape out for sequence_loss
#
outputs = tf.reshape(outputs, [batch_size, sequence_length, num_classes])

weights = tf.ones([batch_size, sequence_length])
sequence_loss = tf.contrib.seq2seq.sequence_loss(logits=outputs, targets=Y, weights=weights)
loss = tf.reduce_mean(sequence_loss)
train = tf.train.AdamOptimizer(learning_rate=learning_rate).minimize(loss)

prediction = tf.argmax(outputs, axis=2)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(50):
        l, _ = sess.run([loss, train], feed_dict={X: x_data, Y: y_data})
        result = sess.run(prediction, feed_dict={X: x_data})

        # print char using dic
        result_str = [idx2char[c] for c in np.squeeze(result)]

        print(i, "loss:", l, "Prediction:", ''.join(result_str))

```

LSTM


Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]

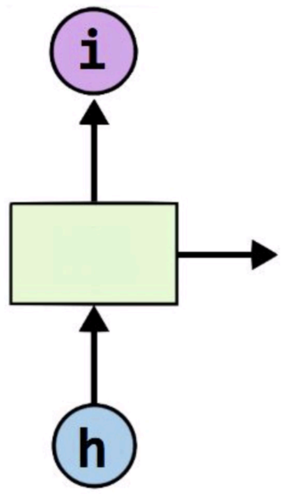
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

$$\boxed{h_t^l = o \odot \tanh(c_t^l)}$$

LSTM

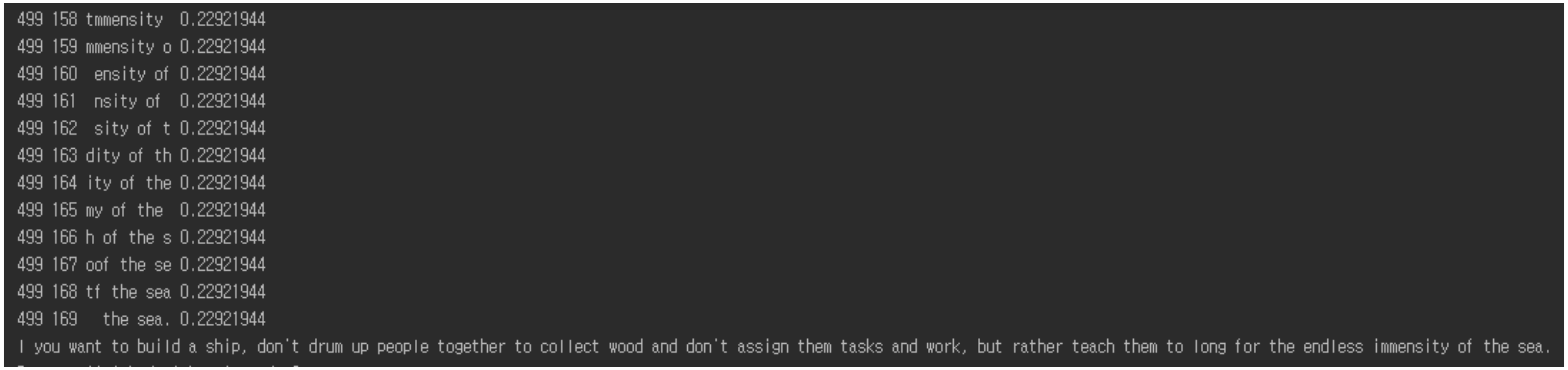
```
# RNN model
rnn_cell = rnn_cell.BasicRNNCell(rnn_size)


rnn_cell = rnn_cell. BasicLSTMCell(rnn_size)
rnn_cell = rnn_cell. GRUCell(rnn_size)
```



과제 : LSTM cell 생성하여 long sequence 학습 시키기

"if you want to build a ship, don't drum up people together to "
"collect wood and don't assign them tasks and work, but rather "
"teach them to long for the endless immensity of the sea."

과제 결과

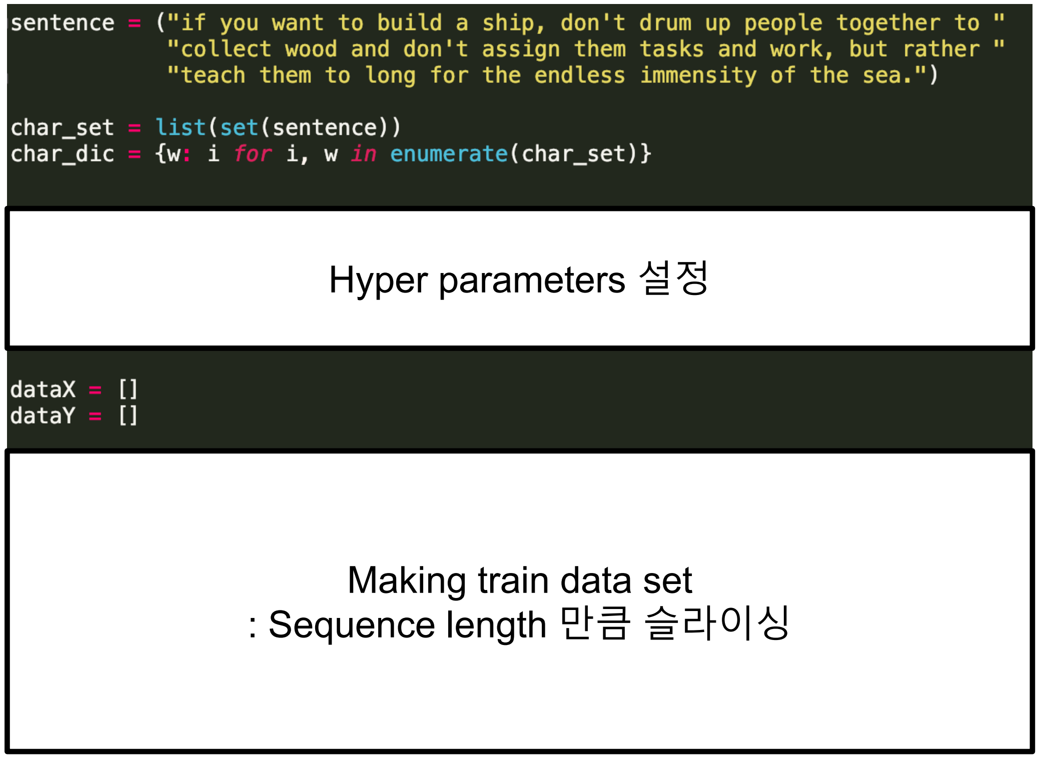


```
499 158 tmmensity  0.22921944
499 159 mmensity o 0.22921944
499 160  ensity of 0.22921944
499 161  nsity of  0.22921944
499 162  sity of t 0.22921944
499 163 dity of th 0.22921944
499 164 ity of the 0.22921944
499 165 my of the  0.22921944
499 166 h of the s 0.22921944
499 167 oof the se 0.22921944
499 168 tf the sea 0.22921944
499 169   the sea. 0.22921944
l you want to build a ship, don't drum up people together to collect wood and don't assign them tasks and work, but rather teach them to long for the endless immensity of the sea.
```

과제 : Making Train data set

```python
sentence = ("if you want to build a ship, don't drum up people together to "
            "collect wood and don't assign them tasks and work, but rather "
            "teach them to long for the endless immensity of the sea.")
```

```
# training dataset
0 if you wan -> f you want
1 f you want ->  you want
2  you want  -> you want t
3 you want t -> ou want to
…
168  of the se -> of the sea
169 of the sea -> f the sea.
```



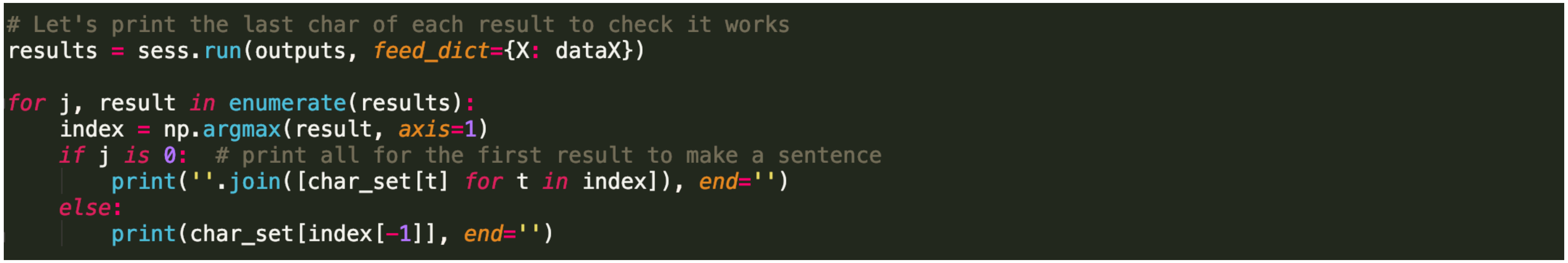
```python
sentence = ("if you want to build a ship, don't drum up people together to "
            "collect wood and don't assign them tasks and work, but rather "
            "teach them to long for the endless immensity of the sea.")

char_set = list(set(sentence))
char_dic = {w: i for i, w in enumerate(char_set)}
```

Hyper parameters 설정

```python
dataX = []
dataY = []
```

Making train data set
: Sequence length 만큼 슬라이싱

과제 : Printing all sentence



```
# Let's print the last char of each result to check it works
results = sess.run(outputs, feed_dict={X: dataX})

for j, result in enumerate(results):
    index = np.argmax(result, axis=1)
    if j is 0:  # print all for the first result to make a sentence
        print(''.join([char_set[t] for t in index]), end='')
    else:
        print(char_set[index[-1]], end='')
```

f you want to build a ship, don't drum up people together to collect wood and don't assign them tasks and work, but rather teach them to long for the endless immensity of the sea.

과제

- LSTM cell 생성하여 long sequence학습 시키기
- 소스와 결과 캡쳐 GitLab에 제출
- 과제 기한 : **다음주 수요일 23:59** 까지
- 수업시간에 한 경우 바로 검사받고 **GitLab**에 제출
- GitLab 관련 사용법은 첨부 파일 확인