Calculating Loss

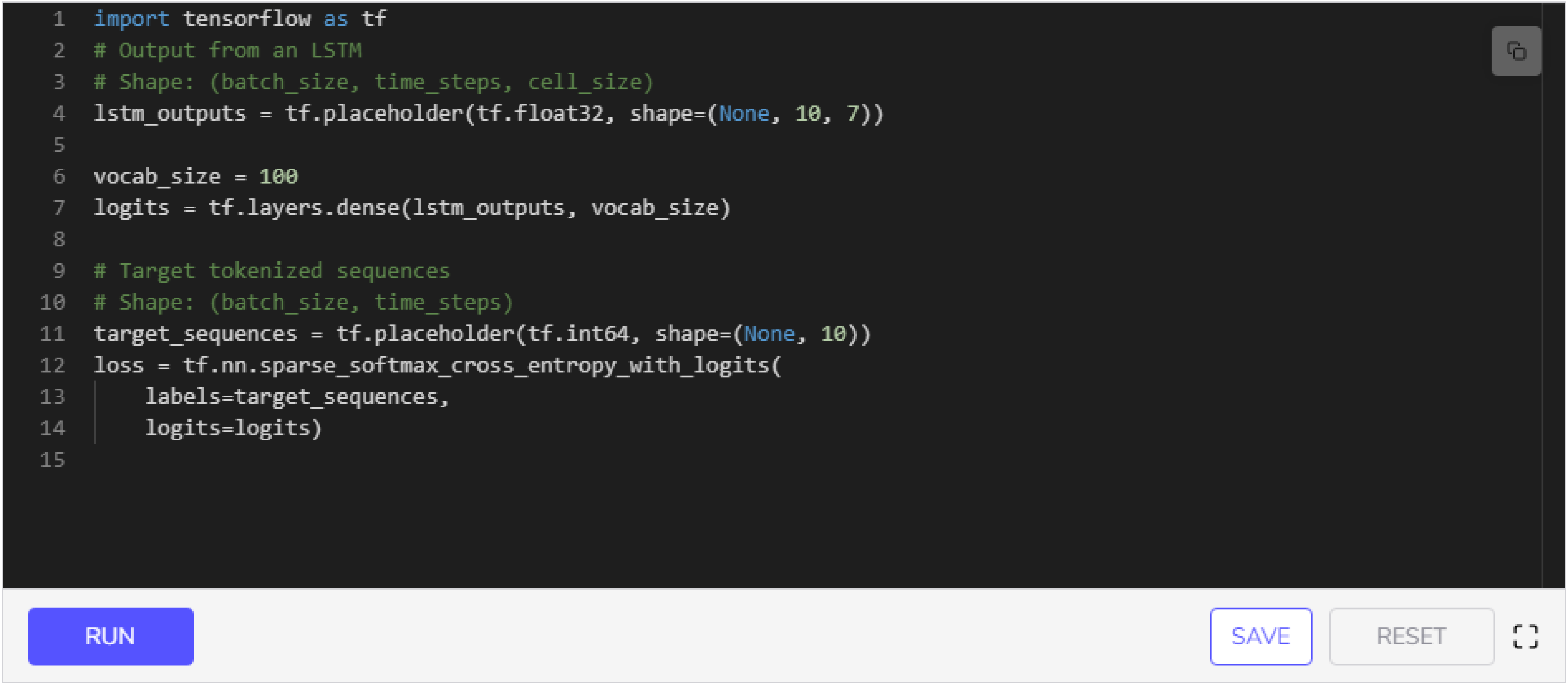
Calculate the loss for your LSTM model.

Chapter Goals:

- Convert your LSTM model's outputs into logits
- Use a padding mask to calculate the overall loss

A. Logits & loss

As mentioned in earlier chapters, the task for a language model is no different from regular multiclass classification. Therefore, the loss function will still be the regular softmax cross entropy loss. We use a final fully-connected layer to convert model outputs into logits for each of the possible classes (i.e. vocabulary words).



```python
import tensorflow as tf
# Output from an LSTM
# Shape: (batch_size, time_steps, cell_size)
lstm_outputs = tf.placeholder(tf.float32, shape=(None, 10, 7))

vocab_size = 100
logits = tf.layers.dense(lstm_outputs, vocab_size)

# Target tokenized sequences
# Shape: (batch_size, time_steps)
target_sequences = tf.placeholder(tf.int64, shape=(None, 10))
loss = tf.nn.sparse_softmax_cross_entropy_with_logits(
    labels=target_sequences,
    logits=logits)
```

Obtaining the loss for an LSTM model (with max sequence length of 5).

The function used to calculate the softmax cross entropy loss for feed-forward neural networks is `tf.nn.softmax_cross_entropy_with_logits`. However, we can only use this function if the `labels` and `logits` arguments both have the same shape.

In our example, `logits` has 3 dimensions while `labels` (`target_sequences`) only has 2. In this case, the `labels` are referred to as *sparse* (i.e. they represent class indexes rather than one-hot vectors), so we use the sparse version of the loss function.

B. Padding mask

When we calculate the loss based on the model's outputs, we don't want to include the logits for every time step in each sequence. Specifically, we want to exclude the loss calculated for the padded time steps, since those values are meaningless. Therefore, we use a *padding mask* to zero-out the loss at padded time steps.

The padding mask will have the same shape as the labels (i.e. target batch), but it will only contain 0's and 1's. Locations containing 0 represent padded time steps while locations containing 1 represent actual input sequence tokens. We multiply the padding mask by the loss to zero-out the padded time step locations.

The code below demonstrates an example usage of a padding mask, with batch size of 1 and max sequence length of 5. Note that we cast the padding mask to `tf.float32` so that it matches the type of the loss.

```python
import tensorflow as tf
# loss: Softmax loss for LSTM
with tf.Session() as sess:
    print(repr(sess.run(loss)))

# Same shape as loss
pad_mask = tf.constant([
    [1., 1., 1., 1., 0.],
    [1., 1., 0., 0., 0.]
])

new_loss = loss * pad_mask
with tf.Session() as sess:
    print(repr(sess.run(new_loss)))
```