Tensor Indexing

Use tensor indexing to retrieve the model's final word prediction.

Chapter Goals:

- Extract the word predictions for the final time step of each sequence

A. Gather functions

When generating word predictions for a batch of sequences, we only want to retrieve the word IDs for each sequence's final time step. If this were regular Python (or NumPy), we could do this through simple list indexing or slicing. However, in TensorFlow we need to use *gather* functions to retrieve data at specific locations in the tensor.

There are two main gather functions: `tf.gather` and `tf.gather_nd`. Both take in the same two required arguments:

- **params:** The tensor that we wish to retrieve data from.
- **indices:** The locations in the tensor that we will index into.

The `tf.gather` function can be used to retrieve specific slices from a tensor, based on what the `axis` keyword argument is set to (default is `0`).

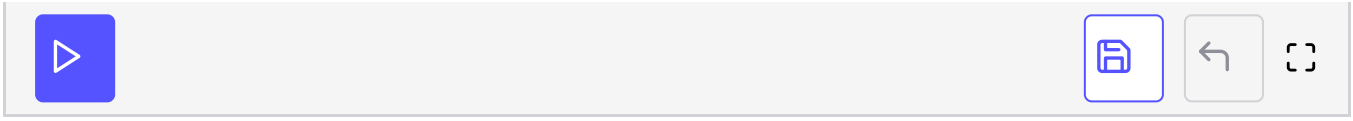
Below are example usages of `tf.gather`. Note that the values in the `indices` argument must by valid indices for the specified `axis`.

```
import tensorflow as tf
t1 = tf.constant([1, 2, 3])
with tf.Session() as sess:
    print(repr(sess.run(tf.gather(t1, 0))))
    print(repr(sess.run(tf.gather(t1, 2))))

print('\n')
t2 = tf.constant([[1, 2, 3], [4, 5, 6]])
with tf.Session() as sess:
    print(repr(sess.run(tf.gather(t2, 0))))
    print(repr(sess.run(tf.gather(t2, 1, axis=1))))
    print(repr(sess.run(tf.gather(t2, [0, 2], axis=1))))
```

```
print('\n')
t3 = tf.constant([
    [[1, 2, 3], [4, 5, 6]],

    [[5, 6, 7], [7, 8, 9]]
])
with tf.Session() as sess:
    print(repr(sess.run(tf.gather(t3, 0))))
    print(repr(sess.run(tf.gather(t3, 1, axis=1))))
    print(repr(sess.run(tf.gather(t3, [0, 2], axis=2))))
```

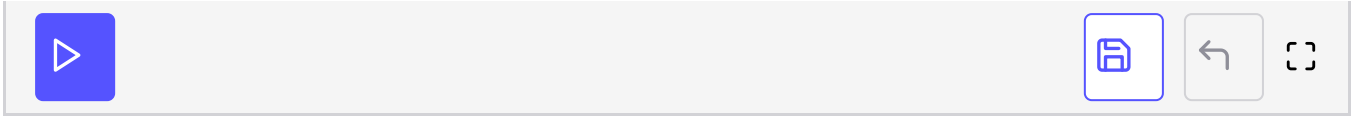


While `tf.gather` retrieves slices from the input tensor, we can use `tf.gather_nd` for specific tensor indexing. Since `tf.gather_nd` is used for specific indexing rather than slicing, there's no `axis` keyword argument.

The code below shows example usages of `tf.gather_nd`. Note that the tensors `t2` and `t3` come from the previous example.

```
with tf.Session() as sess:
    print(repr(sess.run(tf.gather_nd(t2, [0, 1]))))
    print(repr(sess.run(tf.gather_nd(t2, [[0, 1], [1, 1]]))))

print('\n')
with tf.Session() as sess:
    print(repr(sess.run(tf.gather_nd(t3, [0, 1]))))
    print(repr(sess.run(tf.gather_nd(t3, [[0, 0], [1, 1]]))))
    print(repr(sess.run(tf.gather_nd(t3, [0, 1, 2]))))
```



When using `tf.gather_nd`, the `params` argument must be a multi-dimensional tensor (cannot be 1-D), and the `indices` argument cannot be a single integer.

Time to Code!

In this chapter you'll be completing the `get_word_predictions` function, which uses tensor indexing to get the final word ID prediction from the full sequence of predicted IDs (`word_preds`).

To retrieve the final time step word predictions from each sequence, we need to create a 2-D matrix with shape `(batch_size, 2)`. Each row of this matrix corresponds to *(x, y)* pair, representing the final time step of a sequence in the batch.

The first column of the 2-D matrix is a range from `0` to `batch_size - 1` (inclusive), corresponding to the row indices of each sequence in the batch. We can use the `tf.range` function to create this range (TensorFlow's version of Python `range`).

Set `row_indices` equal to `tf.range` applied with `batch_size` as the only argument.

The second column of the 2-D matrix contains the final time steps of each sequence. This is equivalent to the length of each sequence subtracted by 1.

Call `tf.reduce_sum` with `binary_sequences` as the required argument and `axis=1` as the keyword argument. Then set `final_indexes` equal to the output of `tf.reduce_sum`, subtracted by 1.

Now that we created both columns of the 2-D matrix, we can put the columns together. Since the list `[row_indices, final_indexes]` corresponds to a matrix with shape `(2, batch_size)`, we can transpose it to get the matrix we want.

Set `gather_indices` equal to `tf.transpose` applied to the specified list.

Using `gather_indices` and the N-dimensional indexing function, `tf.gather_nd`, we can retrieve the final time step word predictions from `word_preds`, which will be the output of the function.

Set `final_id_predictions` equal to `tf.gather_nd` with `word_preds` and `gather_indices` as the two arguments. Then return `final_id_predictions`.

```
import tensorflow as tf

# LSTM Language Model
class LanguageModel(object):
    # Model Initialization
    def __init__(self, vocab_size, max_length, num_lstm_units, num_lstm_layers):
        self.vocab_size = vocab_size
        self.max_length = max_length
        self.num_lstm_units = num_lstm_units
        self.num_lstm_layers = num_lstm_layers
        self.tokenizer = tf.keras.preprocessing.text.Tokenizer(num_words=vocab_size)

    # Predict next word ID
    def get_word_predictions(self, word_preds, binary_sequences, batch_size):
        # CODE HERE
        pass
```