Embedding Matrix

Create a trainable embedding matrix to calculate word embeddings.

Chapter Goals:

- Learn how the embedding matrix is initialized during training
- Create a function for the forward run of embedding training

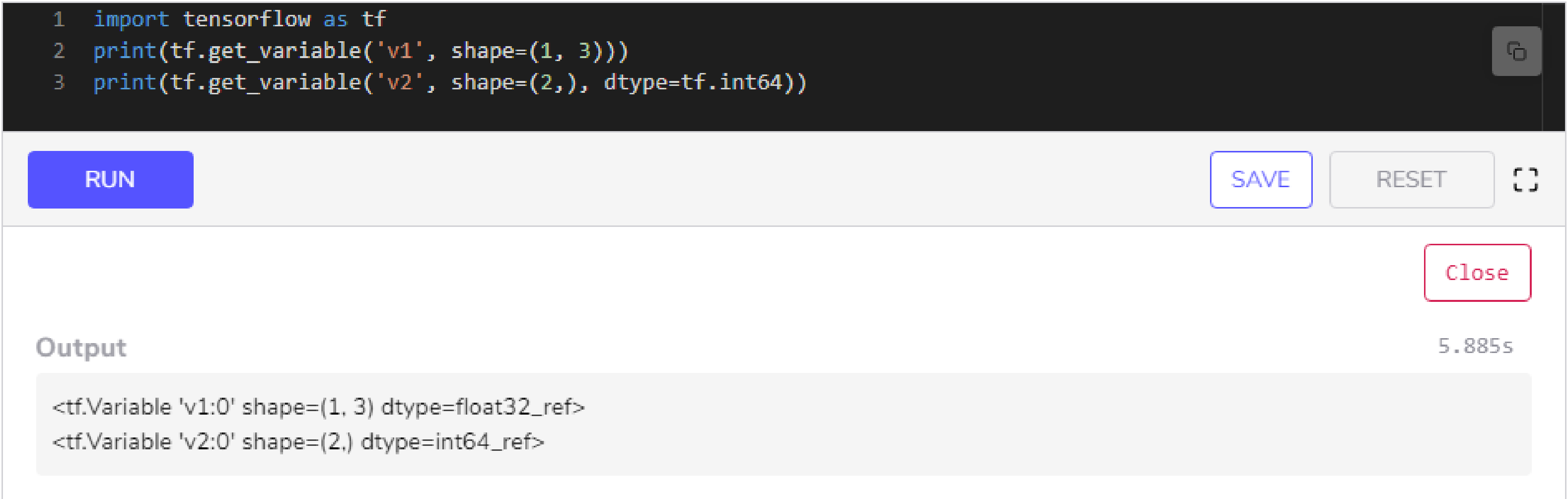
A. Variable creation

In TensorFlow, we use the `tf.layers.dense` function to create each layer of a fully-connected neural network. This is a function provided by TensorFlow that automatically handles all the setup required for the neural network's weights. So previously all we needed to do was initialize the weight variables (via `tf.global_variables_initializer`) prior to running the computation graph.

However, for our embedding model, the variable we need to create is a 2-D matrix that contains the embedding vectors for each vocabulary word ID. Therefore, we need to manually create this variable using the `tf.get_variable` function.

The function only takes in one required argument, which is the name of the variable. Of the possible keyword arguments, a couple important ones to know are `shape` and `dtype`, which allow us to specify the shape and type, respectively, of the created variable.

Below we demonstrate example usages of `tf.get_variable`:



```python
import tensorflow as tf
print(tf.get_variable('v1', shape=(1, 3)))
print(tf.get_variable('v2', shape=(2,), dtype=tf.int64))
```

```
<tf.Variable 'v1:0' shape=(1, 3) dtype=float32_ref>
<tf.Variable 'v2:0' shape=(2,) dtype=int64_ref>
```

In the example above, the first variable is created with just the `shape` keyword argument (so its default type is `tf.float32`), while the second is manually set to type `tf.int64` (the `_ref` suffix does not affect the type).

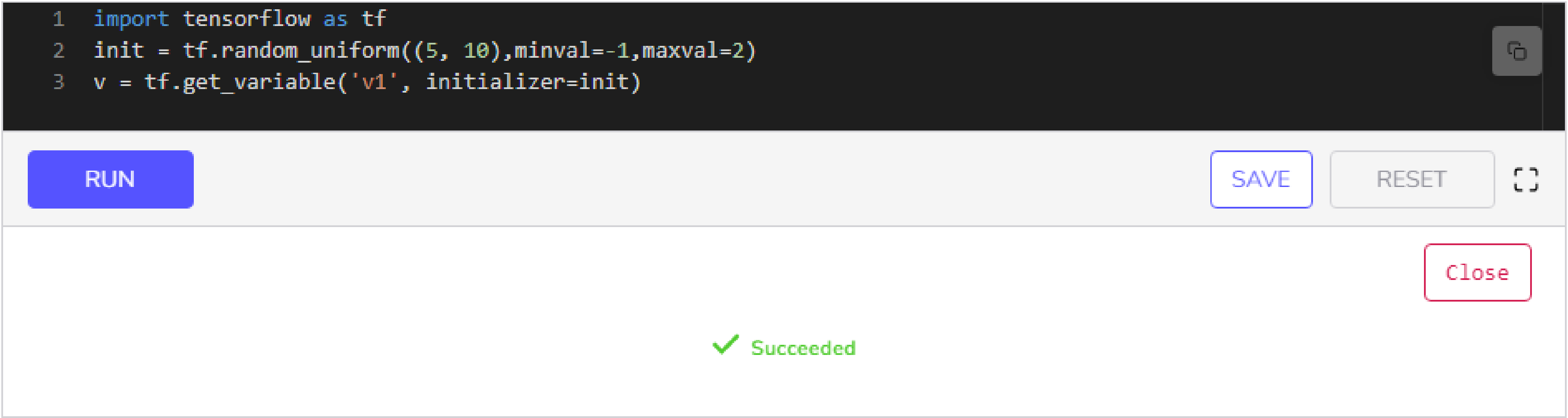
During training, the first call to `tf.get_variable` with a given variable name will create a new variable. Each subsequent call to the function using the same variable name will retrieve the existing variable, rather than create a new one. This allows us to continue training with the same embedding matrix.

B. Variable initializers

Another useful keyword argument to know for `tf.get_variable` is the `initializer` argument, which is used to specify how the variable will be initialized. The argument can also be used to specify the variable's shape (in which case we don't need to set the `shape` keyword argument).

The default initializer for variables created by `tf.get_variable` is a `glorot_uniform_initializer`. However, we can use our own probability distribution to randomly initialize the variable. A popular probability distribution to use is `tf.random_uniform`, corresponding to a uniform distribution.

Below we demonstrate an example usage of `tf.random_uniform` to initialize a variable:



```python
import tensorflow as tf
init = tf.random_uniform((5, 10),minval=-1,maxval=2)
v = tf.get_variable('v1', initializer=init)
```

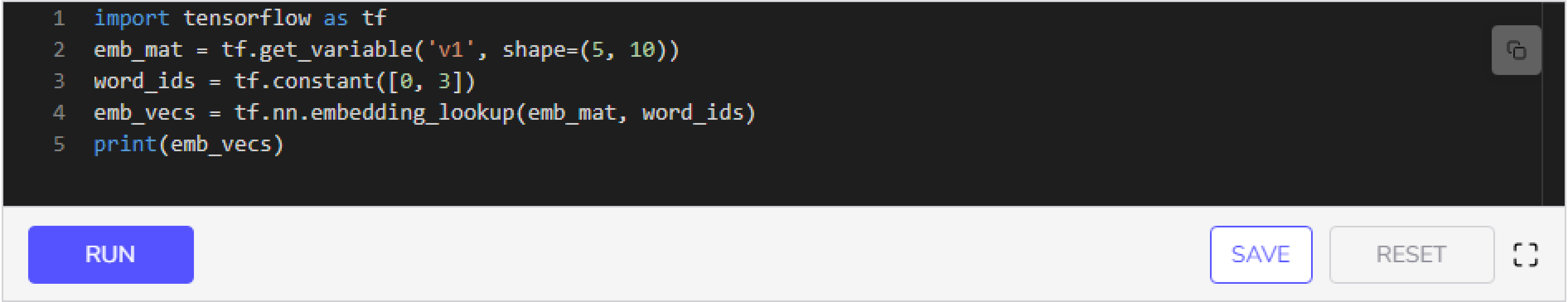
In the example above, we used `tf.random_uniform` to return a tensor with shape `(5, 10)`, containing randomly chosen values from the uniform distribution in the range [-1, 2]. We then used `init` to initialize `v` with its randomly chosen values.

C. Embedding lookup

When training the embedding model, the "forward" run consists of variable initialization/retrieval followed by *embedding lookup* for the current iteration's training batch. Embedding lookup refers to retrieving the embedding vectors for each word ID in the training batch. Since the embedding matrix's rows are each unique embedding vectors, we perform the lookup simply by retrieving the rows corresponding to the training batch's word IDs.

The function we use to retrieve the embedding vectors is `tf.nn.embedding_lookup`. It takes in two required arguments, which are the embedding matrix variable and vocabulary IDs to lookup.

Below we demonstrate an example usage of `tf.nn.embedding_lookup`:



```python
import tensorflow as tf
emb_mat = tf.get_variable('v1', shape=(5, 10))
word_ids = tf.constant([0, 3])
emb_vecs = tf.nn.embedding_lookup(emb_mat, word_ids)
print(emb_vecs)
```

In the example above, we used `tf.nn.embedding_lookup` to retrieve embedding vectors from the embedding matrix, `emb_mat`, for the word IDs `0` and `3`. The output tensor from the embedding lookup will contain the embedding vector for ID `0` in the first row, and the embedding vector for ID `3` in the second row.