Depth

Increase the SqueezeNet model's depth for added performance.

Chapter Goals:

- Add depth to the model through additional fire modules

A. Deeper fire modules

In order to extract more distinguishing features from the data, we add more fire modules to our model. The additional fire modules will use twice as many filters in the expand layer (from 64 to 128). This follows the same approach outlined in the **CNN** section, where we increase the number of filters used in the convolution layers deeper into the model.

To avoid overfitting, we apply dropout after our second multi-fire module block, with a dropout rate of 0.5. There is no max pooling layer after this multi-fire module block, since we'll obtain logits in the next chapter using average pooling.

The code below shows the `model_layers` function with added depth:

```python
import tensorflow as tf

class SqueezeNetModel(object):
    # __init__ and other functions omitted

    # Model Layers
    def model_layers(self, inputs, is_training):
        conv1 = self.custom_conv2d(
            inputs,
            64,
            [3, 3],
            'conv1')
        pool1 = self.custom_max_pooling2d(
            conv1,
            'pool1')
        fire_params1 = [
            (32, 64, 'fire1'),
            (32, 64, 'fire2')
        ]
        multi_fire1 = self.multi_fire_module(
            pool1
```

```python
            pool1,
            fire_params1)
        pool2 = self.custom_max_pooling2d(
            multi_fire1,
            'pool2')
        fire_params2 = [
            (32, 128, 'fire3'),
            (32, 128, 'fire4')
        ]
        multi_fire2 = self.multi_fire_module(
            pool2,
            fire_params2)
        dropout1 = tf.layers.dropout(multi_fire2, rate=0.5,
            training=is_training)
```