Saving model architecture only

November 19, 2020

1 Saving model architecture only

In this reading you will learn how to save a model's architecture, but not its weights.

```
In [1]: import tensorflow as tf
        from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Dense
        import json
        import numpy as np
```

In previous videos and notebooks you have have learned how to save a model's weights, as well as the entire model - weights and architecture.

1.0.1 Accessing a model's configuration

A model's *configuration* refers to its architecture. TensorFlow has a convenient way to retrieve a model's architecture as a dictionary. We start by creating a simple fully connected feedforward neural network with 1 hidden layer.

```
In [2]: # Build the model

        model = Sequential([
            Dense(units=32, input_shape=(32, 32, 3), activation='relu', name='dense_1'),
            Dense(units=10, activation='softmax', name='dense_2')
        ])
```

A TensorFlow model has an inbuilt method `get_config` which returns the model's architecture as a dictionary:

```
In [3]: # Get the model config

        config_dict = model.get_config()
        print(config_dict)
```

```
{'name': 'sequential', 'layers': [{'class_name': 'Dense', 'config': {'name': 'dense_1', 'traina
```

1.0.2 Creating a new model from the config

A new TensorFlow model can be created from this config dictionary. This model will have reinitialized weights, which are not the same as the original model.

```
In [4]: # Create a model from the config dictionary

        model_same_config = tf.keras.Sequential.from_config(config_dict)
```

We can check explicitly that the config of both models is the same, but the weights are not:

```
In [5]: # Check the new model is the same architecture

        print('Same config:',
              model.get_config() == model_same_config.get_config())
        print('Same value for first weight matrix:',
              np.allclose(model.weights[0].numpy(), model_same_config.weights[0].numpy()))
```

```
Same config: True
Same value for first weight matrix: False
```

For models that are not `Sequential` models, use `tf.keras.Model.from_config` instead of `tf.keras.Sequential.from_config`.

1.0.3 Other file formats: JSON and YAML

It is also possible to obtain a model's config in JSON or YAML formats. This follows the same pattern:

```
In [6]: # Convert the model to JSON

        json_string = model.to_json()
        print(json_string)
```

```
{"class_name": "Sequential", "config": {"name": "sequential", "layers": [{"class_name": "Dense
```

The JSON format can easily be written out and saved as a file:

```
In [7]: # Write out JSON config file

        with open('config.json', 'w') as f:
            json.dump(json_string, f)
        del json_string
```

```
In [8]: # Read in JSON config file again

        with open('config.json', 'r') as f:
            json_string = json.load(f)
```

```
In [9]: # Reinitialize the model

        model_same_config = tf.keras.models.model_from_json(json_string)

In [10]: # Check the new model is the same architecture, but different weights

         print('Same config:',
               model.get_config() == model_same_config.get_config())
         print('Same value for first weight matrix:',
               np.allclose(model.weights[0].numpy(), model_same_config.weights[0].numpy()))
```

```
Same config: True
Same value for first weight matrix: False
```

The YAML format is similar. The details of writing out YAML files, loading them and using them to create a new model are similar as for the JSON files, so we won't show it here.

```
In [11]: # Convert the model to YAML

         yaml_string = model.to_yaml()
         print(yaml_string)
```

```
backend: tensorflow
class_name: Sequential
config:
  layers:
  - class_name: Dense
    config:
      activation: relu
      activity_regularizer: null
      batch_input_shape: !!python/tuple
      - null
      - 32
      - 32
      - 3
      bias_constraint: null
      bias_initializer:
        class_name: Zeros
        config: {}
      bias_regularizer: null
      dtype: float32
      kernel_constraint: null
      kernel_initializer:
        class_name: GlorotUniform
        config:
          seed: null
      kernel_regularizer: null
      name: dense_1
```

```
      trainable: true
      units: 32
      use_bias: true
  - class_name: Dense
    config:
      activation: softmax
      activity_regularizer: null
      bias_constraint: null
      bias_initializer:
        class_name: Zeros
        config: {}
      bias_regularizer: null
      dtype: float32
      kernel_constraint: null
      kernel_initializer:
        class_name: GlorotUniform
        config:
          seed: null
      kernel_regularizer: null
      name: dense_2
      trainable: true
      units: 10
      use_bias: true
  name: sequential
keras_version: 2.2.4-tf
```

Writing out, reading in and using YAML files to create models is similar to JSON files.

1.0.4 Further reading and resources

- https://www.tensorflow.org/guide/keras/save_and_serialize#architecture-only_saving
- https://keras.io/getting-started/faq/#how-can-i-save-a-keras-model