Dense

07 November 2024 15:40

Dense Class Documentation

Overview

The Dense class represents a fully connected (dense) layer in a neural network, commonly used in feedforward neural networks. It contains essential functionalities for setting up, training, and interacting with a dense layer in a neural network model.

Attributes

- **name** (str):
  The name of the class for identification. Default is 'optilearn.nn.Dense'.
- **n_neurons** (int):
  The number of neurons in the layer. This is the output dimension of the dense layer.
- **input_dim** (int or None):
  The number of input features to the layer. This is required only for the first layer in the network. If None, it means the layer is not the first one in the network.
- **activation** (str):
  The activation function applied to the output of the layer. Supported options are:
  - 'linear': No activation, linear output.
  - 'sigmoid': Sigmoid activation function.
  - 'tanh': Hyperbolic tangent activation function.
  - 'softmax': Softmax activation for multi-class classification problems.
- **trainable** (bool):
  Indicates whether the layer's parameters (weights and biases) are trainable during the network's training. Default is True.
- **weights** (numpy.ndarray):
  The weights matrix, initialized based on the specified activation function and input/output dimensions.
- **biases** (numpy.ndarray):
  The biases vector, initialized to zeros, one per neuron.

Parameters:

- **n_neurons** (int): The number of neurons in the dense layer.
- **input_dim** (int, optional): The number of input features. Required for the first layer in the network.
- **activation** (str, optional): The activation function. Can be 'linear', 'sigmoid', 'tanh', or 'softmax'. Default is 'linear'.
- **trainable** (bool, optional): Whether the weights and biases are trainable. Default is True.

Example:

```
dense_layer = Dense(n_neurons=64, input_dim=128, activation='relu', trainable=True)
```

info(self)

Prints the current weights and biases of the layer.

Example:

```
dense_layer.info()
```

layer(self)

Returns the current instance of the Dense layer.

Example:

```
dense_layer_instance = dense_layer.layer()
```

get_params(self)

Returns a tuple of the current weights and biases of the layer.
Returns:

- A tuple (weights, biases) of the layer's parameters.

Example:

```
weights, biases = dense_layer.get_params()
```

set_params(self, weights, biases)

Sets the weights and biases of the layer to the provided values.
Parameters:

- **weights** (numpy.ndarray): The new weights matrix.
- **biases** (numpy.ndarray): The new biases vector.

Example:

```
dense_layer.set_params(weights=new_weights, biases=new_biases)
```

Usage Example

```
# Create a Dense layer with 64 neurons, 128 input features, ReLU activation, and trainable
dense_layer = Dense(n_neurons=64, input_dim=128, activation='relu', trainable=True)

# Get layer information (weights and biases)
dense_layer.info()

# Set new parameters (weights and biases)
dense_layer.set_params(weights=new_weights, biases=new_biases)

# Retrieve current parameters
weights, biases = dense_layer.get_params()

# Return the layer instance
layer_instance = dense_layer.layer()
```