Embedding

[source]

```
keras.layers.embeddings.Embedding(input_dim, output_dim, init='uniform', input_length=None, W_r
```

Turn positive integers (indexes) into dense vectors of fixed size. eg. [[4], [20]] -> [[0.25, 0.1], [0.6, -0.2]]

This layer can only be used as the first layer in a model.

Example

```
  model = Sequential()
  model.add(Embedding(1000, 64, input_length=10))
  # the model will take as input an integer matrix of size (batch, input_length).
  # the largest integer (i.e. word index) in the input should be no larger than 999 (vocabulary
  # now model.output_shape == (None, 10, 64), where None is the batch dimension.

  input_array = np.random.randint(1000, size=(32, 10))

  model.compile('rmsprop', 'mse')
  output_array = model.predict(input_array)
  assert output_array.shape == (32, 10, 64)
```

Arguments

- **input_dim**: int > 0. Size of the vocabulary, ie. 1 + maximum integer index occurring in the input data.
- **output_dim**: int >= 0. Dimension of the dense embedding.
- **init**: name of initialization function for the weights of the layer (see: initializations), or alternatively, Theano function to use for weights initialization. This parameter is only relevant if you don't pass a `weights` argument.
- **weights**: list of Numpy arrays to set as initial weights. The list should have 1 element, of shape `(input_dim, output_dim)`.
- **W_regularizer**: instance of the regularizers module (eg. L1 or L2 regularization), applied to the embedding matrix.
- **W_constraint**: instance of the constraints module (eg. maxnorm, nonneg), applied to the embedding matrix.
- **mask_zero**: Whether or not the input value 0 is a special "padding" value that should be masked out. This is useful for recurrent layers which may take variable length input. If this is `True` then all subsequent layers in the model need to support masking or an exception will be raised. If mask_zero is set to True, as a consequence, index 0 cannot be used in the vocabulary (input_dim should equal |vocabulary| + 2).
- **input_length**: Length of input sequences, when it is constant. This argument is required if you are going to

connect `Flatten` then `Dense` layers upstream (without it, the shape of the dense outputs cannot be computed).

- **dropout**: float between 0 and 1. Fraction of the embeddings to drop.

Input shape

2D tensor with shape: `(nb_samples, sequence_length)`.

Output shape

3D tensor with shape: `(nb_samples, sequence_length, output_dim)`.

References

- A Theoretically Grounded Application of Dropout in Recurrent Neural Networks