Embedding [source]

```
keras.layers.Embedding(input_dim, output_dim, embeddings_initializer='uniform', embeddings_regu
```

Turns 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 s
# 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, i.e. maximum integer index + 1.
- **output_dim**: int >= 0. Dimension of the dense embedding.
- **embeddings_initializer**: Initializer for the `embeddings` matrix (see initializers).
- **embeddings_regularizer**: Regularizer function applied to the `embeddings` matrix (see regularizer).
- **embeddings_constraint**: Constraint function applied to the `embeddings` matrix (see constraints).
- **mask_zero**: Whether or not the input value 0 is a special "padding" value that should be masked out. This is useful when using 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 size of vocabulary + 1).
- **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).

Input shape

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

Output shape

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

References

- A Theoretically Grounded Application of Dropout in Recurrent Neural Networks