Add [source]

```
keras.layers.Add()
```

入力のリスト同士を足し合わせるレイヤー.

入力はすべて同じshapeをもったテンソルのリストで，1つのテンソルを返す（shapeは同じ）.

例

```
import keras

input1 = keras.layers.Input(shape=(16,))
x1 = keras.layers.Dense(8, activation='relu')(input1)
input2 = keras.layers.Input(shape=(32,))
x2 = keras.layers.Dense(8, activation='relu')(input2)
added = keras.layers.Add()([x1, x2])  # equivalent to added = keras.layers.add([x1, x2])

out = keras.layers.Dense(4)(added)
model = keras.models.Model(inputs=[input1, input2], outputs=out)
```

Subtract [source]

```
keras.layers.Subtract()
```

2つの入力の引き算をするレイヤー.

入力は同じshapeのテンソルのリストを2つで，1つのテンソルを返す(inputs[0] - inputs[1]).
返すテンソルも同じshapeです.

例

```
import keras

input1 = keras.layers.Input(shape=(16,))
x1 = keras.layers.Dense(8, activation='relu')(input1)
input2 = keras.layers.Input(shape=(32,))
x2 = keras.layers.Dense(8, activation='relu')(input2)
# Equivalent to subtracted = keras.layers.subtract([x1, x2])
subtracted = keras.layers.Subtract()([x1, x2])

out = keras.layers.Dense(4)(subtracted)
model = keras.models.Model(inputs=[input1, input2], outputs=out)
```

Multiply [source]

```
keras.layers.Multiply()
```

入力のリストの要素同士の積のレイヤー.

入力はすべて同じshapeをもったテンソルのリストで，1つのテンソルを返す（shapeは同じ）.

Average [source]

```
keras.layers.Average()
```

入力のリストを平均するレイヤー.

入力はすべて同じshapeをもったテンソルのリストで，1つのテンソルを返す（shapeは同じ）.

Maximum [source]

```
keras.layers.Maximum()
```

入力のリストの要素間の最大値を求めるレイヤー.

入力はすべて同じshapeをもったテンソルのリストで，1つのテンソルを返す（shapeは同じ）.

Concatenate [source]

```
keras.layers.Concatenate(axis=-1)
```

入力のリストをconcatenateするレイヤー.

入力は，concatenateする際のaxisを除き，すべて同じshapeをもったテンソルのリストで，全入力をconcatenateした1つのテンソルを返す.

引数

- **axis**: concatenateする際のaxis.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

Dot [source]

```
keras.layers.Dot(axes, normalize=False)
```

2つのテンソルのサンプル間でdot積を計算するレイヤー.

例．もしshapeが `batch_size, n` の2つのテンソル `a` と `b` に適用する場合，出力されるテンソルのshapeは， `(batch_size, 1)` ，出力の要素 `i` は， `a[i]` と `b[i]` のdot積.

引数

- **axes**: 整数か整数のタプル．dot積をとる際にaxisかaxesのどちらを使うか.
- **normalize**: dot積をとる前にdot積のaxisでサンプルをL2正規化するかどうか． Trueなら，dot積の出力は，2つのサンプルのcosine.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

---

add

```
keras.layers.add(inputs)
```

`Add` レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力の総和のテンソル.

例

```
import keras

input1 = keras.layers.Input(shape=(16,))
x1 = keras.layers.Dense(8, activation='relu')(input1)
input2 = keras.layers.Input(shape=(32,))
x2 = keras.layers.Dense(8, activation='relu')(input2)
added = keras.layers.add([x1, x2])

out = keras.layers.Dense(4)(added)
model = keras.models.Model(inputs=[input1, input2], outputs=out)
```

---

subtract

```
keras.layers.subtract(inputs)
```

`Subtract` レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力の差のテンソル.

例

```
import keras

input1 = keras.layers.Input(shape=(16,))
x1 = keras.layers.Dense(8, activation='relu')(input1)
input2 = keras.layers.Input(shape=(32,))
x2 = keras.layers.Dense(8, activation='relu')(input2)
subtracted = keras.layers.subtract([x1, x2])

out = keras.layers.Dense(4)(subtracted)
model = keras.models.Model(inputs=[input1, input2], outputs=out)
```

---

multiply

```
keras.layers.multiply(inputs)
```

`Multiply`レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力の要素同士の積のテンソル.

---

average

```
keras.layers.average(inputs)
```

`Average`レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力の平均のテンソル.

---

maximum

```
keras.layers.maximum(inputs)
```

`Maximum` レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力の要素間の最大値のテンソル.

---

concatenate

```
keras.layers.concatenate(inputs, axis=-1)
```

`Concatenate` レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- **axis**: Concatenation axis.
- ****kwargs**: 標準的なレイヤーのキーワード引数.

戻り値

入力を `axis` の方向でconcateしたテンソル.

---

dot

```
keras.layers.dot(inputs, axes, normalize=False)
```

`Dot` レイヤーの関数インターフェース.

引数

- **inputs**: 入力テンソルのリスト（最低2つ）.
- **axes**: 整数か整数のタプル．dot積をとる際にaxisかaxesのどちらを使うか.

- **normalize**: dot積をとる前にdot積のaxisでサンプルをL2正規化するかどうか． Trueなら，dot積の出力は，2つのサンプルのcosine．
- ****kwargs**: 標準的なレイヤーのキーワード引数．

戻り値

入力のdot積をとったテンソル．