tf.contrib.kfac.layer_collection.LayerParametersDict

Contents

Class `LayerParametersDict`

Defined in `tensorflow/contrib/kfac/python/ops/layer_collection.py`.

An OrderedDict where keys are Tensors or tuples of Tensors.

Ensures that no Tensor is associated with two different keys.

Methods

`__init__`

```
__init__(
    *args,
    **kwargs
)
```

`__delitem__`

```
__delitem__(key)
```

`__eq__`

```
__eq__(other)
```

od.**eq**(y) <==> od==y. Comparison to another OD is order-sensitive while comparison to a regular mapping is order-insensitive.

`__iter__`

```
__iter__()
```

od.**iter**() <==> iter(od)

`__ne__`

```
__ne__(other)
```

od.**ne**(y) <==> od!=y

__reduce__

```
__reduce__()
```

Return state information for pickling

__repr__

```
__repr__(_repr_running={})
```

od.**repr**() <==> repr(od)

__reversed__

```
__reversed__()
```

od.**reversed**() <==> reversed(od)

__setitem__

```
__setitem__(
    key,
    value
)
```

clear

```
clear()
```

od.clear() -> None. Remove all items from od.

copy

```
copy()
```

od.copy() -> a shallow copy of od

fromkeys

```
fromkeys(
    cls,
    iterable,
    value=None
)
```

OD.fromkeys(S[, v]) -> New ordered dictionary with keys from S. If not specified, the value defaults to None.

items

```
items()
```

od.items() -> list of (key, value) pairs in od

iteritems

```
iteritems()
```

od.iteritems -> an iterator over the (key, value) pairs in od

iterkeys

```
iterkeys()
```

od.iterkeys() -> an iterator over the keys in od

itervalues

```
itervalues()
```

od.itervalues -> an iterator over the values in od

keys

```
keys()
```

od.keys() -> list of keys in od

pop

```
pop(
    key,
    default=__marker
)
```

od.pop(k[,d]) -> v, remove specified key and return the corresponding value. If key is not found, d is returned if given, otherwise KeyError is raised.

popitem

```
popitem(last=True)
```

od.popitem() -> (k, v), return and remove a (key, value) pair. Pairs are returned in LIFO order if last is true or FIFO order if false.

setdefault

```
setdefault(
    key,
    default=None
)
```

od.setdefault(k[,d]) -> od.get(k,d), also set od[k]=d if k not in od

update

```
update(
    *args,
    **kwds
)
```

D.update([E,]**F) -> None. Update D from mapping/iterable E and F. If E present and has a .keys() method, does: for k in E: D[k] = E[k] If E present and lacks .keys() method, does: for (k, v) in E: D[k] = v In either case, this is followed by: for k, v in F.items(): D[k] = v

values

```
values()
```

od.values() -> list of values in od

viewitems

```
viewitems()
```

od.viewitems() -> a set-like object providing a view on od's items

viewkeys

```
viewkeys()
```

od.viewkeys() -> a set-like object providing a view on od's keys

viewvalues

```
viewvalues()
```

od.viewvalues() -> an object providing a view on od's values

Class Members

__hash__

Stay Connected

Blog

GitHub

Twitter

Support

Issue Tracker

Release Notes

Stack Overflow

English

Terms | **Privacy**