
crnn.gluon Documentation

1.0

zhoujun

2018 12 02

Contents:

1	crnn.gluon	1
1.1	config module	1
1.2	convert_rec module	1
1.3	crnn module	1
1.4	dataset module	3
1.5	keys module	4
1.6	predict module	4
1.7	train module	4
2	Indices and tables	5
	Python	7

1.1 config module

1.2 convert_rec module

1.3 crnn module

```
class crnn.BidirectionalGRU(hidden_size, num_layers, nOut)
    mxnet.gluon.block.HybridBlock

    __init__(hidden_size, num_layers, nOut)
        Initialize self. See help(type(self)) for accurate signature.

    hybrid_forward(F, x, *args, **kwargs)
        Overrides to construct symbolic graph for this Block.

        x [Symbol or NDArray] The first input tensor.

        *args [list of Symbol or list of NDArray] Additional input tensors.

class crnn.BidirectionalLSTM(hidden_size, num_layers, nOut)
    mxnet.gluon.block.HybridBlock

    __init__(hidden_size, num_layers, nOut)
        Initialize self. See help(type(self)) for accurate signature.

    hybrid_forward(F, x, *args, **kwargs)
        Overrides to construct symbolic graph for this Block.

        x [Symbol or NDArray] The first input tensor.

        *args [list of Symbol or list of NDArray] Additional input tensors.

class crnn.CRNN(n_class, hidden_size=256, num_layers=1)
    mxnet.gluon.block.HybridBlock
```

__init__ (*n_class*, *hidden_size*=256, *num_layers*=1)
Initialize self. See help(type(self)) for accurate signature.

hybrid_forward (*F*, *x*, **args*, ***kwargs*)
Overrides to construct symbolic graph for this *Block*.

x [Symbol or NDAarray] The first input tensor.

***args** [list of Symbol or list of NDAarray] Additional input tensors.

class `crnn.Decoder` (*n_class*, *hidden_size*=256, *num_layers*=1)
`mxnet.gluon.block.HybridBlock`

__init__ (*n_class*, *hidden_size*=256, *num_layers*=1)
Initialize self. See help(type(self)) for accurate signature.

hybrid_forward (*F*, *x*, **args*, ***kwargs*)
Overrides to construct symbolic graph for this *Block*.

x [Symbol or NDAarray] The first input tensor.

***args** [list of Symbol or list of NDAarray] Additional input tensors.

class `crnn.DenseNet`
`mxnet.gluon.block.HybridBlock`

__init__ ()
Initialize self. See help(type(self)) for accurate signature.

hybrid_forward (*F*, *x*, **args*, ***kwargs*)
Overrides to construct symbolic graph for this *Block*.

x [Symbol or NDAarray] The first input tensor.

***args** [list of Symbol or list of NDAarray] Additional input tensors.

class `crnn.Encoder`
`mxnet.gluon.block.HybridBlock`

__init__ ()
Initialize self. See help(type(self)) for accurate signature.

hybrid_forward (*F*, *x*, **args*, ***kwargs*)
Overrides to construct symbolic graph for this *Block*.

x [Symbol or NDAarray] The first input tensor.

***args** [list of Symbol or list of NDAarray] Additional input tensors.

class `crnn.ResNet`
`mxnet.gluon.block.HybridBlock`

__init__ ()
Initialize self. See help(type(self)) for accurate signature.

hybrid_forward (*F*, *x*, **args*, ***kwargs*)
Overrides to construct symbolic graph for this *Block*.

x [Symbol or NDAarray] The first input tensor.

***args** [list of Symbol or list of NDAarray] Additional input tensors.

class `crnn.VGG`
`mxnet.gluon.block.HybridBlock`

```

__init__()
    Initialize self. See help(type(self)) for accurate signature.

hybrid_forward(F, x, *args, **kwargs)
    Overrides to construct symbolic graph for this Block.

x [Symbol or NDAarray] The first input tensor.

*args [list of Symbol or list of NDAarray] Additional input tensors.

```

1.4 dataset module

```

class dataset.ImageDataset (data_txt: str, data_shape: tuple, img_channel: int, num_label: int,
                             alphabet: str, phase: str = 'train')
    mxnet.gluon.data.dataset.Dataset

```

```

__init__ (data_txt: str, data_shape: tuple, img_channel: int, num_label: int, alphabet: str, phase: str
          = 'train')

```

- **data_txt** – label
- **data_shape** – (h,w)
- **img_channel** –
- **num_label** –
- **alphabet** –

```

label_encoder (label)
    label_label

```

```

    label – label

```

```

pre_processing (img_path)
    resizeresize

```

```

    img_path –

```

```

class dataset.RecordDataset (filename, data_shape: tuple, img_channel: int, num_label: int)
    mxnet.gluon.data.dataset.RecordFileDataset

```

A dataset wrapping over a RecordIO file containing images Each sample is an image and its corresponding label

```

__init__ (filename, data_shape: tuple, img_channel: int, num_label: int)
    Initialize self. See help(type(self)) for accurate signature.

```

```

label_encoder (label)
    label_label

```

```

    label – label

```

```

pre_processing (img)
    :param img_path: :return:

```

1.5 keys module

1.6 predict module

```
class predict.GluonNet (model_path, alphabet, img_shape, net, img_channel=3, gpu_id=None)
    object

    __init__ (model_path, alphabet, img_shape, net, img_channel=3, gpu_id=None)
        gluon :param model_path: :param alphabet: :param img_shape: (w,h) :param net: model_path :param
        img_channel: : 1,3 :param gpu_id: gpu

    pre_processing (img_path)
        resizeresize :param img_path: :return:

    predict (img_path)
        numpy :param img_path: :return:

predict.decode (preds, alphabet, raw=False)

predict.try_gpu (gpu)
    If GPU is available, return mx.gpu(0); else return mx.cpu()
```

1.7 train module

```
train.accuracy (predictions, labels, alphabet)

train.evaluate_accuracy (net, dataloader, ctx, alphabet)

train.setup_logger (log_file_path: str = None)

train.train ()
```


CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

c

config, 1
crnn, 1

d

dataset, 3

k

keys, 4

p

predict, 4

t

train, 4

Symbols

`__init__()` (`crnn.BidirectionalGRU`), 1
`__init__()` (`crnn.BidirectionalLSTM`), 1
`__init__()` (`crnn.CRNN`), 1
`__init__()` (`crnn.Decoder`), 2
`__init__()` (`crnn.DenseNet`), 2
`__init__()` (`crnn.Encoder`), 2
`__init__()` (`crnn.ResNet`), 2
`__init__()` (`crnn.VGG`), 2
`__init__()` (`dataset.ImageDataset`), 3
`__init__()` (`dataset.RecordDataset`), 3
`__init__()` (`predict.GluonNet`), 4

C

`config()`, 1
`crnn()`, 1

D

`dataset()`, 3

H

`hybrid_forward()` (`crnn.BidirectionalGRU`), 1
`hybrid_forward()` (`crnn.BidirectionalLSTM`), 1
`hybrid_forward()` (`crnn.CRNN`), 2
`hybrid_forward()` (`crnn.Decoder`), 2
`hybrid_forward()` (`crnn.DenseNet`), 2
`hybrid_forward()` (`crnn.Encoder`), 2
`hybrid_forward()` (`crnn.ResNet`), 2
`hybrid_forward()` (`crnn.VGG`), 3

K

`keys()`, 4

L

`label_enocder()` (`dataset.ImageDataset`), 3
`label_enocder()` (`dataset.RecordDataset`), 3

P

`pre_processing()` (`dataset.ImageDataset`), 3
`pre_processing()` (`dataset.RecordDataset`), 3

`pre_processing()` (`predict.GluonNet`), 4
`predict()`, 4
`predict()` (`predict.GluonNet`), 4

T

`train()`, 4