nn.col2im

```

Copy col2im( data:@Tensor, image_shape:Span, block_shape:Span, dilations:Option>, pads:Option>, strides:Option>, )->Tensor

```

The operator rearranges column blocks back into a multidimensional image

Col2Im behaves similarly to PyTorch's fold https://pytorch.org/docs/stable/generated/torch.nn.Fold.html, but it only supports batched multi-dimensional image tensors. Another implementation in Python with N-dimension support can be found at https://github.com/f-dangel/unfoldNd/.

Args

- data
- (@Tensor
-) - Input data tensor to be rearranged from column blocks back into an image. This is a 3-dimensional tensor containing [N, C * n-ary-product(block_shape), L], where N is batch dimension, C is image channel dimension and L is number of blocks.
- image_shape
- (Span
-) - The shape of the spatial dimensions of the image after rearranging the column blocks.This is a 1-dimensional tensor with size of at least 2, containing the value [H_img, W_img] for a 2-D image or [dim_i1, dim_i2, ..., dim_iN] for a N-D image.
- block_shape
- (Span
-) - The shape of the block to apply on the input.This is a 1-dimensional tensor of size of at least 2, containing the value [H_block, W_block] for a 2-D image or [dim_b1, dim_b2, ..., dim_bN] for a N-D block.This is the block-shape before dilation is applied to it.
- dilations
- (Option>
-) - 1-dimensional tensor with dilation value along each spatial axis of the image. If not present, the dilation defaults to 1 along each spatial axis of the image.
- pads
- (Option>
-) - 1-dimensional tensor with padding value for the beginning and ending along each spatial axis, it can take any value greater than or equal to 0. The value represent the number of pixels added to the beginning and end part of the corresponding axis.pads
- format should be as follow [x1_begin, x2_begin...x1_end, x2_end,...], where xi_begin is the number of pixels added at the beginning of axisi
- and xi_end is the number of pixels added at the end of axisi
- . If not present, the padding defaults to 0 along start and end of each spatial axis.
- strides
- (Option>
-) - 1-dimensional tensor with stride value along each spatial axis. If not present, the stride defaults to 1 along each spatial axis.
-

Returns

ATensor output tensor produced by rearranging blocks into an image.

Examples

```

Copy useorion::operators::nn::NNTrait; useorion::numbers::FixedTrait; useorion::operators::nn::FP16x16NN; useorion::numbers::FP16x16; useorion::operators::tensor::{Tensor,TensorTrait,FP16x16Tensor};

fnexample_col2im()->Tensor { letmutshape=ArrayTrait:::new(); shape.append(1); shape.append(5); shape.append(5);

letmutdata=ArrayTrait::new(); data.append(FP16x16{ mag:65536, sign:false}); data.append(FP16x16{ mag:393216, sign:false}); data.append(FP16x16{ mag:720896, sign:false}); data.append(FP16x16{ mag:1048576, sign:false}); data.append(FP16x16{ mag:1376256, sign:false}); data.append(FP16x16{ mag:131072, sign:false}); data.append(FP16x16{ mag:458752, sign:false}); data.append(FP16x16{ mag:786432, sign:false}); data.append(FP16x16{ mag:1114112, sign:false}); data.append(FP16x16{ mag:1441792, sign:false}); data.append(FP16x16{ mag:196608, sign:false});
```

```
data.append(FP16x16{ mag:524288, sign:false}); data.append(FP16x16{ mag:851968, sign:false}); data.append(FP16x16{ mag:1179648, sign:false}); data.append(FP16x16{ mag:1507328, sign:false}); data.append(FP16x16{ mag:262144, sign:false}); data.append(FP16x16{ mag:589824, sign:false}); data.append(FP16x16{ mag:917504, sign:false}); data.append(FP16x16{ mag:1245184, sign:false}); data.append(FP16x16{ mag:1572864, sign:false}); data.append(FP16x16{ mag:327680, sign:false}); data.append(FP16x16{ mag:0, sign:false}); data.append(FP16x16{ mag:983040, sign:false}); data.append(FP16x16{ mag:1310720, sign:false}); data.append(FP16x16{ mag:1638400, sign:false}); letmutX=TensorTrait::new(shape.span(), data.span());

letimage_shape=array![5,5].span(); letblock_shape=array![1,5].span();

returnNNTrait::col2im( @X, image_shape, block_shape, Option::None, Option::None, Option::None, );

}

                    [ [ [ [1.0,2.0,3.0,4.0,5.0], [6.0,7.0,8.0,9.0,0.0], [11.0,12.0,13.0,14.0,15.0],
                    [16.0,17.0,18.0,19.0,20.0], [21.0,22.0,23.0,24.0,25.0], ] ] ]
```

Previous nn.grid_sample Next nn.conv_transpose

Last updated15 days ago