ONNX Pre-processing WG

Monthly meeting - Apr 13, 2022

Agenda

- (Wenbing Li) Continuation - Intro about pre-processing in Microsoft
- Status - Open issues/PRs
- Proposal: Resize - Antialiasing
- Discussion: Domain specific functions for higher level abstractions
- Open floor

(Wenbing Li) Continuation - Intro about pre-processing in Microsoft

-

Status - Open issues/PRs

PRs:

[Merged] Add SequenceMap function
https://github.com/onnx/onnx/pull/3892

Add Batch processing with SequenceMap tutorial
https://github.com/onnx/tutorials/pull/265

[In progress] Add Data preprocessing with ONNX: ResNet-50 example
https://github.com/onnx/tutorials/pull/266

Add Resize-16: Antialiasing filter
https://github.com/onnx/onnx/pull/4126

Issues:

ONNX runtime function inlining:

FunctionImpl should consider nested subgraphs when updating graph inputs/outputs
https://github.com/microsoft/onnxruntime/issues/10876

Error about missing type information when nesting model local functions
https://github.com/microsoft/onnxruntime/issues/10250

Local functions with subgraphs: GraphProto attribute inferencing error
https://github.com/microsoft/onnxruntime/issues/10698

[Solved] Submodel extraction with model local functions
https://github.com/onnx/onnx/issues/3933

[Solved] Optional inputs within model local function
https://github.com/microsoft/onnxruntime/issues/10245

Sequence support:

Support for SequenceProto in ONNX parser
https://github.com/onnx/onnx/issues/4043

Can't constant fold SequenceEmpty node
https://github.com/microsoft/onnxruntime/issues/11041

Sequence initializers
https://github.com/onnx/onnx/issues/4105

Models:

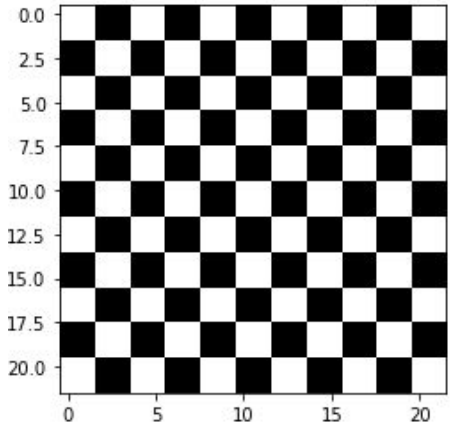
ResNet-50 network has hardcoded batch-size 1
https://github.com/onnx/models/issues/509

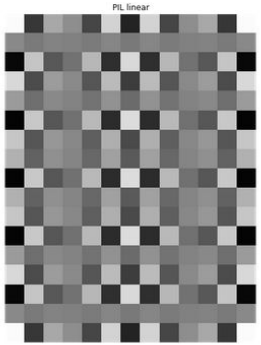
Proposal:
Resize - Antialiasing

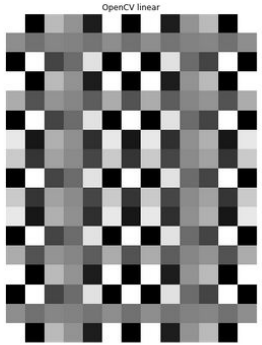
https://github.com/onnx/onnx/pull/4126

See also:
https://github.com/onnx/working-groups/blob/main/preprocessing/notebooks/resize-antialias/resize-antialias.ipynb

From [22, 22, 3] to [17, 13, 3]

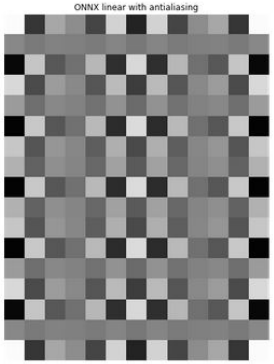


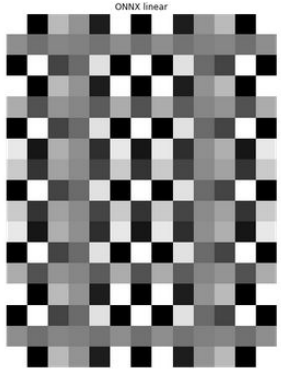




Resize(..., antialias=1)

Resize(..., antialias=0)





Discussion: Domain specific functions for higher level abstractions

Idea: Offer higher level abstractions to users for commonly used operations.

Examples:

image.CenteredCrop(im, [200, 200])

image.Resize<policy='not-smaller'>(im, [200, 200])

image.Resize<policy='not-larger'>(im, [200, 200])

Pros:

- Simple for users
- Possibility to specialize

Cons:

- Need to maintain
- Overlap

```
centered_crop (uint8[H1, W1, C] image, int64[2] crop_win) => (uint8[H2, W2, C] out_image)
{
    k2 = Constant <value = int64[1] {2}> ()
    axes = Constant <value = int64[2] {0, 1}> ()
    x_shape = Shape (image)
    h, w, c = Split <axis = 0> (x_shape)
    hw = Concat <axis = 0> (h, w)
    hw_diff = Sub (hw, crop_win)
    start_xy = Div (hw_diff, k2)
    end_xy = Add (start_xy, crop_win)
    out_image = Slice (image, start_xy, end_xy, axes)
}
```

```
Output sizes:
(200, 200, 3)
(200, 200, 3)
```

```
resize_not_smaller (uint8[H1, W1, C] image, int64[2] target_size) => (uint8[H2, W2, C] out_image)
{
    image_shape = Shape (image)
    h, w, c = Split <axis = 0> (image_shape)
    hw = Concat <axis = 0> (h, w)
    hw_f = Cast <to = 1> (hw)
    target_size_f = Cast <to = 1> (target_size)
    ratios = Div(target_size_f, hw_f)
    ratio_resize = ReduceMax(ratios)
    k1f = Constant <value = float[1] {1.0}> ()
    scales_resize = Concat <axis = 0> (ratio_resize, ratio_resize, k1f)
    out_image = Resize <mode = \"linear\"> (image, , scales_resize)
}
```

```
Output sizes:
(200, 300, 3)
(200, 200, 3)
```

```
resize_not_larger (uint8[H1, W1, C] image, int64[2] target_size) => (uint8[H2, W2, C] out_image)
{
    image_shape = Shape (image)
    h, w, c = Split <axis = 0> (image_shape)
    hw = Concat <axis = 0> (h, w)
    hw_f = Cast <to = 1> (hw)
    target_size_f = Cast <to = 1> (target_size)
    ratios = Div(target_size_f, hw_f)
    ratio_resize = ReduceMin(ratios)
    k1f = Constant <value = float[1] {1.0}> ()
    scales_resize = Concat <axis = 0> (ratio_resize, ratio_resize, k1f)
    out_image = Resize <mode = \"linear\"> (image, , scales_resize)
}
```

```
Output sizes:
(133, 200, 3)
(200, 200, 3)
```

https://github.com/onnx/working-groups/blob/main/preprocessing/notebooks/image_proc_functions/ImageProcFunctions.ipynb

Open floor