ONNX Preprocessing WG

Joaquin Anton (NVIDIA)

June 28, 2023

ONNX Preprocessing

Before

Data source
(e.g. Image)

Preprocessing

ONNX Model

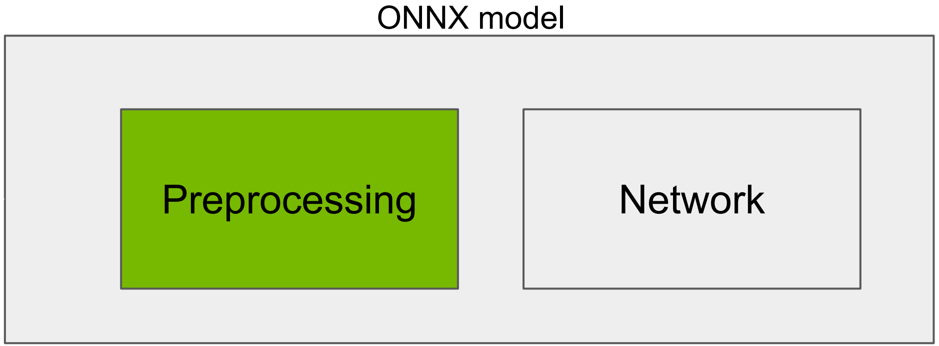
- Not serialized with the model
- Defined vaguely
- Executed by third party tools

- DNN stored in an onnx format
- Executed by one of the supported ONNX runtimes (e.g. TensorRT)

ONNX preprocessing

Group's Mission

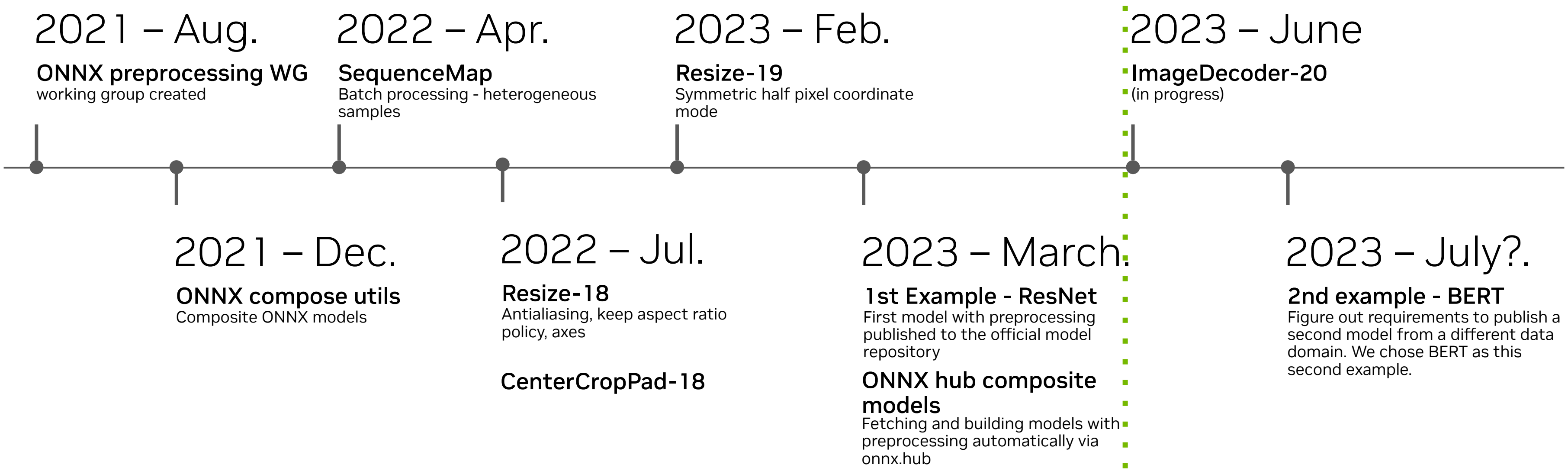
Data source
(e.g. Image)



ONNX

- Serialize with the model
- Add operators
- Add infrastructure
- Publish real example
- Document
- Conclude and hand over to other groups
  - New operators -> Operators SIG
  - New models -> Models SIG
  - Main design, infrastructure, tools -> Infrastructure SIG

Timeline



2021 – Aug.

ONNX preprocessing WG
working group created

2021 – Dec.

ONNX compose utils
Composite ONNX models

2022 – Apr.

SequenceMap
Batch processing - heterogeneous samples

2022 – Jul.

Resize-18
Antialiasing, keep aspect ratio policy, axes

CenterCropPad-18

2023 – Feb.

Resize-19
Symmetric half pixel coordinate mode

2023 – March.

1st Example - ResNet
First model with preprocessing published to the official model repository

ONNX hub composite models
Fetching and building models with preprocessing automatically via onnx.hub

2023 – June

ImageDecoder-20
(in progress)

2023 – July?.

2nd example - BERT
Figure out requirements to publish a second model from a different data domain. We chose BERT as this second example.

Operators update

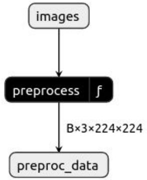
Supporting ResNet preprocessing steps

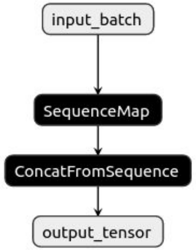
- Resize-18: Antialiasing filter and keep aspect ratio policy
    - Antialiasing optional filter for downscaling.
        - Applied by popular image processing toolkits (e.g. Pillow)
    - Keep aspect ratio semantics.
        - "stretch" (default), "not_larger", "not_smaller"
- CenterCropPad-18: Higher level abstraction on top of Slice and Pad operators
    - ONNX function (can be implemented with existing ONNX ops)
    - Convenient function and possibility to specialize by runtimes
- Resize-19: Half-pixel symmetric coordinate mode
    - Flip invariant version of "half_pixel" mode
    - Important in applications dealing with image locations (e.g. object detection)

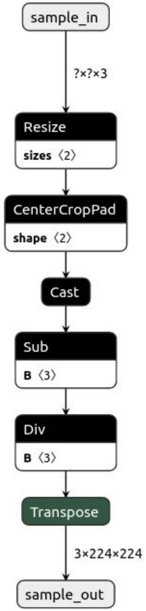
ResNet preprocessing model

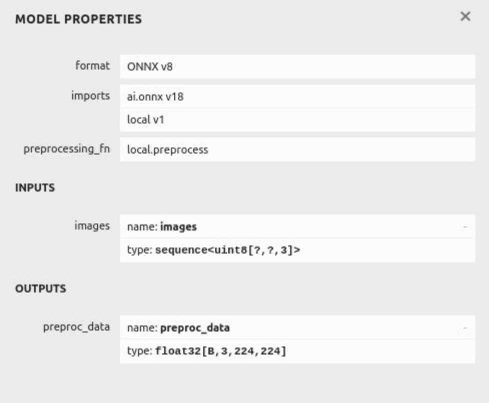
ResNet preprocessing model

https://github.com/onnx/models/tree/main/vision/classification/resnet#preprocessing









MODEL PROPERTIES ×

format: ONNX v8

imports: ai.onnx v18
local v1

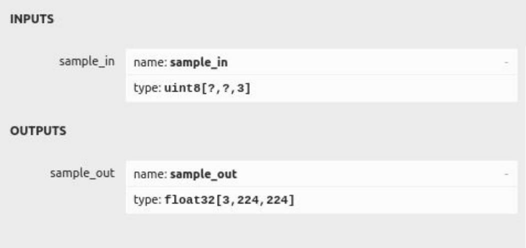
preprocessing_fn: local.preprocess

INPUTS

images: name: **images**
type: `sequence<uint8[?,?,3]>`

OUTPUTS

preproc_data: name: **preproc_data**
type: `float32[B,3,224,224]`



INPUTS

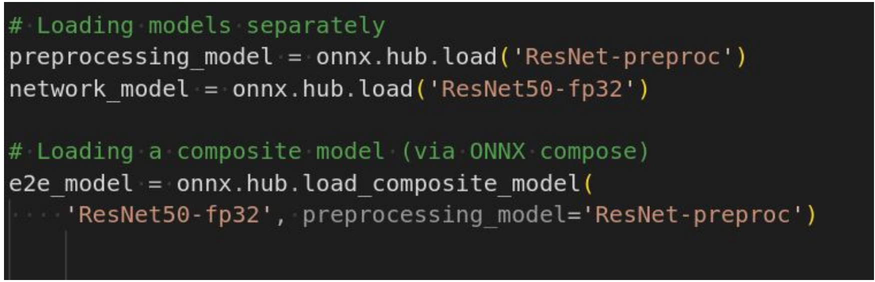
sample_in: name: **sample_in**
type: `uint8[?,?,3]`

OUTPUTS

sample_out: name: **sample_out**
type: `float32[3,224,224]`

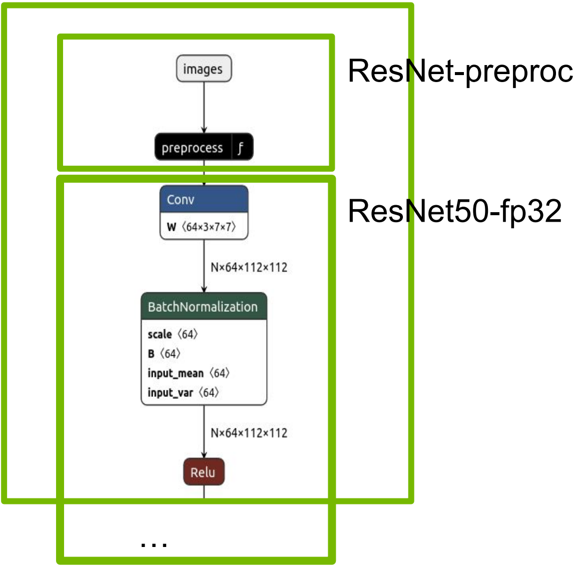
ONNX hub composite models

Automatic generation of preprocessing+network



```python
# Loading models separately
preprocessing_model = onnx.hub.load('ResNet-preproc')
network_model = onnx.hub.load('ResNet50-fp32')

# Loading a composite model (via ONNX compose)
e2e_model = onnx.hub.load_composite_model(
    'ResNet50-fp32', preprocessing_model='ResNet-preproc')
```



ResNet-preproc

ResNet50-fp32

Thank you for listening!

Get Involved!
Github: PRs, Issues, and Discussions
Slack channel: https://slack.lfai.foundation and join **#onnx-preprocessing**
Monthly WG meetings (see slack channel for announcements)