Responsible AI @ ONNX: Metadata, Model Cards, and Provenance

Ria Cheruvu, Rodolfo Esteves, Bhargavi Karumanchi, Rajeev Nalawadi
Intel Corporation
6/24/2022

Agenda

Proposal: Enable Machine-readable AI metadata in ONNX - Github Issue 3958

- Track provenance and other relevant characteristics
- Identify models with Mixed Precision
- Facilitate runtime metadata checks (Pre-processing element of ONNX Graph)
- Enables queries on model zoo to filter for relevant use cases

Timeline of activities

(Oct'21) Presented roadmap proposal to ONNX Steering Committee:

- Suggested first cut of relevant fields
- Proposed specific machine-readable format & query language

(Now) Extend proposal to model creation/consumption workflow:

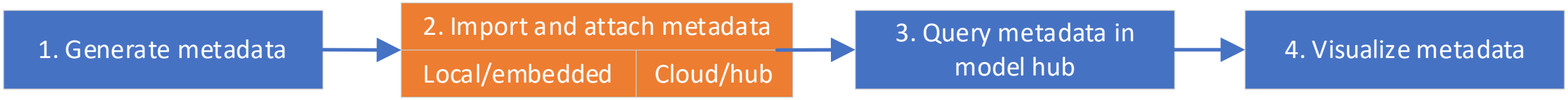
- Recap motivation and context
- Present proposal with E2E design: Creation, querying, and viz of metadata apropos hubs
- Solicit contributions from the ONNX Community

ONNX needs Responsible AI

Motivation

- Metadata is a key enabler of RAI and Explainable AI
  - Enable provenance and characterization for transparency
  - Help identify **why** and **where** an AI model failed (offline + at runtime)

- Current emphasis is on human-comprehensible metadata
  - Industry standard: Model Cards



Metadata is helpful if it is **machine-readable**

Model Card Metadata Support in frameworks (TFX as example)

Context

```
import model_card_toolkit as mctlib

model_card_json = {'model_details': {'name': 'Census Income Classifier'},
                   'model_details': {'overview': '...'},
                   'model_details': {'owners': [{"name": "Model Cards Team",
                                                 "contact": "model-cards@google.com"}]},
                   'considerations': {'use_cases':[{"description": "..."}],
                    'limitations': [{'description': "..."}],
'ethical_considerations': [...]}
                  }
from model_card_toolkit.tfx.component import ModelCardGenerator

mct_gen = ModelCardGenerator(statistics=statistics_gen.outputs['statistics'],
                             evaluation=evaluator.outputs['evaluation'],
                             json=json.dumps(model_card_json))
```

Attaching metadata to an ONNX model

Context

- API support to do this manually:

```
model = onnxmltools.load_model("model.onnx")
meta = model.metadata_props.add()
meta.key = "model_card"
meta.value = <markdown-with-frontmatter string>
onnxmltools.utils.save_model(model, "model_with_metadata.onnx")
```

- Can (and in some cases is supported by frameworks)

```
# from pytorch exporter
torch.onnx.export(model, input, ONNX_MODEL_FILE)
```

```
// In ML.NET
using (var stream = File.Create(ONNX_MODEL_FILE)) {
  mlContext.Model.ConvertToOnnx(trainedModel, data, stream);
}
```

Semantic Web infrastructure (RDF)

Leverage RDF for defining machine readable (model metadata fields)

- Structured metadata, extensible and machine readable
- Embeddable (compatible with existing ONNX metadata fields)
- Controlled vocabularies for Provenance, Explainable and Ethical ML are already being developed (eg IEEE P7003)

Model Provenance Field Examples

Proposal

Model details	Ex: Model name: <Bert-base, Resnet-50, ..> Ex: Architecture: <NLP, CNN, GNN, ..>
Data Provenance	Ex: Training data location: <Disk file, online DB, ..> Ex: Number of examples: < > Ex: Environment: < >
Trainer Provenance	Ex: Trainer algorithm: <single, ensemble-1, ..> Ex: Hyperparameters: <Learning rate, ….>
Applications	Ex: Scope: <..> Ex: Out of Scope: <..>
Metrics	Ex: Performance: <..> Ex: Average precision: <..>
Documentation	

See https://github.com/onnx/onnx/issues/3958 for full detail

Model Provenance + RDF

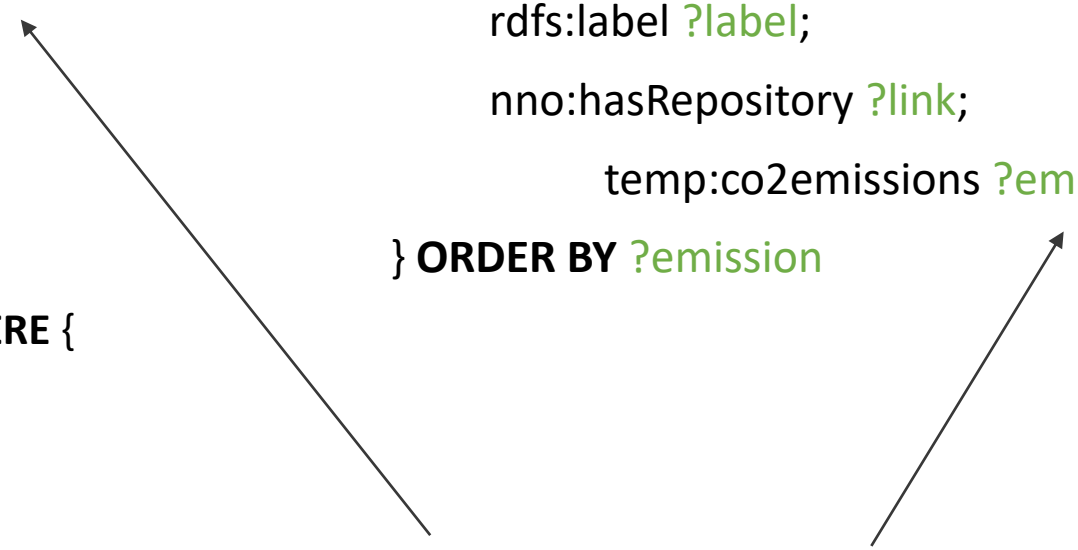
Proposal

- Example SPARQL queries (using FAIRnets ontology: “nno”)

```
SELECT ?label ?category WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        nno:hasIntendedUse “nlp”;
        doap:category ?category.
FILTER(REGEX(LCASE(STR(?category)), “sports”))
}
```

```
SELECT ?label ?creator ?link (COUNT(?layer)) WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        dc:creator ?creator;
        nno:hasRepository ?link;
        nno:hasModel ?model.
?model nno:hasLayer ?layer.
}
```

```
SELECT ?label ?link ?emission WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        nno:hasRepository ?link;
                temp:co2emissions ?emission.
} ORDER BY ?emission
```



Software support for Model Cards has started to show up in frameworks

Model Provenance + RDF (cont'd)

Proposal

- Example SPARQL queries (extending FAIRnets)

For bias control

```
SELECT ?label ?identifiable WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        temp:trainingDS ?ds;
        doap:category ?category.
?ds a temp:DataSet;
   temp:hasPPI: ?identifiable.
}
```

For mixed precision

```
SELECT ?label  WHERE {
?network a nno:NeuralNetwork;
        nno:hasModel ?model.
?model a nno:Model;
                nno:hasLayer
          <https://onnx.org/QuantizedLinear>.
}
```

Querying via the HuggingFace Hub API

Proposal

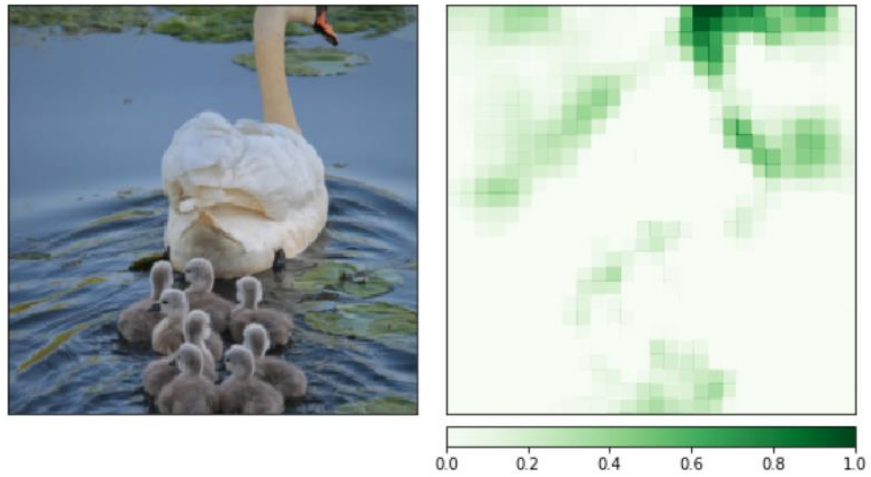
- Example SPARQL queries (using FAIRnets vocabulary)

```
query = “SELECT ?label  WHERE {
?network a nno:NeuralNetwork;
        nno:hasModel ?model.
?model a nno:Model;
                nno:hasLayer
          <https://onnx.org/QuantizedLinear>.
}”
from huggingface_hub import HfApi, ModelFilter, DatasetFilter

api = HfApi()
filt = ModelFilter(sparql_query=query)
api.list_models(filter=filt)
```

Presentation of profiled/XAI data

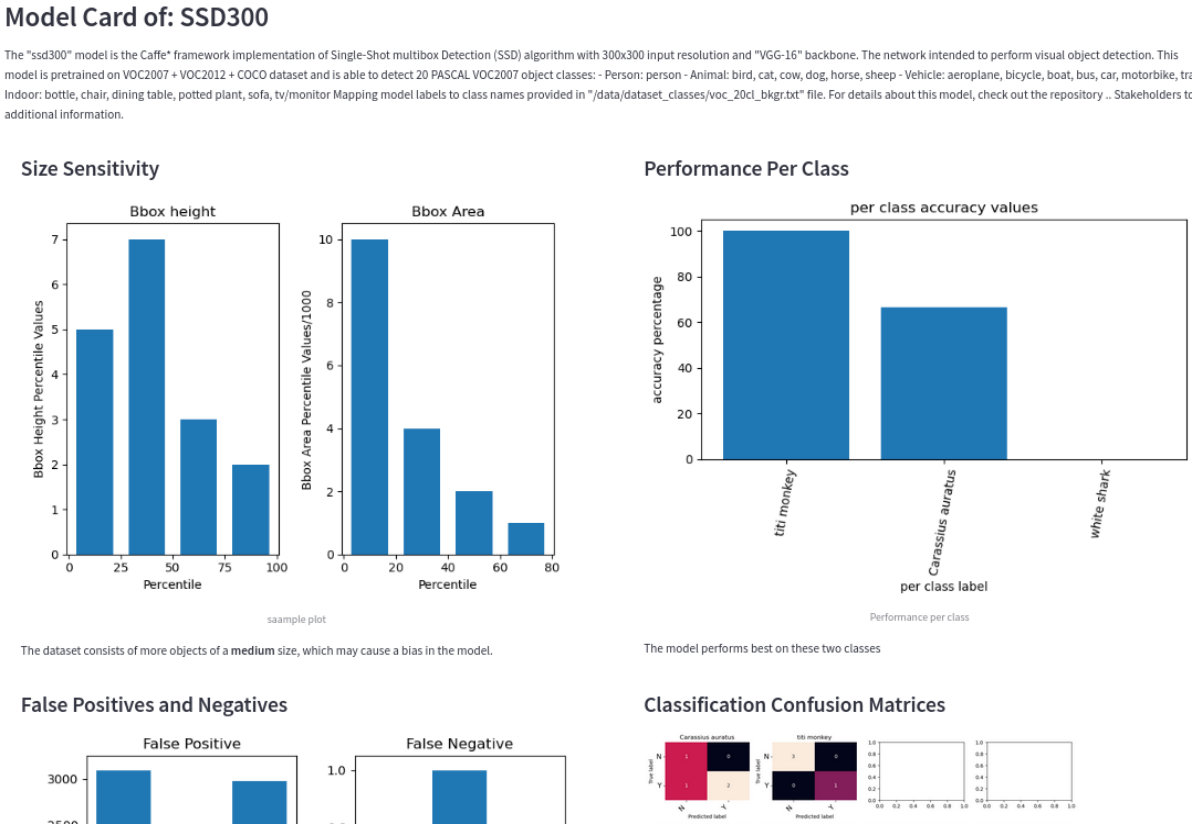
Proposal



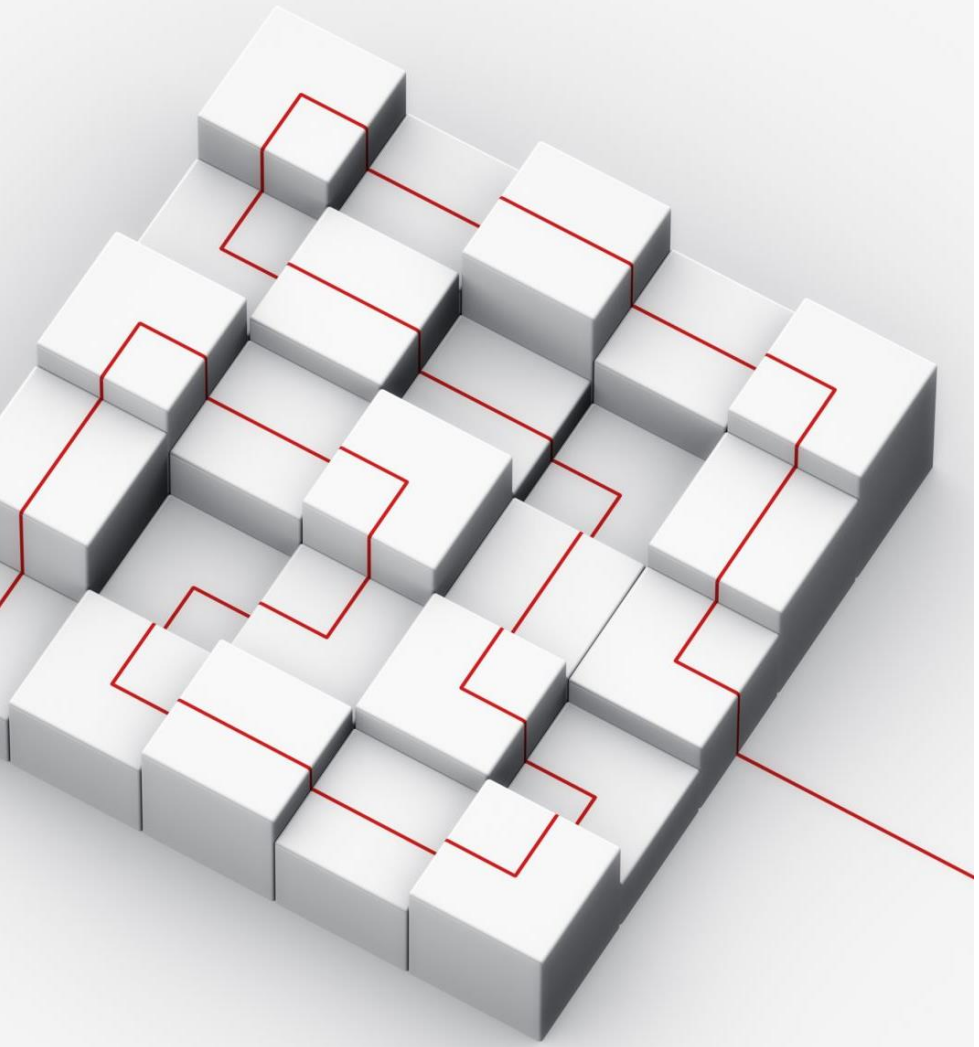
Source: [Captum Occlusion-based attribution for ResNet Intepretability]

us men have right touch in relay duel against australia thens , aug . 17 - so michael phelps
mark spitz . and it is too early to tell if he will match aleksandr dityatin , the soviet gymnast

Source: [Captum LIME for text model intepretability]



Source: Intel OpenVino Performance Profiling Solution

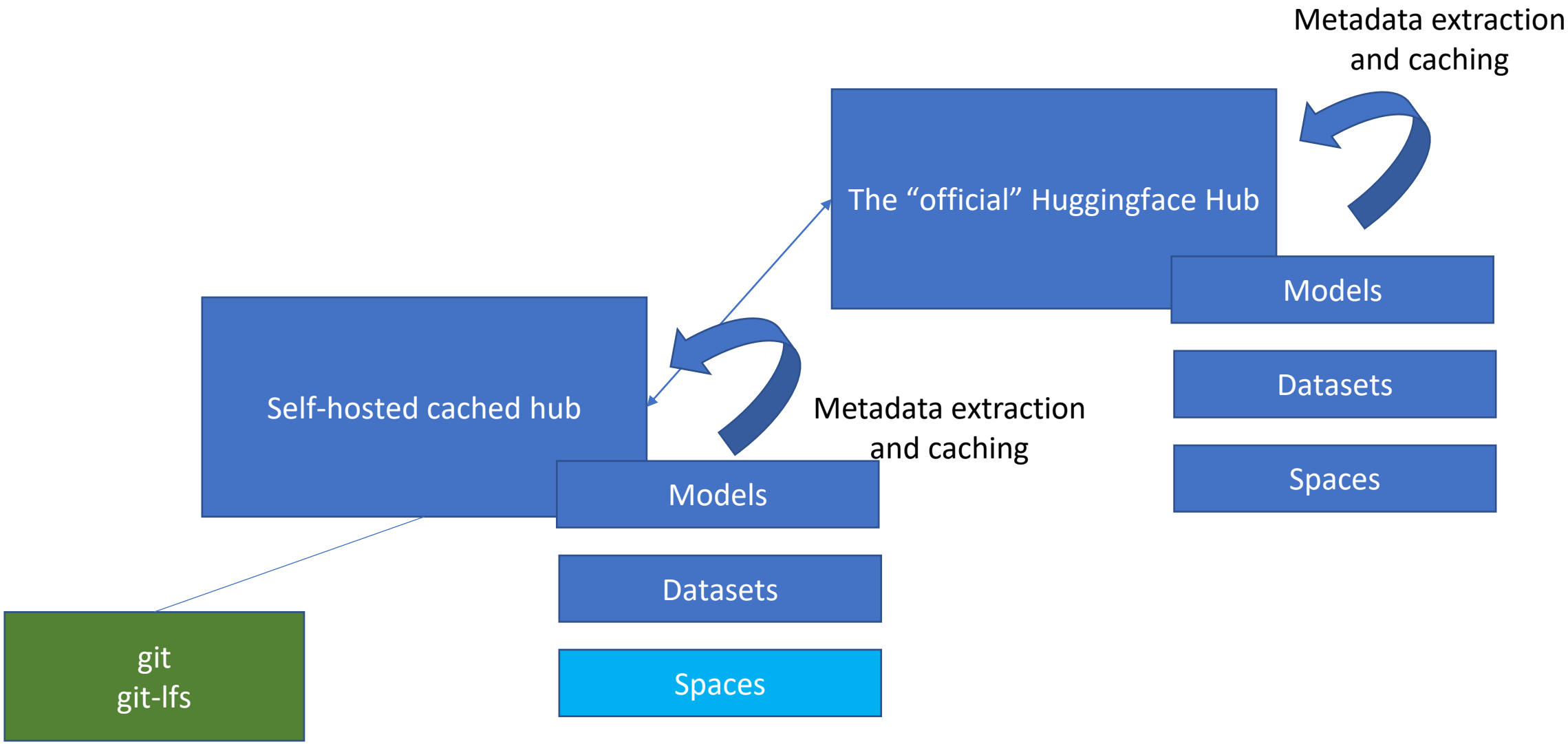


Call to Action: Join us

- Building on Oct'21 roadmap proposal, we extended our proposal to cover:
  - A more concrete design spanning the model-creation/consumption workflow
  - Metadata creation, querying, and filtering from hubs => leveraging currently existing ONNX infrastructure
- We are preparing an initial straw-man implementation, updates to come!
- We welcome collaboration and open dialog on the next steps to develop these solutions.
- See https://github.com/onnx/onnx/issues/3958 for full detail

Thank You !!

Query/frontend interface (prototype: operation)



Model Provenance Fields (first round)

Metadata coverage <examples>:

- Model details
  - Model name: <Bert-base, Resnet-50, ..>
  - Architecture: <NLP, CNN, GNN, ..>
  - Inputs: <sequence of words, RGB Image, ..>
  - Outputs: <sequence of vectors, class-labels, encoded-anchors, ..>
  - Data Precisions: <..>
- Data Provenance
  - Training data location: <Disk file, online DB, ..>
  - Number of examples: < >
  - Number of features: < >
  - Environment: < >
  - Geo groups: < >
  - …
- Trainer Provenance
  - Trainer algorithm: <single, ensemble-1, ..>
  - Hyperparameters: <Learning rate, ….>
  - ….

Model Provenance Fields … (first round)

- Authors
  - Name: <Individual, company, ..>
  - Origination date: <..>
- Citation link
- Documentation link
- Licensing
- Applications
  - Scope: <..>
  - Out of Scope: <..>
- Metrics
  - Performance: <..>
  - Average precision: <..>
  - ….

ONNX Model Zoo Mixed Precision

Presenter(s): Bhargavi Karumanchi, Rodolfo Esteves, Rajeev Nalawadi
Intel Corporation

Models with Mixed Precision

Mixed Precision Trends:

- Trend continues towards availability of “Models with Mixed Precision”
- TCO/Performance driving the need for mixed precisions
- Limited loss of accuracy while deploying mixed precisions
- Various data precisions available in hardware
  - FP32
  - FP16 / FP32
  - BF16 / FP32
  - Int8 / BF16
  - Int8 / BF16 / FP32
  - …..
  - ..many other potential combinations in future (eg: FP8, …)

ONNX Model Zoo Proposal

- Model zoo repository captures the mixed precision specifics as part of metadata

Name	Type	Description
ir_version	int64	The ONNX version assumed by the model.
opset_import	OperatorSetId	A collection of operator set identifiers made available to the model. An implementation must support all operators in the set or reject the model.
producer_name	string	The name of the tool used to generate the model.
producer_version	string	The version of the generating tool.
domain	string	A reverse-DNS name to indicate the model namespace or domain, for example, 'org.onnx'
model_version	int64	The version of the model itself, encoded in an integer.
doc_string	string	Human-readable documentation for this model. Markdown is allowed.
graph	Graph	The parameterized graph that is evaluated to execute the model.
metadata_props	map<string,string>	Named metadata values; keys should be distinct.
training_info	TrainingInfoProto[]	An optional extension that contains information for training.
functions	FunctionProto[]	An optional list of functions local to the model.

Capture in metadata the combinations of data precisions supported in model

Benefits of Mixed Precision (metadata)

Mixed Precision conveyed using metadata:

- Enables model authors to convey all the data precisions utilized in the model
- Allows model consumers to make relevant decisions based on hardware features supported

ONNX Model Provenance

Presenter: Bhargavi Karumanchi (Intel)
Contributors: Rodolfo Esteves, Rajeev Nalawadi
Intel Corporation

Why ONNX Model Provenance

- Industry trends towards digitization have accelerated into broader AI infusion across various vertical segments
- While AI models get integrated into End-2-End process flows it's becoming increasingly hard to detect whether specific models being integrated can meet the characteristics
  - Fairness, Transparency, Human centered approach, Trusted, Secure, Privacy protections, machine readable
- Once the ONNX model is created using converters, provide the ONNX model creators/converters an additional option for establishing provenance prior to broader publishing
- ONNX model provenance as a step towards Responsible AI

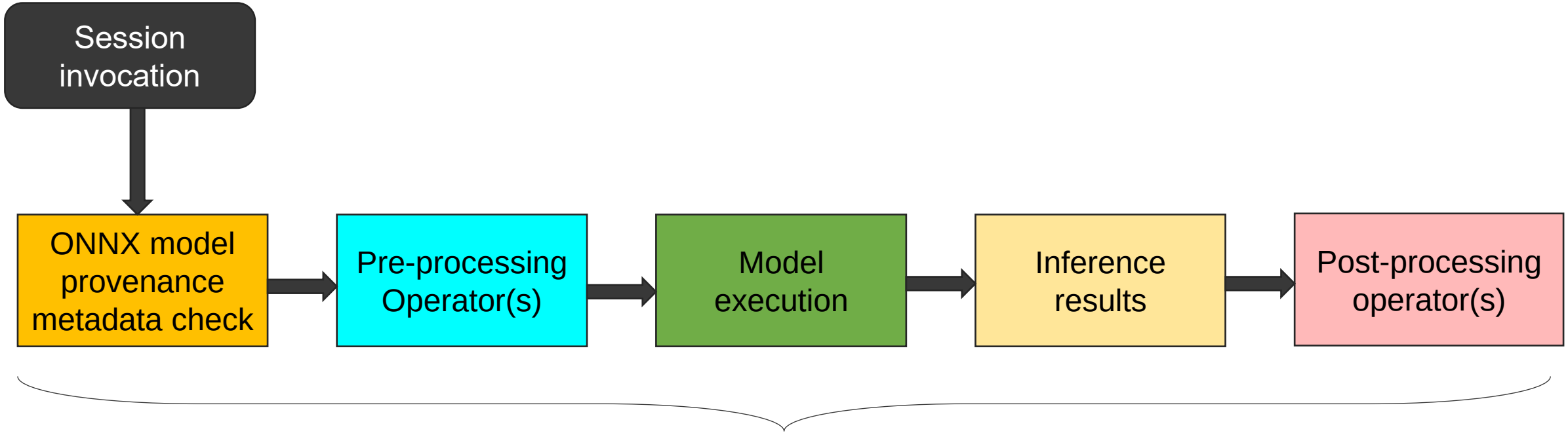
Goals & Requirements

- Native framework / converter tools to generate metadata properties for model
  - Model Information: <Description of model>
  - Model Architecture: <NLP/CNN/…, Dataset(s) used, Format of Inputs & Outputs, Accuracy expected>
  - Usage scenarios: <Typical applications of model>
  - Privacy considerations: <consent required etc..>
  - ……other metadata properties….
- Requires ONNX model provenance (metadata) check to be implemented in runtime(s)
  - Model metadata characteristics performed queried at start of session

Recommended guidelines

- Minimal additions to ONNX(memory) payload
- Minimal requirements for extra processing
- As backwards-compatible as possible (rejecting non-signed/non-annotated models should be an opt-in)

Flow with Model provenance metadata check (proposal)



Name	Type	Description
ir_version	int64	The ONNX version assumed by the model.
opset_import	OperatorSetId	A collection of operator set identifiers made available to the model. An implementation must support all operators in the set or reject the model.
producer_name	string	The name of the tool used to generate the model.
producer_version	string	The version of the generating tool.
domain	string	A reverse-DNS name to indicate the model namespace or domain, for example, 'org.onnx'
model_version	int64	The version of the model itself, encoded in an integer.
doc_string	string	Human-readable documentation for this model. Markdown is allowed.
graph	Graph	The parameterized graph that is evaluated to execute the model.
metadata_props	map<string,string>	Named metadata values; keys should be distinct.
training_info	TrainingInfoProto[]	An optional extension that contains information for training.
functions	FunctionProto[]	An optional list of functions local to the model.

Capture in metadata the model provenance aspects

Maybe we should mandate a format for this information that is also machine-readable. For example, the **Semantic Web infrastructure**!

The Semantic Web infrastructure (RDF)

- Structured metadata, extensible and machine readable
- Embeddable (compatible with existing ONNX metadata fields)
- Controlled vocabularies for Provenance, Explainable and Ethical ML are already being developed (eg IEEE P7003)

Model Cards

MODEL CARD

MediaPipe BlazeFace

FULL RANGE

MODEL DETAILS

A relatively lightweight model (1.1MB in size) for detecting one or multiple faces within an image captured by a smartphone camera, primarily targeting back-facing camera images. Runs super-real-time (~90FPS on Pixel 3 GPU, ~165FPS on Pixel 4 GPU, ~50FPS on Pixel 3 single-core CPU with XNNPACK inference, ~145FPS on Pixel 4 CPU with XNNPACK).

For each detected face, returns:

- Facial bounding box coordinates
- 6 approximate facial keypoint coordinates:
  - Left eye (from the observer's point of view)
  - Right eye
  - Nose tip
  - Mouth
  - Left eye tragion
  - Right eye tragion
- Detection confidence score

MODEL SPECIFICATIONS

Model Type
Convolutional Neural Network

Model Architecture
Convolutional Neural Network: CenterNet-like with a custom encoder.

Input(s)
RGB image (possibly a video frame) resized to 160x192 pixels, represented as a 160x192x3 array of float values in the range [-1.0, 1.0].
Output(s)
Tensor of predicted embeddings representing anchors transformation which are further used in Non Maximum Suppression algorithm.

AUTHORS

Who created this model?
Valentin Bazarevsky, Google

Who provided the model card?
Yury Kartynnik, Google

DATE

January 19, 2021

CITATION

How can users cite your model?
V. Bazarevsky et al. BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs. CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Long Beach, CA, USA, 2019.

Intended Uses

APPLICATION

Detecting prominently displayed faces within images or videos captured by a smartphone camera.

DOMAIN AND USERS

- Live perception pipelines
- Mobile AR (augmented reality) applications
- User interface enhancements (e.g. Google Meet auto-zoom, AutoFlip)

OUT-OF-SCOPE APPLICATIONS

- Counting the number of people in a crowd
- Detecting faces looking away from the camera, significantly inclined from the vertical orientation, or individuals' back of the head
- Detecting people too far away from the camera (e.g. **further than 5 meters**)
- Any form of surveillance or identity recognition is explicitly out of scope and not enabled by this technology

Limitations

PRESENCE OF ATTRIBUTES

Produces only up to a given limit (e.g. 10) of detections even if more people are present.

TRADE-OFFS

The model is optimized for real-time performance on a wide variety of mobile devices, but is sensitive to face position, scale and orientation in the input image.

ENVIRONMENT

In presence of degrading environment light, noise, motion or face overlapping conditions one can expect degradation of quality and increase of "jittering" (although we cover such cases during training with real-world samples and augmentations).

Ethical Considerations

HUMAN LIFE

The model is not intended for human life-critical decisions. The primary intended application is entertainment and assistive technologies.

PRIVACY

This model was trained and evaluated on consented images of people using a mobile AR application captured with smartphone cameras in various "in the wild" conditions.

BIAS

The model has been trained on images captured with smartphone cameras. While these images have significant variability, please consider whether your use case is significantly

Model Cards + RDF

- Example SPARQL queries (using FAIRnets ontology: “nno”)

```
SELECT ?label ?category WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        nno:hasIntendedUse “nlp”;
        doap:category ?category.
FILTER(REGEX(LCASE(STR(?category)), “sports”))
}
```

```
SELECT ?label ?creator ?link (COUNT(?layer)) WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        dc:creator ?creator;
        nno:hasRepository ?link;
        nno:hasModel ?model.
?model nno:hasLayer ?layer.
}
```

```
SELECT ?label ?link ?emission WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        nno:hasRepository ?link;
                temp:co2emissions ?emission.
} ORDER BY ?emission
```

Software support for Model Cards has started to show up in frameworks

Model Cards + RDF (cont'd)

- Example SPARQL queries (extending FAIRnets)

For bias control

```
SELECT ?label ?identifiable WHERE {
?network a nno:NeuralNetwork;
        rdfs:label ?label;
        temp:trainingDS ?ds;
        doap:category ?category.
?ds a temp:DataSet;
   temp:hasPPI: ?identifiable.
}
```

For mixed precision

```
SELECT ?label  WHERE {
?network a nno:NeuralNetwork;
        nno:hasModel ?model.
?model a nno:Model;
                nno:hasLayer
          <https://onnx.org/QuantizedLinear>.
}
```

Future Thoughts for consideration

- Should we establish a model lineage tracker for QAT & finetuning scenarios
  - Custom datasets used to generate a model by fine-tuning & QAT flows
  - Accuracy trends as models progress through the lineage

- Tracking hash/checksum of models progression
  - Any considerations for 3rd party verifications (CCF, ledgers, etc..)
  - Potential monetization avenue for model creators and their customized datasets

Backup

The Semantic Web infrastructure (RDF)

```
@prefix schema: <http://schema.org/>
.
@prefix gndo: <https://d-nb.info/standards/elementset/gnd#> .
@prefix lib: <http://purl.org/library/> .
@prefix marcRole: <http://id.loc.gov/vocabulary/relators/> .
@prefix dcmitype: <http://purl.org/dc/dcmitype/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix geo: <http://www.opengis.net/ont/geosparql#> .

<https://d-nb.info/gnd/1045328480> a gndo:BuildingOrMemorial ;
        foaf:page <http://de.wikipedia.org/wiki/The_Shard> ;
        gndo:gndIdentifier "1045328480" ;
        gndo:geographicAreaCode <https://d-nb.info/standards/vocab/gnd/geographic-area-code#XA-GB> ;
        gndo:definition "72-stöckiges u. 310 m hohes, multifunktionales Hochhaus am Südufer"@de ;
        gndo:dateOfProduction "16.03.2009-01.02.2013" ;
        gndo:preferredNameForThePlaceOrGeographicName "The Shard (London)" .
```

Controlled vocabularies for Provenance, Explainable and Ethical ML are already being developed (eg IEEE P7003)

The Semantic Web infrastructure (RDF)

```
@prefix schema: <http://schema.org/>
.
@prefix gndo: <https://d-nb.info/standards/elementset/gnd#> .
@prefix lib: <http://purl.org/library/> .
@prefix marcRole: <http://id.loc.gov/vocabulary/relators/> .
@prefix dcmitype: <http://purl.org/dc/dcmitype/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix geo: <http://www.opengis.net/ont/geosparql#> .

<https://d-nb.info/gnd/1045328480> a gndo:BuildingOrMemorial ;
        foaf:page <http://de.wikipedia.org/wiki/The_Shard> ;
        gndo:gndIdentifier "1045328480" ;
        gndo:geographicAreaCode <https://d-nb.info/standards/vocab/gnd/geographic-area-code#XA-GB> ;
        gndo:definition "72-stöckiges u. 310 m hohes, multifunktionales Hochhaus am Südufer"@de ;
        gndo:dateOfProduction "16.03.2009-01.02.2013" ;
        gndo:preferredNameForThePlaceOrGeographicName "The Shard (London)" .
```

Controlled vocabularies for Provenance, Explainable and Ethical ML are already being developed (eg IEEE P7003)

RDF alternative serialization: LD-JSON

```
<script type="application/ld+json">
{
 "@context":"http://schema.org",
 "@type":"NewsArticle",“
 description":"Don’t fall for the Trump infrastructure scam.",
 mainEntityOfPage":"https://www.nytimes.com/2016/11/21/opinion/build-he-wont.html",
 "url":"https://www.nytimes.com/2016/11/21/opinion/build-he-wont.html",
 "author":[{
  "@context":"http://schema.org",“
  @type":"Person",“
  url":"https://www.nytimes.com/by/paul-krugman",
  "name":"Paul Krugman"}
 ],
 “dateModified":"2016-11-21T16:31:49.000Z",
 "datePublished":"2016-11-21T08:21:07.000Z“
}
</script>
```

Can be embedded in HTML pages and readily processed in Javascript

RDF alternative serialization: yaml embedded in Markdown

---

description: Use this topic to help manage Windows and Windows Server technologies with Windows PowerShell.

Download Help Link: https://aka.ms/winsvr-2022-pshelp

Help Version: 5.0.2.1

Locale: en-US

Module Guid: af4bddd0-8583-4ff2-84b2-a33f5c8de8a7

Module Name: Hyper-V

ms.date: 12/20/2016

title: Hyper-V

Hyper-V Module

Description

This reference provides cmdlet descriptions and syntax for all Hyper-V-specific cmdlets. It lists the cmdlets in alphabetical order based on the verb at the beginning of the cmdlet.

Can be embedded (unobstrusively) in the Markdown for Model Cards

The Semantic Web infrastructure (SPARQL)

```
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX p: <http://www.wikidata.org/prop/>
PREFIX ps: <http://www.wikidata.org/prop/statement/>
PREFIX pq: <http://www.wikidata.org/prop/qualifier/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT ?laureateName ?awardYear ?warName
WHERE {
  ?laureate p:P166 ?award .          # Winner of some prize
  ?award ps:P166 wd:Q37922 .         # Prize is Nobel Pr. in Lit.
  ?award pq:P585 ?awardDate .        # Get the date of the award
  BIND(YEAR(?awardDate) AS ?awardYear) # Get the year of the award
  ?laureate wdt:P607 ?war .          # Find war(s) laureate was in
  ?war rdfs:label ?warName .
  FILTER(LANG(?warName)="en"         # Only English labels
  && LANG(?laureateName)="en")       #  ...  names only
} ORDER BY ?awardYear ?warStart      # Oldest award (then war) first
```

ONNX APIs can be provided to extract and query (SPARQL) semantic content. Queries useful in provenance and fairness checks can be made available.

Agenda

Roadmap proposal (in October ‘21) covered the following aspects:

- Model metadata for identifying “Models with Mixed Precision”
- Machine readable. format for Model metadata to track provenance and other relevant characteristics
  - Model details, Datasets, Applicable areas, Fairness, Transparency, Human centered approach, Trusted, Secure, Privacy protections, ..
- Machine readable format for model metadata to track provenance and other relevant characteristics
  - Facilitates metadata checks to be implemented in runtime(s) as part of Preprocessing element of ONNX Graph
  - Enables queries to be performed on model zoo to filter for relevant use cases

Discussion Today :

- Define the relevant model metadata fields (first round collection)
- Discuss specific machine readable format & query language for adoption

Thank You !!