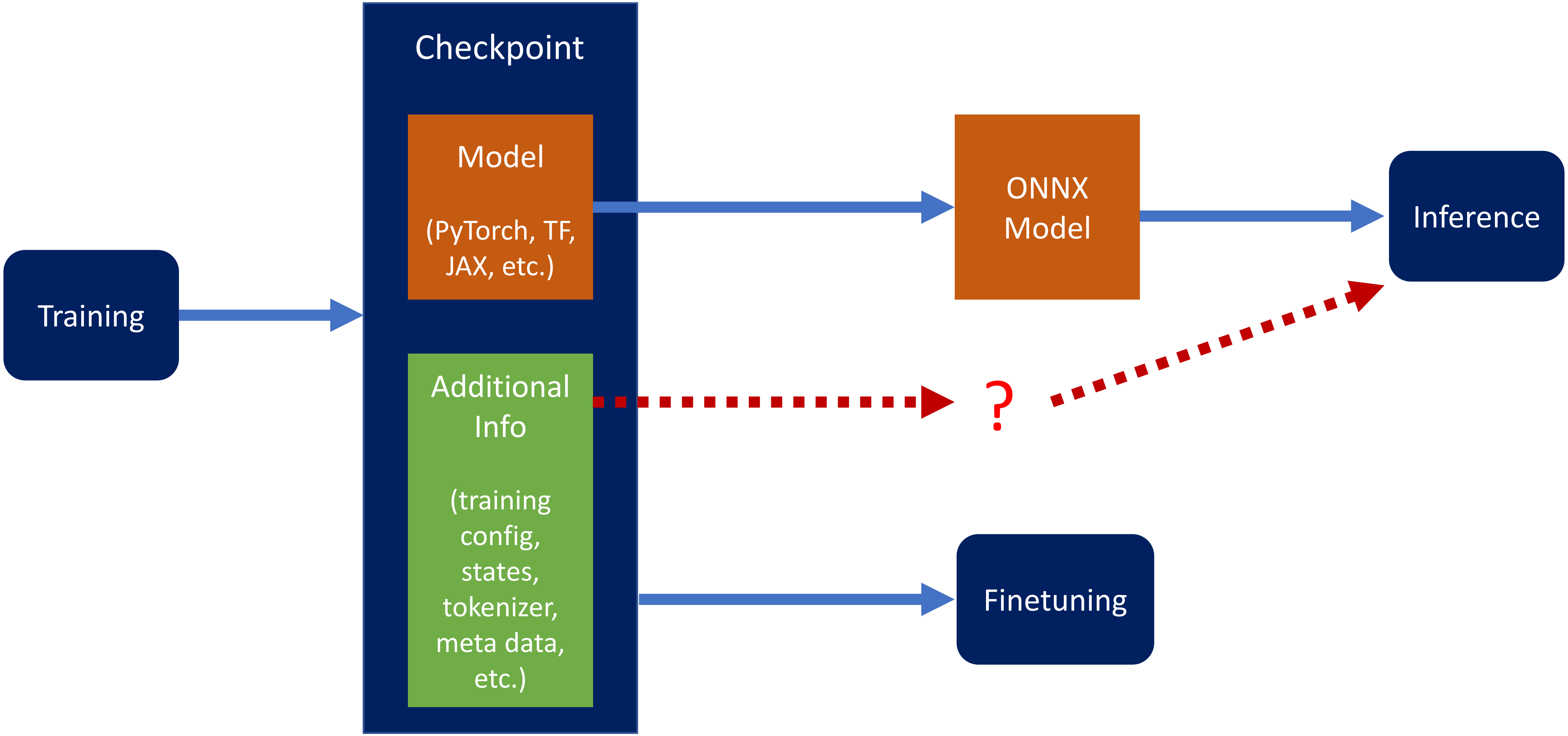
ONNX Proposal

Updating TrainingInfoProto Format for more comprehensive training information

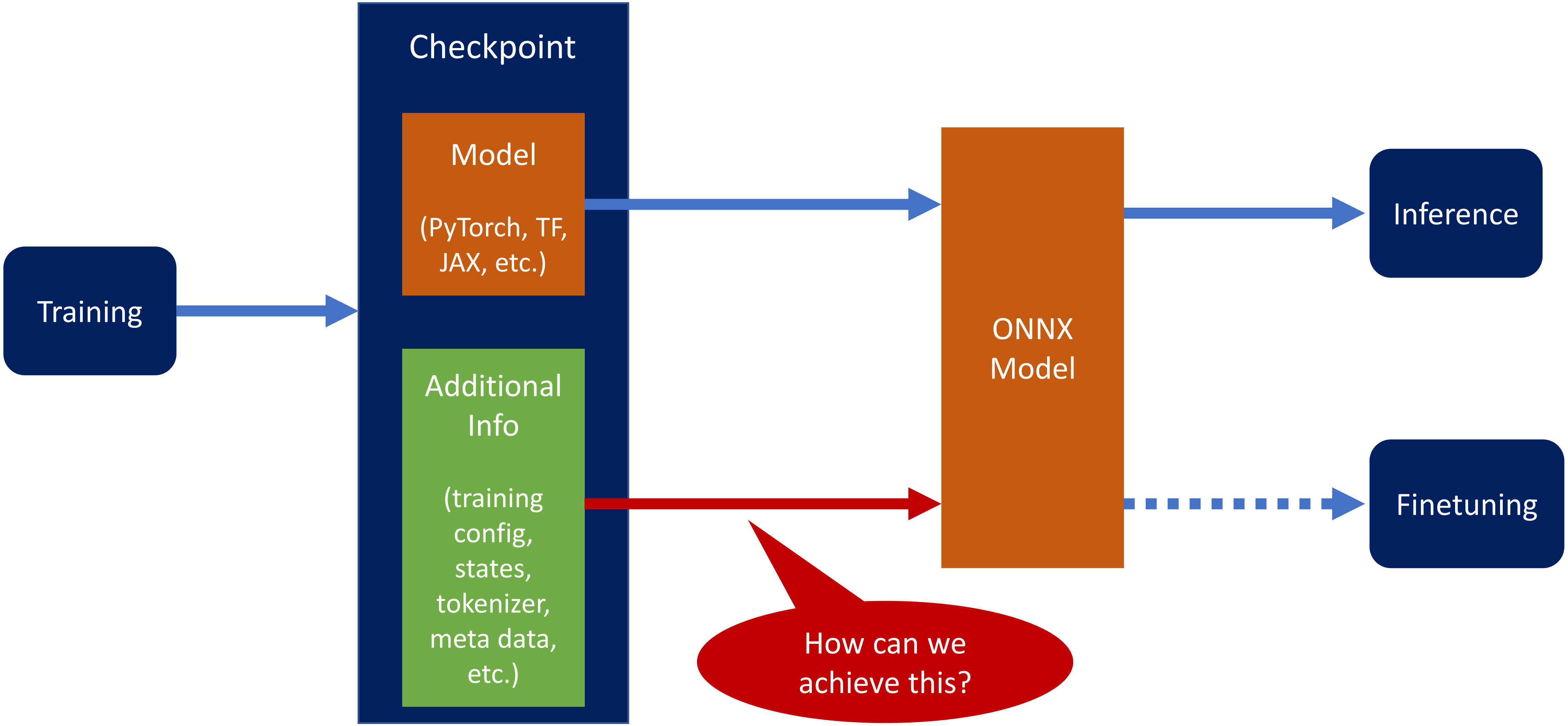
March 29, 2023

Taka Shinagawa
Microsoft

Motivation – Current State



Motivation – Future Desired State



Training

Checkpoint

Model

(PyTorch, TF, JAX, etc.)

Additional Info

(training config, states, tokenizer, meta data, etc.)

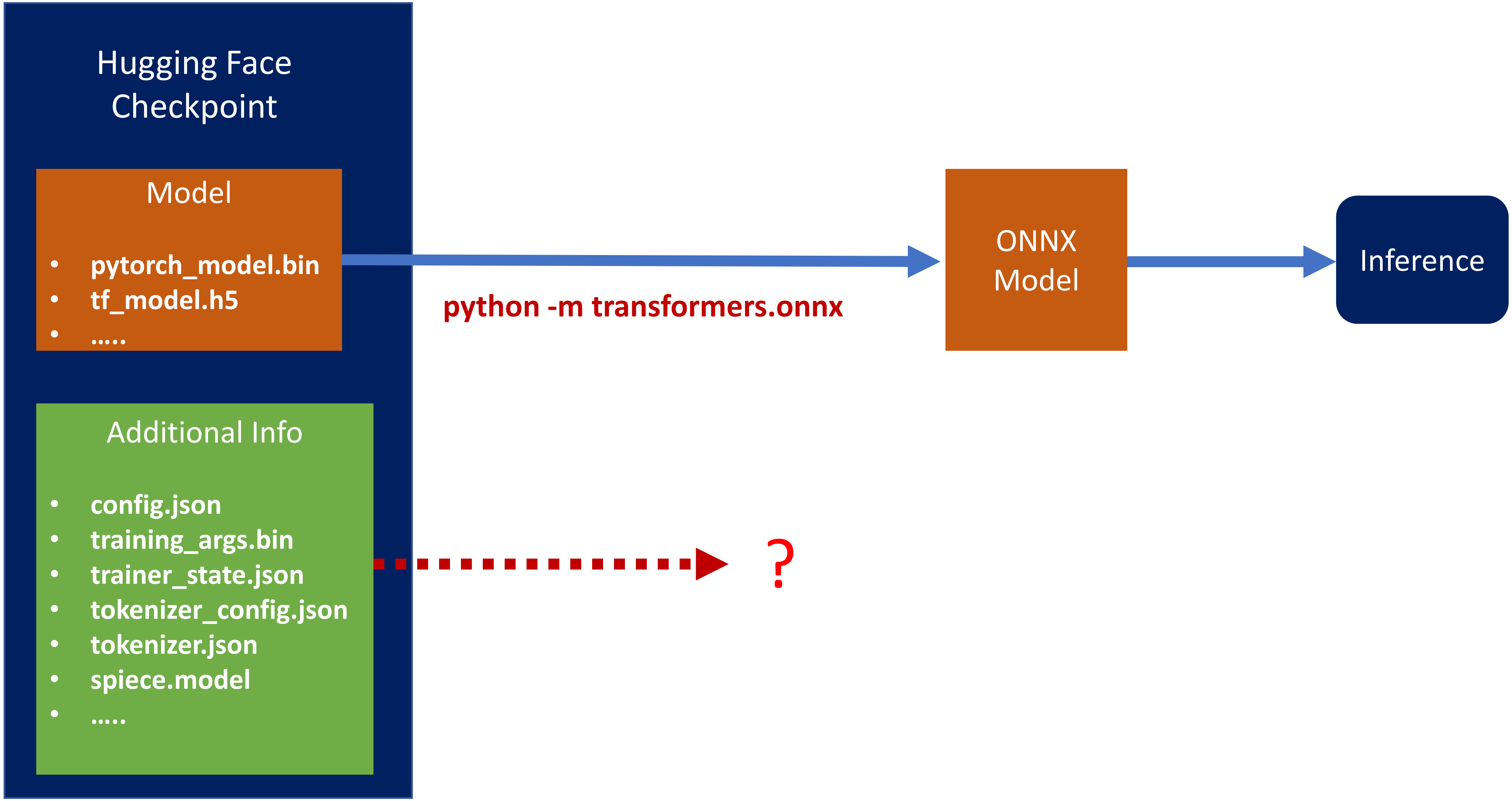
ONNX Model

Inference

Finetuning

How can we achieve this?

Hugging Face Checkpoint



Hugging Face Checkpoint

Model

- pytorch_model.bin
- tf_model.h5
- …..

Additional Info

- config.json
- training_args.bin
- trainer_state.json
- tokenizer_config.json
- tokenizer.json
- spiece.model
- …..

python -m transformers.onnx

ONNX Model

Inference

?

ONNX Model Format (onnx.ModelProto)

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.

onnx/IR.md at main · onnx/onnx (github.com)

TrainingInfoProto Format (Current)

- This field describes a graph to compute the initial tensors upon starting the training process.
- Initialization graph has no input and can have multiple outputs.
- Usually, trainable tensors in neural networks are randomly initialized.
- To achieve that, for each tensor, the user can put a random number operator such as RandomNormal or RandomUniform in TrainingInfoProto.initialization.node and assign its random output to the specific tensor using "initialization_binding".
- This graph can also set the initializers in "algorithm" in the same TrainingInfoProto; a use case is resetting the number of training iteration to zero.

Name	Type	Description
initialization	GraphProto	This field describes a graph to compute the initial tensors upon starting the training process.
algorithm	GraphProto	This field represents a training algorithm step. Given required inputs, it computes outputs to update initializers in its own or inference graph’s initializer lists.
initialization_binding	StringStringEntryProto	This field specifies the bindings from the outputs of "initialization" to some initializers in "ModelProto.graph.initializer" and the "algorithm.initializer" in the same TrainingInfoProto.
update_binding	StringStringEntryProto	This field usually contains names of trainable tensors (in ModelProto.graph), optimizer states such as momentums in advanced stochastic gradient methods (in TrainingInfoProto.graph), and number of training iterations (in TrainingInfoProto.graph).

onnx/onnx.proto at main · onnx/onnx (github.com)

GraphProto Format (Current)

- A graph defines the computational logic of a model and is comprised of a parameterized list of nodes that form a directed acyclic graph based on their inputs and outputs.
- This is the equivalent of the "network" or "graph" in many deep learning frameworks.

Name	Type	Description
node	NodeProto	The nodes in the graph, sorted topologically.
name	string	The name of the graph
initializer	TensorProto	A list of named tensor values, used to specify constant inputs of the graph.
sparse_initializer	SparseTensorProto	Initializers stored in sparse format.
doc_string	string	A human-readable documentation for this graph.
input	ValueInfoProto	The inputs of the graph.
output	ValueInfoProto	The outputs of the graph.
value_info	ValueInfoProto	Information for the values in the graph.
quantization_annotation	TensorAnnotation	This field carries information to indicate the mapping among a tensor and its quantization parameter tensors.

[onnx/onnx.proto at main · onnx/onnx (github.com)]

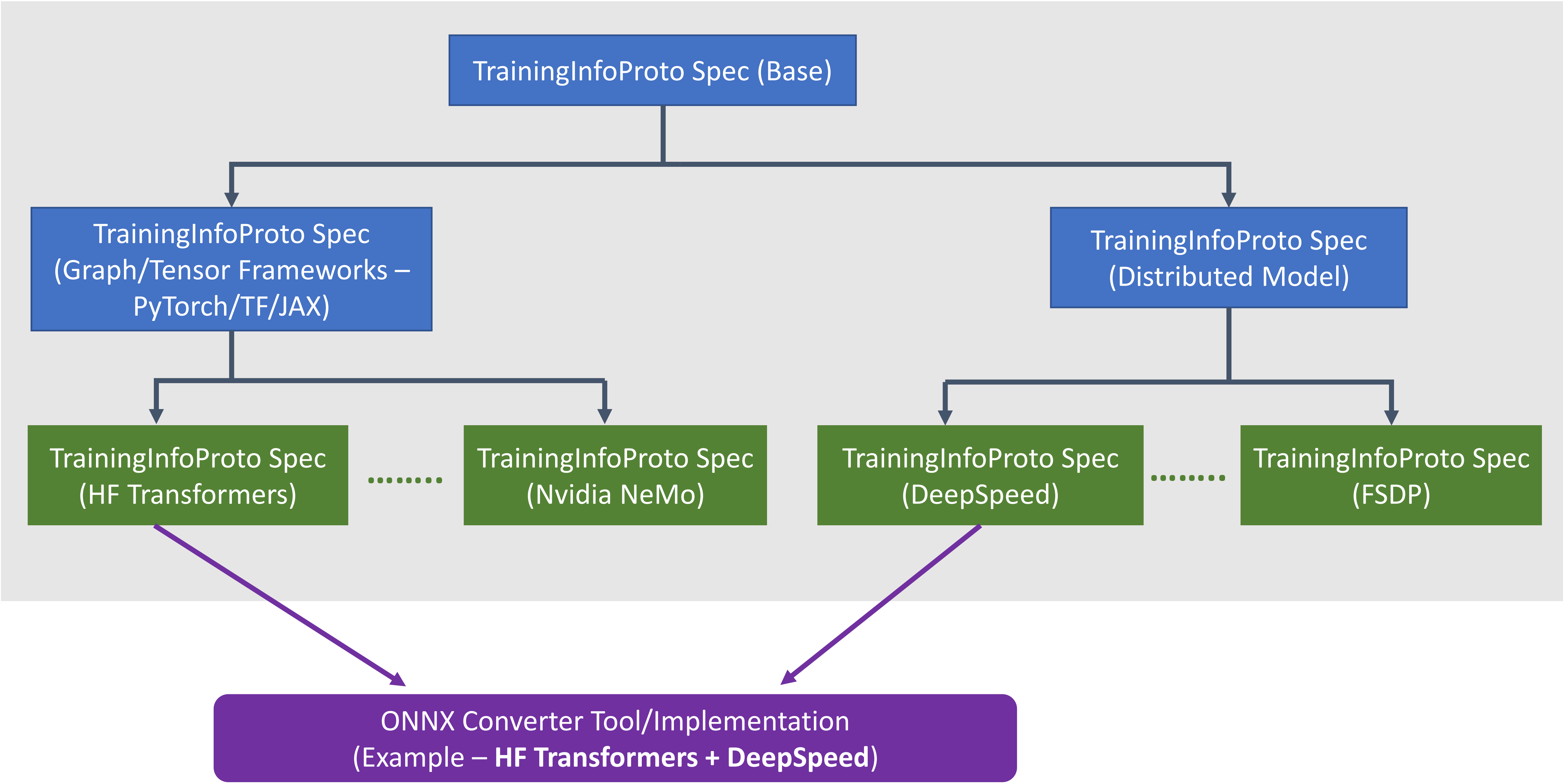
Challenges with the Current TrainingInfoProto

TrainingInfoProto was added to ONNX in 2019 so that users know how to apply the specified optimization algorithm to conduct further training iterations. However, in order to allow finetuning of ONNX models, ONNX files need to store more complete training information from upstream training checkpoints in many cases.

- No (or very few) framework utilizes this field currently
  - Any reference implementation?
- Designed before the transformer/LLM frameworks such as Hugging Face Transformers and Nvidia NeMo
- No support for model parallel training frameworks such as DeepSpeed

onnx/onnx.proto at main · onnx/onnx (github.com)

Proposal – A Design Idea for New TrainingInfoProto Spec


TrainingInfoProto Spec (Base)
TrainingInfoProto Spec (Graph/Tensor Frameworks – PyTorch/TF/JAX)
TrainingInfoProto Spec (Distributed Model)
TrainingInfoProto Spec (HF Transformers)
....
TrainingInfoProto Spec (Nvidia NeMo)
TrainingInfoProto Spec (DeepSpeed)
....
TrainingInfoProto Spec (FSDP)
ONNX Converter Tool/Implementation
(Example – **HF Transformers + DeepSpeed**)

Checkpoint / Model Formats for Investigation

- Hugging Face (transformers, accelerate, etc.)
- Tensorflow
  - Tensorflow Distributed
- PyTorch
- JAX
- Nvidia NeMo

Distributed Training Frameworks

- DeepSpeed
- FSDP (PyTorch Fully Sharded Parallel)
- PyTorch Lightning
- Ray
- ORT Training