

## b-hive:

a modular training framework for state-of-the-art object-tagging within the Python ecosystem at the CMS experiment

Niclas Eich




On behalf

of the CMS Collaboration

[niclas.eich@rwth-aachen.de](mailto:niclas.eich@rwth-aachen.de)

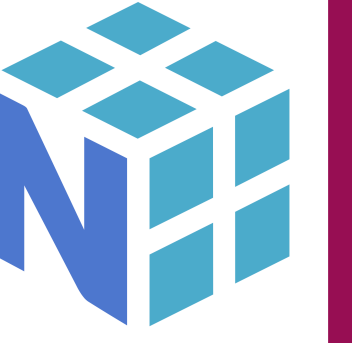
### Motivation:

Many working-groups have shared needs for ML training-tools!

- Software Framework for ML-tasks in HEP
- State-of-the-art object-tagging (b/c jets, taus)
- Based entirely on Python
- Reproducible pipelines with 

### Dataset Construction:

- Convert ROOT files into .npz files
- coffea and awkward for the big data processing
- Weight-calculation for stochastic sampling



Awkward Array

### Training:

- API defined with abstract base classes
- Any ML-library useable
- Task is agnostic of training details  
→ Applications implement their specifics
- Inheritance for fast development



```

42 class Classifier(ABC):
43     @abstractmethod
44     def train_model(
45         self,
46         training_data,
47         validation_data,
48         output_dir,
49         nepochs=0,
50         resume_epochs=0,
51         **kwargs,
52     ):
53         pass
54
55     @abstractmethod
56     def validate_model(self, dataloader, verbose=True,):
57         pass
58
59     @abstractmethod
60     def predict_model(self, dataloader):
61         pass
62
63

```

### Modularity:

Steering complex applications with many parameters leads to a big mess

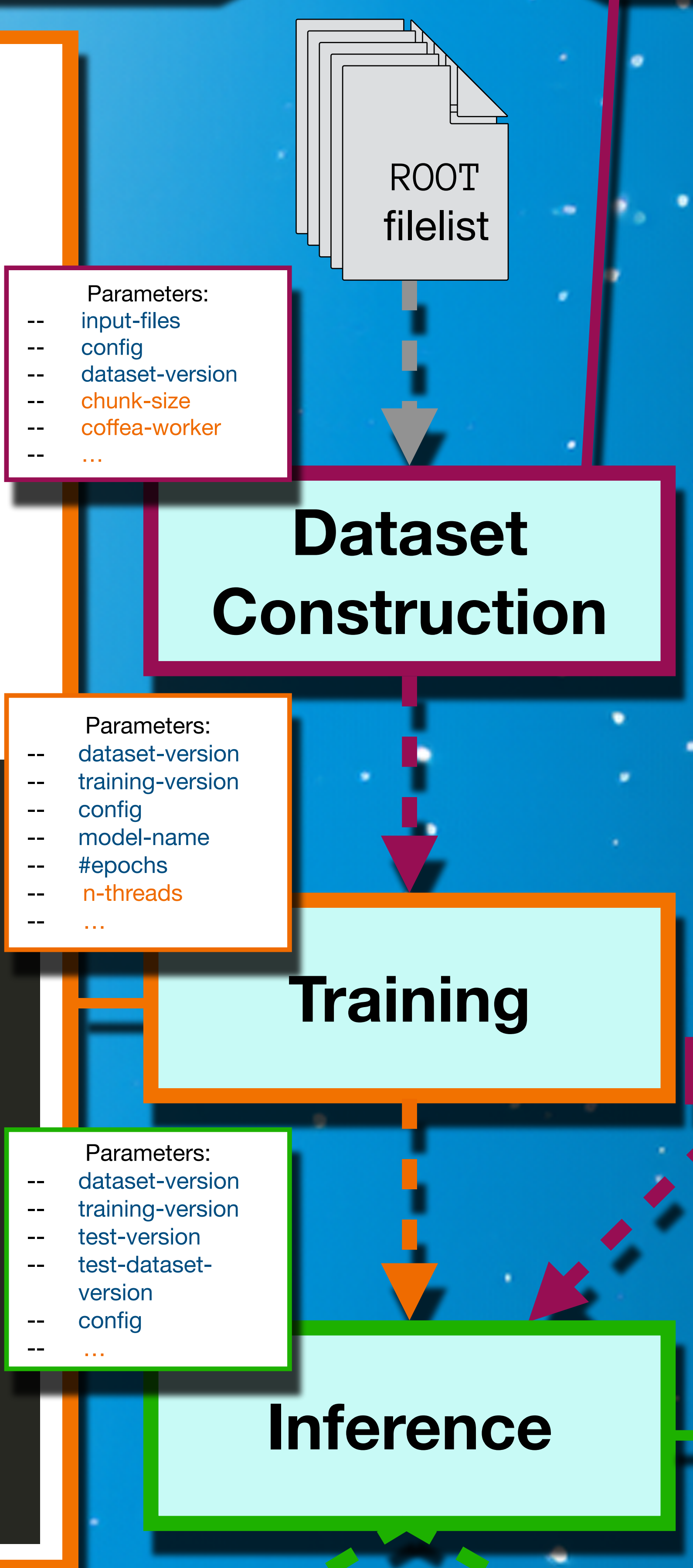
→ Provide a modular setup to split apart code for different applications

- coffea-processors
- Models
- Dataloaders

```

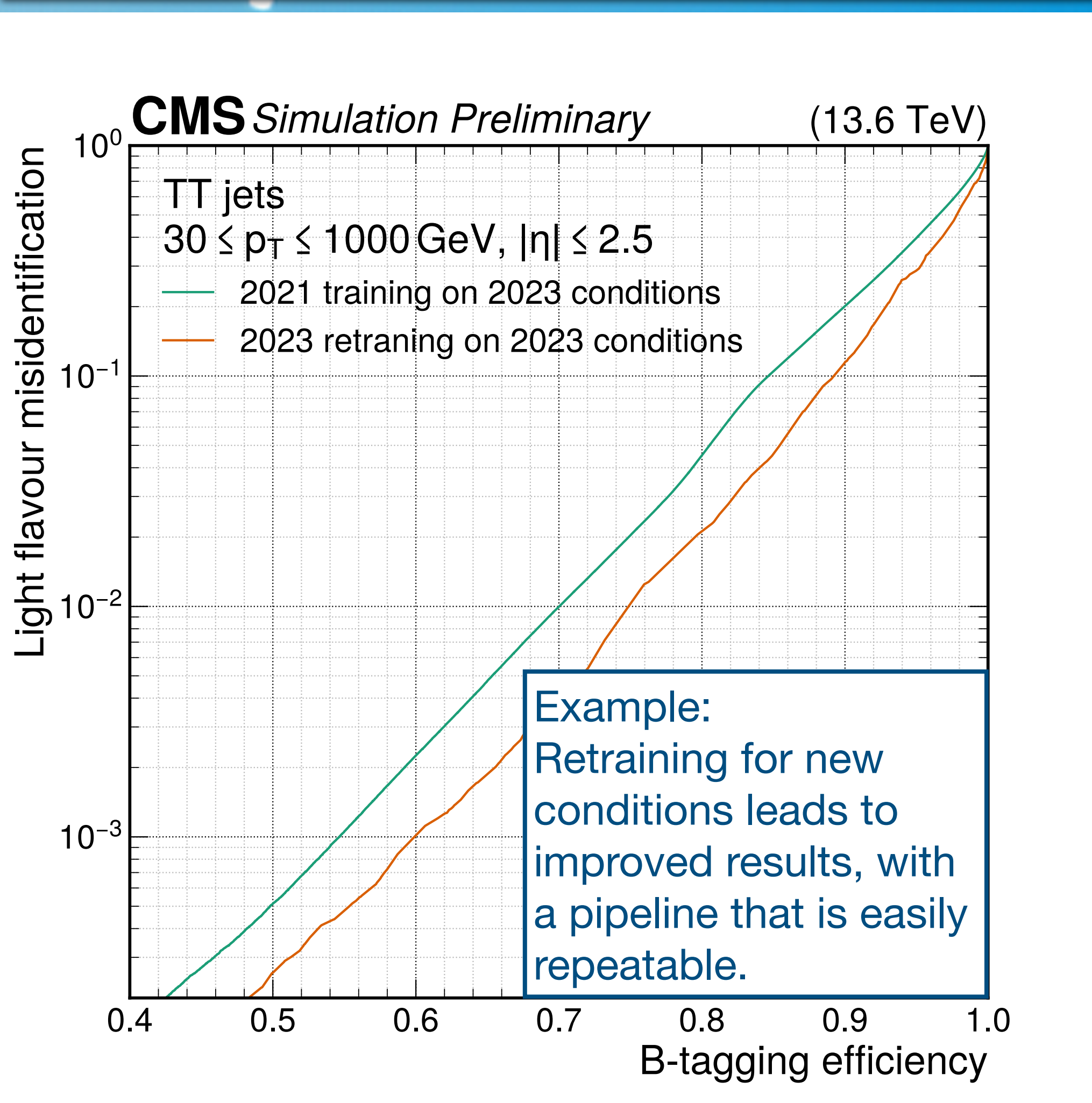
42 def ProcessorLoader(model: str = "", *args, **kwargs):
43     match model:
44         case ProcessorClasses.PFCandidateAndVertexProcessing:
45             return PFCandidateAndVertexProcessing(*args, **kwargs)
46         case ProcessorClasses.L1TriggerProcessor:
47             return L1TriggerProcessor(*args, **kwargs)
48         case _:
49             return DefaultProcessor(*args, **kwargs)

```



### Inference:

- Models can be easily evaluated on different datasets
- Task tree is built to reuse existing datasets, models, ...



### Evaluation:

- Automatic standardised evaluation tools for comparisons
- Production of ROC-curves (left), working point curves (right)
- Application specific tests can be added as tasks

