ESCAPE

European Science Cluster of Astronomy & Particle physics ESFRI research Infrastructures

# CTLearn

## Deep learning for IACT event reconstruction

D. Nieto, J.L. Contreras, T. Miener et al.

Universidad Complutense de Madrid

E-OSSR Onboarding Presentation

December 1, 2022

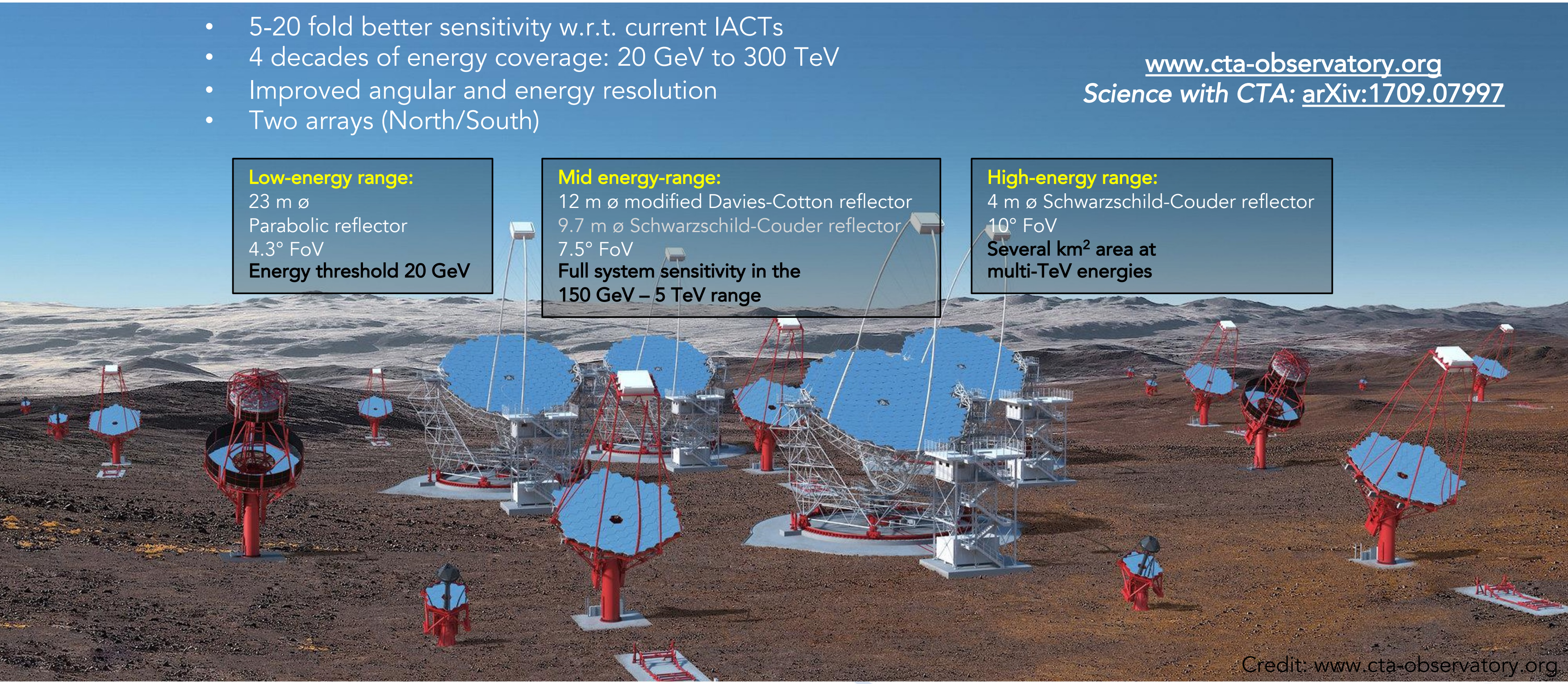
ESCAPE - The European Science Cluster of Astronomy & Particle Physics ESFRI Research Infrastructures has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement n° 824064.

# Introduction

the observatory for
ground-based
gamma-ray astronomy

- 5-20 fold better sensitivity w.r.t. current IACTs
- 4 decades of energy coverage: 20 GeV to 300 TeV
- Improved angular and energy resolution
- Two arrays (North/South)

www.cta-observatory.org
*Science with CTA:* arXiv:1709.07997



**Low-energy range:**
23 m ø
Parabolic reflector
4.3° FoV
**Energy threshold 20 GeV**

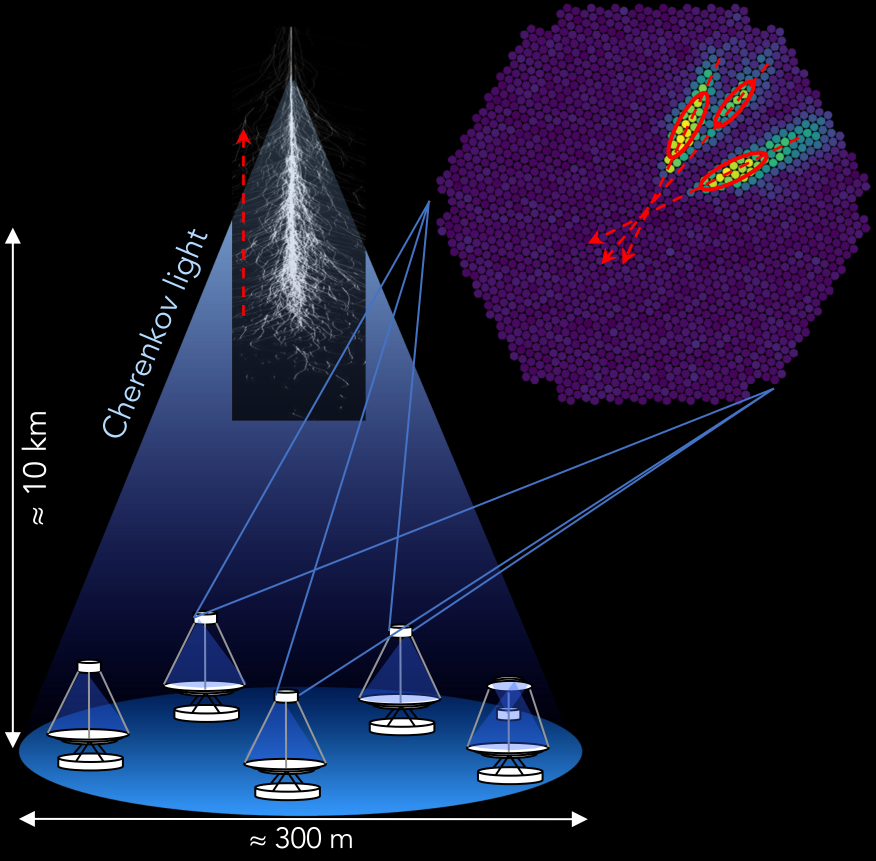
**Mid energy-range:**
12 m ø modified Davies-Cotton reflector
9.7 m ø Schwarzschild-Couder reflector
7.5° FoV
**Full system sensitivity in the 150 GeV – 5 TeV range**

**High-energy range:**
4 m ø Schwarzschild-Couder reflector
10° FoV
**Several km$^2$ area at multi-TeV energies**

Credit: www.cta-observatory.org

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Introduction



- Detection of extended air showers using the atmosphere as a calorimeter
- Huge g–ray collection area (~$10^5$ $m^2$)
- Large background from charged CR
  - Partly irreducible ($e^-$/$e^+$ , single-EM, with current methods)
- Energy window: tens GeV - tens TeV
- Event reconstruction from image:
  - Type of primary event
  - Primary energy estimation
  - Primary arrival direction
- Event reconstruction algorithms
  - Look-up-tables
  - Geometric methods
  - Fit to templates
  - Classic machine learning (RFs, BDTs)
  - **Deep learning?**

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Introduction

CTLearn

Core developers
Tjark Miener, DN (I**PARCOS-UCM**)
Ari Brill, Qi Feng (Columbia)
Bryan Kim (UCLA, now at Meta)
(See contributors here)

- High-level Python package for using deep learning for IACT event reconstruction
- Configuration-file-based workflow and installation with conda drive reproducible training and prediction
- Supports any TensorFlow model that obeys a generic signature

Use case:
“User wants to use a new deep-learning architecture for reconstructing events from her latest IACT observations”

Workflow:

- User provides IACT MC dataset and model
- CTLearn trains model & provides evaluation metrics
- User provides dataset for inference (MC or real data)
- CTLearn performs inference (event reconstruction)

# Software/Service Development

- Development:
  - “Fork-and-pull” Git workflow
  - Semantic versioning
  - Code style enforced by Flake8
  - Sphinx-based documentation hosted at Read the Docs
- Testing and efficiency optimization strategies
  - Continuous integration via GitHub actions
  - Unit testing to be implemented
  - Benchmarking performed before every release
- Platform integration and metadata
  - GitHub and Zenodo
  - Codemeta
- Software licenses
  - BSD 3-Clause "New" or "Revised" License
- General guidelines that are followed
  - Available on GitHub for all potential users/contributors

Funded by the European Union’s
Horizon 2020 - Grant N° 824064

# Software/Service Requirements

- Operating System, compilation environment
  - Works on Linux and MacOS
  - Python-based, TensorFlow as backend engine
  - Installation via conda/pip
- Hardware requirements
  - GPU-enabled hosts strongly recommended but not required
  - Only tested on NVidia GPUs
- Containerization and portability requirements
  - Work in progress towards containerization using Singularity
- Workflow / interface requirements to other software/services
  - Input assumes official CTA data format
  - Inference to be integrated into CTA's analysis pipeline

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# OSSR Integration

- What is available?
  - Code repository on GitHub
  - pip package
  - Documentation on ReadTheDocs
- What will be onboarded?
  - Source code
- Are there open points and requirements?
  - None identified, project already on Zenodo

Funded by the European Union's
Horizon 2020 - Grant N° 824064

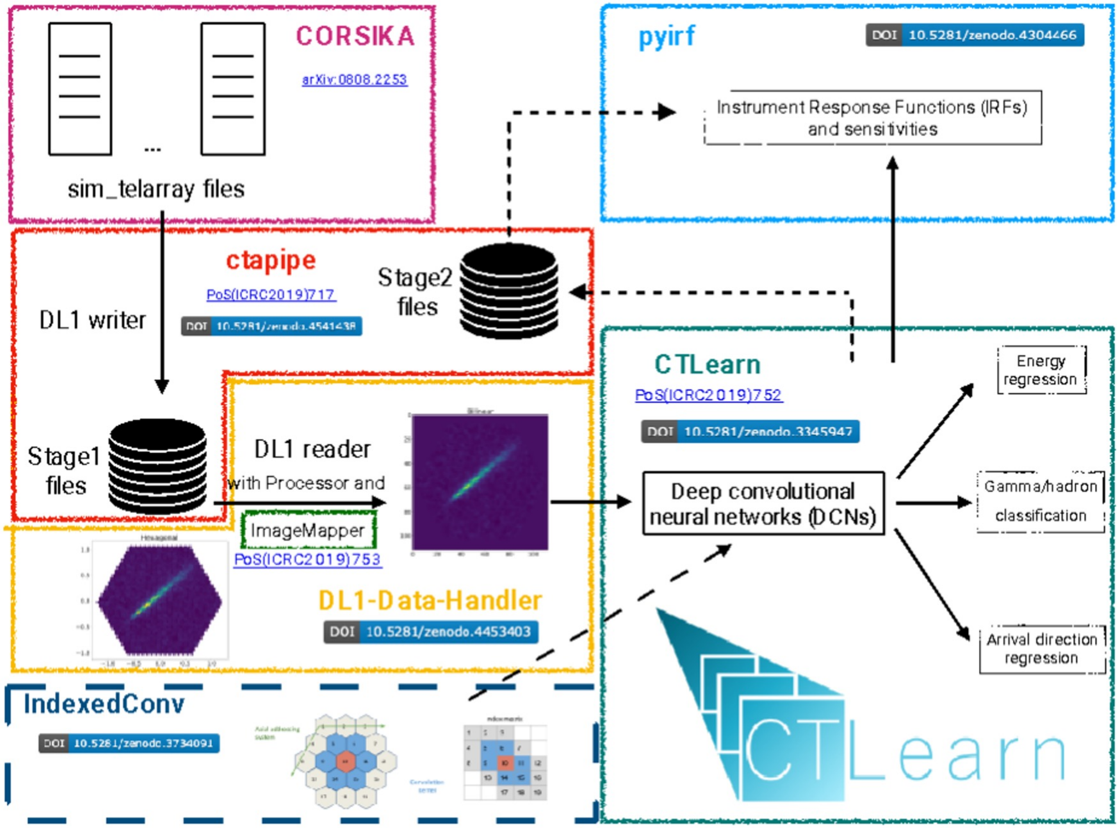
# OSSR Integration

- EOSC user story
  - User wants to train a given DL model for her IACT data analysis
  - OSSR side
    - User finds links to code, documentation and examples
    - CTA datasets for training and testing may be available (WIP)
  - Data side
    - User can utilize her own dataset for training (simulations) and testing (simulation / real data)
    - Custom plugin may be needed if not using CTA's standard format

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

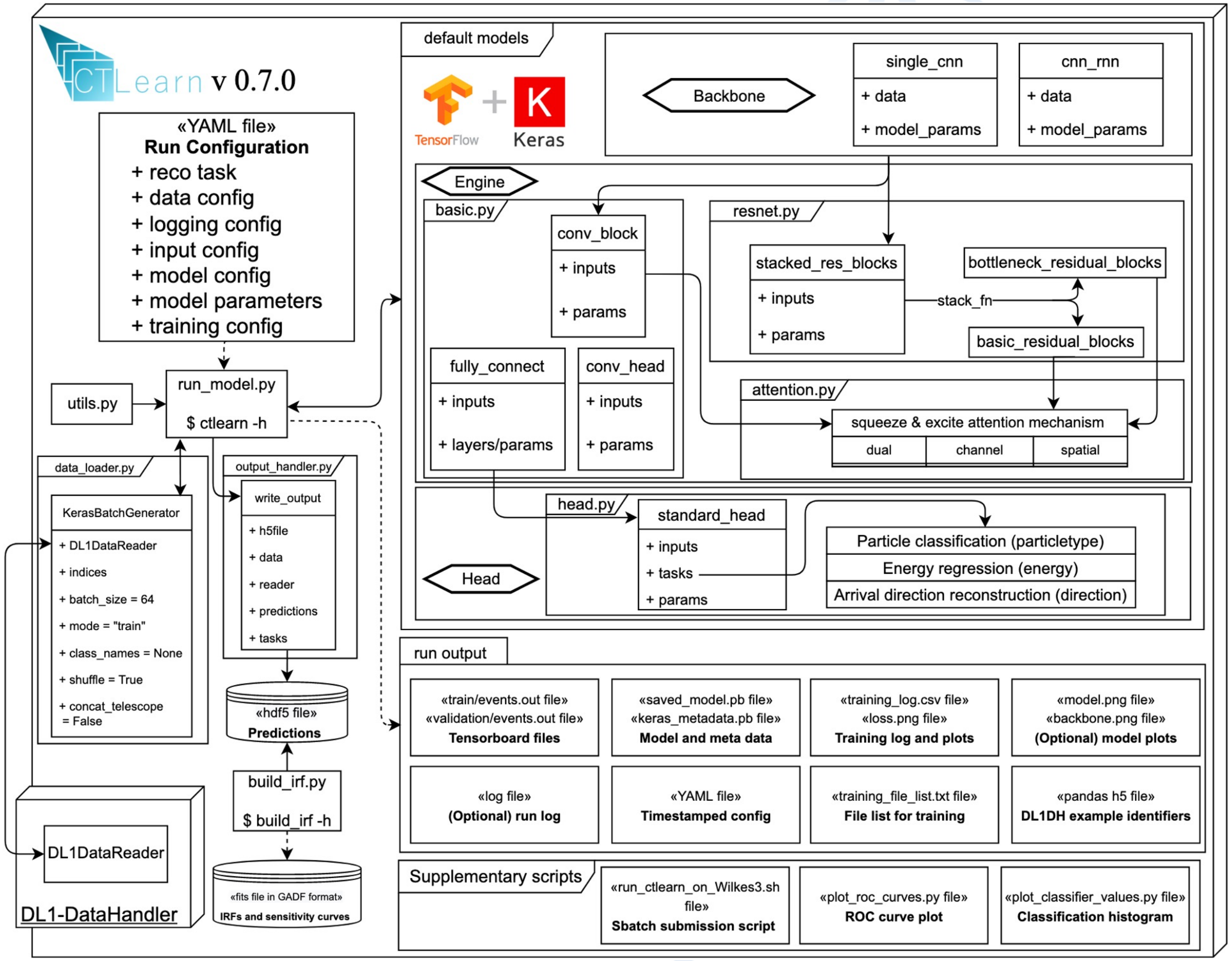
- Standard ecosystem


CORSIKA
arXiv:0808.2253
...
sim_telarray files
pyirf
DOI 10.5281/zenodo.4304466
Instrument Response Functions (IRFs) and sensitivities
ctapipe
PoS(ICRC2019)717
DOI 10.5281/zenodo.4541438
DL1 writer
Stage2 files
Stage1 files
CTLearn
PoS(ICRC2019)752
DOI 10.5281/zenodo.3345947
Energy regression
DL1 reader with Processor and
ImageMapper
PoS(ICRC2019)753
Deep convolutional neural networks (DCNs)
Gamma/hadron classification
DL1-Data-Handler
DOI 10.5281/zenodo.4453403
Arrival direction regression
IndexedConv
DOI 10.5281/zenodo.3734091
CTLearn

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Project architecture



CTLearn v 0.7.0

«YAML file»
**Run Configuration**
+ reco task
+ data config
+ logging config
+ input config
+ model config
+ model parameters
+ training config

utils.py

run_model.py
$ ctlearn -h

data_loader.py
KerasBatchGenerator
+ DL1DataReader
+ indices
+ batch_size = 64
+ mode = "train"
+ class_names = None
+ shuffle = True
+ concat_telescope = False

output_handler.py
write_output
+ h5file
+ data
+ reader
+ predictions
+ tasks

«hdf5 file»
**Predictions**

build_irf.py
$ build_irf -h

«fits file in GADF format»
**IRFs and sensitivity curves**

DL1DataReader
DL1-DataHandler

default models
TensorFlow + Keras

Backbone

single_cnn
+ data
+ model_params

cnn_rnn
+ data
+ model_params

Engine

basic.py
conv_block
+ inputs
+ params

fully_connect
+ inputs
+ layers/params

conv_head
+ inputs
+ params

resnet.py
stacked_res_blocks
+ inputs
+ params

stack_fn

bottleneck_residual_blocks

basic_residual_blocks

attention.py
squeeze & excite attention mechanism
dual | channel | spatial

Head

head.py
standard_head
+ inputs
+ tasks
+ params

Particle classification (particletype)
Energy regression (energy)
Arrival direction reconstruction (direction)

run output

«train/events.out file»
«validation/events.out file»
**Tensorboard files**

«saved_model.pb file»
«keras_metadata.pb file»
**Model and meta data**

«training_log.csv file»
«loss.png file»
**Training log and plots**

«model.png file»
«backbone.png file»
**(Optional) model plots**

«log file»
**(Optional) run log**

«YAML file»
**Timestamped config**

«training_file_list.txt file»
**File list for training**

«pandas h5 file»
**DL1DH example identifiers**

Supplementary scripts

«run_ctlearn_on_Wilkes3.sh file»
**Sbatch submission script**

«plot_roc_curves.py file»
**ROC curve plot**

«plot_classifier_values.py file»
**Classification histogram**

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

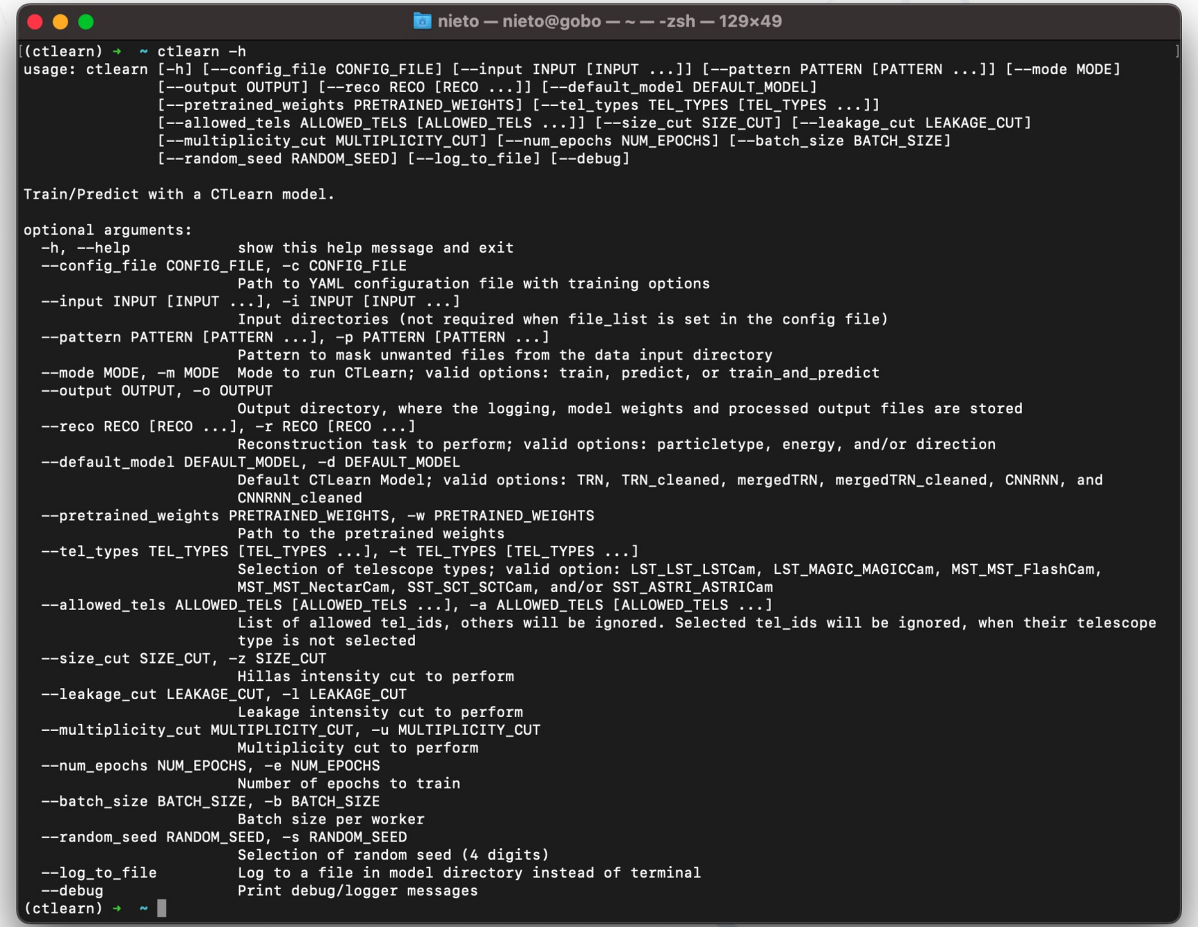
- Installation for end users

The following command lines will set up a conda virtual environment, add the necessary package channels, and install CTLearn specified version and its dependencies:

```
> CTLEARN_VER=0.7.0

> wget https://raw.githubusercontent.com/ctlearn-project/ctlearn/v$CTLEARN_VER/environment.yml

> conda env create -n [ENVIRONMENT_NAME] -f environment.yml

> conda activate [ENVIRONMENT_NAME]

> pip install ctlearn==$CTLEARN_VER

> ctlearn -h
```

# Short demo

- Usage, command line

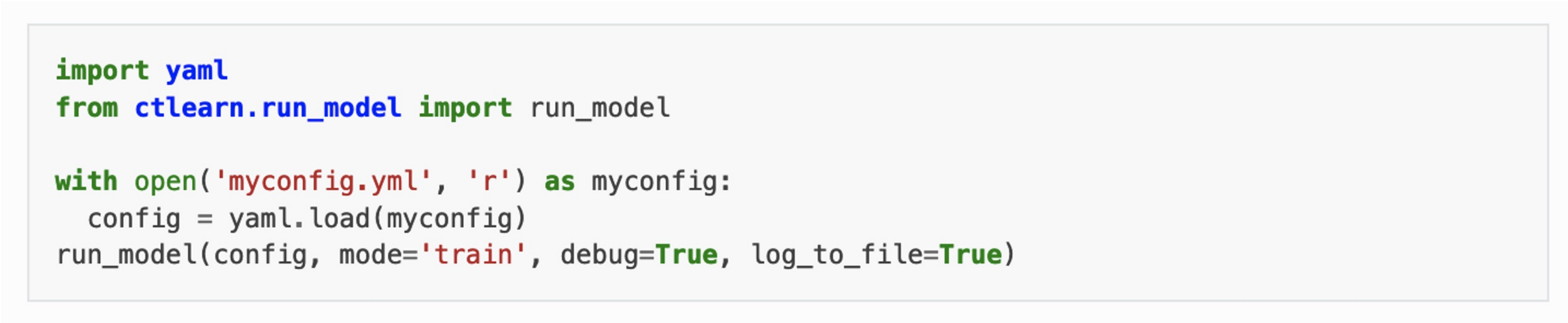


```
(ctlearn) ➜  ~ ctlearn -h
usage: ctlearn [-h] [--config_file CONFIG_FILE] [--input INPUT [INPUT ...]] [--pattern PATTERN [PATTERN ...]] [--mode MODE]
               [--output OUTPUT] [--reco RECO [RECO ...]] [--default_model DEFAULT_MODEL]
               [--pretrained_weights PRETRAINED_WEIGHTS] [--tel_types TEL_TYPES [TEL_TYPES ...]]
               [--allowed_tels ALLOWED_TELS [ALLOWED_TELS ...]] [--size_cut SIZE_CUT] [--leakage_cut LEAKAGE_CUT]
               [--multiplicity_cut MULTIPLICITY_CUT] [--num_epochs NUM_EPOCHS] [--batch_size BATCH_SIZE]
               [--random_seed RANDOM_SEED] [--log_to_file] [--debug]

Train/Predict with a CTLearn model.

optional arguments:
  -h, --help            show this help message and exit
  --config_file CONFIG_FILE, -c CONFIG_FILE
                        Path to YAML configuration file with training options
  --input INPUT [INPUT ...], -i INPUT [INPUT ...]
                        Input directories (not required when file_list is set in the config file)
  --pattern PATTERN [PATTERN ...], -p PATTERN [PATTERN ...]
                        Pattern to mask unwanted files from the data input directory
  --mode MODE, -m MODE  Mode to run CTLearn; valid options: train, predict, or train_and_predict
  --output OUTPUT, -o OUTPUT
                        Output directory, where the logging, model weights and processed output files are stored
  --reco RECO [RECO ...], -r RECO [RECO ...]
                        Reconstruction task to perform; valid options: particletype, energy, and/or direction
  --default_model DEFAULT_MODEL, -d DEFAULT_MODEL
                        Default CTLearn Model; valid options: TRN, TRN_cleaned, mergedTRN, mergedTRN_cleaned, CNNRNN, and
                        CNNRNN_cleaned
  --pretrained_weights PRETRAINED_WEIGHTS, -w PRETRAINED_WEIGHTS
                        Path to the pretrained weights
  --tel_types TEL_TYPES [TEL_TYPES ...], -t TEL_TYPES [TEL_TYPES ...]
                        Selection of telescope types; valid option: LST_LST_LSTCam, LST_MAGIC_MAGICCam, MST_MST_FlashCam,
                        MST_MST_NectarCam, SST_SCT_SCTCam, and/or SST_ASTRI_ASTRICam
  --allowed_tels ALLOWED_TELS [ALLOWED_TELS ...], -a ALLOWED_TELS [ALLOWED_TELS ...]
                        List of allowed tel_ids, others will be ignored. Selected tel_ids will be ignored, when their telescope
                        type is not selected
  --size_cut SIZE_CUT, -z SIZE_CUT
                        Hillas intensity cut to perform
  --leakage_cut LEAKAGE_CUT, -l LEAKAGE_CUT
                        Leakage intensity cut to perform
  --multiplicity_cut MULTIPLICITY_CUT, -u MULTIPLICITY_CUT
                        Multiplicity cut to perform
  --num_epochs NUM_EPOCHS, -e NUM_EPOCHS
                        Number of epochs to train
  --batch_size BATCH_SIZE, -b BATCH_SIZE
                        Batch size per worker
  --random_seed RANDOM_SEED, -s RANDOM_SEED
                        Selection of random seed (4 digits)
  --log_to_file         Log to a file in model directory instead of terminal
  --debug               Print debug/logger messages
(ctlearn) ➜  ~ 
```

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Usage, python module



```
import yaml
from ctlearn.run_model import run_model

with open('myconfig.yml', 'r') as myconfig:
  config = yaml.load(myconfig)
run_model(config, mode='train', debug=True, log_to_file=True)
```

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

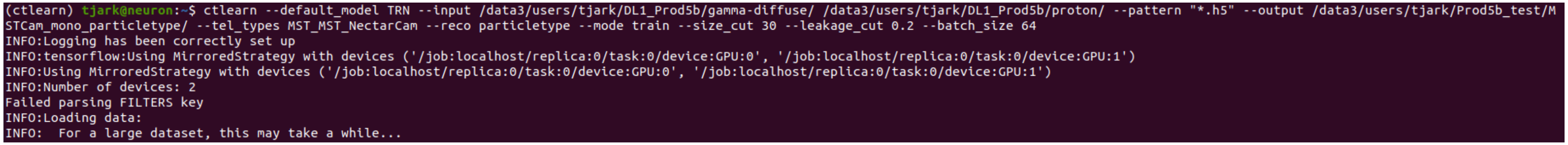
- Configuration file

https://github.com/ctlearn-project/ctlearn/blob/master/config/example_config.yml

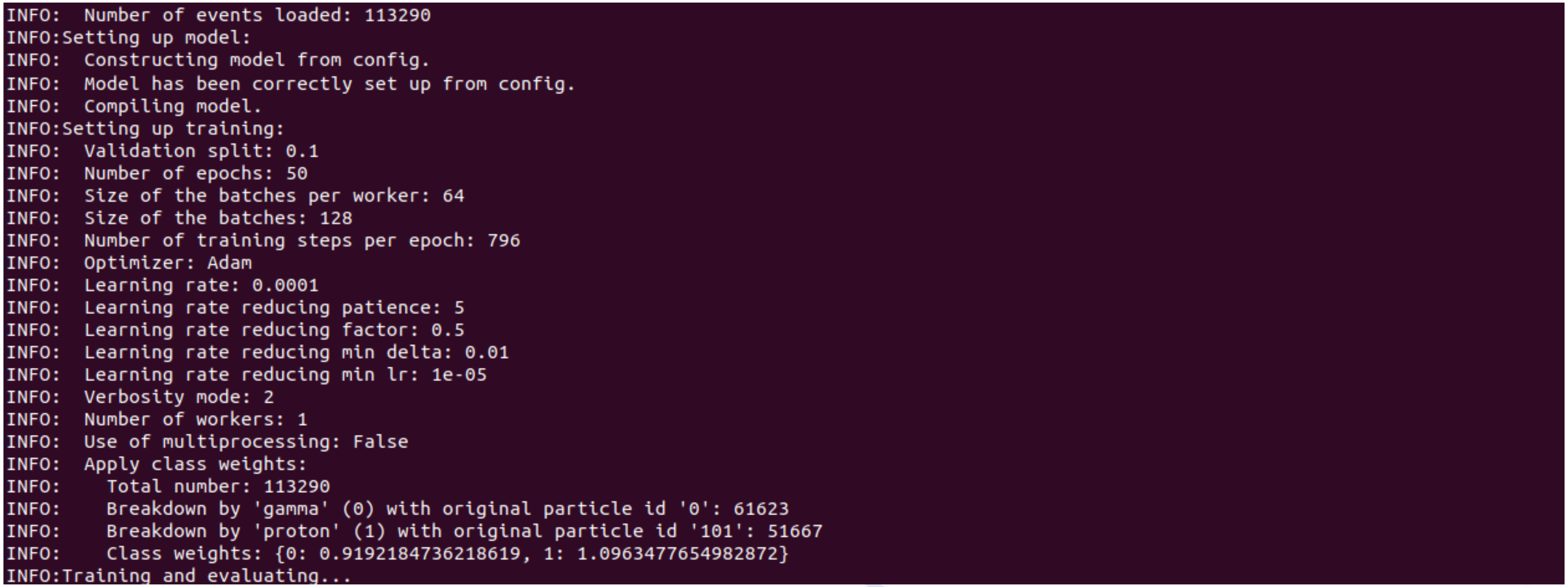
Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Training run



```
(ctlearn) tjark@neuron:~$ ctlearn --default_model TRN --input /data3/users/tjark/DL1_Prod5b/gamma-diffuse/ /data3/users/tjark/DL1_Prod5b/proton/ --pattern "*.h5" --output /data3/users/tjark/Prod5b_test/M
STCam_mono_particletype/ --tel_types MST_MST_NectarCam --reco particletype --mode train --size_cut 30 --leakage_cut 0.2 --batch_size 64
INFO:Logging has been correctly set up
INFO:tensorflow:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0', '/job:localhost/replica:0/task:0/device:GPU:1')
INFO:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0', '/job:localhost/replica:0/task:0/device:GPU:1')
INFO:Number of devices: 2
Failed parsing FILTERS key
INFO:Loading data:
INFO:  For a large dataset, this may take a while...
```

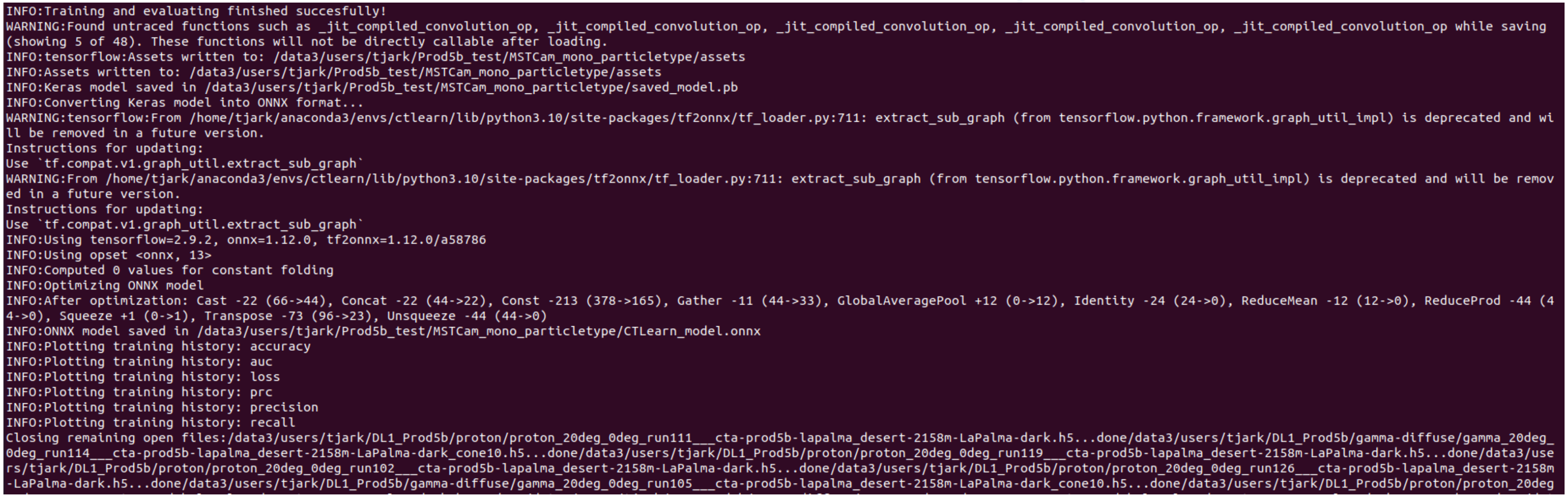


```
INFO:  Number of events loaded: 113290
INFO:Setting up model:
INFO:  Constructing model from config.
INFO:  Model has been correctly set up from config.
INFO:  Compiling model.
INFO:Setting up training:
INFO:  Validation split: 0.1
INFO:  Number of epochs: 50
INFO:  Size of the batches per worker: 64
INFO:  Size of the batches: 128
INFO:  Number of training steps per epoch: 796
INFO:  Optimizer: Adam
INFO:  Learning rate: 0.0001
INFO:  Learning rate reducing patience: 5
INFO:  Learning rate reducing factor: 0.5
INFO:  Learning rate reducing min delta: 0.01
INFO:  Learning rate reducing min lr: 1e-05
INFO:  Verbosity mode: 2
INFO:  Number of workers: 1
INFO:  Use of multiprocessing: False
INFO:  Apply class weights:
INFO:    Total number: 113290
INFO:    Breakdown by 'gamma' (0) with original particle id '0': 61623
INFO:    Breakdown by 'proton' (1) with original particle id '101': 51667
INFO:    Class weights: {0: 0.9192184736218619, 1: 1.0963477654982872}
INFO:Training and evaluating...
```

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Training run

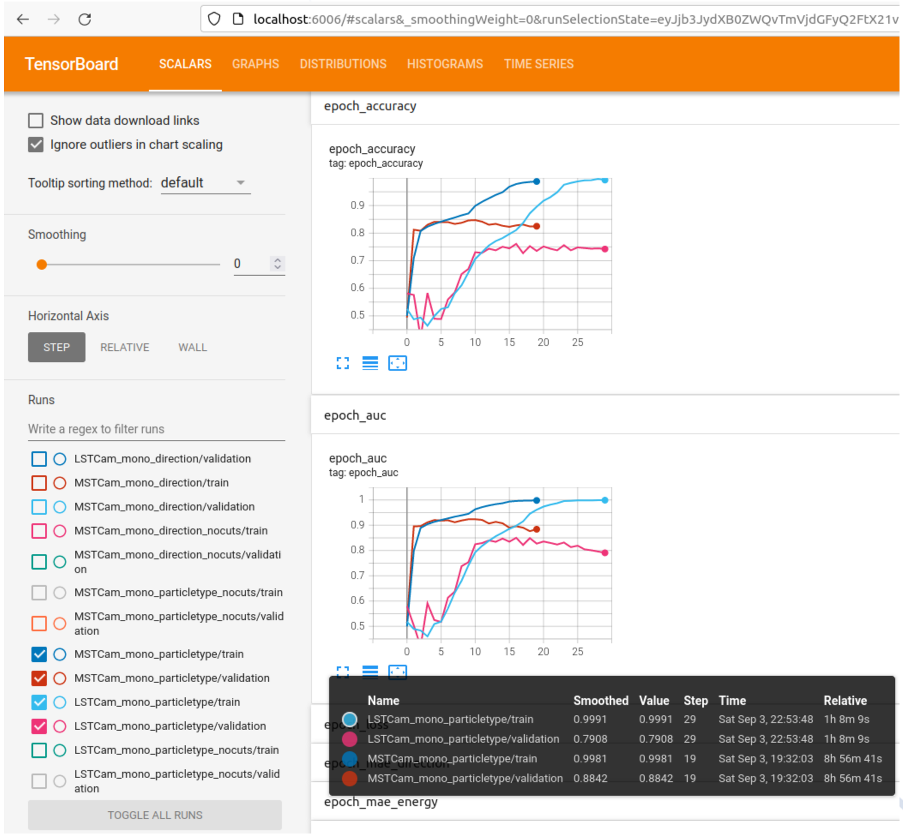


```
INFO:Training and evaluating finished succesfully!
WARNING:Found untraced functions such as _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op while saving
(showing 5 of 48). These functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: /data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/assets
INFO:Assets written to: /data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/assets
INFO:Keras model saved in /data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/saved_model.pb
INFO:Converting Keras model into ONNX format...
WARNING:tensorflow:From /home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/tf2onnx/tf_loader.py:711: extract_sub_graph (from tensorflow.python.framework.graph_util_impl) is deprecated and wi
ll be removed in a future version.
Instructions for updating:
Use `tf.compat.v1.graph_util.extract_sub_graph`
WARNING:From /home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/tf2onnx/tf_loader.py:711: extract_sub_graph (from tensorflow.python.framework.graph_util_impl) is deprecated and will be remov
ed in a future version.
Instructions for updating:
Use `tf.compat.v1.graph_util.extract_sub_graph`
INFO:Using tensorflow=2.9.2, onnx=1.12.0, tf2onnx=1.12.0/a58786
INFO:Using opset <onnx, 13>
INFO:Computed 0 values for constant folding
INFO:Optimizing ONNX model
INFO:After optimization: Cast -22 (66->44), Concat -22 (44->22), Const -213 (378->165), Gather -11 (44->33), GlobalAveragePool +12 (0->12), Identity -24 (24->0), ReduceMean -12 (12->0), ReduceProd -44 (4
4->0), Squeeze +1 (0->1), Transpose -73 (96->23), Unsqueeze -44 (44->0)
INFO:ONNX model saved in /data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/CTLearn_model.onnx
INFO:Plotting training history: accuracy
INFO:Plotting training history: auc
INFO:Plotting training history: loss
INFO:Plotting training history: prc
INFO:Plotting training history: precision
INFO:Plotting training history: recall
Closing remaining open files:/data3/users/tjark/DL1_Prod5b/proton/proton_20deg_0deg_run111___cta-prod5b-lapalma_desert-2158m-LaPalma-dark.h5...done/data3/users/tjark/DL1_Prod5b/gamma-diffuse/gamma_20deg_
0deg_run114___cta-prod5b-lapalma_desert-2158m-LaPalma-dark_cone10.h5...done/data3/users/tjark/DL1_Prod5b/proton/proton_20deg_0deg_run119___cta-prod5b-lapalma_desert-2158m-LaPalma-dark.h5...done/data3/use
rs/tjark/DL1_Prod5b/proton/proton_20deg_0deg_run102___cta-prod5b-lapalma_desert-2158m-LaPalma-dark.h5...done/data3/users/tjark/DL1_Prod5b/proton/proton_20deg_0deg_run126___cta-prod5b-lapalma_desert-2158m
-LaPalma-dark.h5...done/data3/users/tjark/DL1_Prod5b/gamma-diffuse/gamma_20deg_0deg_run105___cta-prod5b-lapalma_desert-2158m-LaPalma-dark_cone10.h5...done/data3/users/tjark/DL1_Prod5b/proton/proton_20deg
```

Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Training run



☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0

Horizontal Axis: STEP RELATIVE WALL

Runs

Write a regex to filter runs

- ☐ ○ LSTCam_mono_direction/validation
- ☐ ○ MSTCam_mono_direction/train
- ☐ ○ MSTCam_mono_direction/validation
- ☐ ○ MSTCam_mono_direction_nocuts/train
- ☐ ○ MSTCam_mono_direction_nocuts/validation
- ☐ ○ MSTCam_mono_particletype_nocuts/train
- ☐ ○ MSTCam_mono_particletype_nocuts/validation
- ☑ ○ MSTCam_mono_particletype/train
- ☑ ○ MSTCam_mono_particletype/validation
- ☑ ○ LSTCam_mono_particletype/train
- ☑ ○ LSTCam_mono_particletype/validation
- ☐ ○ LSTCam_mono_particletype_nocuts/train
- ☐ ○ LSTCam_mono_particletype_nocuts/validation

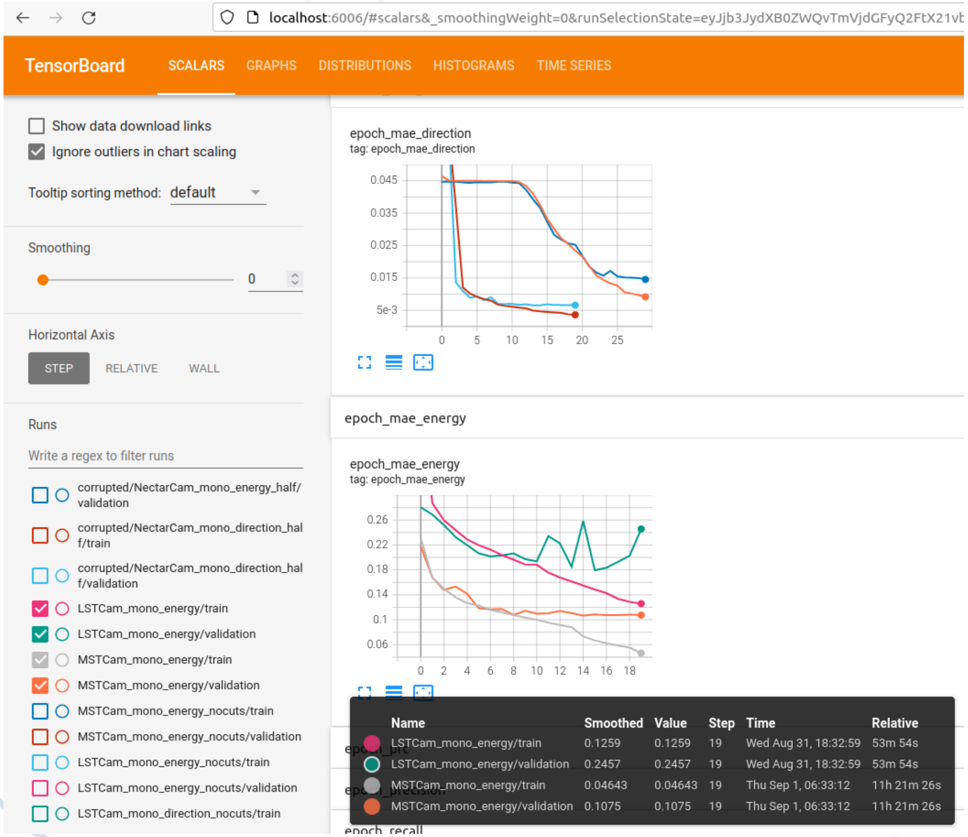
TOGGLE ALL RUNS

epoch_loss

epoch_mae_direction

epoch_mae_energy

| | Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|---|
| ● | LSTCam_mono_particletype/train | 0.9991 | 0.9991 | 29 | Sat Sep 3, 22:53:48 | 1h 8m 9s |
| ● | LSTCam_mono_particletype/validation | 0.7908 | 0.7908 | 29 | Sat Sep 3, 22:53:48 | 1h 8m 9s |
| ● | MSTCam_mono_particletype/train | 0.9981 | 0.9981 | 19 | Sat Sep 3, 19:32:03 | 8h 56m 41s |
| ● | MSTCam_mono_particletype/validation | 0.8842 | 0.8842 | 19 | Sat Sep 3, 19:32:03 | 8h 56m 41s |



☐ Show data download links

☑ Ignore outliers in chart scaling

Tooltip sorting method: default

Smoothing 0

Horizontal Axis: STEP RELATIVE WALL

Runs

Write a regex to filter runs

- ☐ ○ corrupted/NectarCam_mono_energy_half/validation
- ☐ ○ corrupted/NectarCam_mono_direction_half/train
- ☐ ○ corrupted/NectarCam_mono_direction_half/validation
- ☑ ○ LSTCam_mono_energy/train
- ☑ ○ LSTCam_mono_energy/validation
- ☑ ○ MSTCam_mono_energy/train
- ☑ ○ MSTCam_mono_energy/validation
- ☐ ○ MSTCam_mono_energy_nocuts/train
- ☐ ○ MSTCam_mono_energy_nocuts/validation
- ☐ ○ LSTCam_mono_energy_nocuts/train
- ☐ ○ LSTCam_mono_energy_nocuts/validation
- ☐ ○ LSTCam_mono_direction_nocuts/train

epoch_prc

epoch_precision

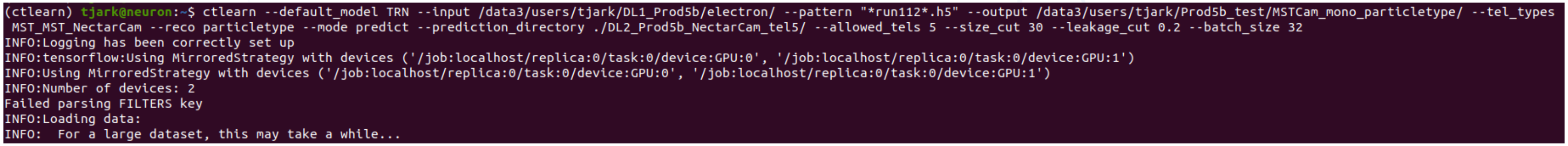
epoch_recall

| | Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|---|
| ● | LSTCam_mono_energy/train | 0.1259 | 0.1259 | 19 | Wed Aug 31, 18:32:59 | 53m 54s |
| ● | LSTCam_mono_energy/validation | 0.2457 | 0.2457 | 19 | Wed Aug 31, 18:32:59 | 53m 54s |
| ● | MSTCam_mono_energy/train | 0.04643 | 0.04643 | 19 | Thu Sep 1, 06:33:12 | 11h 21m 26s |
| ● | MSTCam_mono_energy/validation | 0.1075 | 0.1075 | 19 | Thu Sep 1, 06:33:12 | 11h 21m 26s |

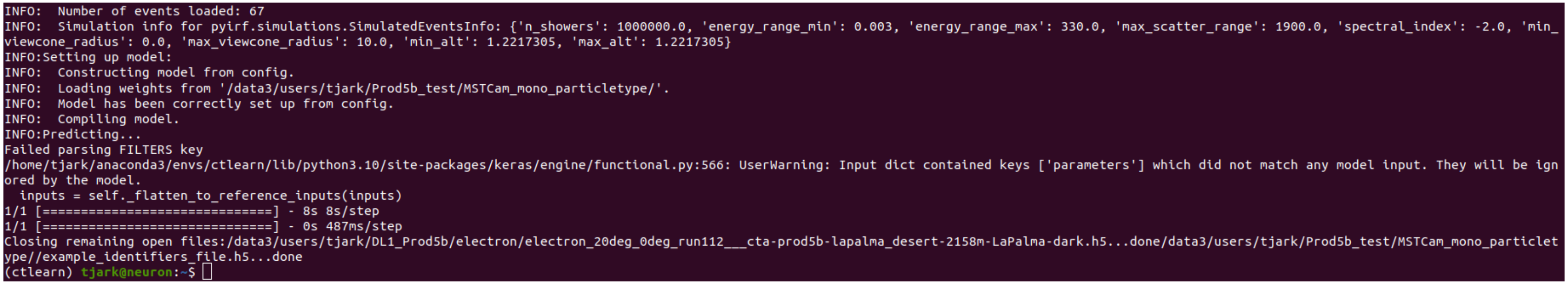
Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Prediction run



```
(ctlearn) tjark@neuron:~$ ctlearn --default_model TRN --input /data3/users/tjark/DL1_Prod5b/electron/ --pattern "*run112*.h5" --output /data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/ --tel_types
 MST_MST_NectarCam --reco particletype --mode predict --prediction_directory ./DL2_Prod5b_NectarCam_tel5/ --allowed_tels 5 --size_cut 30 --leakage_cut 0.2 --batch_size 32
INFO:Logging has been correctly set up
INFO:tensorflow:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0', '/job:localhost/replica:0/task:0/device:GPU:1')
INFO:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0', '/job:localhost/replica:0/task:0/device:GPU:1')
INFO:Number of devices: 2
Failed parsing FILTERS key
INFO:Loading data:
INFO:  For a large dataset, this may take a while...
```

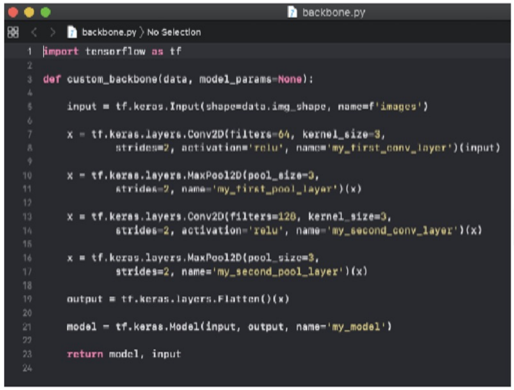


```
INFO:  Number of events loaded: 67
INFO:  Simulation info for pyirf.simulations.SimulatedEventsInfo: {'n_showers': 1000000.0, 'energy_range_min': 0.003, 'energy_range_max': 330.0, 'max_scatter_range': 1900.0, 'spectral_index': -2.0, 'min_
viewcone_radius': 0.0, 'max_viewcone_radius': 10.0, 'min_alt': 1.2217305, 'max_alt': 1.2217305}
INFO:Setting up model:
INFO:  Constructing model from config.
INFO:  Loading weights from '/data3/users/tjark/Prod5b_test/MSTCam_mono_particletype/'.
INFO:  Model has been correctly set up from config.
INFO:  Compiling model.
INFO:Predicting...
Failed parsing FILTERS key
/home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/keras/engine/functional.py:566: UserWarning: Input dict contained keys ['parameters'] which did not match any model input. They will be ign
ored by the model.
  inputs = self._flatten_to_reference_inputs(inputs)
1/1 [==============================] - 8s 8s/step
1/1 [==============================] - 0s 487ms/step
Closing remaining open files:/data3/users/tjark/DL1_Prod5b/electron/electron_20deg_0deg_run112___cta-prod5b-lapalma_desert-2158m-LaPalma-dark.h5...done/data3/users/tjark/Prod5b_test/MSTCam_mono_particlet
ype//example_identifiers_file.h5...done
(ctlearn) tjark@neuron:~$ 
```

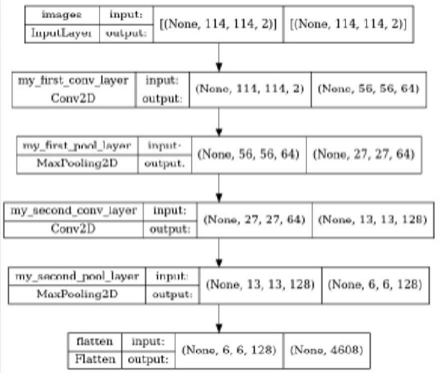
Funded by the European Union's
Horizon 2020 - Grant N° 824064

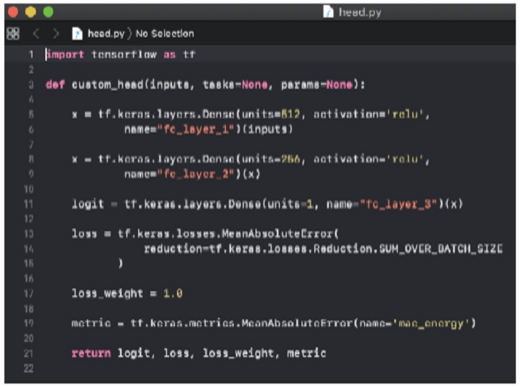
# Short demo

- Customizing models

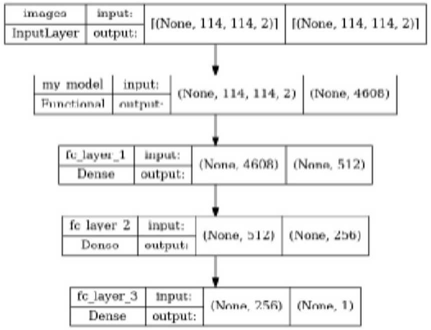


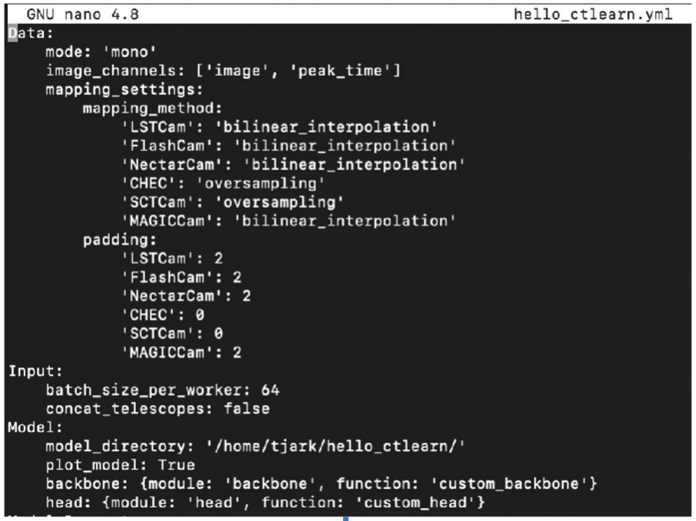
```python
import tensorflow as tf

def custom_backbone(data, model_params=None):

    input = tf.keras.Input(shape=data.img_shape, name=f'images')

    x = tf.keras.layers.Conv2D(filters=64, kernel_size=3,
            strides=2, activation='relu', name='my_first_conv_layer')(input)

    x = tf.keras.layers.MaxPool2D(pool_size=3,
            strides=2, name='my_first_pool_layer')(x)

    x = tf.keras.layers.Conv2D(filters=128, kernel_size=3,
            strides=2, activation='relu', name='my_second_conv_layer')(x)

    x = tf.keras.layers.MaxPool2D(pool_size=3,
            strides=2, name='my_second_pool_layer')(x)

    output = tf.keras.layers.Flatten()(x)

    model = tf.keras.Model(input, output, name='my_model')

    return model, input
```


images | input: | [(None, 114, 114, 2)] | [(None, 114, 114, 2)]
InputLayer | output:
my_first_conv_layer | input: | (None, 114, 114, 2) | (None, 56, 56, 64)
Conv2D | output:
my_first_pool_layer | input: | (None, 56, 56, 64) | (None, 27, 27, 64)
MaxPooling2D | output:
my_second_conv_layer | input: | (None, 27, 27, 64) | (None, 13, 13, 128)
Conv2D | output:
my_second_pool_layer | input: | (None, 13, 13, 128) | (None, 6, 6, 128)
MaxPooling2D | output:
flatten | input: | (None, 6, 6, 128) | (None, 4608)
Flatten | output:

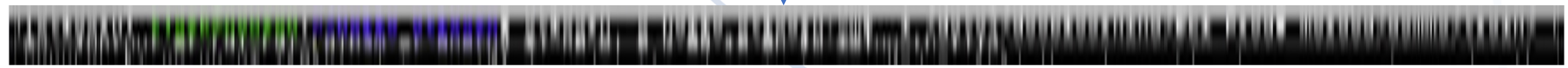


```python
import tensorflow as tf

def custom_head(inputs, tasks=None, params=None):

    x = tf.keras.layers.Dense(units=512, activation='relu',
            name="fc_layer_1")(inputs)

    x = tf.keras.layers.Dense(units=256, activation='relu',
            name="fc_layer_2")(x)

    logit = tf.keras.layers.Dense(units=1, name="fc_layer_3")(x)

    loss = tf.keras.losses.MeanAbsoluteError(
                reduction=tf.keras.losses.Reduction.SUM_OVER_BATCH_SIZE
            )

    loss_weight = 1.0

    metric = tf.keras.metrics.MeanAbsoluteError(name='mae_energy')

    return logit, loss, loss_weight, metric
```


images | input: | [(None, 114, 114, 2)] | [(None, 114, 114, 2)]
InputLayer | output:
my_model | input: | (None, 114, 114, 2) | (None, 4608)
Functional | output:
fc_layer_1 | input: | (None, 4608) | (None, 512)
Dense | output:
fc_layer_2 | input: | (None, 512) | (None, 256)
Dense | output:
fc_layer_3 | input: | (None, 256) | (None, 1)
Dense | output:



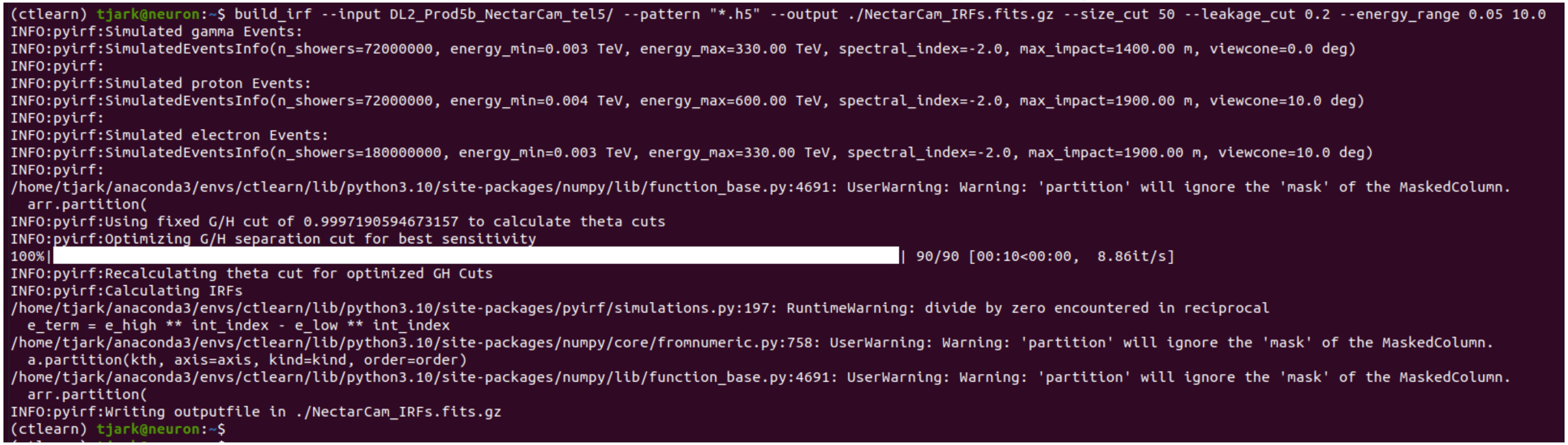
```yaml
Data:
    mode: 'mono'
    image_channels: ['image', 'peak_time']
    mapping_settings:
        mapping_method:
            'LSTCam': 'bilinear_interpolation'
            'FlashCam': 'bilinear_interpolation'
            'NectarCam': 'bilinear_interpolation'
            'CHEC': 'oversampling'
            'SCTCam': 'oversampling'
            'MAGICCam': 'bilinear_interpolation'
        padding:
            'LSTCam': 2
            'FlashCam': 2
            'NectarCam': 2
            'CHEC': 0
            'SCTCam': 0
            'MAGICCam': 2
Input:
    batch_size_per_worker: 64
    concat_telescopes: false
Model:
    model_directory: '/home/tjark/hello_ctlearn/'
    plot_model: True
    backbone: {module: 'backbone', function: 'custom_backbone'}
    head: {module: 'head', function: 'custom_head'}
```



Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Build IRFs

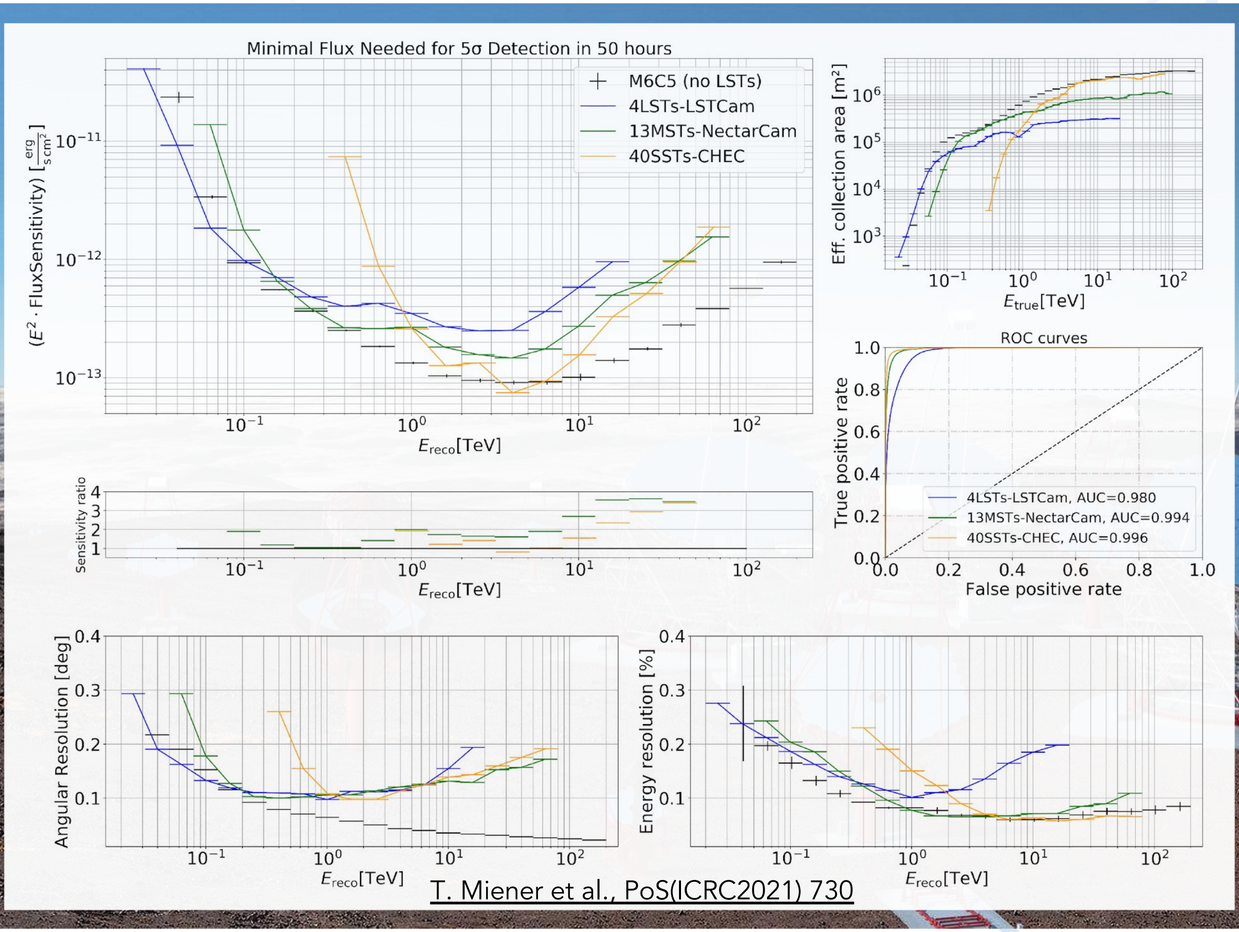


```
(ctlearn) tjark@neuron:~$ build_irf --input DL2_Prod5b_NectarCam_tel5/ --pattern "*.h5" --output ./NectarCam_IRFs.fits.gz --size_cut 50 --leakage_cut 0.2 --energy_range 0.05 10.0
INFO:pyirf:Simulated gamma Events:
INFO:pyirf:SimulatedEventsInfo(n_showers=72000000, energy_min=0.003 TeV, energy_max=330.00 TeV, spectral_index=-2.0, max_impact=1400.00 m, viewcone=0.0 deg)
INFO:pyirf:
INFO:pyirf:Simulated proton Events:
INFO:pyirf:SimulatedEventsInfo(n_showers=72000000, energy_min=0.004 TeV, energy_max=600.00 TeV, spectral_index=-2.0, max_impact=1900.00 m, viewcone=10.0 deg)
INFO:pyirf:
INFO:pyirf:Simulated electron Events:
INFO:pyirf:SimulatedEventsInfo(n_showers=180000000, energy_min=0.003 TeV, energy_max=330.00 TeV, spectral_index=-2.0, max_impact=1900.00 m, viewcone=10.0 deg)
INFO:pyirf:
/home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/numpy/lib/function_base.py:4691: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedColumn.
  arr.partition(
INFO:pyirf:Using fixed G/H cut of 0.9997190594673157 to calculate theta cuts
INFO:pyirf:Optimizing G/H separation cut for best sensitivity
100%|██████████████████████████████████████████████████████████████████████████████████████████████████| 90/90 [00:10<00:00,  8.86it/s]
INFO:pyirf:Recalculating theta cut for optimized GH Cuts
INFO:pyirf:Calculating IRFs
/home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/pyirf/simulations.py:197: RuntimeWarning: divide by zero encountered in reciprocal
  e_term = e_high ** int_index - e_low ** int_index
/home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/numpy/core/fromnumeric.py:758: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedColumn.
  a.partition(kth, axis=axis, kind=kind, order=order)
/home/tjark/anaconda3/envs/ctlearn/lib/python3.10/site-packages/numpy/lib/function_base.py:4691: UserWarning: Warning: 'partition' will ignore the 'mask' of the MaskedColumn.
  arr.partition(
INFO:pyirf:Writing outputfile in ./NectarCam_IRFs.fits.gz
(ctlearn) tjark@neuron:~$
```

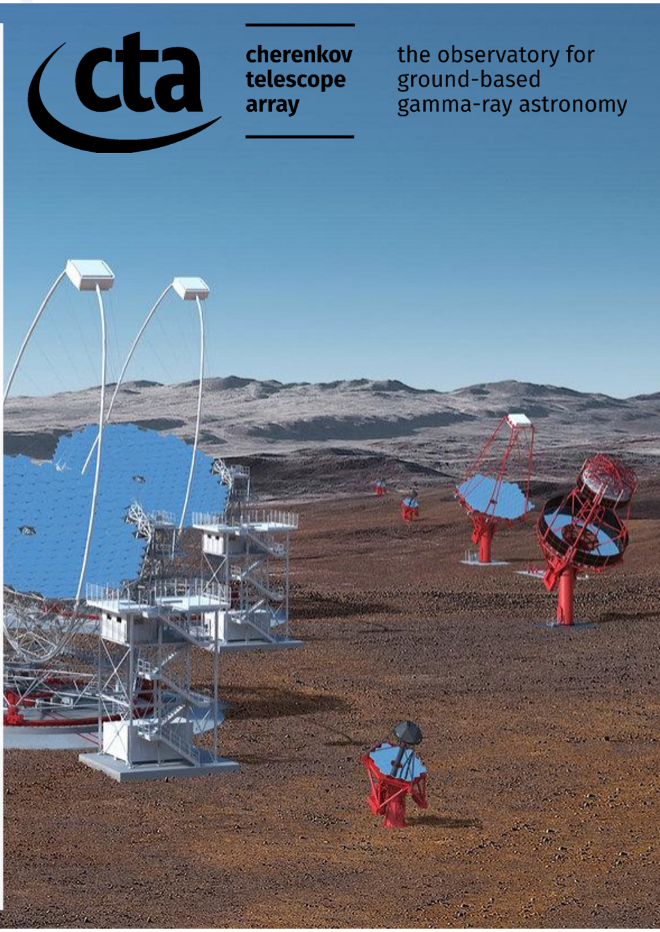
Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Outcome



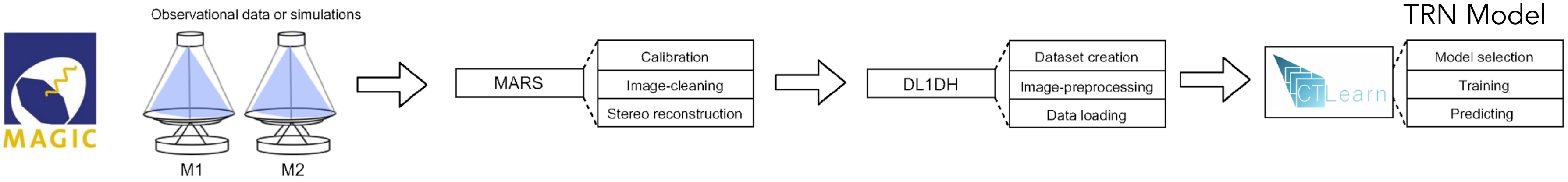
T. Miener et al., PoS(ICRC2021) 730



Funded by the European Union's
Horizon 2020 - Grant N° 824064

# Short demo

- Outcome



Observational data or simulations

M1 M2

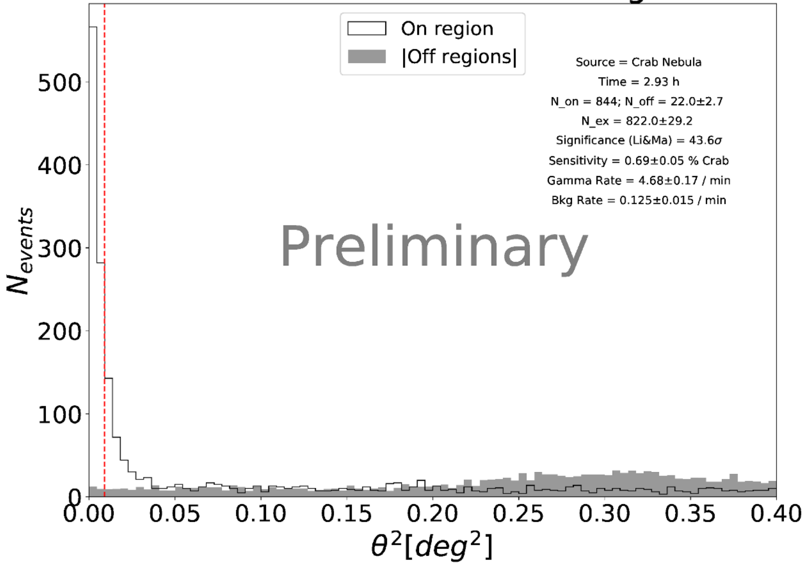
MARS: Calibration, Image-cleaning, Stereo reconstruction

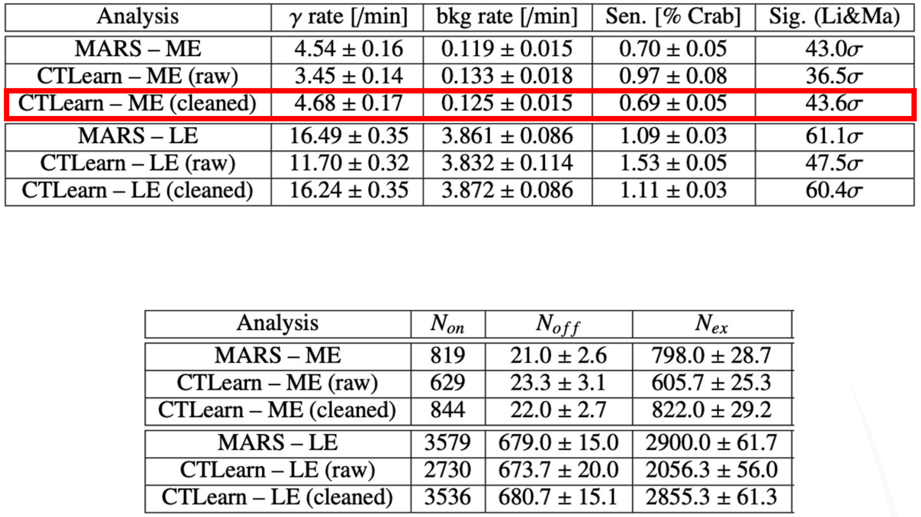
DL1DH: Dataset creation, Image-preprocessing, Data loading

TRN Model

CTLearn: Model selection, Training, Predicting

CTLearn ME - cleaned images





| Analysis | $\gamma$ rate [/min] | bkg rate [/min] | Sen. [% Crab] | Sig. (Li&Ma) |
|---|---|---|---|---|
| MARS – ME | 4.54 ± 0.16 | 0.119 ± 0.015 | 0.70 ± 0.05 | $43.0\sigma$ |
| CTLearn – ME (raw) | 3.45 ± 0.14 | 0.133 ± 0.018 | 0.97 ± 0.08 | $36.5\sigma$ |
| CTLearn – ME (cleaned) | 4.68 ± 0.17 | 0.125 ± 0.015 | 0.69 ± 0.05 | $43.6\sigma$ |
| MARS – LE | 16.49 ± 0.35 | 3.861 ± 0.086 | 1.09 ± 0.03 | $61.1\sigma$ |
| CTLearn – LE (raw) | 11.70 ± 0.32 | 3.832 ± 0.114 | 1.53 ± 0.05 | $47.5\sigma$ |
| CTLearn – LE (cleaned) | 16.24 ± 0.35 | 3.872 ± 0.086 | 1.11 ± 0.03 | $60.4\sigma$ |

| Analysis | $N_{on}$ | $N_{off}$ | $N_{ex}$ |
|---|---|---|---|
| MARS – ME | 819 | 21.0 ± 2.6 | 798.0 ± 28.7 |
| CTLearn – ME (raw) | 629 | 23.3 ± 3.1 | 605.7 ± 25.3 |
| CTLearn – ME (cleaned) | 844 | 22.0 ± 2.7 | 822.0 ± 29.2 |
| MARS – LE | 3579 | 679.0 ± 15.0 | 2900.0 ± 61.7 |
| CTLearn – LE (raw) | 2730 | 673.7 ± 20.0 | 2056.3 ± 56.0 |
| CTLearn – LE (cleaned) | 3536 | 680.7 ± 15.1 | 2855.3 ± 61.3 |

Summary of all performed analyses of the same Crab Nebula sample

T. Miener et al. 2021 (ADASS XXXI)

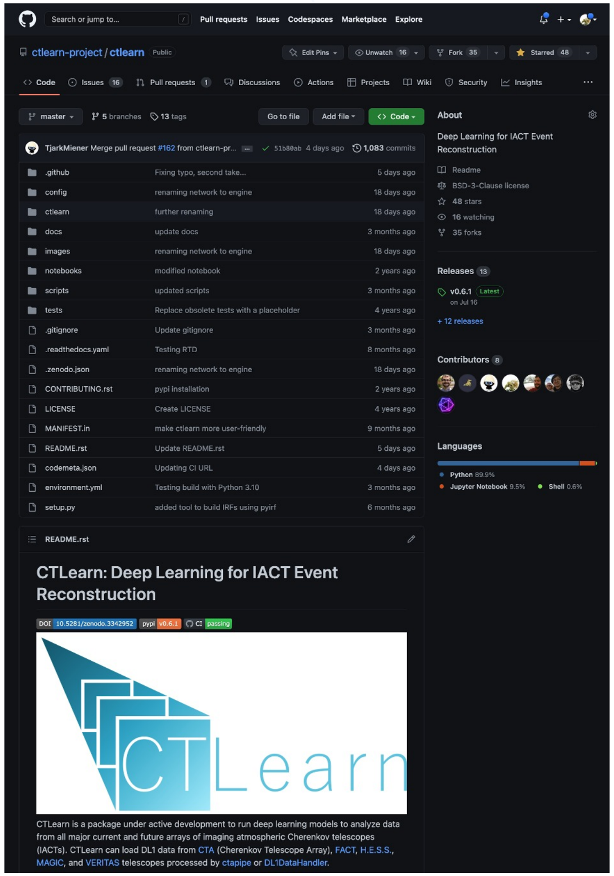
Funded by the European Union’s
Horizon 2020 - Grant N° 824064

# Open Points and Discussion Time

https://github.com/ctlearn-project/ctlearn

https://ctlearn.readthedocs.io

https://zenodo.org/record/6842323



Search or jump to... | Pull requests | Issues | Codespaces | Marketplace | Explore

ctlearn-project / ctlearn Public — Edit Pins · Unwatch 16 · Fork 35 · Starred 48

Code · Issues 16 · Pull requests 1 · Discussions · Actions · Projects · Wiki · Security · Insights

master · 5 branches · 13 tags · Go to file · Add file · Code

TjarkMiener Merge pull request #162 from ctlearn-pr... ✓ 51b80ab 4 days ago · 1,083 commits

| | | |
|---|---|---|
| .github | Fixing typo, second take... | 5 days ago |
| config | renaming network to engine | 18 days ago |
| ctlearn | further renaming | 18 days ago |
| docs | update docs | 3 months ago |
| images | renaming network to engine | 18 days ago |
| notebooks | modified notebook | 2 years ago |
| scripts | updated scripts | 3 months ago |
| tests | Replace obsolete tests with a placeholder | 4 years ago |
| .gitignore | Update gitignore | 3 months ago |
| .readthedocs.yaml | Testing RTD | 8 months ago |
| .zenodo.json | renaming network to engine | 18 days ago |
| CONTRIBUTING.rst | pypi installation | 2 years ago |
| LICENSE | Create LICENSE | 4 years ago |
| MANIFEST.in | make ctlearn more user-friendly | 9 months ago |
| README.rst | Update README.rst | 5 days ago |
| codemeta.json | Updating CI URL | 4 days ago |
| environment.yml | Testing build with Python 3.10 | 3 months ago |
| setup.py | added tool to build IRFs using pyirf | 6 months ago |

About

Deep Learning for IACT Event Reconstruction

Readme
BSD-3-Clause license
48 stars
16 watching
35 forks

Releases 13

v0.6.1 Latest
on Jul 16

+ 12 releases

Contributors 8

Languages

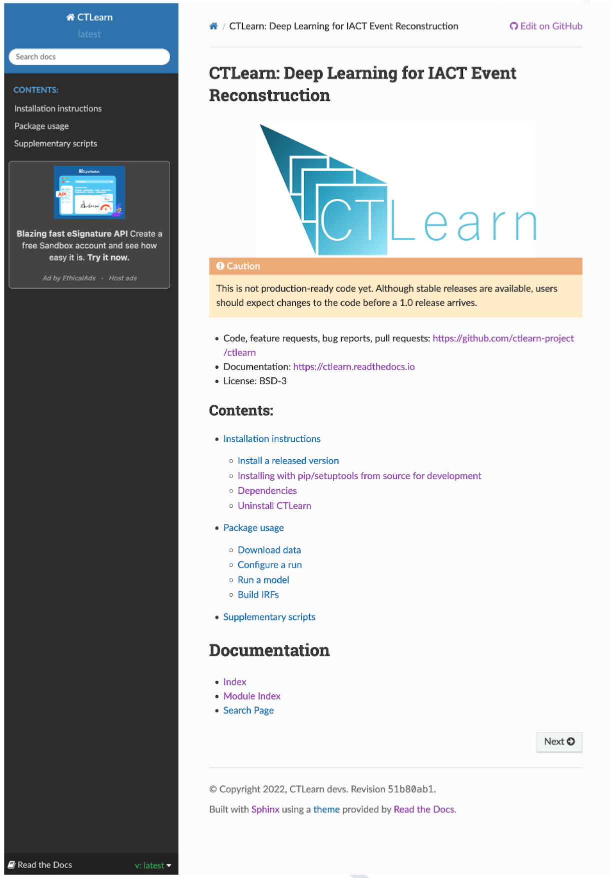
- Python 89.9%
- Jupyter Notebook 9.5%
- Shell 0.6%

README.rst

CTLearn: Deep Learning for IACT Event Reconstruction

DOI 10.5281/zenodo.3342952 · pypi v0.6.1 · CI passing

CTLearn is a package under active development to run deep learning models to analyze data from all major current and future arrays of imaging atmospheric Cherenkov telescopes (IACTs). CTLearn can load DL1 data from CTA (Cherenkov Telescope Array), FACT, H.E.S.S., MAGIC, and VERITAS telescopes processed by ctapipe or DL1DataHandler.



CTLearn
latest

Search docs

CONTENTS:
- Installation instructions
- Package usage
- Supplementary scripts

Blazing fast eSignature API Create a free Sandbox account and see how easy it is. Try it now.
Ad by EthicalAds · Host ads

Read the Docs · v: latest

/ CTLearn: Deep Learning for IACT Event Reconstruction · Edit on GitHub

CTLearn: Deep Learning for IACT Event Reconstruction

Caution

This is not production-ready code yet. Although stable releases are available, users should expect changes to the code before a 1.0 release arrives.

- Code, feature requests, bug reports, pull requests: https://github.com/ctlearn-project/ctlearn
- Documentation: https://ctlearn.readthedocs.io
- License: BSD-3

Contents:

- Installation instructions
  - Install a released version
  - Installing with pip/setuptools from source for development
  - Dependencies
  - Uninstall CTLearn
- Package usage
  - Download data
  - Configure a run
  - Run a model
  - Build IRFs
- Supplementary scripts

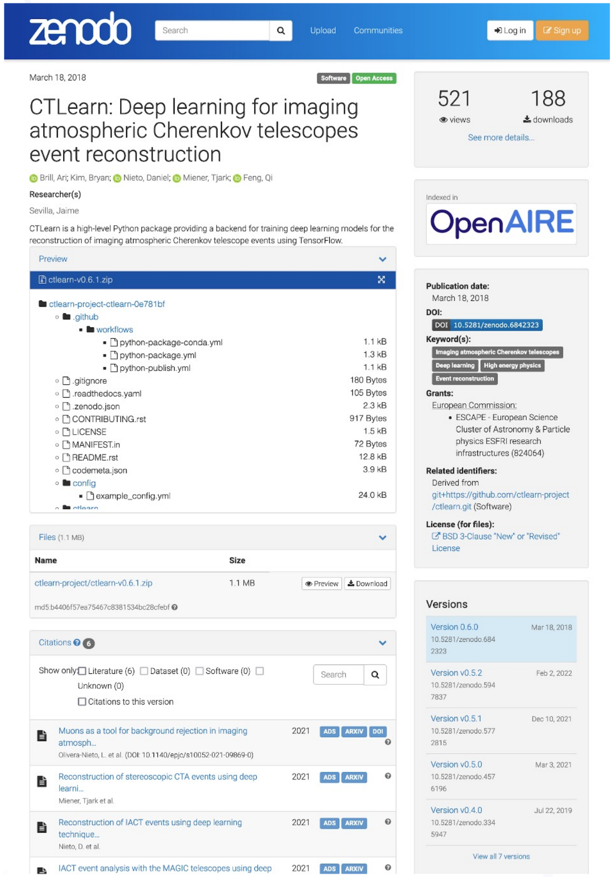
Documentation

- Index
- Module Index
- Search Page

Next

© Copyright 2022, CTLearn devs. Revision 51b80ab1.

Built with Sphinx using a theme provided by Read the Docs.



zenodo · Search · Upload · Communities · Log in · Sign up

March 18, 2018 · Software · Open Access

CTLearn: Deep learning for imaging atmospheric Cherenkov telescopes event reconstruction

Brill, Ari; Kim, Bryan; Nieto, Daniel; Miener, Tjark; Feng, Qi

Researcher(s)

Sevilla, Jaime

CTLearn is a high-level Python package providing a backend for training deep learning models for the reconstruction of imaging atmospheric Cherenkov telescope events using TensorFlow.

Preview

ctlearn-v0.6.1.zip

| | |
|---|---|
| ctlearn-project-ctlearn-0e781bf | |
| .github | |
| workflows | |
| python-package-conda.yml | 1.1 kB |
| python-package.yml | 1.3 kB |
| python-publish.yml | 1.1 kB |
| .gitignore | 180 Bytes |
| .readthedocs.yaml | 105 Bytes |
| .zenodo.json | 2.3 kB |
| CONTRIBUTING.rst | 917 Bytes |
| LICENSE | 1.5 kB |
| MANIFEST.in | 72 Bytes |
| README.rst | 12.8 kB |
| codemeta.json | 3.9 kB |
| config | |
| example_config.yml | 24.0 kB |
| ctlearn | |

Files (1.1 MB)

| Name | Size | |
|---|---|---|
| ctlearn-project/ctlearn-v0.6.1.zip | 1.1 MB | Preview · Download |

md5:b4406f57ea75467c8381534bc28cfebf

Citations 6

Show only: □ Literature (6) □ Dataset (0) □ Software (0) □ Unknown (0)
□ Citations to this version

| | | |
|---|---|---|
| Muons as a tool for background rejection in imaging atmosph...<br>Olivera-Nieto, L. et al. (DOI: 10.1140/epjc/s10052-021-09869-0) | 2021 | ADS ARXIV DOI |
| Reconstruction of stereoscopic CTA events using deep learni...<br>Miener, Tjark et al. | 2021 | ADS ARXIV |
| Reconstruction of IACT events using deep learning technique...<br>Nieto, D. et al. | 2021 | ADS ARXIV |
| IACT event analysis with the MAGIC telescopes using deep | 2021 | ADS ARXIV |

521 views · 188 downloads
See more details...

Indexed in OpenAIRE

Publication date:
March 18, 2018

DOI:
DOI 10.5281/zenodo.6842323

Keyword(s):
Imaging atmospheric Cherenkov telescopes · Deep learning · High energy physics · Event reconstruction

Grants:
European Commission:
- ESCAPE - European Science Cluster of Astronomy & Particle physics ESFRI research infrastructures (824064)

Related identifiers:
Derived from git+https://github.com/ctlearn-project/ctlearn.git (Software)

License (for files):
BSD 3-Clause "New" or "Revised" License

Versions

| | |
|---|---|
| Version 0.6.0<br>10.5281/zenodo.6842323 | Mar 18, 2018 |
| Version v0.5.2<br>10.5281/zenodo.5947837 | Feb 2, 2022 |
| Version v0.5.1<br>10.5281/zenodo.5772815 | Dec 10, 2021 |
| Version v0.5.0<br>10.5281/zenodo.4576196 | Mar 3, 2021 |
| Version v0.4.0<br>10.5281/zenodo.3345947 | Jul 22, 2019 |

View all 7 versions

# TOC of Tech Report

- Introduction
  - ESFRI/RI and Partner, Science Case
  - Software and Service Name
- Software/Service Development Strategy
- Software/Service Requirements
- OSSR Integration
  - Status
  - Content
  - User Story

Funded by the European Union's
Horizon 2020 - Grant N° 824064