Object detection for jet physics

Georgij Ponimatkin

Joint work with Jana Bielčíková (NPI CAS) and Josef Šivic (DI ENS/INRIA/CIIRC CTU)

How can Computer Vision help us in solving physics problems?

Computer Vision and jet physics have a lot of things in common

- In jet physics we usually deal with:
  - Observables that lie in high-dimensional space and that are highly nonlinear
  - With processes that we can't describe using "simple" mathematics
  - Many factors of variation in our data
  - Detectors that are in principle high-speed 3D/2D cameras

But there are also differences

- In comparison to Computer Vision the following is missing:
  - We don't have access to reliable ground truth for our data
  - Our data is usually presented in the form of aggregated statistic (e.g. histogram)
- We can synthesize almost infinite amounts of realistic data

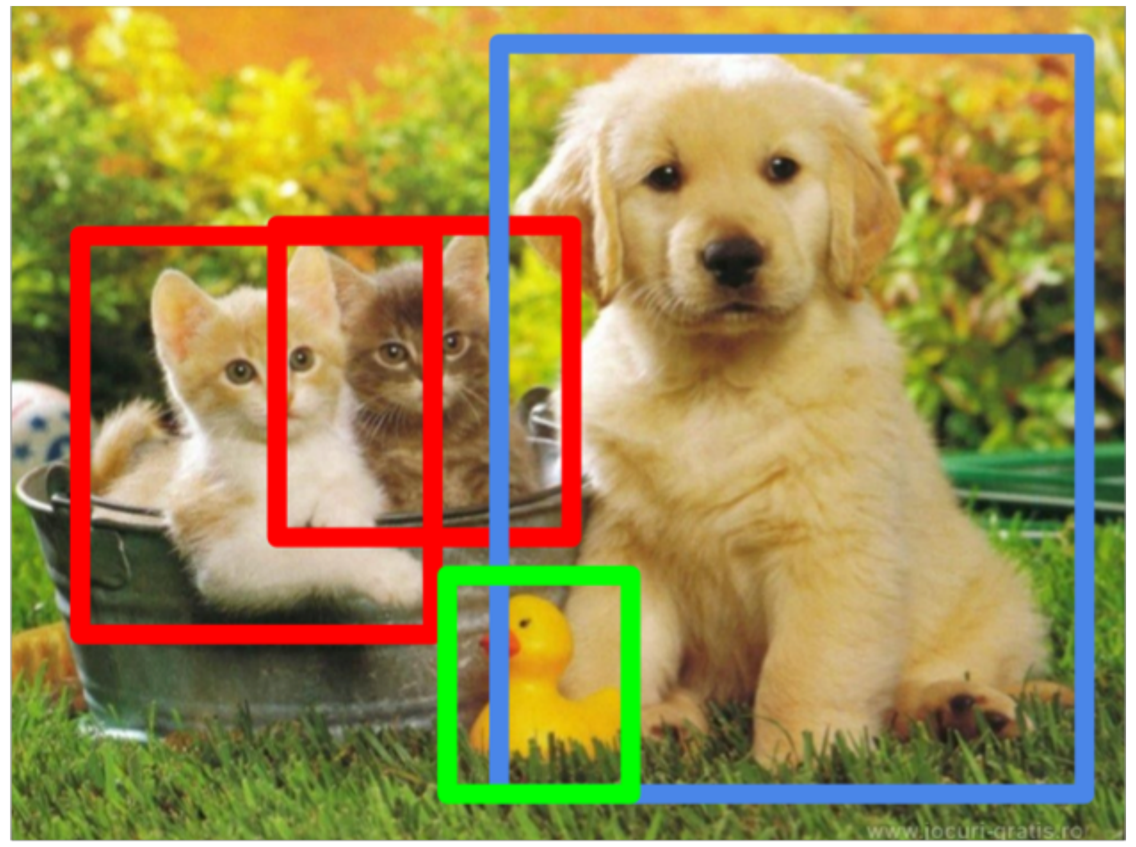
How exactly can we use advancements in Computer Vision for particle physics?

Computer Vision Tasks

Classification

Classification + Localization

Object detection



Cat, Dog, Duck

Instance segmentation

Cat, Dog, Duck

Single object

Multiple object

Credit: Stanford CS231n

Machine Learning

To use recent advancements in Computer Vision we need Machine Learning

- Now we shall briefly introduce main ideas of supervised learning
  - Assume that we have an ordered pair $(\mathbf{x}_i, y_i)_{i=1}^{N}$, which is called dataset
  - Our goal is to construct functional mapping $f: \mathcal{X} \rightarrow \mathcal{Y}$
  - Assume that we have chosen parametric function
  - We will use loss function $L(f(x;\theta), y)$ to quantify model performance
  - For example one can use quadratic loss function $||f(x;\theta) - y||_2^2$
  - Our goal is then to find optimal parameters θ^*, which are given by

$$\arg\min_{\theta} L(f(\mathbf{x};\boldsymbol{\Theta}), y)$$

This parametric function can take many forms

- Linear Model
- Support Vector Machine
- Neural Network

Further on we will focus on convolutional neural networks (ConvNets)

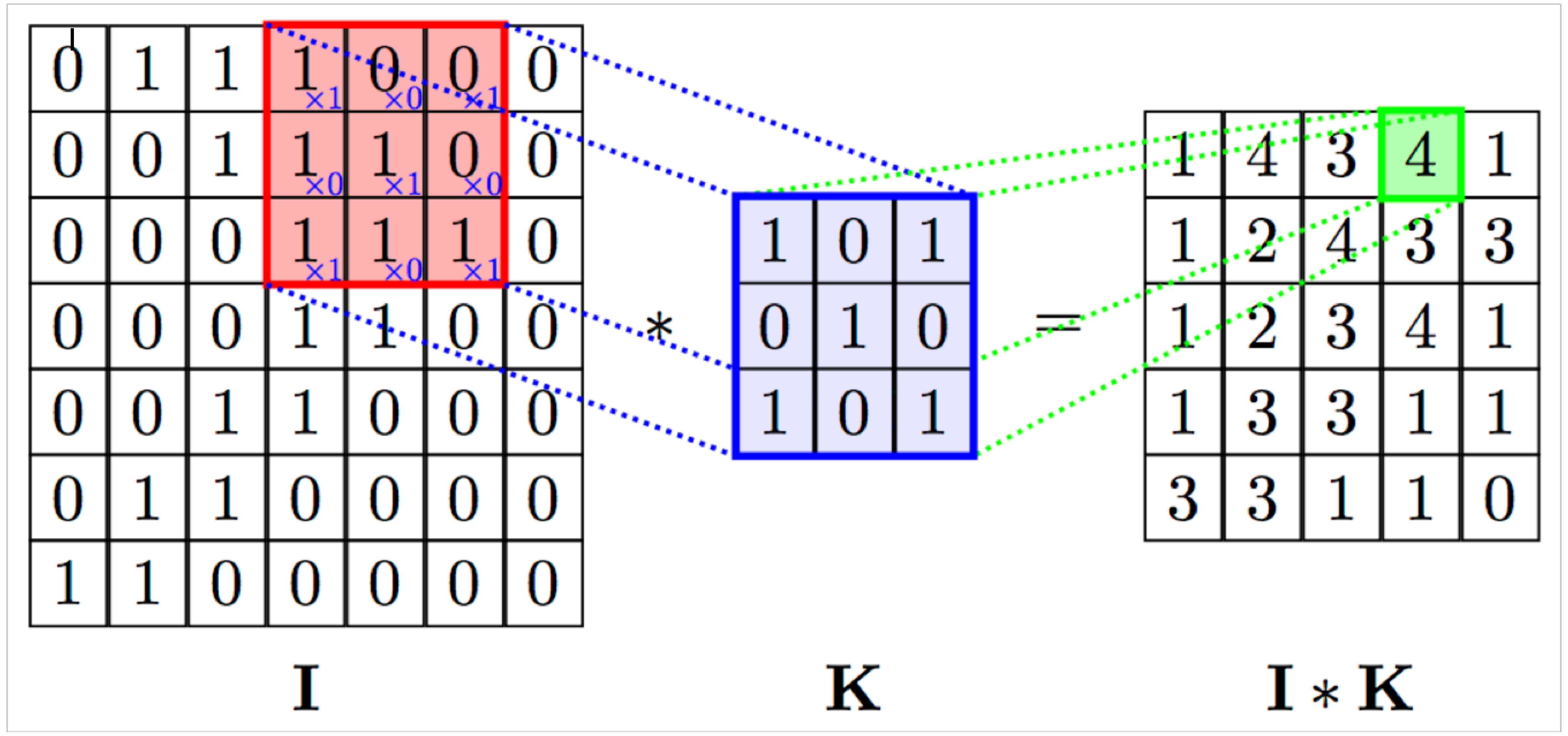
Convolutional Neural Networks

Neural Network is a parametric function composed of many layers

$\mathrm{F}(\mathbf{x}; \mathbf{W}, \mathbf{b}) = f_n(\mathbf{h}_{\mathrm{n}-1}; \mathbf{W}_{\mathrm{n}}, \mathbf{b}_{\mathrm{n}}) \circ \cdots \circ f_1(\mathbf{x}; \mathbf{W}_1, \mathbf{b}_1)$

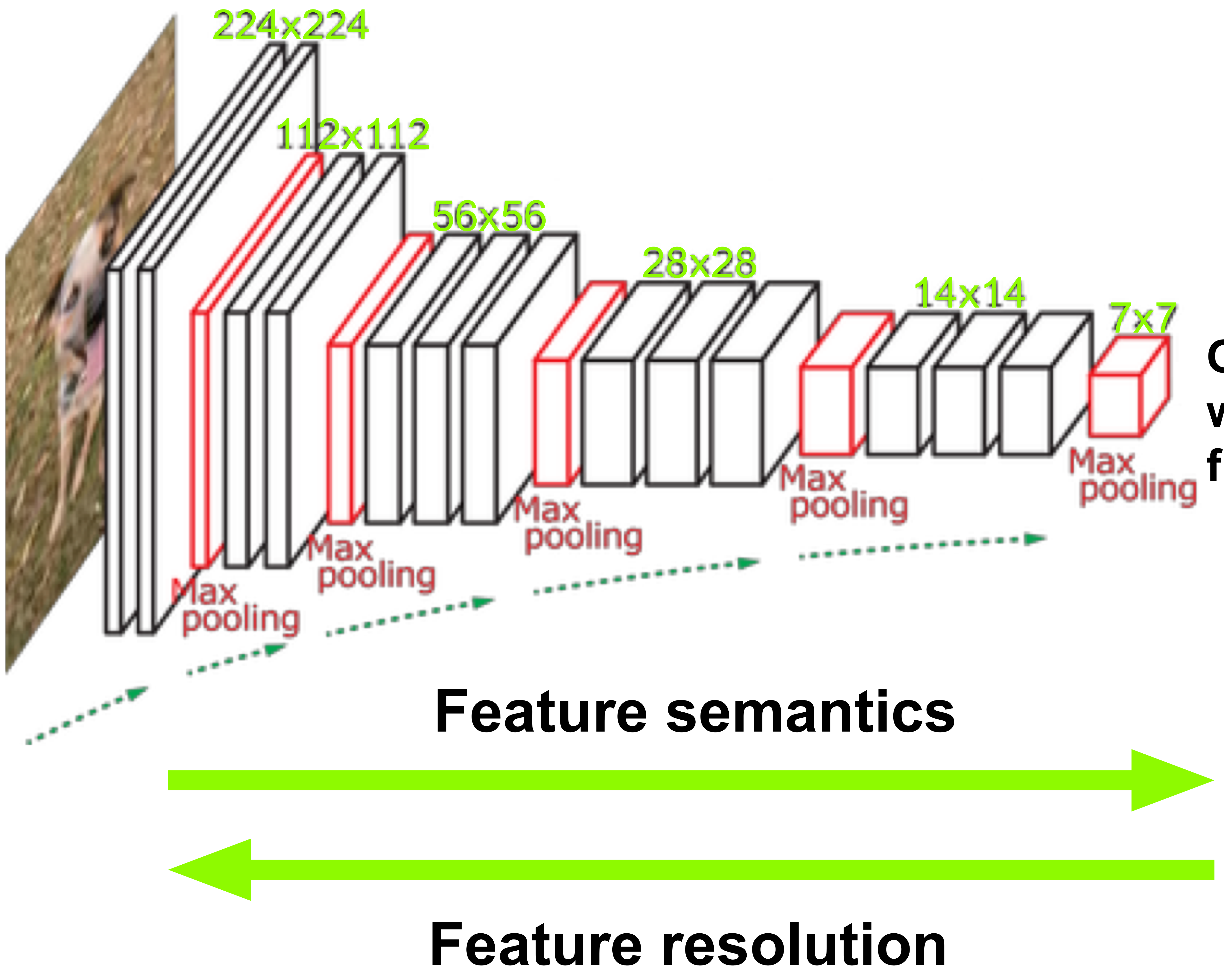
- Here f_i is a layer
  - It first computes affine transformation/convolution between weights and input
  - It is then followed by nonlinearity, nowadays usually $\max(x, 0)$

- We then train our neural network using backpropagation algorithm with SGD
  - In first we use chain rule to compute derivatives of loss function w.r.t parameters
  - Then we update our parameters via $\theta^{t+1} = \theta^t - \eta \nabla L(f(\boldsymbol{x}; \boldsymbol{\Theta}), \mathrm{y})$



Convolutional Neural Network as a Feature Extractor

Input image is a tensor from $\mathbb{R}^{224\times224\times3} \simeq \mathbb{R}^{150528}$



Output is a vector from $\mathbb{R}^{25088}$ which we feed into head for further regression/classification.

Previous Work on Computer Vision for Jet Physics

Previous work was mostly dealing with jet tagging

- Many of the processes were probed using jet image technique, for example:
  - L. de Oliveira et, al, JHEP 07 (2016) 069 – tagging of boosted $W^{\pm}$ vs QCD jets
  - P. T. Komiske et. al, JHEP 01 (2017) 110 – quark vs gluon jet discrimination
  - List continues…

- For the tagging of b/c jets Natural Language Processing models are being used:
  - CMS and ATLAS use recurrent neural network taggers
  - Previous observables can be desribed using jet-shape observables (η, φ, p_T)
  - HF jets require secondary vertex information e.t.c.

Here I'm going to show our efforts towards object detection methods:

- We can use image of the whole event as an input
- Network will implicitly learn rules of anti-k_T clustering
- Model can take into account other factors in the event (for example multiplicity)

Data Generation Pipeline

- We simulate our sensor via TH2F histogram modelling (η, φ, p_T) space
- Pythia8 is initialized with HardQCD:all = on and SoftQCD:nonDiffractive = on
- $\hat{p}_T^{min}$ and $\hat{p}_T^{max}$ are binned so that total jet p_T distribution is uniform
- Event is generated and only final-state particles with appropriate cutoffs are accepted
- FastJet is used to cluster events with anti-k_T algorithm using R = 0.4
- Event and jet masks are then dumped into the TTree
- Each .root files are then converted into HDF5 file

For this talk the train/val/test datasets are splitted as 1M/100K/125K events

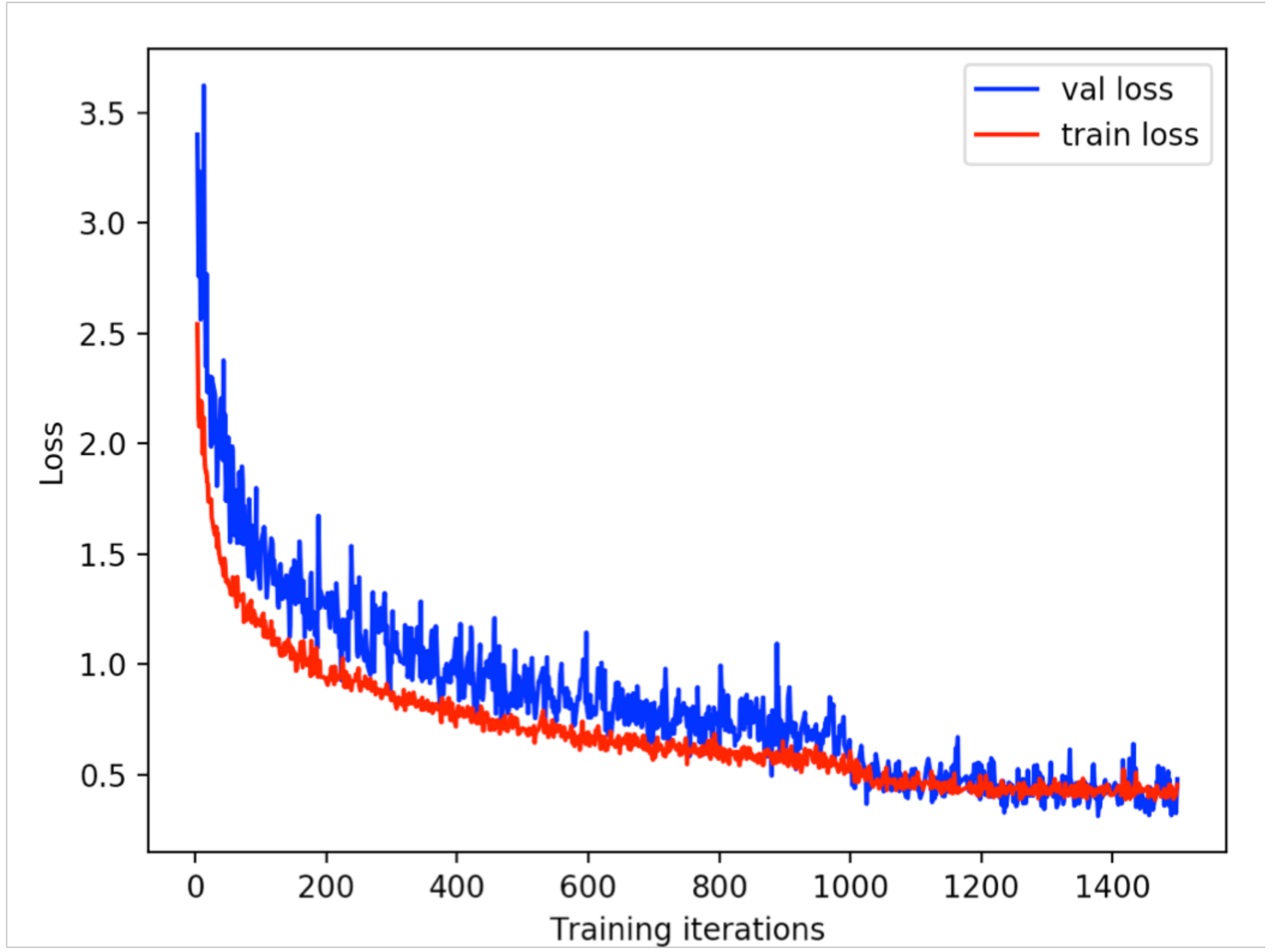
$\|\eta\|< 1$	$p_T > 0.2$ GeV	$p_T^{jet} > 10$ GeV

Selection criteria for particles and jets

Model training

We use Mask R-CNN implementation by Matterport

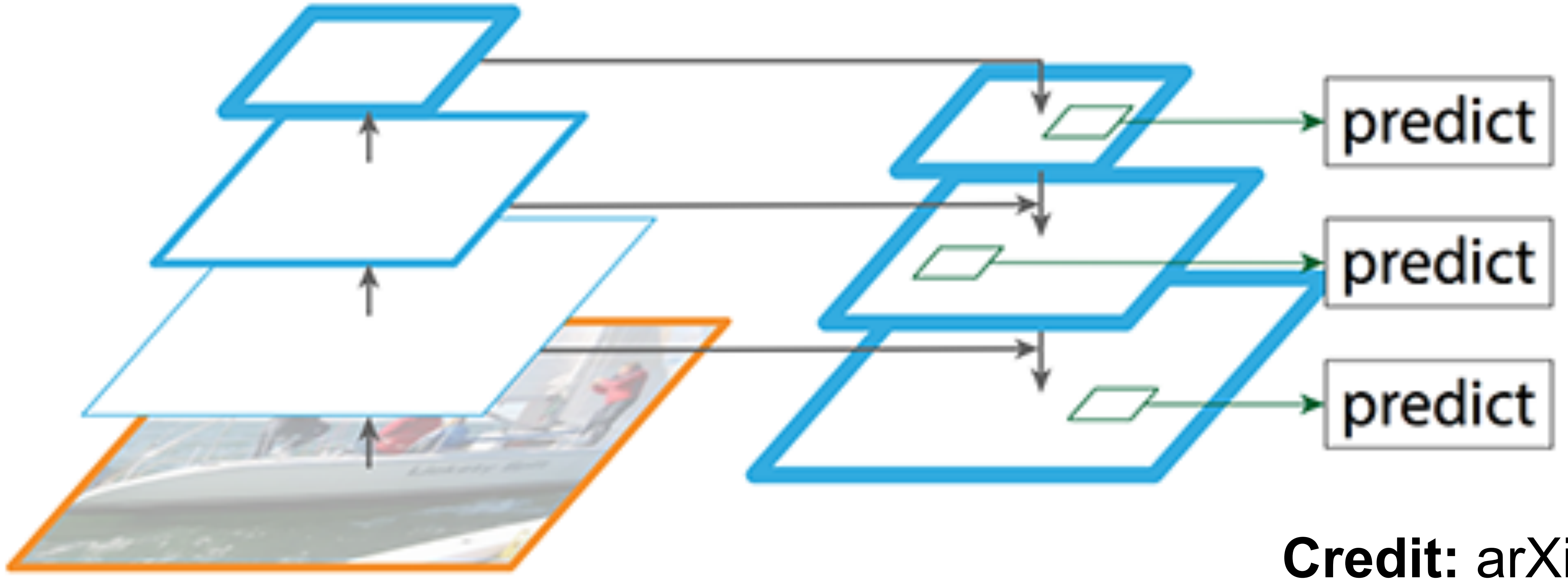
- ResNet18 model is used as backend because the data are visually simple
- Mean pixel value of 0.0022 is subtracted from every image
- Model is trained from scratch
  - Initial LR is set to 0.0025 with stepwise decay at predefined points using Momentum SGD
  - We train for 300,000 iterations in total, with model evaluation every 200 iterations
  - Model weights are initialized to orthogonal matrices (A. M. Saxe et. al, 2013)
  - Each training run takes ~30 hours to complete on 1 Nvidia Titan Xp



$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{bbox} + \mathcal{L}_{mask}$$

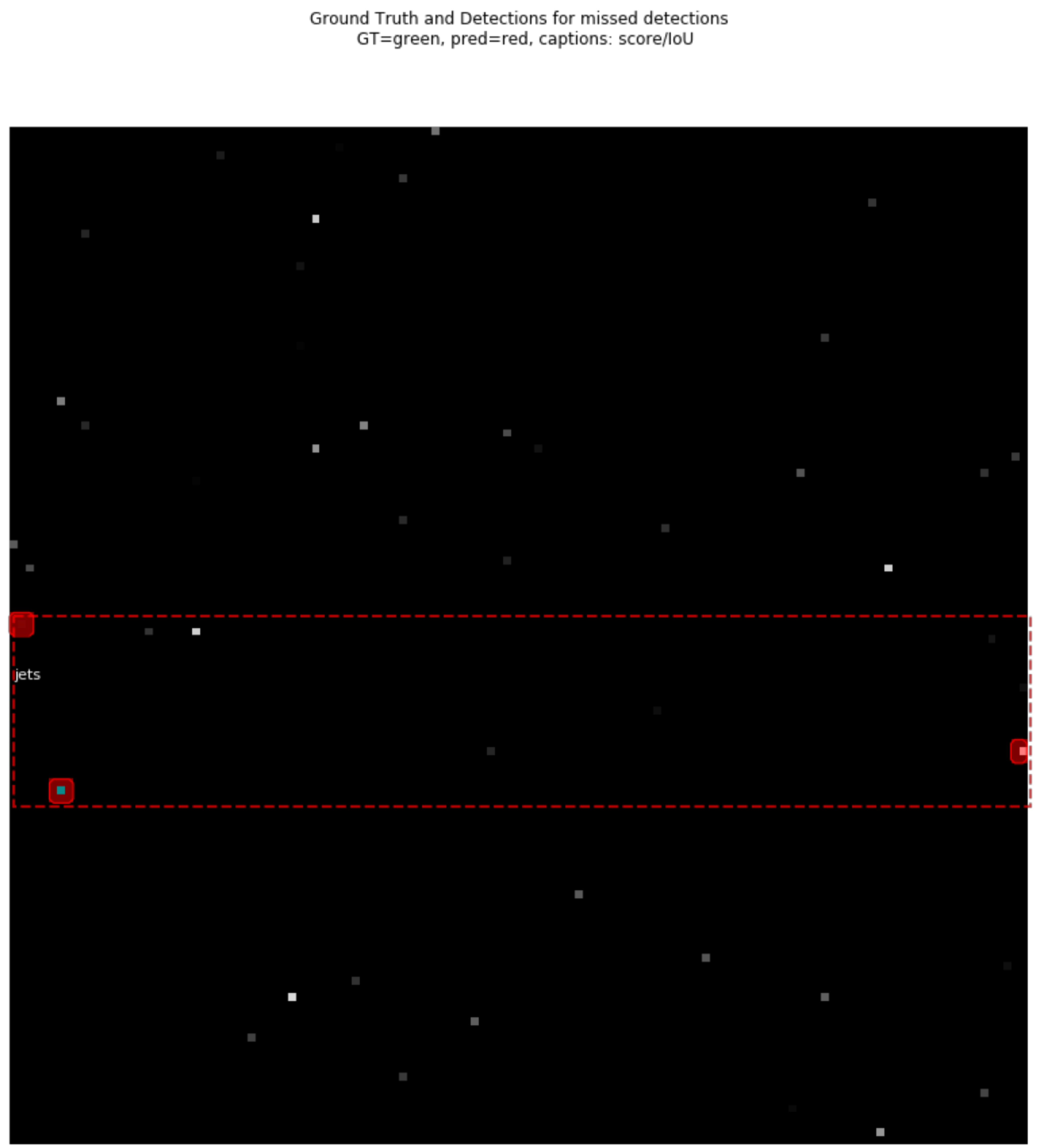
Mask R-CNN - K. He et. al, arXiv:1703.06870 [cs.CV]

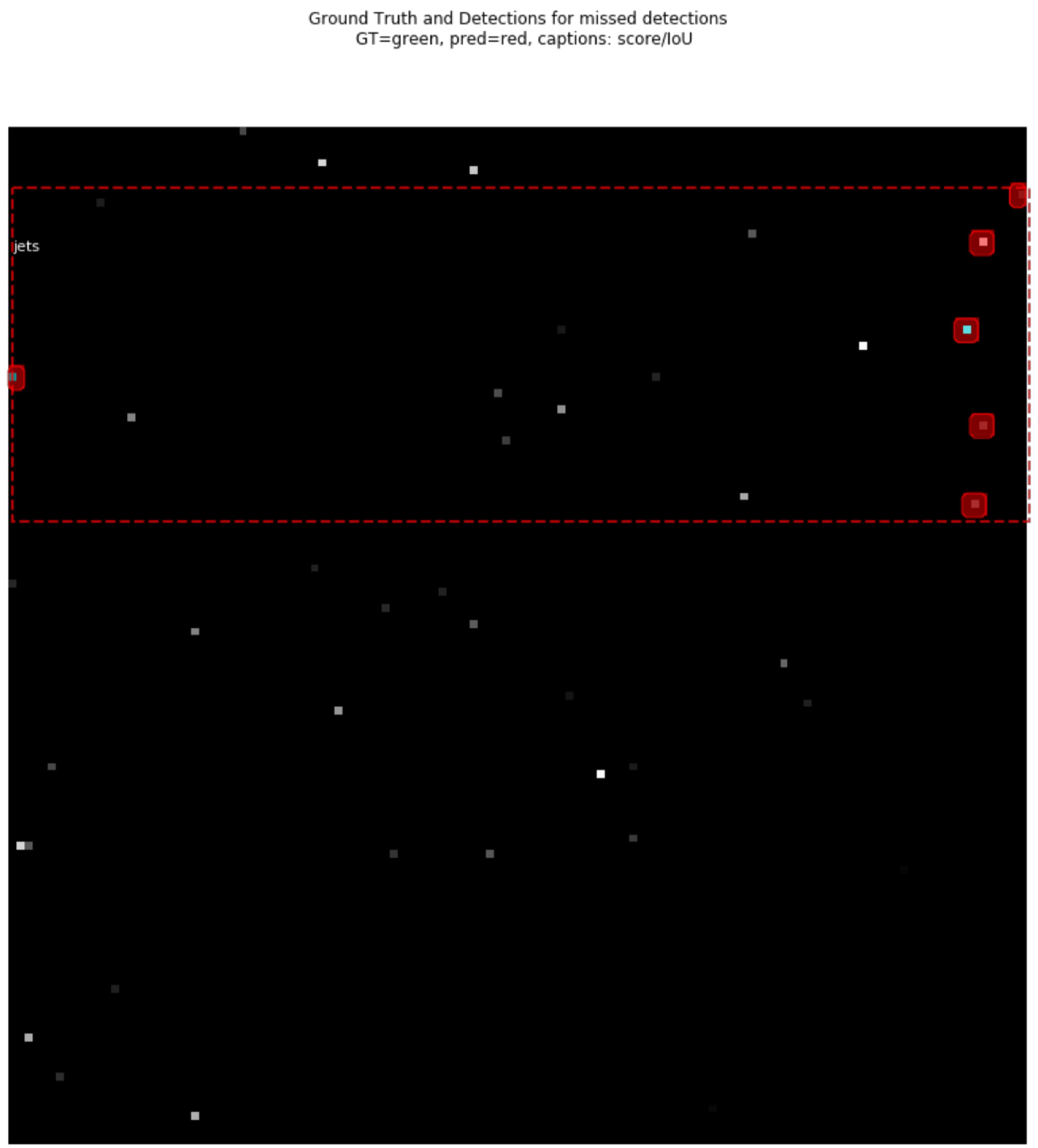
Credit: Jonathan Hui



Credit: arXiv:1612.03114 [cs.CV]

Problem - Discontinuity in azimuthal coordinate

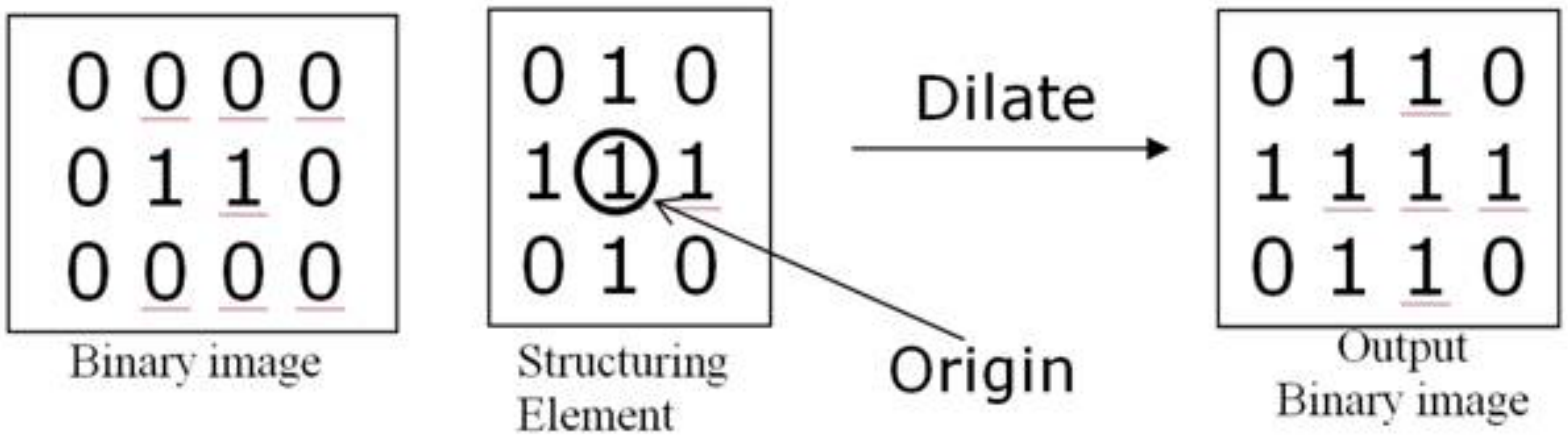




Solving azimuthal discontinuity problem

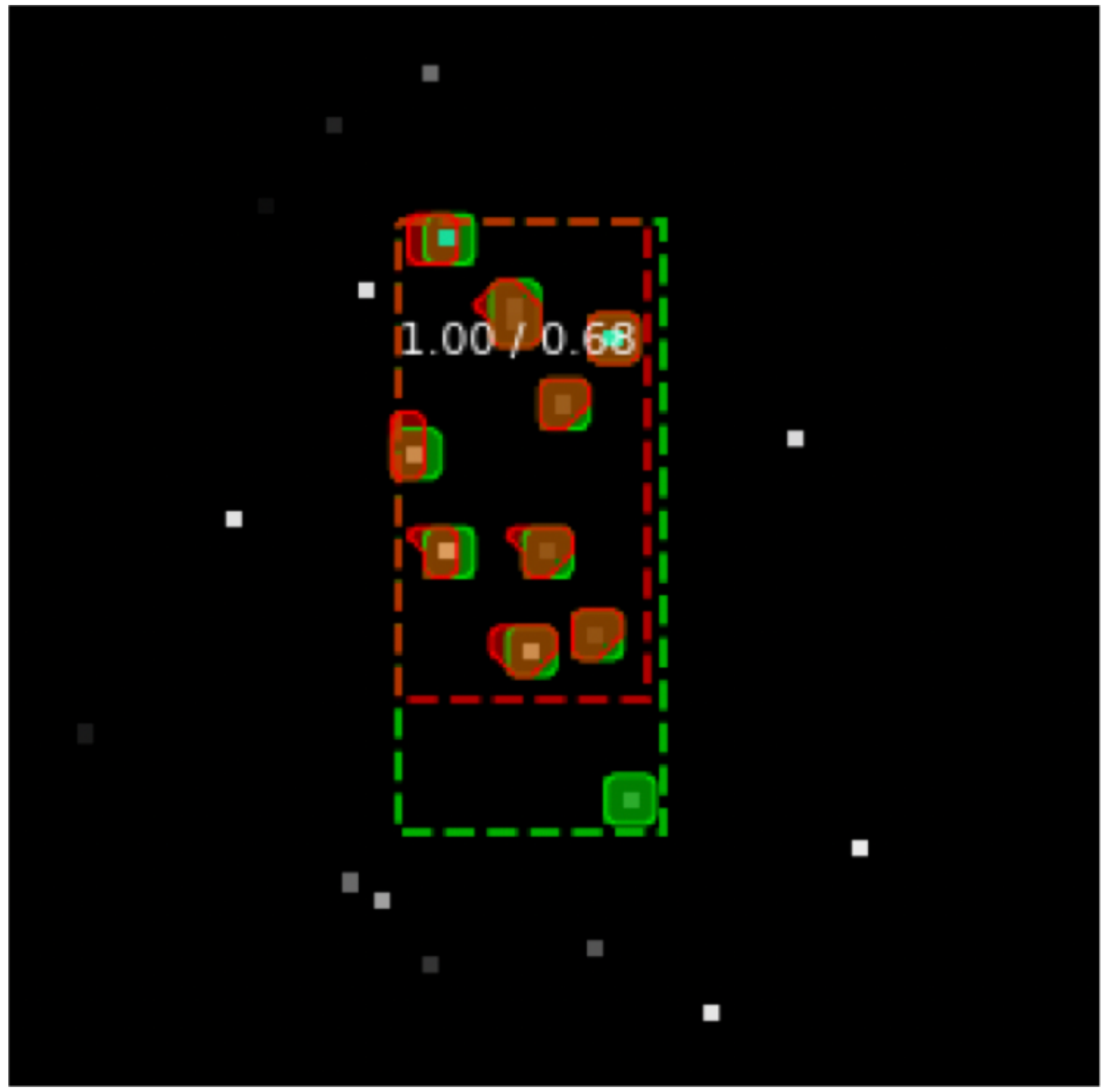
The following procedure was proposed:

- Each image is internally represented by $\mathbf{X} \in \mathbb{R}^{128\times128}$
- Masks are represented by $\mathbf{M} \in \{0,1\}^{128\times128\times N_{jet}}$
- We take each mask $\mathbf{M}_i \in \mathbb{R}^{128\times128}$, which represents single jet
  - We extend φ coordinate from $(0, 2\pi)$ to $(0, 3\pi)$ for image ($\mathbb{R}^{128\times192}$) and mask
  - Using binary dilation operator we enlarge the mask
  - Then the areas of two connected regions are calculated
  - Finally we take AND between biggest region and original mask

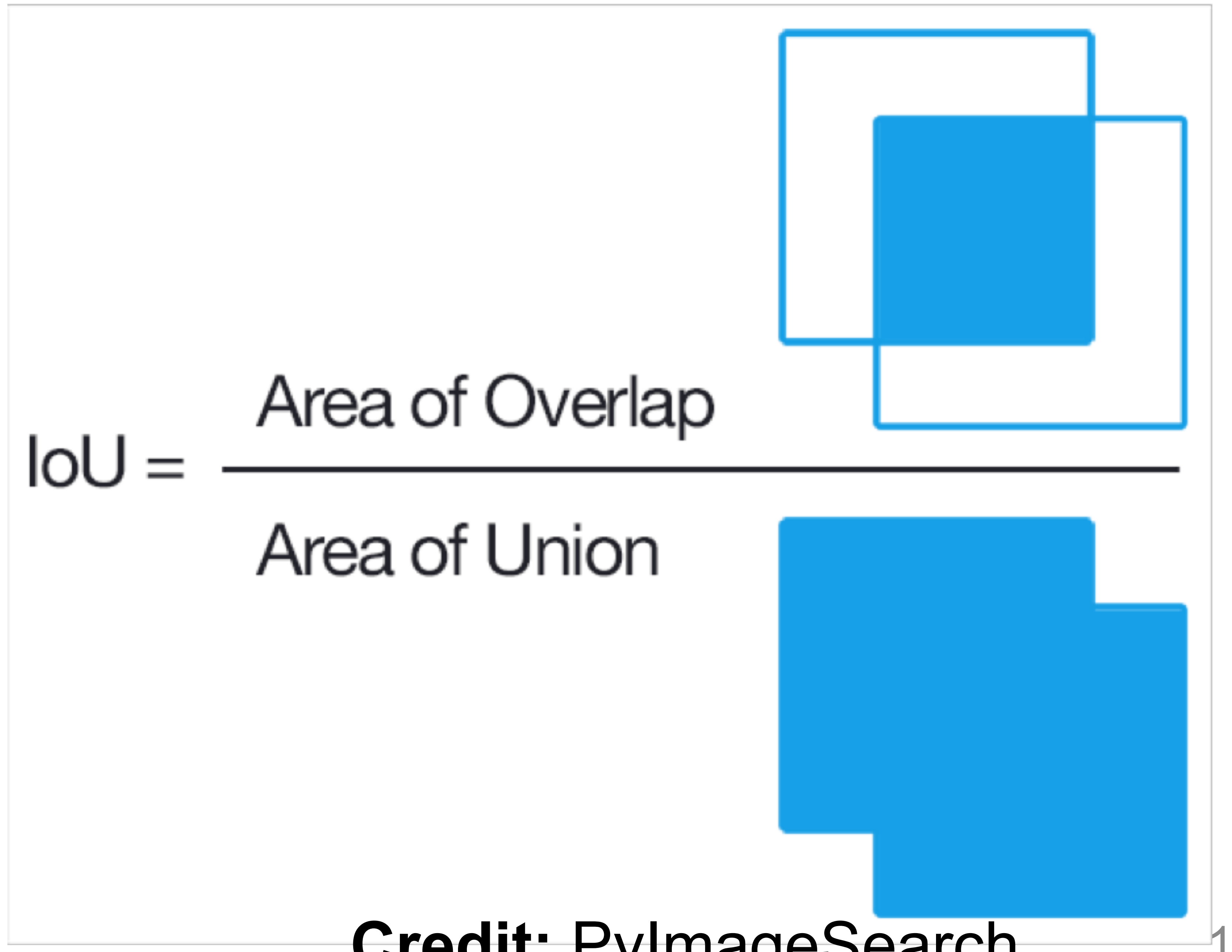


Efficiency calculation

- Efficiency calculation is calculated in each p_T bin as $\#\text{Found jets}/\#\text{All jets}$
- We also use simplified matching procedure:
  - IoU between ground-truth jet and predicted jet shall be greater than 0.3
- Turns out the algorithm is plagued by the false-positive detections
  - For 114754 jets in 112500 events we get 6000 false-positive jets
- We provide two efficiency calculations:
  - First excludes all events where false-positive jets were detected ("lower bound")
  - Second calculates efficiency of all jets, discarding false-positives

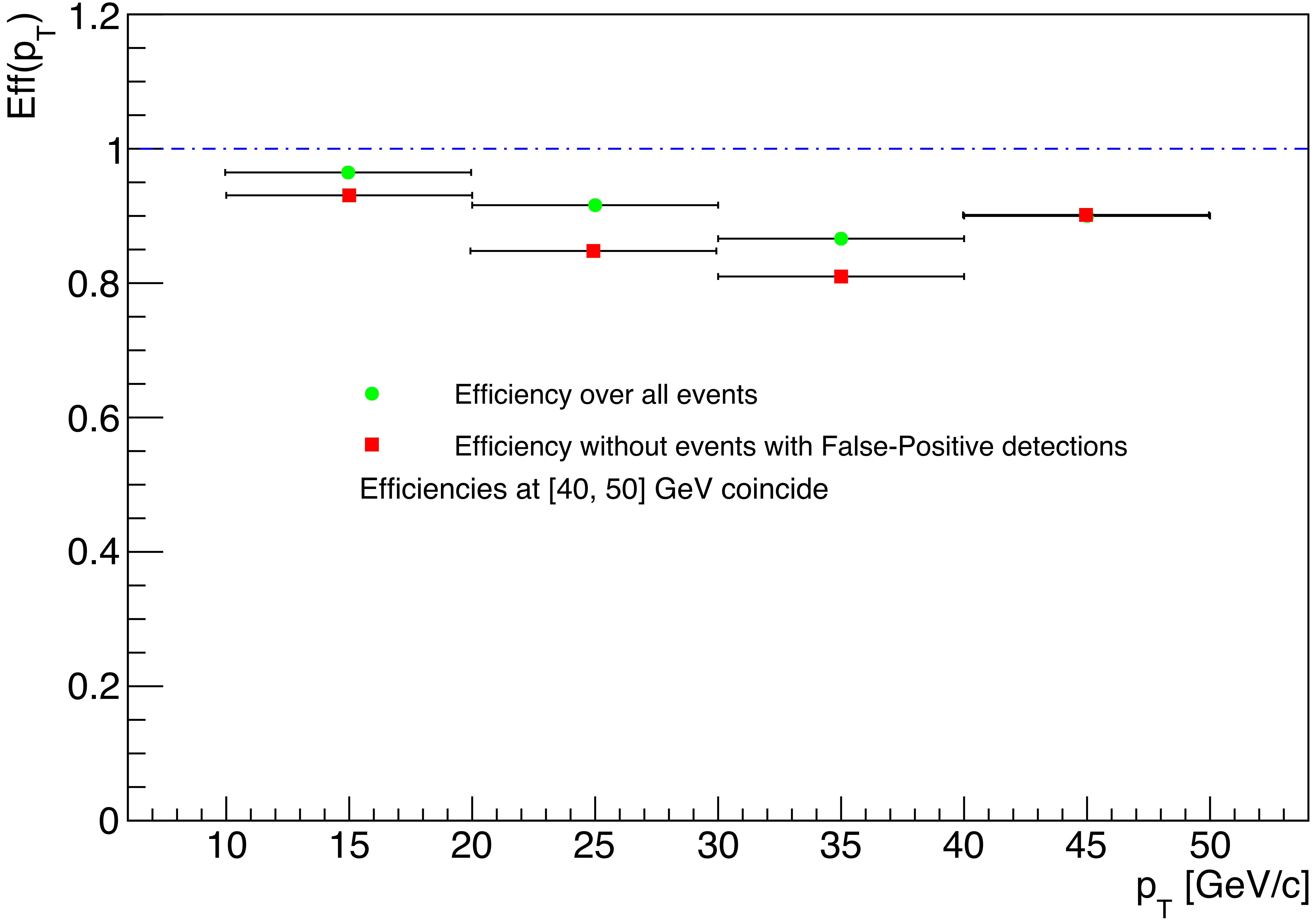


Red – predicted jet
Green – ground truth jet

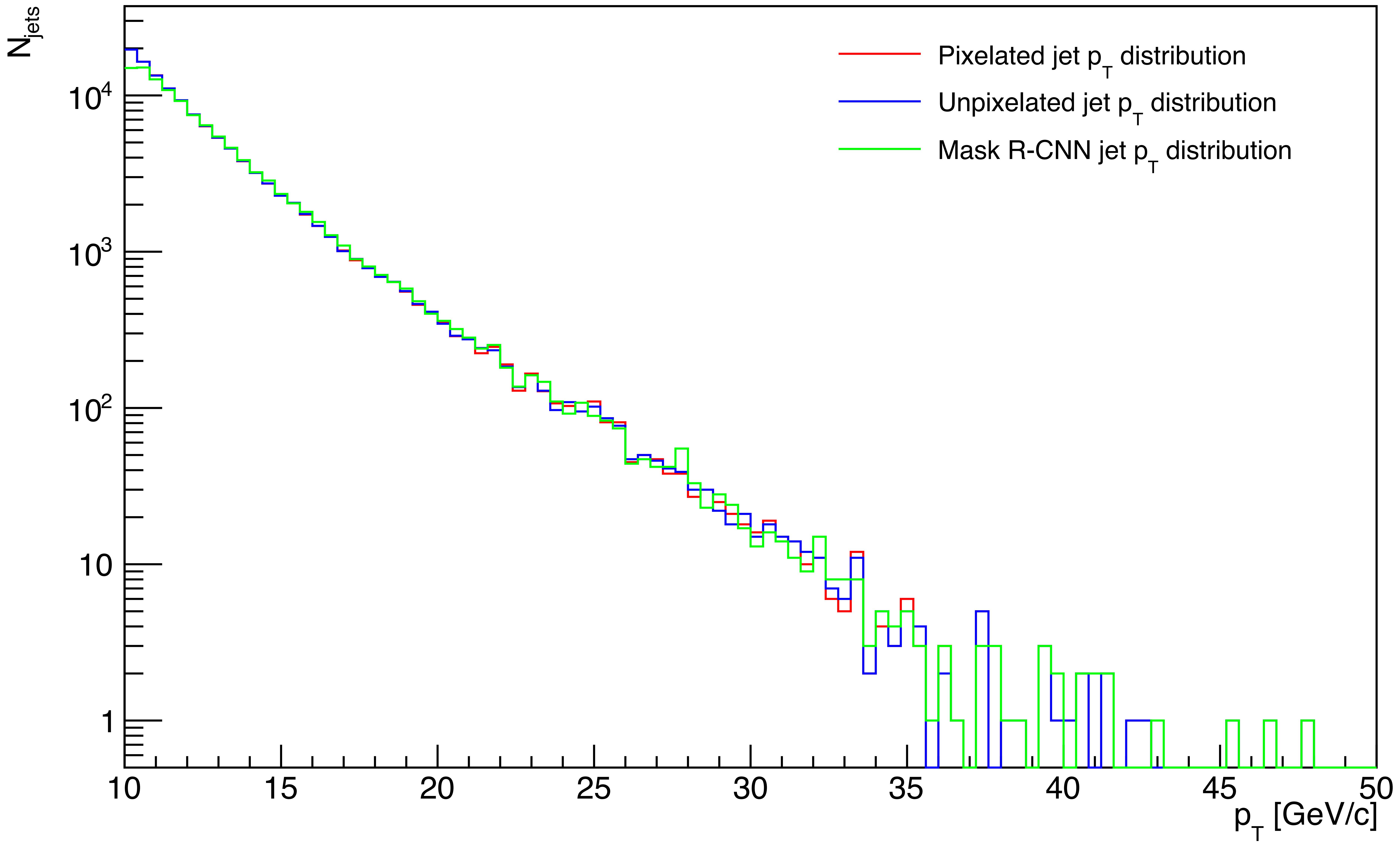


Credit: PyImageSearch

Efficiency dependent on jet transverse momentum



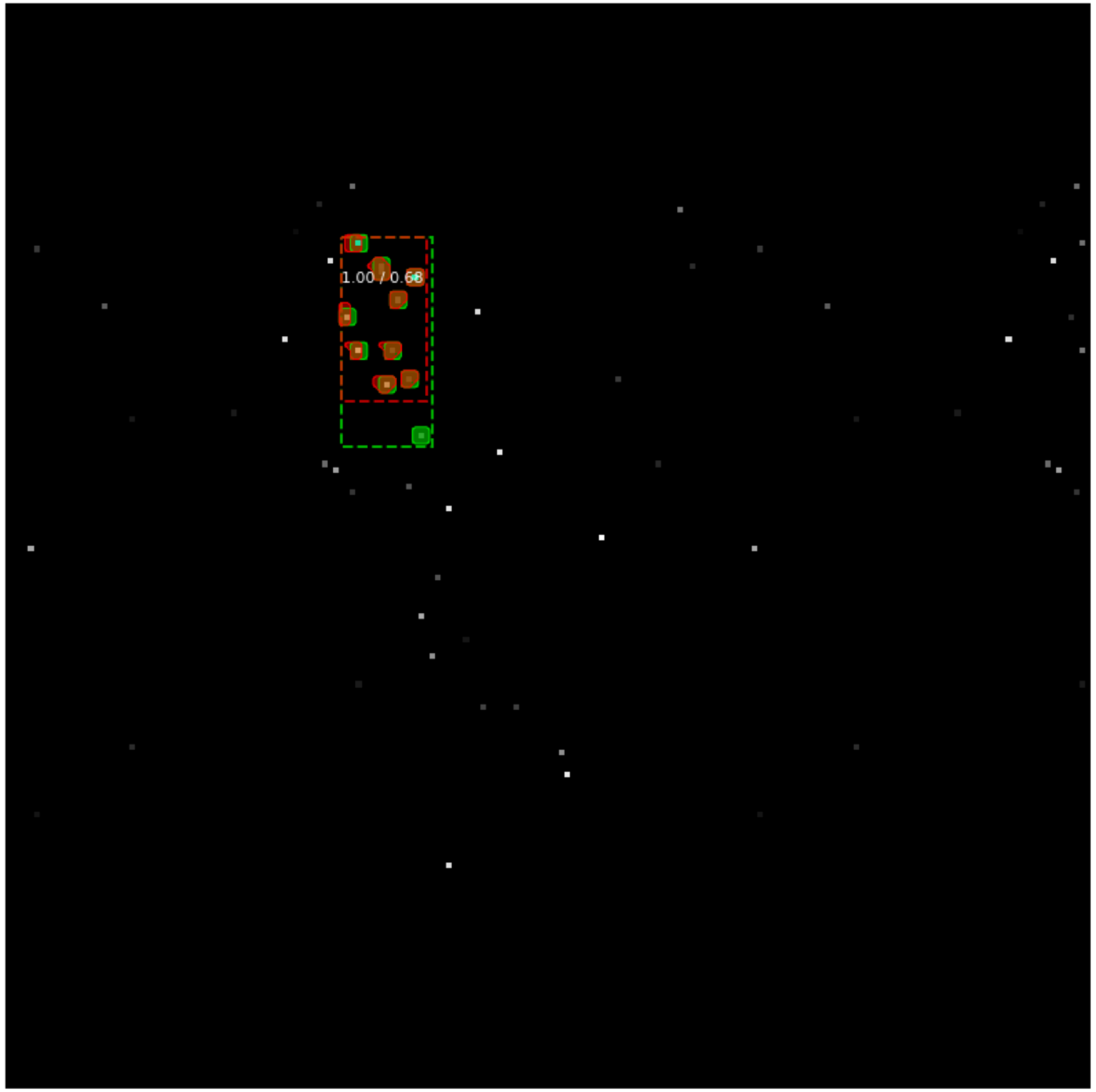
Jet p_T distribution reconstructed by Mask R-CNN



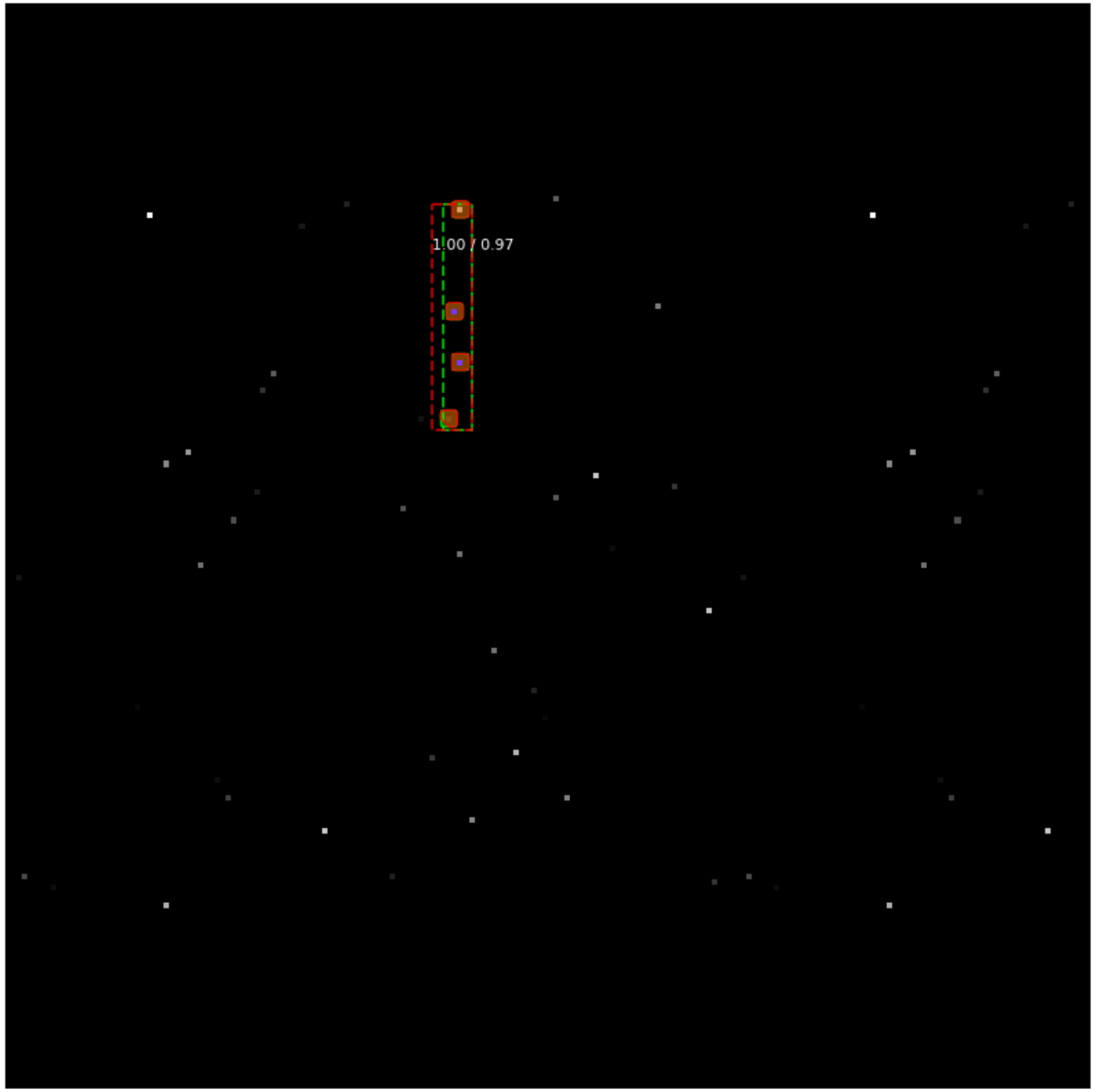
p_T distribution is calculated over all jets (including false-positives)

Detection Examples

Ground Truth and Detections for IoU levels from 1, 0.9, ..., 0
GT=green, pred=red, captions: score/IoU



Ground Truth and Detections for IoU levels from 1, 0.9, ..., 0
GT=green, pred=red, captions: score/IoU



Conclusion and Future Work

We've managed to show that object detection algorithms might be suitable for jet physics

- Model allows us to identify jets directly from single event image
- It also implicitly learns rules of anti-kT clustering
- Further studies are needed:
  - Optimal jet matching technique shall be identified
  - Is there a way to increase efficiency of this method?
- Possible applications?
  - For example, teaching algorithm how unquenched jet signatures look like and then applying it directly to the heavy-ion data. What will response look like?
  - **Can we use this algorithm for event selection?**