EPJ Web of Conferences **245**, 02003 (2020) https://doi.org/10.1051/epjconf/202024502003
CHEP 2019

A deep neural network method for analyzing the CMS High Granularity Calorimeter (HGCAL) events

Gilles Grasseau[1,*], *Abhinav* Kumar[2], *Andrea* Sartirana[1], *Artur* Lobanov[1], and *Florian* Beaudette[1], on behalf of the CMS Collaboration

[1]Leprince-Ringuet Laboratory (LLR), Ecole Polytechnique, Palaiseau, France
[2]Birla Institute of Technology and Science (BITS), Pilani, India

Abstract. For the High Luminosity LHC, the CMS collaboration made the ambitious choice of a high granularity design to replace the existing endcap calorimeters. Thousands of particles coming from the multiple interactions create showers in the calorimeters, depositing energy simultaneously in adjacent cells. The data are similar to 3D gray-scale image that should be properly reconstructed. In this paper, we investigate how to localize and identify the thousands of showers in such events with a Deep Neural Network model. This problem is well-known in the "Vision" domain, it belongs to the challenging class: "Object Detection". Our project shares a lot of similarities with the ones treated in Industry but faces several technological challenges like the 3D treatment. We present the Mask R-CNN model which has already proven its efficiency in Industry (for 2D images). We also present the first results and our plans to extend it to tackle 3D HGCAL data.

1 Introduction

The High-Luminosity LHC is a major evolution of the accelerator and LHC detectors planned to start in 2027. Among the significant enhancements of the CMS Detector, the High Granularity Calorimeter (HGCAL) sub-detector (see Fig. 1) is designed to provide better resolution in the high pseudo-rapidity regions where the particle flux is dense. With HGCAL [1], the CMS collaboration [2] faces to several new challenges:

- The increasing pile-up which will reach about 200 simultaneous collisions,
- The high occupancy (hit energy deposits) of the 6 millions of channels,
- The optimal use of the timing information.

All these factors will involve significant changes in the event reconstruction.

The aim of this work is to explore the capability of modern Deep Neural Network (DNN) technologies to identify the thousands of particles coming from multiple interactions. Each particle produces a cluster of energy deposits in neighboring cells in the calorimeter.

The first task which we will assign to our DNN model is to reconstruct simultaneously the thousands of clusters present in an HGCAL event and to classify them in two categories: dense clusters for the Electromagnetic (EM) particles ($e^{\pm}$, γ, ...) and sparse clusters (or

*e-mail: grasseau@llr.in2p3.fr

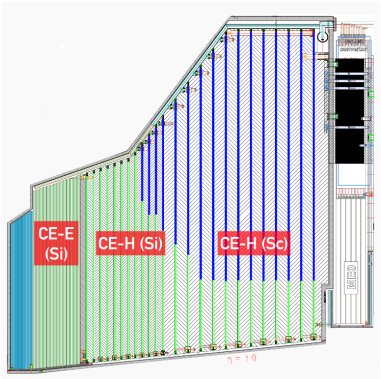

Figure 1. Scheme of the HGCAL subdetector used for this study [1] (not the last design version). Three main kinds of sensors distributed on layers are shown here: the electromagnetic section (CE-E-Si - 28 layers) and the two hadronic sections, one with silicon based sensors (CE-H-Si - 24 layers), the other with scintillator sensors (CE-H-Sc 16 layers).

showers) for hadronic particles (like $\pi^{+/-}$, κ_L^0, n, ...). This problem is well-known in the vision domain and falls in the *Object Detection* class. This class of problems is significantly harder than "only" an image classification/regression because of the mixed goals: the cluster/pattern identification (cluster type in our case), its localization (bounding box), and the object segmentation (mask) in the image.

In the following, we present one of the most famous models in *Object detection*: the Mask R-CNN model, with which we perform a preliminary study on 2D HGCAL images and give inputs to how we will tackle the 3D HGCAL challenge. Other DNN approaches are studied for HGCAL reconstruction, one of the most advanced uses Graph Neural Networks to deal with the complex and irregular detector geometry [3].

2 Mask-RCNN model

The industrial challenges in artificial intelligence are organized on web platforms, like Kaggle [4] where the different models compete. The COCO platform [5] is one of them, dedicated to *Object Detection*, in which the Mask-RCNN model [6] won fame in the last years. It benefits from approximately five years of development from a simple Convolutional Neural Network (CNN) with a sliding window (R-CNN, Fast R-CNN, Faster R-CNN [7–10]) to a more complex model which mixes several neural network components. Today, for *Industrial Vision* or *Object Detection* challenges, other models are in competition in terms of accuracy and processing time : *Single Shot MultiBox Detector* (SSD) [11], *You Only Look Once* (YOLO) [12]. The first implementation of Mask-RCNN can be found here [13].

2.1 Work overview

Our goal is to localize clusters, either EM or hadronic, in the 3D HGCAL data. Due to the complexity of the *Object Detection* models, a preliminary study with HGCAL 2D images resulting from the projection of the 3D data into 256x256 histograms is carried out. It allowed expertise on the hyperparameters of the Mask RCNN models to be acquired. In the first step, the Mask RCNN model should create clusters and:

- predict their localization in the form of bounding boxes,
- assess their class (EM or hadronic), and
- determine their mask.

Then, we can evaluate if this approach is well suited to localize and classify the numerous HGCAL clusters. As will be explained later, it is necessary to build synthetic HGCAL events

to perform this analysis. In the near future, it is foreseen to confirm the results on directly simulated events.

2.2 Mask-RCNN Description

There are 4 main modules in Mask-RCNN architecture (see Fig. 2):

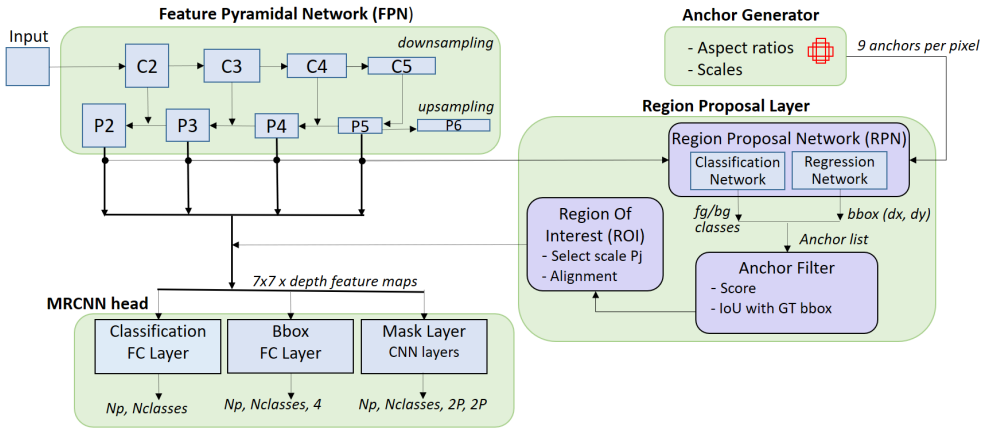

Figure 2. The Mask-RCNN architecture

Feature Pyramidal Network (FPN): It combines different stages of CNN (see [14] for more on FPN) with features maps at different scales C_i. Each C_i is a cutting edge CNN, called ResNet (see [15] for more information). The output feature maps P_i are a combination of low resolution with more semantics coming from the last stage C_5 and a higher resolution (with lower semantic) coming from the C_i.

Anchor Generator: It tiles the image with boxes (called anchors) at different scales and aspect ratios, thus playing the role of a sliding window. Anchors can be generated at the pixel level.

Region Proposal Layer (RPL): Its main role is to find Regions Of Interest (ROI) and to map them to the corresponding P_i feature maps to obtain output feature maps with a constant tensor dimensions (these output feature maps will feed the Mask RCNN head). First, given an anchor, the RPL predicts if the anchor is of interest or not (two classes: foreground/background). In the same sub-module, the Region Proposal Network (RPN), a bounding box prediction is performed. After these two predictions, anchors are filtered according to their score and the ratio of their intersection area over union area (called IoU - Intersection Over Union) between the considered anchor and Ground Truth (GT) bounding boxes provided by the training dataset (see [7, 16] for a more detailed description).

Mask-RCNN head: This module receives one by one the feature maps P_i interpolated on the ROI bounding boxes given by the RPL. Then three neural networks predict the class, the final bounding box and the mask of the feature maps of the ROI.

For our study, we defined two classes: one called *EM* class, for the dense cluster associated with electromagnetic particle and one called here *Pion* class, for the showers associated

with hadronic particles. In this particular case of HGCAL study, the data regularization (or data augmentations) available in the different implementations of Mask RCNN to keep the model not sensitive to optical deformations (zoom, angles with the subject, rotations, ...) must be disabled.

2.3 Loss computation

The *loss* function which controls the whole learning process is composed of five terms, since there are five neural networks in the model to train:

$$L = L^{rpn} + L^{mrcnn}, \text{ with } \begin{cases} L^{rpn} & = L^{rpn}_{class} + L^{rpn}_{bbox}, \\ L^{mrcnn} & = L^{mrcnn}_{class} + L^{mrcnn}_{bbox} + L^{mrcnn}_{mask} \end{cases} \tag{1}$$

The *loss* functions for classification are computed with the cross-entropy [17]:

$$L_{class} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{N_C} P^{th}(c_i = c) \times \log(P^{model}(c_i = c)) \tag{2}$$

where N is the number of predictions, N_c is the number of classes. For the RPN model the number of class is 2 (foreground/background), whereas for the Mask RCNN heads N_c is the number of classes of the model. $P^{th}(c_i = c)$ is the probability that the true class of the i_{th} element is c (i.e. the probability is 0 or 1), and $P^{model}(c_i = c)$ is the probability that the i_{th} predictions belongs to the class c.

For the bounding box regressions in RPN and MRCNN, the *loss* is expressed with a L_1^{smooth} function which is a L2-norm for small values and L1-norm for high values to prevent high weights in the neural networks.

$$L_{bbox} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{4} L_1^{smooth}(x_k^{GT}, x_k^i) \tag{3}$$

x_k^{GT}, x_k^i are respectively the coordinates of the GT bounding boxes and the coordinates of the i^{th} predicted bounding box.

The L_{mask} expression is a binary cross-entropy which is similar to the L_{class} (see [6] for more).

In addition to this main expression of the *loss*, penalization terms can be added like L2-normalization for model weights (also known as weight regularization) to prevent unstable networks and overfitting.

2.4 Building the Dataset

Just like with standard DNN, we need to build a dataset (the training dataset) to learn and another one (the validation dataset) to test the model with different images/events. The model is fed with 256x256 images which represent the 2D projections of the hit energy deposits (3D coordinates). The GT data (classes, bounding boxes and masks) are used iteratively in the training part to compute the *loss* and to find the new weights of the five DNN until the *loss converges*. For the validation part, the model is fed with the 256x256 images of the validation dataset, to find clusters which are classified (EM or hadronic class), localized (bounding boxes) and with predicted masks. The GT data are only used to assess the quality of the model (architecture, model hyperparameters and learned weights), and to check possible overfitting.

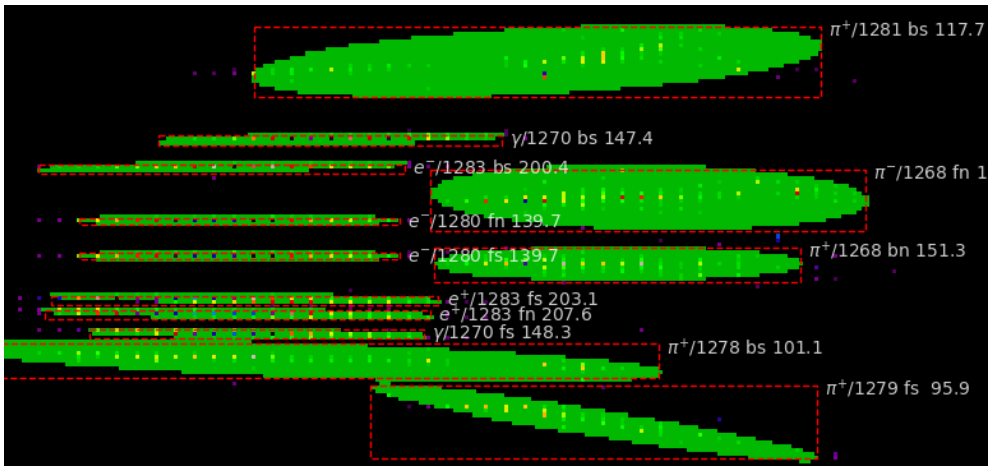

Figure 3. Illustration of a synthetic event (image) used for the training. The different particles coming from different events of one simulation in which we can compute all GT information (used for the training process) are overlaid. The particle labels in the above figure have the following form: particle type, source event ID, "b/f" for backward/forward detector, "s/n" with a mirroring symmetry or not, and the particle total energy. The pixel colors represent the energy deposits in the HGCAL detector (projected on 256x256-sized histograms). The pixel alignment shows the different detector layer positions. In *green*, the masks (ellipses) obtain from a principal component analysis gathering 90 % of particle energy and in *red*, the bounding boxes extracted from the masks/ellipses. All these data: class, bounding box and mask, establish the GT for the learning process.

In the following, a HGCAL design similar to that of [1] is considered and only the hits in the 52 first layers are used (28 from the electromagnetic region and 24 from the hadronic region of the previous version of the HGCAL design - see Fig. 1). These 52 layers are used to build standard 256x256 input images for our Mask RCNN 2D-model.

In our HGCAL study, one of the major problems with Object Detection training is to build a GT dataset. Even if we can easily generate HGCAL events with simulations, getting all the GT data (particle classes, bounding boxes and masks) of all clusters/showers in an event is hardly feasible. Indeed, the amount of information to preserve in simulations would be prohibitive. To get around this problem, we decided to generate complex synthetic events by overlapping simple ones. First, we performed simulations of single particles in the HGCAL detector with the following requirements:

- the particle track must be unique in the forward or backward detector,
- the global particle energy must be greater than a cutoff value (here 20 GeV).

With these criteria, a primary dataset is built with the GT information required for the training (class, bounding box and mask). The 3D deposit energy map is projected on two 2D histograms (one on the $x-z$ plane and one on the $y-z$ plane) to feed the Mask RCNN model with images. The mask of the object is computed by performing a Principal Component Analysis (PCA) used to define a surface enclosing 90 % of the total energy deposit in the detector. The bounding box is calculated from the obtained ellipse.

The final training dataset is produced by overlapping different simple events coming from the primary dataset. The number of object/particles can be chosen as well as the maximal overlapping area between single events. To prevent overfitting (especially on the layer positions) we introduce for *data regularization* (or *data augmentation*) two random processes:

randomly drawing the object symmetry (x/y flip-flop image) and randomly shifting the object by 0, ±1 pixel (equivalent to a 2 cm shift in the detector) on the x or y axes (see Fig. 3).

3 Results

Two different primary datasets in which particles/objects are randomly drawn to build the two independent datasets (training and validation datasets). The number of particles/objects per event is randomly selected from a range of 12 to 20. Datasets containing 5000 and 50 synthetic events (built as explained in Sect. 2.4) have been used for the Mask RCNN training and the validation respectively.

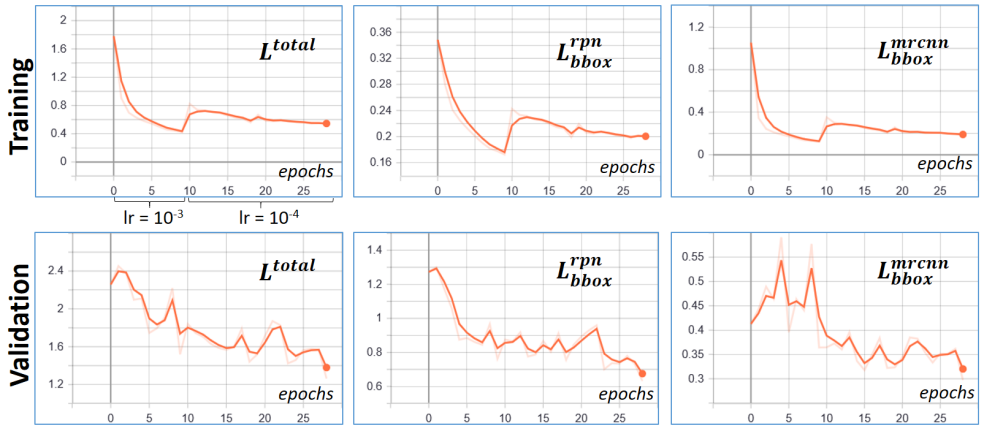

Figure 4. TensorBoard (monitoring tool for DL) plots of the *loss* for the training (top plots) and validation (bottom plots) dataset are displayed. In addition to the total *loss* (left), the most important *loss* contributions are shown: L^{RPN}_{bbox} and L^{rrcnn}_{bbox}. The learning rate is set to 10^{-3} with the 10 first epochs, then set to 10^{-4} up to epoch 30.

The model has been trained on 30 epochs with a learning rate of 10^{-3} for the 10 first epochs then decreased to 10^{-4}. As can be seen in Fig. 4, the model is learning: the *loss* decreases rapidly then seems to converge smoothly. High fluctuations can be observed in the validation plots, caused most probably by the low value of the validation set (50 events). The Figure 4 shows that the total *loss* is dominated by L^{RPN}_{bbox} *loss*, and demonstrates how the RPN is important in the model and attention must be paid to its hyperparameters.

Once the Mask-RCNN model with its hyperparameters is trained, the model is run on the *validation* dataset (this data was not used to train the model) then the predictions are compared with the GT data to compute statistical metrics of the model quality. This mode is called in ML literature *detection* or *inference* or *test*. One of these criteria computed by the Matterport implementation [18] is the *precision* value with 0.5 IoU value, meaning that a True Positive (TP) is found when the overlap area between a GT bounding box and a predicted bounding box is greater than 50 % of the union of the two areas. With these considerations, the *precision* is defined as the ratio:

$$Precision(IoU) = \frac{TP(IoU)}{TP(IoU) + FP(IoU)} \tag{4}$$

In our case, the $Precision(IoU = 0.5) = 0.73$ which is a quite good result for these "out of the box" first measurements . It is illustrated by the different kind of event prediction

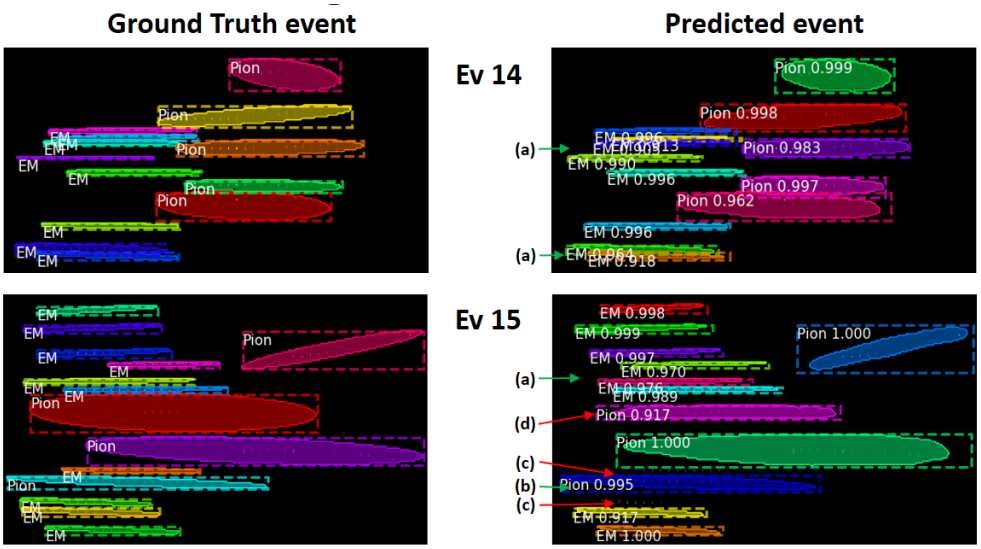

Figure 5. Two event predictions are shown here: (top) one among the best predictions, (bottom) one among the worst predictions. These 2 predictions have been taken from the validation set. The Ground Truth events (EM/Pion classes, cluster bounding box, and the ellipse mask of the cluster) are displayed on the left whereas the predicted clusters (EM/Pion class, bounding box, and mask) are displayed on the right. The probabilities/scores of the prediction are written above the clusters/showers in the right images. Legend: (a) arrows shows the capability to predict dense regions of clusters; (b) the model makes good prediction for hadronic showers which start in the Electromagnetic part of HGCAL detector; (c) the model misses an EM cluster; (d) underestimation of the bounding box and mask of an hadronic shower.

in the Fig. 5. We can notice that the model makes good predictions especially for dense regions and for hadron predictions starting in the EM part of the detector ((a) and (b) arrows in the Fig. 5). However, bad predictions can be observed, mainly particle misidentification (especially for EM class), and for underestimated hadronic shower size ((c) and (d) arrows in the Fig. 5). If the latter can be significantly improved by increasing the number of training events in the dataset, reducing the number of misidentified particles (considered as False Negative or FN) is more difficult. It will require a fine hyperparameter adjustment (especially in the RPN module) for our particular HGCAL-2D model. Globally, the number of FN (or misidentified particle number) is near 15 % of all particles of our data set.

4 Conclusion/perspectives

This preliminary 2D study has been performed in challenging conditions mainly for technical reasons: the small datasets, the rough histogram extraction, with layers far from each other, the hit energy converted in int8 (image pixel), etc. It however gives encouraging results: the Mask-RCNN captures the scattered hits from hadronic showers, dense regions of clusters are well discriminated, however about 15 % of the particles are undetected (or not predicted) by the model. These first results are very promising if we solve the technical problems listed previously. It implies that we have to adapt the Mask RCNN model to the HGCAL topology (low number of layers, removing the energy deposit normalization for each “image”, adjust

the hyperparameters of the model especially those relative to the filtering, modify if required the *loss* function, etc.).

For all these future improvements and adjustments, we will continue to work with the 2D Mask-RCNN model. Our ultimate goal is to implement 3D Mask RCNN. We expect to tackle this new challenge with the evaluation of the 3D implementation Medical Detection Toolkit [19].

5 Acknowledgments

We would like to thank the IN2P3 Project DECALOG/Reprises for helping to present our activities, the Labex P2IO *Accelerated Computing for Physics* for using intensively the GPU platforms to train our model, and Silver Deguzman for contributing in exploring Medical Detection Toolkit implementation [19].

References

[1] CMS Collaboration, The Phase-2 Upgrade of the CMS Endcap Calorimeter, CERN-LHCC-2017-023. CMS-TDR-019 (2017), https://cds.cern.ch/record/2293646

[2] CMS Collaboration, JINST **3** S08004 (2008)

[3] S. R. Qasim, J. Kieseler, Y. Iiyama, et al., Eur. Phys. J. C **79**: 608 (2019), https://doi.org/10.1140/epjc/s10052-019-7113-9

[4] https://www.kaggle.com

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll ar, and C. L. Zitnick, Microsoft COCO: Common objects in context, European Conference on Computer Vision - ECCV **8693** (2014)

[6] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, IEEE International Conference on Computer Vision - ICCV (2017)

[7] R. B. Girshick, J. Donahue, T. Darrell, J. Malik, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, IEEE Conference on Computer Vision and Pattern Recognition (2014), https://doi.org/10.1109/CVPR.2014.81

[8] R. Girshick, Fast R-CNN, IEEE International Conference on Computer Vision - ICCV (2015)

[9] S. Ren, K. He, R. Girshick, and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, Advances in Neural Information Processing Systems - NIPS (2015).

[10] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara,A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al., Speed/accuracy trade-offs for modern convolutional object detectors, IEEE Conference on Computer Vision and Pattern Recognition - CVPR (2017)

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, SSD: Single Shot MultiBox Detector, European Conference on Computer Vision - ECCV (2016), DOI:10.1007/978-3-319-46448-0_2

[12] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You Only Look Once: Unified, Real-Time Object Detection, IEEE Conference on Computer Vision and Pattern Recognition - CVPR (2016), DOI: 10.1109/CVPR.2016.91

[13] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár and K. He, Detectron - Object Detection implementations, https://github.com/facebookresearch/detectron (2018)

[14] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature Pyramid Networks for Object Detection, IEEE Conference on Computer Vision and Pattern Recognition - CVPR (2017)

[15] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, IEEE Conference on Computer Vision and Pattern Recognition - CVPR (2016)

[16] N. Bodla, B. Singh, R. Chellappa, L. S. Davis, Soft-NMS Improving Object Detection with One Line of Code, IEEE International Conference on Computer Vision - ICCV (2017)

[17] Z. Zhang and M. R. Sabuncu, Generalized cross entropy loss for training deep neural networks with noisy labels, Proceedings of the 32^{nd} International Conference on Neural Information Processing Systems, pages 8792-8802, NIPS (2018)

[18] Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow, W. Abdulla (2017), https://github.com/matterport/Mask_RCNN

[19] P. Jaeger *et al.*, Retina U-Net: Embarrassingly Simple Exploitation of Segmentation Supervision for Medical Object Detection", Machine Learning for Health (ML4H) at NeurIPS (2019), https://arxiv.org/abs/1811.08661