Deep Learning Concepts and Datasets for Image Recognition: Overview 2019

Karel Horak*[a,b], Robert Sablatnig[b]
[a]Brno University of Technology, Technicka 12, 616 00 Brno, Czech Republic; [b]Computer Vision Lab, Vienna University of Technology, Favoritenstraße 9, 1040 Vienna, Austria

ABSTRACT

We present basics of a deep learning concept and an overview of well-known deep learning concepts as general Convolutional Neural Networks, R-CNN family, Single Shot Multibox Detector, You Only Look Once architecture and the RetinaNet in the first part of this paper. The all mentioned architectures are described to quickly compare to each other regarding their suitability for given general task. Several selected datasets often used in deep learning competitions are listed in the subsequent chapters in more details. The most known of practically used and listed datasets are COCO, KITTI, PascalVOC and CityShapes. The overview serves as a comparison of the state-of-the-art deep learning methods.

Keywords: Deep learning, dataset, image recognition, convolutional neural network, R-CNN, RetinaNet.

1. INTRODUCTION

In the computer vision domain, a conventional approach of an image recognition is a sequence of image filtering, segmentation, feature extraction and finally rule-based classification. Positives of this approach are explicit task solution and modular structure. As for negatives, it requires high level of expertise, lot of engineering time and it contains many parameters to be manually determined during design and only partial portability to another task is available.

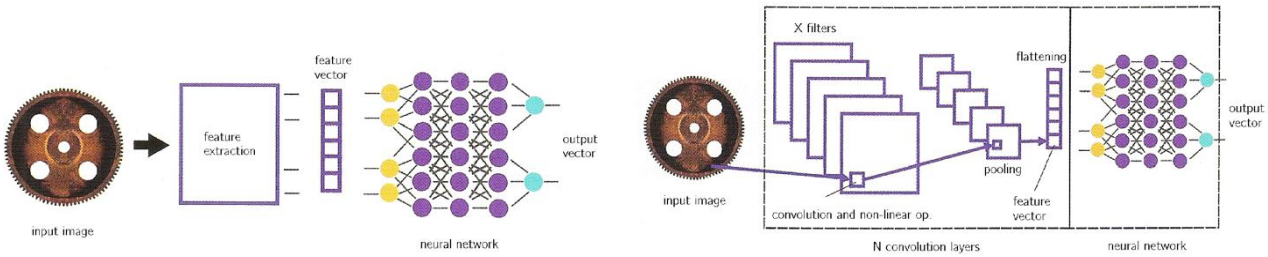

Figure 1. Neural network used as a classifier in the conventional approach (left) and a combination of a neural network-based classifier with convolutional neural network as a feature extractor (right).

On the contrary, using a machine learning approach, algorithms learn a hidden knowledge from a dataset of good and bad samples [1]. Deep learning (DL) is a special case when more than one hidden layer is used in a model [2][3]. Speaking positives of the DL approach, it requires only engineering knowledge of a machine learning tool, not expertise in special areas of machine vision and moreover special implementations of the DL need only tens of learning samples. Unfortunately, the DL approach requires an annotation of good and bad samples, which must be done manually. It represents exhausting and very time-consuming human work, especially in standard traffic applications [4][5] and ADAS systems [6]. Moreover, only partial portability to another task is the same negative as in the classical approach.

2. DEEP LEARNING CONCEPTS

Terms Artificial Intelligence (AI), Machine learning (ML) and Deep learning (DL) are nowadays wrongly used interchangeably in media. For example, when Google DeepMind's AlphaGo program defeated South Korean Master Lee Sedol in the board game Go in March 2016, all three terms were used to describe how DeepMind won. The most basic explanation for these three basic terms could be as follows:

*horak@feec.vutbr.cz; phone 00 420 54114 6417; fax 00 420 54114 6451; http://www.uamt.feec.vutbr.cz/en

Artificial Intelligence: technique used to create a program that mimics human behaviour. Applied commonly to projects, where designed systems show ability to reason, learn, generalize or find meaning.

Machine Learning: uses statistical methods to enable machines to improve with experience - to learn. A subset of AI.

Deep Learning: subset of Machine Learning. Deep learning algorithms are perhaps best exemplified by multi-layer neural networks (NN), which uses multi-layer neural networks to get idea of imputed unsorted data based on learnt traits. Uses basic concepts from brains biology. Useful when there is quantity of input data.

At the very beginning, developers of computer vision systems were using methods like Viola-Jones framework (2001), Histogram of Oriented Gradients (2005) or others, that did not use NN at all. Both were relatively complex with simple reasoning behind them. As computer components got more and more powerful, first techniques employing NN emerged. We went from learning for months for simple NN to learning in a matter of minutes or hours. The most significant breakthrough in using NN in computer vision tasks has been concept of so-called Convolutional Neural Networks tailored to the image recognition.

2.1 General Convolutional Neural Networks

Convolutional Neural Networks (hereinafter CNN) uses sliding window that scans entire picture and for every image window classifier computes probability that an object is present [7]. There is enormous amount of classifications but most of them has small confidence score. Confidence score represent probability or in other words confidence, that object of that category is present. This method works, but is slow due to high amount of comparisons. Thanks to high amount of computations, CNN can be hardly used as real time classifier.

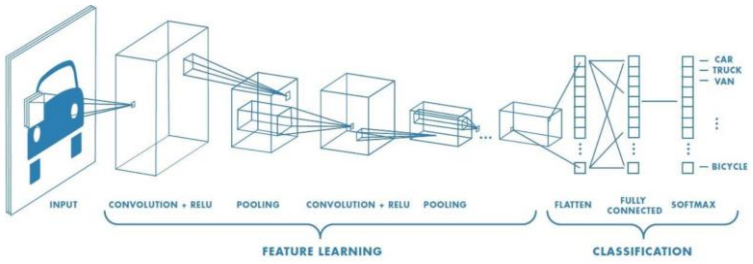

Figure 2. The basic architecture of the Convolutional Neural Network [7].

Algorithm of the CNN can be listed in these four points:

1. Input image is cut into image cuts.
2. On every image cut is applied CNN classifier that computes confidence score for every defined category.
3. Classified tags are stored only if confidence score is higher than predefined threshold.
4. Rectangles around objects with the highest confidence scores are drawn.

The crucial problem in image recognition tasks is that size of object in a picture is not known. Thus, the computational costs rise enormously because of each possible window position should be evaluated. It follows into the very long-term training stage of the DL algorithms. The most often used solution for real application is to use a combination of a so-called pre-trained network running on powerful computing hardware. The general framework of using the pre-trained NN is depicted in the following figure.

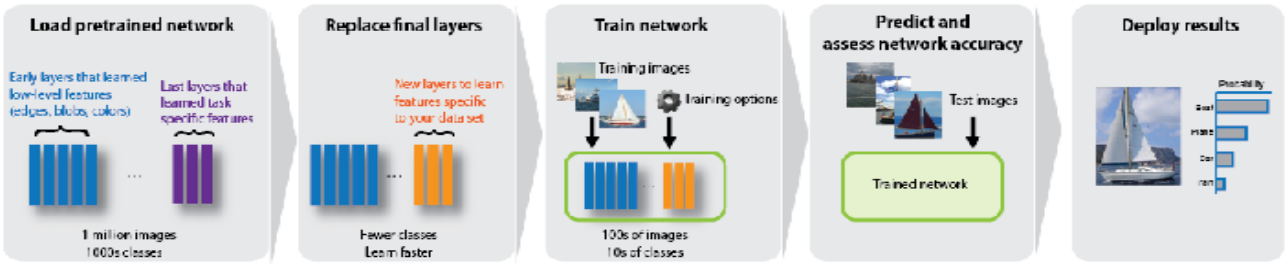

Figure 3. Using pretrained network in generic classification task.

Because of mentioned disadvantage of the generic CNN a lot of modifications were deployed to achieve meaningful computational times in real applications of image recognition using DL concept. Several selected are described in the following chapters and short summary and comparison is at the very end of the list.

2.2 Region-based CNN family

In the R-CNN family there are meta-architectures of CNN which are based on R- CNN. In general, each extension replaces the previous solution. However, R-FCN and Mask R-CNN are significantly different and therefore can be considered as different solutions.

R-CNN

The Region-based Convolutional Neural Networks (hereinafter R-CNN) concept has been published in "Rich feature hierarchies for accurate object detection and semantic segmentation" [8]. The R-CNN, unlike e.g. AlexNet, VGG nets, GoogleNet or Residual Neural Network (ResNet), is used for object detection instead of object classification. It consists of three modules. First one is responsible for generating category-independent region proposals, so called Region of Interest (RoI). These RoIs are served as an input to given CNN, which output is fixed-size vector of features. Last module is a set of linear SVMs (Support Vector Machines), trained for each class independently.

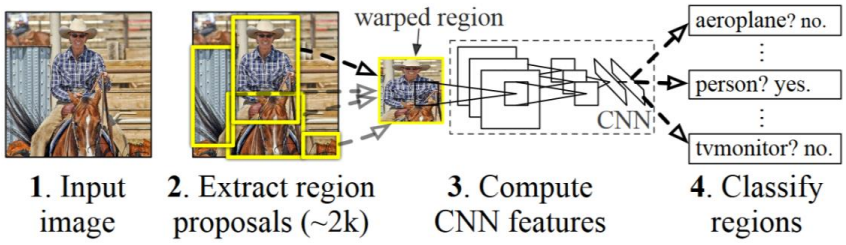

Figure 4. Architecture of the Region-based Convolutional Neural Network.

Despite improved accuracy, this method is not commonly used. The reason for that is its high computational costs. However, this method gave rise of a R-CNN family as described below.

Fast R-CNN

Fast R-CNN [9] is based on R-CNN. It brings several innovations to improve speed and accuracy. Another undisputed advantage is the end-to-end learning. The graphical representation of a Fast R-CNN is displayed in the following picture.

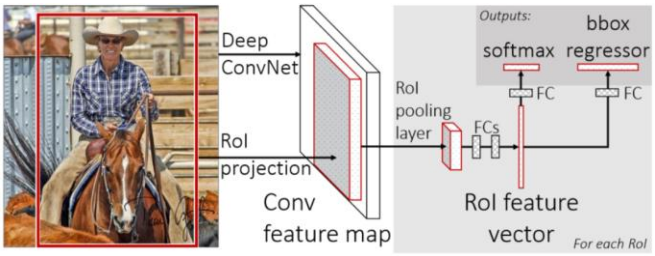

Figure 5. Architecture of the Fast R-CNN [9].

It has a very similar approach as R-CNN. Main difference between these two algorithms is, that in Fast R-CNN the input image is fed into CNN to generate feature map. From the resulting feature maps the regions of proposals are identified and warped into squares. In the next step these regions are handed to RoI Pooling layer, where each individual region is pooled into fixed-size feature map and then mapped to a feature vector by fully connected layers. The output for each RoI is divided into two vectors. First one is a vector of SoftMax probabilities, the second one contains per-class bounding-box regression offsets.

Faster R-CNN

Faster R-CNN is another improvement of R-CNN. It was published in [10] by Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun. Faster R-CNN brought significant acceleration of detection speed that allows a real time detection and furthermore the precision of object detection achieved accuracy of current State-of-the-Art detectors. These are the reasons, why this architecture is one of the most commonly used architectures nowadays. Faster R-CNN changed the

original algorithm for region proposal. The reason for this change is simple. Original region proposal methods are implemented on CPU, while CNNs are implemented for GPUs. Faster R-CNN computes region proposals with a deep CNN. The whole architecture is displayed in the next figure.

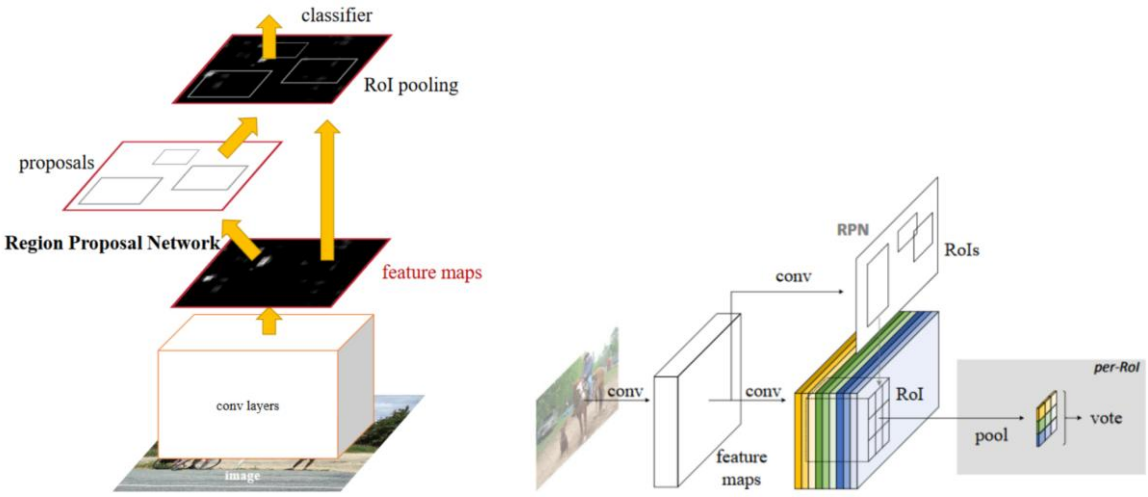

Figure 6. Architecture of the Faster R-CNN [10] (on the left) and the R-FCN [11] (on the right).

The Region Proposal Network (RPN) is responsible for generating object proposals with objectness score for each input image. RPN is much faster and also has better accuracy then the original region proposal because it learns itself to classify foreground and background. RPN is located behind the last convolutional layer. It consists of a fully convolutional network (e.g. like the VGG). Sliding window algorithm with a small network is used on the feature map generated by the last shared convolutional layer of backbone network. Each sliding window is mapped to a lower-dimensional feature which is passed to two fully-connected layers. The first one is a box-regression layer and the second one is a box-classification layer. Each sliding window generates several anchors – small regions with different height/width. Each anchor is centred in the window. During training stage, these anchors are rated by Intersection over Union (IoU). If the IoU of the selected anchor is higher than the threshold for positive, the anchor is marked as positive. Else if the IoU of the selected anchor is lower than the threshold for negative, the anchor is marked as negative. If the IoU of the selected anchor is between these thresholds, the anchor is ignored.

R-FCN

R-FCN was published by Jifeng Dai, Yi Li, Kaiming He and Jian Sun in [11]. R-FCN stands for Region-based Fully Convolutional Network. This architecture is based on Faster R-CNN, but it is fully convolutional. It adopts the two stages strategy for the object detection consisting of the region proposal via RPN and classification of created regions of interest. Given the proposal regions (RoIs), the R-FCN architecture is designed to classify the RoIs into object categories and background. In R-FCN, all learnable weight layers are convolutional and are computed on the entire image. R-FCN ends with a position-sensitive RoI pooling layer which aggregates the outputs of the last convolutional layer and generates scores for each RoI. Position- sensitive RoI layer conducts selective pooling. With the end-to-end training, this RoI layer operates the last convolutional layer to learn specialized position-sensitive score maps. R-FCN uses modified ResNet-101. Basically, it uses only the convolutional layers to compute feature maps. The R-FCN architecture is displayed in the previous figure.

Mask R-CNN

Mask R-CNN [12] is the extension of Faster R-CNN made by Facebook AI Research team. The extension of Faster R-CNN lies in adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. The authors report the speed of about 5 frames per second.

Mask R-CNN uses the same stages as Faster R-CNN but adds the third output – a binary mask for every single RoI. The mask encodes the object spatial layout and therefore it can be addressed by the pixel-to-pixel correspondence provided by convolutions. As the pixel level segmentation is included, the Mask R-CNN can be much more precise while distinguish which parts of RoI belong to the object.

2.3 SSD

The Single Shot Multibox Detector (SSD) is another meta-architecture for the object detection via CNN. It affords different approach to the object detection problem than the R-CNN family. As the name suggests, this method uses only one deep neural network to solve detection problem. The advantage of the SSD is its simplicity, which is related to the speed of this method. Accuracy of this meta-architecture is competitive in the comparison to the other State-of-the-Art methods. This determines it for use in real-time and embedded applications. Originally SSD was published [13] by Wei Liu et al. SSD architecture is displayed in the following figure.

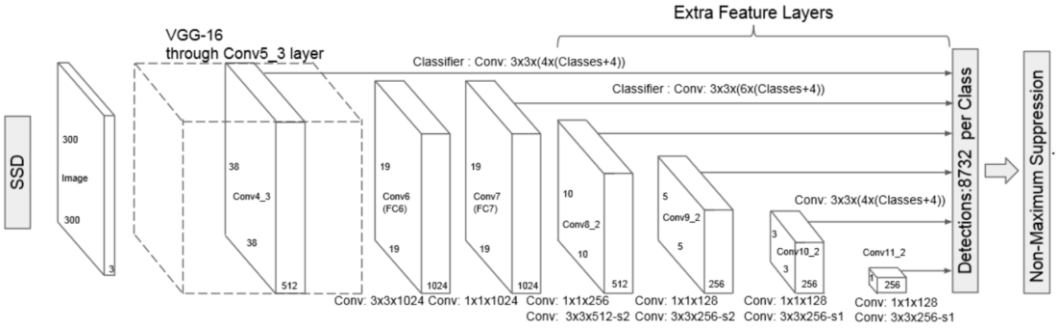

Figure 7. Architecture of the Single Shot Multibox Detector [13].

SSD originally uses truncated VGG16 CNN as backbone. SSD deforms the input image to fixed size of 300x300 pixels or 512x512 pixels because of backbone CNN architecture. There are no fully connected layers in the VGG16 but the last convolutional layer is connected to SSD's convolutional feature layers instead. These layers decrease their sizes and allow the multiple sized objects detection. Each feature layer produces a fixed set of predictions via the set of small convolutional filters. These filters produce either the score for classification or the offset for bounding box since the default bounding boxes are fixed.

2.4 YOLO

The You Only Look Once (YOLO) method is another kind of the SSD. YOLO detector originated earlier than SSD, but this method is still being upgraded. YOLO was originally published [14] by Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi from the University of Washington, Allen Institute for AI and Facebook AI Research. As other single shot detectors, YOLO uses single deep CNN for both classification and detection. The advantage of this method is that unlike sliding window and R-CNN family, the YOLO method sees the entire image during training stage so it can see the object in full background context. It divides the input image into an S x S grid. If the centre of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each cell defines B bounding boxes and score for each class. The confidence scores will be zero, if there is no object in the grid cell. Otherwise the confidence scores will be equal the IOU between the predicted box and the ground truth. Each cell produces the class probability either. The whole YOLO architecture has 24 convolutional layers, followed by two fully connected layers (see the next picture).

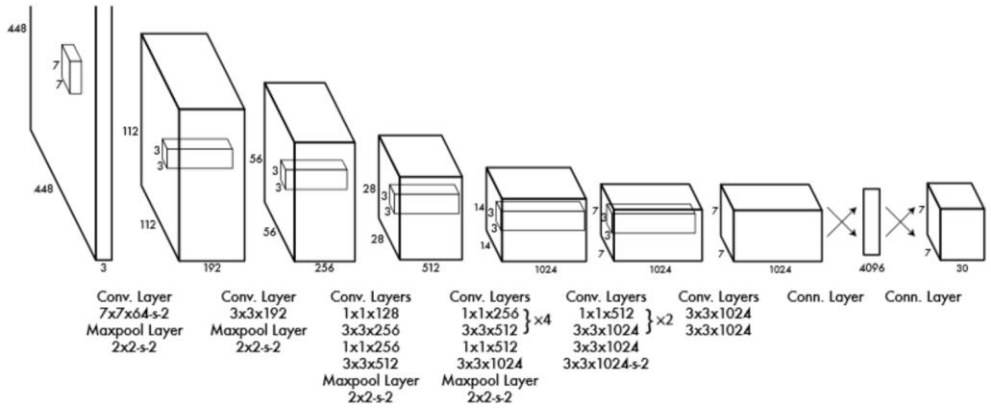

Figure 8. Architecture of the YOLO method [14].

YOLO is not great in detecting multiple small object close to each other (like flock of birds etc.). YOLO also makes multiple predictions of the same object. Non- maximum suppression can be used to remove them. As it has been said, YOLO architecture is being still improved, because the SSD is its main competitor. YOLOv2 also known as YOLO 9000 [15] brings significant improvement of speed and accuracy by batch normalization in convolutional layers. It also changes the training stage, bounding box predictions and several other improvements. The latest version of YOLO named YOLOv3 comes with even more upgrades. It was published by Joseph Redmon and Ali Farhadi in [16]. It has the same accuracy as SSD, but it is three times faster. YOLOv3 uses Darknet53 as the backbone CNN, which is originally a 53 layers deep CNN. Another 53 layers are added for detection purpose. YOLOv3 also predicts ten-times more boxes than YOLO v2. These are the main reasons, why YOLOv3 is slower than YOLOv2. YOLOv3 makes predictions at three scales. They are done by image down-sampling. This makes the small objects easier to be detected.

2.5 RetinaNet

RetinaNet was published by Tsung-Yi Lin et al in [17]. RetinaNet is one stage detector like SSD or YOLO are. It brings novel Focal Loss which focuses training on a sparse set of hard examples and prevents the vast number of easy negatives from overwhelming the detector during training. RetinaNet share lots of similarities with other meta-architectures, like anchors from R-CNN family or feature pyramid from SSD. Next figure shows the principle of this method. This method aims to fill the gap in accuracy between the single shot detectors and two stage detectors, while achieving better speed.

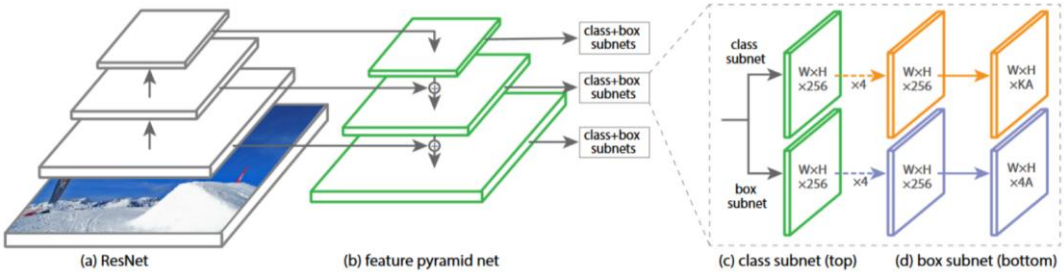

Figure 9. The RetinaNet architecture [17].

RetinaNet uses ResNet CNN architecture as the backbone architecture. On the top of the ResNet there is a Feature Pyramid Network which is used to generate a rich, multi-scale feature pyramid. There are two subnetworks attached by RetinaNet. First of these subnetworks is for classifying anchor boxes, second subnetwork is responsible for the ground-truth object boxes.

2.6 Architectures comparison

There is no easy way to choose which architecture is better because each of the architectures fits to another application. The architecture choice finally depends on the application. This chapter aims to compare State-of-the-Art architectures in terms of speed and accuracy. In the following figure, speed and accuracy of selected meta-architectures are displayed according to [18].

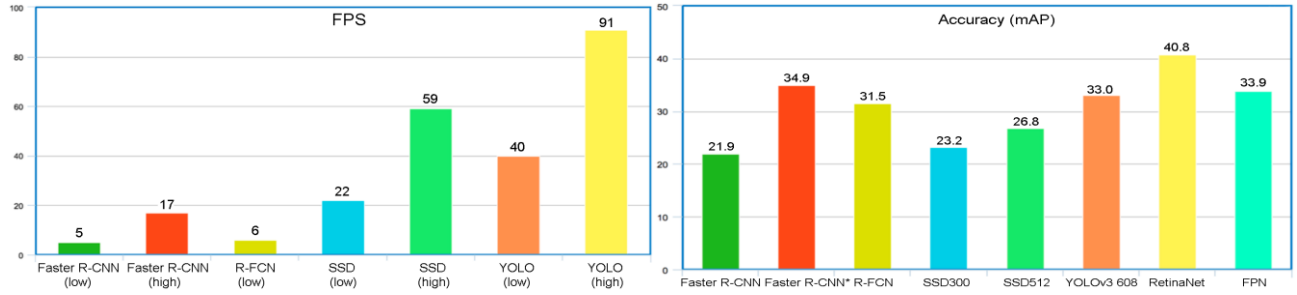

Figure 10. Speed (on the left) and accuracy (on the right) comparison – selected architectures tested on the COCO dataset.

The chart on the left related to speed parameter shows the best and the worst FPS performance. You can see YOLO achieved the best performance whilst Faster R-CNN surprisingly the worst. The second evaluated parameter, accuracy, of each method on the COCO dataset [19] is captured on the right chart.

3. DATASETS FOR DEEP LEARNING

This chapter shortly lists publicly available datasets usable for general object recognition tasks. Majority of the datasets mentioned below provide either labelled (annotated) images and lot of tools for working with datasets like labelling images or evaluating of trained models.

3.1 COCO dataset

COCO stands for Common Object in Context, which is also a name of original paper [20]. It is a large-scale dataset for object detection, segmentation and caption. It contains about 328 000 images and more than 200 000 of them are labelled. There is 80 object categories and 91 stuff categories. It also contains 250 000 people with labelled key points.

Figure 11. Example of the image from the COCO dataset (left) [21] and from the Kitti dataset (right) [22].

3.2 KITTI dataset

Large scale dataset captured by car equipped with two high-resolution colour and grayscale video cameras, Velodyne laser scanner and GPS. Filmed environment contains either urban areas and rural areas including highways. There is up to 15 cars and 30 pedestrians captured in single image. The dataset is divided in three parts. The 2D Object Detection dataset contains 7481 training images and 7518 testing images. Second part is a dataset for 3D Object Detection which consists of 7481 training point clouds (and images) and 7518 testing point clouds (and images). The last part is a Bird's Eye View dataset which consists of 7481 training point clouds (and images) and 7518 testing point clouds (and images). In addition, the entire dataset can be divided according to the difficulty as easy, moderate and hard. This division is done by bounding box height, occlusion level and truncation. For pedestrian detection, there is more than 2000 pedestrians annotated in KITTI dataset images. The dataset is publicly available at [22].

3.3 Pascal VOC dataset

Pascal Visual Object Classes datasets was published for Pascal VOC challenge. The datasets were designed for challenges held between 2005-2012 and are still available. The creators offer both datasets with corresponding annotations and evaluating server. The individual datasets vary both in the range and in the number of classes (since 2007 is the standard of 20 classes). Pascal VOC datasets are available at [23].

Figure 12. Example of the image from the Pascal VOC dataset (left) [23] and from the CityShapes dataset (right) [24].

3.4 CityShapes dataset

Another large dataset from urban environment is called CityShapes. There are 30 classes from urban scenes annotated. The dataset was taken as a record from an on-board camera. The pictures were taken in fifty different locations throughout the daytime and in different weather conditions. Volume of this dataset consist of 5 000 annotated images with fine annotations and 20,000 annotated images with coarse annotations. In addition to images and relevant annotations, the creators also provide metadata such as ambient temperature and GPS coordinates etc. Persons in this dataset are labelled as a person or as a rider. On the top of this dataset there is third-party annotations set named CityPersons, which consist of 3475 annotated images. Both CityShapes and CityPersons are available at [24].

ACKNOWLEDGEMENT

The completion of this paper was supported by the grant No. FEKT-S-17-4234, Industry 4.0 in Automation and Cybernetics by Internal Science Fund of Brno University of Technology and the international mobility project MeMoV No. CZ.02.2.69/0.0/0.0/16_027/00083710 funded by the Ministry of Education Youth and Sports, Czech Republic.

REFERENCES

[1] Watt, J., Borhani, R., Katsaggelos, A.K. Machine Learning Refined. Cambridge University Press New York, USA. 2016. ISBN 9781107123526.
[2] Goodfellow I., Bengio Y., Courville A.: Deep Learning. MIT Press, 2016. ISBN 9780262035613.
[3] Buduma, N., Locascio, N. Fundamentals of Deep Learning. O'Reilly Media, Inc. 2017. ISBN 9781491925614.
[4] Horak, K., et al. Classification of SURF image features by selected machine learning algorithms. 40th International Conference on Telecommunications and Signal Processing, Barcelona, 2017. pp. 636-641. ISBN: 978-150903982-1.
[5] Horak, K., Klecka, J., Novacek, P. License plate detection using point of interest detectors and descriptors. 39th Int. Conference on Telecommunications and Signal Processing. 2016, Vienna. pp. 484-488. ISBN: 978-150901288-6.
[6] Horak, K., Richter, M., Kalova, I. Human eyes localization for driver inattention monitoring system. 15th International Conference on Soft Computing, Brno 2009. pp. 283-288. ISBN: 978-802143884-2.
[7] Rohan, Thomas. Convolutional Networks for everyone. Medium [online]. Jan 15, 2018 [ref. 2018-11-22]. Available on: https://medium.com/@rohanthomas.me/convolutional-networks-for-everyone- 1d0699de1a9d.
[8] Girshick, Ross;donahue. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: *Computer Vision and Pattern Recognition (CVPR)*. New York: IEEE, 2014, p. 580-587 [ref. 2019-01-23].
[9] Girshick, Ross. Fast R-CNN [online]. 27 Sep 2015 [ref. 2018-11-28].
[10] Ren, S., Kaiming H., Ross GIRSHICK a Jian SUN. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [online]. 6 Jan 2016 [ref. 2018-12-08]. Available on: https://arxiv.org/abs/1506.01497.
[11] Dai, Jifeng, Yi LI, Kaiming HE a Jian SUN. R-FCN: Object Detection via Region-based Fully Convolutional Networks. *Arxiv.org* [online]. 2016, 11 [ref. 2019-01-23]. Available on: https://arxiv.org/abs/1605.06409.
[12] He, Kaiming, Georgia GKIOXARI, Piotr DOLLAR a Ross GIRSHICK. Mask R- CNN: Facebook AI Research (FAIR) [online]. 24 Jan 2018 [ref. 2018-12-08]. Available on: https://arxiv.org/pdf/1703.06870.pdf.
[13] Wei, L., Anguelov, D., Erhan, D., et al. SSD: Single Shot MultiBox Detector [online]. 29 Dec 2016 [ref. 2018-12-08]. Available on: https://arxiv.org/pdf/1512.02325.pdf.
[14] Redmon, Joseph, Santosh DIVVALA, Ross GIRSHICK a Ali FARHADI. You Only Look Once: Unified, Real-Time Object Detection [online]. 9 May 2016 [ref. 2018-12- 08]. Available on: https://arxiv.org/pdf/1506.02640.pdf.
[15] Redmon, Joseph; Ali FARHADI. YOLO9000: Better, Faster, Stronger [online]. 2017 [ref. 2018-12-10].
[16] Redmon, Joseph;farhadi. YOLOv3: An Incremental Improvement. *Arxiv.org* [online]. 2018, 6 [ref. 2019-01-23]. Available on: https://arxiv.org/abs/1804.02767.
[17] Lin, T.;. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intellisgence* [online]. IEEE Computer Society, 2018. [ref. 2019-01-23]. DOI: 10.1109/TPAMI.2018.2858826. ISSN 01628828. Available on: https://ieeexplore-ieee-org.ezproxy.lib.vutbr.cz/document/8417976.
[18] Object detection: speed and accuracy comparison. [online], 2018 [ref. 2019-01-23]. Available on: https://medium.com/@jonathan_hui/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-andyolo-5425656ae359.
[19] COCO dataset. COCO dataset [online]. [ref. 2018-12-11]. Available on: http://cocodataset.org/#download.
[20] Lin, Tsung-Yi, Michael MAIRE, Serge BELONGIE, et al. Microsoft COCO: Common Objects in Context. *Arxiv.org* [online]. 2014, 15 [ref. 2019-01-23]. Available on: https://arxiv.org/abs/1405.0312.
[21] COCO: *Common Objects in Context* [online]. Cornell University: Cornell University, Microsoft, 2014 [ref. 2019-01-23]. Available on: http://cocodataset.org/#home.
[22] *KITTI Vision Benchmark Suite* [online]. Chicago: Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago, 2012 [ref. 2019-01-23]. Available on: http://www.cvlibs.net/datasets/kitti.
[23] *The PASCAL Visual Object Classes Homepage* [online]. United Kingdom: Pascal VOC, 2005 [ref. 2019-01-23]. Available on: http://host.robots.ox.ac.uk/pascal/VOC.
[24] City Shapes Dataset: Semantic Understanding of Urban Street Scenes [online]. Germany: City Shapes, 2016 [ref. 2019-01-23]. Available on: https://www.cityscapes-dataset.com.