Computer Science Review 53 (2024) 100645

Contents lists available at ScienceDirect

Computer Science Review

journal homepage: www.elsevier.com/locate/cosrev

Review article

Backbones-review: Feature extractor networks for deep learning and deep reinforcement learning approaches in computer vision

Omar Elharrouss [a,*], Younes Akbari [b], Noor Almadeed [b], Somaya Al-Maadeed [b]

[a] Department of Computer Science and Software Engineering, United Arab Emirates University, UAE
[b] Department of Computer Science and Engineering, Qatar University, Doha, Qatar

ARTICLE INFO

Keywords:
Backbones
Feature extraction
VGGs
ResNets
Deep learning
Deep reinforcement learning

ABSTRACT

To understand the real world using various types of data, Artificial Intelligence (AI) is the most used technique nowadays. While finding the pattern within the analyzed data represents the main task. This is performed by extracting representative features step, which is proceeded using the statistical algorithms or using some specific filters. However, the selection of useful features from large-scale data represented a crucial challenge. Now, with the development of convolution neural networks (CNNs), feature extraction operation has become more automatic and easier. CNNs allow to work on large-scale size of data, as well as cover different scenarios for a specific task. For computer vision tasks, convolutional networks are used to extract features and also for the other parts of a deep learning model. The selection of a suitable network for feature extraction or the other parts of a DL model is not random work. So, the implementation of such a model can be related to the target task as well as its computational complexity. Many networks have been proposed and become famous networks used for any DL models in any AI task. These networks are exploited for feature extraction or at the beginning of any DL model which is named backbones. A backbone is a known network trained and demonstrates its effectiveness. In this paper, an overview of the existing backbones, e.g. VGGs, ResNets, DenseNet, etc, is given with a detailed description. Also, a couple of computer vision tasks are discussed by providing a review of each task regarding the backbones used. In addition, a comparison in terms of performance is also provided, based on the backbone used for each task.

1. Introduction

Artificial intelligence is one of the most research topics dedicated to understanding the real world from various types of data. For time series data, the classification, recognition of patterns, and then making decisions become easier using AI techniques [1]. On image/video data, the processing purpose is to detect or recognize an object, classify the behaviors of a person, analyze and understand a monitored scene, prevent abnormal actions in an event, etc. This is performed using different features which are used to improve the performance of each task. These features are exploited by various techniques starting from traditional statistical methods, passing by neural networks and deep learning, to deep reinforcement learning.

Before, Statistical-based methods were suffering from the finding of patterns, due to the different scenarios of a task, the variation of different aspects, and the data content that should be analyzed. With Machine learning (ML) techniques most of these challenges still exist but learning from various features leads to an improvement in terms of performance. Due to the large-scale datasets that need to be analyzed, ML algorithms cannot process these among of data which prohibits the analyzing of a set of situations that can be contained in it. With the introduction of deep learning techniques including convolutional neural networks (CNNs), the processing of large-scale datasets becomes doable [2]. Also, automatic learning from a large set leads to good learning from a large-scale of features. Nowadays, the use of deep learning models and the combination of deep learning and Reinforcement learning (RL) techniques, which is known under the name of Deep Reinforcement Learning (DRL), lead to a real improvement. While Reinforcement learning (RL) is one of the modern machine learning technologies in which learning is carried out through interaction with the environment and allows taking into account the results of decisions and further actions based on solutions of corresponding tasks [3]. With DL and DRL, data analysis overcomes many challenges with the possibility to learn from large-scale datasets as well as the processing of different scenarios which can lead to generic learning, then robust

* Corresponding author.
E-mail addresses: elharrouss.omar@gmail.com, omar.elharrouss@uaeu.ac.ae (O. Elharrouss), younes.akbari@qu.edu.qa, akbari_younes@yahoo.com (Y. Akbari), n.alali@qu.edu.qa (N. Almadeed), S_alali@qu.edu.qa (S. Al-Maadeed).

https://doi.org/10.1016/j.cosrev.2024.100645
Received 6 September 2021; Received in revised form 4 April 2024; Accepted 23 May 2024
Available online 7 June 2024

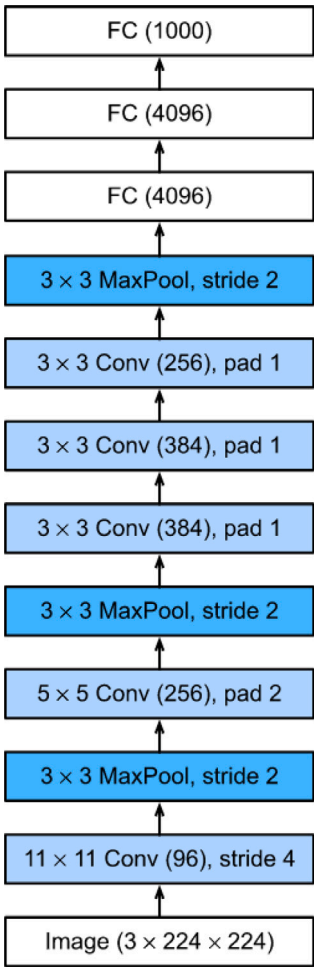


Fig. 1. AlexNet architectures.

Table 1
Summarization of crowd counting methods.

Backbone	Year	# of parameters	Trained task
AlexNet [5]	2012	60M	Img-class
VGG-16 [6]	2014	138M	Img-class
VGG-19 [6]	2014	144M	Img-class
ResNet-18 [7]	2015	11.7M	Img-class
ResNet-34 [7]	2015	25.6M	Img-class
ResNet-50 [7]	2015	26M	Img-class
ResNet-101 [7]	2015	44.6M	Img-class
ResNet-152 [7]	2015	230M	Img-class
Inception-V1 (GoogleNet) [8]	2014	5M	Img-class
Inception-v2 [9]	2015	21.8M	Img-class
Inception-v3 [9]	2015	21.8M	Img-class
Inception-ResNet-V2 [10]	2015	55M	Img-class, obj-det
Darknet-19 2015 [11]	2015	20.8M	Obj-det
Xception [12]	2017	22.9M	Img-class
SqueezeNet 2016 [13]	2016	1.24M	Img-class
ShuffleNet [14](g = 1)	2017	143M	Img-class, obj-det
ShuffleNet-v2[15] (g = 1)	2018	2.3M	Img-class, obj-det
DenseNet-100 (k = 12) [16]	2018	7.0M	Img-class
DenseNet-100 (k = 24) [16]	2018	27.2M	Img-class
DenseNet-250 (k = 24) [16]	2018	15.3M	Img-class
DenseNet-190 (k = 40) [16]	2018	25.6M	Img-class
DetNet [17]	2018	–	Img-class, obj-det
EfficientNet B0-B7 [18]	2020	5.3M–66M	Img-class, obj-det
MobileNet [19]	2017	4.2M	Img-class, obj-det
MobileNet-v2 [20]	2017	3.4M	Img-class, obj-det
WideResNet-40-4 [21]	2016	8.9M	Img-class, obj-det
WideResNet-16-8 [21]	2016	11M	Img-class, obj-det
WideResNet-28-10 [21]	2016	36.5M	Img-class, obj-det
SWideRNet (w_1 = 0.25, w_2 = 0.25, l = 0.75) [22]	2020	7.77M	Panoptic-seg
SWideRNet (w_1 = 1, w_2 = 1, l = 1) [22]	2020	168.77M	Panoptic-seg
SWideRNet (w_1 = 1, w_2 = 1, l = 6) [22]	2020	836.59M	Panoptic-seg
SWideRNet (w_1 = 1, w_2 = 1.5, l = 3) [22]	2020	946.69M	Panoptic-seg
HRNet W32, W48 [23]	2019	28.5M, 63.6M	Human-Pose- est
HRNet V2 [23]	2020	–	Semantic-seg

methods that can work in different environments [4]. For computer vision tasks, features are extracted using different convolutional networks (backbones), while the processing of images/videos can be reached with multi-task learning that can handle scale variations, positions of objects, low resolutions, etc.

In the literature, there is a lack of papers that compare the proposed feature extraction networks for deep-learning-based techniques. The previous works in like in [24,25] provide a broader exploration of deep learning (DL) as the gold standard in machine learning (ML), including the techniques used, challenges, and applications across various domains. While, this work we focused on the solutions used for feature extraction as well as the development of each model based on the limitations compared with the others for the same purpose. For computer vision tasks, the choice of a suitable network (Backbone) for feature extraction can be costly, due to the fact that some tasks are used specific backbones and not suitable for others. In this paper, we attempted to collect and describe various existing backbones used for feature extraction. Presenting the specific backbones used for each task is provided also. For that, a set of backbones are described in this paper including AlexNet, GoogleNet, VGGs, ResNet, Inceptions, Xception, DenseNet, Inception-ResNet, ResNeXt, SqueezeNet, MobileNet, EfficientNet, and many others. Some of the computer vision tasks that used these backbones for feature extraction have also been discussed such as image classification, object detection, face recognition, panoptic segmentation, action recognition, etc. Accordingly, this paper presents a set of contributions that can be summarized as follows:

- Presentation of various networks used as backbone for deep learning (DL) and Deep Reinforcement Learning (DRL) techniques.
- An overview of some computer vision tasks based on the backbones used.
- Evaluation and comparison study of each task based on the backbone used.
- Summarization of the most backbones used for each task.
- Presentation of deep learning challenges as well as some future directions.

The remainder of this paper is organized as follows. An overview of the existing backbones is presented in Section 2. The computer vision tasks that used these backbones are presented in Section 3. Comparison and discussion of each task according to the backbone used in Section 4. Challenges and future directions are provided in Section 5. The conclusion is provided in Section 6.

2. Backbone families

feature extraction is the main step in data analysis domains. Before the feature extraction was provided using statistical algorithms or some filter applied to the input data to be used in the next processing steps. With the introduction of machine and deep learning (DL) techniques, the use of neural networks makes a revolutionary evolution in terms of performance and the number of data that can be processed. Then, the development of convolution neural networks (CNNs) makes the work on the large-scale size of data possible and is also used for feature extraction. The selection of a CNN network for feature extraction or the

Table 2
Summarization of crowd counting methods.

Backbone	Advantage	Disadvantage
AlexNet [5]	Architecture with 8 layers, is able to extract features and worked well for the time with color images.	Small depth which takes more time to achieve higher accuracy results.
VGG [6]	Improved in terms of depth compared to AlexNet model and also introducing pre-trained models	The model experiences the vanishing gradient problem
ResNet [7]	Large number of layers can be trained easily without increasing the training error percentage. Help in tackling the vanishing gradient problem using identity mapping	Require a large-scale dataset to train the model as well as more complexity cost due to the use of residual connections.
Inception [9]	Interested in finding a solution to scale up neural nets without increasing computational cost. while 5 × 5 CONV is used for capturing global features and 3 × 3 CONV is used for capturing small and distributed features. The 1 × 1 CONV is used for depth reduction.	Performance decreases with the increasing number of neurons per fully connected layer.
Inception-ResNet [10]	Introduction of residual connections led to dramatically improved training speed for the Inception architecture	If the number of filters exceeded 1000, the residual variants started to exhibit instabilities and the network has just "died" early in the training,
Xception [12]	Has the Same number of parameters as Inception, but with greater computational efficiency. Also, it can be used for mobile applications	The same like Inception network, the performance decreases with the increasing number of neurons per fully connected layer
SqueezeNet [13]	The intuition is that large activation maps (due to delayed downsampling) can lead to higher classification accuracy	The lightweight network provides less accuracy than other state-of-the-art networks except AlextNet network.
ShuffleNet [15]	Network with reduced computation cost, suitable for mobile device applications with very limited computing power.	Less accurate than the other networks
DenseNet [16]	Strong Gradient Flow. Much smaller in size than ResNet	Each layer receive all preceding layers as input which augments the network cost
EfficientNet [18]	Ability to capture relevant features and extract complex patterns in images	
MobileNet [19]	Reduced network size. Reduced number of parameters. Faster in performance and useful for mobile applications. Small, low-latency convolutional neural network.	Less accurate than other state-of-the-art networks
WideResNet [21]	Wide Residual Networks have 50 times fewer layers and are 2 times faster, Faster to train while they converge to the optimal solution much faster than standard ResNets	The number of parameters is the highest compared with the other methods.
HRNet [23]	It keeps the dimensions for the output. maintains high-resolution representations by connecting high-to-low-resolution convolutions in parallel, where there are repeated multiscale fusions across parallel convolutions.	The calculation allocation among high and low resolution branches are not optimized, and the low-resolution branches with strong semantic representation should be paid more attention

other part of a DL model is not random work. So, the implementation of a such model can be related to the target task as well as its complexity. Some proposed networks become famous networks used for different data analysis domains. These networks are used now for feature extraction or at the beginning of any DL model and its named backbones. A backbone is the recognized architecture or network used for feature extraction and its trained in many other tasks before and demonstrate its effectiveness. In this section, a detailed description of each backbone used in deep learning models is provided. A summarization of each Backbone is presented in Table 1. In addition, the advantages and disadvantages of each Backbone network in provided in Table 2

2.1. AlexNet

AlexNet is a CNN architecture developed by Krizhevsky et al. [5] in 2012 for image classification. The model consists of a set of convolutional and max-pooling layers ended by 3 fully connected layers. The proposed network contains 5 convolutions layers which makes it a simple network as presented in Fig. 1. In addition, it is trained using Rectified Linear Units (ReLUs) as activation. While the regularization function or Dropout is introduced to reduce overfitting in the fully-connected layers. Dropout proved its effectiveness for AlexNet and also for the deep learning model after. AlexNet has 60 million parameters and 650,000 neurons. This network is used for image classification on the ImageNet dataset. Also, it is used as a backbone for many object detection and segmentation models such as R-CNN [26] and HyperNet [27].

2.2. VGGs

The VGG family, which includes VGG-16 and VGG-19 [6], is one of the famous backbone used for computer vision and computer sciences tasks. The VGG architectures are proven their effectiveness in many tasks including image classification and object detection, and many other tasks. While it is widely used for many other architectures as backbone (for feature extraction) for many other recognized models like R-CNN [28], Faster R-CNN [29], and SSD [30].

For VGG-16 [6][1] in one of the fundamental deep learning backbones developed in 2014. VGG-16 contains 16 layers with 13 convolutional layers and 5 max-pooling layers and 3 fully connected layers. In addition, ReLU is used as activation. Compared to AlexNet architecture, VGG has 8 more layers. It has 138 million parameters.

For VGG-19[2] is a deeper version of VGG-16. It contains 3 more layers with 16 convolutional layers, 5 max-pooling layers, and 3 fully

[1] https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py.

[2] https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py.

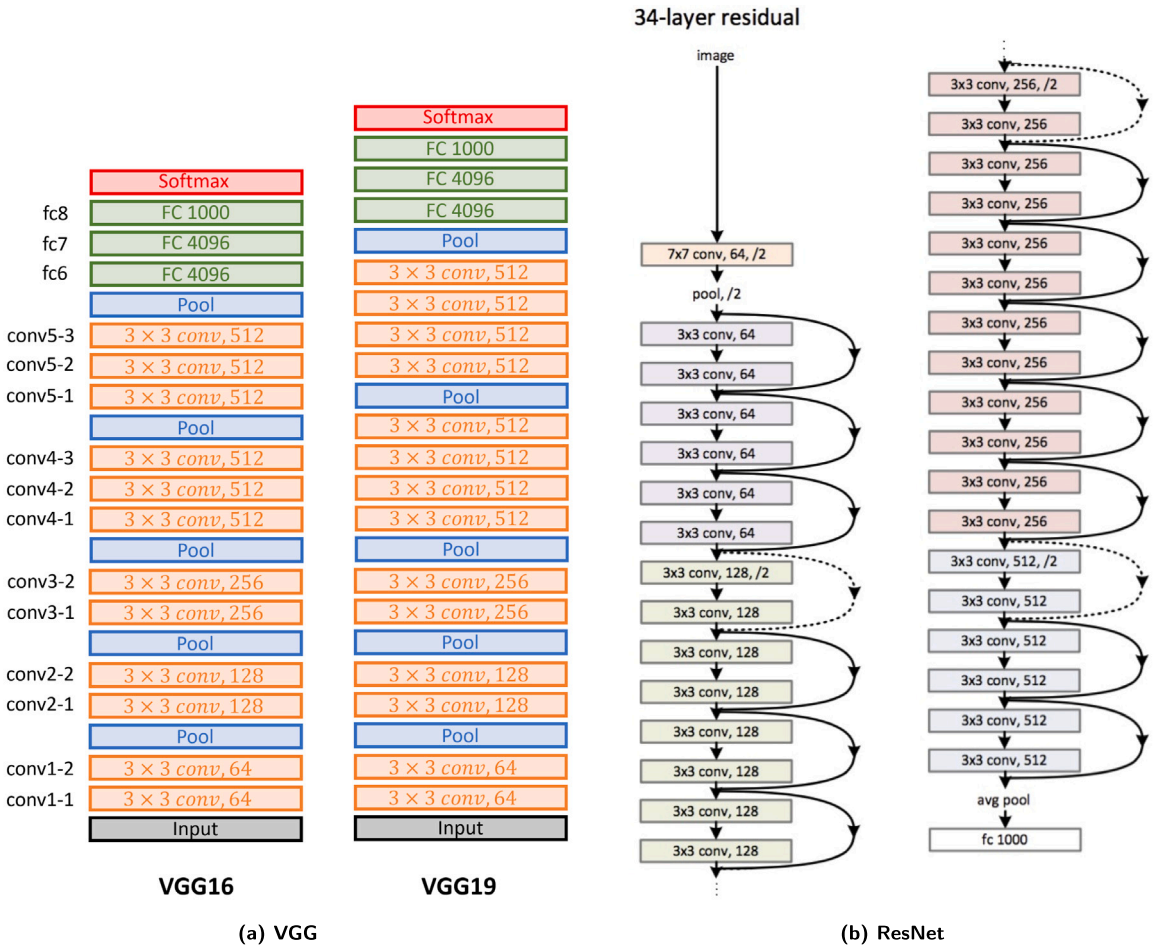


Fig. 2. VGG and ResNet architectures.

connected layers.VGG-19 architecture has 144 million parameters. The architectures of VGG-16 and VGG-19 are presented in Fig. 2(a).

2.3. ResNets

Unlike the previous CNN architecture, Residual Neural Network (ResNet) [7][3] is a CNN-based model while the residual networks are introduced. ResNet or Residual neural networks consists of some skip-connections or recurrent units between blocks of convolutional and pooling layers. Also, the block is followed by a batch normalization [31]. Like the VGG family, ResNet has many versions including ResNet-34 (Fig. 2(b)) and ResNet-50 with 26M parameters, ResNet-101 with 44M, and ResNet-152 which is deeper with 152 layers presented in Table 1. ResNet-50 and ResNet-101 are used widely for object detection and semantic segmentation. ResNet is also used for other deep learning architectures like Faster R-CNN [29] and R-FCN [32], etc.

2.4. Inception-v1 (GoogleNet)

Inception-V1 or GoogleNet[4] is one of the most used convolutional neural networks as backbone for many computer science applications [8]. It is developed based on blocks of inception. Each block is a set of convolution layers, while the filters used vary from 1×1, 3×3 to 5×5, which allows multi-scale learning. The size of each filter makes the variation of dimension between blocks. Also, GoogleNet uses global average pooling instead of the Max-pooling used in AlexNet and VGG. Inception-v1 block is illustrated in Fig. 3(a).

2.5. DenseNet

In traditional CNNs the number of layers L is the same as the number of connections. While the connection between layers can have an impact on the learning. For that, the authors in [16],[5] introduced a new convolutional neural network architecture named DenseNet with $L(L+1)/2$ connections. For each layer, the outputs (feature maps) of all previous layers are used as input to the next layer as presented in Fig. 3(b). The network could work with very small output channel depths (ie. 12 filters per layer), which reduces the number of parameters. The number of filters used in each convolutional layer is initialized, and after each layer, they used more filters than the previous layers with a constant of k or named "growth rate". This makes the number of parameters depend on k. Many versions of DenseNet have been proposed with variations in the number of layers, such as DenseNet-121, DenseNet-169, DenseNet-201, DenseNet-264. The network has an image input size of 224×224.

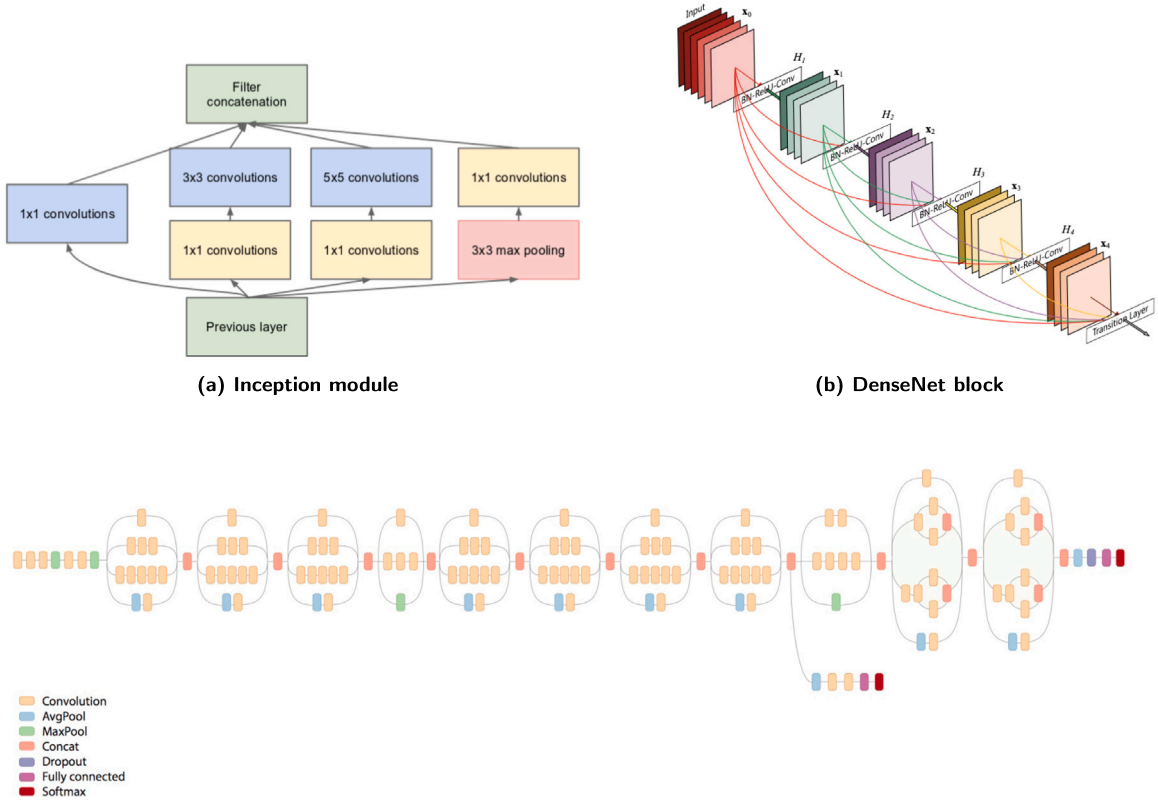
2.6. BN-Inception, Inception-v2, and Inception-v3

The computational cost is the most challenge in any deep learning model. The changes of the parameters between the consecutive layers

[3] https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py.

[4] https://github.com/Lornatang/GoogLeNet-PyTorch.

[5] https://github.com/liuzhuang13/DenseNet.



(a) Inception module

(b) DenseNet block

(c) Inception-v3

Fig. 3. GoogleNet, DenseNet, and Inception-v3 architectures.

and the initialization of it as well as the selection of required learning rates, during the training, make the training slow. The researchers attempted to solve this by normalizing input layers and applying this normalization in all network blocks. This operation is named Batch Normalization (BN) which minimizes the impact of parameters initialization and permits the use of higher learning rates [33]. BN is applied to the Inception network (BN-Inception[6]) and tested on ImageNet for image classification [33]. The obtained results are close to the Inception results on the same dataset with a lower computational time.

The development of deep convolutional networks for a variety of computer vision tasks is often prohibited by the challenges of computational cost caused by the size of these models. For that, the implementation of networks with low number parameters is a real need especially to be able to use them in machines of low performance like mobiles and Raspberry-Pis. In order to implement an efficient network with a low number of parameters, also using the architecture of Inception-v1 the authors in [9] developed Inception-v2[7] and Inception-v3 networks by replacing $n \times n$ convolutional kernels in Inception-v1 by 3×3 convolutional kernels as well as using 1×1 convolutional kernel with a proposed concatenation method. The architecture of Inception-v3 is illustrated in Fig. 3(c). This strategy used less than 25 million parameters. The two networks are trained and tested on image classification datasets like ImageNet and CIFAR-100.

2.7. Inception-ResNet-V2

Combining ResNet and inception architecture, Szegedy et al. developed Inception-ResNet-V2 [10][8] in 2016. Using residual connections (skip-connections between blocks of layers) Inception-ResNet-V2 is composed of 164 layers of 4 max-pooling and 160 convolutional layers, and about 55 million parameters. The Inception-ResNet network have led to better accuracy performance at shorter epochs. Inception-ResNet-V2 is used by many other architectures such as Faster R-CNN G-RMI [34], and Faster R-CNN with TDM [35].

2.8. DarkNet

In order to develop an efficient network with small size, the developer in [11] introduced Darknet-19[9] architecture based on some existing notions like inception and batch normalization [31] used in GoogleNet and ResNet as well as on the notions of network In network [11]. Darknet network is composed of a set of convolutional-max-pooling layers while the DarkNet-19 contains 19 convolutional. In order to reduce the number of parameters, a set of 1×1 convolutional kernels is used, while 3×3 convolutional kernels are not used much like in VGG of ResNet. DarkNet-19 is used for many object detection methods including YOLO-V2, YOLO-v3-v4 [36].

[6] https://github.com/Cadene/pretrained-models.pytorch/blob/master/pretrainedmodels/models/bninception.py.

[7] https://github.com/weiaicunzai/pytorch-cifar100/blob/master/models/inceptionv3.py.

[8] https://github.com/zhulf0804/Inceptionv4_and_Inception-ResNetv2.PyTorch.

[9] https://github.com/visionNoob/pytorch-darknet19.

2.9. Detnet

In the literature, we can find many object detection architectures that are based on pre-trained models to detect objects. This includes SSD, YOLO family, Faster R-CNN, etc. Some of the works are specifically designed for object detection. For the luck of backbones introduced for object detection, the authors in [17][10] proposed a deep-learning-based network. The same architecture is used also for image classification on ImageNet dataset and compared with the other famous network. For object detection, object localization as well as the recognition of it make the process different from the image classification process. For that, DetNet is introduced as an object detection backbone that consists of maintaining high spatial resolution using a dilation added to each part of the network, as illustrated in Fig. 4(a).

2.10. ShuffleNet

In order to reduce the computation cost as well as conserve the accuracy, ShuffleNet [14][11] proposed in [15] is a computation-efficient CNN architecture introduced for mobile devices that have limited computational power. ShuffleNet consists of point-wise group convolution and channel shuffle notions as presented in Fig. 4(b). The group convolution of depth-wise separable convolutions has been introduced in AlexNet for distributing the model over two GPUs, also used in ResNeXt to demonstrate its robustness. For that, point-wise group convolution is exploited by ShuffleNet to reduce the computation complexity of 1×1 convolutions. A Group of convolution layers can affect the accuracy of the network, for that, the authors of ShuffleNet used channel shuffle operation to share the information across feature channels. Besides image classification and object detection, ShuffleNet has been used as backbone for many other tasks. Also, ShuffleNet has two versions, while the second version is proposed in [15].

2.11. ResNeXt

ResNet architecture contains blocks of consecutive convolutional layers while each block is connected with the previous block output. The authors in [37][12] developed a new architecture based on ResNet architecture by replacing consecutive layers in each block with a set of branches of parallel layers like illustrated in Fig. 4(c). This model can be presented as the complex version of ResNet, which increases the model size with more parameters. But, in terms of learning, ResNeXt allows the model to learn from a set of features concatenated at the end of each block and using a variety of transformations with the same block. ResNeXt model has trained on image classification ImageNet dataset, as well as on object detection MS COCO dataset. The results are compared with ResNet results, and the obtained ResNeXt results outperform ResNet on COCO and ImageNet datasets.

2.12. SqueezeNet

To reduce the number of parameters, the authors in [13][13] developed a convolutional neural network named SqueezeNet consist of using 1×1 filter in almost layers instead of 3×3 filter since the number of parameters with 1×1 filters is 9x fewer. Also, the input channels are decreased to 3×3 filters, and delayed down-sampling of large activation maps leads to higher classification accuracy. The SqueezeNet layers are a set of consecutive fire modules as illustrated in Figure

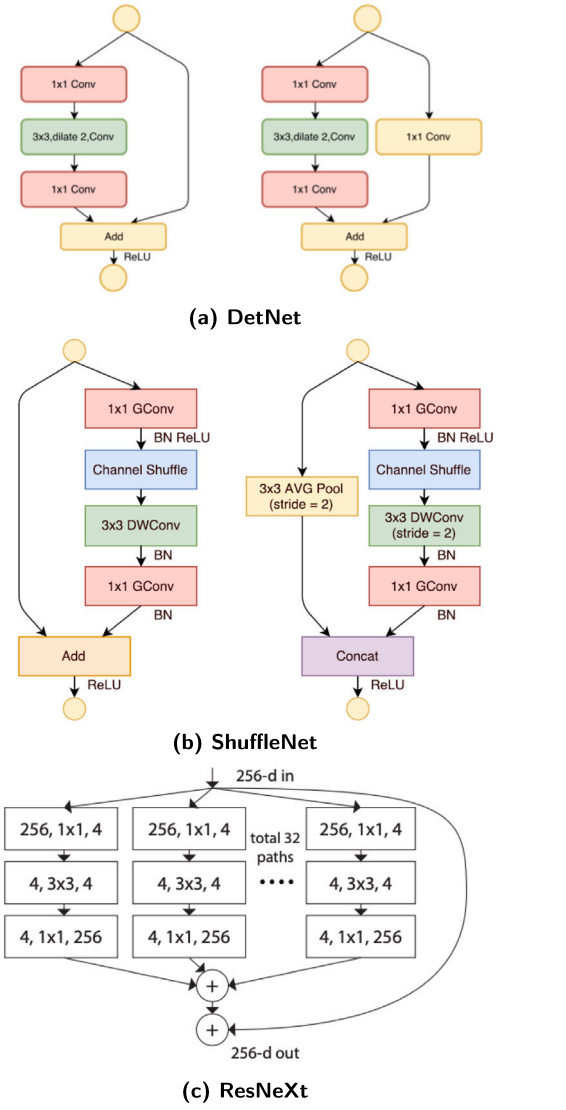


Fig. 4. DetNet, ShuffleNet, and ResNeXt block architectures.

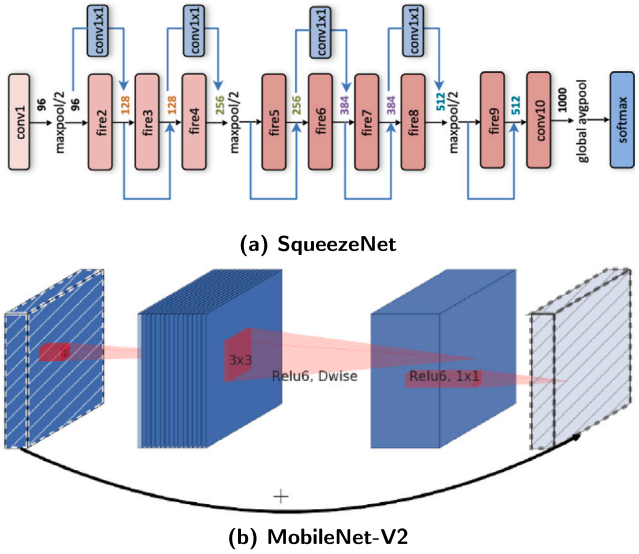


Fig. 5. SqueezeNet and MobileNet-V2 architectures.

[10] https://github.com/yumoxu/detnet.
[11] https://github.com/kuangliu/pytorch-cifar/blob/master/models/shufflenet.py.
[12] https://github.com/facebookresearch/ResNeXt.
[13] https://github.com/pytorch/vision/blob/master/torchvision/models/squeezenet.py.

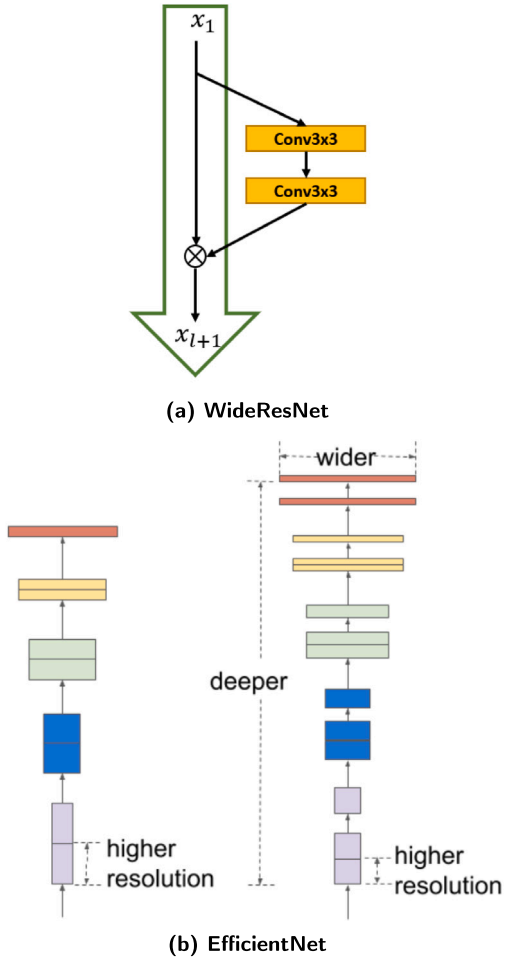


Fig. 6. WideResNet and EfficientNet architectures.

Fig. 5(a). A Fire module is comprised of: a squeeze convolution layer (which has only 1×1 filters), feeding into an expand layer that has a mix of 1×1 and 3×3. Like ResNet, SqueezeNet architecture has the same connection method between layers or fire modules. SqueezeNet contains 1.24M parameters which is the small backbone compared with the other networks.

2.13. MobileNet

In order to implement a deep learning model suitable to use for low machine performance like mobile devices, the authors in [19] developed a model named MobileNet.[14] Depth-wise separable convolutions are used to implement MobileNet architecture which can be represented as a lightweight model. Here two global hyper-parameters are introduced to make the developers choose the suitable sized model for their problem. MobileNet is trained and tested on ImageNet for image classification.

Another version of MobileNet which is MobileNet-v2 proposed in [20] for object detection purposes. Here, the authors introduced an invented residual block that allows a shortcut connection directly between the bottleneck layers as illustrated in Fig. 5(b). Also, depth-wise separable convolutions are used in this version to filter features as a source of non-linearity. the architecture is trained and tested for object detection and image classification (see Fig. 6).

[14] https://github.com/tensorflow/tfjs-models/tree/master/mobilenet.

2.14. WideResNet

Improving the accuracy of a residual network means increasing the number of layers which make the training very slow. The researcher in [21][15] attempted to solve this limitation by proposing a new architecture named WideResNet based on ResNet blocks, but instead of increasing the depth of the network they increase the width of the network and decrease the depth Fig. 6(a). This technique allows for to reduction of the number of layers as well as minimizing the number of parameters. The number of parameters in WideResNet depends on the number of residual layers (total number of convolutional layers) and on the widening factor k. WideResNet is trained and tested on CIFAR dataset, unlike the other backbones which are trained and tested on ImageNet. WideResNet has demonstrated outstanding performance in image classification, object detection, and semantic segmentation.

2.15. EfficientNet

EfficientNet [18][16] Networks which are a recent family of architectures have been shown to significantly outperform other networks in classification tasks while having fewer parameters and FLOPs. It employs compound scaling to uniformly scale the width, depth, and resolution of the network efficiently Fig. 6(b). EfficientNet parameters are 8.4x smaller and 6.1x faster on inference than the best existing networks. Many versions of EfficientNet start from B0 to B7. While EfficientNet-B0 is the baseline network, while Efficient-B1 to B7 are obtained by scaling up the baseline network. This can be easily replaced with any of the EfficientNet models based on the capacity of the resources that are available and the computational cost. EfficientNet-B0 contains 5.3 million parameters while the last version EfficientNet-B7 has 66M parameters.

2.16. SWideRNet

Scaling Wide Residual Network (SWideResNet) is a new network developed for image segmentation [22], obtained by incorporating the simple and effective Squeeze-and-Excitation (SE) and Switchable Atrous Convolution. Using the same principle of WideResNet, SWideResNet also adjusts the number of layers as well as the channel size (depth) of the network. In addition, SWideResNet used a SAC block instead of simple convolutional layers like WideResNet. Also, the Squeeze-and-Excitation block is added after each stage of the network. The number of parameters of SWideResNet is dependent on the scales of channels of the first two stages(w_1) of the network and of the remaining stages(w_2, l) as presented in Table 1.

2.17. Xception

Inspired by the Inception network, Xception [12] is another backbone used for feature extraction. Xception is a depth-wise separable convolution added to the Inception module with a maximally large number of towers. Where Inception V3 modules have been replaced with depth-wise separable convolutions. The proposed network is used in many computer science and vision tasks including object detection crowd counting, image segmentation, etc.

2.18. HRNet

[38] To maintain the high resolution of an image during the training process, the authors in [23] proposed a High-Resolution Net (HRNet)

[15] https://github.com/xternalz/WideResNet-pytorch.

[16] https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet.

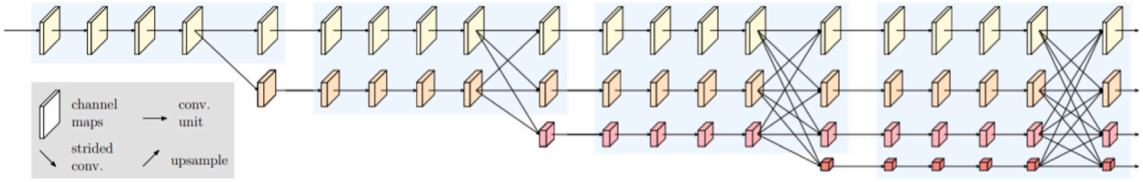


Fig. 7. HRNetV2 architecture.

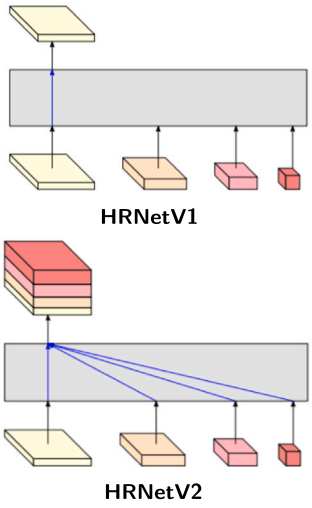


Fig. 8. HRNetV1 and HRNetV2 representation of the fusion strategy.

that consists of starting by high-resolution sub-network then moving to to-low resolution sub-network and so on, using multiple stages with the same scenario. The multi-resolution sub-networks are connected parallelly before the fusion operation. HRNet is implemented for human pose estimation then the second version of HRNet-v2 is proposed and used for semantic segmentation. For the HRNetV2 the difference is in fusion operation between multi-resolution sub-networks presented in Figs. 8 and 7. While for HRNet-v1, the representation from the high-resolution convolution stream is taken into account. For HRNetV2: Concatenate the (up-sampled) representations that are from all the resolutions.

3. Tasks related to backbones

3.1. Image classification

Image classification was and still is one of the hot topics in computer vision using deep-learning-based techniques. It was an evaluation subject of many deep-learning-based models. Due to the evolution of deep convolutional neural networks (CNNs), the performance of image classification methods has become more accurate and also faster. The evaluation of CNN-based image classification methods used the famous dataset in this task which is ImageNet. These networks have also been references for other topics due to the novelty of each one of them, as well as the evaluation of these networks on many datasets. From these backbones or reference networks, we can find all the cited networks including VGGs, ResNets, DenseNet, and others.

The efficiency of such a network is generally related to its performance accuracy. Here the image classification evaluation has been performed on ImageNet for all these proposed networks. But, we can find also another characteristic that can be considered for comparing the models, which is the computational complexity. For image classification, Floating Point Operations Per Second (FLOPs) is the performance measurement metric for computing the complexity in terms of the speed and latency of a model. The existing models that have the best accuracy, in general, have high FLOPs values. For example, the researcher implemented some models used by devices with limited computing power. From these models, we can find MobileNet implemented for smartphones as well as ShuffleNet and Xception. Another factor that can affect the speed of a model is parallelism. For example, under the same FLOPs a model can be faster than another while the degree of parallelism is higher.

Image classification task can be considered as simplest computer vision task in terms of information extracted from the images. This is compared with other computer vision tasks like panoptic segmentation, object tracking, Action recognition, etc. But, it can presented also as the test room of any deep-learning-based model. A comparison between the famous networks for image classification is presented in the discussion section.

3.2. Object detection

Object detection and recognition is one of the hot topics in computer vision. Many challenges can be found in detecting the objects in images including the scale of the objects, the similarity between some objects, and the overlapping between them. Object detection models that are optimal for detection need to have a higher input network size for the smaller objects. Multiple layers are needed to cover the increased size of the input network for a higher receptive field. To detect multiple objects there is a need for different sizes in a single image [13]. The following are different models used for object detection literature [39].

3.2.1. YOLOV3

YOLO-v3 is from the series of YOLO object detection models where YOLO-v3 is the third version. YOLO is a short form for “You Only Look Once”. YOLO detects multiple objects at a time. It is based on a Convolutional Neural Network and predicts classes as well as the localization of the objects. Applying a single Neural Network divides the image into grid cells and from this cell probabilities are generated. It predicts the bounding boxes using anchor boxes and outputs the best bounding box on the object. YOLO-v3 over here consists of 53 convolutional layers also called Darknet-53, for detection it has 53 more layers having a total of 106 layers. The detection happens at layers 82, 94, and 106 [40]. Images are often resized in YOLO according to input network size to improve detection at various resolutions. It predicts offsets to bounding boxes by normalizing the bounding box coordinates to the width and height of the image to eliminate gradients. The score represents the probability of predicting the object inside the bounding box. Predicted probability to the Intersection of the union that is a measure of the predicted bounding box to the ground truth bounding box [13,39]. This model was extensively used in many forms by [39] for training and testing and as a detector in the YOLOv4 architecture.

3.2.2. YOLO-v4

YOLO-v4 produces optimal speed and accuracy compared to YOLO-v3. It is a predictor with a backbone, neck, dense prediction, and sparse prediction. In the backbone several architectures can be used like Resnet, VGG16 and Darknet-53 [41]. The neck enhances feature discriminability by methods like Feature pyramid network (FPN), PAN

and RFB. The head handles the dense prediction using either RPN, YOLO, or SSD. For the backbone, Darknet-53 was tested to be the most superior model [41]. YOLO-v3 is commonly used as the head for YOLO-v4. To optimize training, data augmentation happens in the backbone. It is proven to be faster and more accurate than YOLO-v3 with MS COCO dataset. The advantage of YOLO-v4 is that it runs on a single GPU, enabling less computation. With its wide variety of features. It achieved 43.5% for the mAP metric on the COCO dataset with approximately 65 FPS.

3.2.3. YOLO-v4-tiny

YOLO-v4-tiny is a compressed version of YOLO-v4 with its network size decreased having a lesser number of convolutional layers with CSPDarknet backbone. The layers are reduced to three and anchor boxes for prediction are also reduced enabling faster detection.

3.2.4. Detectron

Detectron is Facebook AI Research's software system that implements state-of-the-art object detection algorithms, including Mask R-CNN. The input goes through a CNN backbone to extract features, which are used to predict region proposals. Regional features and image features are used to predict bounding boxes [42]. The scalability of this model and region proposal feature enables accurate detection.

3.2.5. YOLO-v5

YOLO-v5 is a PyTorch implementation of an improved version of YOLO-v3, published in May 2020 by Glenn Jocher of Ultralytics LLC5 on GitHub6 [43]. It is an improved version of YOLO-v3 implementation for PyTorch. It has a similar implementation to YOLO-v4, where it incorporates several techniques like data augmentation and changes to activation function with post-processing to the YOLO architecture. It combines images for training and uses self-adversarial training (SAT) claiming an accelerated inference [43,44].

3.3. Crowd counting

Crowd counting is the operation of estimating the number of people or objects in a surveillance scene. For people counting in the crowd, many works have been proposed for estimating the crowded mass. The proposed methods can be divided into many categories such as regression-based methods, density estimation-based methods, detection-based methods, and deep-learning-based methods. Comparing the accuracy of each one of these categories the CNN-based methods are the most effective methods.

The introduction of deep learning techniques makes computer vision tasks more effective and the Convolutional Neural Network (CNN) improves the performance accuracy of each task, especially those performed on large-scale datasets. On crowd counting, the use of deep learning techniques allows the estimation of crowd density more accurate compared with the traditional and sequential methods in terms of accuracy and the computational cost [45,46]. Also, the used backbones and the interconnection between the part of a network has an impact on the accuracy of a CNN model. Different backbones including VGG-16, VGG-19, ResNet-101, and others have been used in different crowd counting models, but the most used backbone for crowd counting is VGG-16. The use of these backbones can increase the consumption cost, especially on large-scale datasets. Also using VGG-16 for feature extraction, the authors in [47] proposed a crowd counting method named DENet composed of two-stage networks: detection network (DNet) and estimation network (ENet). Detection network counts the people in each region and the estimation network ENet work on the complex and crowded regions in the images. Using the same backbone, another method has been proposed named CANNet for estimating the crowd density map [48].

Also in [49] the authors based on the contextual and spatial information of the image as well as VGG-16 for crowd counting method implementation. The method named SCAR consists of a Spatial-wise attention module and a Channel-wise Attention module before combining the results of each module for the final estimation. In the same context, the authors in [50] proposed an adaptive dilated self-correction model (ADNet) which estimates the density. Using a multitask model that consists of two proposed networks — Density Attention Network (DANet) and Attention Scaling Network (ASNet) [51]. The authors estimate the attention map which is a segmentation of the crowd regions before using this result to estimate the density of the crowd. For the two methods, [50,50] VGG-16 is also used as backbone.

Using another version of VGG family which is VGG-19, the authors in [52] proposed a method based on density probabilities construction and Bayesian loss function (BL) to estimate the crowd density maps. In the same context, the authors in [53] proposed a crowd counting dataset as well as a crowd counting method based on a Special FCN model (SFCN). This method used ResNet-101 as a backbone, which is from a few crowd counting methods that used ResNet.

In order to reduce the number of parameters and the size of a network, the authors used the MobileNet-v2 backbone to reduce the FLOPs and model a Lightweight encoder–decoder crowd counting model [54].

3.4. Video summarization

Currently, real-time useful information extraction from video is a challenge for different computer vision applications, especially from large videos. The extracted information allows for reducing the time of searching as well as allowing for detecting and identifying some useful features for other tasks. The summarization of videos is one of the techniques applied to extract this information. Summarizing useful information from videos has been the main purpose of many recent studies [55]. It is considered as an important step to improve video surveillance systems in terms of reducing the searching time for a specific event as well as simplifying the analysis of a huge number of data. To do this, many aspects can be considered for a good summarization such as the type of scene analyzed (Private or public, indoor or outdoor, crowded or not). Also, the pre-processing can be used for enhancing the summarization process which is supposed to be with less space for storage in less computational time [56].

Using deep learning and deep reinforcement learning techniques many methods have been proposed. These methods used many backbones in their models for feature extraction. For example, in [57] the authors proposed a language-guided video summarization using a conditional CNN-based model while GoogleNet and ResNet backbones are used for feature extraction. The same backbone GoogleNet and ResNet-50 are used in [58] for multi-stage networks for video summarization. In the same context, and using Sparse Autoencoders network with Random Forest Classifier, the authors in [59] proposed a CNN-based model for key-frames selection. A set of backbones, including AlexNet, GoogleNet, VGG-16, and Inception-ResNet-v2, have been used then compared the impact of each one on the summarization results using VSUMM and OVP datasets. In [60], the ConvNet network is used for feature extraction of the proposed video summarization model. By scene classification for video summarization, the authors in [61] used many backbones for feature extraction including VGG-16, VGG-19, Inception-v3, and ResNet-50. To summarize the daily human behaviors, the authors in [62] proposed a deep learning method using ResNet-152 backbones. Using the same backbone, object-based video summarization with a DRL model has been proposed in [63] to summarize the video based on the detected object in it. The objects are tracked using an encoder–decoder network before selecting the target object to be summarized.

Using deep reinforcement learning, many methods have been proposed for video summarization using different backbones [64–70]. Highlighting or summarizing a video can be different from one user (person) to another. Existing deep learning methods attempted to summarize the videos using different models but with one point of view.

In other to overcome this limitation, as well as to detect different highlights according to different user preferences, deep reinforcement learning has been used in [64]. The algorithm used a reward function to detect each user's performance as a highlight candidate and also used ResNet-50 as backbone.

An effective video summarization method should analyze the semantic information in the videos. By using DRL for selecting the most distinguishable frames in a video, the authors in [65] cut the videos using the Action parsing technique. Here, AlexNet has been used for feature extraction. Using a weakly supervised hierarchical reinforcement learning architecture exploited GoogleNet for feature extraction, the authors in [66] select the representative video key-frames as summarization results which represents also a video shot after collecting them together. In addition to the content-based video summarization, some researchers proposed a query-conditioned video summarization to summarize the video based on a given query [67]. The proposed method named Mapping Network (MapNet) used two users query as inputs and the DRL-based framework provide two different summaries for the requested queries. For feature extraction, ResNet-152 is used. For summarizing a video into key-frames that represents the contextual meaning of the video, the authors in [68,69] proposed Deep Summarization Network(DSN) method based on a new deep reinforcement learning reward function that accounts for the representativeness and diversity of each frame of the video. Two methods used GoogleNet as a backbone. In the same context, using the semantics of the videos the authors in [70] proposed Summary Generation Sub-Network (SGSN) to select the key-frames from a given video using a DRL reward function and Inception-v3 as a backbone.

3.5. Action recognition

Action recognition aims to recognize various actions from a video sequence under different scenarios and distinct environmental conditions. This is a very challenging topic in computer vision due to (i) its wide range of applications, including video surveillance, tracking, health care, and human–computer interaction, and (ii) its corresponding issues that require powerful learning methods to achieve a high recognition accuracy. To that end, deep learning (DL) and deep reinforcement learning (DRL) has been investigated in several frameworks for different purposes, e.g. action predictability to perform early action recognition (EAR), video captioning, and trajectory forecasting.

For deep-learning-based methods, many architectures have been proposed using different backbones for feature extraction. For example, the authors in [71] proposed a temporal pyramid network for recognizing human action. The proposed method consists of classifying the same action that has variant tempos. This method is implemented using 3D ResNet. The same backbone has been used in [72] for an action recognition named Temporal Excitation and Aggregation (TEA). For the same purpose, BN-inception and Inception-v3 backbone have been used in [73], while the authors proposed an action recognition method for learning the powerful representations in the joint spatiotemporal feature space in a video. Using a multi-modal action recognition method on skeleton data, the authors in [74] used 3D ConvNet as a backbone. The same backbone has been investigated in [75] for the same purpose. For action detection and recognition, the authors in [76] proposed a Spatiotemporal attention model using ResNet-152 backbone for feature extraction.

Action recognition DRL-based methods are also exploited different backbones for feature extraction. For example in [77], the authors used ConvNet for extraction features exploited for the proposed action recognition DRL-based method. Accordingly, because of the absence of fine-grained supervision, the authors in [78] use a DRL-based technique for optimizing the evaluator, which is fostered by recognizability rewards and early rewards. In this context, EAR is achieved using predictability computed by the evaluator, where the classifier learns discriminative representations of subsequences. In this method, BN-Inception is used as a backbone.

In [79], due to the fact that (i) using the evolution of overall video frames for modeling actions cannot avoid the noise of the current action, and (ii) losing structural information of the human body reduces the feature capability for describing actions; the authors introduced a part-activated DRL approach to predict actions using VGG-16 backbone.

On the other hand, some works have focused on the use of visual attention to perform action recognition due to its benefit of reducing noise interference. This is possible by concentrating on the pertinent regions of the image and neglecting the irrelevant parts. Accordingly, in [80], a deep visual attention framework is proposed using DRL and GoogleNet for feature extraction, in which RNN with LSTM units are deployed as a learning agent, while DRL is utilized for learning the agent's decision policy. In the same manner, in [81], the irrelevant frames are ignored and only the most discriminative ones are preserved by using an attention-aware sampling scheme. Indeed, the mechanism of the key-frames extraction from videos is formulated as a Markov decision process. In this method, two backbone has been used such as BN-Inception and ConvNet. On the other side, in [82], starting from the fact that the attention mechanism might mimic the human drawing attention process before selecting the next location to focus(i.e. observe analyze, and jump instead of describing continuous features), the authors present a framework relying on designing a recurrent neural network-based agent with GoogleNet backbone, which selects attention regions using DRL at every timestamp.

3.6. Face recognition

The last decades have shown an increased interest in computer vision tasks including face recognition due to the technical development in video surveillance and monitoring. Face recognition is able to recognize uncooperative subjects in a nonintrusive manner compared with other biometrics like fingerprint, iris, and retina recognition. Thus, it is applicable in different sectors, including border control, airports, train stations, and indoors like in companies or offices. Many works have been reported with high performance. Some of these methods have been incorporated with surveillance cameras for person identification exploiting large-scale datasets. Also, analyzing the face is the main task for several biometric and non-biometric applications including the recognition of facial expression [83], face identification in images under low-resolution [84], and face identification and verification under pose variations, which represents the most subjects focusing on the analysis of the face.

The data structure can be considered to improve learning, which is not the case for almost all face recognition proposed methods. Deep face recognition methods use different loss functions to improve the classification results [85–91]. Softmax loss is one of the effective models for CNN-based face recognition. Other methods, combine different features of Softmax to enhance the recognition, including Sofmax+contrastive, Softmax Loss+Contrastive, L-Softmax Loss, Softmax+Center Loss, and Center Loss. Whereas other approaches used some loss function methods like Triplet Loss, Range loss [86], CosFace(LMCL) [85], and FairLoss [90]. Our method provides a new data structure on which the proposed method can be performed with any loss function while maintaining high accuracy.

Using DRL-based techniques face recognition was the subject of many studies. For example, to use DRL for recognizing the face, the authors in [92] proposed an Attention-aware-DRL-based approach to verify the face in a video. To find the face in a video, the proposed model is formulated as a Markov decision process. The image space and the feature space are used to train the proposed model, unlike the existing deep learning models that used one of them. Another research paper that exploited DRL techniques is proposed in [93]. The method introduced margin-aware RL techniques exploiting three loss functions

including angular softmax loss (SphereFace), large margin cosine loss (CosFace), and additive angular margin loss (ArcFace). LFW and YTF datasets are used to train the proposed Fair-Loss method. In the same context, Wang et al. [94] introduced Rl-race-balance-network (RL-RBN) for face recognition. Finding the optimal margins for non-Caucasians is a process formulated as a Markov decision process and exploits Q-learning to make agents learn the selection of the appropriate margin. The proposed method used ResNet34 as a backbone and RFW dataset for their learning process.

3.7. COVID-19 detection

Early work in COVID-19 detection is to extract images of patient lungs using ultrasound technology, a technique to identify and monitor patients affected by viruses. Therefore, the development of detection and recognition techniques is needed which are capable of automating the process without needing the help of skilled specialists. From these techniques, we can find computer vision-based techniques that help in detection using images and videos. The study [118], demonstrated how deep learning could be utilized to detect COVID-19 using images no matter what is the source, X-ray, Ultrasound, or CT scan. They built a CNN model based on a comparison of several known CNN models. Their approach aimed to minimize the noises so that deep learning uses the image features to detect the diseases. This study showed better results in ultrasound images compared to CT scans and x-ray images. They used VGG19 network as the backbone of their detection model. In the same context, Horry et al. [115] proposed a model to detect COVID-19. The system consists of four backbone such as VGG, Xception, Inception, and ResNet.

The accuracy of such a method is related to the annotated data by the experts and the deep learning models used which have the potential for COVID-19 detection. For that, techniques to detect COVID-19 using the concept of transfer learning were proposed with five variants of CNNs. VGG-19, MobileNet-v2, Inception, Xception, and ResNet-v2 are used in the first experiment and MobileNet-v2 in a second assay [119]. Minaee et al. [122] proposed a method named Deep-COVID based on the concept of deep transfer learning to detect COVID-19 from x-ray images. The authors used different backbones such as ResNet-18, ResNet50, SqueezeNet, and DenseNet-121. SqueezeNet technique gives the best performance in their experiment.

In another research paper, Moutounet et al. [116] developed a DL schema to differentiate between COVID-19 and other pneumonia from x-ray images. While VGG-16, VGG-19, Inception-ResNet-v22, InceptionV3, and Xception techniques are invested in their diagnosis. The best performance was obtained using VGG-16. Recently, the authors in [120] developed a CNN variants schema to detect COVID-19 from X-ray images. The experiments proved that the VGG-19 and DenseNet were the best networks to detect Covid-19. In the same context, and to identify COVID-19, Maguolo and Nanni [129] tested the AlexNet technique with 10-fold cross-validation for training and testing. While Chowdhury et al. [121] used transfer learning with image augmentation techniques to train and validate some pre-trained networks. Furthermore, it was proposed [131] a modified CNN to detect coronavirus from x-ray images. They combined Xception, and ResNet50-V2 techniques and applied 5-fold cross-validation techniques to classify three classes including COVID-19.

Loey et al. [123] presented a deep transfer learning model with Generative Adversarial Network (GAN) to detect COVID-19. Three backbones were tested such as AlexNet, GoogleNet, and ResNet-18. Furthermore, the authors in [125] concatenated Xception and ResNet50V2 and used 5-fold cross-validation techniques to detect the COVID-19 virus. In another study, Ucar et al. [130] used Bayes-SqueezeNet to develop a schema named COVIDiagnosis-Net to detect coronavirus using x-ray images. In another study, [132] the authors, presented a network called DarkCovidNet based on DarkNet backbone to detect the COVID-19 virus using x-ray images. In another study, Punn et al. [126] used ResNet, Inception-v3, Inception ResNet-v2, DenseNet-169, and NASNetLarge as a pre-trained CNN to detect COVID-19 virus from X-ray images. While, Narin et al. [127] used pre-trained models including Inception-v3, ResNet50, and Inception-ResNet-V2 to detect the COVID-19 virus with 5-fold cross-validation in the dataset partition. The authors of another research work [124] combined three pre-trained models such as ResNet-18, ResNet-50, and GoogleNet. While the authors in [117] used MobileNet, ResNet-50, VGG-16, and VGG-19 backbones for the COVID-19 detection method. In another study, [128] the authors proposed a deep learning technique using ResNet50 to detect COVID-19.

3.8. Panoptic segmentation

Panoptic segmentation is a new direction in image segmentation and also is the developed version of instance and semantic segmentation. While segmentation is made for things and stuff, unlike instance segmentation which segments the things only. The panoptic segmentation models generate segmentation masks by holding the information from the backbone to the final density map without any explicit connections [133]. Many Backbones are used for image segmentation in general and for panoptic segmentation also, but the most used is the ResNet family including ResNet-50 and ResNet-101.

Many methods have been proposed to segment the images using the panoptic presentation. From these works, we can find the method in [99] which is named fast panoptic segmentation network (FPSNet). It is a panoptic method while the instance segmentation and the merging heuristic part have been replaced with a CNN model called panoptic head. A feature map used to perform dense segmentation exploited the ResNet-50 backbone. Moreover, the authors in [100] employ a position-sensitive attention layer which adds less computational cost instead of the panoptic head. It utilizes a stand-alone method based on the use of Deep-lab as backbone. While in [101] a deep panoptic segmentation based on the bi-directional learning pipeline is utilized. Intrinsic interaction between semantic segmentation and instance segmentation is modeled using a bidirectional aggregation network called as BANet to perform a panoptic segmentation. The Backbone used here is ResNet-50. In the same context, some authors worked on video instead of images. In [102], video panoptic segmentation (VPSnet), which is a new video extension of panoptic segmentation is introduced, in which two types of video panoptic datasets have been used. The authors used as Backbone ResNet50 with FPN to extract feature maps for the rest of the network blocks. A holistic understanding of an image in the panoptic segmentation task can be achieved by modeling the correlation between the object and background. For this purpose, a bidirectional graph reasoning network for panoptic segmentation (BGRNet) is proposed in [103] Using ResNet-50 as well as the architecture in [104,105]. Using the same backbone in [106], the foreground things and background stuff have been dealt with in the attention-guided unified network (AUNet). Also using ResNet-50, an end-to-end occlusion-aware network (OANet) is introduced in [107] to perform a panoptic segmentation, which uses a single network to predict instance and semantic segmentation. Without unifying the instance and semantic segmentation to get the panoptic segmentation, Hwang et al. [109] exploited the blocks and pathways integration that allow unified feature maps that represent the final panoptic outcome. Finally, in [110], aiming at visualizing the hidden enemies in a scene, a panoptic segmentation method is proposed. ResNet-101 is another Backbone used for panoptic segmentation in [111] where the authors attempted to resolve overlaps using a scene overlap graph network (SOGnet). In the same context, and using RestNet-101, a unified method named DR1Mask has been proposed in [112] based on a shared feature map for both instance and semantic segmentation for panoptic segmentation.

While in [134], ResNeXt-101 for implementing the architecture of the DetectoRS method. DetectoRS is a panoptic segmentation method that consists of two levels: macro level and micro level. At the macro

Table 3
Various methods and backbone used for each task.

Task	Method	Backbone	Task	Method	Backbone
Image classification	AlexNet [5]	AlexNet	Object detection	MobileNet [19]	MobileNet
	GoogLeNet	GoogLeNet		ShuffleNet [14]	ShuffleNet
	VGG [6]	VGG		Xception [12]	Xception
	ResNet [7]	ResNet		MobileNet-v2 [19]	MobileNet-v2
	DenseNet [16]	DenseNet		DetNet [17]	DetNet
	DetNet [17]	DetNet		SSD [30]	VGG-16
	SqueezeNet [13]	SqueezeNet		ShuffleNet-v2 [15]	ShuffleNet-v2
	ResNeXt [37]	ResNeXt		WideResNet [21]	WideResNet
	Xception [12]	Xception		RetinaNet [95]	ResNet-101
	BN-Inception [33]	BN-Inception		RetinaNet [95]	ResNeXt-101
	Inception-v2 [9]	Inception-v2		Faster R-CNN G-RMI [34]	Inception-ResNet-v2
	Inception-v3 [9]	Inception-v3		Faster R-CNN TDM [35]	Inception-ResNet-v2
	Inception-ResNet-v1 [10]	Inception-ResNet		YOLOV2 [36]	DarkNet-19
	Inception-v4 [10]	Inception-v4		YOLO-V3 [96]	DarkNet-19
	Inception-ResNet-v2 [10]	Inception-ResNet-v2		YOLO-V4 [41]	CSPDarknet53
	WideResNet [21]	WideResNet		EfficientDet [97]	EfficientNet
	MobileNet [19]	MobileNet		DetectoRS [98]	ResNet-50
	ShuffleNet-v2 [15]	ShuffleNet-v2		DetectoRS [98]	ResNeXt-101
Panoptic segmentation	FPSNet [99]	ResNet-50	Crowd counting	CSRNet [45] (2018)	VGG-16
	Axial-DeepLab [100]	DeepLab		SPN [46] (2019)	VGG-16
	BANet [101]	ResNet-50		DENet [47]	VGG-16
	VSPNet [102]	ResNet-50		CANNet [48]	VGG-16
	BGRNet [103]	ResNet50		SCAR [49]	VGG-16
	SpatialFlow [104]	ResNet50		ADNet [50]	VGG-16
	Weber et al. [105]	ResNet50		ADSCNet [50]	VGG-16
	AUNet [106]	ResNet50		ASNet [51]	VGG-16
	OANet [107]	ResNet50		SCNet [108]	VGG-16
	SPINet [109]	ResNet50		BL [52]	VGG-19
	Son et al. [110]	ResNet-50		MobileCount [54]	MobileNet-V2
	SOGNet [111]	ResNet101		SFCN [53]	ResNet-101
	DR1Mask [112]	ResNet101	Video summarization	GoogleNet+Transformer [57]	GoogleNet
	DetectoRS [98]	ResNeXt-101		ResNet+Transformer [57]	ResNet
	EfficientPS [113]	EfficientNet		MCSF [58]	GoogleNet
	EffPS-b1bs4-RVC [114]	EfficientNet-B5			ResNet
Action recognition	Yang et al. [71]	ResNet-50		Nair et al. [59]	AlexNet
	TEA [72]	ResNet-50			GoogleNet
	Sudhakaran et al. [73]	BN-Inception			VGG-16
		Inception-v3			Inception-ResNet-v2
	MM-SADA [74]	3D ConvNet		Rafiq et al. [60]	VGG-16
	I3D [75]	3D ConvNet			VGG-19
	Li et al. [82]	ResNeXt-101			Inception-v3
		ResNet-18			ResNet-50
		ResNet-152		Zhang et al. [61]	ResNet-152
	Wu et al. [77]	ConvNet		Wang et al. [64]	ResNet50
	PEAR [78]	BN-Inception		Lei et al. [65]	AlextNet
	Chen et al. [79]	VGG-16		Chen et al. [66]	GoogleNet
	Li et al. [80]	GoogleNet		Zhang et al. [67]	ResNet-152
	Dong et al. [81]	BN-Inception		Zhang et al. [63]	ResNet-152
		ConvNet		DR-DSN [68]	GoogleNet
	Wang et al. [82]	GoogleNet		DR-DSN-s [68]	GoogleNet
COVID-19 detection	[115–117]	VGG16		Wang et al. [69]	GoogleNet
	[115,118–121]	VGG19		SGSN [70]	Inception-V3
	[116,121–124]	ResNet-18	Face recognition	CosFace(LMCL) [85]	ConvNet
	[122,125–128]	ResNet-50		Range loss [86]	VGG-19
	[115,116,120,126,127]	Inception-v3		NRA+CD [87]	ResNet-50
	[120,127]	Inception-ResNet-v2		RegularFace [88]	ResNet-20
	[115,116,119,120,125]	Xception		Range loss [86]	VGG-19
	[123,129]	AlexNet		ArcFace [89]	ResNet-100
	[123,124,126]	GoogleNet		FairLoss [90]	ResNet-50
	[120–122,126]	DenseNet		Attention-aware-DRL [92]	ResNet
	[117,119–121]	MobileNet-v2		Margin-aware-DRL [93]	ResNet-50
	[121,122,130]	SqueezeNet		Skewness-aware-DRL [94]	ResNet-34

level, a recursive feature pyramid (RFP) is used to incorporate extra feedback connections from FPNs into the bottom-up backbone layers. While at the micro-level, a switchable atrous convolution (SAC) is exploited to convolve the features with different atrous rates and gather the results using switch functions. Using the EfficientNet backbone, a variant of the Mask R-CNN as well as the KITTI panoptic dataset that has panoptic ground truth annotations are proposed in [113]. The method is named Efficient panoptic segmentation (EfficientPS). Also, a unified network named EffPS-b1bs4-RVC, which is a lightweight version of EfficientPS architecture is introduced in [114].

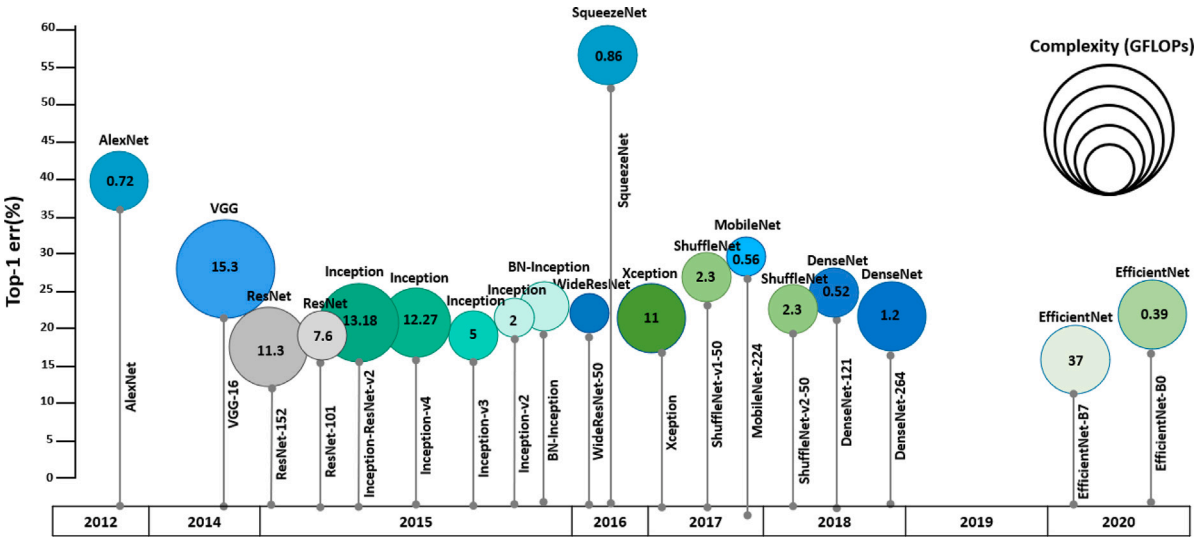


Fig. 9. Timeline and performance accuracies of the proposed networks on ImageNet.

3.9. Used backbones for each task

In order to present the backbone used in each task, we attempted in this paper to collect the proposed methods and their feature extraction networks exploited. Table 3 presents the proposed methods for each task as well as the backbones used for each method. From the tables, we can find that in some tasks some specific backbones are used widely for a task while are not exploited for others. For example, crowd counting methods used VGG-16 network, while just some of them used VGG-19, MobileNet-v2, and ResNet-101. For Panoptic segmentation, ResNet-50 is the backbone that is widely used for segmenting the object in a panoptic way. GoogleNet and ResNeXt are exploited for video summarization. For the others tasks, all the backbones are invested.

Fig. 10 illustrate the computer vision tasks and the used backbones for each task. The used backbones have been presented by taking into account the number of papers used for each backbone in a task. For example, for crowd counting, VGG is the most used backbone for counting the people or the object in the monitored scene. The same observation for face recognition, while the almost proposed method exploits ResNet for extracting features. ResNet is also the most used backbone for action recognition with GoogleNet. The same Backbone ResNet is the most used for panoptic segmentation and video summarization tasks. For object detection and COVID-19 detection tasks, all the backbones are used equally unlike the other tasks we can find a specific backbone widely used.

4. Critical discussion

In order to present the impact of backbones on each task, some performance results are presented for image classification, object detection, action recognition, face recognition, panoptic segmentation, and video summarization. A comparison between the obtained results is performed using the evaluation metrics used for each task as well as the datasets used for training and evaluation.

4.1. Image classification evaluation

Image classification is the most frequent task used for evaluating the new proposed CNN-based architecture. The most used dataset for this is ImageNet. The CNN-based architectures have been evaluated using generally three metrics including Top-1 error, Top-5 error, and the computational complexity of the model using the number of Flops. These metrics are used for comparing the effectiveness of the proposed models presented in Table 4. From the table, we can see that some of the models evaluated just by Top-1 error and others just using Top-5 error. Also, the models that have high complexity reached good results. For example, EfficientNet-B7 [18] have 37G for Flops which is generally high but reached minimum values of Top-1 and Top-5 error rates compared with Inception-v3 [9] that has the second best results with a difference of 3% in terms of Top-1 error rate and 1,2% for Top-5 error rate. The same observation for the other models like Inception-ResNet-v2 [10], ResNet-152 [7], and Inception-v4 [10]. For the low-complexity methods, we can find that the Top-1 and Top-5 error rates are higher than the others. For example, MobileNet has low complexity Folps, but the Top-1 error rate is high with a value of 29.4, as well as AlexNet. Some of the methods do not give the Flops number for their evaluation on ImageNet like WideResNet-50 [21], Inception-v2 [9], and Inception-ResNet-v1 [10]. From Table 4 and Fig. 9 we can

Table 4
Image classification performance results on ImageNet. The **bold** and underline fonts respectively represent the **first** and second place.

Backbone	Complexity (GFLOPs)	Top-1 err (%)	Top-5 err (%)
AlexNet [5]	0.72	42.8	19.7
GoogLeNet	1.5	–	7.89
VGG-16 [6]	15.3	28.5	9.9
ResNet-101 [7]	7.6	19.87	4.60
ResNet-152 [7]	11.3	19.38	4.49
DenseNet-121 [16]	0.525	25.02	7.71
DenseNet-264 [16]	1.125	22.15	6.12
DetNet-50 [17]	3.8	24.1	–
DetNet-59 [17]	4.8	23.5	–
DetNet-101 [17]	7.6	23.0	–
SqueezeNet [13]	0.861	42.5	19.7
ResNeXt-50 [37]	4.1	22.2	–
ResNeXt-101 (32 × 4d) [37]	7.8	21.2	5.6
ResNeXt-101 (64 × 4d) [37]	15.6	20.4	–
Xception [12]	11	21.0	5.5
BN-Inception [33]	2	22.0	5.8
Inception-v2 [9]	–	21.2	5.6
Inception-v3 [9]	5.72	18.7	4.2
Inception-ResNet-v1 [10]	–	21.3	5.5
Inception-v4 [10]	12.27	20.0	5.0
Inception-ResNet-v2 [10]	13.1	19.9	4.9
WideResNet-50 [21]	–	21.9	5.79
MobileNet-224 [19]	0.569	29.4	–
ShuffleNet-v1-50 [14]	2.3	25.2	–
ShuffleNet-v2-50 [15]	2.3	22.8	–
EfficientNet-B0 [18]	0.39	22.9	6.7
EfficientNet-B7 [18]	37	**15.7**	**3.0**

Fig. 10. Used backbones for each task. In some tasks, some backbones are used more than others and are illustrated with a large scale in the figure.

Table 5
Object detection performance results on MS COCO. The **bold** and underline fonts respectively represent the **first** and second place.

Type	Method	Backbone	mAP
Backbone	MobileNet-224 [19]	MobileNet	19.8
	ShuffleNet [14] 2 × (g = 3)	ShuffleNet	25.0
	Xception [12]	Xception	32.9
	MobileNet-v2 [20]	MobileNet-v2	30.6
	DetNet-50 [17]	DetNet	37.9
	DetNet-59 [17]	DetNet	**40.2**
	DetNet-101 [17]	DetNet	39.8
	ShuffleNet-v2 [15]	ShuffleNet-v2	34.2
	WideResNet-34 (tset) [21]	WideResNet	35.2
Backbone+ Network	RetinaNet [95]	ResNet-101-FPN	39.1
	RetinaNet [95]	ResNeXt-101-FPN	40.8
	Faster R-CNN G-RMI [34]	Inception-ResNet-v2	34.7
	Faster R-CNN TDM [35]	Inception-ResNet-v2-TDM	36.8
	YOLOV2 [36]	DarkNet-19	20.6
	YOLO-V3-spp [96]	DarkNet-19	**60.6**
	YOLO-V4-P7 [41]	CSPDarknet53	55.7
	EfficientDet-D7 [97]	EfficientNet-B7	55.1
	DetectoRS [98]	ResNet-50	53.0
	DetectoRS [98]	ResNeXt-101-64 × 4d	55.7

find that the number of parameters and complexity of a model have an impact on its accuracy. Also, the development of the different versions of the same model increased the accuracy but with more complexity like in Inception-v3 [9] and Inception-v4 [10].

4.2. Object detection evaluation

The proposed deep learning methods for object detection are evaluated using the mean Average Precision (mAP) metric on different datasets. The proposed DL-based methods used MS COCO for performance evaluation. This dataset has been taken into consideration due to its wide usage for all the proposed object detection methods.

The evaluation of proposed methods on the MS-COCO dataset was performed on the validation set using the mAP metric. For that, we attempted to present the obtained results for each method in Table 5. In this table, we attempted to separate the methods into backbone methods and the backbones+Network methods. We mean by backbone methods, those pre-trained models that used object detection for proving the performance of the proposed networks then these methods are used also as backbone for other object detection methods. Backbones+Network denotes the proposed methods that used famous backbones just for the feature extraction step and they implement other blocks in their networks for detecting the objects. For the pre-trained models that evaluated on MS COCO for detecting the objects, we can see that DetNet-59 [17] achieved the best results with an mAP of 40.2 followed by DetNet-101 [17] by a difference of 0.4. While MobileNet-v1 [19] is the lowest reached results of an mAP of 19.8. The others method including ShuffleNet [14], Xception [12], MobileNet-v2 [19], and ShuffleNet-v2 [15] reached close results.

For the backbone+Network methods, we can find from the table that YOLO-V3-spp [96], YOLO-V4-P7 [41], and DetectoRS [98] reached the first best results, while YOLO family used DarkNet as backbone and DetectoRS used ResNet and ResNeXt as backbone. The results using EfficientDet-D7 [97] are also good with a difference of 0.6 compared with DetectoRS and YOLO-v3.

According to mAP values achieved, we can find that the performance of these methods using DL still needs improvement even with the number of works proposed. This is due to the complexity of scenes that can contain many objects with different situations like small scales, different positions, occlusion between objects, etc.

Table 6
Face verification accuracy (%) on LFW and YTF datasets trained on WebFace dataset. The **bold** and underline fonts respectively represent the **first** and second place.

Type	Method	Backbone	LFW	YTF
DL	CosFace(LMCL) [85] (2018)	ConvNet	99.33	96.1
	Range loss [86]	VGG-19	99.52	93.7
	NRA+CD [87]	ResNet-50	99.53	96.04
	RegularFace [88]	ResNet-20	99.33	94.4
	ArcFace [89]	ResNet-100	**99.83**	**98.02**
	FairLoss [90]	ResNet-50	99.57	96.2
DRL	Attention-aware-DRL [92]	ResNet	–	**96.52**
	Margin-aware-DRL [93]	ResNet-50	**99.57**	96.2

Table 7
Performance of action recognition methods on HMDB-51 and UCF-101 datasets. The **bold** and underline fonts respectively represent the **first** and second place.

Type	Method	Backbone	HMDB-51	UCF-101
DL	TEA [72]	ResNet-50	73.3	**96.9**
	I3D [75]	3D ConvNet	66.4	93.4
	Li et al. [76]	ResNeXt-101	**75.0**	95.0
DRL	PEAR [78]	BN-Inception	–	84.99
	PA-DRL [79]	VGG-16	–	87.7
	Li et al. [80]	GoogleNet	66.8	93.2
	TSN-AS [81]	BN-Inception	**71.2**	**94.6**
	Wang et al. [82]	GoogleNet	60.6	–

4.3. Face recognition evaluation

In order to evaluate the proposed face recognition methods, many datasets are used. The most used datasets include LFW and YTF which are common face recognition datasets. For that, we collected some of DL and DRL-based methods tested on the same datasets by mentioning the backbones used as feature extraction models. Table 6 represents some obtained results by face recognition approaches on LFW and YTF datasets. From the table, we can observe that the performance accuracies are close for DL and DRL-based approaches. For example, the DL-based methods ArcFace [89] and FairLoss [90] come in first place with an accuracy of 99.83 and 99.75 on LFW and 98.02 and 96.2 for YTF respectively. The others methods are also in the same range, while we can find the difference between the accuracies not exceeding 0.5% for LFW and 5% for YTF. The same observation for DRL-based methods reached more than 99% on LFW and 96% for YTF.

Another observation for DL and DRL-based approaches, the same backbone family is used for almost all methods while the depth of the networks varies. For these backbones, we can find ResNet-20, ResNet-50, and ResNet-100. While VGG is used by Range loss [86].

4.4. Action recognition evaluation

Action recognition methods based on Deep learning (DL) and Deep reinforcement learning (DRL) are presented in 7. These methods are collected based on the backbones and datasets used. For example, we choose for comparison the dataset used by more than two DL and DRL-based methods, because some datasets are used in just one paper. The results presented in this comparison are on UCF-101 and HMDB51 datasets. In order to evaluate their methods the performance accuracy is invested by the proposed architectures on UCF-101 and HMDB51.

The obtained results on each dataset are presented in Table 7. On HMDB-51 dataset, the DL-based method in [76] that exploited ResNeXt-101 as backbone reached the highest accuracy, while [72] is the second best result by an accuracy of 73.3. The same method [72] achieved the best performance accuracy on UCF-101, while [76] results comes in the second place by a difference of 1.9 point. For [76] they used ResNet-50 as backbone. For I3D [76] method, ConvNet is used as backbone while the obtained results is less than [76] by 8.6 point for on HMDB-51 dataset and 2.4 points for UCF-101 dataset.

For DRL-based methods, the performance evaluations are performed on the same two datasets and presented in 7. For example, TSN-As [81] is the highest accuracy on HMDB-51 followed by [80] with a difference of 4.4 points using BN-Inception and GoogleNet respectively. The same observation on UCF-101, while the same methods [80,81] achieved the highest accuracy values of 94.6 and 93.2 respectively.

From the table, We can observe that the DL and DRL-based action recognition methods tested on different datasets using different backbones have a difference in terms of performance accuracies reached. While the DL-based method achieved better results than the DRL-based method. Also for DL-based methods, we found that the method that used ResNet and ResNeXt reached higher accuracies than the other used GoogleNet or VGG. which means the use of specific backbones for a specific task can make the difference in terms of performance.

4.5. Panoptic segmentation evaluation

Cityscapes and COCO datasets are the most commonly preferred datasets for experimenting with the efficiency of panoptic segmentation solutions. A detailed report on the methods that used these datasets with the evaluation metrics is given in Table 8. In addition, the obtained results have been presented considering the backbones used. Though it is common to use the validation set for reporting the results. Here we presented just the evaluation on the validation set. All the models are representative, and the results listed in Table 8 have been published in the reference documents.

On Cityscapes, we can observe that different methods, such as [110, 112] have evaluated their results using the three metrics, i.e. PQ, SQ, and RQ. While some approaches have also been evaluated using these metrics on things (PQ^{th}, SQ^{th}, and RQ^{th}) and stuff (PQ^{st}, SQ^{st}, and RQ^{st}), e.g. EfficientPS [113]. Moreover, from the results in Table 8, Axial-DeepLab [100] reaches the highest PQ values, with an improvement of 2.6% than the second-best result obtained by EfficientPS Multi-scale [113]. Regarding the SQ metric, EFFIcientPS achieves the best result using single-scale and multi-scale, by a difference of 0.7%. The same thing Uses SQ metric, EfficientPS provides the best accuracy results. The difference between EfficientPS and other methods is that it utilizes a pre-trained model on the Vistas dataset, whereas the schemes do not use any pre-training. In addition, EfficientPS uses EfficientNet as backbone while the most of discussed techniques have exploited RestNet-50 except Axial-Deeplab, which uses DeepLab as a backbone.

On the COCO validation set, the evaluation results are slightly different from the obtained results on the Cityscapes dataset although there are some frameworks that have reached the highest performance, such as DR1Mask [112] for PQ, PQ^{st}, and SQ, Axial-DeepLab [100] for PQ^{st}, BANet [101] for SQ^{st}, RQ^{st}, and SQ^{th}, OANet [107] for SQ^{th}. For example, using DR1Mask, the performance rate for using PQ and PQ^{th} metrics has reached 46.1% and 53.1%, respectively. The difference between the methods that reach the highest results and those in the second and third places, is around 1%–4%, which demonstrates the effectiveness of these panoptic segmentation schemes.

4.6. Crowd counting evaluation

The obtained results using MAE and MSE metrics are presented in Table 9. We can observe that many methods succeed to estimate the number of people in the crowd with promising results especially for the ShanTech_Part_B dataset due to simple crowd density in this dataset and all the images contains the same depth of the crowd and the same distribution of the people in the scenes. We can find also that the ShanTech_Part_A comes in the second place in terms of MAE reached due to the same reasons as ShanTech_Part_B but here the images are more crowded that the images in ShanTech_Part_B. For the other datasets such as UCF_QNRF and UCF_CC_50, the images are more crowded and can reach 4000 people per image with make the

Table 8
Performance comparison of existing panoptic segmentation schemes on the val set under Cityscapes and COCO datasets. The **bold** and underline fonts respectively represent the **first** and second place.

Dataset	Method/Backbone	PQ			SQ			RQ		
		PQ	PQ^{st}	PQ^{th}	SQ	SQ^{st}	SQ^{th}	RQ	RQ^{st}	RQ^{th}
Cityscapes	FPSNet [99]/ResNet-50	55.1	60.1	48.3	–	–	–	–	–	–
	Axial-DeepLab [100]/DeepLab	**67.7**	–	–	–	–	–	–	–	–
	EfficientPS Single-scale [113]/EfficientNet	63.9	66.2	60.7	81.5	81.8	81.2	77.1	79.2	74.1
	EfficientPS Multi-scale [113]/EfficientNet	65.1	**67.7**	**61.5**	**82.2**	**82.8**	**81.4**	**79.0**	**81.7**	**75.4**
	VPSNet [102]/ResNet-50	62.2	65.3	58.0	–	–	–	–	–	–
	SpatialFlow [104]/ResNet-50	58.6	61.4	54.9	–	–	–	–	–	–
	SPINet [109]/ResNet-50	63.0	67.3	57.0	–	–	–	–	–	–
	Son et al. [110]/ResNet-50	58.0	–	–	79.4	–	–	71.4	–	–
COCO	Axial-DeepLab [100]/DeepLab	43.9	**36.8**	48.6	–	–	–	–	–	–
	BANet [101]/ResNet-50	43.0	31.8	50.5	79.0	**75.9**	81.1	52.8	**39.4**	**61.5**
	BGRNet [103]/ResNet-50	43.2	33.4	49.8	–	–	–	–	–	–
	SpatialFlow [104]/ResNet-50	40.9	31.9	46.8	–	–	–	–	–	–
	Weber et al. [105]/ResNet-50	32.4	28.6	34.8	–	–	–	–	–	–
	SOGNet [111]/ResNet-102	43.7	33.2	50.6	–	–	–	–	–	–
	OANet [107]/ResNet-50	40.7	26.6	50.0	78.2	72.5	**82.0**	49.6	34.5	59.7
	SPINet [109]/ResNet-50	42.2	31.4	49.3	–	–	–	–	–	–
	DR1Mask [112]/ResNet-101	**46.1**	35.5	**53.1**	**81.5**	–	–	**55.3**	–	–

Table 9
The performance of each method on the existing crowd counting dataset. The **bold** and underline fonts respectively represent the **first** and second place .

Method	Backbone	ShanTech_A		ShanTech_B		UCF_QNRF		UCF_CC_50	
		MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
CSRNet [45] (2018)	VGG-16	68.2	115.0	10.6	16.0	–	–	266.1	397.5
SPN [46] (2019)	VGG-16	61.7	99.5	9.4	14.4	–	–	259.2	335.9
DENet [47] (2020)	VGG-16	65.5	101.2	9.6	15.4	–	–	241.9	345.4
CANNet [48](2019)	VGG-16	62.3	100.0	7.8	12.2	107.0	183.0	212.2	243.7
SCAR [49] (2019)	VGG-16	66.3	114.1	9.5	15.2	–	–	259.0	374.0
ADNet [50] (2020)	VGG-16	61.3	103.9	7.6	12.1	90.1	147.1	245.4	327.3
ADSCNet [50] (2020)	VGG-16	**55.4**	97.7	**6.4**	**11.3**	**71.3**	**132.5**	198.4	267.3
ASNet [51] (2020)	VGG-16	57.7	**90.1**	–	–	91.5	159.7	**174.8**	251.6
SCNet [108] (2021)	VGG-16	58.5	99.1	8.5	13.4	93.9	150.8	197.0	**231.6**
BL [52] (2019)	VGG-19	62.8	101.8	7.7	12.7	88.7	154.8	229.3	308.2
MobileCount [54] (2020)	MobileNet-V2	84.8	135.1	8.6	13.8	127.7	216.5	284.5	421.2
SFCN [53] (2019)	ResNet-101	64.8	107.5	7.6	13.0	102.0	171.4	214.2	318.2

estimation of the density maps more complex. Also, the scale and shape variations in these datasets affect the performance of each method.

For the obtained MAE and MSE of each method, the results in the table show that each method reaches good results in a dataset better than the others. And this comes from the treatment used for each method as well as the backbone exploited for feature extraction. For example, some methods are working on the scale variation while others used segmentation of the crowd region before estimating the crowd density. For the ADSCNet method, we can see that it outperforms the others method in three datasets including ShanTech_Part_A, ShanTech_Part_B, and UCF_QNRF with an MAE of 55.4 on ShanTech_Part_A and less by 2.3 points than the ASNet method which come in the second place. While we can find that the SPN, SCAR, and CANNet methods reached close results of MAE values. On the UCF_CC_50 dataset, SCNet achieved fewer MSE results better than ASNet with 20 points. For these results, we can conclude that the methods used VGG-16 are the most effective methods compared with the method used MobileNet and ResNet-101 as backbones. Also, some proposed architectures can be better in some cases while it is not in others like ADSCNet on hanTech_Part_A, ShanTech_Part_B, and UCF_QNRF datasets and AsNet and SCNet on UCF_CC_50 dataset.

4.7. Video summarization evaluation

To show the performance of each video summarization method using DL and DRL, The obtained results using state-of-the-art methods on SumMe and TVSum datasets are presented in Table 10. From the table, we can find that GoogleNet and ResNet are the most used feature extraction backbones For deep learning and deep reinforcement learning-based approaches. For DL-based approaches, the method

Table 10
Comparison of the performance of video summarization methods. The **bold** and underline fonts respectively represent the **first** and second place.

Type	Method	Backbone	SumMe	TVSum
DL	GoogleNet+Transf [57]	GoogleNet	51.6	64.2
	ResNet+Transf [57]	ResNet	**52.8**	**65.0**
	MCSF [58]	GoogleNet	48.1	56.4
	Zhang et al. [63]	ResNet-152	37.7	51.1
DRL	Lei et al. [65]	AlextNet	41.2	51.3
	Chen et al. [66]	GoogleNet	**43.6**	58.4
	DR-DSN [68]	GoogleNet	41.4	57.6
	DR-DSN-s [68]	GoogleNet	42.1	58.1
	Wang et al. [69]	GoogleNet	43.4	**58.5**
	SGSN [70]	Inception-V3	41.5	55.7

in [57] achieved the best results on SumMe and TVSum datasets. While ResNet+Transformer [57] comes in the first place by values of 52.8% and 65.0% on SumME and TVSum, and GoogleNet+Transformer [57] reached the second best results by a difference of 1.2% for the two datasets. For the other approaches like MCSF [58,63] the obtained results achieved close results with a difference of 9.6% for SumMe and 5.3% for TVSum. Generally, all the methods are in the same range in terms of accuracy reached on the two datasets.

For DRL-based methods including [65,66], DR-DSN [68], DR-DSN-s [68,69], and SGSN [70], the obtained results are close while we can find the difference between the Best and worst result on SumMe dataset not exceed 2.4% and 7.1% for TVSum. On SumMe dataset, the method in [66] reached 43.6% which is the best result, followed by [69] with a value of 43.4%. We can observe that the two methods used GoogleNet as backbone. The same methods reached the best results on TVSum but

Table 11
COVID-19 detection techniques using different methods that are based on cited backbones. The **bold** and underline fonts respectively represent the **first** and second place.

Authors	Model	Classes	Accuracy	Sensitivity	Specificity
Chowdhury et al.[121]	MobileNet-v2	3	96.22	96.22	97.80
	Inception-v3	3	96.2	96.4	97.5
	ResNet-101	3	96.22	96.2	97.8
	DenseNet-201	3	97.9	97.9	98.8
Ucar et al. [130]	SqueezeNet	3	98.3	98.3	99.1
Ozturk et al. [132]	DarkNet	3	98.1	95.1	95.3
Punn et al. [126]	Inception-v2	3	88.0	79.0	89.0
	Inception-ResNet-v2	3	92.0	92.0	89.0
	DenseNet-169	3	95.0	96.0	95.0
Narin et al. [127]	Inception-v3	3	**99.5**	**100**	**100**
	ResNet-50	3	91.7	57.0	91.3
	ResNet-152	3	97.3	93.2	99.3
	Inception-ResNet-v2	3	96.3	78.0	96.8
Ozcan et al. [124]	ResNet-50	4	97.6	97.2	97.9

this time [69] reached the first best results and [66] comes in second place with 58.5%.

From the presented results we can find that the performance of these methods using DL and DRL is challenging according to the accuracy rate achieved. In addition, the performance of these methods on TVSum is improved than SumMe dataset.

4.8. COVID-19 detection evaluation

To demonstrate the performance of proposed COVID-19 detection methods, many metrics have been used including accuracy, specificity, and sensitivity. Some methods used transfer learning using many pre-trained models. We collect a set of these methods to compare the impact of each model on COVID-19 detection dataset. Table 11 represents a set of methods providing the obtained results with the three metrics. From the table, we can find that all the methods succeed in detection COVID-19 from X-ray images with convincing accuracies. While [127] using Inception-v3 reached the highest results of 100% for sensitivity and specificity metrics and 99.5% for model accuracy. Using SqueezeNet in [130] the obtained results come in the second place with a difference from 1%–2% compared with [127] that used Inception-v3. We can find also that, the used model has different results even using the same architectures, due to the representation of data as well as the pre-possessing operations performed before starting the training. Unlike the other tasks like crowd counting and panoptic segmentation, COVID-19 proposed methods do not have any special backbone used for detection and all the backbone have been used.

5. Challenges and future directions

5.1. Deep learning challenges

Deep learning is a trending technology for all computer science and robotics tasks to help and assist human actions. Using artificial neural networks, that are supposed to work like a human brain, deep learning is an aspect of AI that consists of solving the classification and recognition goals for making machines learn from specific data for specific scenarios. Deep learning has many challenges even the development reached in different tasks. For that, a list of deep learning challenges will be discussed in this section. Some these challenges are presented in Fig. 11. While the possible solution corresponding to each challenge is summarized in Fig. 12.

Data Quantity for learning: A large-scale dataset is a necessary condition for a deep-learning model to work well. Also, the performance of such a deep learning system is related to the size of the data used. For that, the annotation and availability of data are real challenges for deep learning methods [135]. For example, we can find many tasks while the data cannot be available to the public like for industrial applications, or a task has few scenarios, also for medical purposes times the data size is small due to the uniqueness of some diseases.

Non-contextual Understanding: The capability of a deep learning model is related to the architecture used which is deep and contains many layers and levels, but it is not related to the level of understanding of it [136]. For example, if a model is proficient in a specific task, and to use this trained model in another close task, all the training and the processing should be re-trained because this model does not understand the context, but lean what it is trained on only. Also, with the development in different domains, a deep learning model should be maintained every time with new features and data to understand the new scenarios.

Data labeling and annotation: In CV, the segmentation of scenes and objects in an image or video represents a crucial challenge. For automatic segmentation, data should be prepared first by annotating and labeling the object or the scenes of interest before starting the training of such a method. The annotations represent also a challenge, due to the number of objects that should be labeled, also any changes to the scenes require another labeling according to the types of the objects and the categories of scenes [137].

Complexity of Architecture: In the literature the effective architectures used as feature extraction are generally complex which them challenging to train, interpret, and optimize [138]. Balancing model complexity with performance requirements is crucial but it can be difficult to achieve, Due to the other parameters like computational resources especially for large scale dataset, and the number of parameters of each model (GFLOPS). In addition, training a complex architecture on some specific tasks can cost in terms of time-consuming more than others.

Overfitting: Deeper features extraction architectures, are sensitive to overfitting, where the model memorizes the training data rather than learning generalizable features [139]. While finding the best parameters such as dropout and weight decay can minimize the impact of this challenge, but finding the right combination require a lot of tests and it can change from a task to another.

Data quality requirements: Training a CNN model requires large-scale annotated datasets, which can be expensive, time-consuming, or even unavailable for certain domains or applications. Data augmentation techniques can help to handle this challenge to some cases but may not address all the scenarios for a representative training data. The quality of data represents also a challenge for deep learning architecture, while using high resolution image for example can be robust to obtains good results, but training its need a computational resource which is another challenge.

Computational Resources: CNN-based models require significant computational resources, including powerful GPUs or TPUs, for training

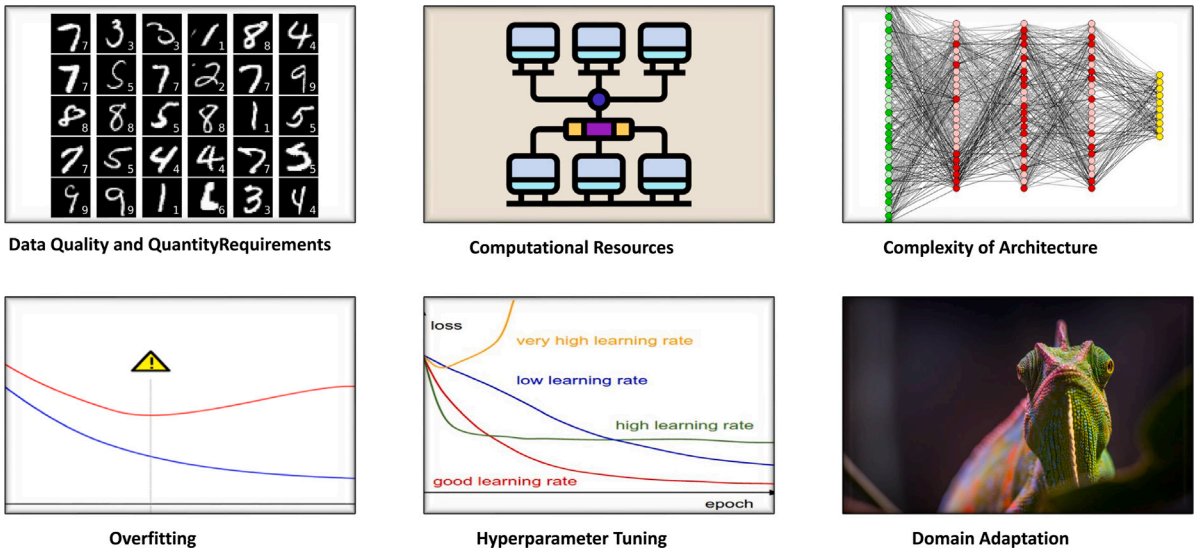


Fig. 11. Challenges of selecting suitable backbone for computer vision tasks.

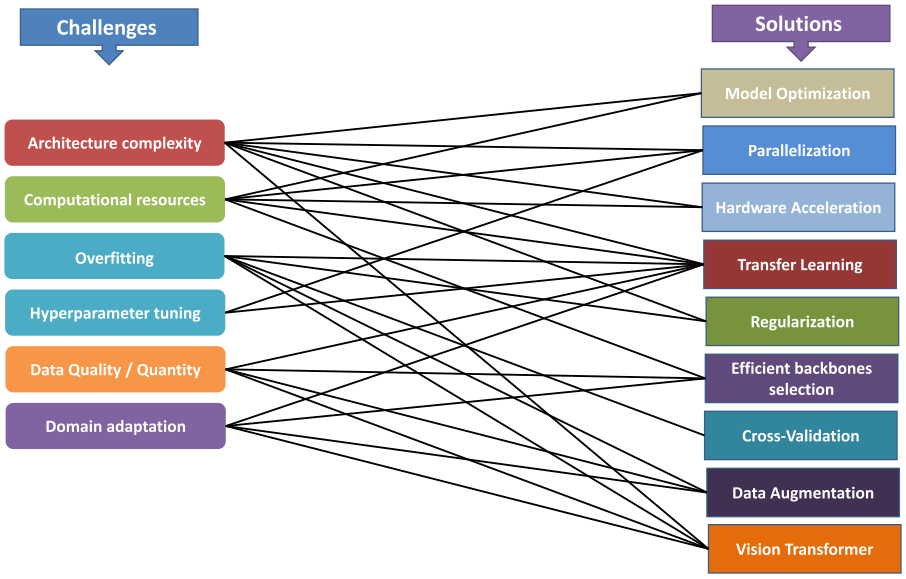


Fig. 12. The possible solution for each computer vision challenge.

and inference. Scaling CNNs to handle larger datasets or more complex architectures increase computational resources demand and limit the accessibility for researchers.

Domain Adaptation: CNNs trained on specific datasets or for a task may not be suitable for different datasets or real-world environments due to domain shifts or biases. While adapting pre-trained CNNs to new domains or tasks with limited annotated data represent a challenge, especially when the target domain differs significantly from the source domain [140]. This is show in the feature extraction models used for some specific tasks in previous section. For example, for panoptic segmentation we can see that the backbones used as features extraction are ResNet ad DeepLab, while VGG-16 and VGG-19 are the suitable backbones from Crowd counting.

5.2. Future directions

Nowadays, DL and DRL techniques are used not just for analyzing the content of images but also replacing the work of humans, like making decisions and annotating the data. in this section, a couple of future directions of DL and DRL are discussed.

Data augmentation:

The lack of large-scale datasets for some tasks represents a challenge for deep learning and deep reinforcement learning methods. For that, the researchers started using DL and DRL techniques for data augmentation, especially for medical imaging that suffer from the lack of data for many diseases. For example, DRL is used for creating new images to be used for training like in [141]. While the authors proposed a DRL architecture for kidney Tumor segmentation. The proposed method starts by augmenting the data before using it for segmentation.

Data annotations: Image and video annotation is a big challenge for researchers specializing in computer vision tasks which need an enormous effort for annotating the objects, labeling the scenes and object for segmentation, or separating the classes for image classification. Also, the format of the annotations can be different from one type of method to another. For example, We can find that for object detection many methods like DetectronV2, YOLO, or EfficientDet used several formats including txt, XML, DarkNet, or JSON formats. A common format for all the methods can be used to overcome this problem. Also, in order to automatize the annotations process for object segmentation

in a video sequence, the authors in [142] proposed a DRL method using an extended version of Dueling DQN. The researchers also started labeling the data for image segmentation using DRL-based techniques. The obtained results using this method are not effective at this time due to the complexity of the purpose, as well as it is the first method that attempted to label the video and images instead of using manual labeling by humans. But with the development of the DRL technique in different computer vision applications, automatic data labeling and annotations can be reached.

Vision Transformer (ViT): Vision Transformer (ViT) presents a good solution to many challenges in deep learning. By replacing convolutional layers with self-attention mechanisms, ViT simplifies architecture and reduces complexity [143,144]. The self-attention mechanism allows a better capturing of long-range dependencies, by learning more robust representations. While ViT benefits from large-scale datasets for pre-training [145]. However, its ability to capture global context through self-attention makes it adaptable to different domains and tasks. ViT introduces new hyperparameters, but automated methods can efficiently optimize these parameters [146]. Overall, while ViT offers promising capabilities that can help existing architectures to improve the performance of deep learning models in terms of feature extraction and robustness of the obtained results.

6. Conclusion

This paper presents an overview of deep learning networks used as a backbone for many proposed architectures for computer vision tasks. A detailed description is provided for each network. In addition, some computer vision tasks are discussed regarding the backbone used for extracting the features. We attempted also to collect the experimental results for each method within each task and compare them based on the backbone used. This review can help the researcher because it is a detailed summarization and comparison of the famous backbones also linked to the code-sources are given. In addition, a set of DL and DRL challenges are presented with some future directions.

Declaration of competing interest

All authors have participated in (a) conception and design, or analysis and interpretation of the data; (b) drafting the article or revising it critically for important intellectual content; and (c) approval of the final version.

This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue.

The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript

Data availability

No data was used for the research described in the article.

Acknowledgments

This publication was made possible by NPRP grant #NPRP12S-0312-190332 from the Qatar National Research Fund (a member of the Qatar Foundation). The statement made here is solely the responsibility of the authors.

References

[1] L. Alzubaidi, A. Al-Sabaawi, J. Bai, A. Dukhan, A.H. Alkenani, A. Al-Asadi, . . ., Y. Gu, Towards risk-free trustworthy artificial intelligence: Significance and requirements, Int. J. Intell. Syst. (2023) 2023.

[2] S. Suganyadevi, V. Seethalakshmi, K. Balasamy, A review on deep learning in medical image analysis, Int. J. Multimed. Inf. Retr. 11 (1) (2022) 19–38.

[3] J. Chai, H. Zeng, A. Li, E.W. Ngai, Deep learning in computer vision: A critical review of emerging techniques and application scenarios, Mach. Learn. Appl. 6 (2021) 100134.

[4] S. Dong, P. Wang, K. Abbas, A survey on deep learning and its applications, Comp. Sci. Rev. 40 (2021) 100379.

[5] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 25, Curran Associates, Inc., 2012, pp. 1097–1105.

[6] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv:1409.1556 [cs].

[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Las Vegas, 2016, pp. 770–778.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Boston, 2015, pp. 1–9.

[9] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, 2015, arXiv:1512.00567 [cs].

[10] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-ResNet and the impact of residual connections on learning, 2016, arXiv:1602.07261 [cs].

[11] M. Lin, Q. Chen, S. Yan, Network in network, 2014, arXiv:1312.4400 [cs].

[12] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258.

[13] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, Kurt Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size, 2016, Preprint, submitted for publication. https://arxiv.org/abs/1602.07360.

[14] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856.

[15] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, Jian Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 116–131.

[16] G. Huang, Z. Liu, L.Van.Der. Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4700–4708.

[17] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, J. Sun, Detnet: A backbone network for object detection, 2018, arXiv preprint arXiv:1804.06215.

[18] Mingxing Tan, Quoc Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 6105–6114.

[19] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv preprint arXiv:1704.04861.

[20] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.

[21] S. Zagoruyko, N. Komodakis, Wide residual networks, 2016, arXiv preprint arXiv:1605.07146.

[22] L.C. Chen, H. Wang, S. Qiao, Scaling wide residual networks for panoptic segmentation, 2020, arXiv preprint arXiv:2011.11675.

[23] Ke Sun, et al., Deep high-resolution representation learning for human pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019.

[24] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, . . ., L. Farhan, Review of deep learning: concepts, CNN architectures, challenges, applications, future directions, J. Big Data 8 (2021) 1–74.

[25] L. Alzubaidi, J. Bai, A. Al-Sabaawi, J. Santamaría, AS. Albahri, BSN. Al-dabbagh, . . ., Y. Gu, A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications, J. Big Data 10 (1) (2023) 46.

[26] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, 2013, arXiv:1311.2524 [cs].

[27] T. Kong, A. Yao, Y. Chen, F. Sun, HyperNet: towards accurate region proposal generation and joint object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Las Vegas, 2016, pp. 845–853.

[28] R. Girshick, Fast R-CNN, 2015, arXiv:1504.08083 [cs].

[29] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2017) 1137–1149.

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD: Single shot MultiBox detector, 2016, pp. 21–37, arXiv:1512.02325 [cs]. 9905.

[31] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015, arXiv:1502.03167 [cs].

[32] J. Dai, Y. Li, K. He, J. Sun, R-FCN: object detection via region-based fully convolutional networks, in: D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 29, Curran Associates, Inc., 2016, pp. 379–387.

[33] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 448–456.

[34] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, K. Murphy, Speed/accuracy trade-offs for modern convolutional object detectors, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Honolulu, 2017, pp. 3296–3297.

[35] A. Shrivastava, R. Sukthankar, J. Malik, A. Gupta, Beyond skip connections: Top-down modulation for object detection, 2016, arXiv:1612.06851 [cs].

[36] J. Redmon, A. Farhadi, YOLO9000: Better, faster, stronger, 2016, arXiv:1612.08242 [cs].

[37] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1492–1500.

[38] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, Bin Xiao, Deep high-resolution representation learning for visual recognition, IEEE Trans. Pattern Anal. Mach. Intell. (2019).

[39] B. Roy, S. Nandy, D. Ghosh, D. Dutta, P. Biswas, T. Das, MOXA: A deep learning based unmanned approach for real-time monitoring of people wearing medical masks, Trans. Indian Natl. Acad. Eng. 5 (3) (2020) 509–518, http://dx.doi.org/10.1007/s41403-020-00157-z, 2020/09/01.

[40] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.

[41] A. Bochkovskiy, C.Y. Wang, H.Y.M. Liao, Yolov4: Optimal speed and accuracy of object detection, 2020, arXiv preprint arXiv:2004.10934.

[42] detectron-v2, [online] Available: https://github.com/facebookresearch/detectron2.

[43] YOLO-v5, 2020, [online] Available: https://github.com/ultralytics/yolov5, (Accessed 24 December 2020).

[44] M.H. Yap, R. Hachiuma, A. Alavi, R. Brungel, M. Goyal, H. Zhu, B. Cassidy, et al., Deep learning in diabetic foot ulcers detection: A comprehensive evaluation, 2020, arXiv preprint arXiv:2010.03341.

[45] Y. Li, X. Zhang, D. Chen, Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1091–1100.

[46] X. Chen, Y. Bin, N. Sang, C. Gao, Scale pyramid network for crowd counting, in: 2019 IEEE Winter Conference on Applications of Computer Vision, WACV, IEEE, 2019, pp. 1941–1950.

[47] L. Liu, W. Jia, J. Jiang, S. Amirgholipour, Y. Wang, M. Zeibots, X. He, Denet: A universal network for counting crowd with varying densities and scales, IEEE Trans. Multimed. (2020).

[48] W. Liu, M. Salzmann, P. Fua, Context-aware crowd counting, in: Proc. Conf. Comput. Vis. Pattern Recognit., 2019, pp. 5099–5108.

[49] J. Gao, Q. Wang, Y. Yuan, Scar: Spatial-/channel-wise attention regression networks for crowd counting, Neurocomputing 363 (2019) 1–8.

[50] S. Bai, Z. He, Y. Qiao, H. Hu, W. Wu, J. Yan, Adaptive dilated network with self-correction supervision for counting, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 4594–4603.

[51] X. Jiang, L. Zhang, M. Xu, T. Zhang, P. Lv, B. Zhou, X. Yang, Y. Pang, Attention scaling for crowd counting, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 4706–4715.

[52] Z. Ma, X. Wei, X. Hong, Y. Gong, Bayesian loss for crowd count estimation with point supervision, in: Proc. IEEE Int. Conf. Comput. Vis., 2019, pp. 6142–6151.

[53] Q. Wang, J. Gao, W. Lin, Y. Yuan, Learning from synthetic data for crowd counting in the wild, in: Proc. Conf. Comput. Vis. Pattern Recognit., 2019, pp. 8198–8207.

[54] P. Wang, C. Gao, Y. Wang, H. Li, Y. Gao, MobileCount: An efficient encoder–decoder framework for real-time crowd counting, Neurocomputing 407 (2020) 292–299.

[55] O. Elharrouss, N. Almaadeed, S. Al-Maadeed, A. Bouridane, A. Beghdadi, A combined multiple action recognition and summarization for surveillance video sequences, Appl. Intell. 51 (2) (2021) 690–712.

[56] O. Elharrouss, N. Al-Maadeed, S. Al-Maadeed, Video summarization based on motion detection for surveillance systems, in: 2019 15th International Wireless Communications & Mobile Computing Conference, IWCMC, IEEE, 2019, pp. 366–371.

[57] M. Narasimhan, A. Rohrbach, T. Darrell, CLIP-it! language-guided video summarization, 2021, arXiv preprint arXiv:2107.00650.

[58] H. Kanafani, J.A. Ghauri, S. Hakimov, R. Ewerth, Unsupervised video summarization via multi-source features, 2021, arXiv preprint arXiv:2105.12532.

[59] M.S. Nair, J. Mohan, Static video summarization using multi-CNN with sparse autoencoder and random forest classifier, Signal Image Video Process. 15 (4) (2021) 735–742.

[60] J.H. Huang, L. Murn, M. Mrak, M. Worring, GPT2MVS: Generative pre-trained transformer-2 for multi-modal video summarization, 2021, arXiv preprint arXiv:2104.12465.

[61] M. Rafiq, G. Rafiq, R. Agyeman, G.S. Choi, S.I. Jin, Scene classification for sports video summarization using transfer learning, Sensors 20 (6) (2020) 1702.

[62] Y. Zhang, Q. Li, X. Zhao, M. Tan, Robot learning through observation via coarse-to-fine grained video summarization, Appl. Soft Comput. 99 (2021) 106913.

[63] Y. Zhang, X. Liang, D. Zhang, M. Tan, E.P. Xing, Unsu-pervised object-level video summarization with online motionautoencoder, Pattern Recognit. Lett. 130 (2020) 376–385.

[64] H. Wang, K. Wang, Y. Wu, Z. Wang, L. Zou, User preference-aware video highlight detection via deep reinforcement learn-ing, Multimedia Tools Appl. (2020) 1–10.

[65] J. Lei, Q. Luan, X. Song, X. Liu, D. Tao, M. Song, Actionparsing-driven video summarization based on reinforcementlearning, IEEE Trans. Circuits Syst. Video Technol. 29 (7) (2018) 2126–2137.

[66] Y. Chen, L. Tao, X. Wang, T. Yamasaki, Weakly supervised video summarization by hierarchical reinforcement learning, in: Proceedings of the ACM Multimedia Asia, 2019, pp. 1–6.

[67] Y. Zhang, M. Kampffmeyer, X. Zhao, M. Tan, Deep rein-forcement learning for query-conditioned video summarization, Appl. Sci. 9 (4) (2019) 750.

[68] K. Zhou, Y. Qiao, T. Xiang, Deep reinforcement learning for unsupervised video summarization with diversity representativeness reward, Proc. AAAI Conf. Artif. Intell. 32 (1) (2018).

[69] L. Wang, Y. Zhu, H. Pan, Unsupervised reinforcement learning for video summarization reward function, in: Proceedings of the 2019 International Conference on Image, Video and Signal Processing, 2019, pp. 40–44.

[70] Z. Li, L. Yang, Weakly supervised deep reinforcement learning for video summarization with semantically meaningful reward, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 3239–3247.

[71] C. Yang, Y. Xu, J. Shi, B. Dai, B. Zhou, Temporal pyramid network for action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 591–600.

[72] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, L. Wang, Tea: Temporal excitation and aggregation for action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 909–918.

[73] S. Sudhakaran, S. Escalera, O. Lanz, Gate-shift networks for video action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1102–1111.

[74] Joao Carreira, Andrew Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, in: Computer Vision and Pattern Recognition, CVPR, Vol. 2, 2017, p. 5.

[75] J. Munro, D. Damen, Multi-modal domain adaptation for fine-grained action recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 122–132.

[76] J. Li, X. Liu, W. Zhang, M. Zhang, J. Song, N. Sebe, Spatio-temporal attention networks for action recognition and detection, IEEE Trans. Multimed. 22 (11) (2020) 2990–3001.

[77] W. Wu, D. He, X. Tan, S. Chen, S. Wen, Multi-agent reinforcement learning based frame sampling for effective untrimmed video recognition, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 6222–6231.

[78] C. Xiaokai, G. Ke, C. Juan, Predictability analyzing: Deep reinforcement learning for early action recognition, in: 2019 IEEE International Conference on Multimedia and Expo, ICME, IEEE, 2019, pp. 958–963.

[79] L. Chen, J. Lu, Z. Song, J. Zhou, Part-activated deep reinforcement learning for action prediction, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 421–436.

[80] H. Li, J. Chen, R. Hu, M. Yu, H. Chen, Z. Xu, Action recognition using visual attention with reinforcement learning, in: International Conference on Multimedia Modeling, Springer, 2019, pp. 365–376.

[81] W. Dong, Z. Zhang, T. Tan, Attention-aware sampling via deep reinforcement learning for action recognition, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 8247–8254.

[82] G. Wang, W. Wang, J. Wang, Y. Bu, Better deep visual attention with reinforcement learning in action recognition, in: 2017 IEEE International Symposium on Circuits and Systems, ISCAS, IEEE, 2017, pp. 1–4.

[83] S. Ge, et al., Low-resolution face recognition in the wild via selective knowledge distillation, IEEE Trans. Image Process. 28 (4) (2019) 2051–2062.

[84] I. Masi, et al., Pose-aware face recognition in the wild, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016.

[85] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, Wei Liu, Cosface: Large margin cosine loss for deep face recognition, in: CVPR, 2018.

[86] X. Zhang, Z. Fang, Y. Wen, Z. Li, Y. Qiao, Range loss for deep face recognition with long-tailed training data, in: IEEE International Conference on Computer Vision, ICCV, 2017.

[87] Yaoyao Zhong, Weihong Deng, Mei Wang, et al., Unequal-training for deep face recognition with long-tailed noisy data, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 7812–7821.

[88] Kai Zhao, Jingyi Xu, Ming-Ming Cheng, Regularface: Deep face recognition via exclusive regularization, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 1136–1144.

[89] Jiankang Deng, Jia Guo, Niannan Xue, et al., Arcface: Additive angular margin loss for deep face recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 4690–4699.

[90] Bingyu Liu, Weihong Deng, Yaoyao Zhong, et al., Fair loss: Margin-aware reinforcement learning for deep face recognition, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, 2019, pp. 10052–10061.

[91] O. Elharrouss, N. Almaadeed, S. Al-Maadeed, F. Khelifi, Pose-invariant face recognition with multitask cascade networks, Neural Comput. Appl. (2022) 1–14.

[92] Y. Rao, J. Lu, J. Zhou, Attention-aware deep reinforcementlearning for video face recognition, in: Proceedings of the IEEEinternational Conference on Computer Vision, 2017, pp. 3931–3940.

[93] B. Liu, W. Deng, Y. Zhong, M. Wang, J. Hu, X. Tao, Y. Huang, Fair loss: Margin-aware reinforcement learning for deep facerecognition, in: Proceedings of the IEEE/CVF International Confer-Ence on Computer Vision, 2019, pp. 10052–10061.

[94] M. Wang, W. Deng, Mitigating bias in face recognition usingskewness-aware reinforcement learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9322–9331.

[95] T.Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.

[96] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, 2018, arXiv preprint arXiv:1804.02767.

[97] Mingxing Tan, Ruoming Pang, Quoc V. Le, EfficientDet: Scalable and efficient object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2020.

[98] S. Qiao, L.C. Chen, A. Yuille, Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 10213–10224.

[99] D. de Geus, P. Meletis, G. Dubbelman, Fast panoptic segmentation network, IEEE Robot. Autom. Lett. 5 (2) (2020) 1742–1749.

[100] H. Wang, Y. Zhu, B. Green, H. Adam, A. Yuille, L.-C. Chen, Axial-deeplab: Stand-alone axial-attention for panoptic segmentation, in: European Conference on Computer Vision, Springer, 2020, pp. 108–126.

[101] Y. Chen, G. Lin, S. Li, O. Bourahla, Y. Wu, F. Wang, J. Feng, M. Xu, X. Li, Banet: Bidirectional aggregation network with occlusion handling for panoptic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 3793–3802.

[102] D. Kim, S. Woo, J.-Y. Lee, I.S. Kweon, Video panoptic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9859–9868.

[103] Y. Wu, G. Zhang, Y. Gao, X. Deng, K. Gong, X. Liang, L. Lin, Bidirectional graph reasoning network for panoptic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9080–9089.

[104] Q. Chen, A. Cheng, X. He, P. Wang, J. Cheng, Spatialflow: Bridging all tasks for panoptic segmentation, IEEE Trans. Circuits Syst. Video Technol. (2020).

[105] M. Weber, J. Luiten, B. Leibe, Single-shot panoptic segmentation, 2019, arXiv preprint arXiv:1911.00764.

[106] Y. Li, X. Chen, Z. Zhu, L. Xie, G. Huang, D. Du, X. Wang, Attention-guided unified network for panoptic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7026–7035.

[107] H. Liu, C. Peng, C. Yu, J. Wang, X. Liu, G. Yu, W. Jiang, An end-to-end network for panoptic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6172–6181.

[108] O. Elharrouss, N. Almaadeed, K. Abualsaud, A. Al-Ali, A. Mohamed, T. Khattab, S. Al-Maadeed, Drone-SCNet: Scaled cascade network for crowd counting on drone images, IEEE Trans. Aerosp. Electron. Syst. (2021).

[109] S. Hwang, S.W. Oh, S.J. Kim, Single-shot path integrated panoptic segmentation, 2020, arXiv preprint arXiv:2012.01632.

[110] J. Son, S. Lee, Hidden enemy visualization using fast panoptic segmentation on battlefields, in: 2021 IEEE International Conference on Big Data and Smart Computing, BigComp, IEEE, 2021, pp. 291–294.

[111] Y. Yang, H. Li, X. Li, Q. Zhao, J. Wu, Z. Lin, Sognet: Scene overlap graph network for panoptic segmentation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 12637–12644.

[112] H. Chen, C. Shen, Z. Tian, Unifying instance and panoptic segmentation with dynamic rank-1 convolutions, 2020, arXiv preprint arXiv:2011.09796.

[113] R. Mohan, A. Valada, Efficientps: Efficient panoptic segmentation, Int. J. Comput. Vis. (2021) 1–29.

[114] R. Mohan, A. Valada, Robust vision challenge 2020–1st place report for panoptic segmentation, 2020, pp. 1–8, arXiv preprint arXiv:2008.10112.

[115] Michael J. Horry, Manoranjan Paul, Anwaar Ulhaq, Biswajeet Pradhan, Manash Saha, Nagesh Shukla, X-ray image based COVID-19 detection using pre-trained deep learning models, 2020.

[116] P.G. Moutounet-Cartan, Deep convolutional neural networks to diagnose COVID-19 and other pneumonia diseases from posteroanterior chest X-rays, 2020, arXiv preprint arXiv:2005.00845.

[117] Eduardo José da S. Luz, Pedro Lopes Silva, Rodrigo Silva, Ludmila Silva, Gladston Moreira, David Menotti, Towards an effective and efficient deep learning model for COVID-19 patterns detection in X-ray images, 2020, CoRR.

[118] M.J. Horry, et al., COVID-19 detection through transfer learning using multimodal imaging data, IEEE Access 8 (2020) 149808–149824.

[119] I.D. Apostolopoulos, T.A. Mpesiana, Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks, Phys. Eng. Sci. Med. (2020) 1.

[120] Ezz El-Din Hemdan, Marwa A. Shouman, Mohamed Esmail Karar, Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images, 2020, arXiv preprint arXiv:2003.11055.

[121] M.E.H. Chowdhury, et al., Can AI help in screening viral and COVID-19 pneumonia? IEEE Access 8 (2020) 132665–132676, http://dx.doi.org/10.1109/ACCESS.2020.3010287.

[122] Shervin Minaee, Rahele Kafieh, Milan Sonka, Shakib Yazdani, Ghazaleh Jamalipour Soufi, Deep-covid: Predicting covid-19 from chest x-ray images using deep transfer learning, 2020, arXiv preprint arXiv:2004.09363.

[123] M. Loey, F. Smarandache, NE. M Khalifa, Within the lack of chest COVID-19 X-ray dataset: A novel detection model based on GAN and deep transfer learning, Symmetry 12 (4) (2020) 651.

[124] Tayyip Ozcan, A deep learning framework for coronavirus disease (COVID-19) detection in X-ray images, 2020.

[125] M. Rahimzadeh, A. Attar, A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2, Inform. Med. Unlocked (2020) 100360.

[126] N.S. Punn, S. Agarwal, Automated diagnosis of COVID-19 with limited posteroanterior chest X-ray images using fine-tuned deep neural networks, 2020, arXiv preprint arXiv:2004.11676.

[127] A. Narin, C. Kaya, Z. Pamuk, Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks, 2020, arXiv preprint arXiv:2003.10849.

[128] Muhammad Farooq, Abdul Hafeez, Covid-resnet: A deep learning framework for screening of covid19 from radiographs, 2020, arXiv preprint arXiv:2003.14395.

[129] G. Maguolo, L. Nanni, A critic evaluation of methods for covid-19 automatic detection from x-ray images, 2020, arXiv preprint arXiv:2004.12823.

[130] Ferhat Ucar, Deniz Korkmaz, Covidiagnosis-net: Deep Bayes-SqueezeNet based diagnostic of the coronavirus disease 2019 (COVID-19) from X-ray images, Med. Hypotheses (2020) 109761.

[131] M. Rahimzadeh, A. Attar, A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2, Inform. Med. Unlocked 19 (2020) 100360.

[132] T. Ozturk, M. Talo, E.A. Yildirim, U.B. Baloglu, O. Yildirim, U.R. Acharya, Automated detection of COVID-19 cases using deep neural networks with X-ray images, Comput. Biol. Med. (2020) 103792.

[133] O. Elharrouss, S. Al-Maadeed, N. Subramanian, N. Ottakath, N. Almaadeed, Y. Himeur, Panoptic segmentation: A review, 2021, arXiv preprint arXiv:2111.10250.

[134] S. Qiao, L.-C. Chen, A. Yuille, Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution, 2020, arXiv preprint arXiv:2006.02334.

[135] F. Abtahi, Z. Zhu, A.M. Burry, A deep reinforcement learning approach to character segmentation of license plate images, in: 2015 14th IAPR International Conference on Machine Vision Applications, MVA, IEEE, 2015, pp. 539–542.

[136] Y. Akbari, N. Almaadeed, S. Al-maadeed, O. Elharrouss, Applications, databases and open computer vision research from drone videos and images: a survey, Artif. Intell. Rev. 54 (5) (2021) 3887–3938.

[137] M. Elasri, O. Elharrouss, S. Al-Maadeed, H. Tairi, Image generation: A review, Neural Process. Lett. (2022) 1–38.

[138] X. Hu, L. Chu, J. Pei, W. Liu, J. Bian, Model complexity of deep learning: A survey, Knowl. Inf. Syst. 63 (2021) 2585–2619.

[139] L. Rice, E. Wong, Z. Kolter, Overfitting in adversarially robust deep learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 8093–8104.

[140] A. Farahani, S. Voghoei, K. Rasheed, H.R. Arabnia, A brief review of domain adaptation, in: Advances in Data Science and Information Engineering: Proceedings from ICDATA 2020 and IKE 2020, 2021, pp. 877–894.

[141] T. Qin, Z. Wang, K. He, Y. Shi, Y. Gao, D. Shen, Automatic data augmentation via deep reinforcement learning for effective kidney tumor segmentation, in: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2020, pp. 1419–1423.
[142] V. Varga, A.L. őrincz, Reducing human efforts in video segmentation annotation with reinforcement learning, Neurocomputing 405 (2020) 247–258.
[143] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, . . ., I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[144] S. Jamil, M. Jalil Piran, O.J. Kwon, A comprehensive survey of transformers for computer vision, Drones 7 (5) (2023) 287.
[145] J. Bi, Z. Zhu, Q. Meng, Transformer in computer vision, in: 2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology, CEI, IEEE, 2021, pp. 178–188.
[146] S. Khan, M. Naseer, M. Hayat, S.W. Zamir, F.S. Khan, M. Shah, Transformers in vision: A survey, ACM Comput. Surv. (CSUR) 54 (10s) (2022) 1–41.