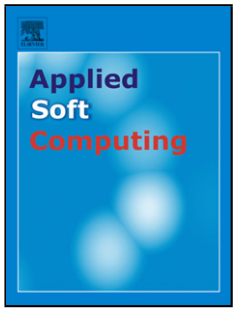


Title: A Survey on Deep Learning Techniques for Image and Video Semantic Segmentation

Author: Alberto Garcia-Garcia Sergio Orts-Escolano Sergiu Oprea Victor Villena-Martinez Pablo Martinez-Gonzalez Jose Garcia-Rodriguez

PII: S1568-4946(18)30281-3
DOI: https://doi.org/doi:10.1016/j.asoc.2018.05.018
Reference: ASOC 4884

To appear in: *Applied Soft Computing*

Received date: 9-10-2017
Revised date: 25-4-2018
Accepted date: 12-5-2018

Please cite this article as: Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Pablo Martinez-Gonzalez, Jose Garcia-Rodriguez, A Survey on Deep Learning Techniques for Image and Video Semantic Segmentation, *<![CDATA[Applied Soft Computing Journal]]>* (2018), https://doi.org/10.1016/j.asoc.2018.05.018

A Survey on Deep Learning Techniques for Image and Video Semantic Segmentation

Alberto Garcia-Garcia[a,*], Sergio Orts-Escolano[a], Sergiu Oprea[a], Victor Villena-Martinez[a], Pablo Martinez-Gonzalez[a], Jose Garcia-Rodriguez[a]

[a] *3D Perception Lab, University of Alicante (Spain)*

Abstract

Image semantic segmentation is more and more being of interest for computer vision and machine learning researchers. Many applications on the rise need accurate and efficient segmentation mechanisms: autonomous driving, indoor navigation, and even virtual or augmented reality systems to name a few. This demand coincides with the rise of deep learning approaches in almost every field or application target related to computer vision, including semantic segmentation or scene understanding. This paper provides a review on deep learning methods for semantic segmentation applied to various application areas. Firstly, we formulate the semantic segmentation problem and define the terminology of this field as well as interesting background concepts. Next, the main datasets and challenges are exposed to help researchers decide which are the ones that best suit their needs and goals. Then, existing methods are reviewed, highlighting their contributions and their significance in the field. We also devote a part of the paper to review common loss functions and error metrics for this problem. Finally, quantitative results are given for the described methods and the datasets in which they were evaluated, following up with a discussion of the results. At last, we point out a set of promising future works and draw our own conclusions about the state of the art of semantic segmentation using deep learning techniques.

Keywords: Semantic Segmentation, Deep Learning, Scene Labeling

* *Email addresses:* agarcia@dtic.ua.es (Alberto Garcia-Garcia), sorts@ua.es (Sergio Orts-Escolano), soprea@dtic.ua.es (Sergiu Oprea), vvillena@dtic.ua.es (Victor Villena-Martinez), pmartinez@dtic.ua.es (Pablo Martinez-Gonzalez), jgarcia@dtic.ua.es (Jose Garcia-Rodriguez)

1. Introduction

Nowadays, semantic segmentation – applied to still 2D images, video, and even 3D or volumetric data – is one of the key problems in the field of computer vision. Looking at the big picture, semantic segmentation is one of the high-level tasks that paves the way towards complete scene understanding. The importance of scene understanding as a core computer vision problem is highlighted by the fact that an increasing number of applications nourish from inferring knowledge from imagery. Some of those applications include autonomous driving [1][2][3], human-machine interaction [4], computational photography [5], image search engines [6], and augmented reality to name a few. Such problem has been addressed in the past using various traditional computer vision and machine learning techniques. Despite the popularity of those kind of methods, the deep learning revolution has turned the tables so that many computer vision problems – semantic segmentation among them – are being tackled using deep architectures, usually Convolutional Neural Networks (CNNs) [7][8][9][10][11], which are surpassing other approaches by a large margin in terms of accuracy and sometimes even efficiency. However, deep learning is far from the maturity achieved by other old-established branches of computer vision and machine learning. Because of that, there is a lack of unifying works and state of the art reviews. The ever-changing state of the field makes initiation difficult and keeping up with its evolution pace is an incredibly time-consuming task due to the sheer amount of new literature being produced. This makes it hard to keep track of the works dealing with semantic segmentation and properly interpret their proposals, prune subpar approaches, and validate results.

To the best of our knowledge, this is the first review to focus explicitly on deep learning for semantic segmentation. Various semantic segmentation surveys already exist such as the works by Zhu *et al.*[12] and Thoma[13], which do a great work summarizing and classifying existing methods, discussing datasets and metrics, and providing design choices for future research directions. However, they lack some of the most recent datasets, they do not analyze frameworks, and none of them provide details about deep learning techniques. Because of that, we consider our work to be novel and helpful thus making it a significant contribution for the research community.

The key contributions of our work are as follows:

- We provide a broad survey of existing datasets that might be useful for segmentation projects with deep learning techniques.
- An in-depth and organized review of the most significant methods that use deep learning for semantic segmentation, their origins, and their contributions.
- A thorough performance evaluation which gathers quantitative metrics such as accuracy, execution time, and memory footprint.
- A discussion about the aforementioned results, as well as a list of possible future works that might set the course of upcoming advances, and a conclusion summarizing the state of the art of the field.

The remainder of this paper is organized as follows. Firstly, Section 2 introduces the semantic segmentation problem as well as notation and con-

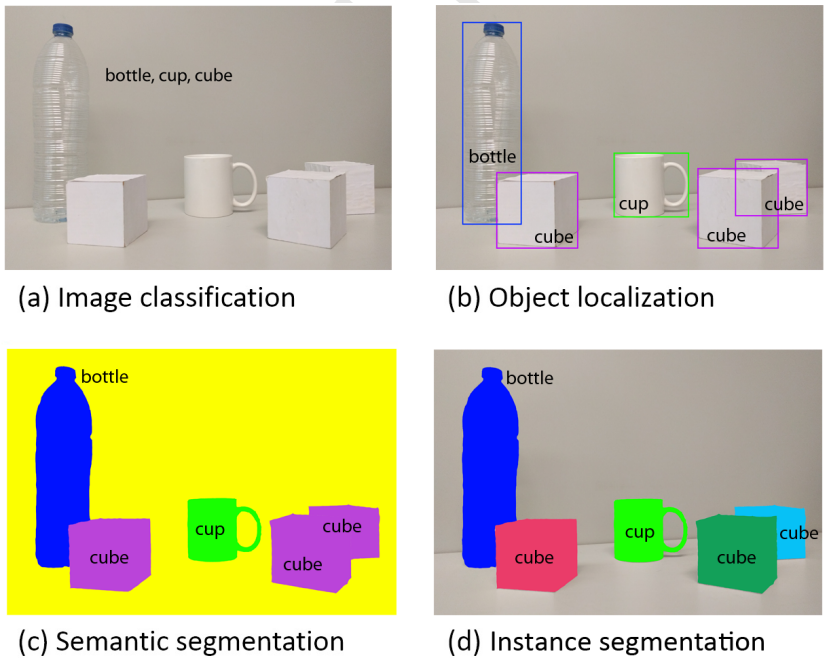


Figure 1: Evolution of object recognition or scene understanding from coarse-grained to fine-grained inference: classification, detection or localization, semantic segmentation, and instance segmentation.

ventions commonly used in the literature. Other background concepts such as common deep neural networks are also reviewed. Next, Section 3 describes existing datasets, challenges, and benchmarks. It also reviews existing methods following a bottom-up complexity order based on their contributions. This section focuses on describing the theory and highlights of those methods rather than performing a quantitative evaluation. Finally, Section 4 presents a brief discussion on the presented methods based on their quantitative results on the aforementioned datasets. In addition, future research directions are also laid out. At last, Section 5 summarizes the paper and draws conclusions about this work and the state of the art of the field.

2. Terminology and Background Concepts

In order to properly understand how semantic segmentation is tackled by modern deep learning architectures, it is important to know that it is not an isolated field but rather a natural step in the progression from coarse to fine inference. The origin could be located at classification, which consists of making a prediction for a whole input, i.e., predicting which is the object in an image or even providing a ranked list if there are many of them. Localization or detection is the next step towards fine-grained inference, providing not only the classes but also additional information regarding the spatial location of those classes, e.g., centroids or bounding boxes. Providing that, it is obvious that semantic segmentation is the natural step to achieve fine-grained inference, its goal: make dense predictions inferring labels for every pixel; this way, each pixel is labeled with the class of its enclosing object or region. Further improvements can be made, such as instance segmentation (separate labels for different instances of the same class) and even part-based segmentation (low-level decomposition of already segmented classes into their components). Figure 1 shows the aforementioned evolution. In this review, we will mainly focus on generic scene labeling, i.e., per-pixel class segmentation, but we will also review the most important methods on instance and part-based segmentation.

In the end, the per-pixel labeling problem can be reduced to the following formulation: find a way to assign a state from the label space $\mathcal{L} = \{l_1, l_2, ..., l_k\}$ to each one of the elements of a set of random variables $\mathcal{X} = \{x_1, x_2, ..., x_N\}$. Each label l represents a different class or object, e.g., aeroplane, car, traffic sign, or background. This label space has k possible states which are usually extended to $k + 1$ and treating l_0 as background or a void

class. Usually, $\mathcal{X}$ is a 2D image of $W \times H = N$ pixels x. However, that set of random variables can be extended to any dimensionality such as volumetric data or hyperspectral images.

Apart from the problem formulation, it is important to remark some background concepts that might help the reader to understand this review. Firstly, common networks, approaches, and design decisions that are often used as the basis for deep semantic segmentation systems. In addition, common techniques for training such as transfer learning. At last, data preprocessing and augmentation approaches.

2.1. Common Deep Network Architectures

As we previously stated, certain deep networks have made such significant contributions to the field that they have become widely known standards. It is the case of AlexNet, VGG-16, GoogLeNet, and ResNet. Such was their importance that they are currently being used as building blocks for many segmentation architectures. For that reason, we will devote this section to review them.

2.1.1. AlexNet

AlexNet was the pioneering deep CNN that won the ILSVRC-2012 with a TOP-5 test accuracy of 84.6% while the closest competitor, which made use of traditional techniques instead of deep architectures, achieved a 73.8% accuracy in the same challenge. The architecture presented by Krizhevsky *et al.* [14] was relatively simple. It consists of five convolutional layers, max-pooling ones, Rectified Linear Units (ReLUs) as non-linearities, three fully-connected layers, and dropout. Figure 2 shows that CNN architecture.

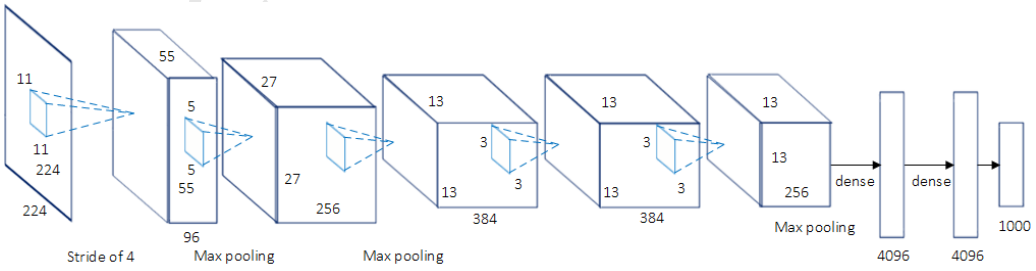


Figure 2: AlexNet Convolutional Neural Network architecture. Figure reproduced from [14].

2.1.2. VGG

Visual Geometry Group (VGG) is a CNN model introduced by the Visual Geometry Group (VGG) from the University of Oxford. They proposed various models and configurations of deep CNNs [15], one of them was submitted to the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)-2013. That model, also known as VGG-16 due to the fact that it is composed by 16 weight layers, became popular thanks to its achievement of 92.7% TOP-5 test accuracy. Figure 3 shows the configuration of VGG-16. The main difference between VGG-16 and its predecessors is the use of a stack of convolution layers with small receptive fields in the first layers instead of few layers with big receptive fields. This leads to less parameters and more non-linearities in between, thus making the decision function more discriminative and the model easier to train.

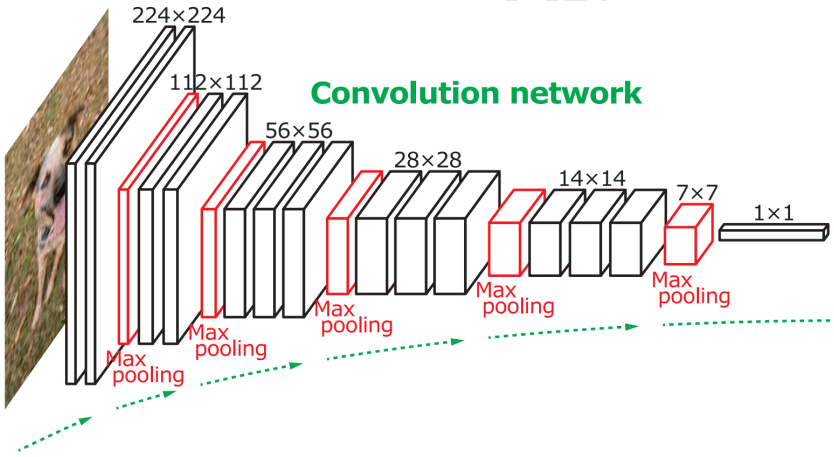


Figure 3: VGG-16 CNN architecture. Figure extracted from [16].

2.1.3. GoogLeNet

GoogLeNet is a network introduced by Szegedy *et al.* [17] which won the ILSVRC-2014 challenge with a TOP-5 test accuracy of 93.3%. This CNN architecture is characterized by its complexity, emphasized by the fact that it is composed by 22 layers and a newly introduced building block called *inception* module (see Figure 4). This new approach proved that CNN layers could be stacked in more ways than a typical sequential manner. In fact, those modules consist of a Network in Network (NiN) layer, a pooling operation, a large-sized convolution layer, and small-sized convolution layer. All of

them are computed in parallel and followed by 1×1 convolution operations to reduce dimensionality. Thanks to those modules, this network puts special consideration on memory and computational cost by significantly reducing the number of parameters and operations.

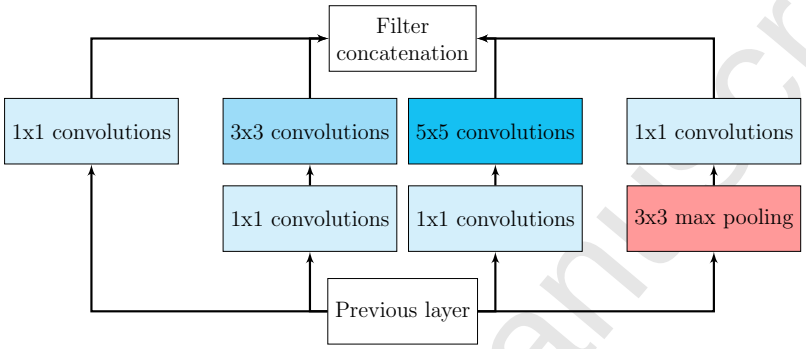


Figure 4: Inception module with dimensionality reduction from the GoogLeNet architecture. Figure reproduced from [17].

2.1.4. ResNet

Microsoft's ResNet[18] is specially remarkable thanks to winning ILSVRC-2016 with 96.4% accuracy. Apart from that fact, the network is well-known due to its depth (152 layers) and the introduction of residual blocks (see Figure 5). The residual blocks address the problem of training a really deep architecture by introducing identity skip connections so that layers can copy their inputs to the next layer.

The intuitive idea behind this approach is that it ensures that the next layer learns something new and different from what the input has already encoded (since it is provided with both the output of the previous layer and its unchanged input). In addition, this kind of connections help overcoming the vanishing gradients problem.

2.1.5. ReNet

In order to extend Recurrent Neural Networks (RNNs) architectures to multi-dimensional tasks, Graves et al. [19] proposed a Multi-dimensional Recurrent Neural Network (MDRNN) architecture which replaces each single recurrent connection from standard RNNs with d connections, where d is the number of spatio-temporal data dimensions. Based on this initial approach,

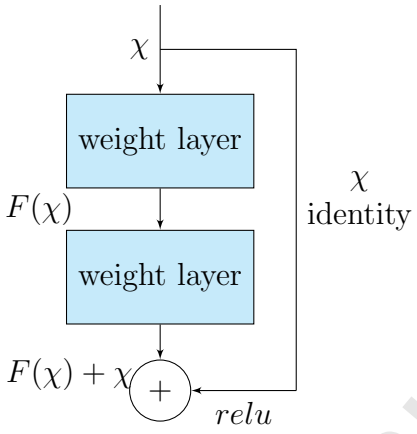


Figure 5: Residual block from the ResNet architecture. Figure reproduced from [18].

Visin el al. [20] proposed ReNet architecture in which instead of multidimensional RNNs, they have been using usual sequence RNNs. In this way, the number of RNNs is scaling linearly at each layer regarding to the number of dimensions d of the input image ($2d$). In this approach, each convolutional layer (convolution + pooling) is replaced with four RNNs sweeping the image vertically and horizontally in both directions as we can see in Figure 6.

2.2. Transfer Learning and Fine-tuning

Training a deep neural network from scratch is often not feasible because of various reasons: a dataset of sufficient size is required (and not usually available) and reaching convergence can take too long for the experiments to be worth. Even if a dataset large enough is available and convergence does not take that long, it is often helpful to start with pre-trained weights instead of random initialized ones[21][22]. Fine-tuning the weights of a pre-trained network by continuing with the training process is one of the major transfer learning scenarios.

Yosinski *et al.*[23] proved that transferring features even from distant tasks can be better than using random initialization, taking into account that the transferability of features decreases as the difference between the pre-trained task and the target one increases.

However, applying this transfer learning technique is not completely straightforward. On the one hand, there are architectural constraints that must be met to use a pre-trained network. Nevertheless, since it is not usual to come up with a whole new architecture, it is common to reuse already existing

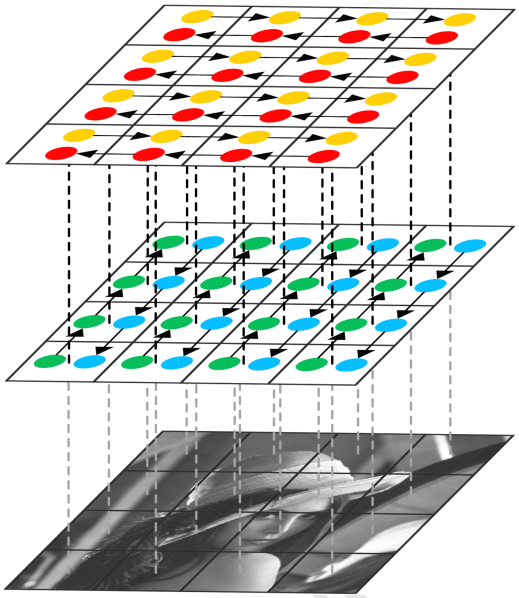


Figure 6: One layer of ReNet architecture modeling vertical and horizontal spatial dependencies. Extracted from [20].

network architectures (or components) thus enabling transfer learning. On the other hand, the training process differs slightly when fine-tuning instead of training from scratch. It is important to choose properly which layers to fine-tune – usually the higher-level part of the network, since the lower one tends to contain more generic features – and also pick an appropriate policy for the learning rate, which is usually smaller due to the fact that the pre-trained weights are expected to be relatively good so there is no need to drastically change them.

Transfer learning from ImageNet pre-trained networks is common for semantic segmentation. Nevertheless, the interest in modeling representations using self-supervised, weakly-supervised and unsupervised learning increasingly predominates. Pre-trained weights of context encoders used for context-based pixel prediction aiming semantic hole-filling, have been used for initializing a FCN [24]. This setting outperform random initialized networks and plain autoencoders thus advancing the state of the art in semantic segmentation.

Due to the inherent difficulty of gathering and creating per-pixel labelled segmentation datasets, their scale is not as large as the size of classification

datasets such as ImageNet[25][26]. This problem gets even worse when dealing with RGB-D or 3D datasets, which are even smaller. For that reason, transfer learning, and in particular fine-tuning from pre-trained classification networks is a common trend for segmentation networks and has been successfully applied in the methods that we will review in the following sections.Although the datasets are smaller the more detailed the semantic labelling, a feasible solution to this problem is the use of synthetic datasets extracted from commercial video games. Some experiments on semantic segmentation shows that models trained with game data and $\frac{1}{3}$ of the CamVid dataset outperform methods trained on the complete CamVid training set [27]. This technique relies on a two-staged process: (1) real and synthetic data are used jointly to train the model, (2) model will be fine-tunned with only real data.

Due to the inherent difficulty of gathering and creating per-pixel labelled segmentation datasets, their scale is not as large as the size of classification datasets such as ImageNet[25][26]. This problem gets even worse when dealing with RGB-D or 3D datasets, which are even smaller. For that reason, transfer learning, and in particular fine-tuning from pre-trained classification networks is a common trend for segmentation networks and has been successfully applied in the methods that we will review in the following sections.

2.3. Data Preprocessing and Augmentation

Data augmentation is a common technique that has been proven to benefit the training of machine learning models in general and deep architectures in particular; either speeding up convergence or acting as a regularizer, thus avoiding overfitting and increasing generalization capabilities[28].

It typically consist of applying a set of transformations in either data or feature spaces, or even both. The most common augmentations are performed in the data space. That kind of augmentation generates new samples by applying transformations to the already existing data. There are many transformations that can be applied: translation, rotation, warping, scaling, color space shifts, crops, etc. The goal of those transformations is to generate more samples to create a larger dataset, preventing overfitting and presumably regularizing the model, balance the classes within that database, and even synthetically produce new samples that are more representative for the use case or task at hand.

Augmentations are specially helpful for small datasets, and have proven their efficacy with a long track of success stories. For instance, in [29], a

dataset of 1500 portrait images is augmented synthesizing four new scales $(0.6, 0.8, 1.2, 1.5)$, four new rotations $(-45, -22, 22, 45)$, and four gamma variations $(0.5, 0.8, 1.2, 1.5)$ to generate a new dataset of 19000 training images. That process allowed them to raise the accuracy of their system for portrait segmentation from 73.09 to 94.20 Intersection over Union (IoU) when including that augmented dataset for fine-tuning.

3. Materials and Methods

In this section we present the main content of this paper: an in-depth review of existing datasets and methods together with an evaluation of their quality, relevance, and impact in the field. Both of them are presented according to a well thought taxonomy of the research completed in the area.

Datasets were sampled from a wide variety of application scenarios and subdivided according to their data representation (2D, 2.5D, or 3D) since that is the first factor to take into account when choosing a dataset. Apart from that we categorize them according to other application-specific factors which are later described in detail. The datasets were selected mainly by using a criteria of relevance (popularity by means of citation reports, usage as benchmarking tools, and scientific quality enforced by top-tier venues and journals) and usefulness for the community (novel data or application scenarios). We also focused on taking into account their scale and possibilities for training deep architectures.

The same criteria of relevance was applied when selecting methods. Certain outdated methods were still selected for completeness when describing improved versions of those or even newer ones. Others were also included due to their importance in laying the fundamentals for future research. The rest of the methods were selected due to the impact of their main contributions either by creating a new research direction (e.g., instance segmentation, sequence processing, 3D segmentation) or pushing forward the limits of a certain aspect (e.g., accuracy, efficiency, or training capabilities). More details about the followed taxonomy can be found later in the corresponding section.

3.1. Datasets and Challenges

Two kinds of readers are expected for this type of review: either they are initiating themselves in the problem, or either they are experienced enough and they are just looking for the most recent advances made by other researchers in the last few years. Although the second kind is usually aware

Table 1: Popular large-scale segmentation datasets.

Name and Reference	Purpose	Year	Classes	Data	Resolution	Sequence	Synthetic/Real	Samples (training)	Samples (validation)	Samples (test)
PASCAL VOC 2012 Segmentation [30]	Generic	2012	21	2D	Variable	✗	R	1464	1449	Private
PASCAL-Context [31]	Generic	2014	540 (59)	2D	Variable	✗	R	10103	N/A	9637
PASCAL-Part [32]	Generic-Part	2014	20	2D	Variable	✗	R	10103	N/A	9637
SBD [33]	Generic	2011	21	2D	Variable	✗	R	8498	2857	N/A
Microsoft COCO [34]	Generic	2014	+80	2D	Variable	✗	R	82783	40504	81434
SYNTHIA [35]	Urban (Driving)	2016	11	2D	960×720	✗	S	13407	N/A	N/A
Cityscapes (fine) [36]	Urban	2015	30 (8)	2D	2048×1024	✓	R	2975	500	1525
Cityscapes (coarse) [36]	Urban	2015	30 (8)	2D	2048×1024	✓	R	22973	500	N/A
CamVid [37]	Urban (Driving)	2009	32	2D	960×720	✓	R	701	N/A	N/A
CamVid-Sturgess [38]	Urban (Driving)	2009	11	2D	960×720	✓	R	367	100	233
KITTI-Layout [39][40]	Urban/Driving	2012	3	2D	Variable	✗	R	323	N/A	N/A
KITTI-Ros [41]	Urban/Driving	2015	11	2D	Variable	✗	R	170	N/A	46
KITTI-Zhang [42]	Urban/Driving	2015	10	2D/3D	1226×370	✗	R	140	N/A	112
Stanford background [43]	Outdoor	2009	8	2D	320×240	✗	R	725	N/A	N/A
SiftFlow [44]	Outdoor	2011	33	2D	256×256	✗	R	2688	N/A	N/A
Youtube-Objects-Jain [45]	Objects	2014	10	2D	480×360	✓	R	10167	N/A	N/A
Adobe's Portrait Segmentation [29]	Portrait	2016	2	2D	600×800	✗	R	1500	300	N/A
MINC [46]	Materials	2015	23	2D	Variable	✗	R	7061	2500	5000
DAVIS [47][48]	Generic	2016	4	2D	480p	✓	R	4219	2023	2180
NYUDv2 [49]	Indoor	2012	40	2.5D	480×640	✗	R	795	654	N/A
SUN3D [50]	Indoor	2013	–	2.5D	640×480	✓	R	19640	N/A	N/A
SUNRGBD [51]	Indoor	2015	37	2.5D	Variable	✗	R	2666	2619	5050
RGB-D Object Dataset[52]	Household objects	2011	51	2.5D	640×480	✓	R	207920	N/A	N/A
ShapeNet Part[53]	Object/Part	2016	16/50	3D	N/A	✗	S	31,963	N/A	N/A
Stanford 2D-3D-S[54]	Indoor	2017	13	2D/2.5D/3D	1080×1080	✓	R	70469	N/A	N/A
3D Mesh [55]	Object/Part	2009	19	3D	N/A	✗	S	380	N/A	N/A
Sydney Urban Objects Dataset[56]	Urban (Objects)	2013	26	3D	N/A	✗	R	41	N/A	N/A
Large-Scale Point Cloud Classification Benchmark[57]	Urban/Nature	2016	8	3D	N/A	✗	R	15	N/A	15

of two of the most important aspects to know before starting to research in this problem, it is critical for newcomers to get a grasp of what are the top-quality datasets and challenges. Therefore, the purpose of this section is to help novel scientists get to speed, providing them with a brief summary of datasets that might suit their needs as well as data augmentation and pre-processing tips. Nevertheless, it can also be useful for hardened researchers who want to review the fundamentals or maybe discover new information.

In the following lines we describe the most popular large-scale datasets currently in use for semantic segmentation. All datasets listed here provide appropriate pixel-wise or point-wise labels. The list is structured into three parts according to the nature of the data: 2D or plain RGB datasets, 2.5D or RGB-Depth (RGB-D) ones, and pure volumetric or 3D databases. Table 1 shows a summarized view, gathering all the described datasets and providing useful information such as their purpose, number of classes, data format, and training/validation/testing splits.

3.1.1. 2D Datasets

Throughout the years, semantic segmentation has been mostly focused on two-dimensional images. For that reason, 2D datasets are the most abundant ones. In this section we describe the most popular 2D large-scale datasets for semantic segmentation, considering 2D any dataset that contains any kind of two-dimensional representations such as gray-scale or Red Green Blue (RGB) images.

- **PASCAL Visual Object Classes (VOC)**[30][1]: this challenge consists of a ground-truth annotated dataset of images and five different competitions: classification, detection, segmentation, action classification, and person layout. The segmentation one is specially interesting since its goal is to predict the object class of each pixel for each test image. There are 21 classes categorized into vehicles, household, animals, and other: aeroplane, bicycle, boat, bus, car, motorbike, train, bottle, chair, dining table, potted plant, sofa, TV/monitor, bird, cat, cow, dog, horse, sheep, and person. Background is also considered if the pixel does not belong to any of those classes. The dataset is divided into two subsets: training and validation with 1464 and 1449 images respectively. The test set is private for the challenge. This dataset is arguably the most popular for semantic segmentation so almost every remarkable method in the literature is being submitted to its performance evaluation server to validate against their private test set. Methods can be trained either using only the dataset or either using additional information. Furthermore, its leaderboard is public and can be consulted online[2].

- **PASCAL Context**[31][3]: this dataset is an extension of the PASCAL VOC 2010 detection challenge which contains pixel-wise labels for all training images (10103). It contains a total of 540 classes – including the original 20 classes plus background from PASCAL VOC segmentation – divided into three categories (objects, stuff, and hybrids). Despite the large number of categories, only the 59 most frequent are remarkable. Since its classes follow a power law distribution, there are many of them which are too sparse throughout the dataset. In this regard, this subset of 59 classes is usually selected to conduct studies on this dataset, relabeling the rest of them as background.

- **PASCAL Part**[32][4]: this database is an extension of the PASCAL VOC 2010 detection challenge which goes beyond that task to provide

[1] http://host.robots.ox.ac.uk/pascal/VOC/voc2012/

[2] http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=6

[3] http://www.cs.stanford.edu/~roozbeh/pascal-context/

[4] http://www.stat.ucla.edu/~xianjie.chen/pascal_part_dataset/pascal_part.html

per-pixel segmentation masks for each part of the objects (or at least silhouette annotation if the object does not have a consistent set of parts). The original classes of PASCAL VOC are kept, but their parts are introduced, e.g., bicycle is now decomposed into back wheel, chain wheel, front wheel, handlebar, headlight, and saddle. It contains labels for all training and validation images from PASCAL VOC as well as for the 9637 testing images.

- **Semantic Boundaries Dataset (SBD)**[33][5]: this dataset is an extended version of the aforementioned PASCAL VOC which provides semantic segmentation ground truth for those images that were not labelled in VOC. It contains annotations for 11355 images from PASCAL VOC 2011. Those annotations provide both category-level and instance-level information, apart from boundaries for each object. Since the images are obtained from the whole PASCAL VOC challenge (not only from the segmentation one), the training and validation splits diverge. In fact, SBD provides its own training (8498 images) and validation (2857 images) splits. Due to its increased amount of training data, this dataset is often used as a substitute for PASCAL VOC for deep learning.

- **Microsoft Common Objects in Context (COCO)**[34][6]: is another image recognition, segmentation, and captioning large-scale dataset. It features various challenges, being the detection one the most relevant for this field since one of its parts is focused on segmentation. That challenge, which features more than 80 classes, provides more than 82783 images for training, 40504 for validation, and its test set consist of more than 80000 images. In particular, the test set is divided into four different subsets or splits: test-dev (20000 images) for additional validation, debugging, test-standard (20000 images) is the default test data for the competition and the one used to compare state-of-the-art methods, test-challenge (20000 images) is the split used for the challenge when submitting to the evaluation server, and test-reserve (20000 images) is a split used to protect against possible overfitting in the challenge (if a method is suspected to have made too many

[5] `http://home.bharathh.info/home/sbd`
[6] `http://mscoco.org/`

submissions or trained on the test data, its results will be compared with the reserve split). Its popularity and importance has ramped up since its appearance thanks to its large scale. In fact, the results of the challenge are presented yearly on a joint workshop at the European Conference on Computer Vision (ECCV)[7] together with ImageNet's ones.

- **SYNTHetic Collection of Imagery and Annotations (SYNTHIA)**[35][8]: is a large-scale collection of photo-realistic renderings of a virtual city, semantically segmented, whose purpose is scene understanding in the context of driving or urban scenarios.The dataset provides fine-grained pixel-level annotations for 11 classes (void, sky, building, road, sidewalk, fence, vegetation, pole, car, sign, pedestrian, and cyclist). It features 13407 training images from rendered video streams. It is also characterized by its diversity in terms of scenes (towns, cities, highways), dynamic objects, seasons, and weather.

- **Cityscapes**[36][9]: is a large-scale database which focuses on semantic understanding of urban street scenes. It provides semantic, instance-wise, and dense pixel annotations for 30 classes grouped into 8 categories (flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void). The dataset consist of around 5000 fine annotated images and 20000 coarse annotated ones. Data was captured in 50 cities during several months, daytimes, and good weather conditions. It was originally recorded as video so the frames were manually selected to have the following features: large number of dynamic objects, varying scene layout, and varying background.

- **CamVid**[58][37][10]: is a road/driving scene understanding database which was originally captured as five video sequences with a 960×720 resolution camera mounted on the dashboard of a car. Those sequences were sampled (four of them at 1 fps and one at 15 fps) adding up to 701 frames. Those stills were manually annotated with 32 classes: void, building, wall, tree, vegetation, fence, sidewalk, parking block,

[7] http://image-net.org/challenges/ilsvrc+coco2016
[8] http://synthia-dataset.net/
[9] https://www.cityscapes-dataset.com/
[10] http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/

column/pole, traffic cone, bridge, sign, miscellaneous text, traffic light, sky, tunnel, archway, road, road shoulder, lane markings (driving), lane markings (non-driving), animal, pedestrian, child, cart luggage, bicyclist, motorcycle, car, SUV/pickup/truck, truck/bus, train, and other moving object. It is important to remark the partition introduced by Sturgess *et al.*[38] which divided the dataset into 367/100/233 training, validation, and testing images respectively. That partition makes use of a subset of class labels: building, tree, sky, car, sign, road, pedestrian, fence, pole, sidewalk, and bicyclist.

- **KITTI**[59]: is one of the most popular datasets for use in mobile robotics and autonomous driving. It consists of hours of traffic scenarios recorded with a variety of sensor modalities, including high-resolution RGB, grayscale stereo cameras, and a 3D laser scanner. Despite its popularity, the dataset itself does not contain ground truth for semantic segmentation. However, various researchers have manually annotated parts of the dataset to fit their necessities. Álvarez *et al.*[39][40] generated ground truth for 323 images from the road detection challenge with three classes: road, vertical, and sky. Zhang *et al.*[42] annotated 252 (140 for training and 112 for testing) acquisitions – RGB and Velodyne scans – from the tracking challenge for ten object categories: building, sky, road, vegetation, sidewalk, car, pedestrian, cyclist, sign/pole, and fence. Ros *et al.* [41] labeled 170 training images and 46 testing images (from the visual odometry challenge) with 11 classes: building, tree, sky, car, sign, road, pedestrian, fence, pole, sidewalk, and bicyclist.

- **Youtube-Objects**[60] is a database of videos collected from YouTube which contain objects from ten PASCAL VOC classes: aeroplane, bird, boat, car, cat, cow, dog, horse, motorbike, and train. That database does not contain pixel-wise annotations but Jain *et al.*[45] manually annotated a subset of 126 sequences. They took every 10th frame from those sequences and generated semantic labels. That totals 10167 annotated frames at 480×360 pixels resolution.

- **Adobe's Portrait Segmentation**[29][11]: this is a dataset of 800×600

[11] `http://xiaoyongshen.me/webpage_portrait/index.html`

pixels portrait images collected from Flickr, mainly captured with mobile front-facing cameras. The database consist of 1500 training images and 300 reserved for testing, both sets are fully binary annotated: person or background. The images were labeled in a semi-automatic way: first a face detector was run on each image to crop them to 600×800 pixels and then persons were manually annotated using Photoshop quick selection. This dataset is remarkable due to its specific purpose which makes it suitable for person in foreground segmentation applications.

- **Materials in Context (MINC)**[46]: this work is a dataset for patch material classification and full scene material segmentation. The dataset provides segment annotations for 23 categories: wood, painted, fabric, glass, metal, tile, sky, foliage, polished stone, carpet, leather, mirror, brick, water, other, plastic, skin, stone, ceramic, hair, food, paper, and wallpaper. It contains 7061 labeled material segmentations for training, 5000 for test, and 2500 for validation. The main source for these images is the OpenSurfaces dataset [61], which was augmented using other sources of imagery such as Flickr or Houzz. For that reason, image resolution for this dataset varies. On average, image resolution is approximately 800×500 or 500×800.

- **Densely-Annotated VIdeo Segmentation (DAVIS)**[47][48][12]: this challenge is purposed for video object segmentation. Its dataset is composed by 50 high-definition sequences which add up to 4219 and 2023 frames for training and validation respectively. Frame resolution varies across sequences but all of them were downsampled to 480p for the challenge. Pixel-wise annotations are provided for each frame for four different categories: human, animal, vehicle, and object. Another feature from this dataset is the presence of at least one target foreground object in each sequence. In addition, it is designed not to have many different objects with significant motion. For those scenes which do have more than one target foreground object from the same class, they provide separated ground truth for each one of them to allow instance segmentation.

- **Stanford background**[43][13]: dataset with outdoor scene images im-

[12] http://davischallenge.org/index.html
[13] http://dags.stanford.edu/data/iccv09Data.tar.gz

ported from existing public datasets: LabelMe, MSRC, PASCAL VOC and Geometric Context. The dataset contains 715 images (size of 320×240 pixels) with at least one foreground object and having the horizon position within the image. The dataset is pixel-wise annotated (horizon location, pixel semantic class, pixel geometric class and image region) for evaluating methods for semantic scene understanding.

- **SiftFlow** [44]: contains 2688 fully annotated images which are a subset of the LabelMe database [62]. Most of the images are based on 8 different outdoor scenes including streets, mountains, fields, beaches and buildings. Images are 256×256 belonging to one of the 33 semantic classes. Unlabeled pixels, or pixels labeled as a different semantic class are treated as unlabeled.

3.1.2. 2.5D Datasets

With the advent of low-cost range scanners, datasets including not only RGB information but also depth maps are gaining popularity and usage. In this section, we review the most well-known 2.5D databases which include that kind of depth data.

- **NYUDv2** [49][14]: this database consists of 1449 indoor RGB-D images captured with a Microsoft Kinect device. It provides per-pixel dense labeling (category and instance levels) which were coalesced into 40 indoor object classes by Gupta *et al.*[63] for both training (795 images) and testing (654) splits. This dataset is specially remarkable due to its indoor nature, this makes it really useful for certain robotic tasks at home. However, its relatively small scale with regard to other existing datasets hinders its application for deep learning architectures.

- **SUN3D** [50][15]: similar to the NYUDv2, this dataset contains a large-scale RGB-D video database, with 8 annotated sequences. Each frame has a semantic segmentation of the objects in the scene and information about the camera pose. It is still in progress and it will be composed by 415 sequences captured in 254 different spaces, in 41 different buildings. Moreover, some places have been captured multiple times at different moments of the day.

[14] `http://cs.nyu.edu/~silberman/projects/indoor_scene_seg_sup.html`
[15] `http://sun3d.cs.princeton.edu/`

- **SUNRGBD** [51][16]: captured with four RGB-D sensors, this dataset contains 10000 RGB-D images, at a similar scale as PASCAL VOC. It contains images from NYU depth v2 [49], Berkeley B3DO [64], and SUN3D [50]. The whole dataset is densely annotated, including polygons, bounding boxes with orientation as well as a 3D room layout and category, being suitable for scene understanding tasks.

- **The Object Segmentation Database (OSD)** [65][17] this database has been designed for segmenting unknown objects from generic scenes even under partial occlusions. This dataset contains 111 entries, and provides depth image and color images together withper-pixel annotations for each one to evaluate object segmentation approaches. However, the dataset does not differentiate the category of different objects so its classes are reduced to a binary set of objects and not objects.

- **RGB-D Object Dataset**[52][18]: this dataset is composed by video sequences of 300 common household objects organized in 51 categories arranged using WordNet hypernym-hyponym relationships. The dataset has been recorded using a Kinect style 3D camera that records synchronized and aligned 640×480 RGB and depth images at $30Hz$. For each frame, the dataset provides, the RGB-D and depth images, a cropped ones containing the object, the location and a mask with per-pixel annotation. Moreover, each object has been placed on a turntable, providing isolated video sequences around 360 degrees. For the validation process, 22 annotated video sequences of natural indoor scenes containing the objects are provided.

3.1.3. 3D Datasets

Pure three-dimensional databases are scarce, this kind of datasets usually provide Computer Aided Design (CAD) meshes or other volumetric representations, such as point clouds. Generating large-scale 3D datasets for segmentation is costly and difficult, and not many deep learning methods are able to process that kind of data as it is. For those reasons, 3D datasets are not quite popular at the moment. In spite of that fact, we describe the most promising ones for the task at hand.

[16] `http://rgbd.cs.princeton.edu/`
[17] `http://www.acin.tuwien.ac.at/?id=289`
[18] `http://rgbd-dataset.cs.washington.edu/`

- **ShapeNet Part**[53][19]: is a subset of the ShapeNet[66] repository which focuses on fine-grained 3D object segmentation. It contains $31,693$ meshes sampled from 16 categories of the original dataset (airplane, earphone, cap, motorbike, bag, mug, laptop, table, guitar, knife, rocket, lamp, chair, pistol, car, and skateboard). Each shape class is labeled with two to five parts (totalling 50 object parts across the whole dataset), e.g., each shape from the airplane class is labeled with wings, body, tail, and engine. Ground-truth labels are provided on points sampled from the meshes.

- **Stanford 2D-3D-S**[54][20]: is a multi-modal and large-scale indoor spaces dataset extending the Stanford 3D Semantic Parsing work [67]. It provides a variety of registered modalities – 2D (RGB), 2.5D (depth maps and surface normals), and 3D (meshes and point clouds) – with semantic annotations. The database is composed of $70,496$ full high-definition RGB images (1080×1080 resolution) along with their corresponding depth maps, surface normals, meshes, and point clouds with semantic annotations (per-pixel and per-point). That data were captured in six indoor areas from three different educational and office buildings. That makes a total of 271 rooms and approximately 700 million points annotated with labels from 13 categories: ceiling, floor, wall, column, beam, window, door, table, chair, bookcase, sofa, board, and clutter.

- **A Benchmark for 3D Mesh Segmentation**[55][21]: this benchmark is composed by 380 meshes classified in 19 categories (human, cup, glasses, airplane, ant, chair, octopus, table, teddy, hand, plier, fish, bird, armadillo, bust, mech, bearing, vase, fourleg). Each mesh has been manually segmented into functional parts, the main goal is to provide a sample distribution over "how humans decompose each mesh into functional parts".

- **Sydney Urban Objects Dataset**[56][22]: this dataset contains a variety of common urban road objects scanned with a Velodyne HDK-64E

[19] `http://cs.stanford.edu/~ericyi/project_page/part_annotation/`
[20] `http://buildingparser.stanford.edu`
[21] `http://segeval.cs.princeton.edu/`
[22] `http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml`

Table 2: Summary of semantic segmentation methods based on deep learning.

		Targets								
Name and Reference	**Architecture**	**Accuracy**	**Efficiency**	**Training**	**Instance**	**Sequences**	**Multi-modal**	**3D**	**Source Code**	**Contribution(s)**
Fully Convolutional Network[68]	VGG-16(FCN)	⋆	⋆	⋆	✗	✗	✗	✗	✓	Forerunner
U-Net[69]	Fully CNN, 4 downsampling/upsampling steps	⋆⋆	⋆⋆	⋆	✗	✗	✗	✗	✓	Data augmentation, Skip-layer, Patch wise training/inference
SegNet[70]	VGG-16 + Decoder	⋆⋆⋆	⋆⋆	⋆	✗	✗	✗	✗	✓	Encoder-decoder
Bayesian SegNet[71]	SegNet	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Uncertainty modeling
DeepLab[72][73]	VGG-16/ResNet-101	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Standalone CRF, atrous convolutions
MINC-CNN [46]	GoogLeNet(FCN)	⋆	⋆	⋆	✗	✗	✗	✗	✓	Patchwise CNN, Standalone CRF
CRFasRNN[74]	FCN-8s	⋆	⋆⋆	⋆⋆⋆	✗	✗	✗	✗	✓	CRF reformulated as RNN
Dilation[75]	VGG-16	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Dilated convolutions
ENet[76]	ENet bottleneck	⋆⋆	⋆⋆⋆	⋆	✗	✗	✗	✗	✓	Bottleneck module for efficiency
Multi-scale-CNN-Raj[77]	VGG-16(FCN)	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✗	Multi-scale architecture
Multi-scale-CNN-Eigen[78]	Custom	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Multi-scale sequential refinement
Multi-scale-CNN-Raj[79]	Multi-scale-CNN-Eigen	⋆⋆⋆	⋆	⋆	✗	✗	⋆⋆	✗	✗	Multi-scale coarse-to-fine refinement
Multi-scale-CNN-Bian[80]	FCN	⋆⋆	⋆	⋆⋆	✗	✗	✗	✗	✗	Independently trained multi-scale FCNs
ParseNet[81]	VGG-16	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Global context feature fusion
PSPNet [82]	ResNet50	⋆⋆⋆	⋆	⋆⋆	✗	✗	✗	✗	✓	Image context modelling, training optimization strategy for ResNet
ReSeg[83]	VGG-16 + ReNet	⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Extension of ReNet to semantic segmentation
LSTM-CF[84]	Fast R-CNN + DeepMask	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Fusion of contextual information from multiple sources
2D-LSTM[85]	MDRNN	⋆⋆	⋆⋆	⋆	✗	✗	✗	✗	✗	Image context modelling
rCNN[86]	MDRNN	⋆⋆⋆	⋆⋆	⋆	✗	✗	✗	✗	✓	Different input sizes, image context
DAG-RNN[87]	Elman network	⋆⋆⋆	⋆	⋆	✗	✗	✗	✗	✓	Graph image structure for context modelling
SDS[10]	R-CNN + Box CNN	⋆⋆⋆	⋆	⋆	⋆⋆	✗	✗	✗	✓	Simultaneous detection and segmentation
DeepMask[88]	VGG-A	⋆⋆⋆	⋆	⋆	⋆⋆	✗	✗	✗	✓	Proposals generation for segmentation
SharpMask[89]	DeepMask	⋆⋆⋆	⋆	⋆	⋆⋆⋆	✗	✗	✗	✓	Top-down refinement module
MultiPathNet[90]	Fast R-CNN + DeepMask	⋆⋆⋆	⋆	⋆	⋆⋆⋆	✗	✗	✗	✓	Multi path information flow through network
Huang-3DCNN[91]	Own 3DCNN	⋆	⋆	⋆	✗	✗	✗	⋆⋆⋆	✗	3DCNN for voxelized point clouds
PointNet[92]	Own MLP-based	⋆⋆	⋆	⋆	✗	✗	✗	⋆⋆⋆	✓	Segmentation of unordered point sets
PointNet++[93]	Own PointNet-based	⋆⋆	⋆	⋆	✗	✗	✗	⋆⋆⋆	✓	Improve PointNet by capturing local information
Dynamic Graph CNN (DGCNN)[94]	Own EdgeConv	⋆⋆	⋆	⋆	✗	✗	✗	⋆⋆⋆	✗	EdgeConvolution module for point clouds as graphs
Clockwork Convnet[95]	FCN	⋆⋆	⋆⋆	⋆	✗	⋆⋆⋆	✗	✗	✓	Clockwork scheduling for sequences
3DCNN-Zhang	Own 3DCNN	⋆⋆	⋆	⋆	✗	⋆⋆⋆	✗	✗	✓	3D convolutions and graph cut for sequences
End2End Vox2Vox[96]	C3D	⋆⋆	⋆	⋆	✗	⋆⋆⋆	✗	✗	✗	3D convolutions/deconvolutions for sequences
SegmPred [97]	Own multi-scale net	⋆⋆	⋆	⋆	✗	⋆⋆⋆	✗	✗	✓	Predicting future frames in the space of semantic segmentation

LIDAR. There are 631 individual scans (point clouds) of objects across classes of vehicles, pedestrians, signs and trees. The interesting point of this dataset is that, for each object, apart from the individual scan, a full 360-degrees annotated scan is provided.

- **Large-Scale Point Cloud Classification Benchmark** [57][23]: this benchmark provides manually annotated 3D point clouds of diverse natural and urban scenes: churches, streets, railroad tracks, squares, villages, soccer fields, castles among others. This dataset features statically captured point clouds with very fine details and density. It contains 15 large-scale point clouds for training and another 15 for testing. Its scale can be grasped by the fact that it totals more than one billion labelled points.

3.2. Methods

The relentless success of deep learning techniques in various high-level computer vision tasks – in particular, supervised approaches such as Convolutional Neural Networks (CNNs) for image classification or object detection [14][15][17] – motivated researchers to explore the capabilities of such networks for pixel-level labelling problems like semantic segmentation. The key advantage of these deep learning techniques, which gives them an edge over

[23] `http://www.semantic3d.net/`

traditional methods, is the ability to learn appropriate feature representations for the problem at hand, e.g., pixel labelling on a particular dataset, in an end-to-end fashion instead of using hand-crafted features that require domain expertise, effort, and often too much fine-tuning to make them work on a particular scenario.

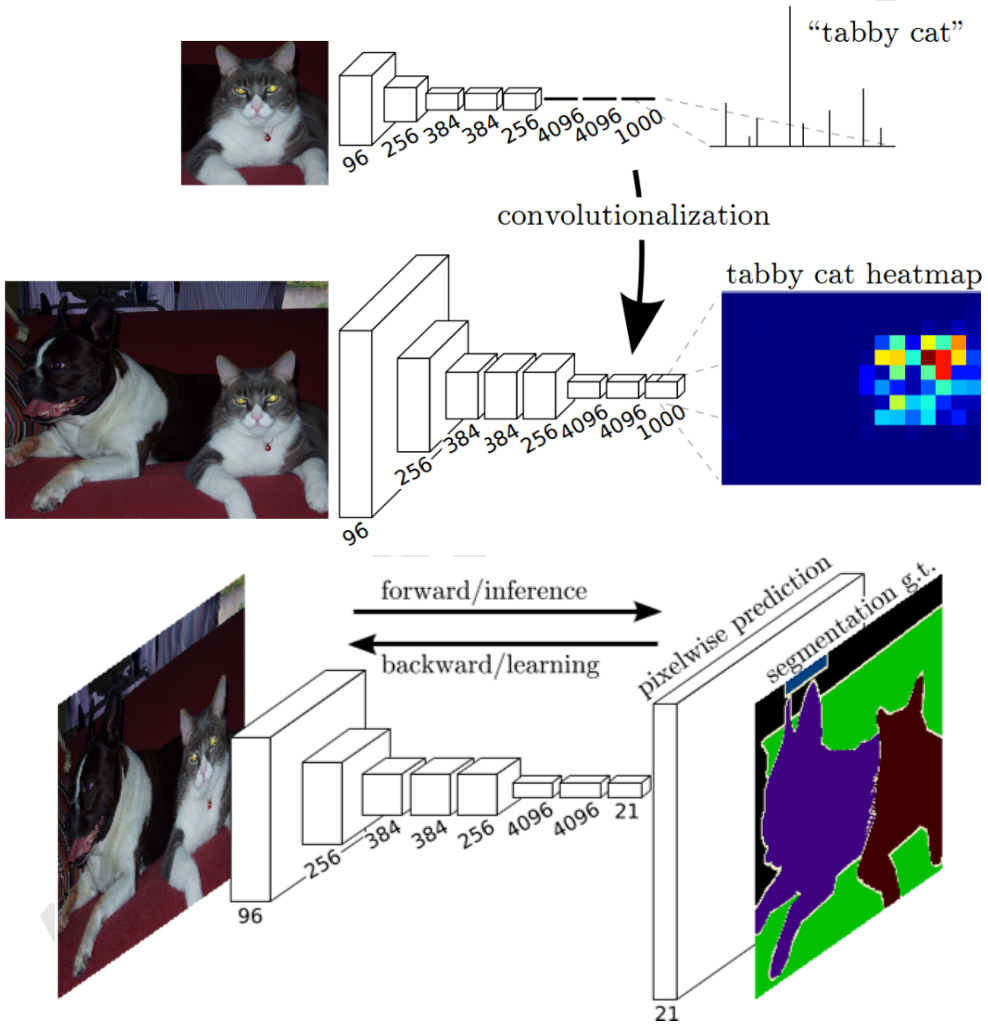


Figure 7: Fully Convolutional Network figure by Long *et al.*[68]. Transforming a classification-purposed CNN to produce spatial heatmaps by replacing fully connected layers with convolutional ones. Including a deconvolution layer for upsampling allows dense inference and learning for per-pixel labeling.

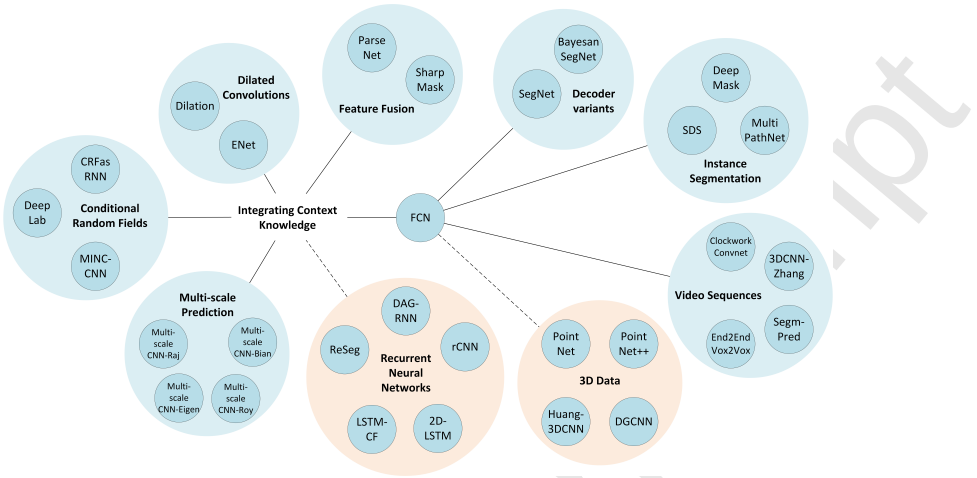


Figure 8: Visualization of the reviewed methods.

Currently, the most successful state-of-the-art deep learning techniques for semantic segmentation stem from a common forerunner: the *Fully Convolutional Network (FCN)* by Long *et al.*[68]. The insight of that approach was to take advantage of existing CNNs as powerful visual models that are able to learn hierarchies of features. They transformed those existing and well-known classification models – AlexNet[14], VGG (16-layer net)[15], GoogLeNet[17], and ResNet [18] – into fully convolutional ones by replacing the fully connected layers with convolutional ones to output spatial maps instead of classification scores. Those maps are upsampled using fractionally strided convolutions (also named deconvolutions [98][99]) to produce dense per-pixel labeled outputs. This work is considered a milestone since it showed how CNNs can be trained end-to-end for this problem, efficiently learning how to make dense predictions for semantic segmentation with inputs of arbitrary sizes. This approach achieved a significant improvement in segmentation accuracy over traditional methods on standard datasets like PASCAL VOC, while preserving efficiency at inference. For all those reasons, and other significant contributions, the FCN is the cornerstone of deep learning applied to semantic segmentation. The convolutionalization process is shown in Figure 7.

Despite the power and flexibility of the FCN model, it still lacks various features which hinder its application to certain problems and situations: its inherent spatial invariance does not take into account useful global context

information, no instance-awareness is present by default, efficiency is still far from real-time execution at high resolutions, and it is not completely suited for unstructured data such as 3D point clouds or models. Those problems will be reviewed in this section, as well as the state-of-the-art solutions that have been proposed in the literature to overcome those hurdles. Table 2 provides a summary of that review. It shows all reviewed methods (sorted by appearance order in the section), their base architecture, their main contribution, and a classification depending on the target of the work: accuracy, efficiency, training simplicity, sequence processing, multi-modal inputs, and 3D data. Each target is graded from one to three stars ($\star$) depending on how much focus puts the work on it, and a mark ($\boldsymbol{\times}$) if that issue is not addressed. In addition, Figure 8 shows a graph of the reviewed methods for the sake of visualization.

3.2.1. Decoder Variants

Apart from the FCN architecture, other variants were developed to transform a network whose purpose was classification to make it suitable for segmentation. Arguably, FCN-based architectures are more popular and successful, but other alternatives are also remarkable. In general terms, all of them take a network for classification, such as VGG-16, and remove its fully connected layers. This part of the new segmentation network often receives the name of encoder and produce low-resolution image representations or feature maps. The problem lies on learning to decode or map those low-resolution images to pixel-wise predictions for segmentation. This part is named decoder and it is usually the divergence point in this kind of architectures.

SegNet[70] is a clear example of this divergence (see Figure 9). The decoder stage of SegNet is composed by a set of upsampling and convolution layers which are at last followed by a softmax classifier to predict pixel-wise labels for an output which has the same resolution as the input image. Each upsampling layer in the decoder stage corresponds to a max-pooling one in the encoder part. Those layers upsample feature maps using the max-pooling indices from their corresponding feature maps in the encoder phase. The upsampled maps are then convolved with a set of trainable filter banks to produce dense feature maps. When the feature maps have been restored to the original resolution, they are fed to the softmax classifier to produce the final segmentation.

On the other hand, FCN-based architectures make use of learnable decon-

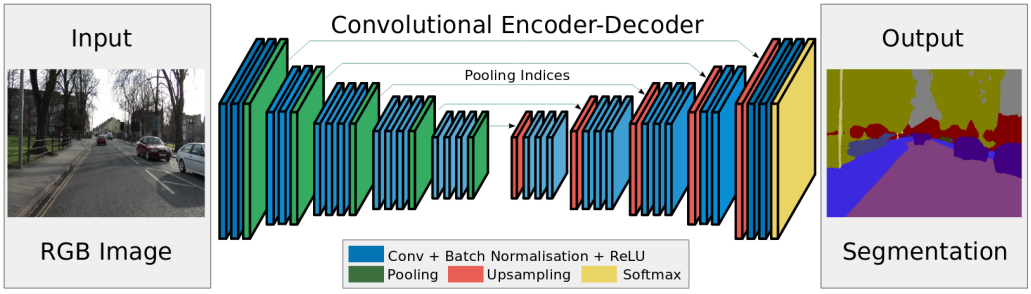


Figure 9: SegNet architecture with an encoder and a decoder followed by a softmax classifier for pixel-wise classification. Figure extracted from [70].

volution filters to upsample feature maps. After that, the upsampled feature maps are added element-wise to the corresponding feature map generated by the convolution layer in the encoder part. Figure 10 shows a comparison of both approaches.

U-Net [69] is another example of a Fully Convolutional Neural Network that has been used for image segmentation. It was initially proposed for biomedical image segmentation, but in the last years it has also been successfully used in other applications, such as aerial imagery [100] and regular foreground/background segmentation problems. It consists of a contracting path (downsampling) which captures context and a symmetric expanding path (upsampling) that enables precise localization. The architecture also has skip connections that allow the decoder at each stage to learn relevant features from the contracting path.

3.2.2. Integrating Context Knowledge

Semantic segmentation is a problem that requires the integration of information from various spatial scales. It also implies balancing local and global information. On the one hand, fine-grained or local information is crucial to achieve good pixel-level accuracy. On the other hand, it is also important to integrate information from the global context of the image to be able to resolve local ambiguities.

Vanilla CNNs struggle with this balance. Pooling layers, which allow the networks to achieve some degree of spatial invariance and keep computational cost at bay, dispose of the global context information. Even purely CNNs – without pooling layers – are limited since the receptive field of their units can only grow linearly with the number of layers.

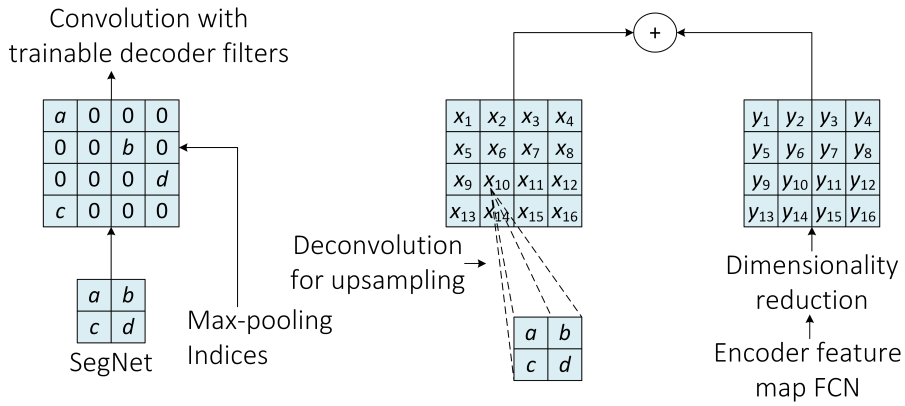
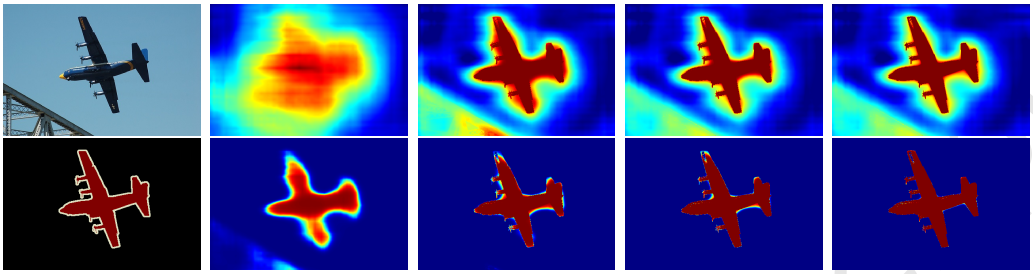


Figure 10: Comparison of SegNet (left) and FCN (right) decoders. While SegNet uses max-pooling indices from the corresponding encoder stage to upsample, FCN learns deconvolution filters to upsample (adding the corresponding feature map from the encoder stage). Figure reproduced from [70].

Many approaches can be taken to make CNNs aware of that global information: refinement as a post-processing step with Conditional Random Fields (CRFs), dilated convolutions, multi-scale aggregation, or even defer the context modeling to another kind of deep networks such as RNNs.

Conditional Random Fields. As we mentioned before, the inherent invariance to spatial transformations of CNN architectures limits the very same spatial accuracy for segmentation tasks. One possible and common approach to refine the output of a segmentation system and boost its ability to capture fine-grained details is to apply a post-processing stage using a Conditional Random Field (CRF). CRFs enable the combination of low-level image information – such as the interactions between pixels [101][102] – with the output of multi-class inference systems that produce per-pixel class scores. That combination is especially important to capture long-range dependencies, which CNNs fail to consider, and fine local details.

The DeepLab models [72][73] make use of the fully connected pairwise CRF by Krähenbühl and Koltun[103][104] as a separated post-processing step in their pipeline to refine the segmentation result. It models each pixel as a node in the field and employs one pairwise term for each pair of pixels no matter how far they lie (this model is known as dense or fully connected factor graph). By using this model, both short and long-range interactions are taken into account, rendering the system able to recover detailed structures



(a) GT (b) CNNout (c) CRFit1 (d) CRFit2 (e) CRFit10

Figure 11: CRF refinement per iteration as shown by the authors of DeepLab[72]. The first row shows the score maps (inputs before the softmax function) and the second one shows the belief maps (output of the softmax function).

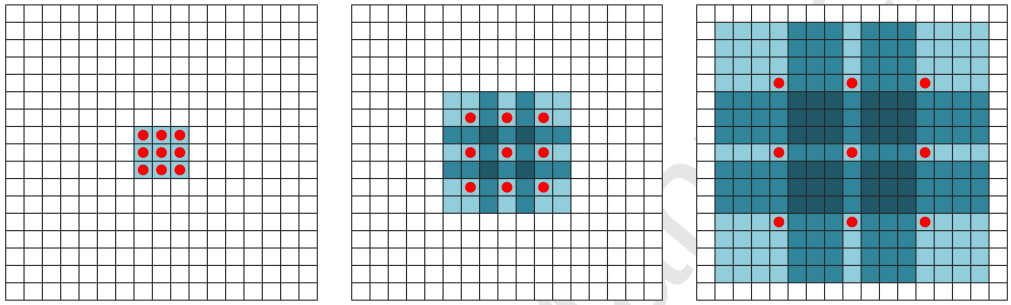
in the segmentation that were lost due to the spatial invariance of the CNN. Despite the fact that usually fully connected models are inefficient, this model can be efficiently approximated via probabilistic inference. Figure 11 shows the effect of this CRF-based post-processing on the score and belief maps produced by the DeepLab model.

The material recognition in the wild network by Bell *et al.*[46] makes use of various CNNs trained to identify patches in the MINC database. Those CNNs are used on a sliding window fashion to classify those patches. Their weights are transferred to the same networks converted into FCNs by adding the corresponding upsampling layers. The outputs are averaged to generate a probability map. At last, the same CRF from DeepLab, but discretely optimized, is applied to predict and refine the material at every pixel.

Another significant work applying a CRF to refine the segmentation of a FCN is the CRFasRNN by Zheng *et al.*[74]. The main contribution of that work is the reformulation of the dense CRF with pairwise potentials as an integral part of the network. By unrolling the mean-field inference steps as RNNs, they make it possible to fully integrate the CRF with a FCN and train the whole network end-to-end. This work demonstrates the reformulation of CRFs as RNNs to form a part of a deep network, in contrast with Pinheiro et al. [86] which employed RNNs to model large spatial dependencies.

Dilated Convolutions. Dilated convolutions, also named *à-trous* convolutions, are a generalization of Kronecker-factored convolutional filters [105] which support exponentially expanding receptive fields without losing resolution. In other words, dilated convolutions are regular ones that make use of up-

sampled filters. The dilation rate l controls that upsampling factor. As shown in Figure 12, stacking l-dilated convolution makes the receptive fields grow exponentially while the number of parameters for the filters keeps a linear growth. This means that dilated convolutions allow efficient dense feature extraction on any arbitrary resolution. As a side note, it is important to remark that typical convolutions are just 1-dilated convolutions.



(a) 1-dilated (b) 2-dilated (c) 3-dilated

Figure 12: As shown in [75], dilated convolution filters with various dilation rates: (a) 1-dilated convolutions in which each unit has a 3×3 receptive fields, (b) 2-dilated ones with 7×7 receptive fields, and (c) 3-dilated convolutions with 15×15 receptive fields.

In practice, it is equivalent to dilating the filter before doing the usual convolution. That means expanding its size, according to the dilation rate, while filling the empty elements with zeros. In other words, the filter weights are matched to distant elements which are not adjacent if the dilation rate is greater than one. Figure 13 shows examples of dilated filters.

The most important works that make use of dilated convolutions are the multi-scale context aggregation module by Yu *et al.*[75], the already mentioned DeepLab (its improved version)[73], and the real-time network ENet[76]. All of them use combinations of dilated convolutions with increasing dilation rates to have wider receptive fields with no additional cost and without overly downsampling the feature maps. Those works also show a common trend: dilated convolutions are tightly coupled to multi-scale context aggregation as we will explain in the following section.

Multi-scale Prediction. Another possible way to deal with context knowledge integration is the use of multi-scale predictions. Almost every single parameter of a CNN affects the scale of the generated feature maps. In other words, the very same architecture will have an impact on the number of pixels of

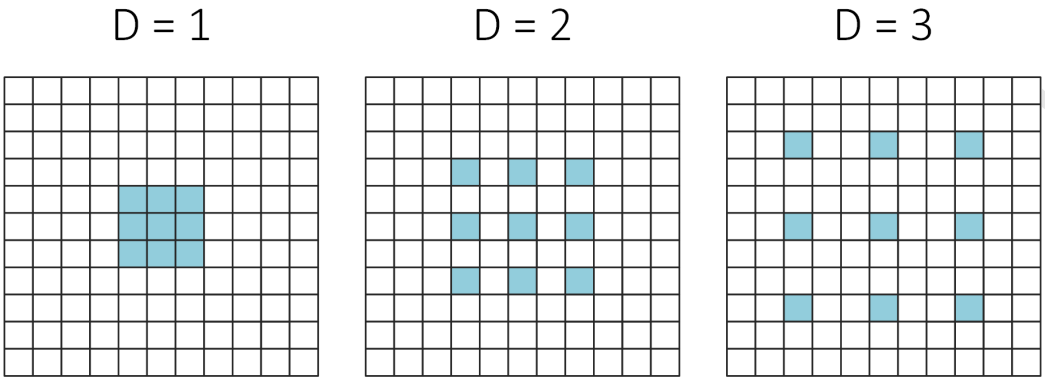


Figure 13: Filter elements (green) matched to input elements when using 3×3 dilated convolutions with various dilation rates. From left to right: $1, 2$, and 3.

the input image which correspond to a pixel of the feature map. This means that the filters will implicitly learn to detect features at specific scales (presumably with certain invariance degree). Furthermore, those parameters are usually tightly coupled to the problem at hand, making it difficult for the models to generalize to different scales. One possible way to overcome that obstacle is to use multi-scale networks which generally make use of multiple networks that target different scales and then merge the predictions to produce a single output.

Raj *et al.*[77] propose a multi-scale version of a fully convolutional VGG-16. That network has two paths, one that processes the input at the original resolution and another one which doubles it. The first path goes through a shallow convolutional network. The second one goes through the fully convolutional VGG-16 and an extra convolutional layer. The result of that second path is upsampled and combined with the result of the first path. That concatenated output then goes through another set of convolutional layers to generate the final output. As a result, the network becomes more robust to scale variations.

Roy *et al.*[79] take a different approach using a network composed by four multi-scale CNNs. Those four networks have the same architecture introduced by Eigen *et al.* [78]. One of those networks is devoted to finding semantic labels for the scene. That network extracts features from a progressively coarse-to-fine sequence of scales (see Figure 14).

Another remarkable work is the network proposed by Bian *et al.*[80]. That

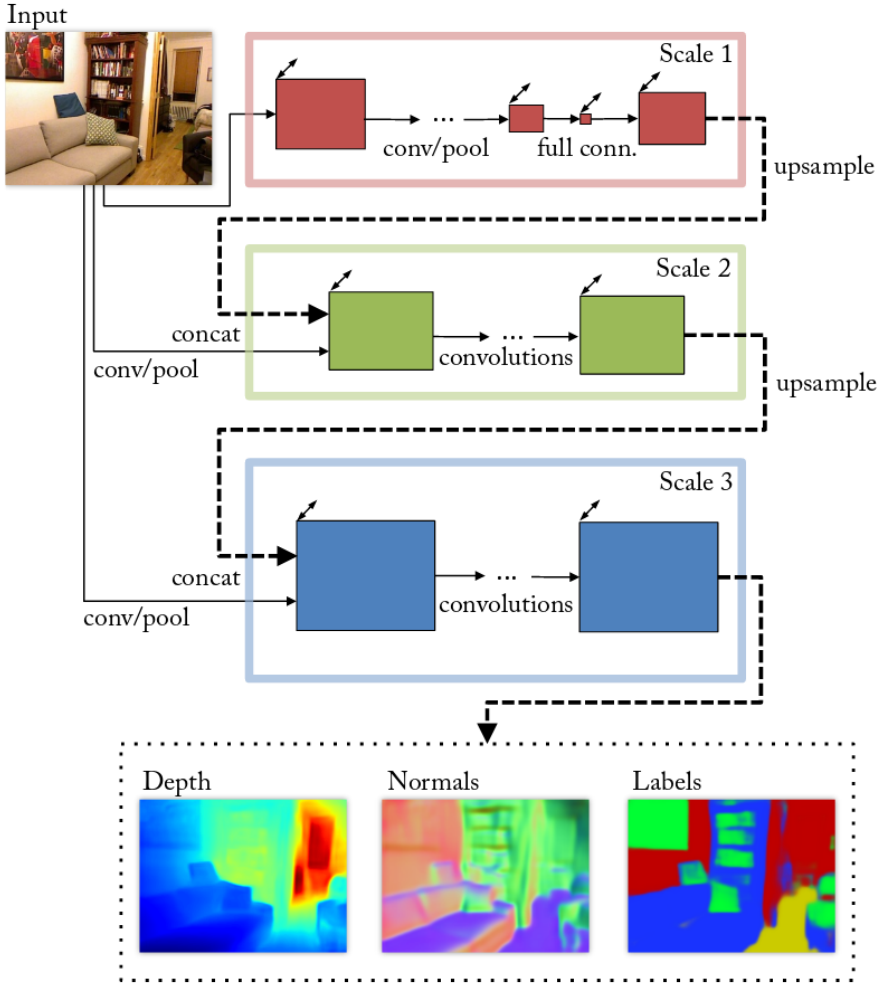


Figure 14: Multi-scale CNN architecture proposed by Eigen *et al.*[78]. The network progressively refines the output using a sequence of scales to estimate depth, normals, and also perform semantic segmentation over an RGB input. Figure extracted from [78].

network is a composition of n FCNs which operate at different scales. The features extracted from the networks are fused together (after the necessary upsampling with an appropriate padding) and then they go through an additional convolutional layer to produce the final segmentation. The main contribution of this architecture is the two-stage learning process which involves, first, training each network independently, then the networks are combined and the last layer is fine-tuned. This multi-scale model allows to

add an arbitrary number of newly trained networks in an efficient manner.

Feature Fusion. Another way of adding context information to a fully convolutional architecture for segmentation is feature fusion. This technique consists of merging a global feature (extracted from a previous layer in a network) with a more local feature map extracted from a subsequent layer. Common architectures such as the original FCN make use of skip connections to perform a late fusion by combining the feature maps extracted from different layers (see Figure 15).

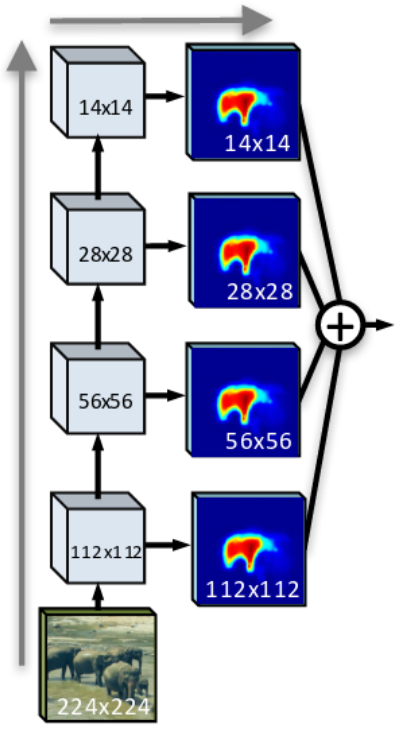


Figure 15: Skip-connection-like architecture, which performs late fusion of feature maps as if making independent predictions for each layer and merging the results. Figure extracted from [89].

Another approach is performing early fusion. This approach is taken by ParseNet[81] in their context module. The global feature is unpooled to the same spatial size as the local feature and then they are concatenated to generate a combined feature that is used in the next layer or to learn a classifier. Figure 16 shows a representation of that process.

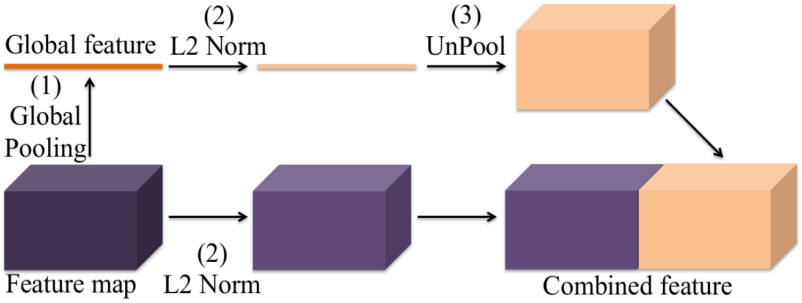


Figure 16: ParseNet context module overview in which a global feature (from a previous layer) is combined with the feature of the next layer to add context information. Figure extracted from [81].

This feature fusion idea was continued by Pinheiro *et al.* in their SharpMask network [89], which introduced a progressive refinement module to incorporate features from the previous layer to the next in a top-down architecture. This work will be reviewed later since it is mainly focused on instance segmentation.

In contrast to the pooling operation performed by ParseNet to incorporate global features and in addition to dilated FCNs [72][75], pyramid pooling empirically demonstrates the capability of global feature extraction by different-region-based context aggregation [82]. Figure 17 shows Pyramid Scene Parsing Networks (PSPNets) [24] which provide a pyramid parsing module focused into feature fusion at four different pyramid scales in order to embed global contexts from complex scenes. Pyramid levels and size of each level can be arbitrarily modified. The better performance of PSPNet facing FCNs-based models lies to: (1) the lack of ability in collecting contextual information, (2) the absence of category relationships and (3) not using sub-regions. This approach achieves state-of-the-art performance on various datasets.

Recurrent Neural Networks. As we noticed, CNNs have been successfully applied to multi-dimensional data, such as images. Nevertheless, these networks rely on hand specified kernels limiting the architecture to local contexts. Taking advantage of its topological structure, Recurrent Neural Networks have been successfully applied for modeling short- and long-temporal sequences.

[24] `https://github.com/hszhao/PSPNet`

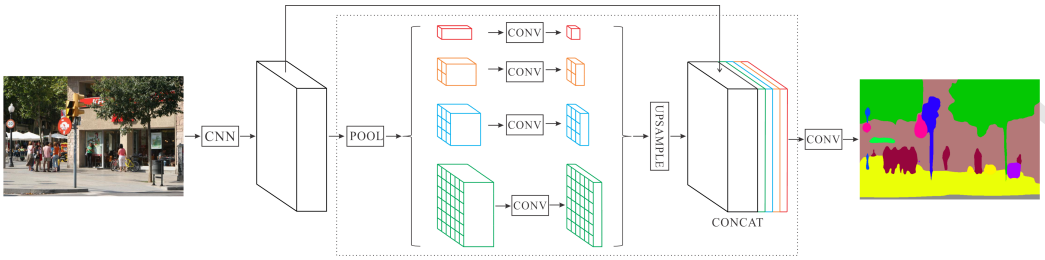


Figure 17: PSPNet architecture. Initial feature maps (b) are extracted from input images (a) by using a pretrained ResNet [18] alongside dilated network strategy. Pyramid pooling module (c) covers from the whole, half of to small regions of the image. Finally, initial feature map is concatenated with pooling module output and applying a convolution layer final predicted maps (d) are generated. Figure extracted from [82].

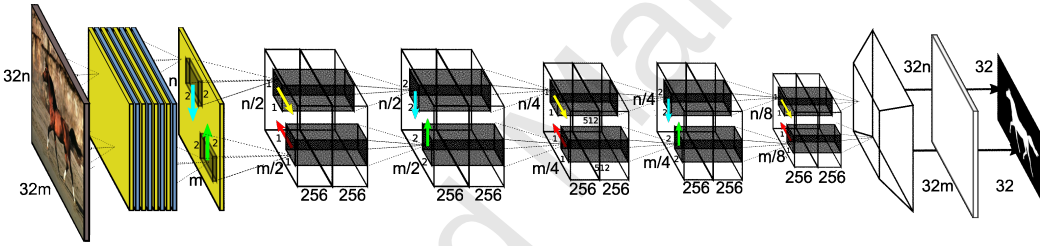


Figure 18: Representation of ReSeg network. VGG-16 convolutional layers are represented by the blue and yellow first layers. The rest of the architecture is based on the ReNet approach with fine-tuning purposes. Figure extracted from [83].

In this way and by linking together pixel-level and local information, RNNs are able to successfully model global contexts and improve semantic segmentation. However, one important issue is the lack of a natural sequential structure in images and the focus of standard vanilla RNNs architectures on one-dimensional inputs.

Based on ReNet model for image classification Visin *et al.*[20] proposed an architecture for semantic segmentation called ReSeg [83] represented in Figure 18. In this approach, the input image is processed with the first layers of the VGG-16 network [15], feeding the resulting feature maps into one or more ReNet layers for fine-tuning. Finally, feature maps are resized using upsampling layers based on transposed convolutions. In this approach Gated Recurrent Units (GRUs) have been used as they strike a good performance balance regarding memory usage and computational power. Vanilla RNNs have problems modeling long-term dependencies mainly due to the vanishing

gradients problem. Several derived models such as Long Short-Term Memory (LSTM) networks [106] and GRUs [107] are the state-of-art in this field to avoid such problem.

Inspired on the same ReNet architecture, a novel Long Short-Term Memorized Context Fusion (LSTM-CF) model for scene labeling was proposed by [108]. In this approach, they use two different data sources: RGB and depth. The RGB pipeline relies on a variant of the DeepLab architecture [32] concatenating features at three different scales to enrich feature representation (inspired by [109]). The global context is modeled vertically over both, depth and photometric data sources, concluding with a horizontal fusion in both direction over these vertical contexts.

As we noticed, modeling image global contexts is related to 2D recurrent approaches by unfolding vertically and horizontally the network over the input images. Based on the same idea, Byeon et al. [85] purposed a simple 2D LSTM-based architecture in which the input image is divided into non-overlapping windows which are fed into four separate LSTMs memory blocks. This work emphasizes its low computational complexity on a single-core CPU and the model simplicity.

Another approach for capturing global information relies on using bigger input windows in order to model larger contexts. Nevertheless, this reduces images resolution and also implies several problems regarding to window overlapping. However, Pinheiro et al. [86] introduced Recurrent Convolutional Neural Networks (rCNNs) which recurrently train with different input window sizes taking into account previous predictions by using a different input window sizes. In this way, predicted labels are automatically smoothed increasing the performance.

Undirected cyclic graphs (UCGs) were also adopted to model image contexts for semantic segmentation [87]. Nevertheless, RNNs are not directly applicable to UCG and the solution is decomposing it into several directed graphs (DAGs). In this approach, images are processed by three different layers: image feature map produced by CNN, model image contextual dependencies with DAG-RNNs, and deconvolution layer for upsampling feature maps. This work demonstrates how RNNs can be used together with graphs to successfully model long-range contextual dependencies, overcoming state-of-the-art approaches in terms of performance.

3.2.3. Instance Segmentation

Instance segmentation is considered the next step after semantic segmentation and at the same time the most challenging problem in comparison with the rest of low-level pixel segmentation techniques. Its main purpose is to represent objects of the same class splitted into different instances. The automation of this process is not straightforward, thus the number of instances is initially unknown and the evaluation of performed predictions is not pixel-wise such as in semantic segmentation. Consequently, this problem remains partially unsolved but the interest in this field is motivated by its potential applicability. Instance labeling provides us extra information for reasoning about occlusion situations, also counting the number of elements belonging to the same class and for detecting a particular object for grasping in robotics tasks, among many other applications.

For this purpose, Hariharan et al. [10] proposed a Simultaneous Detection and Segmentation (SDS) method in order to improve performance over already existing works. Their pipeline uses, firstly, a bottom-up hierarchical image segmentation and object candidate generation process called Multi-scale COmbinatorial Grouping (MCG) [110] to obtain region proposals. For each region, features are extracted by using an adapted version of the Region-CNN (R-CNN) [111], which is fine-tuned using bounding boxes provided by the MCG method instead of selective search and also alongside region foreground features. Then, each region proposal is classified by using a linear Support Vector Machine (SVM) on top of the CNN features. Finally, and for refinement purposes, Non-Maximum Suppression (NMS) is applied to the previous proposals.

Later, Pinheiro et al. [88] presented DeepMask model, an object proposal approach based on a single ConvNet. This model predicts a segmentation mask for an input patch and the likelihood of this patch for containing an object. The two tasks are learned jointly and computed by a single network, sharing most of the layers except last ones which are task-specific.

Based on the DeepMask architecture as a starting point due to its effectiveness, the same authors presented a novel architecture for object instance segmentation implementing a top-down refinement process [89] and achieving a better performance in terms of accuracy and speed. The goal of this process is to efficiently merge low-level features with high-level semantic information from upper network layers. The process consisted in different refinement modules stacked together (one module per pooling layer), with the purpose

of inverting pooling effect by generating a new upsampled object encoding. Figure 19 shows the refinement module in SharpMask.

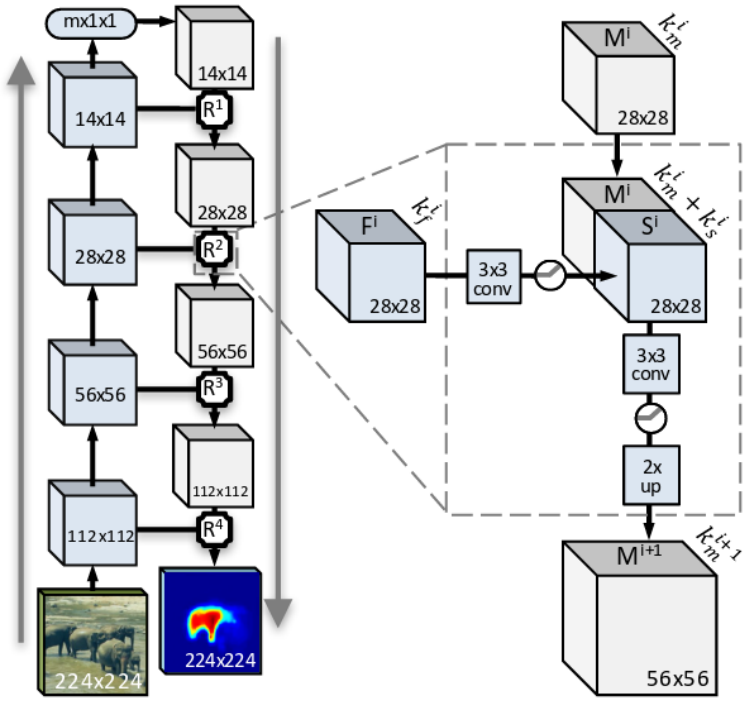


Figure 19: SharpMask's top-down architecture with progressive refinement using their signature modules. That refinement merges spatially rich information from lower-level features with high-level semantic cues encoded in upper layers. Figure extracted from [88].

Another approach, based on Fast R-CNN as a starting point and using DeepMask object proposals instead of Selective Search was presented by Zagoruyko et al [90]. This combined system called MultiPath classifier, improved performance over COCO dataset and supposed three modifications to Fast R-CNN: improving localization with an integral loss, provide context by using foveal regions and finally skip connections to give multi-scale features to the network. The system achieved a 66% improvement over the baseline Fast R-CNN.

As we have seen, most of the methods mentioned above rely on existing object detectors limiting in this way model performance. Even so, instance

segmentation process remains an unresolved research problem and the mentioned works are only a small part of this challenging research topic.

3.2.4. RGB-D Data

As we noticed, a significant amount of work has been done in semantic segmentation by using photometric data. Nevertheless, the use of structural information was spurred on with the advent of low-cost RGB-D sensors which provide useful geometric cues extracted from depth information. Several works focused on RGB-D scene segmentation have reported an improvement in the fine-grained labeling precision by using depth information and not only photometric data. Using depth information for segmentation is considered more challenging because of the unpredictable variation of scene illumination alongside incomplete representation of objects due to complex occlusions. However, various works have successfully made use of depth information to increase accuracy.

The use of depth images with approaches focused on photometric data is not straightforward. Depth data needs to be encoded with three channels at each pixel as if it was an RGB images. Different techniques such as Horizontal Height Angle (HHA) [11] are used for encoding the depth into three channels as follows: horizontal disparity, height above ground, and the angle between local surface normal and the inferred gravity direction. In this way, we can input depth images to models designed for RGB data and improve in this way the performance by learning new features from structural information. Several works such as [108] are based on this encoding technique.

In the literature, related to methods that use RGB-D data, we can also find some works that leverage a multi-view approach to improve existing single-view works.

Zeng *et al.*[112] present an object segmentation approach that leverages multi-view RGB-D data and deep learning techniques. RGB-D images captured from each viewpoint are fed to a FCN network which returns a 40-class probability for each pixel in each image. Segmentation labels are threshold by using three times the standard deviation above the mean probability across all views. Moreover, in this work, multiple networks for feature extraction were trained (AlexNet [14] and VGG-16 [15]), evaluating the benefits of using depth information. They found that adding depth did not yield any major improvements in segmentation performance, which could be caused by noise in the depth information. The described approach was presented during the 2016 Amazon Picking Challenge. This work is a minor contribution towards

multi-view deep learning systems since RGB images are independently fed to a FCN network.

Ma *et al.*[113] propose a novel approach for object-class segmentation using a multi-view deep learning technique. Multiple views are obtained from a moving RGB-D camera. During the training stage, camera trajectory is obtained using an RGB-D SLAM technique, then RGB-D images are warped into ground-truth annotated frames in order to enforce multi-view consistency for training. The proposed approach is based on FuseNet[114], which combines RGB and depth images for semantic segmentation, and improves the original work by adding multi-scale loss minimization.

3.2.5. 3D Data

3D geometric data such as point clouds or polygonal meshes are useful representations thanks to their additional dimension which provides methods with rich spatial information that is intuitively useful for segmentation. However, the vast majority of successful deep learning segmentation architectures – CNNs in particular – are not originally engineered to deal with unstructured or irregular inputs such as the aforementioned ones. In order to enable weight sharing and other optimizations in convolutional architectures, most researchers have resorted to 3D voxel grids or projections to transform unstructured and unordered point clouds or meshes into regular representations before feeding them to the networks. For instance, Huang *et al.*[91] take a point cloud and parse it through a dense voxel grid, generating a set of occupancy voxels which are used as input to a 3D CNN to produce one label per voxel. They then map back the labels to the point cloud. Although this approach has been applied successfully, it has some disadvantages like quantization, loss of spatial information, and unnecessarily large representations. For that reason, various researchers have focused their efforts on creating deep architectures that are able to directly consume unstructured 3D point sets or meshes.

PointNet[92] is a pioneering work which presents a deep neural network that takes raw point clouds as input, providing a unified architecture for both classification and segmentation. Figure 20 shows that two-part network which is able to consume unordered point sets in 3D.

As we can observe, PointNet is a deep network architecture that stands out of the crowd due to the fact that it is based on fully connected layers instead of convolutional ones. The architecture features two subnetworks: one for classification and another for segmentation. The classification sub-

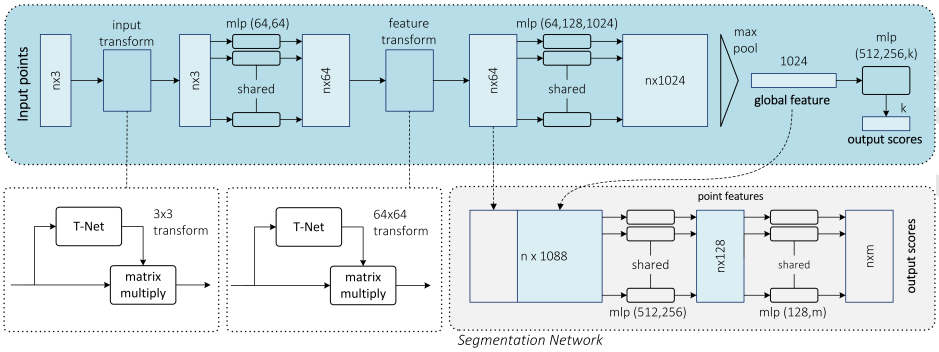


Figure 20: The PointNet unified architecture for point cloud classification and segmentation. Figure reproduced from [92].

network takes a point cloud and applies a set of transforms and Multi Layer Perceptrons (MLPs) to generate features which are then aggregated using max-pooling to generate a global feature which describes the original input cloud. That global feature is classified by another MLP to produce output scores for each class. The segmentation subnetwork concatenates the global feature with the per-point features extracted by the classification network and applies another two MLPs to generate features and produce output scores for each point. As an improvement, the same authors proposed PointNet++ [93] which is able to capture local features with increasing context scales by using metric space distances.

Another remarkable work to deal with point clouds as graphs and directly apply convolutions without any kind of discretization is the DGCNN [94]. This novel architecture proposes a new neural network module, namely *EdgeConv*, which operates directly over the point cloud and incorporates several important properties (local neighborhood information, it can be stacked, and it is able to capture long-distance properties). That module is easily pluggable into existing architectures and has been proven to capture and exploit fine-grained and global properties of point clouds expressed as graphs.

3.2.6. Video Sequences

As we have observed, there has been a significant progress in single-image segmentation. However, when dealing with image sequences, many systems rely on the naïve application of the very same algorithms in a frame-by-frame manner. This approach works, often producing remarkable results. Nevertheless, applying those methods frame by frame is usually non-viable due to

computational cost. In addition, those methods completely ignore temporal continuity and coherence cues which might help increase the accuracy of the system while reducing its execution time.

Arguably, the most remarkable work in this regard is the clockwork FCN by Shelhamer *et al.*[95]. This network is an adaptation of a FCN to make use of temporal cues in video to decrease inference time while preserving accuracy. The clockwork approach relies on the following insight: feature velocity – the temporal rate of change of features in the network – across frames varies from layer to layer so that features from shallow layers change faster than deep ones. Under that assumption, layers can be grouped into stages, processing them at different update rates depending on their depth. By doing this, deep features can be persisted over frames thanks to their semantic stability, thus saving inference time. Figure 21 shows the network architecture of the clockwork FCN.

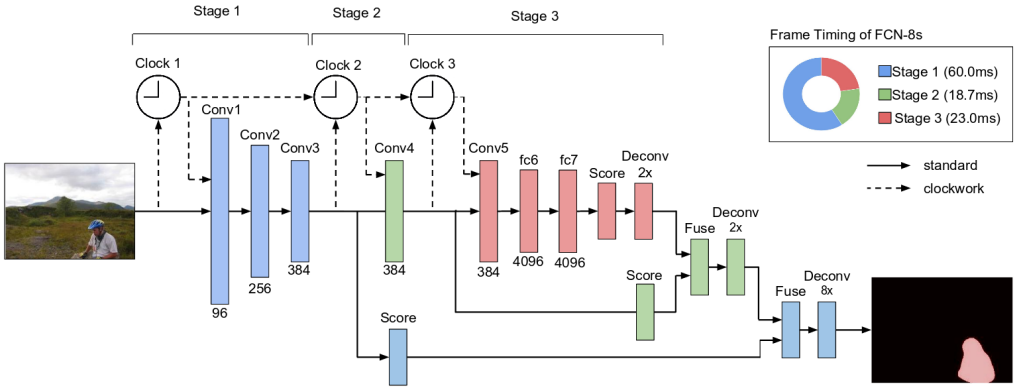


Figure 21: The clockwork FCN with three stages and their corresponding clock rates. Figure extracted from [95].

It is important to remark that the authors propose two kinds of update rates: fixed and adaptive. The fixed schedule just sets a constant time frame for recomputing the features for each stage of the network. The adaptive schedule fires each clock on a data-driven manner, e.g., depending on the amount of motion or semantic change. Figure 22 shows an example of this adaptive scheduling.

Zhang *et al.*[115] took a different approach and made use of a 3DCNN, which was originally created for learning features from volumes, to learn hierarchical spatio-temporal features from multi-channel inputs such as video

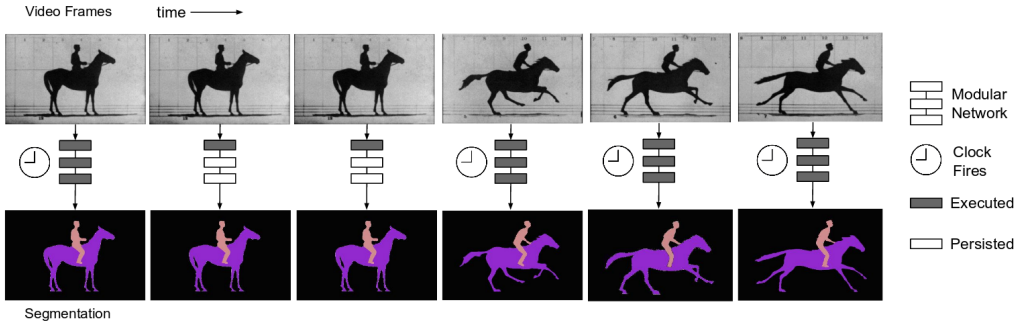
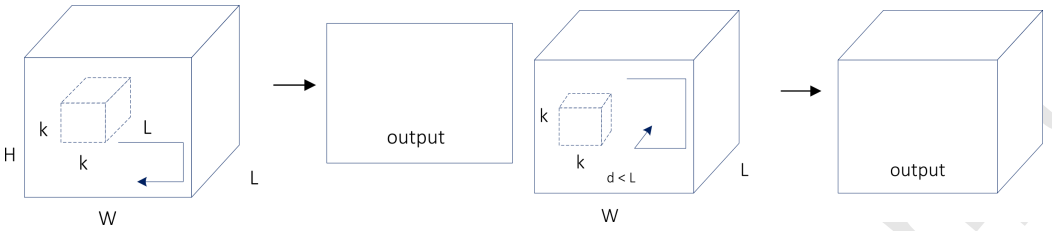


Figure 22: Adaptive clockwork method proposed by Shelhamer *et al.*[95]. Extracted features persists during static frames while they are recomputed for dynamic ones. Figure extracted from [95].

clips. In parallel, they over-segment the input clip into supervoxels. Then they use that supervoxel graph and embed the learned features in it. The final segmentation is obtained by applying graph-cut[116] on the supervoxel graph.

Another remarkable method, which builds on the idea of using 3D convolutions, is the deep end-to-end voxel-to-voxel prediction system by Tran *et al.*[96]. In that work, they make use of the Convolutional 3D (C3D) network introduced by themselves on a previous work [117], and extend it for semantic segmentation by adding deconvolutional layers at the end. Their system works by splitting the input into clips of 16 frames, performing predictions for each clip separately. Its main contribution is the use of 3D convolutions. Those convolutions make use of three-dimensional filters which are suitable for spatio-temporal feature learning across multiple channels, in this case frames. Figure 23 shows the difference between 2D and 3D convolutions applied to multi-channel inputs, proving the usefulness of the 3D ones for video segmentation.

Novel approaches such as SegmPred model proposed by Luc et al. [97] are able to predict semantic segmentation maps of not yet observed video frames in the future. This model consists in a two-scale architecture which is trained in both, adversarial and non-adversarial ways in order to deal with blurred predicted results. Model inputs have been previously per-frame annotated and consists in the softmax output layer pre-activations. Model performance drops when predicting more than a few frames in the future. However, this approach is able to model the object dynamics on the semantic segmentation



(a) 2D Convolution (b) 3D Convolution

Figure 23: Difference between 2D and 3D convolutions applied on a set of frames. (a) 2D convolutions use the same weights for the whole depth of the stack of frames (multiple channels) and results in a single image. (b) 3D convolutions use 3D filters and produce a 3D volume as a result of the convolution, thus preserving temporal information of the frame stack.

maps, which remains an open challenge for current computer vision systems.

3.2.7. Loss functions for semantic segmentation

Particular choices of the loss function can strongly affect deep models accuracy and their learning process. For training image and patch-based classification models, we find that the vast majority of research works and applications simply use a cross entropy loss function. However, for regression models, we find that L1 and L2 losses are the most common functions. In this work, we are targeting a slightly different problem, pixel-based classification. An important issue to consider when moving from image or patch-based classification to pixel-based classification is that the last one is more prone to suffer the data imbalance problem. Reviewing existing works in the literature we found that categorical cross entropy and the dice similarity coefficient are the main loss functions used for training semantic segmentation models. We also found different variations for these methods, such as combining both in a weighted manner. For example, ReLayNet [118] is trained to optimize a joint loss function comprising of weighted categorical cross-entropy and Dice similarity score. Another common variation that has been used in existing works is the use of a weighted scheme for the categorical cross entropy itself. Ronneberger et al [69] precompute a weight map for the categorical cross entropy loss. It is also very common when we have a single background class and few foreground classes, in that case, the data imbalance problem becomes overwhelming. Several previous approaches resorted to loss functions based on sample re-weighting where foreground regions are given more importance than background ones during learning. However, other approaches only op-

timize the dice similarity score instead [119]. More recently, a new variant has been presented, it is a loss strategy coined as Focal Loss [120], that adds a factor γ to the standard cross entropy criterion. It basically reduces the relative loss for well-classified pixels and puts more focus on hard, misclassified ones. A similar strategy can be applied for refining object boundaries, which often contain misclassified pixels.

Reviewing loss functions used in existing architectures for semantic segmentation, we find that the FCN [68] was trained using a per-pixel multinomial logistic loss and was validated with the standard metric of mean pixel intersection over union. The more recent DeepLab [73] architecture, similarly to FCN, uses as a loss function the sum of cross-entropy terms for each spatial position in the CNN output map. In this case, the original input is subsampled by 8 and all positions are equally weighted in the loss function. PSPNet [82] also uses the cross-entropy loss function, but in this work, two loss functions are optimized during the training of the network. Apart from the main branch using cross-entropy loss to train the final classifier, another classifier is applied after the fourth stage. The optimization of these two functions is performed in a weighted manner, applying different weights to each loss function and therefore balancing the auxiliary loss function. SegmPred [97] model relies on Gradient Difference Loss (GDL), designed to sharpen results by penalizing high-frequency mismatches such as errors along the object boundaries. Using GDL alongside L1 loss function, SegmPred model results significantly improved by sharpening its outputs.

In general, the particular choice of the loss function will depend on the type, amount of classes and samples that your dataset contains for each class. Moreover, it is important to consider the number of pixels that are hard to classify pixels in your dataset (compared to the total amount in the ground truth segmentation masks). Based on those aspects, some of the previously presented approaches may help you facing the data imbalance problem and therefore, provide you with a more accurate model for semantic segmentation.

4. Discussion

In the previous section we reviewed the existing methods from a literary and qualitative point of view, i.e., we did not take any quantitative result into account. In this Section we are going to discuss the very same methods from a numeric standpoint. First of all, we will describe the most popular

evaluation metrics that can be used to measure the performance of semantic segmentation systems from three aspects: execution time, memory footprint, and accuracy. Next, we will gather the results of the methods on the most representative datasets using the previously described metrics. After that, we will summarize and draw conclusions about those results. At last, we enumerate possible future research lines that we consider significant for the field.

4.1. Evaluation Metrics

For a segmentation system to be useful and actually produce a significant contribution to the field, its performance must be evaluated with rigor. In addition, that evaluation must be performed using standard and well-known metrics that enable fair comparisons with existing methods. Furthermore, many aspects must be evaluated to assert the validity and usefulness of a system: execution time, memory footprint, and accuracy. Depending on the purpose or the context of the system, some metrics might be of more importance than others, i.e., accuracy may be expendable up to a certain point in favor of execution speed for a real-time application. Nevertheless, for the sake of scientific rigor it is of utmost importance to provide all the possible metrics for a proposed method.

4.1.1. Execution Time

Speed or runtime is an extremely valuable metric since the vast majority of systems must meet hard requirements on how much time can they spend on the inference pass. In some cases it might be useful to know the time needed for training the system, but it is usually not that significant, unless it is exaggeratedly slow, since it is an offline process. In any case, providing exact timings for the methods can be seen as meaningless since they are extremely dependant on the hardware and the backend implementation, rendering some comparisons pointless.

However, for the sake of reproducibility and in order to help fellow researchers, it is useful to provide timings with a thorough description of the hardware in which the system was executed on, as well as the conditions for the benchmark. If done properly, that can help others estimate if the method is useful or not for the application as well as perform fair comparisons under the same conditions to check which are the fastest methods.

4.1.2. Memory Footprint

Memory usage is another important factor for segmentation methods. Although it is arguably less constraining than execution time – scaling memory capacity is usually feasible – it can also be a limiting element. In some situations, such as onboard chips for robotic platforms, memory is not as abundant as in a high-performance server. Even high-end Graphics Processing Units (GPUs), which are commonly used to accelerate deep networks, do not pack a copious amount of memory. In this regard, and considering the same implementation-dependent aspects as with runtime, documenting the peak and average memory footprint of a method with a complete description of the execution conditions can be extraordinarily helpful.

4.1.3. Accuracy

Many evaluation criteria have been proposed and are frequently used to assess the accuracy of any kind of technique for semantic segmentation. Those metrics are usually variations on pixel accuracy and IoU. We report the most popular metrics for semantic segmentation that are currently used to measure how per-pixel labeling methods perform on this task. For the sake of the explanation, we remark the following notation details: we assume a total of $k+1$ classes (from L_0 to L_k including a void class or background) and p_{ij} is the amount of pixels of class i inferred to belong to class j. In other words, p_{ii} represents the number of true positives, while p_{ij} and p_{ji} are usually interpreted as false positives and false negatives respectively (although either of them can be the sum of both false positives and false negatives)..

- **Pixel Accuracy (PA)**: it is the simplest metric, simply computing a ratio between the amount of properly classified pixels and the total number of them.

$$PA = \frac{\sum_{i=0}^{k} p_{ii}}{\sum_{i=0}^{k}\sum_{j=0}^{k} p_{ij}}$$

- **Mean Pixel Accuracy (MPA)**: a slightly improved PA in which the ratio of correct pixels is computed in a per-class basis and then

averaged over the total number of classes.

$$MPA = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ii}}{\sum_{j=0}^{k} p_{ij}}$$

- **Mean Intersection over Union (MIoU)**: this is the standard metric for segmentation purposes. It computes a ratio between the intersection and the union of two sets, in our case the ground truth and our predicted segmentation. That ratio can be reformulated as the number of true positives (intersection) over the sum of true positives, false negatives, and false positives (union). That IoU is computed on a per-class basis and then averaged.

$$MIoU = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ji} - p_{ii}}$$

- **Frequency Weighted Intersection over Union (FWIoU)**: it is an improved over the raw MIoU which weights each class importance depending on their appearance frequency.

$$FWIoU = \frac{1}{\sum_{i=0}^{k} \sum_{j=0}^{k} p_{ij}} \sum_{i=0}^{k} \frac{\sum_{j=0}^{k} p_{ij} p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ji} - p_{ii}}$$

Of all metrics described above, the MIoU stands out of the crowd as the most used metric due to its representativeness and simplicity. Most challenges and researchers make use of that metric to report their results.

4.2. Results

As we stated before, Section 3.2 provided a functional description of the reviewed methods according to their targets. Now we gathered all the quantitative results for those methods as stated by their authors in their corresponding papers (see Table 3. These results are organized into three parts

depending on the input data used by the methods: 2D RGB or 2.5D RGB-D images, volumetric 3D, or video sequences.

The most used datasets have been selected for that purpose. It is important to remark the heterogeneity of the papers in the field when reporting results. Although most of them try to evaluate their methods in standard datasets and provide enough information to reproduce their results, also expressed in widely known metrics, many others fail to do so. That leads to a situation in which it is hard or even impossible to fairly compare methods.

Table 3: Accuracy results for the most relevant methods and datasets.

Method	PASCAL VOC-2012	PASCAL-Context	Pascal Person-Part	CamVid	CityScapes	Stanford Background	SiftFlow	SUN RGB-D	NYUDv2	SUN3D	ShapeNet Part	Stanford 2D-3D-S	Youtube-Objects
PSPNet[82]	**85.40**	-	-	-	**80.20**	-	-	-	-	-	-	-	-
DeepLab[73]	79.70	**45.70**	**64.94**	-	70.40	-	-	-	-	-	-	-	-
Dilation-10[75]	75.30	-	-	-	67.10	-	-	-	-	-	-	-	-
CRFasRNN[74]	74.70	39.28	-	-	62.50	-	-	-	-	-	-	-	-
ParseNet[81]	69.80	-	-	-	-	-	-	-	-	-	-	-	-
FCN-8s[68]	67.20	39.10	-	-	65.30	-	-	-	-	-	-	-	-
M.scale-CNN-Eigen[78]	62.60	-	-	-	-	-	-	-	-	-	-	-	-
Bayesian SegNet[71]	60.50	-	-	63.10	-	-	-	-	-	-	-	-	-
SegNet[70]	-	-	-	60.10	-	-	-	-	-	-	-	-	-
DAG-RNN[87]	-	-	-	**91.60**	-	-	**85.30**	-	-	-	-	-	-
ReSeg [83]	-	-	-	58.80	-	-	-	-	-	-	-	-	-
ENet[76]	-	-	-	55.60	58.30	-	-	-	-	-	-	-	-
rCNN[86]	-	-	-	-	-	**80.20**	77.70	-	-	-	-	-	-
2D-LSTM[85]	-	-	-	-	-	78.56	70.11	-	-	-	-	-	-
LSTM-CF[84]	-	-	-	-	-	-	-	**48.10**	**49.40**	**58.50**	-	-	-
PointNet[92]	-	-	-	-	-	-	-	-	-	-	**83.70**	**47.71**	-
PointNet++[93]	-	-	-	-	-	-	-	-	-	-	**85.10**	-	-
DGCNN[94]	-	-	-	-	-	-	-	-	-	-	**85.10**	**56.10**	-
Clockwork Convnet[95]	-	-	-	-	**64.40**	-	-	-	-	-	-	-	**68.50**
SegmPred [97] [25]	-	-	-	**46.80**	**59.40**	-	-	-	-	-	-	-	-

Furthermore, we also came across the fact few authors provide information about other metrics rather than accuracy. Despite the importance of other metrics, most of the papers do not include any data about execution time nor memory footprint. In some cases that information is provided, but no reproducibility information is given so it is impossible to know the setup that produced those results which are of no use.

[25] This model is focused on predicting future frames in the space of semantic segmentation, thus a direct comparison with the other methods listed in this table would not be fair.

4.2.1. RGB

For the single 2D image category we have selected seven datasets: PASCAL VOC2012, PASCAL Context, PASCAL Person-Part, CamVid, CityScapes, Stanford Background, and SiftFlow. That selection accounts for a wide range of situations and targets.

The first, and arguably the most important dataset, in which the vast majority of methods are evaluated is PASCAL VOC-2012. This set of results shows a clear improvement trend from the firs proposed methods (SegNet and the original FCN) to the most complex models such as CRFasRNN and the winner (PSPNet) with 85.70 IoU.

Apart from the widely known VOC we also collected metrics of its Context counterpart in which DeepLab is the top scorer (45.70 IoU).

In addition, we also took into account the PASCAL Part dataset. In this case, the only analyzed method that provided metrics for this dataset is DeepLab which achieved a 64.94 IoU.

Moving from a general-purpose dataset such as PASCAL VOC, we also gathered results for two of the most important urban driving databases. For CamVid, an RNN-based approach (DAG-RNN) is the top one with a 91.60 IoU. Results on a more challenging and currently more in use database like CityScapes change. The trend on this dataset is similar to the one with PASCAL VOC with PSPNet leading with a 80.20 IoU.

The results of various recurrent networks on the Stanford Background dataset are also remarkable. The winner, rCNN, achieves a maximum accuracy of 80.20 IoU.

At last, results for another popular dataset such as SiftFlow are also dominated by recurrent methods. In particular DAG-RNN is the top scorer with 85.30 IoU.

4.2.2. 2.5D

Regarding the 2.5D category, i.e., datasets which also include depth information apart from the typical RGB channels, we have selected three of them for the analysis: SUN-RGB-D and NYUDv2. Results for SUN-RGB-D are only provided by LSTM-CF, which achieves 48.10 IoU. In the case of NYUDv2, results are exclusive too for LSTM-CF. That method reaches 49.40 IoU. LSTM-CF is the only one which provides information for SUN-3D, in this case a 58.50 accuracy.

4.2.3. 3D

Two 3D datasets have been chosen for this discussion: ShapeNet Part and Stanford-2D-3D-S. PointNet++ and DGCNN are the most promising alternatives in part segmentation with 85.10 mean IoU. In the case of Stanford-2D-3D-S, DGCNN raised the bar set by PointNet from 47.71 to 56.10 mean IoU.

4.2.4. Sequences

The last category included in this discussion is video or sequences. For that part we gathered results for two datasets which are suitable for sequence segmentation: CityScapes and YouTube-Objects. Only one of the reviewed methods for video segmentation provides quantitative results on those datasets: Clockwork Convnet. That method reaches 64.40 IoU on CityScapes, and 68.50 on YouTube-Objects.

4.3. Summary

In light of the results, we can draw various conclusions. The most important of them is related to reproducibility. As we have observed, many methods report results on non-standard datasets or they are not even tested at all. That makes comparisons impossible. Furthermore, some of them do not describe the setup for the experimentation or do not provide the source code for the implementation, thus significantly hurting reproducibility. Methods should report their results on standard datasets, exhaustively describe the training procedure, and also make their models and weights publicly available to enable progress.

Another important fact discovered thanks to this study is the lack of information about other metrics such as execution time and memory footprint. Almost no paper reports this kind of information, and those who do suffer from the reproducibility issues mentioned before. This void is due to the fact that most methods focus on accuracy without any concern about time or space. However, it is important to think about where are those methods being applied. In practice, most of them will end up running on embedded devices, e.g., self-driving cars, drones, or robots, which are fairly limited from both sides: computational power and memory.

Regarding the results themselves, we can conclude that DeepLab is the most solid method which outperforms the rest on almost every single RGB images dataset by a significant margin. The 2.5D or multimodal datasets are dominated by recurrent networks such as LSTM-CF. 3D data segmentation

still has a long way to go with PointNet paving the way for future research on dealing with unordered point clouds without any kind of preprocessing or discretization. Finally, dealing with video sequences is another green area with no clear direction, but Clockwork Convnets are the most promising approach thanks to their efficiency and accuracy duality. 3D convolutions are worth remarking due to their power and flexibility to process multichannel inputs, making them successful at capturing both spatial and temporal information.

4.4. Future Research Directions

Based on the reviewed research, which marks the state of the art of the field, we present a list of future research directions that would be interesting to pursue.

- *3D datasets*: methods that make full use of 3D information are starting to rise but, even if new proposals and techniques are engineered, they still lack one of the most important components: data. There is a strong need for large-scale datasets for 3D semantic segmentation, which are harder to create than their lower dimensional counterparts. Although there are already some promising works, there is still room for more, better, and varied data. It is important to remark the importance of real-world 3D data since most of the already existing works are synthetic databases. A proof of the importance of 3D is the fact that the ILSVRC will feature 3D data in 2018.

- *Sequence datasets*: the same lack of large-scale data that hinders progress on 3D segmentation also impacts video segmentation. There are only a few datasets that are sequence-based and thus helpful for developing methods which take advantage of temporal information. Bringing up more high-quality data from this nature, either 2D or 3D, will unlock new research lines without any doubt.

- *Point cloud segmentation using Graph Convolutional Networks (GCNs)*: as we already mentioned, dealing with 3D data such as point clouds poses an unsolved challenge. Due to its unordered and unstructured nature, traditional architectures such as CNNs cannot be applied unless some sort of discretization process is applied to structure it. One promising line of research aims to treat point clouds as graphs and apply convolutions over them [121] [122] [123]. This has the advantage of preserving spatial cues in every dimension without quantizing data.

- *Context knowledge*: while FCNs are a consolidated approach for semantic segmentation, they lack several features such as context modelling that help increasing accuracy. The reformulation of CRFs as RNNs to create end-to-end solutions seems to be a promising direction to improve results on real-life data. Multi-scale and feature fusion approaches have also shown remarkable progress. In general, all those works represent important steps towards achieving the ultimate goal, but there are some problems that still require more research.

- *Real-time segmentation*: In many applications, precision is important; however, it is also crucial that these implementations are able to cope with common camera frame rates (at least 25 frames per second). Most of the current methods are far from that framerate, e.g., FCN-8s takes roughly 100 ms to process a low-resolution PASCAL VOC image whilst CRFasRNN needs more than 500 ms. Therefore, during the next years, we expect a stream of works coming out, focusing more on real-time constraints. These future works will have to find a trade-off between accuracy and runtime.

- *Memory:* some platforms are bounded by hard memory constraints. Segmentation networks usually do need significant amounts of memory to be executed for both inference and training. In order to fit them in some devices, networks must be simplified. While this can be easily accomplished by reducing their complexity (often trading it for accuracy), another approaches can be taken. Pruning is a promising research line that aims to simplify a network, making it lightweight while keeping the knowledge, and thus the accuracy, of the original network architecture [124][125][126].

- *Temporal coherency on sequences:* some methods have addressed video or sequence segmentation but either taking advantage of that temporal cues to increase accuracy or efficiency. However, none of them have explicitly tackled the coherency problem. For a segmentation system to work on video streams it is important, not only to produce good results frame by frame, but also make them coherent through the whole clip without producing artifacts by smoothing predicted per-pixel labels along the sequence.

- *Multi-view integration:* Use of multiple views in recently proposed seg-

mentation works is mostly limited to RGB-D cameras and in particular focused on single-object segmentation.

5. Conclusion

To the best of our knowledge, this is the first review paper in the literature which focuses on semantic segmentation using deep learning. In comparison with other surveys, this paper is devoted to such rising topic as deep learning, covering the most advanced and recent work on that front.

We formulated the semantic segmentation problem and provided the reader with the necessary background knowledge about deep learning for the task. We covered the contemporary literature of datasets and methods, providing a comprehensive survey of 28 datasets and 29 methods.

Datasets were carefully described, stating their purposes and characteristics so that researchers can easily pick the one that best suits their needs. We presented a comparative summary of datasets in a tabular form to ease the comparison.

Methods were surveyed from two perspectives: contributions (from a result-agnostic point of view) and raw results, i.e., accuracy (quantitative evaluation on the most common datasets). We also presented a comparative summary of methods in tabular form and grouped them hierarchically in a graph.

In the end, we discussed the results and provided useful insight for future research directions and open problems in the field. A general conclusion that we can draw from this study is that semantic segmentation has been approached with many success stories but still remains an open problem whose solution would prove really useful for a wide set of real-world applications. Furthermore, deep learning has proved to be extremely powerful to tackle this problem so we can expect a flurry of innovation and spawns of research lines in the upcoming years.

Acknowledgments

This work has been funded by the Spanish Government TIN2016-76515-R funding for the COMBAHO project, supported with Feder funds. It has also been supported by a Spanish national grant for PhD studies FPU15/04516 (Alberto Garcia-Garcia). In addition, it was also funded by the grant Ayudas para Estudios de Master e Iniciacion a la Investigacion from the University of Alicante.

References

[1] A. Ess, T. Müller, H. Grabner, L. J. Van Gool, Segmentation-based urban traffic scene understanding., in: BMVC, Vol. 1, 2009, p. 2.

[2] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3354–3361. doi:10.1109/CVPR.2012.6248074.

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.

[4] M. Oberweger, P. Wohlhart, V. Lepetit, Hands deep in deep learning for hand pose estimation, arXiv preprint arXiv:1502.06807.

[5] Y. Yoon, H.-G. Jeon, D. Yoo, J.-Y. Lee, I. So Kweon, Learning a deep convolutional network for light-field image super-resolution, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2015, pp. 24–32.

[6] J. Wan, D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, J. Li, Deep learning for content-based image retrieval: A comprehensive study, in: Proceedings of the 22nd ACM international conference on Multimedia, ACM, 2014, pp. 157–166.

[7] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, P. E. Barbano, Toward automatic phenotyping of developing embryos from videos, IEEE Transactions on Image Processing 14 (9) (2005) 1360–1371.

[8] D. Ciresan, A. Giusti, L. M. Gambardella, J. Schmidhuber, Deep neural networks segment neuronal membranes in electron microscopy images, in: Advances in neural information processing systems, 2012, pp. 2843–2851.

[9] C. Farabet, C. Couprie, L. Najman, Y. LeCun, Learning hierarchical features for scene labeling, IEEE transactions on pattern analysis and machine intelligence 35 (8) (2013) 1915–1929.

[10] B. Hariharan, P. Arbeláez, R. Girshick, J. Malik, Simultaneous detection and segmentation, in: European Conference on Computer Vision, Springer, 2014, pp. 297–312.

[11] S. Gupta, R. Girshick, P. Arbeláez, J. Malik, Learning rich features from rgb-d images for object detection and segmentation, in: European Conference on Computer Vision, Springer, 2014, pp. 345–360.

[12] H. Zhu, F. Meng, J. Cai, S. Lu, Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation, Journal of Visual Communication and Image Representation 34 (2016) 12 – 27. doi:http://dx.doi.org/10.1016/j.jvcir.2015.10.012.
URL http://www.sciencedirect.com/science/article/pii/S1047320315002035

[13] M. Thoma, A survey of semantic segmentation, CoRR abs/1602.06541.
URL http://arxiv.org/abs/1602.06541

[14] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

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

[16] H. Noh, S. Hong, B. Han, Learning deconvolution network for semantic segmentation, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1520–1528.

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

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

[19] A. Graves, S. Fernández, J. Schmidhuber, Multi-dimensional Recurrent Neural Networks, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 549–558.

[20] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. C. Courville, Y. Bengio, Renet: A recurrent neural network based alternative to convolutional networks, CoRR abs/1505.00393.
URL http://arxiv.org/abs/1505.00393

[21] A. Ahmed, K. Yu, W. Xu, Y. Gong, E. Xing, Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks, in: European Conference on Computer Vision, Springer, 2008, pp. 69–82.

[22] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1717–1724.

[23] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Advances in neural information processing systems, 2014, pp. 3320–3328.

[24] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, A. A. Efros, Context encoders: Feature learning by inpainting, CoRR abs/1604.07379.
URL http://arxiv.org/abs/1604.07379

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 248–255.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International Journal of Computer Vision 115 (3) (2015) 211–252.

[27] S. R. Richter, V. Vineet, S. Roth, V. Koltun, Playing for Data: Ground Truth from Computer Games, Springer International Publishing, Cham, 2016, pp. 102–118.

[28] S. C. Wong, A. Gatt, V. Stamatescu, M. D. McDonnell, Understanding data augmentation for classification: when to warp?, in: Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on, IEEE, 2016, pp. 1–6.

[29] X. Shen, A. Hertzmann, J. Jia, S. Paris, B. Price, E. Shechtman, I. Sachs, Automatic portrait segmentation for image stylization, in: Computer Graphics Forum, Vol. 35, Wiley Online Library, 2016, pp. 93–102.

[30] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: A retrospective, International Journal of Computer Vision 111 (1) (2015) 98–136.

[31] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, A. Yuille, The role of context for object detection and semantic segmentation in the wild, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[32] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, A. Yuille, Detect what you can: Detecting and representing objects using holistic models and body parts, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[33] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, J. Malik, Semantic contours from inverse detectors, in: 2011 International Conference on Computer Vision, IEEE, 2011, pp. 991–998.

[34] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European Conference on Computer Vision, Springer, 2014, pp. 740–755.

[35] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, A. M. Lopez, The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3234–3243.

[36] M. Cordts, M. Omran, S. Ramos, T. Scharwächter, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset, in: CVPR Workshop on The Future of Datasets in Vision, 2015.

[37] G. J. Brostow, J. Fauqueur, R. Cipolla, Semantic object classes in video: A high-definition ground truth database, Pattern Recognition Letters 30 (2) (2009) 88–97.

[38] P. Sturgess, K. Alahari, L. Ladicky, P. H. Torr, Combining appearance and structure from motion features for road scene understanding, in: BMVC 2012-23rd British Machine Vision Conference, BMVA, 2009.

[39] J. M. Alvarez, T. Gevers, Y. LeCun, A. M. Lopez, Road scene segmentation from a single image, in: European Conference on Computer Vision, Springer, 2012, pp. 376–389.

[40] G. Ros, J. M. Alvarez, Unsupervised image transformation for outdoor semantic labelling, in: Intelligent Vehicles Symposium (IV), 2015 IEEE, IEEE, 2015, pp. 537–542.

[41] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, A. M. Lopez, Vision-based offline-online perception paradigm for autonomous driving, in: Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on, IEEE, 2015, pp. 231–238.

[42] R. Zhang, S. A. Candra, K. Vetter, A. Zakhor, Sensor fusion for semantic segmentation of urban scenes, in: Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, 2015, pp. 1850–1857.

[43] S. Gould, R. Fulton, D. Koller, Decomposing a scene into geometric and semantically consistent regions, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 1–8.

[44] C. Liu, J. Yuen, A. Torralba, Nonparametric scene parsing: Label transfer via dense scene alignment, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 1972–1979.

[45] S. D. Jain, K. Grauman, Supervoxel-consistent foreground propagation in video, in: European Conference on Computer Vision, Springer, 2014, pp. 656–671.

[46] S. Bell, P. Upchurch, N. Snavely, K. Bala, Material recognition in the wild with the materials in context database, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 3479–3487.

[47] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, A. Sorkine-Hornung, A benchmark dataset and evaluation methodology for video object segmentation, in: Computer Vision and Pattern Recognition, 2016.

[48] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, L. Van Gool, The 2017 davis challenge on video object segmentation, arXiv:1704.00675.

[49] N. Silberman, D. Hoiem, P. Kohli, R. Fergus, Indoor segmentation and support inference from rgbd images, in: European Conference on Computer Vision, Springer, 2012, pp. 746–760.

[50] J. Xiao, A. Owens, A. Torralba, Sun3d: A database of big spaces reconstructed using sfm and object labels, in: 2013 IEEE International Conference on Computer Vision, 2013, pp. 1625–1632. doi:10.1109/ICCV.2013.458.

[51] S. Song, S. P. Lichtenberg, J. Xiao, Sun rgb-d: A rgb-d scene understanding benchmark suite, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 567–576.

[52] K. Lai, L. Bo, X. Ren, D. Fox, A large-scale hierarchical multi-view rgb-d object dataset, in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 1817–1824.

[53] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas, A scalable active framework for region annotation in 3d shape collections, SIGGRAPH Asia.

[54] I. Armeni, A. Sax, A. R. Zamir, S. Savarese, Joint 2D-3D-Semantic Data for Indoor Scene Understanding, ArXiv e-printsarXiv:1702.01105.

[55] X. Chen, A. Golovinskiy, T. Funkhouser, A benchmark for 3D mesh segmentation, ACM Transactions on Graphics (Proc. SIGGRAPH) 28 (3).

[56] A. Quadros, J. Underwood, B. Douillard, An occlusion-aware feature for range images, in: Robotics and Automation, 2012. ICRA'12. IEEE International Conference on, IEEE, 2012.

[57] T. Hackel, J. D. Wegner, K. Schindler, Contour detection in unstructured 3d point clouds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1610–1618.

[58] G. J. Brostow, J. Shotton, J. Fauqueur, R. Cipolla, Segmentation and recognition using structure from motion point clouds, in: European Conference on Computer Vision, Springer, 2008, pp. 44–57.

[59] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The kitti dataset, The International Journal of Robotics Research 32 (11) (2013) 1231–1237.

[60] A. Prest, C. Leistner, J. Civera, C. Schmid, V. Ferrari, Learning object class detectors from weakly annotated video, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 3282–3289.

[61] S. Bell, P. Upchurch, N. Snavely, K. Bala, OpenSurfaces: A richly annotated catalog of surface appearance, ACM Trans. on Graphics (SIGGRAPH) 32 (4).

[62] B. C. Russell, A. Torralba, K. P. Murphy, W. T. Freeman, Labelme: a database and web-based tool for image annotation, International journal of computer vision 77 (1) (2008) 157–173.

[63] S. Gupta, P. Arbelaez, J. Malik, Perceptual organization and recognition of indoor scenes from rgb-d images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 564–571.

[64] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, T. Darrell, A Category-Level 3D Object Dataset: Putting the Kinect to Work, Springer London, London, 2013, pp. 141–165.

[65] A. Richtsfeld, The object segmentation database (osd) (2012).

[66] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al., Shapenet: An information-rich 3d model repository, arXiv preprint arXiv:1512.03012.

[67] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1534–1543.

[68] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[69] O. Ronneberger, P.Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Medical Image Computing and Computer-Assisted Intervention (MICCAI), Vol. 9351 of LNCS, Springer, 2015, pp. 234–241, (available on arXiv:1505.04597 [cs.CV]). URL http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a

[70] V. Badrinarayanan, A. Kendall, R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for scene segmentation, IEEE transactions on pattern analysis and machine intelligence.

[71] A. Kendall, V. Badrinarayanan, R. Cipolla, Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding, arXiv preprint arXiv:1511.02680.

[72] C. Liang-Chieh, G. Papandreou, I. Kokkinos, K. Murphy, A. Yuille, Semantic image segmentation with deep convolutional nets and fully connected crfs, in: International Conference on Learning Representations, 2015.

[73] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, CoRR abs/1606.00915. URL http://arxiv.org/abs/1606.00915

[74] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, P. H. Torr, Conditional random fields as recurrent neural networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1529–1537.

[75] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122.

[76] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, Enet: A deep neural network architecture for real-time semantic segmentation, arXiv preprint arXiv:1606.02147.

[77] A. Raj, D. Maturana, S. Scherer, Multi-scale convolutional architecture for semantic segmentation.

[78] D. Eigen, R. Fergus, Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2650–2658.

[79] A. Roy, S. Todorovic, A multi-scale cnn for affordance segmentation in rgb images, in: European Conference on Computer Vision, Springer, 2016, pp. 186–201.

[80] X. Bian, S. N. Lim, N. Zhou, Multiscale fully convolutional network with application to industrial inspection, in: Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on, IEEE, 2016, pp. 1–8.

[81] W. Liu, A. Rabinovich, A. C. Berg, Parsenet: Looking wider to see better, arXiv preprint arXiv:1506.04579.

[82] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, CoRR abs/1612.01105.
URL `http://arxiv.org/abs/1612.01105`

[83] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, A. Courville, Reseg: A recurrent neural network-based model for semantic segmentation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2016.

[84] Z. Li, Y. Gan, X. Liang, Y. Yu, H. Cheng, L. Lin, LSTM-CF: Unifying Context Modeling and Fusion with LSTMs for RGB-D Scene Labeling, Springer International Publishing, Cham, 2016, pp. 541–557.

[85] W. Byeon, T. M. Breuel, F. Raue, M. Liwicki, Scene labeling with lstm recurrent neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3547–3555.

[86] P. H. Pinheiro, R. Collobert, Recurrent convolutional neural networks for scene labeling., in: ICML, 2014, pp. 82–90.

[87] B. Shuai, Z. Zuo, B. Wang, G. Wang, Dag-recurrent neural networks for scene labeling, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3620–3629.

[88] P. O. Pinheiro, R. Collobert, P. Dollar, Learning to segment object candidates, in: Advances in Neural Information Processing Systems, 2015, pp. 1990–1998.

[89] P. O. Pinheiro, T.-Y. Lin, R. Collobert, P. Dollár, Learning to refine object segments, in: European Conference on Computer Vision, Springer, 2016, pp. 75–91.

[90] S. Zagoruyko, A. Lerer, T. Lin, P. O. Pinheiro, S. Gross, S. Chintala, P. Dollár, A multipath network for object detection, in: Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016, 2016.
URL `http://www.bmva.org/bmvc/2016/papers/paper015/index.html`

[91] J. Huang, S. You, Point cloud labeling using 3d convolutional neural network, in: Proc. of the International Conf. on Pattern Recognition (ICPR), Vol. 2, 2016.

[92] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, arXiv preprint arXiv:1612.00593.

[93] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Advances in Neural Information Processing Systems, 2017, pp. 5105–5114.

[94] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph cnn for learning on point clouds, arXiv preprint arXiv:1801.07829.

[95] E. Shelhamer, K. Rakelly, J. Hoffman, T. Darrell, Clockwork convnets for video semantic segmentation, in: Computer Vision–ECCV 2016 Workshops, Springer, 2016, pp. 852–868.

[96] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Deep end2end voxel2voxel prediction, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016, pp. 17–24.

[97] N. Neverova, P. Luc, C. Couprie, J. J. Verbeek, Y. LeCun, Predicting deeper into the future of semantic segmentation, CoRR abs/1703.07684.

[98] M. D. Zeiler, G. W. Taylor, R. Fergus, Adaptive deconvolutional networks for mid and high level feature learning, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 2018–2025.

[99] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833.

[100] Z. Zhang, Q. Liu, Y. Wang, Road extraction by deep residual u-net, CoRR abs/1711.10684. arXiv:1711.10684.
URL http://arxiv.org/abs/1711.10684

[101] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, in: ACM transactions on graphics (TOG), Vol. 23, ACM, 2004, pp. 309–314.

[102] J. Shotton, J. Winn, C. Rother, A. Criminisi, Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context, International Journal of Computer Vision 81 (1) (2009) 2–23.

[103] V. Koltun, Efficient inference in fully connected crfs with gaussian edge potentials, Adv. Neural Inf. Process. Syst 2 (3) (2011) 4.

[104] P. Krähenbühl, V. Koltun, Parameter learning and convergent inference for dense random fields., in: ICML (3), 2013, pp. 513–521.

[105] S. Zhou, J.-N. Wu, Y. Wu, X. Zhou, Exploiting local structures with the kronecker layer in convolutional networks, arXiv preprint arXiv:1512.09194.

[106] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[107] K. Cho, B. van Merrienboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, in: Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014, 2014, pp. 103–111.
URL http://aclweb.org/anthology/W/W14/W14-4012.pdf

[108] Z. Li, Y. Gan, X. Liang, Y. Yu, H. Cheng, L. Lin, RGB-D scene labeling with long short-term memorized fusion model, CoRR abs/1604.05000.
URL http://arxiv.org/abs/1604.05000

[109] G. Li, Y. Yu, Deep contrast learning for salient object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 478–487.

[110] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, J. Malik, Multiscale combinatorial grouping, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 328–335.

[111] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.

[112] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, J. Xiao, Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge, in: Robotics and Automation (ICRA), 2017 IEEE International Conference on, IEEE, 2017, pp. 1386–1383.

[113] L. Ma, J. Stuckler, C. Kerl, D. Cremers, Multi-view deep learning for consistent semantic mapping with rgb-d cameras, in: arXiv:1703.08866, 2017.

[114] C. Hazirbas, L. Ma, C. Domokos, D. Cremers, Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture, in: Proc. ACCV, Vol. 2, 2016.

[115] H. Zhang, K. Jiang, Y. Zhang, Q. Li, C. Xia, X. Chen, Discriminative feature learning for video semantic segmentation, in: Virtual Reality and Visualization (ICVRV), 2014 International Conference on, IEEE, 2014, pp. 321–326.

[116] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, IEEE Transactions on pattern analysis and machine intelligence 23 (11) (2001) 1222–1239.

[117] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3d convolutional networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4489–4497.

[118] A. G. Roy, S. Conjeti, S. P. K. Karri, D. Sheet, A. Katouzian, C. Wachinger, N. Navab, Relaynet: retinal layer and fluid segmentation of macular optical coherence tomography using fully convolutional networks, Biomed. Opt. Express 8 (8) (2017) 3627–3642. doi:10.1364/BOE.8.003627.
URL http://www.osapublishing.org/boe/abstract.cfm?URI=boe-8-8-3627

[119] F. Milletari, N. Navab, S. Ahmadi, V-net: Fully convolutional neural networks for volumetric medical image segmentation, CoRR abs/1606.04797.
URL http://arxiv.org/abs/1606.04797

[120] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: International Conference on Computer Vision (ICCV), Venice, Italy, 2017, oral.
URL https://vision.cornell.edu/se3/wp-content/uploads/2017/09/focal_loss.pdf

[121] M. Henaff, J. Bruna, Y. LeCun, Deep convolutional networks on graph-structured data, arXiv preprint arXiv:1506.05163.

[122] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907.

[123] M. Niepert, M. Ahmed, K. Kutzkov, Learning convolutional neural networks for graphs, in: Proceedings of the 33rd annual international conference on machine learning. ACM, 2016.

[124] S. Anwar, K. Hwang, W. Sung, Structured pruning of deep convolutional neural networks, ACM Journal on Emerging Technologies in Computing Systems (JETC) 13 (3) (2017) 32.

[125] S. Han, H. Mao, W. J. Dally, Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, arXiv preprint arXiv:1510.00149.

[126] P. Molchanov, S. Tyree, T. Karras, T. Aila, J. Kautz, Pruning convolutional neural networks for resource efficient transfer learning, arXiv preprint arXiv:1611.06440.

- An in-depth review of deep learning methods for semantic segmentation applied to various areas.
- An overview of background concepts and formulation for newcomers.
- An analysis of datasets and challenges for semantic segmentation.
- A structured and logical review of methods, highlighting their contributions and significance.
- Quantitative comparison of performance and accuracy on common datasets.
- A discussion of future works and promising research lines and conclusions about the state of the art of the field.