Semantic Segmentation, Urban Navigation, and Research Directions

Qasim Nadeem
Princeton University
qnadeem@cs.princeton.edu

Abstract

We introduce and give a detailed review of semantic segmentation. We begin with some key early methods, and our discussion will make its way to the state of the art. We also talk about the use of semantic segmentation for navigation in outdoor urban environments. Finally, we touch on two *fascinating research directions that we believe hold a lot of potential.*

Semantic segmentation is the task of taking an input image and producing dense, pixel level, semantic predictions. Accurate and rich semantic segmentation is a driver of visual understanding and reasoning, and has a direct impact on many real world applications.

1. Introduction

A *semantic segmentation* system takes an image or video frame as input, and outputs a heat-map classifying each of the pixels into one of k pre-defined categories. Normally, the system undergoes learning on some training data where the ground truth labels are known. Then, provided a new unlabeled test image, the learner predicts the label for each pixel with one of k semantic classes. The most popular evaluation criteria is the very intuitive metric mean Intersection-over-Union (mIoU).

There are numerous applications to justify the usefulness of semantic segmentation. Robotic navigation and autonomous driving immediately benefit; obstacle detection, path planning, recognizing traversable terrain are some uses. Two features in smart-phones that rely on semantic segmentation are (i) Portrait mode where background pixels are blurred to sharpen the person's silhouette, (ii) several of the clever filters you see on Snapchat and Instagram. A third area of interest is biomedical image segmentation; many processes in radiology, and lab testing can be made more efficient and/or accurate with automatic semantic segmentation.

Instance segmentation (or instance aware segmentation) is a stronger task where separate instances of the same class (say 'bicycle') are made distinctly identified. We do not cover systems that achieve this but many important ideas naturally overlap.

We start with a quick discussion of significant datasets for semantic segmentation. We then explore papers in semantic segmentation starting from traditional methods, and making our way to the state of the art. As much as possible, we explore ideas in chronological order.

2. Datasets

PASCAL VOC 2012 [10] has been the most tested upon benchmark for semantic segmentation, although that is bound to change since state of the art has reached 89.0 mIOU on its test set. MS COCO [21] is another large-scale, popular segmentation dataset. Other significant datasets include SiftFlow [22], NYUDv2 [36], Stanford Background [13]. DAVIS [26] is a significant video object segmentation dataset.

Then, some important datasets geared towards urban-scene understanding, especially with autonomous vehicles' perception in mind. The most significant ones are CityScapes [8], KITTI [12], and CamVid [3]. A couple others include Urban LabelMe [30], and CBCL StreetScenes [2]. A special mention must be made for the large-scale ApolloScape dataset (2018) [15] released by Baidu to further autonomous driving research.

2.1. PASCAL VOC 2012

The dataset is composed of images from the image-hosting website Flickr. The images are hand annotated. There are several detection related challenges for the PASCAL VOC 2012, one of which is a semantic segmentation challenge. There are 21 classes (airplane, bicycle, background...). The public dataset has 1464 training and 1449 validation images. The test set is privately held. Although formally no new competition has been held since 2012, algorithms continue to be evaluated on the 2012 challenge for segmentation (for example DeepLab v3, top of the leaderboard, is a 2018 submission).

2.2. MS COCO

MS COCO is a popular and very challenging large-scale detection data, which includes pixel segmentations. State of the art mIoU on MS COCO is around 48.0. The dataset contains high-resolution images, and features 80 object categories and 91 stuff categories. The semantic segmentation challenge provides 82000 train images, 40500 validation images, and the evaluation is carried out on subsets of 80000 test images. MS COCO has receives continued attention due to its quality annotations and large size. Several detection challenges are hosted at ECCV each year.

2.3. Urban-Scene understanding datasets

Several of the datasets are targeted towards urban scene understanding. Commonly, they have been gathered from cars with high-resolution cameras mounted, some kind of automatic segmentation/detection ran on them, and then manually cleaned by human volunteers.

Cityscapes contains 2D semantic, instance-wise, dense pixel annotations for 30 classes. It has 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 large number of dynamic objects, and varying scene layout and background.

CamVid is a road scene understanding database captured as video sequences with a 960×720 resolution camera mounted on a car's dashboard. Those sequences were sampled to collect 701 frames, and manually annotated with 32 classes. A popular evaluation partition divides this dataset into train/val/test sets of 367/100/233, and uses 11 of the class labels only.

KITTI is a dataset for different computer vision tasks such as stereo, optical flow, 2D/3D object detection and tracking. There are 7481 training and 7518 test images annotated with 2D and 3D bounding boxes for object detection and orientation estimation. There are up to 15 cars and 30 pedestrians in each image. However, pixel-level annotations (7000 images) were only provided later by various third party researchers with differing quality controls.

ApolloScape is large-scale comprehensive dataset for urban street views. The eventual dataset will include RGB videos with 1 million+ high resolution images with per-pixel semantic labels, survey-grade dense 3D points with semantic segmentation, stereoscopic video with rare events, and night-vision sensors. The collection is also careful to cover a wide range of environment, weather, and traffic conditions. The initial release has 143906 video frames and corresponding annotations, 25 class labels and 28 lane markings type labels, for semantic segmentation task (see Figure 1). In addition, 89430 instance-level annotations for movable objects are further provided, to evaluate for instance-level video object segmentation. ApolloScape is thus orders of magnitude larger than other urban dataset.

Figure 1. ApolloScape: sample frame and semantic labeling. [15]

3. Traditional, pre-neural net approaches

Traditional methods for semantic segmentation used hand-crafted and carefully engineered features such as SIFT or HoG, along with learning algorithms like SVMs, and Random Decision Forests. The advent of neural networks has coalesced these traditionally separate parts of the pipeline; NNs learn a suitable *feature representation* as well as a classifier. Below we talk about different feature extractors, learning models successfully used along with them, and a few significant papers that proposed complete pipelines for semantic segmentations.

Image segmentation has often been modeled as an energy minimization problem. CRFs are rich, probabilistic graphical models; pixels of an image can be viewed as variable nodes in a CRF. They generally only model local interactions, and global effects only arise as an indirect consequence. The seminal paper by P Krhenbhl and V Koltun (2011) [17] gave an extremely efficient, approximate inference algorithm for fully-connected CRFs where pairwise edge potentials are defined by a linear combination of Gaussian kernels. These continue to be used to the current day as a post-processing step to boost segmentation accuracy.

Moving on, the choice of feature descriptors and incorporation of domain knowledge has been very significant in computer vision. It is significant even now, in many applications of machine learning, especially when large amount of data is not available. We assume that the reader is familiar with **SIFT** and **HoG** feature descriptors already.

Bag of Visual Words (BOV) features are histogram based counts. As the name suggests, the idea is an extension of 'Bag of Words' for text documents which is a vector of counts of the vocabulary. For BOV, quantization on local image features is done to build a vocabulary of *visual words* (representative vectors chosen for example using k-means clustering). Then, BOV is a histogram of occurrences of

these visual words.

Textons (described neatly in [42]) computed on $d \times d$ patches of images can be thought of as a BOV, where the visual words are vector quantized exemplar responses of a linear filter bank. These features are thought capable of modeling object and class shape, appearance and context, thus capturing semantics of pixels.

3.1. Some significant papers

J Shotton et al (2008) [35] introduced the powerful **Semantic Texton Forests** (STFs) and achieved state of the art results on the PASCAL VOC 2007 challenge for object detection and segmentation. They also increased the execution speed by at least $5\times$. STFs are a type of random decision forest that can efficiently compute powerful, low-level features. Each decision tree acts directly on image pixels, and therefore bypassing the need for expensive computation of filter-bank responses or local descriptors. STFs are extremely fast to both train and test, especially when compared with k-means clustering and nearest-neighbor assignment of feature descriptors, which was needed for traditional textons and BOV models. The nodes in the decision trees provide (i) an implicit hierarchical clustering into *semantic textons*, and (ii) an explicit local classification estimate. The actual *decision rules* for nodes in the trees are simple functions of raw image pixels within a $d \times d$ patch: either the raw value of one pixel, or sum/difference/absolute difference of a pair of pixels. The bag of semantic textons combines a histogram of semantic textons over an image region. And then image-level object detection priors are considered to allow for coherent, refined segmentations. See Figure 2.

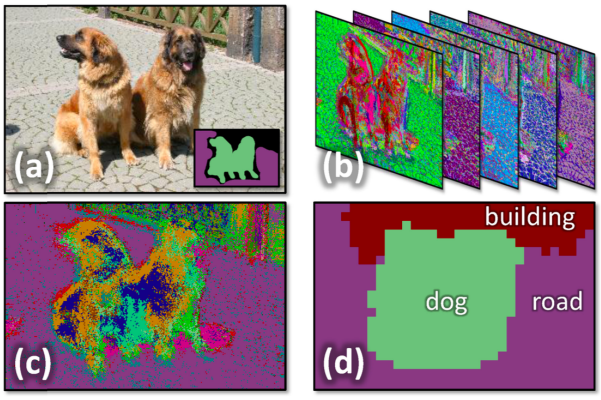

Figure 2. (a) Test image with ground truth. STFs efficiently compute (b) a set of semantic textons per-pixel and (c) a rough local segmentation prior. The algorithm combines both textons and priors as features to give coherent semantic segmentation (d) [35]

G Brostow et al (2008) [4] estimated 3D point clouds from videos (e.g. from a driving car), introduced features that project those 3D cues back to the 2D image plane while modeling spatial layout and context. They then trained a randomized decision forest that combines many such features to achieve 2D semantic segmentation of urban scenes they collected (11 classes). The most important point to note is that the only features they use are *motion* and *structure* based (derived from the point cloud). This is in support of the idea that segmentation for videos can and should leverage the incredibly valuable spatio-temporal information available.

P Sturgess et al (2009) [37] advanced the state of the art, and improved segmentation accuracy on CamVid from 69% to 84%. They integrate motion and appearance-based features. The motion-based features are extracted from 3D point clouds (nearly the same way as [4]), and appearance-based features consist of textons, colour, location, and HOG. They (i) formulate the problem in a CRF framework in order to probabilistically model the label likelihoods and priors; (ii) use a *novel boosting* approach to combine motion and appearance-based features, which selects discriminative features for each class to generate likelihood terms; (iii) incorporate higher order potentials in their CRF model.

L Ladicky (2010) [18] released a paper on *scene understanding* (umbrella term encompassing object recognition/detection, segmentation, and 3D scene recovery). They achieved competitive results on PASCAL VOC and CamVid. Their learning model is a CRF defined on pixels, segments and objects. They define a global energy function for the model, which combines results from sliding window detectors (any object detector can be plugged into their model), and low-level pixel-based unary and pairwise relations. A major contribution is showing that their proposed global energy function is efficiently solvable using a graph-cut based algorithm.

4. Modern approaches (2015 onwards)

Successful deep neural net architectures for image level classification like AlexNet, VGG net, GoogLeNet, and ResNet are a natural precursor to, and often a direct part of semantic segmentation architectures. We assume familiarity with those architectures.

It's helpful to note that pixel-level classification involves *two simultaneous tasks* of classification (correct semantic concept needs to be picked) and localization (pixel label must be aligned to correct coordinate in the output heatmap). These two tasks are naturally in conflict: classification requires models to be invariant to transformations of various kinds, but localization should indeed be sensitive to it because pinning the precise location is the point. Since classification CNNs were made with the first task in mind, adapting them introduces the second disagreeing task, and as we will see most papers will work to resolve this friction.

In this section, we will talk about how the current state of the art developed. All state of the art model involve con-

volutional neural nets. In our discussion below, it proves helpful to arrange modern semantic segmentation architectures in two divisions. The first division contains architectures primarily influenced by the 2015 Fully Convolutional Networks (FCN) paper; these can also be called encoder-decoder architectures. The second division contains architectures that are also influenced by the FCN paper but additionally they employ *dilated convolutions* which we will learn about later. Note that down-sampling and up-sampling still occurs, as in encoder-decoder architectures, but it's not nearly as severe.

4.1. FCNs and other encoder-decoder architectures

CNNs had previously been adapted, albeit awkwardly, for segmentation. Those earlier ideas had intricacies like patch-wise processing, small models restricting receptive fields, various post-processing ideas, multi-scale pyramid processing, and ensembles. In 2014, J Long et al invented a seminal CNN architecture named **Fully Convolutional Networks** [23]. It offered simplicity, end-to-end training, efficient learning and inference, and significantly improved the state of the art on VOC 2012. They took pre-trained classification networks (VGG-16, AlexNet, GoogLeNet), and replaced fully connected layers with convolutional layers (thus the name *fully convolutional*) to output spatial maps instead of image-level classification scores. The spatial maps are still low resolution because of earlier pooling layers, so they need to be up-sampled. The up-sampling is done in a learn-able manner, as opposed to say bilinear interpolation, using *de-convolutions* (also called *fractionally strided convolutions*) to produce pixel-level classification scores. This already gave them state of the art results but the segmentation is *coarse* because naturally early pooling layers lose spatial information; this is the key weakness of FCNs that later papers tried to address. The FCN paper offered *skip connections* from lower high resolution feature maps with fine strides to the final prediction layer to improve granularity and accuracy. Figure 3 provides an overview.

The part of FCNs taken from classification architectures is called an *encoder* since it encodes the input image as a low resolution feature map, and the part after is called a *decoder* since it gradually increases the resolution back to the original image. **SegNet** (2015) [1] is a encoder-decoder architecture that gave a novel way to up-sample in the decoder leading to competitive inference time and accuracy, and state of the art memory efficiency. In particular, it copies max-pooling indices from encoding layers to corresponding decoding layers for a non-linear up-sampling. Thus, the up-sampling does not depend on learned parameters. Figure 4 provides an overview.

U-Net (2015) [28], introduced for biomedical image segmentation, is an important architecture that works really

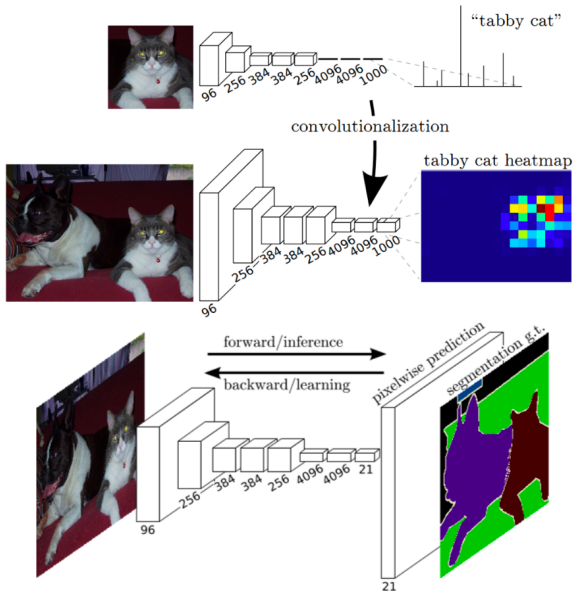

Figure 3. How FCNs re-purpose classification networks for segmentation. [23]

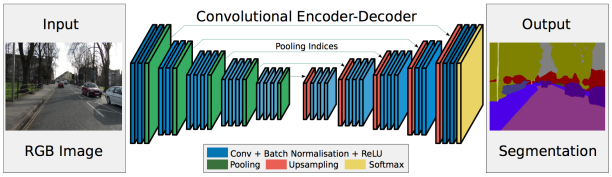

Figure 4. SegNet's architecture. [1]

well in practice too. As there was little training data for the biomedical task, the authors used excessive data augmentation by applying elastic deformations to available training images. This allows the network to learn invariance to such deformations. This is particularly nice for biomedical segmentation as deformation is a common variation in tissue, and realistic deformations can be simulated efficiently. The architecture is quite simple. The *encoder* consists of 3×3 convolutions, ReLUs, and 2×2 max pooling for downsampling. In each downsampling step, they double the number of feature channels - increasing the "what" information and decreasing the "where". A step in the *decoder* consists of a de-convolution followed by a 2×2 convolution that halves the number of feature channels, *a concatenation with the correspondingly cropped feature map from the encoder*, a 3×3 convolution, and a ReLU. They achieved the state of the art on several biomedical image segmentation challenges. Figure 5 provides an overview; each blue rectangle is a feature map and the number written above it gives the number of channels.

RefineNet (2016) [20] achieved state of the art results on 6 datasets which included PASCAL VOC 2012, and Cityscapes. The encoder comes from a pre-trained ResNet

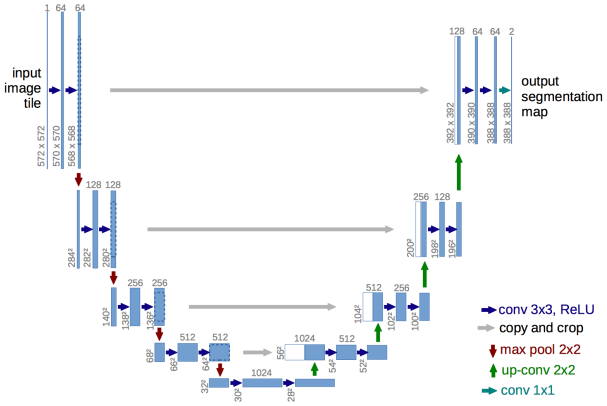

Figure 5. UNet's architecture. [28]

divided into blocks. The decoder consists of intricate *RefineNet blocks* which iteratively fuse features for several ranges of resolutions; low resolution features from previous RefineNet block and high resolution ones from the encoder blocks. They introduce *chained residual pooling* which essentially captures background context from a large image region, by gathering features from several scales, and fusing them in a learn-able manner. Residual connections with identity mappings found throughout the network accommodate gradient propagation to allow end-to-end training. Figure 6 provides an overview.

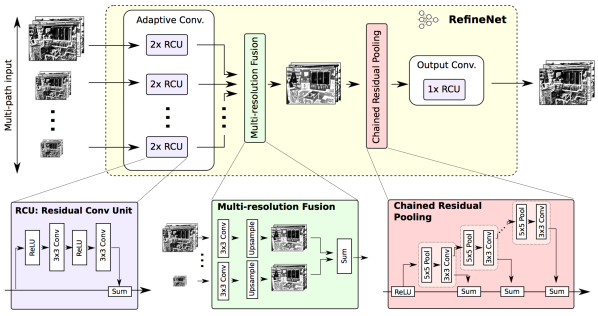

Figure 6. RefineNet's architecture. [20]

Large Kernel Matters (2017) [25] achieved state of the art on VOC 2012 and Cityscapes. The keen insight they had was that stacked small convolution filters (3×3 etc) are in vogue because they're much more efficient to compute than larger kernel filters ($k \times k$ for large k). But they argue that for the segmentation task, a large kernel with larger effective receptive field can assist the simultaneous classification and localization task of semantic segmentation. So they approximate $k \times k$ filters using a novel *Global Convolutional Network* (GCN), that uses combinations of $(1 \times k) + (k \times 1)$ and $(k \times 1) + (1 \times k)$ convolutions, which mimic dense connections in a $k \times k$ region in the feature map. The encoder comes from ResNet followed by GCNs, and the decoder is along FCN lines. They also present a learn-able *Boundary Refinement* module which improves pixel classification on class boundaries, substituting CRF post processing. Figure 7 provides an overview.

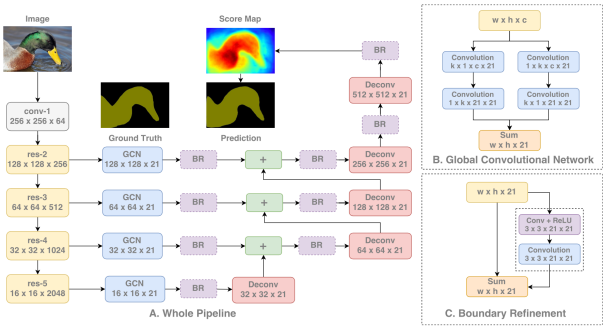

Figure 7. Large Kernel Matters' pipeline. [25]

4.2. Dilated Convolutions paper and inspired architectures

In 2015, F Yu and V Koltun [39] offered an alternative CNN architecture to the encoder-decoder paradigm, and achieved state of the art on VOC 2012. They *dilated convolutions* where dilation factor l determines the expansion *and* dilation of the manner in which the convolution filter is applied. So it's a generalization of standard convolution which has $l = 1$. The insight here is that dilated convolutions allow an exponential increase, with l, in the receptive field without losing spatial resolution (unlike pooling layers for example). They take pre-trained VGG net, remove last two pooling/striding layers, and add dilated convolution layers to produce dense output. They also propose a context module composed of cascades of mutli-scale dilated convolution layers, which takes C feature maps as input and produces same-sized C feature maps. Intuitively, the context module increases the contextual or global 'awareness' of each feature. Figure 8 shows how stacked dilated convolutions work.

L C Chen, G Papandreou et al have published a series of 3 papers (2014, 2016, 2017) on a semantic segmentation project named **DeepLab** [5]. They chop off final layers of ResNet/VGG-16, and use dilated (or atrous) convolution layers similar to [39]. A very nice visual of the effectiveness of dilated convolution versus encoding-decoding for the purpose of dense classification of pixels is given in 9.

They also try two approaches for handling scale variability in semantic segmentation. The first, simpler idea was to extract DCNN feature maps from 3 image scales using parallel net branches, and then bi-linearly interpolate and fuse them using max pooling. The second approach that worked really well is inspired by the successful R-CNN (regional-CNN) spatial pyramid pooling approach; they use several parallel dilated convolution layers with different sampling rates. The features are separately processed and then fused

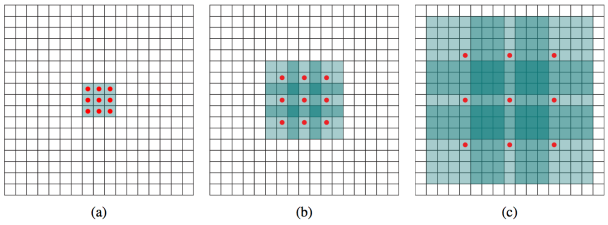

Figure 8. Systematic dilation supports exponential expansion of receptive field without loss of resolution. Consider a 3×3 filter applied in succession with increasing dilation. **(a)** 1-dilated convolution applied; each element has a receptive field of 3×3. **(b)** Produced from (a) by a 2-dilated convolution; each element now has a receptive field of 7×7. **(c)** Produced from (b) by a 4-dilated convolution; each element now has a receptive field of 15×15. The number of parameters in each layer is equal. The receptive field grows exponentially while the number of parameters grows linearly. [39]

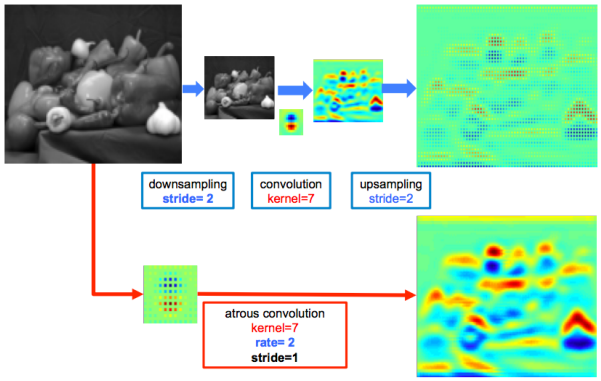

Figure 9. Sparse feature extraction with standard convolutional layers versus dense feature extraction with dilated convolutions. [6]

to generate the final result. They name this *atrous spatial pyramid pooling* (ASPP); see 10.

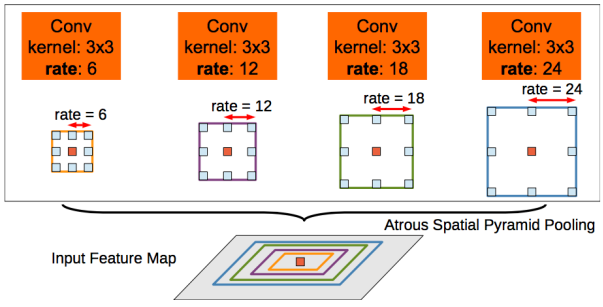

Figure 10. Atrous Spatial Pyramid Pooling: to classify the center orange pixel. [6]

The final segmentation map is still coarse, but it is not severely down-sampled (only $8\times$ compared to $32\times$ in a standard encoder), so they simply use bi-linear interpolation to enlarge feature map to original image resolution (no fancy up-sampling ideas), followed by a fully-connected CRF to refine segmentation result particularly at class boundaries. Their 2016 paper [6] achieved state of the art on VOC 2012 and Cityscapes. Figure 11 gives an overview.

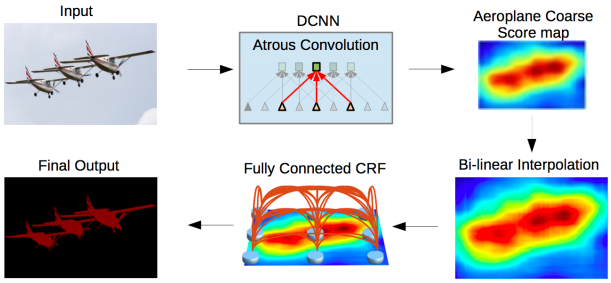

Figure 11. DeepLab pipeline. [6]

Their 2017 paper [7] worked on better ways to incorporate information from various scales. One line of effort was devoted to improving the effectiveness of the ASPP module; the two main changes were *batch normalization parameters* are fine-tuned, and image-level features are augmented into the pyramid. The second line of effort was taking cascading ResNet blocks and replacing last 4 layers with a cascade of dilated convolutions. These blocks are applied to intermediate feature maps. It's worthwhile to note that CRF post-processing was no longer needed. Both of these efforts individually lead to improvement over their earlier papers. And they achieved competitively to the state of the art.

Pyramid Scene Parsing (PSP) Net (2017) [41] achieved state of the art on VOC 2012 and Cityscapes. Their main contribution is a *Pyramid Pooling Module* (PPM) that aggregates global information in a novel manner. The intuition is that in urban scene datasets that have high-resolution images with some categories that cover a lot of pixels (like road, sky etc), global scene categories strongly influence the distribution of segmentation classes and boundaries. Dilated convolution layers are generously used, modifying ResNet similar to how DeepLab did. The feature maps are fed to the PPM which has parallel pooling layers with kernels covering increasing portion of the feature map; the idea is that this *harvests* both local and global context information per pixel. Finally, the output of the PPM is up-sampled and concatenated with the input to the PPM, followed by dilated convolution layers to get the final predictions. Figure 12 provides an overview.

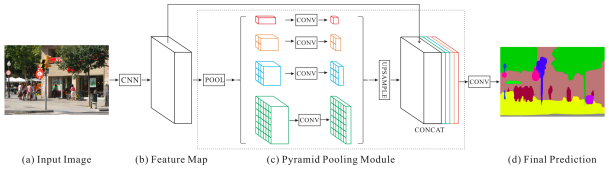

Figure 12. PSPNet's pipeline. [41]

4.3. Research Direction 1: Synthetic Datasets and Dataset Augmentation

Annotated data for dense pixel-level prediction is difficult and costly to gather. There is ongoing research in the community to improve the data efficiency of learning systems. CNNs (and NNs in general) are data hungry models; almost every segmentation architecture we saw had a pre-trained component (on ImageNet and/or MS COCO). One line of research that I think particularly fascinating is to augment real-world human annotated datasets in different ways. A large set of approaches for this involve taking the training data and introducing different levels of transformations to introduce variance and richness e.g. several papers have pursued *cut and paste ideas* where they cut object instances from one training image and paste it into a different image. Another set of approaches involves creating synthetic datasets which have automatic annotations. Some examples follow; they're not all focused only on segmentation but the ideas are similar.

SYNTHIA (CVPR 2016) [29] was created by rendering a photo-realistic virtual city with fine-grained pixel-level annotations for 11 classes (void, road, car, sign, pedestrian, cyclist...), using the Unity game engine. It's purpose is urban scene understanding especially in the context of driving. It contains images and image-sequences. It features more than 213400 synthetic images in total. The authors also characterize data based on diversity in terms of scenes (towns, cities, roads), dynamic objects, seasons, and weather.

Authors experimented with two segmentation networks: the first was an FCN with a VGG-16 encoder, and the second was a T-Net with a VGG-F encoder. They trained and evaluated these nets on four real world urban scene segmentation datasets, and what they critically showed was that augmenting training with SYNTHIA, improved test accuracy by a significant margin, nearly across the board. Figure 13 shows an example frame.

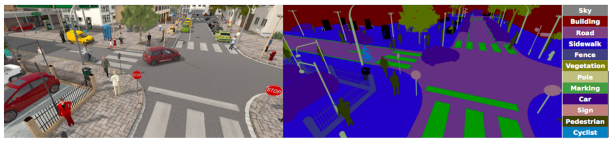

Figure 13. SYNTHIA: A frame and its semantic segmentation. [29]

The **'Driving in the matrix'** paper (2017) [16] from a group at University of Michigan is another example showing the utility of synthetic data. They look at object detection (with a bounding box) not pixel-level prediction but I feel the same idea could be extended for the latter. They generated synthetic urban scenes and pixel annotations from the video game GTA V. They evaluated a Faster R-CNN object detector (VGG-16 pre-trained on ImageNet) for the task of detecting cars on the entire KITTI dataset (7500 images). They trained one instance of this on Cityscapes' 3000 images (a dataset that "resembles" KITTI, the authors say, since both come from roads of Germany), and a second instance on large volumes of synthetic data. The second instance with 50000 synthetic images out-performed the first instance. As expected, the *training value* of a synthetic image is much lower than a real image, however a synthetic image is arguably free of human labor cost. Results are in figure 14.

Data set	Easy	Moderate	Hard
Sim 10k	0.5542	0.3828	0.2904
Sim 50k	**0.6856**	0.5008	0.3926
Sim 200k	0.6803	**0.5257**	**0.4207**
Cityscapes [20]	0.6247	0.4274	0.3566

Figure 14. R-CNN trained on GTA video game outperforms Cityscapes, when evaluated on KITTI for detecting cars (partitioned into 3 based on difficulty of detection). [16]

There are many other papers showing the promise of synthetic datasets for various low and high-level computer vision tasks. We mention a few more below:

(i) House3D (Y Wu et al 2018) [38] by FAIR which sources from the earlier SunCG dataset (Song et al., 2017) and provides 45000 diverse indoor 3D scenes of visually realistic houses with dense 3D annotations of all objects into 80 categories. The uses of this dataset as imagined by the authors are object and scene understanding, 3D navigation, embodied question answering.

(ii) 'Playing for Data: Ground Truth from Computer Games' (2016) [27] showed that although source code of games may be unknown, we can use communication between the game and the graphics hardware to deduce region and pixel annotations. This allows them to extract 25000 pixel-annotated frames from a photo-realistic game (GTA V) in only 49 hours. They showed that training on this data and only $\frac{1}{3}$ of CamVid outperforms model trained entirely on CamVid.

(iii) H Hattori, V N Boddeti et al (2015) [14] showed a way to learn scene-specific pedestrian detectors in the absence of prior data on a novel location for example when a new security camera is installed somewhere. They infer the geometry of the scene, and a pedestrian rendering system then creates a scene simulation of pedestrian motion respecting the geometry (walls, obstacles, walk-able regions) of the scene. This data is then used for training a pedestrian detector. They show that augmenting with real-world data improves accuracy.

Synthetic datasets have room for improvement. Computer graphics and video game visuals are improving, and the fidelity synthetic data can thus be improved. It seems to me that real world data provides much stronger low level visual knowledge (texture, colors, edges are all clearer and more realistic), but for semantic richness and variance (which objects appear close or in context with each other e.g. people sit in cars, people walk pets etc) synthetic datasets can come close to real world data, because it's a matter of rendering varied scenarios. It may be interesting to experimentally judge the training value of subsets of synthetic data to understand how to better create synthetic data. It might also be interesting to experiment with interactive learning, where the learner decides what kind of training data it requires which can then be synthesized.

4.4. Research Direction 2: Video Semantic Segmentation, and Faster Inference

Recent lines of work have moved towards achieving faster inference times; this is useful for real-time applications and mobile hardware. A *common theme* amongst these papers is to find ways to cleverly condense existing segmentation networks like PSPNet, SegNet to trade-off speed for accuracy, at a good rate. Some examples are **ENet** [24], **ICNet** [40], **ShuffleSeg** [11].

Talking about video, the first question that should come to mind is what's special about video, and why not just segment each frame separately. That's certainly possible and a strategy that's often used in practice. But it leaves a lot to be desired. Some reasons are: (i) frames close-in-time are strongly dependent, thus they markedly inform the segmentation of the other, and this information should be leveraged; (ii) motion and structural features in videos can inform segmentation; (iii) it is desirable to have smooth segmentation changes (often called temporal continuity in literature) from frame to frame, which is nearly impossible if frames are independently segmented; (iv) there is orders of magnitude more unlabeled or weakly labeled data available in video format, but annotated data is rare and prohibitively expensive to get, so weakly supervised and unsupervised methods become vastly more important to work on.

Two interesting examples are clockwork FCNs (2016) [33], and the CVPR 2018 paper **Low-Latency Video Semantic Segmentation** [19]. The latter has a *feature propagation module* that fuses features over time via spatially-variant convolutions, thus saving computation cost per frame, and an *adaptive scheduler* that dynamically allocates computation based on it's estimate of current segmentation accuracy. The two components work together to ensure low latency and high segmentation quality. They get competitive performance on Cityscapes and CamVid, and reduce latency from 360 ms to 119 ms (see figure 15). A summary of the pipeline is provided in figure 16.

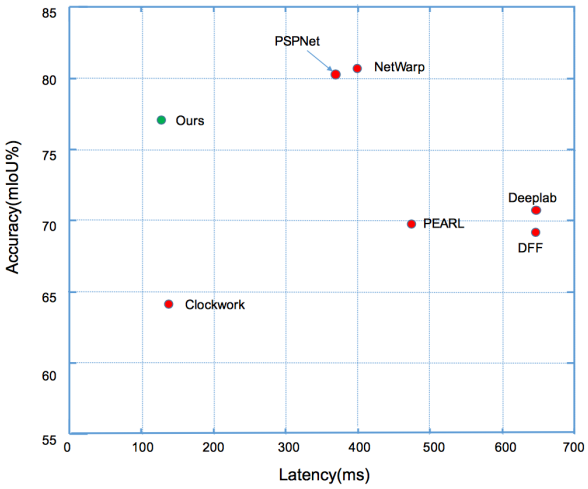

Figure 15. Latency vs accuracy on Cityscapes dataset.[19]

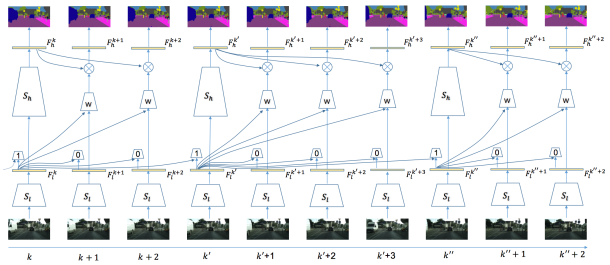

Figure 16. Overall pipeline: At each time step t, the lower-part of the CNN S_l first computes the low-level features ${F_l}^t$. Based on both ${F_l}^k$ (the low-level features of the previous key frame) and ${F_l}^t$, the framework will decide whether to set it as a new *key frame*. If yes, the high-level features ${F_h}^t$ will be computed based on the expensive higher-part S_h; otherwise, they will be derived by propagating from ${F_h}^k$ using spatially variant convolution. The high-level features, obtained in either way, will be used in predicting semantic labels. [19]

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence 39, no. 12*, 2017.

[2] S. M. Bileschi. Streetscenes: Towards scene understanding in still images. *MASSACHUSETTS INST OF TECH CAMBRIDGE*, 2006.

[3] G. J. Brostow, J. Fauqueur, and R. Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters 30, no. 2*, 2009.

[4] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. *ECCV*, 2008.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *ICLR*, 2015.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint*, 2016.

[7] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint*, 2017.

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. *CVPR*, 2016.

[9] G. Csurka and F. Perronnin. A simple high performance approach to semantic segmentation. *BMVC*, 2008.

[10] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[11] M. Gamal, M. Siam, and M. Abdel-Razek. Shuffleseg: Real-time semantic segmentation network. *arXiv preprint*, 2018.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research 32, no. 11*, 2013.

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

[14] H. Hattori, V. N. Boddeti, K. Kitani, and T. Kanade. Learning scene-specific pedestrian detectors without real data. *CVPR*, 2015.

[15] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. *CVPR 2018 Workshop on Autonomous Driving Challenge*, 2018.

[16] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *Robotics and Automation (ICRA)*, 2017.

[17] P. Krhenbhl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *NIPS, pp. 109-117*, 2011.

[18] . Ladick, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and crfs. *ECCV*, 2010.

[19] Y. Li, J. Shi, and D. Lin. Low-latency video semantic segmentation. *CVPR*, 2018.

[20] G. Lin, A. Milan, C. Shen, and I. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, P. D. Deva Ramanan, and C. L. Zitnick. Microsoft coco: Common objects in context. *ECCV*, 2014.

[22] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. *CVPR*, 2009.

[23] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[24] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv*, 2016.

[25] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun. Large kernel matters–improve semantic segmentation by global convolutional network. *CVPR*, 2017.

[26] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. *CVPR*, 2016.

[27] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. *ECCV*, 2016.

[28] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention, pp. 234-241. Springer, Cham*, 2015.

[29] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[30] B. C. Russell and A. Torralba. Labelme. *IJCV*, 2008.

[31] F. Schroff, A. Criminisi, and A. Zisserman. Object class segmentation using random forests. *BMVC*, 2008.

[32] N. Sebastian and C. H. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision 6.34*, 2011.

[33] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. *ECCV*, 2016.

[34] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[35] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[36] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. *ECCV*, 2012.

[37] P. Sturgess, K. Alahari, L. Ladicky, and P. H. Torr. Combining appearance and structure from motion features for road scene understanding. *BMVC*, 2009.

[38] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint*, 2018.

[39] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[40] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. *arXiv*, 2017.

[41] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[42] S.-C. Zhu, C.-E. Guo, Y. Wang, and Z. Xu. What are textons? *IJCV*, 2005.