ACTIVE REINFORCEMENT LEARNING FOR THE SEMANTIC SEGMENTATION OF IMAGES CAPTURED BY MOBILE SENSORS

MAHYA JODEIRI RAD

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

GRADUATE PROGRAMME IN EARTH AND SPACE SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

DECEMBER 2022

ABSTRACT

Neural Networks have been employed to attain acceptable performance on semantic segmentation. To perform well, many supervised learning algorithms require a large amount of annotated data. Furthermore, real-world datasets are frequently severely unbalanced, resulting in poor detection of underrepresented classes. The annotation task requires time-consuming human labor. This thesis investigates the use of a reinforced active learning as region selection method to reduce human labor while achieving competitive results. A Deep Query Network (DQN) is utilized to identify the best strategy to label the most informative regions of the image. A Mean Intersection over Union (MIoU) training performance equivalent to 98% of the fully supervised segmentation network was achieved with labeling only 8% of dataset. Another 8% of labelled dataset was used for training the DQN. The performance of all three segmentation networks trained with regions selected by Frequency Weighted Average (FWA) IoU is better in comparison with baseline methods.

ACKNOWLEDGEMENTS

I have so many people to thank for not only being a part of this journey, but for being a part of my life. This work would not be possible without their support.

Firstly, I would like to extend my sincere gratitude to Dr. Costas Armenakis as my supervisor throughout my Master's program. Being exposed to the research world, I have always been provided with immersive support and endless encouragement from Dr. Armenakis. Appreciation is also granted to Dr. Armenakis for his mental and physical care for me, especially during the Covid-19 lockdowns, which were brutally hard for those living alone. In addition, I would like to thank Dr. Gunho Sohn, my committee member, for his critical suggestions on my research work. I am very humble and thankful to my exam committee members for their acceptance to assess and add value to my research.

I would like to thank all the colleagues of Armenakis's lab, Sowmya Natesan, Agata Szeremeta, and Evangelos Bousiais, for their support and guidance upon my arrival in the lab. It was a pleasure to work with them. I would like to extend my appreciation to my other colleagues and faculty in Earth and Space Science Department.

Additional thanks to Aman Usmani and Aleena Qureshi for their constant support and friendship. I would like to thank Zahra Arjmandi, who always inspired me with her calm personality and insightful opinions. I would like to also extend my appreciation to Mahta Mansouri and Ali Jalalifar, two of my best friends ever since I stepped into Canada, their support, love, and help, made dark days brighter. I cannot thank my dearest friend, Maryam Jameela, enough for her endless love and care. Going through the past two years,

was only possible by having her by my side. Additionally, I would like to thank Vahid Saberi, for his love, care, and guidance in hardest times.

Last but not least, I would like to also thank my family for their endless support and love. I would like to express my deepest gratitude to my parents, Farzaneh Parsian and Jalil Jodeiri Rad, who has always been there for me with their unconditional love and made this possible. I would like to thank my dear brother Ata Jodeiri Rad who was there for me, supported me unconditionally, and upheld me in hard times. I cannot express how thankful I am to have my brother as my best friend. I give endless thanks to my aunt, Sanaz Parsian, one of the strongest women in my life, who has always inspired me with her character and courage. Additionally, I need to thank my grandmother Mansoreh Zargarsalehi and my aunt and uncle, Amir Parsian and Sima Gerami, for their endless love and support from long distances. Also, special thanks to my dear cousins, Asal Parsian and Sara Mortazavi, for always being there to make me smile, even through our video chats.

Needless to mention that I would not have been the person that I am today without these amazing people in my life.

TABLE OF CONTENTS

Table of Contents

LIST OF ACRONYMS

AI	Artificial Intelligence
AL	Active Learning
ASPP	Atrous spatial pyramid pooling
BR block	Boundary Refinement block
CNN	Convolutional Neural Network
CRF	Conditional Random Field
DCNN	Deep Convolutional Neural Network
DNN	Deep Neural Network
DQN	Deep Query Network
DRL	Deep Reinforcement Learning
FCN	Fully Connected Network
FCN	Fully Convolutional Network
FoV	Field of View
FPN	Feature Pyramid Network
FWA IOU	Frequency Weighted Network
GPU	Graphics Processing Unit
HD map	High-Definition map
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
LSTM	Long Short-Term Memory
MDP	Markov Decision Process
MIoU	Mean IoU
MLP	Multi-Layer Perception
PSPnet	Pyramid Scene Parsing Network
RALis	Reinforcement Active Learning for image segmentation
ReLu	Rectified Linear Unit
ResNet	Residual Network

RL	Reinforcement Learning
SGD	Stochastic gradient descent
SMOTE	Synthetic Minority Over-sampling Technique
SSD	Single Shot Detection
TD	Temporal Difference

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

Chapter 1 Introduction

1.1 Background and motivation

Scene understanding and image interpretation are critical to various machine vision tasks like medical image analysis (Gu et al., 2019; Hesamian et al., 2019), autonomous driving (B. Chen et al., 2017), and augmented reality (Guan et al., 2020). Semantic segmentation is an essential process in scene understanding as it generates classified raster regions of the input images, which in turn supports the generation of semantic maps. Semantic maps are one of the layers of high-definition maps, which are essentially precise and contain accurate information about the environment. For example, the rapid developments in the industry of self-driving cars have made the need for High-Definition maps more pressing than ever. High-Definition maps are, by definition, a 1:1 set of three-dimensional, exceedingly precise maps with centimeter-level accuracy. Another important reason for the necessity of these maps is the minimal tolerance for failure in some

circumstances involving self-driving vehicles. A minor error or delay on the map could have severe consequences and jeopardize human life. As a result, the demand for HD maps is unquestionable.

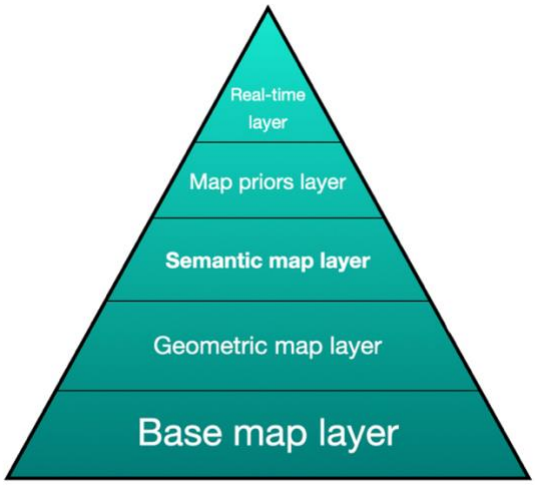


Figure 1.1. High-definition maps contain five layers

High-definition maps are composed of five layers (Figure 1.2): the base map, the geometric map, the semantic map, the map priors, and the real-time layer. The geometric map layer consists of 3D objects generated by processing a 3D dense point cloud. Semantic map layer is produced on top of the geometric map layer by adding semantic objects. Semantic objects, whose identities are distinctly described, define the environment better and help with understanding the map and its information. Finally, the real-time layer is dynamically updated with real-time information from the map priors. This study, will

concentrate on making development of large-scale semantic maps practical. Image data collected rapidly from mobile image sensors could be used to produce high-resolution maps. The critical challenge is converting these image data to high-precision information.

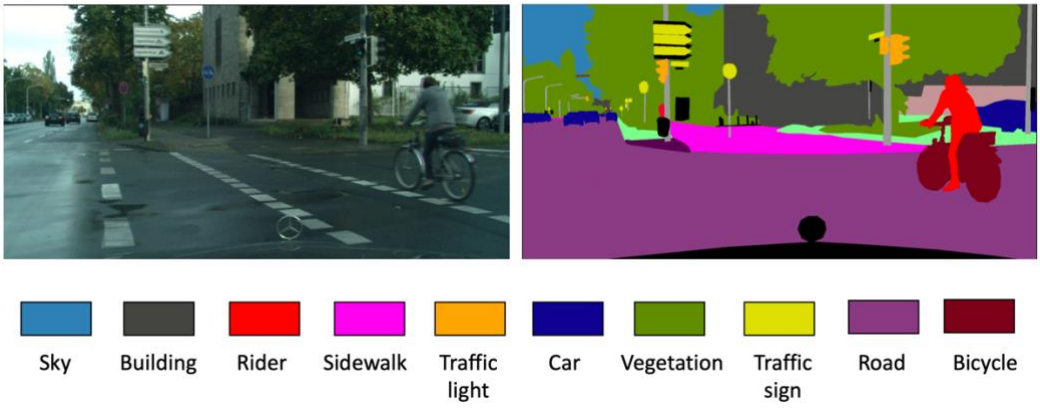


Figure 1.2. A semantic segmentation map from street view (Cityscapes dataset)

In the image segmentation process, classes are defined, containing the objects and things in a scenery. For instance, in an image urban area, classes like buildings, cars, humans, streets, and sky could be defined (Figure 1.1). Then each pixel is annotated to one of the defined classes. Deep learning techniques have enabled many aspirations in computer vision to come true. However, they come with their own challenges and limitations.

1.2 Problem definition and motivation

Machine learning is a subfield of artificial intelligence that is widely described as a machine's ability to imitate intelligence. Supervised learning is an algorithm that learns from a training dataset, with the dataset serving as a guide to the learning process. The

development of machine learning algorithms can result in the precise prediction of segments in an image. However, to achieve a precise prediction, the supervised learning model should be trained on the same or similar dataset to the one we need to be predicted. This not only comes with a great price computationally, but also it is very labor-intensive. Because, we need huge high-quality dataset for each task and this process is mainly or completely manual.

The process of semantic segmentation with supervised learning, which has been extraordinarily successful, begins with data collection. Preprocessing those data, cleaning them, and creating training, validation, and test sets is the next step. On average, an "Oracle agent" (e.g., human operator, software) would spend over an hour and a half on creating a standard dataset (depending on the size and details of the images, these statics may vary).

Nowadays, collecting this data happens automatically. But, when some variables, such as the sensor and the inner parameters or classes of objects in the environment, vary or the circumstances change, the model trained -the knowledge learned- on one known sampled dataset might not lead to a good prediction on the other datasets. As a result, each type of collected data necessitates establishing a new training set to generate precise predictions. Even while some transfer learning and data augmentation techniques may aid in the use of trained model knowledge, most of the time, fine-tuning is still required to achieve acceptable results.

An imbalanced number of pixels in datasets (i.e., some categories take a more significant portion of images compared to others) is another challenge for semantic segmentation. For instance, in an image with street views, context categories like buildings

and streets will contain almost twice the pixels compared to categories such as humans and trees (Figure 1.3). Therefore, the prediction accuracy compared to categories that contain a larger portion of the image would likely be smaller. This is an important issue because these smaller underrepresented categories (e.g., humans, cars, traffic signs) are more critical for applications such as self-driving cars.

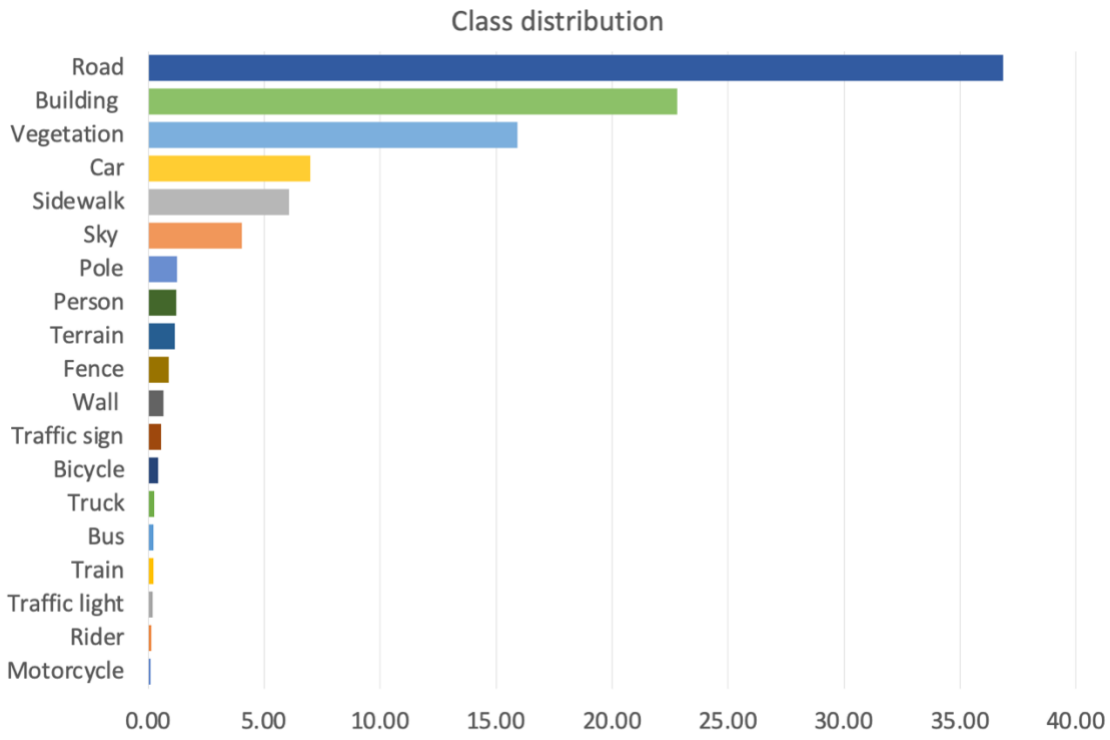


Figure 1.3. Imbalance class distribution in the Cityscapes dataset.

Another persistent concern in semantic segmentation with supervised learning is that some categories are substantially more plentiful by default than others, biassing network performance to the more represented ones. For example, in the urban area, the category of buildings, sky, or roads typically dominates the image. In contrast, categories

such as street signs or persons represent a minor number of pixels. Due to the computational expense, as we convolute images during training, these small classes in images shrink even more. The imbalance is visible in the Cityscapes dataset (Cordts et al., 2016), which is based on street views. In this dataset of 19 classes, the six most underrepresented classes account for less than 2% of the pixels in the training dataset. In contrast, a class-like road takes up more than 36% of the pixels in the training set. This disparity indicates the disparity in the model's performance (Figure 1.3).

In many cases, like self-driving cars, these underrepresented classes are more critical than others. FCN (fully convolutional network) is one of the well-known architectures for the task of supervised semantic segmentation (Long et al., 2015). Figure 1.4, represents the contrast in the performance of the network based on the mean intersection over union (MIoU) on the Cityscapes dataset. It is evident that while some classes, like road and sky, perform over 90 percent, the six underrepresented classes perform less than 50 IoU.

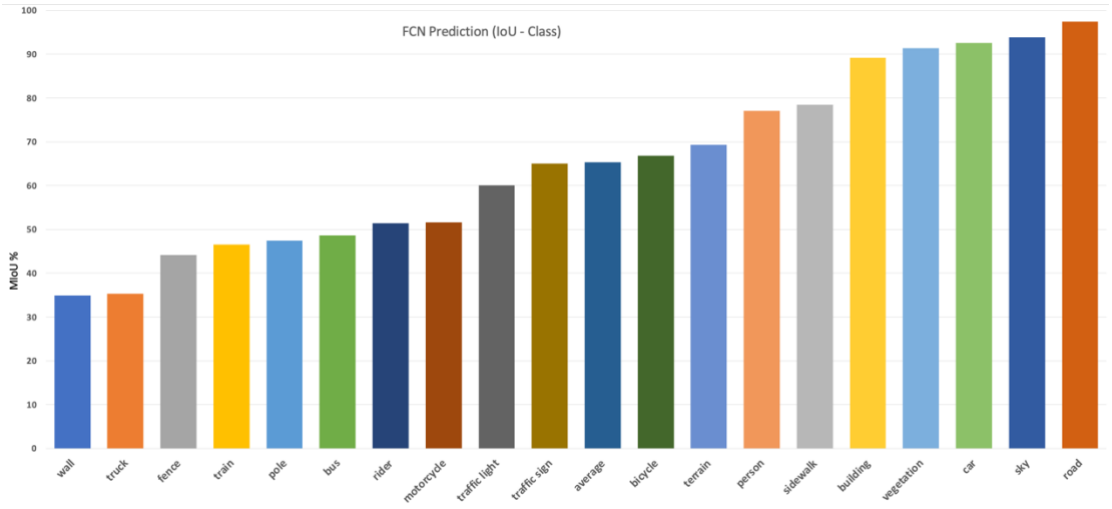


Figure 1.4. The performance of FCN (fully convolutional network) on each class of the Cityscapes dataset was evaluated by MIoU (mean intersection over union).

1.3 Research objective

The simplest way of addressing the problem of time-consuming labeling would be choosing random regions from images to be labeled. However, then we do not choose these regions smartly to contain the most informative regions from the image. Since it is random selection, the performance would also not be stable and reliable.

Generally, for addressing class imbalance problems in computer vision field, some techniques are suggested. For example, accuracy is not a good choice of evaluation metrics when we deal with class imbalance. In some cases, in data preparation over-sampling and under sampling is done.

We choose region-based approach in order to address these problems. By using this approach, the oracle in charge of labeling the images, labels only selected regions from the images. And by using a Reinforcement learning technique, we choose these regions in a

way that they contain more of underrepresented classes. In another world, we use a reinforcement active learning technique to over-sample the underrepresented classes.

Active learning is a form of semi-supervised learning that addresses the need for a vast labeled dataset by actively selecting a subset of the data to be annotated by an "oracle", which could be a human operator or a software. The approaches have been shown to be helpful in lowering training size while maintaining same level of performance. To perform the image classification challenge, (Joshi et al., 2009) devised a method based on uncertainty sampling. Later, for the same problem, an adaptable active learning strategy was presented (Li and Guo, 2013). This strategy coupled information density and uncertainty metrics with the selection of critical occurrences to be labeled.

To perform the task of Active Learning, a reinforcement learning approaches were applied. Reinforcement learning is a computational method in which an agent learns to act in an environment based on its interaction with the environment. The agent gets trained through trial-and-error experience to reach a goal while maximizing the reward it obtains from taking action in the environment.

1.3.1 Methodology

Based on a deep reinforcement learning algorithm, we offer a new reinforced active learning technique. In this research a modified Deep Q-Learning formulation for active learning is being proposed. An agent learns the method of selecting a collection of small image regions from an unlabeled data pool. These regions provide the segmentation networks with the most knowledge by choosing from underrepresented classes. The area

of selection is determined by the assumptions and segmentation model uncertainties used during training (Figure 1.5).

The objective is to train a segmentation network by only annotating a small amount of data. Therefore, training dataset is generated in an iterative manner using an active reinforcement learning approach. The iterative process starts with a small pool of training image datasets that train an image segmentation network. Then a small number of unlabeled datasets are chosen to be labeled. The labeled data go through the segmentation network; the trained segmentation model is now evaluated on a subset of data to create state and reward signals for the deep query network (DQN). The state and reward are calculated based on predictions and uncertainties of the segmentation model being trained. Then based on this feedback, an agent (e.g., a model) in the query network follows a strategy and executes a function to select a small informative region of the image to be labeled from a pool of unlabeled data. The actions taken based on a specified strategy (policy) are to select regions (states) that are assessed to be more informative by the query network.

The query network's chosen regions are annotated by the operator acting as an oracle, and these annotated regions are added to the training set. The segmentation network is now trained again on the updated training set and then evaluated on the reward set. The improvement in the segmentation network's performance on the reward set in one iteration compared to the previous one provides the query network agent with a signal. This signal indicated how well the DQN did in selecting the most informative regions of the images.

The performance improvement was measured by the performance metric, mean Intersection over union.

This loop is repeated until the operator labels a predetermined budget of the unlabeled data. The budget is the number of regions the DQN could choose for training the segmentation network. Finally, when an optimal strategy/policy is achieved by the query network, that policy is used to choose a number of regions that will assumingly need to be labeled in a novel dataset. Then the pretrained segmentation network is trained with these regions, and its performance will be evaluated with the mean Intersection over Union (MIoU) on the validation set to detect the underrepresented class regions as well.

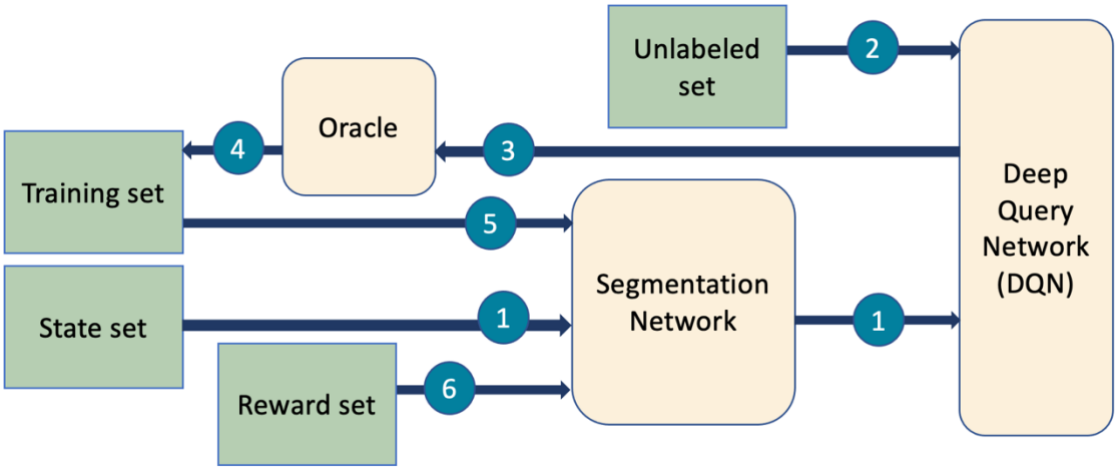


Figure 1.5. The architecture of the end-to-end active reinforcement learning method.

The steps of the active reinforcement learning method are as following (Fig. 1.5):

1) The state is calculated as a result of the performance of segmentation networks on the state set.

2) From the unlabeled set, a number of unlabeled regions are sampled evenly. The segmentation network computes the representation of their possible sub-actions.

3) Query network selects regions from pool of regions, known as pool of sub-actions, as action.

4) Labeled regions are selected and added to the labeled dataset (and removed from unlabeled set).

5) The segmentation network is trained using the new training dataset.

6) The reward is then computed depending on the performance of the newly trained segmentation network (with selected regions) on the reward set compared to performance of the network that had been trained with regions selected from previous step.

This cycle is repeated until a budget *B* of labelled regions is reached.

1.3.2 Contributions

As mentioned before, the key problems are identified with fully supervised semantic segmentation models and a data-driven region-based reinforcement active learning approach is utilized for segmenting images captured by mobile sensor. The active learning process was carried out by a reinforcement learning agent converging on a policy that selects the action with the highest calculated rewards most of the time (ε-greedy approach). The optimal policy would select the most informative samples from a pool of small regions of the image in order to reduce annotation labor while maintaining the same level of performance. The contributions of this study are as follows:

- Addressing the requirement of a significant amount of data for training Deep Neural networks for the task of semantic segmentation by proposing the use of an active reinforcement learning approach in labeling the dataset.

- The problem of underrepresented classes was addressed even further by adopting a Reinforced Active Learning for image segmentation (RALis) strategy with a frequency-weighted average IoU (FWA IoU) score to reward the DQN network. We expect that by putting greater weight on underrepresented classes, we can improve the network's performance in these areas even more. The proposed method's performance is tested on one of the well-known semantic segmentation datasets, the Cityscape dataset, and report the mean IoU score compared to different baselines while training with varying amounts of data, including a supervised method.
- The performance of FWA RALis method is evaluated and analyzed on three segmentation networks. The Feature Pyramid Network (FPN) (Lin et al., 2017), DeeplabV3 (Chen et al., 2017), and DeeplabV3+ (Chen et al., 2018) with ResNet-50 (He et al., 2015) backbone network architectures are chosen as our segmentation model, considering that they are all sophisticated state-of-the-art semantic segmentation models.
- Executing extensive experiments and analyses to evaluate the performances of the different model architectures.

1.4 Thesis organization

This thesis is organized into six chapters. Chapter 1 introduces the motivation of our study and clarifies the problem domain and the research objectives and contributions

of our work. Chapter 2 provides reviews of relevant research work and open-source datasets for semantic segmentation. In chapter 3, relevant background, including segmentation models, transfer learning, active learning, and reinforcement learning, have been reviewed. Chapter 4 explains the proposed end-to-end reinforced active learning methodology and the Query Networks architecture. In chapter 5, the details of the experiments, results, and analysis are presented. Finally, in Chapter 6, we draw the conclusions of this study and indicate future works.

Chapter 2 Related Work

Our research is related to deep learning methods for semantic segmentation (section 2.1), the active learning methods for choosing data to be labeled (Section 2.2), and more precisely reinforcement active learning methods (sections 2.3). At the end of the chapter, we review the available relevant datasets (section 2.4).

Our research was motivated by existing work in deep learning and active reinforcement learning. The chapter provides reviews on the advantage and disadvantages of each state-of-the-art segmentation method. Also, we review the general problems using only the deep learning method.

Then, we review the existing active learning methods. After gaining knowledge on the inadequacies of active learning methods based on only entropy, we study the active learning methods which used reinforcement learning procedures as acquisition functions. Since there are not any reinforced active learning models suggested for semantic segmentation tasks, we review the methods for the relevant task of classification. In the end, we briefly review some of the popular datasets for the semantic segmentation task to understand the reason we chose the GTAv and the Cityscapes datasets for the experiments.

2.1 Deep Learning for Semantic Segmentation

We will use a deep learning model for the task of semantic segmentation. Hence, the basic concepts of semantic segmentation, deep learning in computer vision and specifically semantic segmentation alongside with the state-of-the-art methods in this field will be briefly reviewed in this section.

Figure 2.1. Semantic segmentation versus instance segmentation (image from Sik-Ho Tsang , 2018)

In general, there are two forms of segmentation: semantic and instance. As displayed in Figure 2.1, the semantic segmentation assigns class labels to each pixel of the image (in the middle of the figure, image pixels are labelled according to their categories). The instance segmentation not only produces pixel-wise labels but also predicts instance-aware labels, which distinguishes the individual objects (different chairs are separated using labels of different colors in the right part of the Figure 2.1). There is also panoptic segmentation, which is a merged form of two basic segmentation procedures. In this work we will concentrate on semantic segmentation task for images acquired by mobile sensors.

Fukushima used the first convolutional neural network (CNN) in 1979. The convolutional network typically have three types of layers (Fukushima, 1980).

The convolutional layers, which convolve a kernel (or filter) of weights to extract features. The nonlinear layers, which apply an activation function to feature maps (usually element-wise), allowing the network to model nonlinear functions. And the pooling layers, which reduce spatial resolution by replacing small neighborhoods in a feature map with statistical information about those neighborhoods (mean, max, etc.). Then the concept of Backpropagation was introduced later, it is a method which efficiently calculates the gradient of the loss function with respect to the network weights for a single input-output example.

One of the persisting problems that most CNN models had was overfitting. Visual data is very complex. Therefore, the models tend to have a high dimension of input and have lots of parameters to fit. Without access to a big dataset, overfitting tends to happen. Hence, the algorithms will not generalize well. To address the problem of recognizing most objects in the world and overcoming the bottleneck of overfitting, scientists started a project called ImageNet. The dataset contains 15 million images organized in about 22,000 categories. Later the deep convolutional neural networks thrive even more with the realization that Graphics processing units (GPUs) can be used for general-purpose that require complex, simultaneous calculations.

The mentioned discoveries, set the way for CNN's continued success in a variety of high-level computer vision tasks. Additionally, it prompted researchers to investigate the potential of such networks for pixel-level classification challenges such as image segmentation. It is important to mention that, the supervised learning methods are highly

dependent on the quality of dataset. If the training dataset lack quality, it will directly reflect on the performance of the network.

2.1.1 State-of-the-art Deep Learning Models for Semantic segmentation

VGG network was introduced by Oxford in 2014, it had 19 layers (Figure 2.2), and it became one of the most successful networks in the ILSVRC competition. In previous networks, derivatives focused on reduced window sizes and strides in the first convolutional layer. VGGNet addresses another critical feature of convolutional neural networks (CNNs), which is the depth (Simonyan and Zisserman, 2015). VGG's convolutional layers have a relatively narrow receptive field (3x3, the smallest size that still captures left/right and up/down). There are other 1 by 1 convolution filters that perform a linear transformation of the input before being followed by a ReLu (rectified linear unit). Because of the small size of the convolution filters, VGG can have a considerable number of weight layers; of course, more layers lead to better performance.

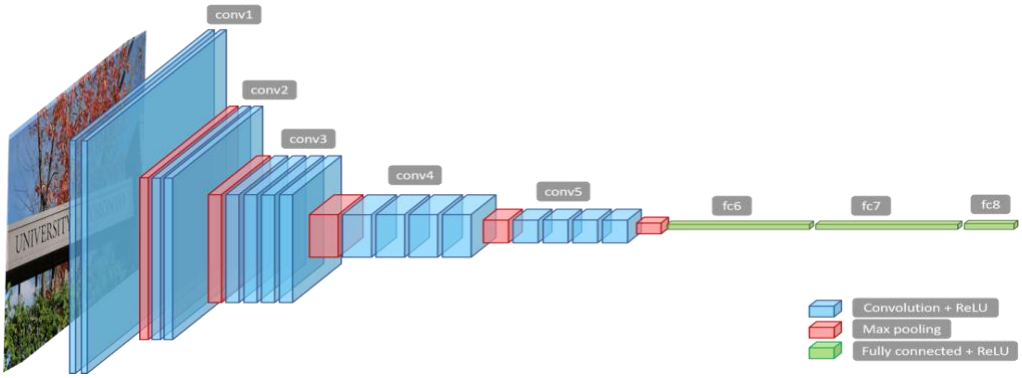


Figure 2.2. VGG-19 architecture (image from Ferguson et al., 2017)

In this time there was a realization that depth contributes to higher performance on the networks, the networks get deeper and deeper each year on ILSVRC. In 2015 Microsoft Research Asia released a model named residual networks (ResNet), which was 152 layers. The researchers tried to go deeper for better results, but GPU memory limitation stopped them from going deeper than 200 layers of (He et al., 2015).

ResNet had the least error in the ILSVRC challenge even in comparison to a human operator, and it comes first in so many other challenges. ResNet has many variants depending on the number of neural network layers that had been the number of neural network layers used (ResNet-34, ResNet-50, ResNet-101, and ResNet-152). The additional paths are proven to be beneficial for the convergence of the model. The ResNet used short skip connections, the connection between sequential convolutional layers with the same input dimension (Extensively studied in section 3.1.4).

Semantic segmentation aims to extract features before using them to segment an image into several segments. However, because of the max-pooling layers, the size of the image is lowered as it passes through the network in convolutional networks. To efficiently segment the image, we must up-sample it by using an interpolation technique that utilizes deconvolutional layers. Models like VGG-19, ResNet, and Inception had been trained on large-scale datasets like ImageNet; they learned the feature extraction task. Since many researchers do not have enough resources and time to train networks on such massive datasets, they transfer the knowledge these networks have gained and use it for their own applications. This is called transfer learning, which is defined as the act of transferring as much knowledge as possible from an existing model to a new model designed for a similar

but not the same task (Further studied in Section 3.1.4). Many architectures use mentioned trained models as an encoder or the backbone of their network.

FCN is a network with several convolutional layers with no fully connected layers. The FCN is super expensive for semantic segmentation tasks because we are applying several convolutions that all keep the same spatial size of the input image (Long et al., 2015). For semantic segmentation tasks, FCN are designed as several convolutional layers with down-sampling and up-sampling of the maps inside the network. By adding up-sampling layers to a standard convolution network, FCN recovers spatial information from down-sampling levels. They proposed a skip architecture (shallow fine layer) that combines semantic information from a deep coarse layer with appearance information to obtain precise and detailed segmentation. The primary aim was to redesign and fine-tune a classification model (image classification) to learn efficiently from entire image inputs and ground truths, extending these classification models to segmentation. This is computationally very efficient because you can make the network deeper and work at lower spatial resolution for many layers of the network.

The need for these methods for the large labeled datasets was an obstacle to some semantic segmentation applications like segmenting medical images. The U-Net modified FCN with a similar design to an encoder-decoder. The former is used to extract features through down-sampling, while the latter is used to up-sample the extracted features through deconvolutional layers. The sole difference between the FCN and U-Net is that the FCN up-samples using the final extracted characteristics, whereas U-Net employs something called a shortcut connection (Ronneberger et al., 2015).

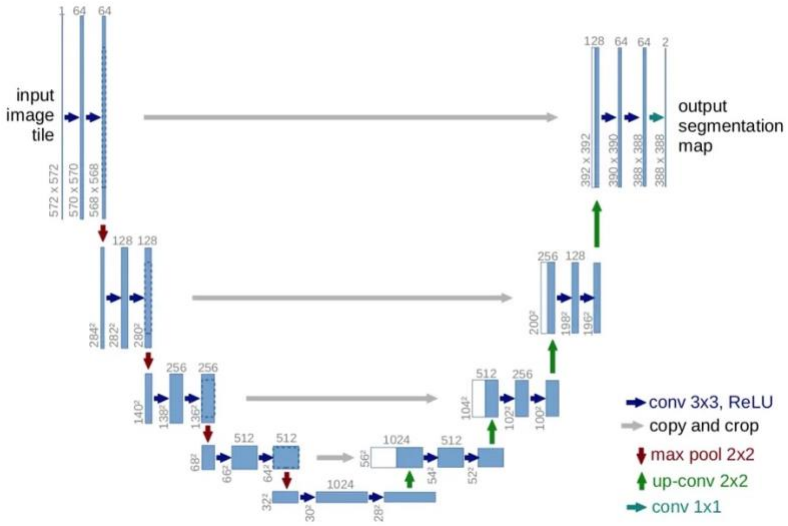


Figure 2.3. U-Net architecture (image from Ronneberger et al., 2015)

The architecture is divided into three parts: I) the down-sampling pipeline, which consists of four stages and mainly uses Resnet101 as the backbone. As illustrated in Figure 2.3, each step essentially employs two 3 by 3 convolutions with batch norms followed by two 2 by 2 max-pooling. Two 2 by 2 up-convolutions follow two 3 by 3 convolutions to form the horizontal bottleneck. II) The up-sampling approach likewise has four phases, which are depicted as a decoder with two 3 by 3 convolutional layers followed by 2 by 2 up-sampling. At each stage, the feature maps are cut in half. To give local and global information during up-sampling, the model makes long skips connections between up-sampling and down-sampling paths. III) Finally, the segmented output is provided by the output 1 by 1 convolutional layer, where the number of feature mappings is proportional to the number of desired segments. This approach of concatenating the information from

various blocks enables U-Net to yield finer and more accurate results. U-Net additionally employed a loss weighting technique for each pixel, so that the weight towards the boundary of segmented objects is higher. This loss weighting approach assisted their U-Net model in discontinuously segmenting cells in biomedical images.

Later, the Feature Pyramid Networks (FPN) was presented. The model contains a multi-scale pyramidal hierarchy of deep convolutional networks that generate feature pyramids with semantics at all levels. It is used to replace featurized image pyramids at a low cost (power, speed, and memory). The fundamental distinction between FPN and U-Net is that FPN has many prediction layers, one for each up-sampling layer (Lin et al., 2017). The FPN, like the U-Net, contains lateral connections between the bottom-up pyramid (left) and the top-down pyramid. U-Net copies and appends the features, while FPN applies a 1 by 1 convolution layer before appending. This permits the bottom-up pyramid known as the "backbone" to be any model (more details in Section 3.1.1).

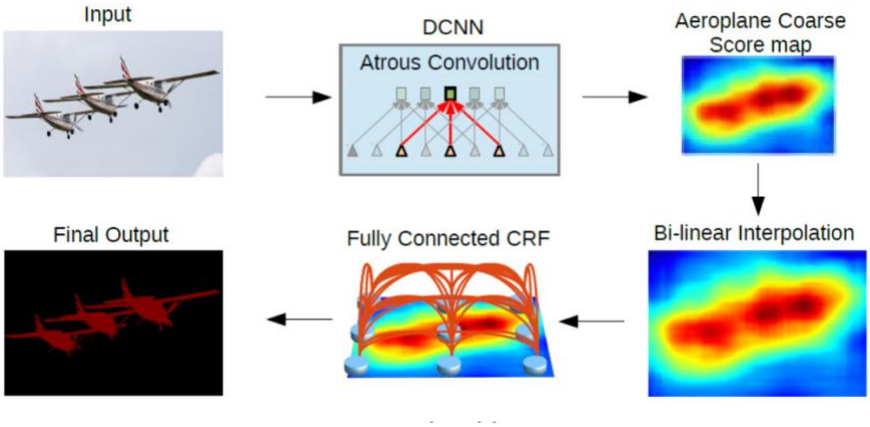


Figure 2.4. DeepLab architecture (image from Chen et al., 2017)

DeepLab use Atrous convolution and fully connected CRFs (Conditional Random Fields) methods (Figure 2.4) (Chen et al., 2016). The recurrent use of max-pooling and striding at successive layers of deep convolutional neural networks (DCNNs), decreases the spatial resolution of the generated feature maps dramatically. DeepLab implements atrous convolution for up-sampling, a convolution with defined gaps known as atrous convolution (or dilated convolution). DeepLab also utilizes features yielded by the last convolutional block before up-sampling it, as opposed to U-Net, which uses features from every convolutional block and then concatenates them with their corresponding deconvolutional block.

Inspired by FPN, later Chen et al. introduced the DeepLabV2 model in 2017. The additional feature of DeepLabV2 was that it used Atrous Spatial Pyramid Pooling (ASPP) which further improved the performance of DeepLab to represent the object on multiple scales. Additionally, DeepLabV2 used ResNet as the backbone in addition to VGGNet. Atrous Spatial Pyramid Pooling (ASPP), is a semantic segmentation module for resampling a given feature layer at multiple rates before convolution. This amounts to probing the original image with various filters that have effective complementary fields of view, thus capturing objects and useful image context at multiple scales.

In DeepLab and DeepLabV2, CRF was used to take the rough segmentation results of neural networks and refine the boundaries. Later the DeeLabV3 adopted an encoder-decoder architecture with atrous convolution; this adoption resulted in sharper boundaries of classes and objects while capturing a high semantic information (Chen et al., 2017). Additionally, the DeeplabV3+ network incorporates an encoder-decoder architecture to

improve the DeeplabV3 architecture even further (Chen et al., 2018). We will explore these two architectures in section 3.1.2 and 3.1.3.

ParseNet is another state-of-the-art method for semantic segmentation, which puts the spotlight on the fact that FCN cannot represent the global context information (Zhao et al., 2017). Contextual representation of images has been shown to improve performance on segmentation tasks.

Even though development of reviewed architectures achieved high performance on semantic segmentation task, still the class imbalance problem stayed one of the core problems in the way of achieving even better results. As mentioned before, the class imbalance problem in semantic segmentation with supervised learning methods usually stems from the nature of the data. In each image, because of the objects' size and perspective, there are classes that occupy a small number of pixels in the picture. Since the network gets these images as input data, naturally, it would learn to segment the classes that occupy more pixels than the underrepresented classes. One way to address this problem is with a weighted loss function. As mentioned before, U-Net used the pixel-weighted loss function to achieve better borders on its specific binary segmentation task.

The most popular loss functions for segmentation are Dice loss, cross-entropy loss, or a combination of the two. Dice loss is based on the Dice coefficient, which is a measure of overlap between two samples (Yeung et al., 2022). This measure ranges from zero to one, where one means total overlap. This coefficient was initially developed for binary data. To use this coefficient as loss for the segmentation task, we can compute the union of prediction and ground truth as element-wise multiplication between the prophecy and

ground truth and sum the result. Because the target mask is binary, any pixels from our forecast which are not “activated” in the target mask effectively had been zeroed out. For the remaining pixels, essentially penalizing low-confidence predictions had been punished. The higher the value, the better the Dice coefficient (Jadon, 2020).

Cross entropy, (Equ. 2.1), has its roots in information theory and is a measure of the difference between two probability distributions for a given random variable or sequence of events. It is equivalent to the negative log-likelihood loss as a loss function. Here, $y^{(k)}$ is binary (0 or 1), indicating whether the class label $k(= 1{:}K)$ is the correct classification. $y^{(k)}$ is the ground truth and $\hat{y}^{(k)}$ is the prediction.

$$L(\hat{y}, y) = -\sum_{k=1}^{K} y^{(k)} \log \hat{y}^{(k)} \tag{2.1}$$

The weighted loss function is a neat way to address the class imbalance problem. The idea is to weigh the loss computed for different classes differently based on whether they belong to the abundant classes or the underrepresented classes. The goal is to assign a higher weight to the loss encountered by the samples associated with underrepresented classes. Both weighted Dice loss and weighted cross entropy are popular loss functions for segmentation models to deal with imbalanced datasets.

Another way to address the imbalance dataset was Synthetic Minority Over-sampling Technique (SMOTE). In this method which was first introduced for the classification task, they use the general approach of under-sampling the majority class and oversampling the minority class (Chawla et al., 2002). This method was introduced for

decision trees which was a more naïve machine learning method. By adding synthetic data to the minority class, they improved this specific algorithm's performance.

Following the synthetic data approach, another idea was raised to pretrain the networks on better-balanced synthetic datasets before training them with the real-world dataset. This approach addresses both the need for a large-scale dataset and an imbalanced dataset; however, considering the nature of the semantic segmentation problem, this approach does not eliminate the class imbalance problem. Even for the issue of huge data requirements, it could easily overfit while training on the real-world dataset, and it would not generalize well.

With all mentioned, state-of-the-art supervised learning methods still lack to address the problem of the need for huge datasets and the fundamental problem of imbalanced classes in the datasets for semantic segmentation tasks.

2.2 Active learning

Active learning is a form of semi-supervised learning, in which unlike the supervised learning methods only a small subset of human-labeled data is given to the machine learning algorithm. Collecting new data for training, particularly labeling and annotating them, is a labor-intensive operation. As a result, the active learning plays a critical role on spending our time and resources on gathering data that matters.

Also, as previously established, predictions are dependent not only on the architecture of the model but also on the specific data with which the model was trained. In problems where the prevalence of classes is imbalanced, it is necessary to prevent the

resultant model from being skewed toward the majority class and to ensure that the model is capable of reflecting the true nature of the minority class.

Another effect of class imbalance can be seen in domains where the ground-truth labels in the dataset are not available ahead of time and must be acquired on-demand at a cost. The expenses involved with label acquisition may be attributable to human labor or too costly incentives, interventions, or experiments, but at the end identifying all the datasets may not be feasible in these situations. In many cases, the goal is to guarantee that the budget is not primarily spent on labeling examples of the majority class and that the list of instances to be labeled contains a corresponding amount of minority class instances. Which also works in favor of the models.

Active learning is also very influential regarding the imbalance dataset problem; the importance of active learning (AL) in learning from imbalanced datasets can be regarded from two different perspectives. The first viewpoint analyses the situation in which all the labels for all the cases in a reasonably large, imbalanced dataset are readily available. In this situation, AL's function is to mitigate and potentially remove any negative consequences that the class imbalance may have on the model's generalization ability. The other viewpoint tackles the situation in which we have prior knowledge that the dataset is imbalanced and would like to use AL to select informative samples from both the majority and minority classes for classification, subject to budget limitations. The first viewpoint focuses on AL's ability to handle class issues. In contrast, the second perspective is concerned with the impact of class imbalance on the sampling performance of the AL.

Classic active learning focused on assessing sample informativeness through hand-crafted heuristics derived from sample uncertainty. Shannon entropy has been used as a factor of uncertainty (Shannon and Weaver, 1948). Later Freund et al. used a query-based method for filtering informative queries from a random stream of input data Fields (Freund et al., 1993). In another work, they used a Query By Committee approach and extended the committee-based paradigm to the context of the probabilistic classification (Argamon-Engelson and Dagan, 1995). In the method proposed by Roy et al., the future error was optimized by taking a Monte Carlo approach to estimate the expected reduction in error due to the labeling of a query (Roy and McCallum, 2001).

When utilized in an active learning setting, deep learning presents various challenges. First of all, active learning methods rely on the ability to learn and update models from small amounts of data. Deep learning methods, on the other hand, are known for their reliance on massive volumes of data. Second, while conventional AL acquisition algorithms rely on model uncertainty, deep learning methods do not usually represent model uncertainty. In a revolutionary work, researchers took advantage of Bayesian deep learning in the active learning framework (Gal et al., 2017). They developed an active learning framework for high-dimensional data, which was previously highly challenging. Using specialized models such as Bayesian convolutional neural networks, they demonstrate that these active learning techniques can be applied to image data, resulting in a considerable improvement over previous active learning method.

In another work, pre-clustering was used in an active learning framework to do the task of image classification. While it is typical practice in active learning to select samples

close to the classification boundary, better performance can be obtained by accounting for the prior data distribution (Nguyen and Smeulders, 2004).

Some researchers also address the problem of change detection in computer vision by deep active learning. They made use of the fact that the model is uncertain in some region of the input space Adding a tagged example in that region should significantly reduce that uncertainty (Růžička et al., 2020). In this work, they used a U-Net architecture for detecting the changes in remote sensing images with the least budget of labels possible. The entropy metric has been used as an acquisition function for choosing the most informative labels.

To improve AL performance, some approaches integrate various methodologies like exploration-exploitation trade-offs. To select query samples, all of the above-mentioned active learning systems use a variety of criteria based on uncertainty and diversity. Despite their widespread use, their performance is hampered in practice by various issues such as non-calibrated uncertainties, the insufficient trade-off between data exploration and exploitation, the prevalence of confirmation bias, and so on. To solve these issues, an Ask-n-Learn method was developed, which is an active learning strategy based on gradient embeddings obtained utilizing the pseudo-labels calculated in each algorithm iteration. More crucially, prediction calibration was pushed for in order to achieve credible gradient embeddings, and a data augmentation method was employed to mitigate the impacts of confirmation bias during the pseudo-labeling (Venkatesh and Thiagarajan, 2020).

Other researchers have used a bandit formulation for active learning. They prefer to rely on active learning by learning strategy rather than a human-designed strategy. The active learning method derives a reward scheme that closely relates to the performance measure of interest. This approach results in a clever probabilistic blending of the strategies subject to their time-varying performance (Hsu and Lin, 2015). In another work, researchers try to transfer the active learning experience. They used a linear upper-confidence-bound algorithm in their proposed model to update the weights of the contextual bandit. The biased regularization technique has been used to transfer the experience across various datasets (Chu and Lin, 2016). These solutions, however, are constrained in that they combine hand-crafted strategies rather than learning new ones. Active learning approaches that are more recent rely on an acquisition function.

In a work named "learning active learning from data," the authors suggested a data-driven active learning approach. The query selection procedure had been formulated as a regression problem not restricted to working with existing AL heuristics. Instead, the strategies are learned based on experience from previous AL outcomes. In this method, they estimate the error reduction of labeling a particular sample and select the ones that optimize the error reduction. This data-driven approach was a game changer in the active learning field (Konyushkova et al., 2017).

(Wang et al., 2017) proposed a cost-effective active learning approach for image classification, which surpasses the previously mentioned active learning methods in two ways. First, using a well-designed framework that integrates deep convolutional neural networks into active learning. In this framework, the feature representation and classifier

can both be updated at the same time be updated simultaneously with progressively annotated informative data. Second, by providing a low-cost sample selection technique for improving classification performance with fewer manual annotations. Unlike previous approaches, which focus solely on uncertain samples with low prediction confidence, many high-confidence samples are chosen from the unlabeled set for feature learning.

2.2.1 Active Learning for semantic segmentation

In recent years, active learning has been applied to more complicated scenarios like segmentation based on uncertainty sampling methods without feature projection. Owing to the traditional active learning methods being restricted when applying to large-scale datasets, the active learning research conducted on computer vision is not generalized. This basically means there are not many works in the field of deep-active learning for large-scale problems like semantic segmentation.

Jain and Grauman, (2016) incorporates metrics (based on hand-crafted heuristics) that promote labeled sample diversity and representativeness. They created a stage-by-stage active propagation strategy that actively decides the most valuable images for human annotation and then adjusts the foreground predictions in all unlabeled images accordingly. They offer an active selection approach that operates on the joint segmentation graph over all images to discover images that, once labeled, will propagate well to other examples. It emphasizes human involvement in images in the graph that are both uncertain and influential, as well as mutually different. Another active learning approach had been proposed to train a segmentation classifier. The classifier used an uncertainty sampling

technique, which accounts not only for the uncertainty of the prediction at a single location but also in its surroundings (Konyushkova et al., 2015).

In another work, Super-pixel based active learning method had been used for segmenting hyperspectral images. They had used the spatial information derived from super-pixels in active learning process. In first stage, a graph based super-pixel generation method had been utilized to over segment the images. In the next step, super-pixel boundaries had been used for computing gray-level co-occurrence matrix based on texture features. At last these spatial features had been fed into active learning loop (Guo et al., 2017). The problem with such works was that their performance was utterly dependent on the quality of the super-pixels.

Transferring from the feature of classic active learning, the scientists investigate the active learning for the handwriting digits classification and experiment on the MNIST dataset. As mentioned above, the AL methods carried out on classification task is diversified and achieve different results. (Gorriz et al., 2017) Constructed a framework combining training CNN for medical image segmentation and active learning based on uncertainty estimation using Monte Carlo sampling. He classified the sample into four categories by the goodness of the prediction and uncertainty and selected the uncertain and wrong predictions as the sample. The medical image segmentation he investigated is the 0-1 case, and the selection criteria he set are not precise enough.

Another work employed a similar idea from the previously mentioned work to multi-class scene semantic segmentation, which also used Monte Carlo Sampling for the information evaluation (Mackowiak et al., 2018). Moreover, he took the number of clicks

into account and designed a cost function that balanced the trade-off between the cost of annotation in his framework and informativeness. Mackowiak extended the sample selection criteria based on Gorriz's work and achieved a state-of-the-art result on multi-class segmentation. However, the number of clicks is not a standard parameter provided by the dataset and is not critical to consider in the general active learning framework. In this work, the cost of annotating an image is not the same for all images. They adopt a region-based strategy to deal with the enormous number of samples on a segmentation dataset. Their labeling technique is based on manually set heuristics, restricting the representability of the acquisition function.

In more recent work, it had been suggested to collaborate between a deep neural network and a human in the loop to rapidly get accurate segmentation maps of remote sensing images. In a nutshell, the agent interacts with the network iteratively to correct its initial incorrect predictions. These interactions, in concrete terms, are annotations that represent semantic labels. Two interactive learning strategies have been presented to incorporate user inputs into deep neural networks. The first concatenates the annotations with the inputs of her network. The second retrains the network using the annotations as a sparse ground-truth. Furthermore s, an active learning technique had been proposed which directs the oracle's annotations to the most relevant areas (Lenczner et al., 2022). Active learning has been done based on uncertainty and the goal is to refine the predictions by having an operator control the quality of prediction and expanding the labelled data.

2.3 Reinforced active learning

In recent years, reinforcement learning (RL) has been used as a method to find the policy which would optimize the performance of various algorithms. However, from the perspective of deep learning, reinforcement learning raises multiple obstacles. To begin, most effective deep learning applications have required massive volumes of hand-labeled training data. In contrast, RL algorithms must be able to learn from a scalar reward signal that is usually sparse, noisy, and delayed. When compared to the immediate correlation between inputs and targets found in supervised learning, the wait between actions and consequent rewards, which can be millions of timesteps long, appears particularly daunting. Another issue is that most deep learning algorithms presume that data samples are independent, yet it is common to encounter sequences of highly connected states in reinforcement learning. Additionally, as the algorithm learns new behaviors, the data distribution in RL changes, which might be problematic. Furthermore, when the algorithm learns new behaviors, the data distribution in RL varies, which can be troublesome for deep learning approaches that assume a stable underlying distribution. It is noteworthy to mention that there are not many works on reinforcement learning for semantic segmentation due to challenges in computing state. Therefore, we mainly review reinforcement learning methods that had been used in classification tasks which inspired this work.

In the work “Playing Atari by deep reinforcement learning,” Mnih et al. (2013) proposed a model which is a convolutional neural network trained with a variation of Q-learning; this work demonstrates how a convolutional neural network may overcome these

hurdles to learning appropriate control policies from raw video input in complicated RL contexts. The network is trained using a variation of the Q-learning technique, with stochastic gradient descent used to update the weights. An experience replay method was employed to address the issues of correlated data and non-stationary distributions. It randomly samples earlier transitions and smooths the training distribution over multiple previous behaviors with raw pixels as input and a value function forecasting future rewards as output.

In another work, Liu et al. (2018) leverage expert knowledge from oracle policies to learn a labeling policy. They used an imitation learning method that utilizes an effective algorithmic expert that provides the agent with good actions in the active lea ring situation. Then a feed-forward network learns the active learning strategy of mapping situations to the most informative data. Later, Bachman et al. (2017) leverage expert knowledge from oracle policies to learn a labeling policy. Meta-learning is used to train a model that learns active learning algorithms (Lake et al., 2019).

In some other works, we rely on policy gradient methods to learn the acquisition function of active learning. Similar to previous work, Pang et al., (2018) also used a meta-learning framework for active learning. Although learning the optimal criterion inside a generic function class is tempting, it does not provide a general solution to AL unless the learned criterion generalizes across varied datasets/learning problems. We can train an excellent query policy for a particular dataset using Deep Reinforcement Learning (DRL), but we need the dataset's labels to do so; and if we had those labels, we would not need to use AL in the first place. In this work, researchers examine ways to train AL query criteria

that generalize across tasks/datasets. This framework defines a Deep Neural Network (DNN) query criterion parameterized by dataset embedding.

Contardo et al., (2017) took a similar meta-learning approach. However, they used a pool-based setting, where the system observes all the examples of the dataset of a problem and has to choose the subset of examples to the label in a single shot. This differs from prior approaches in one-shot learning that considered a stream of instances to classify one after the other. Unlike this work that gathered all labeled data in one step, (Sener and Savarese, 2018) proposes that a batch of representative samples be chosen to maximize coverage of the unlabeled set. Unfortunately, the bounded core-set loss performs poorly when the number of classes increases.

Woodward and Finn (2017) proposed an active one-shot learning method where unlabeled images are provided one by one, and the decision is to label them or not. They combined meta-learning and reinforcement learning to develop an effective active learning method. In this framework, the deep recurrent model learns to make labeling decisions. They used LSTM (long short-term memory) as their action-value function (Hochreiter and Schmidhuber, 1997). Unlike this work which was stream-based active learning, In some jobs, (Konyushkova et al., 2019) used a pool-based active learning method in which unlabeled data is provided beforehand, and the decision is later taken on which samples to choose.

Ebert et al., 2012 used a Markov decision process (MDP) to construct a feedback-driven framework that learns the process during experience without the requirement for

prior knowledge (Lim et al., 2016). These strategies just generalize learning the task with less data, ignoring the issues caused by unbalanced classes.

Kampffmeyer et al., 2016 focused on the problem of an imbalanced dataset with remote sensing data. It evaluated the performance of various CNNs on small objects in the image. (Chen et al., 2016) employed a semi-supervised method that uses maximum square loss rather than reducing entropy to avoid relying on a simple strategy of selecting easy-to-transfer data. Konyushkova et al., 2019 developed a data-driven reinforced active learning technique for general use. Because their classification challenge is significantly simpler than the semantic segmentation problem, where DQN training has a higher computational cost. Mackowiak et al., 2018 suggested an approach that concentrated on selecting small sections from images to be labeled by humans to maximize network performance while lowering annotation effort.

The nature of the semantic segmentation tasks necessitates a significantly distinct conception of actions, states, and rewards. Furthermore, we must modify the DQN formulation to make the problem computationally practical, as it would be high memory-consuming. Casanova et al. (2020) developed a data-driven, region-based method to reduce the oracle's labeling effort in their approach to reinforce active learning for semantic segmentation. This solution tackled the class imbalance problem by using a mean intersection over union (MIoU) performance metric to evaluate the segmentation network's performance. The Query network is rewarded for selecting useful regions for training the segmentation network, which has the potential to increase the MIoU. Furthermore, with this region-based method, the model can learn to select regions from images that contain

the most informative data, which the segmentation model had previously seen less of. To increase the attention on imbalanced classes, we used a frequency weighted average IoU for rewarding the query network (Jodeiri Rad and Armenakis, 2022). Furthermore, with this region-based method, the model can learn to select regions from photos that contain the most informative data, which the segmentation model had previously seen less of. They chose Feature Pyramid Network (FPN) architecture (Lin et al., 2017a) as their segmentation network with the backbone of the ResNet-50 (He et al., 2016). Additionally, we examined the performance of our frequency weighted reinforcement active learning methods with two other state-of-the-art segmentation models: DeeplabV3 (Chen et al., 2017) and DeeplabV3+ (Chen et al., 2018).

2.4 Introduction to available datasets for semantic segmentation

Various open datasets are publicly available to evaluate the models for the semantic segmentation task. Here we will have an overview of some of the available street-scene datasets for semantic segmentation.

- **KITTI** is a prominent dataset for autonomous driving that contains videos of traffic scenarios taken with various sensor modalities (including high-resolution RGB, grayscale stereo cameras, and 3D laser scanners). Although the original dataset lacks ground truth for semantic segmentation, researchers have manually labeled parts of it; for example, Alvarez et al. constructed ground truth for 323 images (1242 by 375 pixels) from the road

recognition challenge using three classes: road, vertical, and sky (Geiger et al., 2013).

- **CamVid (Cambridge-driving Labeled Video Database)** is a database for understanding road/driving scenes that was first taken as five video sequences with a 960 by 720 resolution camera installed on a car dashboard. These sequences were sampled, totaling 701 frames. Then they were manually labeled into 32 classes; however, in many circumstances, researchers chose to employ 10 classes out of 32 Fields (Brostow et al., 2009).
- **Cityscapes** is a large-scale database focused on the understanding and interpreting urban street scenes. It offers semantic, instance-wise, and dense pixel annotations for 30 classes divided into eight categories (flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void). Like the previous dataset the researchers usually would prefer the version with 19 classes and choose to ignore 11 of the existing classes or map them to other similar classes. The dataset contains approximately 5000 finely annotated images. Data was collected in 50 cities over several months, during the day, and in good weather. Because it was initially captured as video, the frames were hand-picked to have the following characteristics: a large number of dynamic objects, a variable scene structure, and a varying background. The image size is 2048 by 1024 in the RGB channels (Cordts et al., 2016).

- The **SYNTHIA** dataset is a simulated dataset composed of 9400 photorealistic frames generated from a virtual city. Images are created by simulating different seasons, weather, and illumination conditions from multiple viewpoints. It provides semantic annotations at the pixel level for 13 classes. Each frame has a resolution of 1280 by 960 pixels (Ros et al., 2016).
- The **GTAv (Grand theft auto 5)** dataset contains 24966 synthetic images with semantic annotation at the pixel level. 12402 is spitted for the training set, 6347 images for validation, and 6217 images for the test set. The images were created with the open-world computer game Grand Theft Auto 5 and are all from the perspective of a car driving through the streets of American-style virtual cities. There are 19 semantic classifications that are interchangeable with those in the Cityscapes dataset, which makes this dataset a perfect one for pretraining the models before training them on the cityscapes dataset. Additionally, the image size is the same as the cityscape dataset as well (Richter et al., 2016).

Since the cityscapes dataset has a large number of classes alongside many imbalanced classes and it is also compatible with large-scale synthetic GTAv datasets, we will use these two open-source datasets for our research.

2.5 Summary

In this chapter, we present the related work to our research. First, the basic concepts of convolutional neural networks, deep neural networks, and semantic segmentation were introduced. Then we were familiarized with state-of-the-art semantic segmentation models. Later, we studied active learning methods and state-of-the-art active learning methods in semantic segmentation. Next, we review reinforcement learning approaches. Last, we present a number of publicly available datasets for semantic segmentation and choose to continue with GTAv and cityscapes datasets.

Chapter 3 Background

In this chapter, we first review the architecture of segmentation networks that has examined with our method alongside with the transfer learning concept and the backbone that has been used. The FPN, DeeplabV3, and DeeplabV3+ DNNs, which have been used in this framework with the backbone of ResNet-50, will be presented. Then we will briefly study the active learning and reinforcement learning methods to gain the background knowledge necessary to understand our methodology.

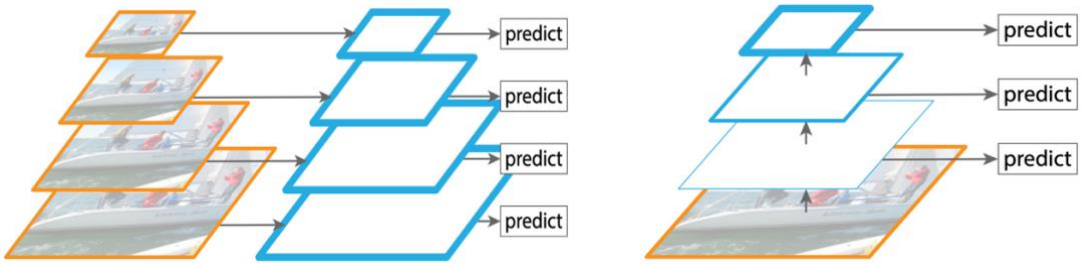
3.1 Segmentation Networks Architecture

Various segmentation models have been used in this work. First, we will study the FPN and then proceed to study DeepLabV3 and DeepLabV3+ architecture. And finally, we will briefly review transfer learning and the ResNet-50 architecture we used as the backbone of all three segmentation networks.

3.1.1 Feature Pyramid Network (FPN)

Detecting objects at different scales is difficult, especially for small objects. FPN utilize a pyramid of the same image at different scales. Processing numerous scale images take plenty of time, and the memory demand is too great. As a result, we can only use it in inference to increase accuracy as much as feasible, especially for cases where speed is not an issue.

Alternative design is a Pyramidal feature hierarchy (Figure 3.1), and it is utilized to detect objects closer to the image layer (Lin et al., 2017). However, feature maps are formed of low-level structures that are ineffective for reliable object detection. This method is generally used in single-shot detection methods (SSD) (Liu et al., 2016). These networks reuse the multi-scale feature maps computed in the forward pass from multiple layers, so they are computationally inexpensive. To avoid exploiting low-level features, SSD avoids reusing previously computed layers and instead builds the pyramid from the top of the network down. SSD detects features from several feature maps. The lower layers, however, are not used for object detection. They have a high resolution, but the semantic value is insufficient to justify their use because they are slow. As a result, SSD only employs the highest layers for detection and hence performs substantially poorer for little objects.



(a) (b)

Figure 3.1 . (a) Pyramidal feature hierarch, (b) Pyramidal feature hierarchy

(image from Lin et al., 2017)

The Feature Pyramid Network (FPN) is a feature extractor developed with accuracy and speed in mind. It generates many feature map layers (multi-scale feature maps) with higher quality information than a conventional feature pyramid for object detection. FPN consists of two pathways: bottom-up and top-down. For feature extraction, the bottom-up pathway is the standard convolutional network Figure 3.2. The spatial resolution degrades as we ascend. The semantic worth of each layer increases as more high-level structures are detected (Lin et al., 2017). ResNet-50 is used to construct the bottom-up pathway.

FPN offers a top-down approach to building higher-resolution layers from a semantic-rich layer. While the reconstructed layers are semantically strong, the locations of objects after all the down-sampling and up-sampling are not precise. To assist the detector in predicting the position better, we add lateral links between reconstructed layers and the relevant feature maps. It also acts as skip connections to make training easier (Figure 3.2).

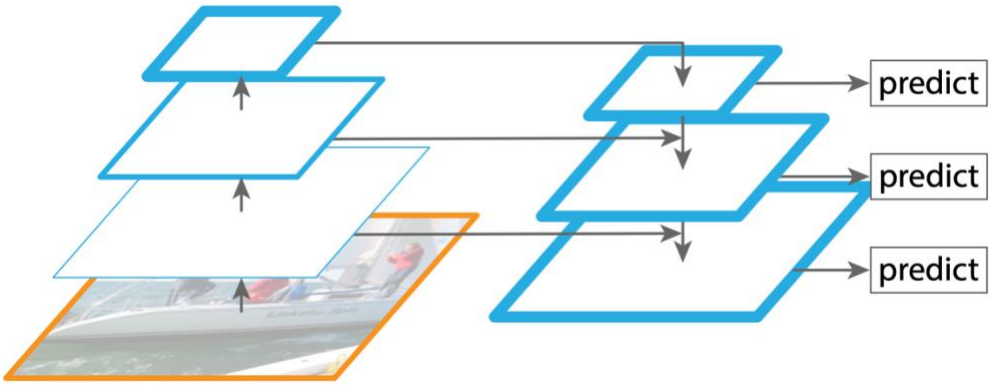


Figure 3.2. Feature pyramid network (image from Lin et al., 2017)

The FPN was originally proposed for object detection. However, since it is a generic pyramid representation, they extend their proposal by introducing FPN for segmentation by using the SharpMask/DeepMask framework (Pinheiro et al., 2016), (Pinheiro et al., 2015). On image crops, DeepMask and SharpMask were trained to predict instance segments and object/non-object scores. These models are run convolutionally during inference to generate dense proposals in an image. Image pyramids are required to generate segments at multiple scales. It is simple to modify FPN to generate mask proposals. For both training and inference, a fully convolutional configuration was employed. A 5 by 5 multi-layer perception (MLP) on top of each level of the feature pyramid to predict 14 by 14 masks and object scores in a fully convolutional approach. In addition, driven by the image pyramid's use of two scales per octave, we utilize a second MLP of input size 7 by 7 to handle half octaves (Figure 3.3).

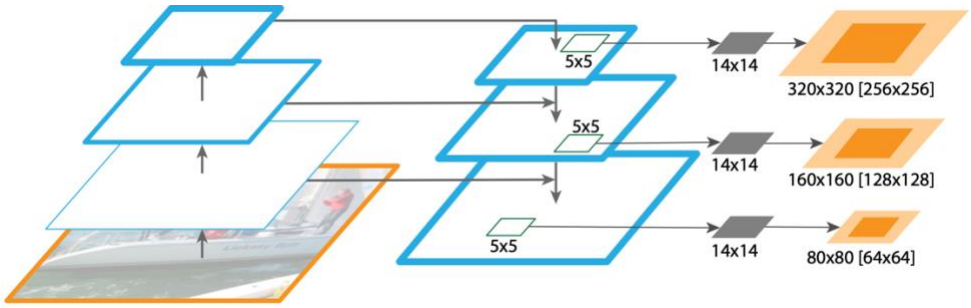


Figure 3.3. FPN architecture (image from Lin et al., 2017)

With rich semantic information, the top-down pathway restores resolution. However, lateral connections are required to restore more detailed object spatial information. A top-down approach with lateral connections considerably improves accuracy and efficiency to make FPN one of the state-of-the-art methods for the task of semantic segmentation.

3.1.2 DeepLabV3

One of the existing problems in DCNNs (Deep CNNs) was reduced feature resolution. In machine learning, generally, models try to reduce the dimensionality of the data so that they can learn various features from the input. Usually, when we perform a computer vision task with convolutional neural networks, first we down-sample the input image. Next, we perform convolution and get a feature map.

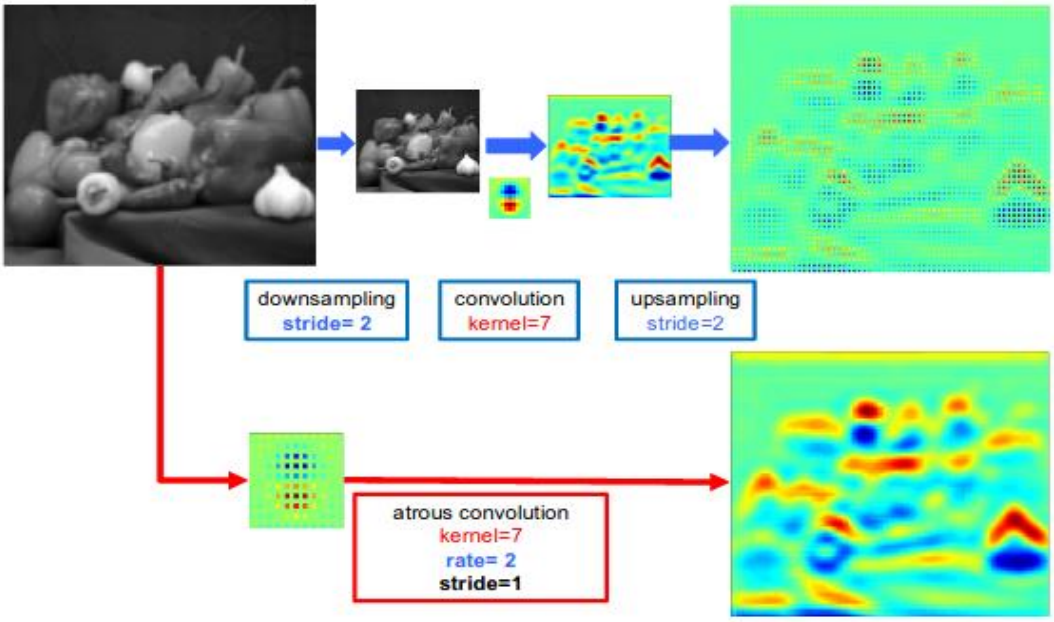
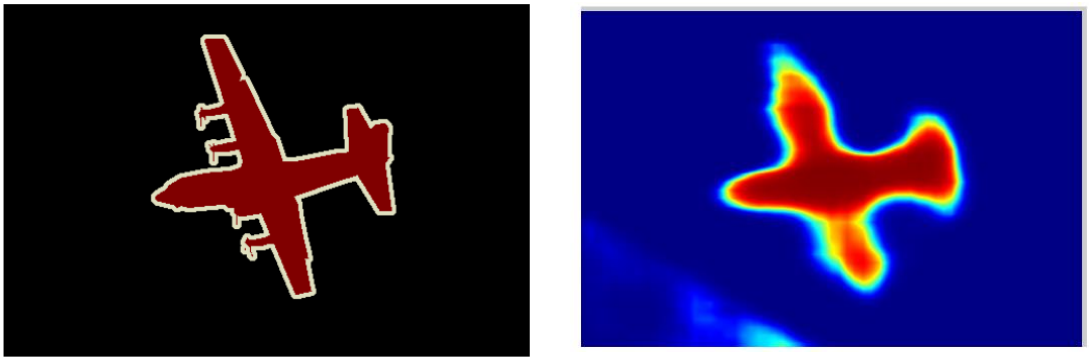


Figure 3.4. Authors used the image saying the interpolation is bi-linear, in fact it is bed of nails interpolation and not bilinear (image from Chen et al., 2017)

Typically, we reduce the dimensionality of input data by performing max-pooling and convolution operations. For instance, in a max-pooling with windows size 2 by 2, we take the maximum value from each 2 by 2 window (Figure 3.4). When we do convolution by default, we shift the pixels of the kernel by 1 pixel. When we set the stride to two, we shift by 2 pixels by time, so this is how the dimensionality of input is reduced by half. In classification, we can keep on reducing the dimensionality. However, for the segmentation task, we have to at some point, up-sample the feature map in order to save critical information for this specific task. Because of the nature of the segmentation problem, in which we need to classify each pixel in the image, we have to perform up-sampling.

As you can see in the Figure 3.5 the details of objects get lost in deep neural network's output feature map. This is because of convolutions, max pooling, down-sampling and methods that had been used for up-sampling. In classification task, this is actually helpful because it achieves invariance. However, in the segmentation task, we aim to preserve spatial information since each pixel should be classified separately.



(a) **(b)**

Figure 3.5. (a) Ground truth, (b) DCNN output (image from Chen et al., 2016)

DeepLabV3 addressed all these problems and performed better than all the state-of-the-art methods proposed before. It conquers these challenges by implementing three techniques: a) Atrous or dilated convolution, b) Atrous Spatial Pyramid Pooling (ASPP), c) Going deeper with Atrous convolution.

In order to overcome the issue of reduced feature resolution caused by successive pooling operations or convolution striding, they advocate the use of **atrous convolution,** which has shown promising results in semantic segmentation. Atrous is a French word which translated to "with holes in". Atrous or dilated convolution, as its name suggests, is type of convolution that considers dilates the part of image which it operates the

convolution on. As it is more clear in the (Figure 3.6) in dilated convolution instead of getting a 3 by 3 sample from the image we chose a field of 5 by 5 in the image however, we are still only considering 9 pixels when we apply the convolution operation, therefore, the field of view (FoV) expands with atrous convolution while computational wise it does not get much more expensive. The parameter rate adjusts the field of view, so with the same number of parameters and the same computational expenses, we get access to a bigger field of view.

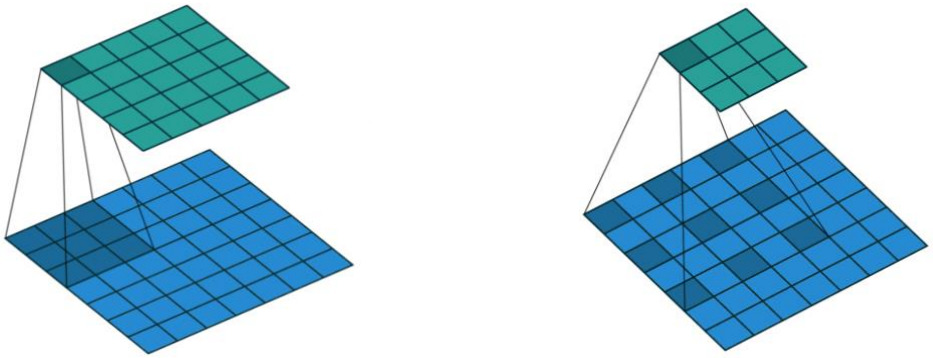


Figure 3.6. Right regular convolution, left atrous convolution (image from Dumoulin and Visin, 2018)

Atrous convolution generalizes normal convolution operation by allowing us to explicitly modify the resolution of features calculated by deep convolutional neural networks and adjust the field-of-view of the filter in order to collect multi-scale information. In a two-dimensional signal, where atrous convolution is conducted over the input feature map *x* at each position *i* on the output *y* and a filter *w*, the stride with which we sample the input signal corresponds to the atrous rate *r*, which is similar to convolving

the input *x* with up-sampled filters formed by adding *r-1* zeros between two successive filter values along each spatial dimension (Equ. 3.1). Atrous convolution also allows us to explicitly choose how densely feature responses are computed. Output stride is the term that is used for the ratio of the spatial resolution of the input image to the ultimate output resolution (Chen et al., 2016).

$$y[i] = \sum_{k} x[i + r.k]w[k] \tag{3.1}$$

The second method that has proposed is the use of **atrous spatial pyramid pooling (ASPP)**. Parallel atrous convolution with varying rates is applied on the input feature map and fused together in ASPP, therefore, the network would receive multiple fields of view from input image (Figure 3.7). Which helps to account for different object scales in the image, which can enhance accuracy because objects of the same class can have varied scales in the image. Additionally, on DeepLabV3 architecture batch normalization is included within the ASSP.

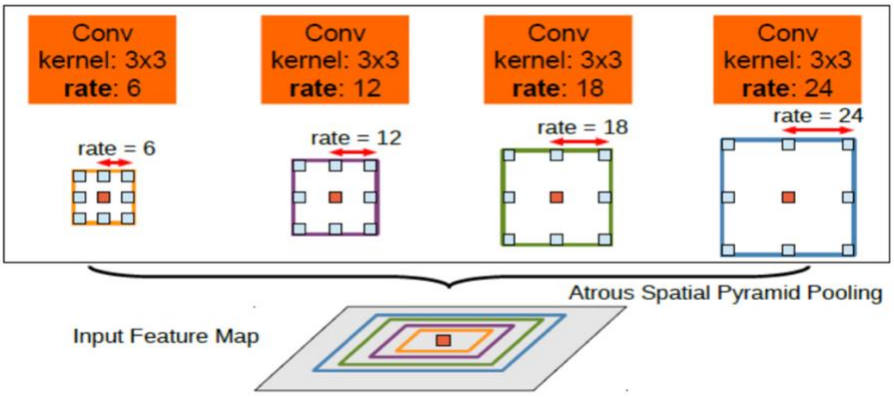


Figure 3.7. Atrous spatial pyramid pooling (ASPP) (image from Chen et al., 2017)

Numerous copies of the previous ResNet block had been duplicated, cascaded, and then denoted as block4 (Figure 3.8). In those blocks, there are three 3×3 convolutions, and the last convolution contains stride 2 except for the one in the last block, which is similar to the original ResNet (He et al., 2015). The model's motivation is that the introduced striding makes it easy to capture long-range information in the deeper blocks.

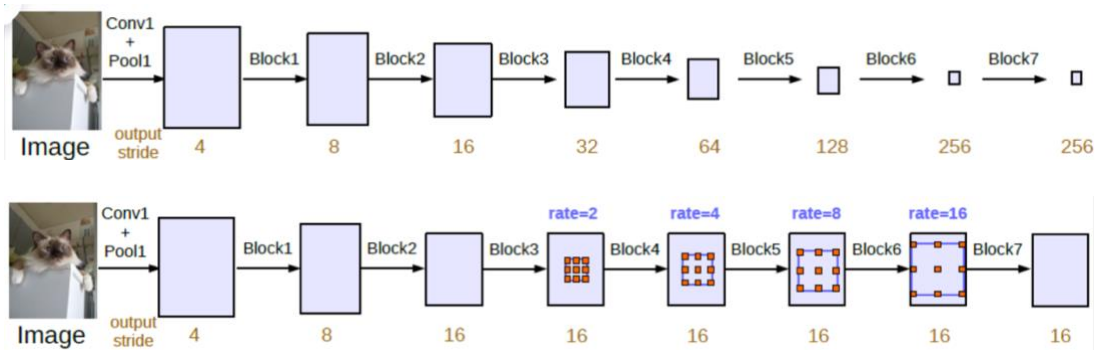
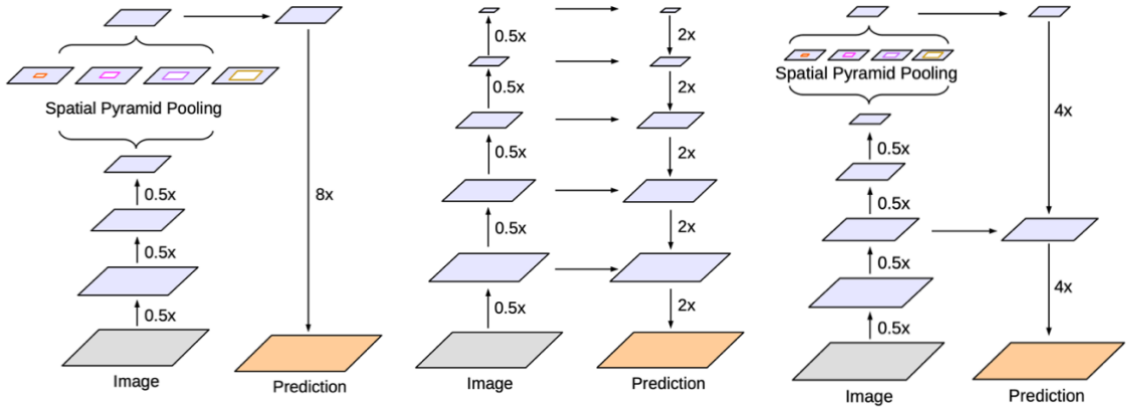


Figure 3.8. The concept of going deeper with atrous convolution (image from Chen et al., 2017)

Without Atrous Convolution, standard conventional and pooling are used, causing the output stride to increase and the output feature map to shrink as you go deeper. Consecutive striding, on the other hand, is detrimental to semantic segmentation since location/spatial information is lost at deeper layers. Hence, we can use atrous convolution to keep the stride constant while widening the field of view without increasing the number of parameters or the amount of computation.

3.1.3 DeepLabV3+

One of the other common architectures for semantic image segmentation networks is the encoder-decoder, in which, as the name suggests, the network contains two main modules. The encoder reduces the feature map size of the input image by applying numbers of convolution and pooling operations. And the decoder module tries to up-sample and recover the spatial features, in comparison to DeepLabV3 architecture only contains an encoder module. That being said, DeepLabV3+ utilizes the DeepLabV3 as the encoder and adds a decoder to it to gain a better quality on the boundaries (Figure 3.9).



(a) (b) (c)

Figure 3.9. (a) spatial pyramid pooling, (b) Encoder-decoder, (c) DeeplabV3+ contains both methods (image from Chen et al., 2018)

While the Atrous spatial pyramid pooling is especially good at extracting contextual information, the encoder-decoder architecture could extract sharp boundaries. The DeepLabV3+ network incorporates an upper version of Atrous spatial pyramid pooling known as Atrous separable pyramid pooling, along with the encoder-decoder architecture to take advantage of both and gain the upper hand in comparison to benchmark methods.

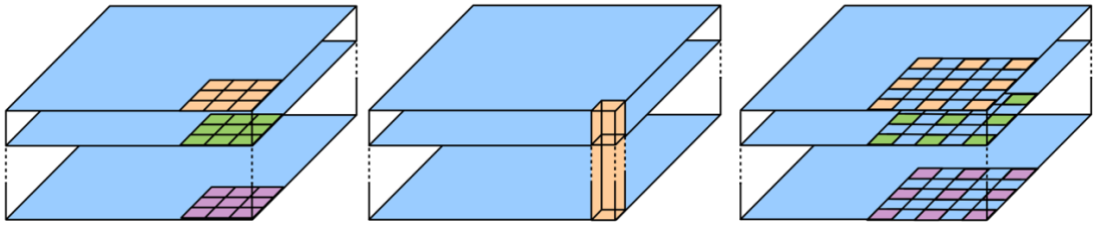


Figure 3.10. (a) depthwise convolution, (b) pointwise convolution, (c) Atrous depthwise convolution, (image from Chen et al., 2018)

Depth-wise convolution (spatial convolution) is a convolution that applies a single convolution for each input channel (Figure 3.10). In contrast, the point-wise convolution combines the outputs from depth-wise convolution across all channels (Howard et al., 2017). The combination of depth-wise and point-wise convolution forms the depth-wise separable convolution. In DeepLabV3+, an atrous separable convolution is used, which basically applies a rate two atrous convolution instead of a rate 1 Depth-wise convolution. This method was applied to reduce the complexity of the task while maintaining similar performance (Figure 3.11).

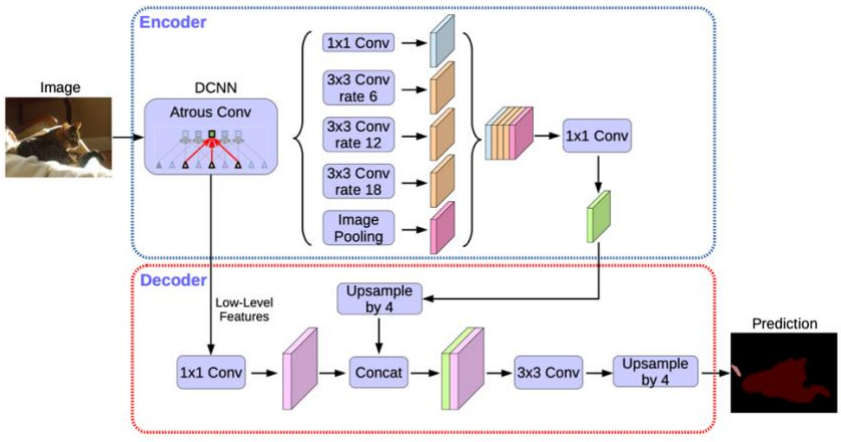


Figure 3.11. The architecture of DeeplabV3+ (image from Chen et al., 2018)

Unlike the DeepLabV3+ originally suggested the use of the Xception architecture (Chollet, 2017) as the backbone, we still used the ResNet-50 (He et al., 2016) to keep consistency with the other two networks.

3.1.4 Transfer learning for segmentation models

In transfer learning, the knowledge of an already trained machine learning model is transferred to a new but related problem. This approach, which is more of a design methodology than a technique, is popular in deep learning because it can train deep neural networks with a smaller dataset. Transfer learning attempts to use what has been learned in one task to enhance generalization in another.

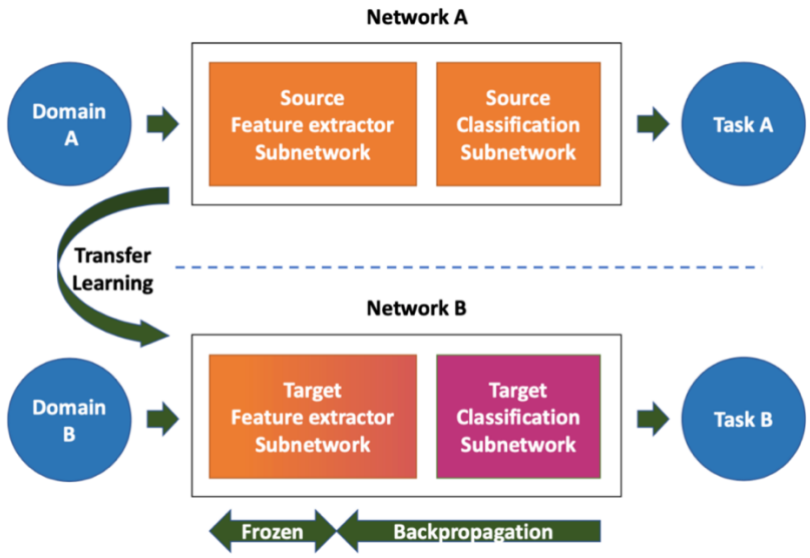


Figure 3.12. Transfer learning from one domain to another (image from Minhas and Zelek, 2019)

Neural networks in computer vision often attempt to detect edges in the early layers, shapes in the middle layer, and task-specific properties in the latter levels. The early and

middle layers are utilized in transfer learning, and only the last layers are retrained. Transfer learning offers various advantages, but the primary ones include decreasing training time, improving neural network performance, and not requiring much data. Because the model is already pre-trained, transfer learning allows construction of a good machine learning model with relatively minimal training data

There are three approaches to transfer learning. In the first approach, you train a model to reuse it because you do not have access to sufficient data to train a model for a task; by finding another related job with an abundance of data, you train your model to use it as the inception point of your model for the original task. For example, object detection models as the backbone of semantic segmentation tasks falls under this category. Another approach is to use a pre-trained model for your task (Figure 3.12). As stated before, there are various architectures like Xception (Chollet, 2017), VGG (Simonyan and Zisserman, 2015), MobileNet (Howard et al., 2017), and ResNet (He et al., 2015), which had been trained on massive datasets like ImageNet (Krizhevsky et al., 2017). We impose upon both these approaches in our research. To do transfer learning, the target segmentation sub-network is modified to include the requirements necessary for the task. The network will then be trained on either a subset or the entire network. The learning rate selected is lower than in the case of traditional training. This is due to the fact that the network already has excellent weights for the source task. We don't want to adjust the weights too quickly. Also, the initial layers are sometimes frozen because it is suggested that these layers extract general traits and can be employed without any alterations.

ResNet-50

As noted previously, ResNet demonstrated exceptional generalization performance on other recognition. ResNet architecture has numerous variations with the same concept but a different number of layers. We used ResNet-50 as the backbone of all segmentation networks in our work.

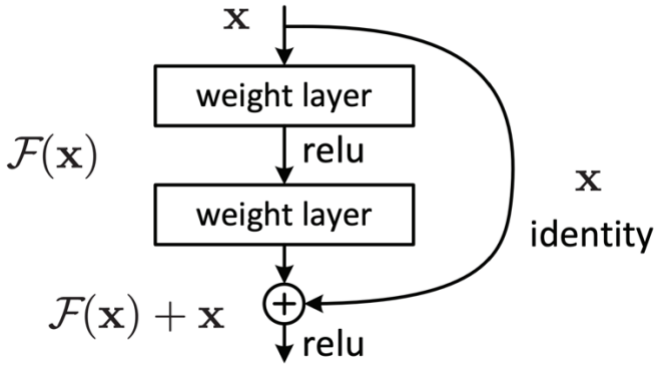


Figure 3.13. Identity mapping in ResNet architecture (image from He et al., 2015)

Deep Residual Networks are remarkably similar to networks that contain convolution, pooling, activation, and fully-connected layers stacked one on top of the other. The identity connection between the layers is the single addition to the basic network, making it a residual network. The identity mapping addition to the residual output, will provide training support. In a nutshell, applying identity mapping to the input yields the same output as the input (Figure 3.15). The residual mapping is the value that will be added to the information in order to approximate the block's final function. Alternatively stated, residual mapping is the amount of error that can be added to the input to reach the end destination (He et al., 2016). Since the addition of the identity connection adds no new

parameters, the computation complexity for deep residual networks is similar to the simple deep network. ResNet50's architecture is divided into four stages. The network begins with the input image and then conducts initial convolution and max-pooling with 7 by 7 and 3 by 3 kernel sizes, respectively. Following then, Stage 1 of the network begins with three Residual blocks, each containing three layers. The curved arrows represent the identity connection. The dashed connecting arrow indicates that the convolution operation in the Residual Block is conducted with stride 2, so the input size is decreased to half in height and breadth, but the channel width is doubled. As we advance through the stages, the channel width doubles, and the input size is cut in half.

The bottleneck design is employed in the ResNet-50 architecture. Three layers are placed using transfer learning in computer vision tasks; researchers manage to use the models trained for classification tasks on top of the other for each residual function F. The three layers comprise 1×1, 3×3, and 1×1 convolution. The 1×1 convolution layer is in charge of shrinking and then restoring the dimensions. With lower input/output dimensions, the 3×3 layer is left as a bottleneck. Finally, an Average Pooling layer is followed by a fully connected layer with 1000 neurons.

3.2 Active Learning Aspects

An active learning algorithm studies how to obtain the most useful samples from unlabeled dataset and pass them over to an "oracle" for label, aiming at reducing the labeling cost as much as possible. The oracle in the loop which can either be a human or a software, would not label all of the dataset. The dataset is created by active learning method

choosing samples of dataset to be labeled by the human in the loop, the human labeling the selected samples and updating the dataset.

Semi-supervised machine learning is a fusion of supervised and unsupervised learning techniques. It employs a little quantity of labelled data and a large number of unlabeled data, providing the advantages of both unsupervised and supervised learning while avoiding the difficulties associated with obtaining a large amount of labelled data. Active learning could be categorized under "semi-supervised learning." A semi-supervised active learning approach will provide the model only with a small portion of the labeled dataset, assuming that not all data is necessary or valuable for training purposes.

The loop of active learning involves prioritizing which data out of the dataset should be labeled for training the model. In this work we achieved this by using a reinforcement learning method. Then a human in the loop annotates the selected regions, the dataset is appended, and the model is trained on the new training set. In the context of high-dimensional data processing and automatic feature extraction, Deep Learning has an excellent learning capability, but Active Learning has significant potential to cut labelling costs efficiently. As a result, combining DL and AL is a natural method, as this considerably expands their application potential. Nevertheless, even though AL-related research on the query approach is relatively extensive, applying this strategy directly to DL remains problematic. This is primarily due to the problems listed below.

In Deep Learning, model uncertainty is significant, so the query technique based on uncertainty is the most important direction for active learning. Although DL can use the softmax layer to determine the probability distribution of the label in classification

problems, the facts demonstrate that they are overconfident. The final output's SR (Softmax Response) is unreliable as a measure of confidence, and so the performance of this method could be worse than that of random sampling.

Insufficient data is another challenge, deep learning is dependent on large amount of data to generalize, however, the signature of active learning is ability to learn from small amount of sampled data. This could be seen both in form of incompatibility of these methods and the point that they could complete each other. However, the classic active learning methods use the query method of one-by-one sampling which is not applicable for deep learning models.

Processing pipeline of active learning and deep learning are inconsistence. The active learning strategies generally focus on training classifier, then the query strategies that had been commonly used are based on fixed feature representations. However, in deep learning these two steps are optimized jointly.

Active learning data flow can be directed in three ways: continuous sampling, pool-based learning, and stream-based learning. In continuous sampling, any point in the input space can be chosen. When the input space can be queried to arbitrary accuracy, this is appropriate. In pool-based active learning, one has access to a collection (pool) of unlabeled data from which to select spots for annotation (Figure 3.14). Stream-based active learning is similar to pool-based learning, except that the pool is given sequentially, and the algorithm must determine whether or not to collect the label of the point in real time (Bayesian AL). Due to nature of this problem this research pool-based active learning had been used.

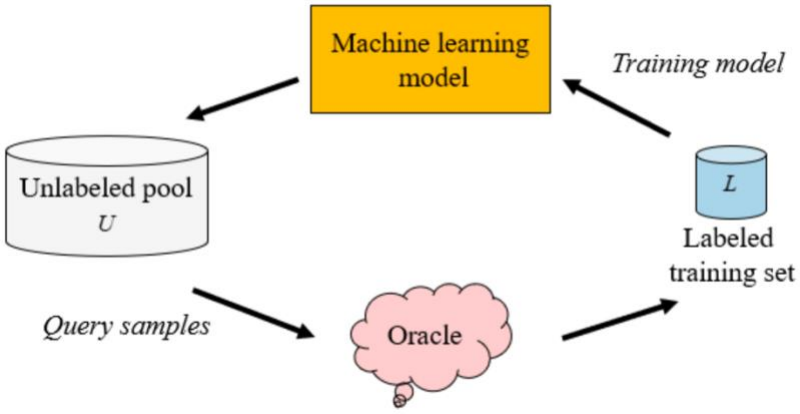


Figure 3.14. Pool based active learning (image from Ren et al., 2021)

Additionally, there are two types of active learning cycles commonly used in deep learning, a pool based active learning method and the deep active learning. With the first method a query strategy is used to sample the unlabeled pool and passed over to an oracle to annotate the data; then the training set is updated with the new labeled samples. Then the next round of training begins with the newly added information. In the deep active learning process, the parameters of the deep learning model are pretrained on a labeled training set, or initialized, then the. The samples of the unlabeled pool are used to extract features by going through the deep learning model (Ren et al., 2021). Samples are selected based on the corresponding query strategy, label the samples to create a new label training set, then train the DL model on the new set while updating the unlabeled set. This process is iterated in both methods until the label budget is depleted or the pre-defined termination conditions are met. This proposed method uses a combination of two (Liu et al., 2018).

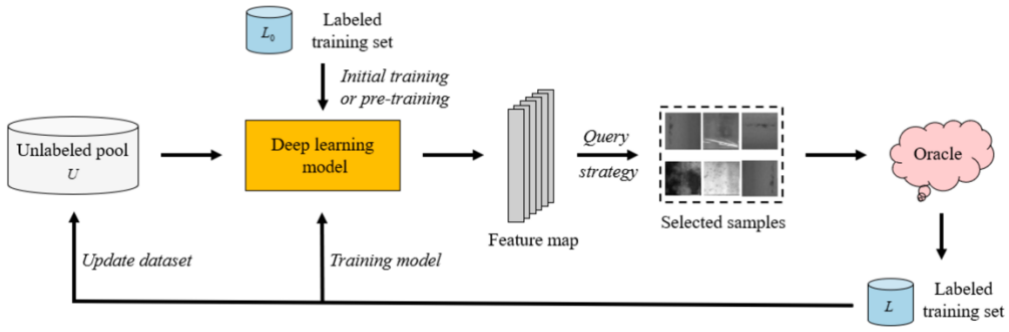


Figure 3.15. Deep active learning (image from Ren et al., 2021)

Classic active learning techniques rely on hand-crafted heuristics derived from sample uncertainty to estimate sample informativeness. Many active learning methods works used Shannon entropy for this task. Simply put, Shannon entropy is the amount of information in a bit. Bits, as they are known in computer science, are either 0 or 1, so a bit can represent two different facts. The entropy of a variable could be explained as the amount of information in a variable; in another world, entropy in information theory capture increasing randomness.

3.3 Reinforcement Learning Aspects

In some recent research, Reinforcement Learning (RL) has been used as a model to learn labeling policy. Reinforcement learning is a branch of machine learning that studies how intelligent agents should behave in a given environment in order to maximize cumulative reward. To briefly review reinforcement learning, we need to familiarize ourselves with its main components. The environment is the world in which the agent operates. The agent's state is its current situation. The reward is the scalar feedback signal

received by the agent from the environment that allows it to determine whether or not an action conducted is preferable. The agent's job is to maximize the total reward. The policy, or behavior, is the strategy for mapping an agent's state to actions. And value is the future benefits that an agent might receive by performing a specific action in a specific condition.

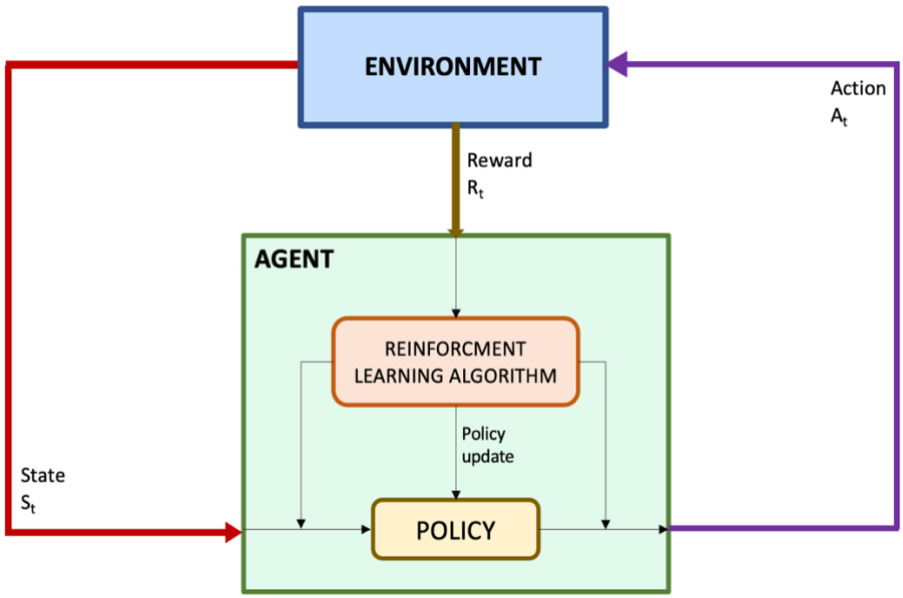


Figure 3.16. Interaction of various concepts of reinforcement learning

When the agent performs an activity, it is rewarded and transitions into a new state. We aim to determine the optimal policy that will drive the agent to the best action in each state which would accomplish the objective and receive the highest cumulative reward feasible (Figure 3.16). To achieve this optimal strategy, the agent must concurrently explore new states while attempting to maximize rewards. This is referred to as the Exploration vs. Exploitation trade-off (Sutton and Barto, 2015).

The Markov Decision Processes are the mathematical framework for representing the environment in reinforcement learning (MDPs). Markov property states that "Given the present, the future is independent of the past." (Equ. 3.2). MDP performs optimal policy search. The Markov Decision Process uses discrete actions to address Reinforcement Learning problems, allowing an agent to arrive at an optimal policy for the greatest rewards over time. The MDP simply asserts that the next state is related to the present state and actions made in that state, and activities that correspond to a given state may result in rewards. Overall, the MDP tries to teach an agent to choose a policy that results in the greatest cumulative rewards from a series of acts in a large number of states (Lim et al., 2016).

$$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, \ldots, S_t] \tag{3.2}$$

The model is another key notion in RL. RL methods can be divided into two categories: model-free and model-based. The agent learns directly from experience with its environment in model-free methods. In model-based methods, the agent learns from a restricted number of interactions within the training environment and then utilizes the gained knowledge to develop a representative model that can be applied to the real environment.

There are several approaches and algorithms under the RL concept that are used given specific conditions or tasks, each with its own set of advantages and downsides. The technique's requirements include whether it is Model-Free or Model-Based, whether it

involves an On-Policy or Off-Policy learner, the type of reinforcement (or reward) function, and whether the problem can be represented by a Markov decision process/chain. Q-learning, Temporal Difference (TD), and State-Action-Reward-State-Action (SARSA) are examples of well-established RL algorithms used to decide the best actions to execute.

Model-free refers to the agent learning directly from experience with its environment in the former RL condition, whereas model-based refers to the agent learning from a limited number of interactions within the training environment and then using the obtained knowledge to create a representative model that can be applied to the real environment. In the latter RL technique, an On-Policy learner determines the optimum action to execute by considering the policy (the mapping from a state to the likelihood of selecting the performed action, given that state), whereas an Off-Policy learner can learn independently of the agent's actions. The Off-Policy learner, on the other hand, can only do so if they have explored/interacted with the environment sufficiently.

TD learning is the approach in which the agent learns after each action, rather than at the completion of each iteration. An agent learning from an environment through episodes with no prior knowledge of the environment is referred to as temporal difference. This indicates that temporal difference is approached using a model-free or unsupervised learning strategy. You can think of it as trial-and-error learning.

The Q-learning method is a fundamental of-policy, value-based algorithm that approaches based on the so-called Q-Table. It learns the best Q-function while storing overall predicted rewards in the Q-Table. The table is updated using the Bellman equation (Equ. 3.3) and action selection is made with a ε-greedy policy which is a policy that

always takes the action which provides the largest reward. ε-greedy policy basically means that the agent will act at random. This approach is used to boost exploration because the agent may become stuck in a local optimum without it. However, this technique is constrained by the number of training iterations required to discover the ideal Q-function. Equation 4 states the updating policy for Q-learning (Sutton and Barto, 2015).

$$Q_{t+1}(S_t,A_t) = Q_t(S_t,A_t) + \alpha\ (R_{t+1} + \gamma \max Q_t(S_{t+1}, a) - Q_t\ (S_t,A_t) \quad (3.3)$$

Finally, the SARSA method is an on-policy TD method in which an agent learns what actions to take in each state based on the policy chosen or policy to be evaluated through the learning. SARSA updates the Q-table with samples created by the current policy (S_t, A_t, R_t, S_{t+1}). In transition samples, (S_{t+1}, A_{t+1}) represents the next state and action. When it reaches S_{t+1}, it will perform action A_{t+1} and update the Q-value with $Q(S_{t+1}, A_{t+1})$. While Q-Learning is against the policy, it uses the highest feasible Q-value in state S_{t+1} to update Q-value in the future. However, the action with the maximum Q-value may not be the actual action taken by the agent in the future because the agent will most likely take a random action. In other words, the action used to update policy in Q-Learning differs from the actual action taken by the agent. Equation 3.4 shows the updating policy for SARSA. In this work, we use the SARSA method for updating the DQN since it fits our description of MDP problem well (Sutton and Barto, 2015).

$$Q(S_t,A_t) = Q(S_t,A_t) + \alpha\ (R + \gamma(Q(S_{t+1}, A_{t+1}))) - Q_t\ (S_t,A_t) \quad (3.4)$$

3.4 Summary

In this chapter, the FPN, DeepLabV3, and DeepLabV3+ segmentation networks, alongside with concept of transfer learning and the ResNet-50 backbone, have been presented. Also, active learning and its various types based on information flow have been reviewed. Finally, we studied reinforcement learning, the Markov decision process, and various methods of reinforcement learning, including TD-learning, Q-learning, and SARSA method. By gaining this background knowledge, we can proceed to learn about the methodology.

Chapter 4 **Methodology**

This chapter presents the active reinforcement learning method used in this work for the semantic segmentation of images captured by mobile sensors. The aim is to obtain sufficient labeled data from a larger pool of unlabeled data to run a segmentation net successfully, based on an initially small number of training data, which gradually increases to the desired data budget *B*. For detecting the underrepresented class regions, we use a class frequency weighted mean IoU. We will proceed to explain the architecture of the end-to-end active reinforcement learning method, define the query network used to achieve the goal, and the metric used to assess the results obtained.

In this research, we used SARSA as our reinforcement learning algorithm. The SARSA (State-Action-Reward-State-Action) transition (S_t, A_t, r_t, S_{t+1}, A_{t+1}) is an iterative algorithm for determining the best policy. The action is defined as selecting the number of regions to be labeled by annotators. The network is then trained using specified regions, and the reward is determined in a piece of data that has been held out. To stabilize the training, we employ a double Deep Query Network formulation in which action selection and evaluation are separated.

4.1 Active Reinforcement Learning for Semantic Segmentation

In active reinforcement learning, an agent learns a policy to select small informative regions of the image to be labeled from a pool of unlabeled data. The region selection decision is taken based on the performance of the segmentation model being trained. The query network will be represented as a reinforcement learning agent, more specifically as a deep Q-network. The Markov decision process will be used to formulate the Active Learning problem. When we describe the State in this MDP architecture, we find it impossible to directly incorporate a segmentation network into a state representation, so we express the state space S with the help of a set-aside state set. In this situation, an action is requesting pixel-wise annotation of an unlabeled region from an operator. Because of the large-scale nature of the semantic segmentation job, computing features for each region in the unlabeled set at each step would be prohibitively expensive.

Therefore, the entire unlabeled dataset is approximated by sampling K unlabeled regions. We compute these sub-actions for each region, which has N sampled regions. The agent is rewarded r_{t+1}, which is estimated by the difference in the performance of the segmentation network trained with the newly updated dataset and previous training set on the reward dataset. The MDP is defined by the transition sequence (s_t, a_t, r_t, s_{t+1}, a_{t+1}) with $s_t \in S_t$; $a_t \in A_t$; $r_t \in R_t$; $s_{t+1} \in S_{t+1}$; $a_{t+1} \in A_{t+1}$. For each state S_t, the agent can take actions, A_t to select which samples from the unlabeled dataset to annotate. Each sub-action a_t requests that a specific location be labelled. After training the segmentation network with

the given samples, it receives a r_{t+1} reward based on the growth in mean intersection of unions per class.

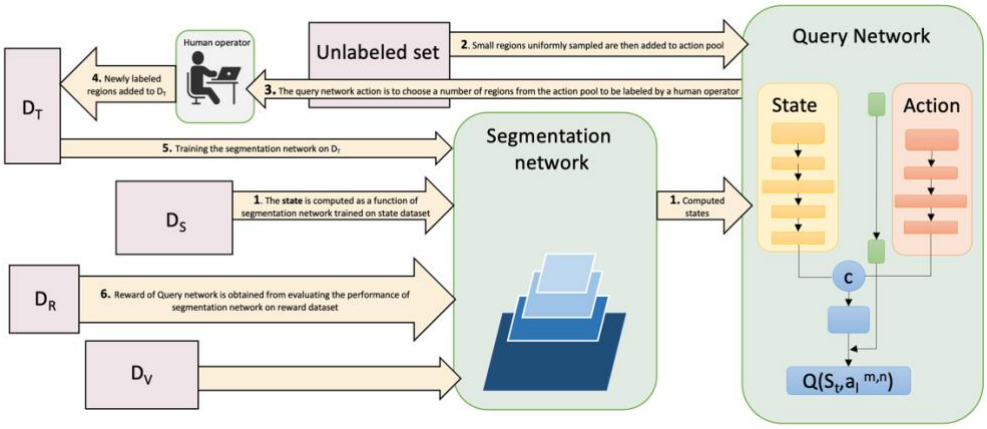


Figure 4.1. The architecture of end-to-end Active Reinforcement Learning model

The process starts with unlabeled U_t and labeled datasets L_t. The labeled data are divided into 4 subsets. We tried to keep the best distribution of classes in each subset. In case of unlabeled dataset this could be done by visual assessment of images. Also, the number of images in each subset are kept to minimum efficient number. We started the experiments with least reasonable amount of data.

- Training dataset D_T, on which active learning process had learnt an optimal acquisition function which would achieve best performance on budget B of regions.
- Reward dataset D_R to conduct a reward signal to evaluate the performance of segmentation network with chosen regions.
- State dataset D_S to generate a representation of the state.

- Validation set D_V on which the query network is evaluated.

As per Markov Decision Process, for every state $s_t \in S$ which is a function of segmentation model in timestep *t*, the agent chose samples to be annotated from the unlabeled dataset U_t which is action $a_t \in A$. The action $a_t = \{a_t^k\}_{k=1}^{k}$ is also function of segmentation model and it contains *K* sub-actions, and the sub-actions are representing the regions to be labeled. Finally, the query network scores the reward r_{t+1} based on the increase on a selected metric (MIoU) after the segmentation network had been trained with chosen samples. To train the query network, a deep Q-network and samples from an experience buffer ε had been employed. We begin by giving the segmentation network a set of initial weights θ_0 of alongside with no annotated data. The following steps are executed at each iteration *t* (Fig. 4.1):

Prior to training the query network, the segmentation network had been pretrained on a synthetic dataset, in this case, GTAv (Richter et al., 2016), and finetuned on the D_T. The steps of the reinforced active learning method are given in Figure 1.5:

1) The state of the Q network is computed as a function of the segmentation network on the state set D_S.
2) The regions that were uniformly selected from the unlabeled set are combined to form an action pool. A sub-action representation has been calculated for each action a^K_t.
3) Following that, the query network employs a ε -greedy technique to choose sub-actions from the action pool. The ε-greedy policy is an action selection policy in which the agent exploits prior information while simultaneously exploring

other alternatives. Most of the time, this technique selects the action with the most considerable estimated reward.

4) The query network's chosen regions are annotated by the human operator acting as an oracle, and these annotated regions are included in the training set.
5) The segmentation network is trained on the training set and then evaluated on the reward set D_R.
6) The improvement in the segmentation network's performance on the reward set D_R in one iteration compared to the previous one provides the query network agent with a reward reflecting how well it did in selecting the most informative regions of the images. In case there was no improvement in performance of the network, this policy would not be selected as optimal policy.

This cycle is repeated until a budget *B* of labelled regions is reached, which is the predefined number of regions the query network could ask to be labeled.

4.2 Query network architecture

The query network is comprised of two paths: one for computing state representation and another for computing action representation, which are then fused at the end. Each layer includes Batch Normalization, ReLu activation, and a fully-connected layer. The state path is composed of four layers and action is build up from three layers. A final layer merges them together to produce the global features, which are gated by a sigmoid and controlled by the KL distance distributions in the action representation. At

each step of the active learning loop, the weights are modified by sampling batches of 16 experience tuples from an experience replay buffer.

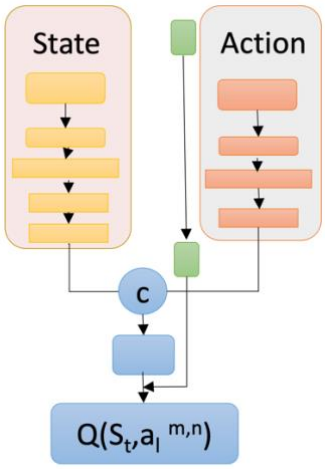


Figure 4.2. Query network architecture

The Kullback-Leibler Divergence score, abbreviated as KL divergence score, indicates how much one probability distribution differs from another. For probability Q and P the KL divergence can be calculated as the negative sum of each event's probability in P multiplied by the log of the event's probability in Q over the probability of the event in P. The KL divergence between two distributions Q and P is formulated as Equ 4.1 (Ganegedara, T., 2018.).

$$KL(P \parallel Q) = -\sum_{x \epsilon X} P(x) \log(\frac{P(x)}{Q(x)}) \quad 4.1$$

4.2.1 State representation

In this work MDP state is defined as state of the segmentation network. As we cannot directly embed segmentation network into state representation, as per suggested on (Konyushkova et al., 2017) the state is represented with the D_S dataset. The D_S is designed to represent all classes the best it can. When a new dataset is being labeled this would be done by visual estimation of classes in selecting images. Any improvement in performance of segmentation network on D_S will reflect on the dataset we are building up. It is observed that for best results, the class distribution in D_S should represent that of the training dataset. To generate the global state representation S_t, the prediction of the segmentation network on D_S is used. To prevent high memory consumption due to pixel-wise predictions, we require a compact representation. In D_S, samples are divided into patches, and compact feature vectors are produced for each of them. In D_S, each area x_i is represented as a concatenation of two features, one based on entropy and one on class predictions. The ultimate state S_t is the concatenation of all regions features.

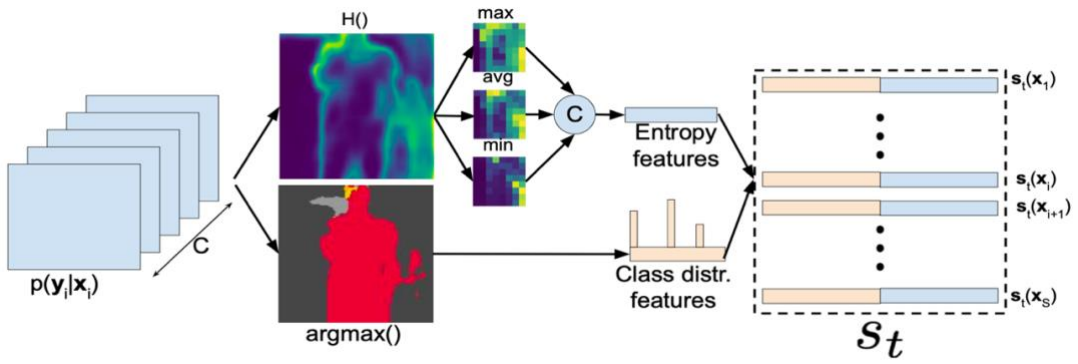


Figure 4.3. State representation (image from Casanova et al., 2020)

The first set of features is a normalized count of how many pixels are predicted to belong to each category. This feature encodes the segmentation forecast on a specific patch while ignoring spatial information, which is less essential for small patches. Furthermore, we calculate the predictor's uncertainty using the entropy over the likelihood of predicted classes. Entropy measures the amount of information in a random variable, more especially its probability distribution. In relation to a reference measure ρ, the Shannon entropy of a random variable X with distribution μ is (Shalizi, C,. 2006.):

$$H_\rho[X] \equiv -E_\mu \left[\log \frac{d\mu}{d\rho}\right] \tag{4.2}$$

To generate a spatial entropy map, we compute the entropy of each pixel location in each region. We apply min, average, and max-pooling to the entropy map to create down sampled feature maps in order to compress this representation. The second set of features is consequently created by flattening and concatenating these entropy features.

4.2.2 Action representation

In our case, executing an action is requesting the pixel-by-pixel annotation of an unlabeled region. Due to the large-scale nature of semantic segmentation, computing features for each location in the unlabeled set at each step would be prohibitively expensive. As a result, at each step *t,* we approximate the entire unlabeled set by sampling *K* pools of unlabeled regions P_t^k, each of which contains *N* (uniformly) sampled regions. We compute the sub-action representation $a_t^{k,n}$ for each region. Each sub-action $a_t^{k,n}$ is a concatenation of four distinct features:

1. The entropy feature,
2. Class distribution features (as in the state representation),
3. A measure of similarity between the region x_k and the labelled set,
4. Another measure of similarity between the region and the unlabeled set.

The idea is that the query network can learn to produce a more class-balanced labeled set while continuing sampling from the unlabeled set. This could help to minimize the segmentation dataset's hard imbalance and enhance overall performance.

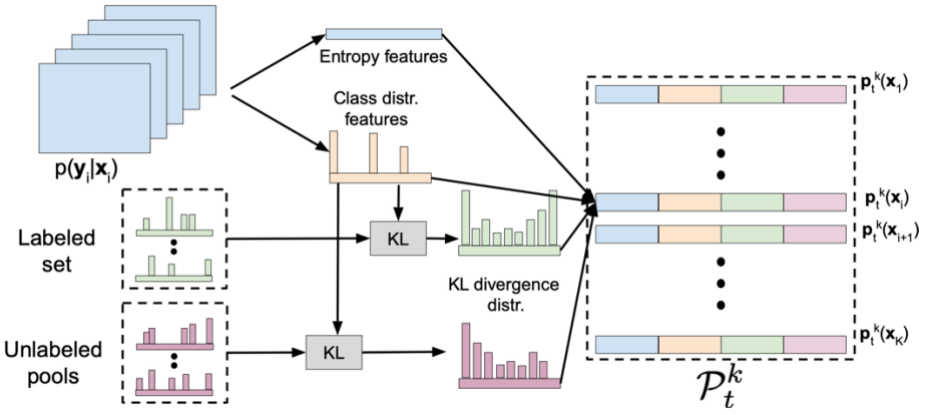


Figure 4.4. Action representation (image from Casanova et al., 2020)

Kullback-Leibler (KL) Divergence is merely a simple change to the entropy formula. The probability distribution is merged with the approximation distribution. We compute the KL divergence between the class distributions of the prediction map of region *x* and the class distributions of each labeled and unlabeled region for each candidate region, *x*, in a pool P_t^k (using the ground-truth annotations and network predictions, respectively).

For the labeled set, we compute a KL divergence score between the class distributions of the labeled areas and the one of region *x*. Taking the maximum or adding

all of these KL divergences could be used to summarize them. To extract informative features, a normalized histogram of KL divergence was calculated. The same process had been followed for the unlabeled dataset to obtain another KL divergences distribution. And finally, they are concatenated to action representation.

4.2.3 DQN batch mode

Query network converges to a policy that maps a state to an action that would optimize the future reward. A labeled set D_T is used to train the DQN parametrized by ϕ and a held-out split D_R to compute the rewards. The query agent selects *K* regions before moving on to the next state. The assumption is that each region is chosen individually, as in the case of *K* annotators labeling one region concurrently. In this situation, the action a_t is formed into *K* separate sub-actions $\{a^k_t\}^K_{k=1}$, each with a limited action space, avoiding the action space's combinatorial explosion (Equ. 4.3). We restrict each sub-action a^k_t to selecting a region x_k in P_t^k for each k ={1,…, K} action taken in timestep *t* to simplify computation and avoid selecting repeated regions in the same time-step (Casanova et al., 2020).

$$a_t^k = \underset{a_t^{k,n} \in \rho_t^k}{argmax}\ Q\left(s_t, a_t^{k,n}; \emptyset\right) \tag{4.3}$$

The network is trained by optimizing a loss based on the temporal difference (TD) error (Sutton , 1998). The loss is defined as the expectation over deconstructed transitions

generated from standard transitions (Casanova et al., 2020). E is the experience replay buffer, and y_t^k is the TD target for each sub-action (Equ. 4.4).

$$\mathbb{E}_{\mathcal{T}_k} \sim \left[\left(y_t^k - Q(s_t, a_t^k; \emptyset) \right)^2 \right] \tag{4.4}$$

The target network with weights and the double DQN formulation was used for a more stable training process (van Hasselt et al., 2016). The selection and evaluation of actions are decoupled; the action is chosen with the target network and evaluated with the query network. The TD target for each sub-action is computed as explained in Equation 4.5 where γ represents the discount factor (Casanova et al., 2020).

$$y_t^k = r_{t+1} + \gamma Q\left(s_{t+1}, \underset{a_{t+1}^{k,n} \in \rho_{t+1}^k}{argmax} Q\left(s_{t+1}, a_{t+1}^{k,n}; \emptyset'\right); \emptyset\right) \tag{4.5}$$

4.2.4 Evaluation Metric and Reward Calculation

To understand the methodology better, we first need to cover the evaluation metric we used for semantic segmentation. The concept of pixel accuracy is perhaps the most straightforward performance metric to grasp (Equ. 4.6). In the semantic segmentation task, it is simply the percentage of pixels in your image that have been correctly categorized. However, a high accuracy score does not always indicate accurate segmentation. This problem stems from an imbalance in class. When our classes are extremely imbalanced, it indicates that one or more classes dominate the image, while others account for only a small fraction of it. While the precision of prediction on those classes might be even more

critical. Unfortunately, class imbalance exists in many real-world data sets and cannot be disregarded. Therefore, there is a need for another metric which would make a better representation of segmentation quality.

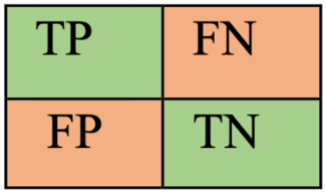


Figure 4.5. Confusion matrix

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.6}$$

$$IoU = \frac{TP}{TP+FP+FN} \tag{4.7}$$

The Intersection-Over-Union (IoU), often known as the Jaccard Index, is a popular metric to evaluate the performance of semantic segmentation task (Equ. 4.7). The area of overlap between the predicted segmentation and the ground truth is divided by the area of union between the predicted segmentation and the ground truth Figure 4.5. Confusion matrix. The IoU is a straightforward metric that is extremely effective for the task. The mean IoU of the image is calculated for binary (two classes) or multi-class segmentation by averaging the IoU of each class. IoU value would naturally range from 0 to 1, with 1 denoting perfect segmentation.

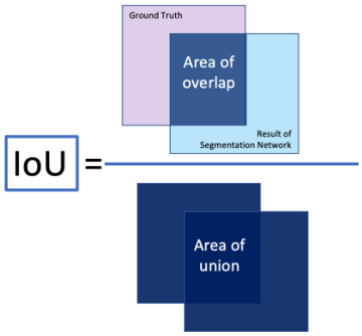


Figure 4.6. Intersection over union

Instead of penalizing the query network using the Mean IoU measure, we employed a weighted average of IoU across classes based on the frequency of the classes to focus even more attention on the underrepresented classes. Intersection-Over-Union is a popular statistic for evaluating semantic image segmentation. To compute IoU for each class, the prediction confusion matrix is employed. Instead of taking the mean of IoU, we calculated the W_c as the weight of each class that is inversely proportional to its f_c frequency which is number of pixels of each class (Equ. 4.8). And ε is just a very small value we add to frequency to avoid the complication in case of classes with zero frequency (Mohapatra et al., 2021).

$$W_c = \frac{1}{\log\left(f_c + \varepsilon\right)} \tag{4.8}$$

Frequency Weighted Average IoU, also known as FWA IoU, is an extension of the IoU statistic that is used to prevent class imbalance. If one class dominates the majority of the photos in a dataset, such “Building”, this class must be dragged down in comparison to

other classes. As a result, rather than taking the mean of all class results, a weighted average is calculated based on the frequency of the class region in the dataset. Thus, by using the weight expression, we see that high-frequency classes get lower weight and low-frequency classes get larger weight.

4.3 Summary

In this chapter, we explained our methodology. The data-driven reinforced active learning method is studied. We explained the query networks architecture and learned the state and action representations that made the model efficient. And at the end we learned about batch mode DQN and the query network that had been used in the procedure. Finally, we studied the evaluation metric used on segmentation network and the metric that we used for rewarding our query network in order to introduce attention on imbalanced classes.

Chapter 5 Experiment and Results

In this chapter, we will introduce the characteristics of the data (section 5.1) and the data partition (section 5.2) describe the selected data used in the experiments and the performed pre-processing, the implementation details of the pretraining, finetuning, and the networks implementation details for the training and testing process (section 5.3). The experimental results will be presented in the section 5.4, including frequency weighted average IoU scores of semantic segmentation in comparison to the baseline model (supervised, uniform sampling, entropy, and Bayesian). Furthermore, the advantages, limitations and problems of the model will be discussed (section 5.5).

5.1 Data Characteristics

Cityscapes Dataset

For this work we have used the Cityscapes dataset. It is a large-scale real street scene view images with resolutions of 2048 by 1024 (Cordts et al., 2016). There are 35 and 19 classes version of cityscapes dataset and we used the one with 19 classes. The training

set contains 2975 images, and the validation set is composed of 500 images (Figure 5.1). The training data contains images of 18 cities, while validation set contains the images of 3 other cities.

a) Hanover (training set)

b) Ulm (training set)

c) Munster (validation)

d) Frankfurt (validation)

Figure 5.1. The Cityspaces dataset

As it is a real-world dataset, it is extremely imbalanced. With class Road and Building occupying more than 58% of the total pixels in the whole dataset, 10 most

underrepresented classes in total occupy less than 3% of the dataset Figure 5.2. The Cityscapes dataset was used to both train the query network and test the segmentation networks performance trained with regions selected by the query network (Fig. 5.4).

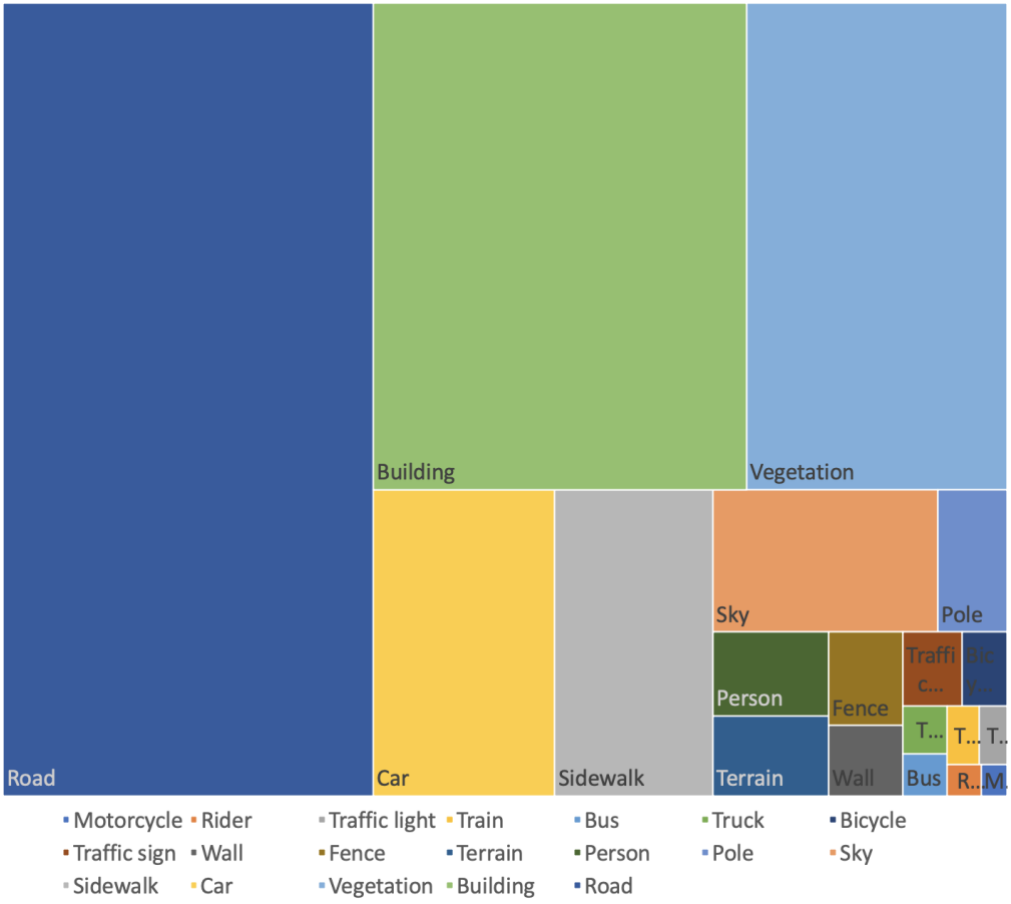


Figure 5.2. Class distribution in the cityscapes training set

GTAv dataset

We use the synthetic GTAv dataset, the images are acquired by real time rendering in game of Grand Theft Auto v. The games environment is very appealing for extracting

close to real world dataset with least effort, their realism extends beyond the integrity of material appearance and light transport modelling. It can also be found in the game worlds' content, such as the layout of items and environments, realistic texturing, the movement of vehicles, the existence of little objects that add detail (Richter et al., 2016). This dataset had been used in our work to pretrain the segmentation networks before start of training the query network.

Figure 5.3. GTAv dataset

GTAv contains images from street scene usually with resolution of 1914 by 1052 pixels (there are few images slightly bigger than mentioned sized in the dataset). We used the GTAv version which corresponds to cityscapes dataset with 19 classes. The dataset contains 24966 images in total, we split the dataset into 18582 images in training set and 6384 images in validation set (Figure 5.3).

5.2 Dataset Division

As mentioned in chapter 4, the training data had been partitioned to four sub-sets in order to train the query network and evaluate the performance of trained segmentation network on regions chosen from query network. Table 5.1 lists the number of images on each sub-set. We try to use the least amount of data that for be sufficient. The query network needs 360 images in total which is 12% of all training data and 8% of all the dataset.

Table 5.1. The cityscapes dataset partition

D_S	D_T	D_R	D_V
10	150	200	2615

5.3 Implementation steps

In these experiments we have the data initially preprocessed, then: 1) we trained the segmentation networks on a large-scale synthetic dataset; 2) we fine-tuned the model on the training set, 3) we trained the query network to learn the best policy that would select the most informative regions and 4) we evaluated the performance of query network by training the segmentation network on budgeted data selected by query network. We compare our method with a fully supervised method, other active learning methods like a random region selection, a region selection based on entropy, and finally with a Bayesian method which are some of the primary and best available methods for active learning in semantic segmentation tasks (Figure 5.4).

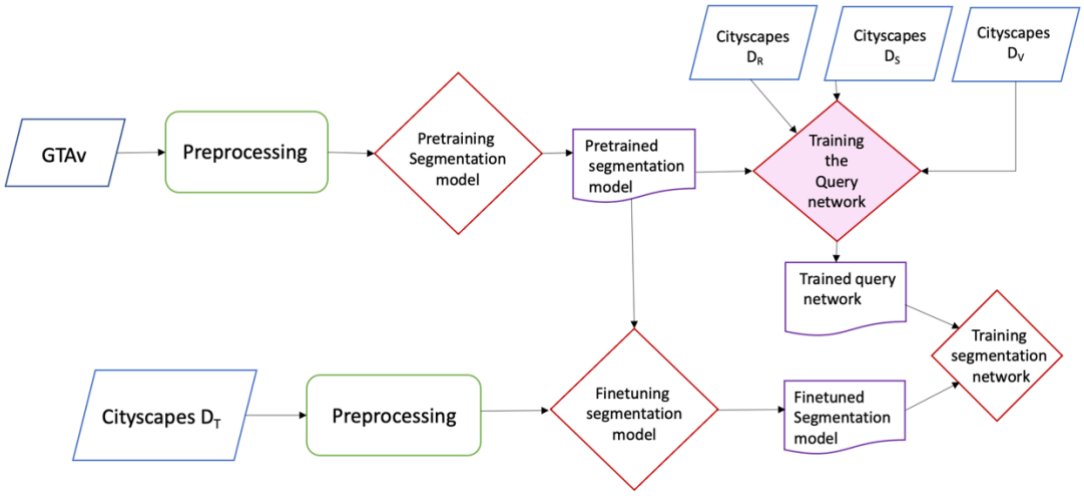


Figure 5.4.Implementation flowchart

5.3.1 Data preprocessing

For these experiments, as we have huge images in the dataset, we randomly crop smaller regions from images to overcome memory problems with larger batch sizes; in various steps, we use different region sizes (the respective region size will be mentioned). We performed a random horizontal flip in all steps of training, and additionally we applied standard normalization to all images.

5.3.2 Pretraining with GTA v Dataset

For pretraining the segmentation networks on the GTAv dataset, we used image crops of 1904 by 1024 pixel to have uniform size across all images. We used the available ResNet-50 backbone from Pytorch library which had been trained on ImageNet dataset. We used learning rate of 10 × e-3 with gamma rate of 0.995 and training batch size of 4. Additionally, we used Adam optimizer and weighted cross entropy loss function. We had

also tried SGD optimizer (Stochastic gradient descent), however we obtained better results with Adam optimizer. In all networks we froze the pretrained backbone at the beginning then proceed to unfreeze it and training it with learning rate of 10 × e-4. We used an early stopping condition in which if the Mean IoU does not increase in 70 epochs, we would stop the training.

5.3.3 Fine-tuning

After pretraining the segmentation networks on GTAv dataset, we fine-tuned the segmentation networks on D_T. As inputs of the network, we used random crops of 512 by 1024 pixels. For fine-tuning first, we freeze the backbone and used a learning rate of 10 x e-5 with the gamma rate of 0.995 and training batch size of 16. Later we unfroze the backbone and used a backbone of 10 x e-6 to train the network some more so all layers would have been fine-tuned, we did it with a very small learning rate to prevent divergence. The optimizer and loss functions are same as ones we used on pretraining the GTAv dataset. Again, we used a batch of 100 epochs for early stopping.

5.3.4 Network Implementation and configurations

Query network

The query network is trained on D_T. On each step the state is computed and the action of choosing 256 regions of 128 by 128 is taken. The acquisition functions performance is tested by having it select the regions from D_V until the budget of 3840 regions is met. We chose this number which is 1% of the dataset because it was sufficient. The segmentation network is then trained with the labeled data chosen (256 × 512 crops)

based on the policy until early stopping condition is met on D_R. Then the reward is calculated.

As mentioned before, this method is data driven and by default with using the increase on mean IoU to compute reward, we increase the performance of the network on underrepresented classes. To further address the class imbalance issue, we used a frequency weighted average IoU for calculating the reward of segmentation network.

At the end the acquisition function which result in best reward will be chose. For training the query network we used a learning rate of 10 × e-4 for both DQN and segmentation networks. The query network had been trained for 50 epochs. Full resolution images were also used.

Segmentation networks

We used three state-of-the-art segmentation network architectures: Feature Pyramid Network (FPN), DeepLabV3, and DeepLabV3+. For all of them, we used similar settings.

After acquiring the acquisition function by training the query network, we test the policy by training the segmentation network with six small budgets of regions chosen by the query network. The budget varies from 0.5% of the training dataset to 8%, which equals 1920,3840,7680,11520, 19200, and 30720 regions (Table 5.2). We chose regions with the size of 256 by 512 pixels. We used the learning rate of 10 × e-6, training batch size of 16, and gamma rate of 0.995 alongside Adam optimizer. Patience of 70 epochs was used in this step as well.

Table 5.2. the percentage of region numbers from all of the dataset

Regions number	1920	3840	7680	11520	19200	30720
Percentage	0.5%	1%	2%	3%	5%	8%

Training and testing details

To build the action pool for the query network, each image in the D_T is divided into 128 patches with a size of 128 by 128 pixels each. During each training stage, the Query network selects 256 regions to label. With a budget of 3840 regions, the DQN network is then trained.

Active learning Baseline approaches

We compared our results with the supervised learning method and the following three active learning baseline approaches, since they were the primary and best available methods:

i) Random which is the uniform random sampling of regions from action pool,

ii) The Entropy method which is an uncertainty sampling method which applies the active learning policy of choosing maximum pixel-wise Shannon entropy,

iii) Bayesian Active Learning by Disagreement (BALD) method which selects the regions based on maximum cumulative pixel-wise BALD metric (Gal et al., 2017).

Note that all methods used a model that had been pretrained on GTAv dataset and then fine-tuned on D_V. Pool sizes of 500, 200, 200 and 100 were used respectively for random, entropy, Bayesian and the proposed in this work method. Pool sizes were determined in accordance with to the best validation mean IoU.

5.4 Experimental results and discussion

First, all our segmentation networks were run on GTAv synthetic dataset. Then, the networks are fine-tuned on training subset which contains 150 images from training set. Finally, the Query network was trained and tested by training the segmentation network on image chosen by the query network. Each segmentation networks performance was compared with the random, entropy and BALD (Deep Bayesian Active Learning) active learning methods and supervised method with the budgets. Finally, the performances of all segmentation networks were compared.

5.4.1 Pretraining Segmentation networks on GTAv dataset

We used a patience of 70 epochs for this set of training, which means if validation mean IoU does not increase in 70 epochs, we will terminate the experiment. The benefit of training GTAv synthetic dataset is that we have many similar data, and it helps model to generalize better. As mentioned, we used a pretrained ResNet-50 as backbone of the networks, we froze the Backbone at beginning of our experiment and train it with learning rate of $10 \times e{-}3$. Later we unfroze the backbone and train it with smaller learning rate of

$10 \times e{-}4$. Also, the weighted cross entropy was used as loss function, the weights we calculated by formula presented in Table 5.3 lists the setting of our experiment.

5.4.2 Pretraining on GTAv dataset

Table 5.3. Pretraining details

	Backbone	*Input Size*	*Learning Rate*	*Loss*	*Batch size*	*optimizer*	*Validation Mean IoU*
FPN	ResNet-50	1904 × 1024	10 × e-3, 10 × e-4	Weighted cross entropy	4	Adam	73%
DeepLabV3	ResNet-50	1904 × 1024	10 × e-3, 10 × e-4	Weighted cross entropy	4	Adam	73%
DeepLabV3+	ResNet-50	1904 × 1024	10 × e-3, 10 × e-4	Weighted cross entropy	4	Adam	74%

As results of the experiment, the DeepLabV3+ segmentation network, achieve slightly better results compared to other two networks. It is worth mentioning that by changing the setting of the network and more experiments, we can achieve better validation mean IoU on the networks. But since the sole purpose is to pretrain the network so that segmentation network would initiate from better point, we did not focus on obtaining the best results from this experiment. All details are gathered in Table 5.3. This step is very time consuming, a GeForce RTX 3090ti GPU was used which have ram of 24 GB and CUDA Cores: 10,753. Base Clock Speed: 1670MHz. Boost Clock Speed: 1890MHz (as tested, stock is 1860MHz). Each experiments took around 250 hours.

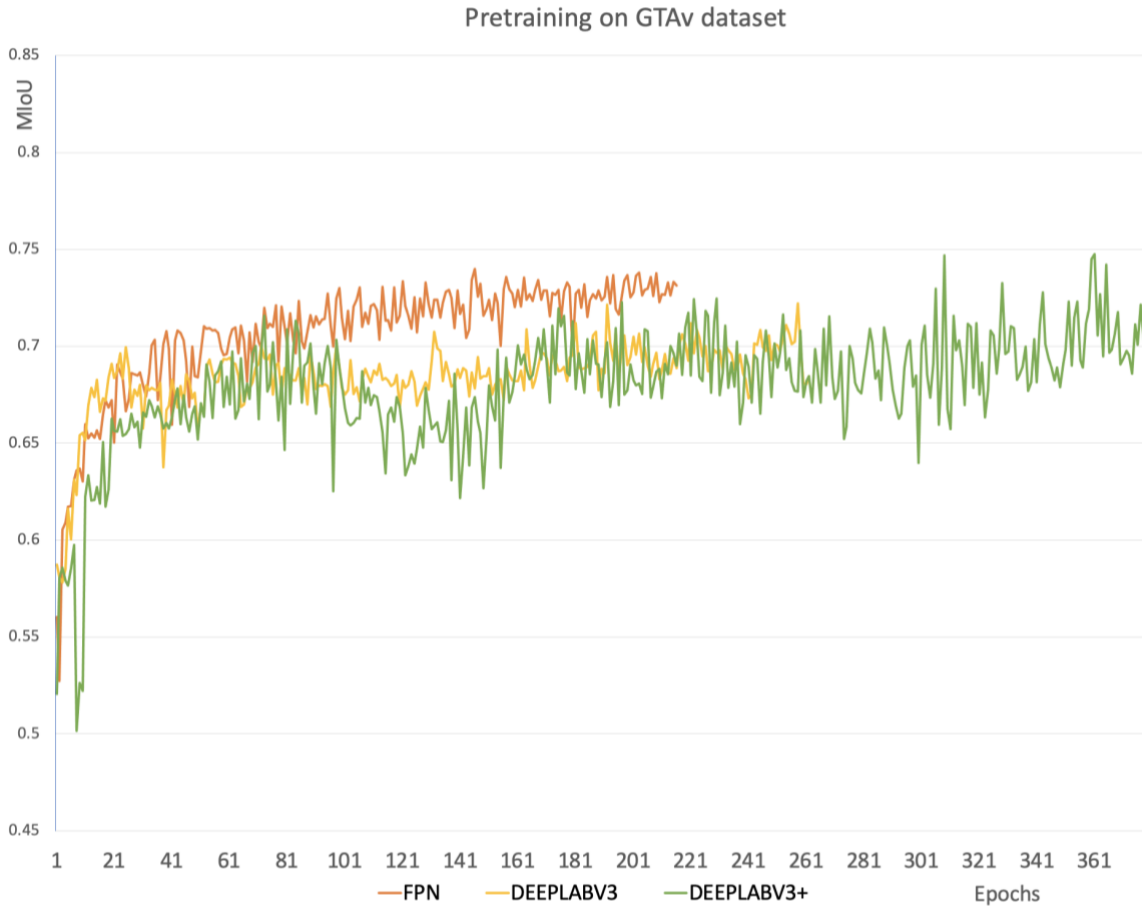


Figure 5.5. Pretraining FPN, DeepLabV3, and DeepLabV3+ segmentation networks on GTAv synthetic dataset.

5.4.3 Fine-tuning the networks on training subset of Cityscapes

In next step, we fine-tune the segmentation networks with 150 images from the training dataset of cityscapes. As mentioned before, we use similar setting as GTAv training on this step. The only difference is while freezing the backbone we used a learning rate of 10 × e-5 and then proceed to train it more with learning rate of 10 × e-6. We arrived

at this method of freezing and unfreezing the backbone, after trying many experiments like, completely freezing network, or freezing part of backbone instead of all (Table 5.4).

Table 5.4. Fine-tuning details

	Backbone	*Input Size*	*Learning Rate*	*Loss*	*Batch size*	*optimizer*	*Validation Mean IoU*
FPN	ResNet-50	512 × 1024	10 × e-5, 10 × e-6	Weighted cross entropy	16	Adam	52.2%
DeepLabV3	ResNet-50	512 × 1024	10 × e-5, 10 × e-6	Weighted cross entropy	16	Adam	54.4%
DeepLabV3+	ResNet-50	512 × 1024	10 × e-5, 10 × e-6	Weighted cross entropy	16	Adam	64.6%

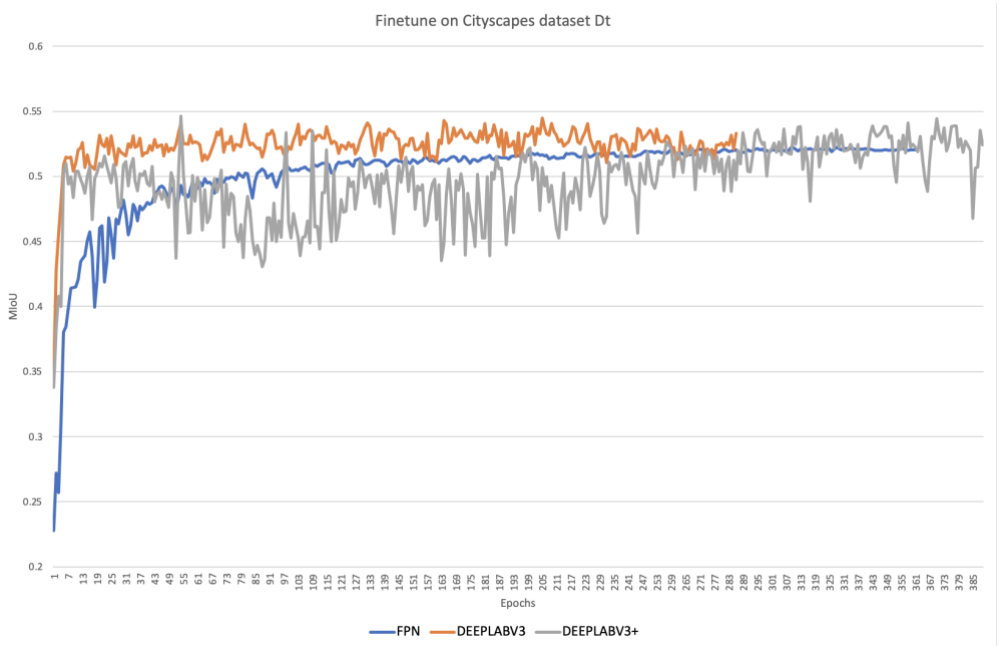


Figure 5.6. Fine-tuning on Dt subset

Since the model had been trained on a very compatible training set of GTAv, it performed well on this stage and achieved over 50% Mean IoU on each these segmentation networks.

5.4.4 Training of the DQN network

In this stage we train the DQN network with budget of 3840 trained images and with regions of size of 128 by 128 pixels. We use input size of 512 by 512 pixels while evaluating DQN network for computing the state. The computation of the state takes the longest time during this stage. The DQN network is trained for 50 epochs and each epoch approximately takes an hour. This is one of the down side the RALis method (Reinforcement Active Learning for image segmentation), where training only pretraining, fine tuning and training the Query network might take up to two weeks. However, if our goal is to take advantage of unlabeled dataset, this still saves time and resources since it would optimize the human assistance on labeling the dataset. In each episode the Query network chooses 256 more regions, when reaching the budget 3840, it trains the model for 16 epochs with the chosen data. Then the query network is evaluated based on its performance on reward set, and the state is again computed. Finally, the optimal policy is saved to be tested on segmentation network.

5.4.5 Testing with the FPN segmentation network

We test the query networks policy with various data budgets on several segmentation networks. The results are compared with other active learning policies like Random selection, selection based on entropy and BALD method. Additionally, in FPN model we used the mean IoU and frequency weighted average IoU to compute reward of query network (Table 5.5). The graph demonstrates that rewarding the query network based

on weighted IoU achieves the best result, only slightly better than using Mean IoU. From this point on we only experiment with rewarding query network with weighted IoU (Figure 5.7).

Table 5.5. The comparison of performance of FPN network across various methods with different budgets (MIoU)

Budget	Supervised	Random	Entropy	BALD	MIoU RALis	FWA IoU RALis
1920	0.28	0.53	0.59	0.56	0.58	**0.60**
3840	0.29	0.57	0.60	0.58	0.61	**0.62**
7680	0.30	0.61	0.64	0.62	**0.65**	**0.65**
11520	0.26	0.62	0.65	0.64	**0.66**	0.65
19200	0.30	0.65	0.67	0.55	0.67	**0.68**
30720	0.28	0.64	0.66	0.66	0.67	**0.71**

Additionally, a budget with 1920 samples of supervised network is equal to 15 images in Cityscapes dataset. The model is trained with weights saved from finetuning on D_t dataset and the results is not even comparable with other active learning methods. As Figure 5.7 indicates, the results of supervised method with number of images equal to same budget, is comparably very low. This displays how the supervised methods does not perform well with small amount of data.

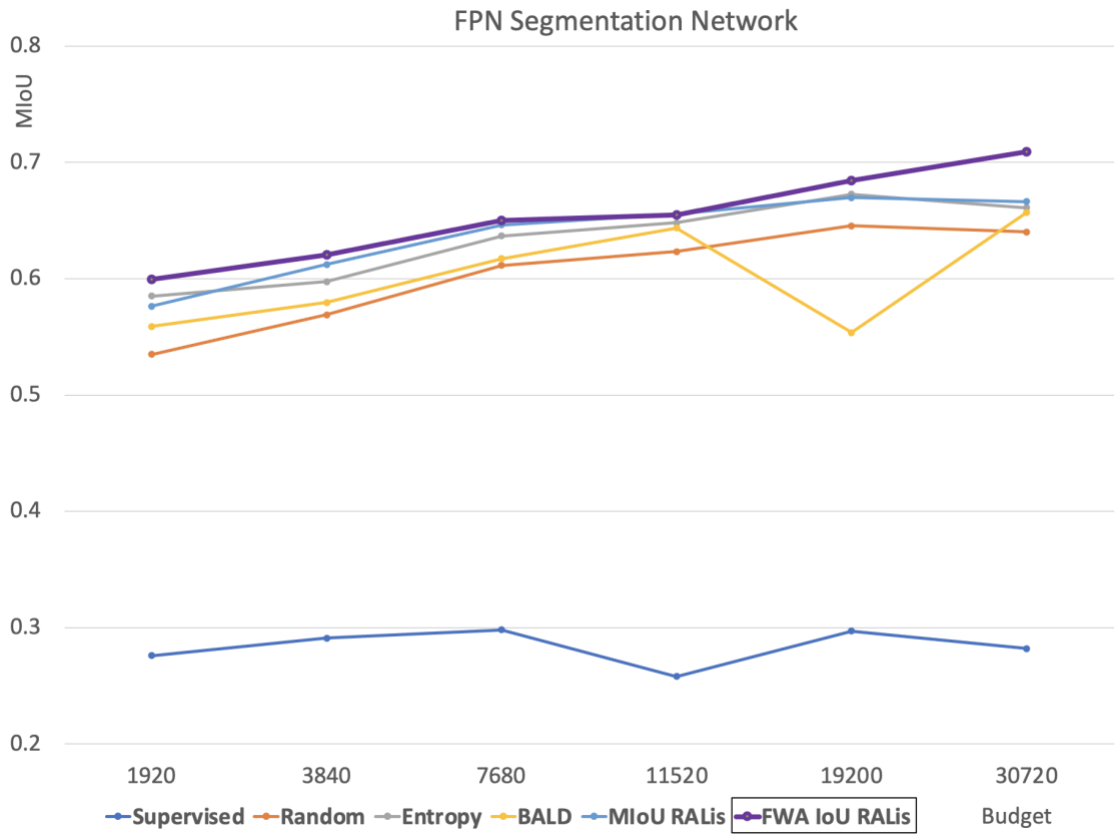


Figure 5.7. Performance of our method with FPN segmentation network compared to baselines.

Additionally, to understand the effect of using FWE IoU method in comparison to Mean IoU, Figure 5.7 shows the mean IoU on 6 underrepresented classes. The weighted IoU outperforms the RALis mean IoU method in six most underrepresented classes.

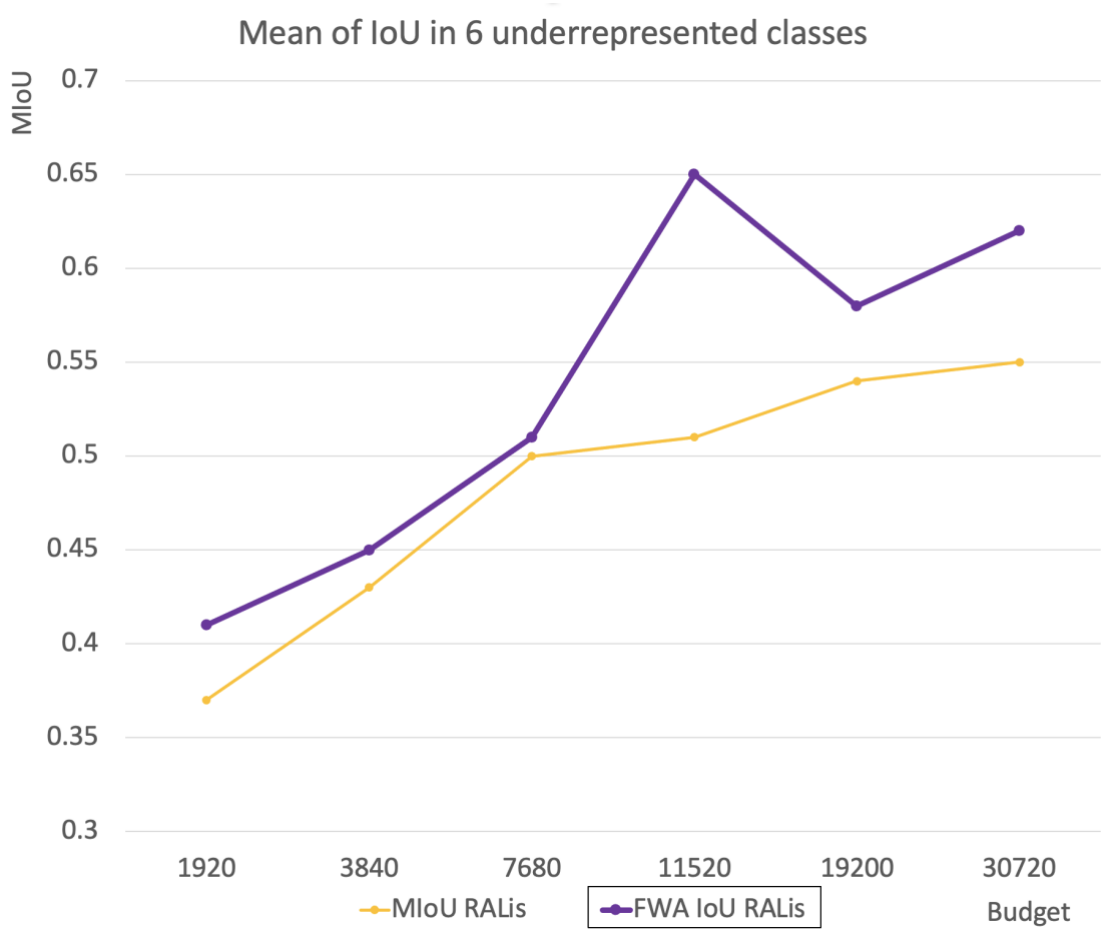


Figure 5.8. The performance of RALis method rewarded by MIoU and FWA IoU

The Table 5.6 shows the performance of RALis FWA IoU method on six underrepresented classes. From the percentage of each class, it can be seen that all the 6 classes in total, occupy less than 1.5 % of the Cityscapes training set. However, in 8% budget they achieved between 42 to 66% MIoU (Fig. 5.8).

Table 5.6. Performance of our method on six underrepresented classes (MIoU)

Budget	Bicycle	Motorcycle	Train	Bus	Truck	Rider	Total
1920	0.39	0.34	0.28	0.50	0.38	0.61	0.42
3840	0.40	0.37	0.32	0.54	0.48	0.64	0.46
7680	0.42	0.46	0.41	0.63	0.52	0.66	0.52
11520	0.44	0.54	0.44	0.46	0.65	0.48	0.66
19200	0.54	0.51	0.49	0.72	0.59	0.69	0.59
30720	0.54	0.63	0.61	0.72	0.52	0.72	0.62
Pixel %	0.41	0.1	0.23	0.24	0.27	0.14	1.39

5.4.6 Testing with the DeepLabV3 segmentation network

In this experiment also, using RALis method achieved the best result in comparison to other baselines in all the budgets. However, the results are not as good as what FPN network achieved. Figure 5.9 represent the results of this experiment.

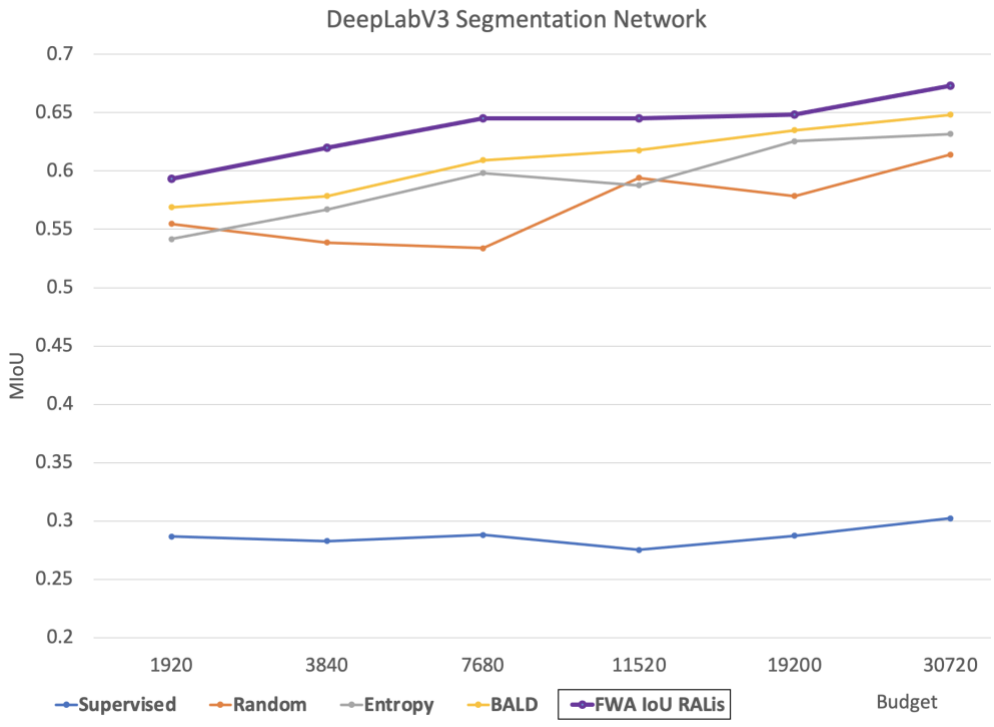


Figure 5.9. Our RALis method with DeeplabV3 segmentation network in comparison to baselines.

According to Table 5.7, segmentation network has the best performance across all budgets when it had been trained by regions selected by query network that had been rewarded with FWA IoU.

Table 5.7. The comparison of performance of DeepLabV3 network across various methods with different budgets (MIoU)

Budget	Supervised	Random	Entropy	BALD	FWA IoU RALis
1920	0.29	0.55	0.54	0.30	**0.59**
3840	0.28	0.54	0.57	0.58	**0.62**
7680	0.29	0.53	0.60	0.61	**0.65**
11520	0.28	0.59	0.59	0.62	**0.65**
19200	0.29	0.58	0.63	0.63	**0.65**
30720	0.30	0.61	0.63	0.65	**0.67**

5.4.7 Testing with the DeepLabV3+ segmentation network

DeepLabV3+ achieved the best results in comparison to other baselines in most of budgets, however, in budget of 3840, Random selection obtained the best result, it even surpasses the performance of RALis with 7680 budgets. But, at last three budgets RALis overcome other methods. Also, a deeper look at the graphs suggests that RALis methos performance decreased on 3840 budgets, indicating that in this budget RALis methods did not achieve its best performance (Fig. 5.10).

As it is displayed in Table 5.8 the DeepLabV3+ network performs the best in most budgets with regions selected by query network that had been rewarded with FWA IoU in comparison with random selection of regions, selection of regions based on Entropy, and BALD method, respectively.

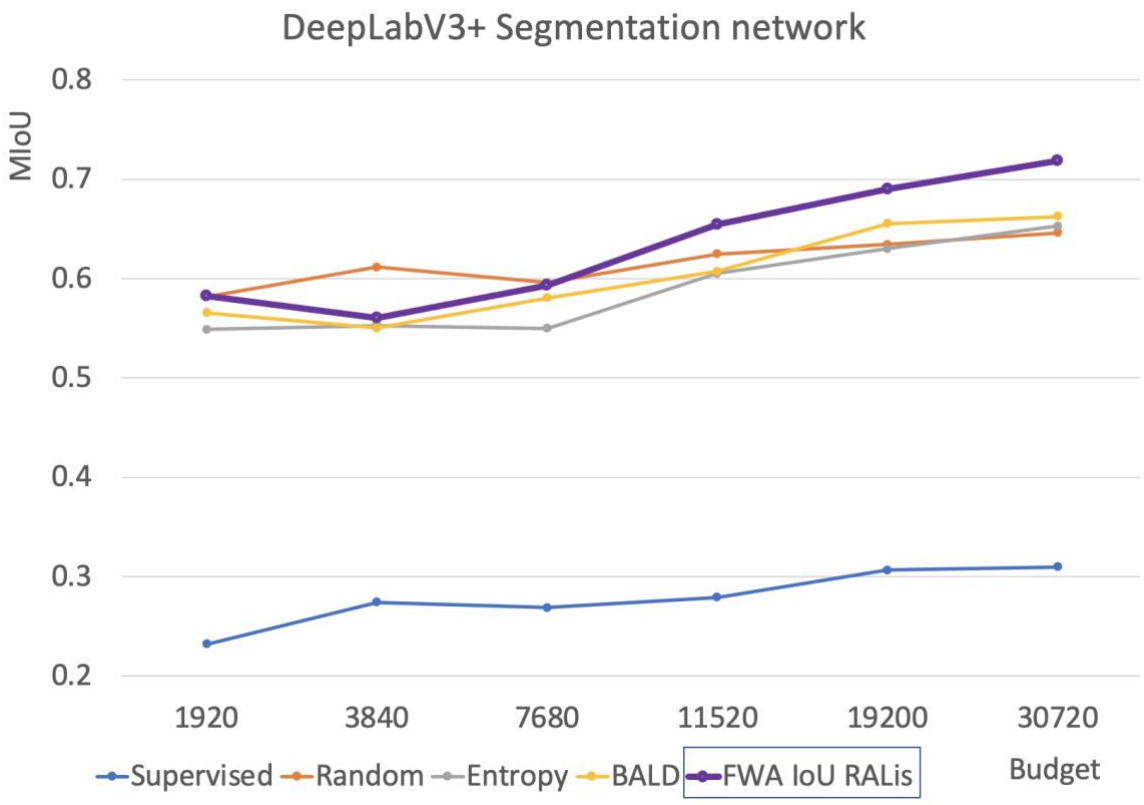


Figure 5.10. RALis method with DeeplabV3+ segmentation network in comparison to baselines.

Table 5.8. The comparison of performance of DeepLabV3 network across various methods with different budgets (MIoU)

Budget	Supervised	Random	Entropy	BALD	FWA IoU RALis
1920	0.23	**0.58**	0.55	0.56	**0.58**
3840	0.27	**0.61**	0.55	0.55	0.56
7680	0.27	**0.60**	0.55	0.58	0.59
11520	0.28	0.62	0.61	0.61	**0.65**
19200	0.31	0.63	0.63	0.66	**0.69**
30720	0.31	0.65	0.65	0.66	**0.72**

5.4.8 Testing with a Fully supervised method

In order to have a good baseline data, we trained the segmentation models fully supervised, with all the cityscapes dataset (Figure 5.11). We did used the fine-tuned method mentioned before instead of initiating the models' weights from scratch in order to have a fair comparison. For this experiment we used the full size of cityscapes image with training batch size of 4 and learning rate of $10 \times e{-}6$.

Figure 5.11. The segmentation models are training on all of the cityscapes dataset, containing 2975 images and validation IoU is compared.

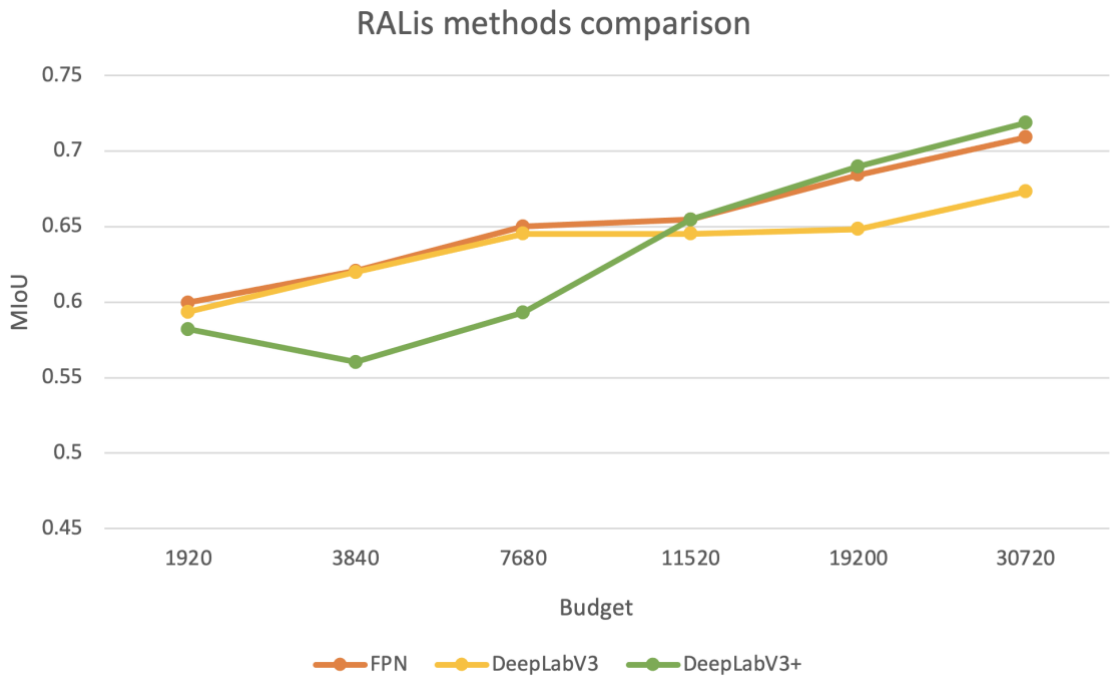


Figure 5.12. The performance of RALis method rewarded with weighted IoU compared together.

In Figure 5.12, we compare the performance of segmentation networks trained with regions chosen by reinforced active learning methods. The DeepLabV3+ performs the best on the last two budgets; however, with budgets 3840, and 7680, it falls significantly behind the other two networks. This suggests that DeepLabV3+ might be the superior method with more than 5% of the dataset budget, but with smaller budgets, it does not outperform others.

Table 5.9.The performance of supervised and FWA RALis method across three networks (MIoU).

	Supervised (100% data)	**FWA RALis (8% data)**
FPN	71.8%	70.9%
DeepLabV3	73.2%	67.3%
DeepLabV3+	75.1%	71.8%

FPN network with reinforced active learning method achieved 98.7% of performance with 8% of total data budget in comparison to fully supervised process, relatively DeepLabV3 achieved 91.9% and DeepLabV3+ achieved 95.6% respectively of the performance of the supervised process (Table 5.9).

Table 5.10. The performance of baselines and RALis method across three networks with 8% budget of all data (MIoU).

	Random	**Entropy**	**BALD**	**FWA IoU RALis**
FPN	64%	66%	66%	**71%**
DeepLabV3	61%	63%	65%	**67%**
DeepLabV3+	65%	65%	66%	**72%**

As displayed in Table 5.10 the FWA IoU RALis method achieves better results than all other region selection methods across all three segmentation networks, in comparison with our three baselines.

5.5 Summary

In this chapter, we presented the details and results of all our experiments. The experiments show that we obtained excellent results by using a synthetic dataset to train the network and only labeling 8% of the dataset for training the query network and another 8% for training the segmentation network. The FPN achieved almost 98% of the effects the network could achieve with all the cityscapes datasets. The other two segmentation networks also achieved excellent results. The experiments show that the Reinforced Active Learning method for image segmentation performs well with various segmentation

networks. Also, it comparably achieves better results than all other tested active learning baseline methods with the same budgets.

Chapter 6 Conclusions and Future work

6.1 Conclusions

In this study, we aimed to improve the deep learning method for image segmentation. At first, the problems and disadvantages of supervised learning methods for semantic segmentation is investigated. The main issues standing in the way of creating datasets, is the expensive labor cost of annotating data for semantic segmentation task. The second problem is the imbalanced dataset issue which is ever present because of nature of semantic segmentation.

Inspired by these problems, we suggested an end-to-end reinforce active learning method. We used the reinforcement learning agent as acquisition function for active learning method. The query network selects a batch of most informative regions in images and choose them to be labeled by human operator. Afterward, the selected regions are added to labeled dataset and segmentation network is trained

with these data. Afterward the performance of segmentation network is evaluated on the reward set. Then query network is rewarded based on weighted averaged validation IoU across all classes. Then the same cycle continues until an optimal policy is reached by query network, which would be choosing the most informative regions with specific budget.

Afterward, for testing the performance of query network, we tested it on segmentation network. The query network chooses number of regions based on budget. Then segmentation network which had been previously training on synthetic GTAv dataset and then finetuned on training set with 150 images of Cityscapes dataset, get trained with selected regions. The performance is tested by MIoU evaluation metric with various baselines including fully supervised method. We used three different architectures as segmentation network, and comprehensive experiments are conducted.

These comprehensive study shows that active reinforcement learning method could be used while labeling the collected data, in order to only label the regions with most important information. This method saves the operators time on labeling the data while achieving the same level performance. Additionally, the performance of underrepresented classes increases noticeably. However, it is important to note that in order to make this method work we need a synthetic dataset which would have similar content and match the classes of our data.

In this work, we have successfully trained a query network to select informative regions of each image to be labeled. For training, the query network,

340 full size images have been used which is equal to 8% of dataset. We examined the query network performance with various budgets of labeled. The budgets varied from 0.5% of dataset which is 1920 image regions of 128 by 128 pixels, to 8% which would be 30720 regions of same size. We trained three state-of-the-art segmentation networks using the reinforced active learning approach. All three segmentation networks, achieved better results of 8% budget with FWA RALis method in comparison with our baseline methods, which are random selection, entropy based active learning selection, and BALD method. The equivalent MIoU training performances compared to the image segmentation using the fully supervised approach were 98.7%f or the FPN network, 91.9% for the DeepLabV3, and 95.6% for the DeepLabV3+ , respectively.

There are some limitations to this work. As mentioned before for pretraining the segmentation networks we need a compatible synthetic dataset. It is possible to acquire this synthetic data with similar method of playing game, however, the game needs to have the same environment. Another limitation is the length of training time related to query network, and pretraining of segmentation network. The other challenge is the same problem that supervised learning methods deal with: the quality of the data. The quality of the training data directly reflects on prediction of the segmentation networks. So, if the human in the loop makes a mistake while labeling the images, the error will result on poor performance of the trained model.

6.2 Future work

Based on the problems regarding the semantic segmentation task, the future directions of our work are summarized as follows:

- **Additional dataset(s) can be used to evaluate the performance of the query networks.** In this work, we used the Cityscapes dataset with GTAv synthetic dataset; they are street content datasets with images taken from inside a car. A more imbalanced dataset with different characteristics can be used. For instance, datasets collected with other mobile sensors like UAVs can be used. Or data collected from the indoor scene by mobile robot sensors
- **Additional architectures for segmentation** networks can be utilized. In this work, we used a feature pyramid network (FPN), DeepLabV3, and DeepLabV3+. They are all state-of-the-art segmentation networks. However, other novel architectures can be tested. ParseNet and PSPNet can be used, and their performance with the data selected by the query network can be studied.
- **Instead of square-shaped region selection for the training query network, horizontal and vertical rectangles can be used.** It is essential to know the effect of the shape of the regions on the performance of query networks. Since many underrepresented classes in the image, like pedestrians and cars, are more likely to fit in a rectangle shape region, we

might get a better fit by choosing rectangles-shaped regions instead of square-shaped areas.

- **More quantitative analysis on the performance of query networks can be conducted.** It is important to study the effect of quality and quantity of reward and state set on training and performance of the query network. Additionally, further studies could be conducted on labelled data budget of training the query network.

References

Argamon-Engelson, S., Dagan, I., 1999. Committee-Based Sample Selection For Probabilistic Classi ers 26. ArXiv abs/1106.0220 (1999).

Brostow, G.J., Fauqueur, J., Cipolla, R., 2009. Semantic object classes in video: A high-definition ground truth database. Pattern Recognit. Lett. 30, 88–97. https://doi.org/10.1016/j.patrec.2008.04.005

Casanova, A., Pinheiro, P.O., Rostamzadeh, N., Pal, C.J., 2020. Reinforcement active learning for semantic segmentation. https://doi.org/10.48550/arXiv.2002.06583.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Intell. Res. 16, 321–357. https://doi.org/10.1613/jair.953

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. IEEE TPAMI.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2016. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 834-848, 1 April 2018, doi: 10.1109/TPAMI.2017.2699184.

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H., 2017. Rethinking Atrous Convolution for Semantic Image Segmentation. ArXiv170605587 Cs.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. ArXiv180202611 Cs.

Chollet, F., 2017. Xception: Deep Learning with Depthwise Separable Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR): 1800-1807.

Chu, H.M. and Lin, H.T., 2016, December. Can active learning experience be transferred?. In 2016 IEEE 16th international conference on data mining (ICDM) (pp. 841-846). IEEE.

Contardo, G., Denoyer, L. and Artières, T., 2017. A meta-learning approach to one-step active learning. arXiv preprint arXiv:1706.08334.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. ArXiv160401685 Cs.

Dumoulin, V., Visin, F., 2018. A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285.

Ebert, S., Fritz, M., Schiele, B., 2012. RALF: A reinforced active learning formulation for object class recognition, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition. Presented at the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Providence, RI, pp. 3626–3633. https://doi.org/10.1109/CVPR.2012.6248108

Ferguson, M., Ak, R., Lee, Y.-T.T., Law, K.H., 2017. Automatic localization of casting defects with convolutional neural networks, in: 2017 IEEE International Conference on Big Data (Big Data). IEEE, Boston, MA, pp. 1726–1735. https://doi.org/10.1109/BigData.2017.8258115

Freund, Y., Seung, H.S., Shamir, E., Tishby, N., 1992. Information, Prediction, and Query by Committee. In Proceedings of the 5th International Conference on Neural Information Processing Systems (NIPS'92). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 483–490.

Fukushima, K., 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. 36, 193–202. https://doi.org/10.1007/BF00344251

Gal, Y., Islam, R., Ghahramani, Z., 2017. Deep Bayesian Active Learning with Image Data. ArXiv170302910 Cs Stat.

Ganegedara, T., 2018. Intuitive Guide to Understanding KL Divergence. https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-understanding-kl-divergence-2b382ca2b2a8

Geiger, A., Lenz, P., Stiller, C., Urtasun, R., 2013. Vision meets robotics: The KITTI dataset. Int. J. Robot. Res. 32, 1231–1237. https://doi.org/10.1177/0278364913491297

Gorriz, M., Carlier, A., Faure, E., Giro-i-Nieto, X., 2017. Cost-Effective Active Learning for Melanoma Segmentation. ArXiv171109168 Cs.

Guo, J., Zhou, X., Li, J., Plaza, A., Prasad, S., 2017. Superpixel-Based Active Learning and Online Feature Importance Learning for Hyperspectral Image Analysis. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10, 347–359. https://doi.org/10.1109/JSTARS.2016.2609404

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep Residual Learning for Image Recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep Residual Learning for Image Recognition. ArXiv151203385 Cs.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv170404861 Cs.

Hsu, W.N. and Lin, H.T., 2015, February. Active learning by learning. In twenty-ninth AAAI conference on AI.

Jadon, S., 2020. A survey of loss functions for semantic segmentation, in: 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB). Presented at the 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), IEEE, Via del Mar, Chile, pp. 1–7. https://doi.org/10.1109/CIBCB48159.2020.9277638 Jain, S.D., Grauman, K., 2016. Active Image Segmentation Propagation, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, pp. 2864–2873. https://doi.org/10.1109/CVPR.2016.313

Jodeiri Rad, M., Armenakis, C., 2022. Active reinforcement learning for the semantic segmentation of images captured by mobile sensors. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. XLIII-B2-2022, 593–599. https://doi.org/10.5194/isprs-archives-XLIII-B2-2022-593-2022

Joshi, A.J., Porikli, F., Papanikolopoulos, N., 2009. Multi-class active learning for image classification, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition. Presented at the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), IEEE, Miami, FL, pp. 2372–2379. https://doi.org/10.1109/CVPR.2009.5206627

Kampffmeyer, M., Salberg, A.-B., Jenssen, R., 2016. Semantic Segmentation of Small Objects and Modeling of Uncertainty in Urban Remote Sensing Images Using Deep Convolutional Neural Networks, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Las Vegas, NV, USA, pp. 680–688. https://doi.org/10.1109/CVPRW.2016.90

Konyushkova, K., Sznitman, R., Fua, P., 2019. Discovering General-Purpose Active Learning Strategies. ArXiv181004114 Cs Stat.

Konyushkova, K., Sznitman, R., Fua, P., 2015. Introducing Geometry in Active Learning for Image Segmentation. ArXiv150804955 Cs.

Konyushkova, K., Sznitman, R., Fua, P., 2017. Learning Active Learning from Data. Advances in neural information processing systems, 30.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 84–90. https://doi.org/10.1145/3065386

Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B., 2019. The Omniglot challenge: a 3-year progress report. Curr. Opin. Behav. Sci. 29, 97–104. https://doi.org/10.1016/j.cobeha.2019.04.007

Lenczner, G., Chan-Hon-Tong, A., Saux, B.L., Luminari, N., Besnerais, G.L., 2022. DIAL: Deep Interactive and Active Learning for Semantic Segmentation in Remote Sensing. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 15, pp.3376-3389.

Li, X., Guo, Y., 2013. Adaptive Active Learning for Image Classification, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition. Presented at the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Portland, OR, USA, pp. 859–866. https://doi.org/10.1109/CVPR.2013.116

Lim, S.H., Xu, H., Mannor, S., 2016. Reinforcement Learning in Robust Markov Decision Processes. Math. Oper. Res. 41, 1325–1353. https://doi.org/10.1287/moor.2016.0779

Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature Pyramid Networks for Object Detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, HI, pp. 936–944. https://doi.org/10.1109/CVPR.2017.106

Liu, M., Buntine, W., Haffari, G., 2018. Learning How to Actively Learn: A Deep Imitation Learning Approach, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Presented at the Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, pp. 1874–1883. https://doi.org/10.18653/v1/P18-1174

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C., 2016. SSD: Single Shot MultiBox Detector. https://doi.org/10.1007/978-3-319-46448-0_2

Long, J., Shelhamer, E., Darrell, T., 2015. Fully Convolutional Networks for Semantic Segmentation. ArXiv1411.4038.

Mackowiak, R., Lenz, P., Ghori, O., Diego, F., Lange, O., Rother, C., 2018. CEREALS - Cost-Effective REgion-based Active Learning for Semantic Segmentation. ArXiv181009726 Cs.

Minhas, M.S., Zelek, J., 2019. Anomaly Detection in Images. Minhas, M.S., Zelek, J., 2019. Anomaly Detection in Images. Arxiv1905.13147.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M., 2013. Playing Atari with Deep Reinforcement Learning. ArXiv13125602 Cs.

Nguyen, H.T., Smeulders, A., 2004. Active learning using pre-clustering, in: Twenty-First International Conference on Machine Learning - ICML '04. Presented at the Twenty-first international conference, ACM Press, Banff, Alberta, Canada, p. 79. https://doi.org/10.1145/1015330.1015349

Pang, K., Dong, M., Wu, Y., Hospedales, T., 2018. Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning. arXiv preprint arXiv:1806.04798.

Pinheiro, P.O., Collobert, R., Dollar, P., 2015. Learning to Segment Object Candidates 9. Advances in neural information processing systems, 28. NIPS.

Pinheiro, P.O., Lin, T.-Y., Collobert, R., Dollàr, P., 2016. Learning to Refine Object Segments. In European conference on computer vision (pp. 75-91). Springer, Cham.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B.B., Chen, X., Wang, X., 2021. A Survey of Deep Active Learning. ACM computing surveys (CSUR), 54(9), pp.1-40.

Richter, S.R., Vineet, V., Roth, S., Koltun, V., 2016. Playing for Data: Ground Truth from Computer Games. ArXiv160802192 Cs.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. ArXiv150504597 Cs.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M., 2016. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, NV, USA, pp. 3234–3243. https://doi.org/10.1109/CVPR.2016.352

Roy, N., McCallum, A., 2001. Toward optimal active learning through monte carlo estimation of error reduction. ICML, Williamstown, 2, pp.441-448.

Růžička, V., D'Aronco, S., Wegner, J.D., Schindler, K., 2020. Deep Active Learning in Remote Sensing for data efficient Change Detection. In Proceedings of MACLEAN: MAChine Learning for EArth ObservatioN Workshop co-located with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2020) (Vol. 2766). RWTH Aachen University.

Sener, O., Savarese, S., 2018. Active Learning for Convolutional Neural Networks: A Core-Set Approach. arXiv preprint arXiv:1708.00489.

Shalizi, C., 2006. 36-754, Advanced Probability II. https://www.stat.cmu.edu/~cshalizi/754/2006/notes/all.pdf

Shannon, C. and Weaver, W., 1948. The Mathematical Theory of Communication. Bell System Technical Journal, 27, 379-423, 623-656. http://dx.doi.org/10.1002/j.1538-7305.1948.tb00917.x

Sik-Ho Tsang, 2018. Review: DeepMask (Instance Segmentation). shtsang.medium.com/review-learning-to-segment-every-thing- c4e5fc3b3bfe

Simonyan, K., Zisserman, A., 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.

Sutton, R.S. & Barto, A.G., 2018. Reinforcement learning: An introduction, MIT press.

Van Hasselt, H., Guez, A., Silver, D., 2015. Deep Reinforcement Learning with DoubleQ-Learning.ArXiv,abs/1509.06461.

Venkatesh, B., Thiagarajan, J.J., 2020. Ask-n-Learn: Active Learning via Reliable Gradient Representations for Image Classification. arXiv preprint arXiv:2009.14448.

Wang, K., Zhang, D., Li, Y., Zhang, R., Lin, L., 2017. Cost-Effective Active Learning for Deep Image Classification. IEEE Trans. Circuits Syst. Video Technol. 27, 2591–2600. https://doi.org/10.1109/TCSVT.2016.2589879

Woodward, M., Finn, C., 2017. Active One-shot Learning. arXiv preprint arXiv:1702.06559.

Yeung, M., Sala, E., Schönlieb, C.-B., Rundo, L., 2022. Unified Focal loss: Generalising Dice and cross entropy-based losses to handle class imbalanced medical image segmentation.Comput.Med. Imaging Graph. 95, 102026. https://doi.org/10.1016/j.compmedimag.2021.102026

Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J., 2017. Pyramid Scene Parsing Network. ArXiv161201105 Cs.