Deep learning for visual semantic segmentation

Yifu Chen

THÈSE DE DOCTORAT DE
SORBONNE UNIVERSITÉ

Spécialité
Informatique
École Doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Yifu Chen

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Sujet de la thèse :

Deep learning for visual semantic segmentation

Apprentissage profond pour la segmentation sémantique d'images

devant le jury composé de :

M. Patrick LAMBERT	Université de Savoie	Rapporteur
M. Sébastien LEFÈVRE	Université Bretagne Sud	Rapporteur
Mme. Catherine ACHARD	Sorbonne Université	Examinatrice
Mme. Camille COUPRIE	Facebook AI Research	Examinatrice
M. Frederic PRECIOSO	Université UCA	Examinateur
M. Arnaud DAPOGNY	Datakalab	Examinateur
M. Matthieu CORD	Sorbonne Université	Directeur de thèse

ABSTRACT

With the proliferation of the Internet and cameras, the amount of visual data is increasing dramatically. With these data, many fascinating applications can be imagined, such as automated driving systems or computer-assisted medical diagnosis. Therefore, it becomes of critical importance to develop the technology for an automatic and reliable analysis of visual data.

In this thesis, we are interested in Visual Semantic Segmentation, one of the high-level task that paves the way towards complete scene understanding. Specifically, it requires a semantic understanding at the pixel level. With the success of deep learning in recent years, semantic segmentation problems are being tackled using deep architectures. Typically, these approaches consist of three components: a deep network, a loss function and an optimization process on an annotated dataset. In the first part, we focus on the construction of a more appropriate loss function for semantic segmentation. More precisely, we define a novel loss function by employing a semantic edge detection network. This loss imposes pixel-level predictions to be consistent with the ground truth semantic edge information, and thus leads to better shaped segmentation results. In the second part, we address another important issue, namely, alleviating the need for training segmentation models with large amounts of fully annotated data. We propose a novel attribution method that identifies the most significant regions in an image considered by classification networks. We then integrate our attribution method into a weakly supervised segmentation framework. The semantic segmentation models can thus be trained with only image-level labeled data, which can be easily collected in large quantities. All models proposed in this thesis are thoroughly experimentally evaluated on multiple datasets and the results are competitive with the literature.

RÉSUMÉ

Avec la prolifération de l'Internet et des caméras, la quantité de données visuelles disponible augmente de façon spectaculaire. Grâce à ces données, on peut imaginer de nombreuses applications fascinantes, telles que les systèmes de conduite automatisés ou le diagnositic médical assisté. Il devient donc essentiel de développer la technologie permettant une analyse automatique et précise des données visuelles.

Dans cette thèse, nous nous intéressons à la segmentation sémantique visuelle, une des tâches de haut niveau qui ouvre la voie à une compréhension complète des scènes. Plus précisément, elle requiert une compréhension sémantique au niveau du pixel. Avec le succès de l'apprentissage approfondi de ces dernières années, les problèmes de segmentation sémantique sont abordés en utilisant des architectures profondes. En général, trois éléments structurent ces approaches : un réseau profond, une fonction de coût et un processus d'optimisation sur un ensemble de données annotées. Dans la première partie, nous nous concentrons sur la construction d'une fonction de coût plus appropriée pour la segmentation sémantique. En particulier, nous définissons une nouvelle fonction de coût basé sur un réseau de neurone de détection de contour sémantique. Cette fonction de coût impose des prédictions au niveau du pixel cohérentes avec les informations de contour sémantique de la vérité terrain, et conduit donc à des résultats de segmentation mieux délimités. Dans la deuxième partie, nous abordons une autre question importante, à savoir l'apprentissage de modèle de segmentation avec peu de données annotées. Pour cela, nous proposons une nouvelle méthode d'attribution qui identifie les régions les plus importantes dans une image considérée par les réseaux de classification. Nous intégrons ensuite notre méthode d'attribution dans un contexte de segmentation faiblement supervisé. Les modèles de segmentation sémantique sont ainsi entraînés avec des données étiquetées au niveau de l'image uniquement, facile à collecter en grande quantité. Tous les modèles proposés dans cette thèse sont évalués expérimentalement de manière approfondie sur plusieurs ensembles de données et les résultats sont compétitifs avec ceux de la littérature.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AI	Artificial Intelligence
AGI	Artificial General Intelligence
ANN	Artificial Neural Networks
AutoML	Automated Machine Learning
CNN	Convolutional Neural Network
CRF	Conditional Random Fields
CV	Computer Vision
DL	Deep Learning
FCN	Fully Convolutional Network
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
IoU	Intersection over Union
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
mIoU	mean Intersection over Union
MIL	Multi Instance Learning
ML	Machine Learning
NAS	Neural Architecture Search
NLP	Natural Language Processing
PPCE	Per Pixel Cross Entropy
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SSIM	Structural Similarity Measure
WSL	Weakly Supervised Learning

Chapter 1

INTRODUCTION

Contents

1.1 Context

Artificial Intelligence (AI) has been a subject of great interest for many decades, aiming at reproducing and extending human intelligence into machines. Although we are still far away from achieving Artificial General Intelligence (AGI), scientists have made breakthroughs in many aspects ranging from image and text understanding to automatically playing games. What's even more exciting is that these breakthroughs almost all happened in just a few years. Nowadays, with the investment of numerous research laboratories and technology companies, applications of AI can be found in many aspects of our lives. Those include face recognition, virtual assistant (e.g. Siri), automatic translation, information retrieval (e.g. Youtube video recommendation), autopilot, objects tracking in videos, speech recognition, clinical diagnoses, digital marketing, etc.

Given that 75% - 80% of human perception of the objective world comes from vision, Computer Vision (CV) is one of the most remarkable and fundamental areas of AI, which aims to automatically acquire, process, analyze and understand digital images. Due to the development and popularization of the Internet, the volume of visual data generated around the world grows exponentially. For example, over 500 hours of video are uploaded every minute to YouTube (Hale et al. 2019) and over 300 million new photos are uploaded daily to Facebook (Noyes 2019), making manual processing practically impossible. Making sense of this huge amount of visual data is very useful both for societal and economical reasons. As a consequence, it is becoming important and urgent to develop more efficient methods to process and understand visual data.

When CV started to take shape as a field in the 1960s, its aim was to try to mimic human vision systems and ask computers to tell us what they see (Richard

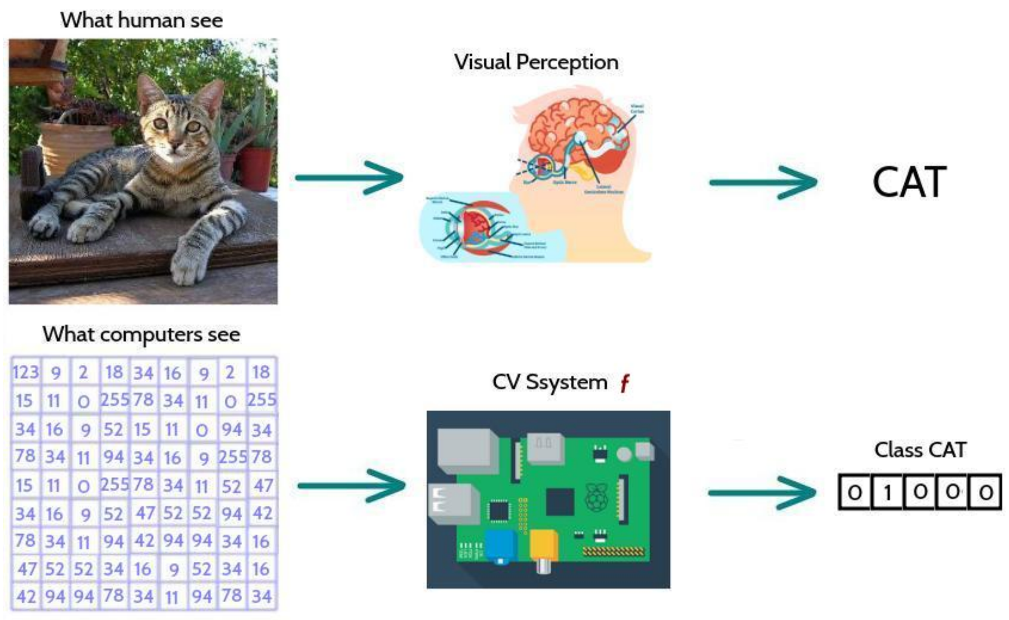


Figure 1.1. – **Image Recognition** Unlike human visual perception, computer vision methods for image recognition define a function f that maps a large matrix to vector representations of semantic classes.

2010). The latter problem, also known as image recognition, is the first step for computers to understand an image and has been one of the most fundamental and difficult problems in the CV domain for decades. As humans, it seems effortless to describe what we see. For example, we can easily recognize a cat from any angle or color and even with an occlusion. However, computers "see" the world differently from us human, as illustrated in Figure 1.1. For computers, an image is just a large matrix (or a tensor) of numbers. Finding a mapping between seemingly unrelated numbers and high-level concepts such as cats or dogs is a very challenging problem.

The naive way to deal with the image recognition problem is to predetermine what an object looks like. For example, a handwritten digit "8" can be defined as two vertically composed circles. However, given the complex structure of real objects as well as various image transformations such as change of scale, color and point of view, this is obviously not a good idea. To that end, CV quickly turned to Machine Learning (ML) methods that do not use explicit instructions but rely solely on patterns and inference. ML algorithms build a mathematical model based on sample data and make predictions or decisions without being explicitly programmed to perform the task. Given an **objective function**, which evaluates the performance of models, and an optimization algorithm, the ML models automatically improve their predictions by simply providing these algorithms the initial sample data. The intuition is that the models will then be able to generalize well to never-before-seen data. The models are thus learning autonomously from the data, hence the term "machine learning".

In order to compare different CV methods, open datasets have been built to evaluate performances within the same framework. One of the most important large-scale datasets for image classification is undoubtedly the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset created in 2009 (J. Deng et al. 2009). This dataset contains more than 1.2 million (which later became 1.4 million) hand-annotated images from 1,000 classes covering a wide range of objects such as "balloon" or "hammer". The images vary in dimensions and resolution, but in a standard setting, all of the images are resized to 224×224 pixels. Since a RGB image of a resolution 224×224 contains 150,528 variables, directly applying ML algorithms on these images seems impossible at that time. Therefore, in the years 2010 and 2011, leading methods used a combination of handcrafted feature extractions and ML models. Feature extraction, also called feature engineering, aims to extract meaningful information from pixel representation of images and to provide much simpler image representations for ML models. Feature extraction was therefore the key component of those CV algorithms. However, specific handcrafting and expertise were required for this step to find features that would best represent an image. These features should not only be robust for different images of the same class but should also be different enough from one class to another in order to correctly distinguish different classes.

The year 2012 is remarkable in CV history because of the victory of Krizhevsky on ILSVRC (Alex et al. 2012). Their method achieved a top-5 error (top-5 error is the percentage of test images for which the correct label is not among the five labels considered most probable by the model) of 15.3%, more than 10.8 percentage points lower than that of the runner up. Actually, they proposed to use a specific Convolutional Neural Network (CNN), called AlexNet, which directly learns the mapping from the raw images (pixel representation) into the semantic prediction, e.g. the classes of ILSVRC. As mentioned before, the large dimension of image data (196,608 pixels in this case) was the main reason why people tended to use feature engineering before applying ML algorithms. So how was AlexNet able to automatically process this huge amount of variables and achieve such results?

To this end, we need to present a brief history of one specific type of ML methods called Artificial Neural Networks (ANN). As the name suggests, ANN were inspired by the biological neural networks that constitute human brains, and the basic building element is called a Neuron. Its functionality is similar to a human neuron which takes in some input and fires output. An artificial neural network is then built from a set of connected neurons where generally, neurons are aggregated into layers. Each layer perform a different transformation on its inputs and the outputs of this layer are then passed to another layer. Therefore, signals travel from one layer to another until to the last layer, possibly after traversing the layers multiple times. This kind of models was proposed by Warren

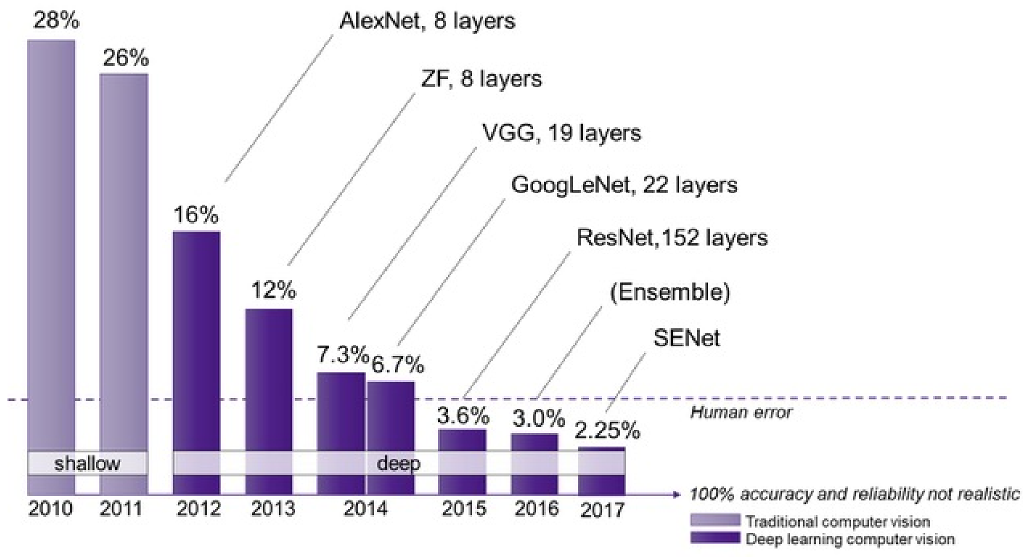


Figure 1.2. – **Historic results on ImageNet** This plot shows the *top-5* error obtained by the annual winner of the ImageNet image classification challenge. A significant performance gap was achieved by AlexNet in 2012. In 2015, ResNet (K. He et al. 2016) exceed human level accuracy. Credits to (Gordon 2019).

McCulloch and Walter Pitts (Mcculloch et al. 1943) in 1943 and since then, many improvements have been made by outstanding researchers. However, the large number of parameters and expensive computations make these models difficult to train especially for multi-layers models. Krizhevsky et al, (Alex et al. 2012) are one of the first teams that successfully trained a multi layer (eight layers for AlexNet, therefore "deep") CNN model for image recognition problem at large scale. Their success is mainly due to the utilization of Graphics Processing Unit (GPU) during training, the large number of data and some ML and optimization improvements such as Dropout. Since then, Deep Learning (DL)-based models have won all ILSVRC competitions. With the increasingly advanced models and optimization algorithms, the top-5 error on the ILSVRC is getting lower and lower, reaching 3.6% in 2015 which actually exceeds human performance on the same dataset (about 5.1%) as shown in Figure 1.2.

With DL, humans no longer have to design complicated feature engineering processes, but rather build a framework of mathematical transformations and let the machine determine itself which information to extract from the raw data. It is also important to note that DL models have proven to be extremely versatile for many other CV tasks as well as tasks in other areas such as Natural Language Processing (NLP). In just a few years, DL-based models have made great breakthroughs in a wide range of tasks and have been applied to a variety of applications. As some researchers assessed, the artificial intelligence industry has entered the era of deep learning.

1.2 Motivation

Although great breakthroughs have been made in image recognition, the problem of image understanding is much more complicated. We are not satisfied with just knowing which objects are in the image; we want to know more about the image, such as where the objects are and what their boundaries are. In other words, we want to partition the image into meaningful regions: pixels in the same region share certain characteristics while adjacent regions are significantly different with respect to the same characteristics. This task is known as **image segmentation**, one of the great challenges throughout the long history of computer vision. Depending on the separation criteria, there are different types of segmentation problems. The most classic one is **semantic segmentation** where the separation criteria is the semantic meaning. Each pixel is classified into a unique class from a set of predefined semantic classes as presented in Figure 1.3, and a connected region with the same label represents an "object" in semantic segmentation task. In this manner, we know exactly where the person and the plants are in the image as well as their explicit semantic contours. Other type of image segmentation can also be formed accordingly. For example, foreground-background segmentation aims to segment the most salient object in the image while instance segmentation solves the problem of segmenting multiple instances of the same object in a scene. Segmentation in the temporal space is also an important domain. For example, in object tracking and motion segmentation problems, pixels must be classified not only based on spatial information but also across time.

Segmentation algorithms have many important and interesting applications in the real world. In autonomous driving systems, semantic segmentation algorithms are used to fully analyze the environment so that self-driving cars can safely integrate into existing roads. In robotics, segmentation algorithms can provide the exact position and contour of objects. Another important area is medical image processing, where segmentation algorithms are essential in applications such as locating tumors and other pathologies, measuring tissue volumes and the simulating virtual surgery. Other application areas include photo (video) editing, image retrieval, visual question answering, satellite imagery, video surveillance, traffic control systems, action recognition etc.

Due to the large number of applications of segmentation algorithms and the recent success of CNNs in image recognition, more and more research is attempting to adapt CNN models to image segmentation problems. In this thesis, we focus on the classical problem of segmentation: semantic segmentation, and we will

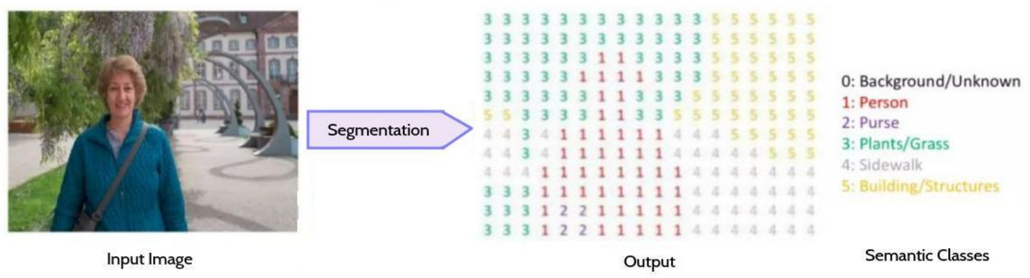


Figure 1.3. – **Semantic Segmentation** An illustration of the semantic segmentation task. The whole image is partitioned into segments based on semantic meaning. Each pixel in the image is classified into one of the six predefined semantic classes and a connected region with the same label represents an "object". Illustration inspired by (Jeremy 2019).

mainly address three major difficulties of using DL-based models for semantic segmentation.

D1: Spatial and contextual information Unlike the problem of image recognition, which should only determine the semantic class of the object contained in the image, semantic segmentation requires a much finer understanding, so that the semantic class of all pixels in the image must be determined. Therefore, spatial and contextual information is crucial for semantic segmentation. **Spatial information** contains all spatial relationships between pixels, such as the relative position between pixels or the location of edges, and is essential for determining the position and boundary of objects. **Contextual information**, on the other hand, contains the semantic relationships between pixels, allowing the prediction of a pixel's class to remain consistent in context. However, CNNs designed for image classification often discard these information because their goal is to identify objects, not their positions, contours, and relationships to each other. Particularly, operations like pooling, stride and flatten are used in classification CNNs, which reduce spatial resolution and extract robust and simpler features for classification. In order to build suitable CNN models for segmentation, it is crucial to better explore and integrate spatial and contextual information into the network or learning process.

D2: Objective function As mentioned above, the objective function (also known as the loss function) is the function used to evaluate the performance of the model, and it determines the predictions we want the model to produce. Different tasks require different objective functions, and finding a suitable objective function for semantic segmentation is not trivial. In the image classification problem, the cross-entropy loss function is the standard choice for training. Since

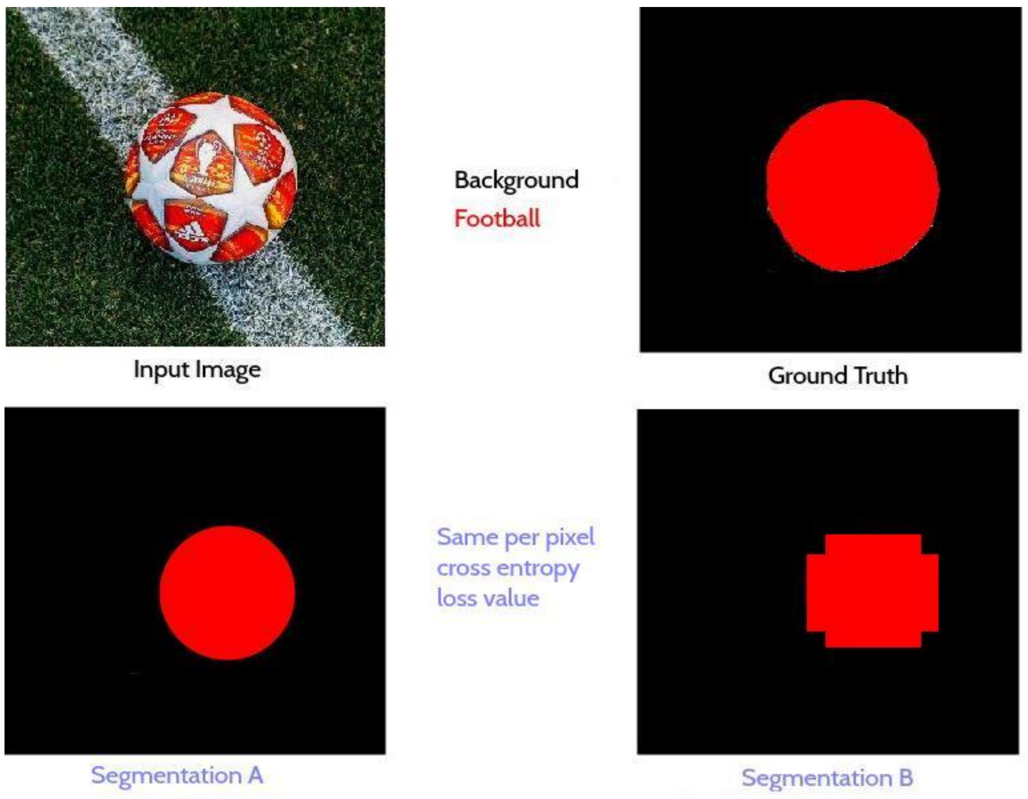


Figure 1.4. – **Loss Function** An example of two segmentation predictions (A and B) that have same loss value for per-pixel cross entropy. Prediction A provides a smaller circle while prediction B is composed of two rectangles. Though both predictions are not perfect and have same loss, prediction A is preferred.

segmentation can be regarded as a pixel-wise classification problems, the most natural loss function for segmentation is the pixel-wise cross-entropy loss. With this loss function, each pixel is considered to be equally important and the performance of segmentation result is the average performance of all pixels. However, this loss function has many drawbacks, such as the problem of data imbalance, where some categories have more data than others. Another major problem is that this loss does not take into account the shape of each segment. Let's consider a simple example in which the image contains a football where the ground truth segmentation mask is only a circle as shown in Figure 1.4. Suppose that there are two different predictions S_a and S_b both have 80% pixels correct. S_a has a circular form while S_b is composed of two rectangles. Both predictions have the same loss value under per-pixel cross-entropy loss, but we probably prefer the prediction S_a as it preserves the shape of the object. As a consequence, defining appropriate loss functions is an important problem for segmentation models and can help the model obtain results that respect shapes and geometry of objects.

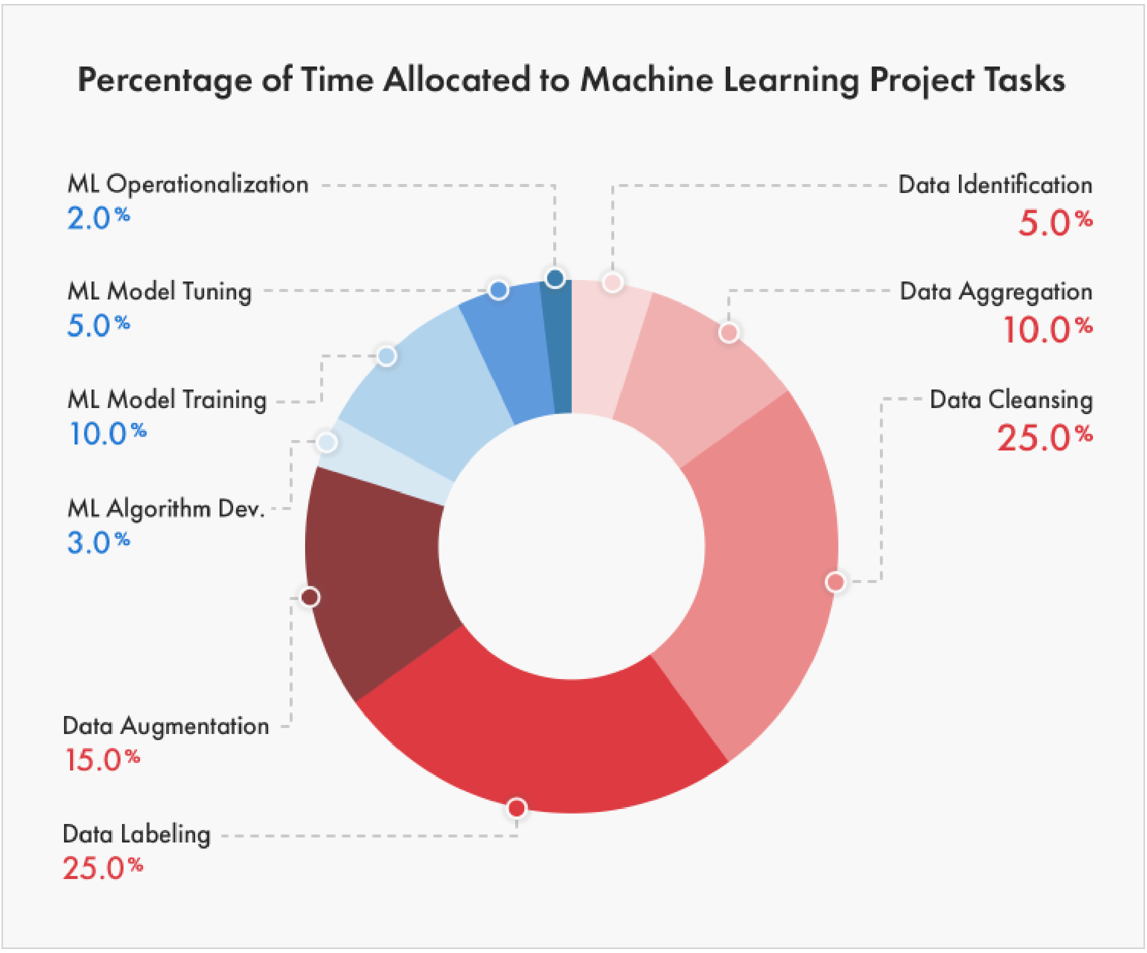


Figure 1.5. – Percentage of time allocated to machine learning projects according to analyst firm Cognilytica (CloudFactory 2019). 80% of machine learning project time is spent on data preparation. This analysis reflects, to some extent, the importance of labeled data and the cumbersomeness of the data labeling process. This motivates the use of alternatives to fully supervised learning, such as weakly supervised learning.

D3: Lack of data As mentioned earlier, one of the main reasons for the success of DL models is the large amount of annotated data. For example, the ImageNet dataset provides one million manually annotated images for training, which obviously requires a lot of time and work. As estimated by the analyst firm Cognilytica (CloudFactory 2019), 25% of AI project time is spent on data labeling and 40% of time is spent on gathering, organizing, and cleaning data (see Figure 1.5). Moreover, in the context of semantic segmentation, all pixels of an image must be annotated, which requires meticulous effort to obtain accurate annotations at object boundaries. Thus, creating a large scale fully annotated dataset for semantic segmentation task is extremely time-consuming and typically requires substantial financial investments. For example, it takes an average of 239.7 seconds to manually annotate a 500×500 image (A. Bearman et al. 2016) since it requires a total of 250,000 labels. In addition, some objects can have extremely complicated forms such as melanoma in medical images. It is usually very difficult to have perfect annotations for boundary pixels. Therefore, it is virtually impossible to have one million fully annotated images for segmentation tasks as is the case for image recognition. In the context of segmentation, it is thus particularly interesting to develop methods which use unannotated or partially-annotated images.

1.3 Contributions and Outline

In this thesis, we tackle the semantic segmentation problem from a DL perspective and mainly address the three problems described in Section 1.2. Most of the research has focused on problem $D1$, mainly by modifying the network structure and introducing different modules to make better use of spatial and contextual information. Instead, we believe that the network can also learn spatial and contextual information through an appropriate loss function. In combination with the other problems described in $D2$, it leads us to the work presented in Chapter 3. Then, in Chapter 4, we tackle the problem $D3$ in a specific Weakly Supervised Learning (WSL) framework which only uses image-level labeled images to train segmentation models.

The manuscript is organized as follows:

- Chapter 2 RELATED WORKS
  In the first part, we propose an overview of semantic segmentation algorithms, and position the work of this thesis.
  Starting from some of the traditional approaches to image segmentation, we provide an intuitive understanding of the main techniques that have significantly contributed to the field of semantic segmentation. In particular, we provide a comprehensive and in-depth analysis of DL-based approaches. Through a detailed study of the milestone work Fully Convolutional Network (FCN) (Long et al. 2015), we present the main advantages and challenges of applying DL methods for semantic segmentation. Thereafter, most of the recent segmentation algorithms have been logically categorized according to the described challenges. With a large number of intuitive explanations, we can better understand the state-of-the-art segmentation methods and gain insight into future research directions.

- Chapter 3 SEMEDA: ENHANCING SEGMENTATION PRECISION WITH SEMANTIC EDGE AWARE LOSS
  For the second part, we present a new loss function called SEMEDA (SE-

Mantic EDge-Aware) loss to better constraint the segmentation models.
As we described in *D*2, the loss function commonly used for semantic segmentation is a naive extension of the loss function used for image classification. Among several problems of this loss function, a major drawback is that it cannot measure the structural similarity (such as shapes and geometries of object) between the predicted mask and the ground truth mask, as it considers each pixel independently.
To overcome this problem, we introduce the SEMEDA loss function. This loss function is defined using a semantic edge detection network whose features encode local structure information of segmentation masks. We validate SEMEDA loss on several datasets ranging from classical image datasets to a face dataset and a medical dataset with a comparison to other leading methods. We also propose an in-depth analysis of the different terms of the loss function as well as visualizations to gain insight regarding the behavior of the model.

- Chapter 4 WEAKLY SUPERVISED SEGMENTATION WITH ATTRIBUTION METHODS
  Finally, in the third part, we tackle the WSL segmentation problem with attribution methods. We propose a novel attribution method VGatt (Value-Gradient Attribution) to explain and interpret the prediction of CNNs and then integrate this method into a WSL segmentation framework.
  The lack of fully annotated data is a major concern for DL-based methods, as described in *D*3. To solve this problem, WSL methods propose to use partially labeled data to train segmentation models, especially with images that have only labels at the image level. The main idea is to extract the relevant pixels from the image-level labels and to consider them as ground truth to train the segmentation models. To this end, we propose a new attribution model that mathematically decomposes the output of CNNs on pixels contributions, and can therefore be used to identify the most relevant pixels for the image labels. We then integrate our method into WSL segmentation methods and validate the model on Pascal VOC 2012.
  Although major part of our work is focused on semantic segmentation, the proposed attribution method can apply to broader contexts. We finish this part by applying our method to understand network behavior and find adversarial examples.

We conclude this thesis in Chapter 5 and discuss several interesting directions for future work.

1.4 Related Publications

This thesis is based on the material published in the following papers:

- Yifu Chen, Antoine Saporta, Arnaud Dapogny, and Matthieu Cord (2019). "Delving Deep into Interpreting Neural Nets with Piece-Wise Affine Representation". In: *IEEE International Conference on Image Processing (ICIP)*;
- Yifu Chen, Arnaud Dapogny, and Matthieu Cord (2020). "SEMEDA: Enhancing Segmentation Precision with Semantic Edge Aware Loss". In: *Pattern Recognition (under review, major revision)*;

Chapter 2

RELATED WORKS

Contents

Chapter abstract

As we stated in the introduction, image segmentation has long been one of the fundamental problems in computer vision due to its wide applications. However, it is only now that we have really seen possible solutions to these problems. With the success of the Convolutional Neural Network (CNN), CNNs-based methods are being increasingly applied to semantic segmentation and have achieved extraordinary improvements. Among these approaches, Fully Convolutional Network (FCN) (Long et al. 2015) is considered the milestone in the semantic segmentation literature: almost all current state of the art methods can be considered as extensions or improvements of FCN. Since the work of this thesis is focused on applying CNNs methods for semantic segmentation, understanding the pros and cons of FCN is important to develop better methods.

In this chapter, we chronologically examine semantic segmentation algorithms and focus on CNN-based methods. First, we briefly present traditional techniques before the Deep Learning (DL) era and then we give an in-depth analysis of the FCN model. After analyzing the shortcomings of FCN, we present recent

improvements in CNN-based methods. Finally, we expose our contributions and position them with respect to the existing literature.

2.1 Traditional Methods

Image segmentation is the problem of partitioning pixels into groups according to specific criteria. Particularly, semantic segmentation consists in grouping pixels according to their semantic meaning. This task is generally considered as the third step in image understanding: which objects are present in the image (recognition), where they are located (detection) and what are their semantic boundaries (semantic segmentation). In order to build a system that determines the semantic meaning at the pixel level, both low and high level features of the image are required. Low level features, such as edges, can be obtained from local variations in intensity, while high level features require a semantic understanding of the image. As a result, the development of semantic segmentation heavily relies on image recognition algorithms. Before the image recognition problem is solved, semantic segmentation is undoubtedly an extremely complicated problem. This is why people usually tackle other segmentation problems that do not require semantic information, such as foreground background segmentation or unsupervised image segmentation. Some image segmentation processes are also considered as important steps for other image analysis. The result of segmentation provides simpler and more effective representations of the image and thus facilitates a high level understanding of the images.

Early algorithms for image segmentation can be generally divided into three groups: region-based, edge or boundary based and graphical models based methods (Figure 2.1). **Region-based methods**, including clustering, region growth and thresholding, are mainly based on some uniformity criterion to define regions. The criterion is based on some region property depending on applications, and could be one of many measurable image attributes such as intensity, color, distance, *etc*. Taking a popular region growth method (Pavlidis 1972) as example, it initializes all pixels as regions and iteratively merges adjacent regions if the variance of the grouped region computed on gray level values is less than a threshold. On the other hand, **Edge-based methods** aim to define segments by their boundary, since edges are a sign of lack of continuity and can therefore be used to separate regions. This is a complex process that usually takes place in two steps: edge point detection and grouping edge points into contours. The main feature used to determine whether a pixel can be classified as an edge is its intensity gradient. The higher the gradient the more likely it is an edge point. A number of methods have been proposed for edge detection, and most of them use the method of convolution of the image with a filter such Prewitt (Prewitt 1970) and Sobel detection (Sobel 2014). After obtaining edge maps from a raster image,

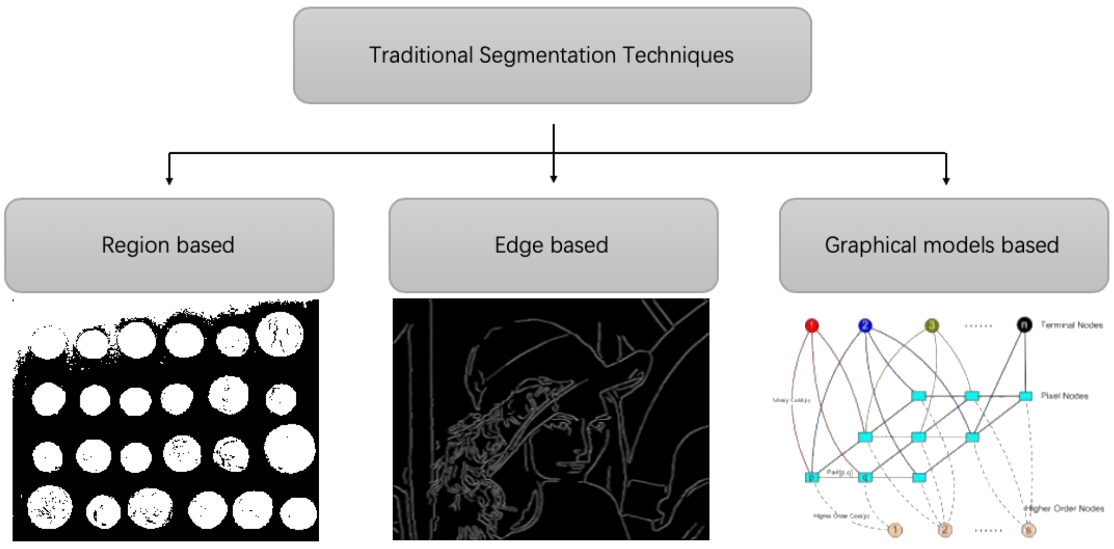


Figure 2.1. – Traditional methods for Image Segmentation can be generally divided into three groups: regions-based, edge-based and graphical model based models.

algorithms such as the active contour (Kass et al. 1988) (also called snake) are then applied to these edge maps to define closed formed boundaries. However, for images that contain many objects with a lot of detail, it is difficult to decide which boundary segment goes with which object, so edge-based methods often fail in this case. To a comprehesive survey of these approaches, see (Cocquerez et al. 1995).

The image segmentation methods mentioned above are mainly based on local features (i.e. edges, blobs) and the long-range dependency between pixels is not well exploited. To this end, **probabilistic graphical models**, such as Conditional Random Fields (CRF) (Lafferty et al. 2001), are often integrated into semantic segmentation systems to model pixels dependencies (Ulusoy et al. 2006; X. He et al. 2009; Shotton et al. 2008; Lempitsky et al. 2011; Krähenbühl et al. 2011). CRFs are a type of discriminative undirected probabilistic graphical model that are used to encode known relationships between observations and to construct consistent interpretations. The fully connected pairwise CRF is one of the most used model in the context of semantic segmentation. It is mainly characterized by a Gibbs energy function with a unary and a pairwise potential function defined in Equation 2.1:

$$E(x|I) = \underbrace{\sum_i \varphi_u(x_i), \theta_u}_{\text{unary potential}} + \underbrace{\sum_{i,j} \varphi_p(x_i, x_j), \theta_p}_{\text{pairwise potential}} \tag{2.1}$$

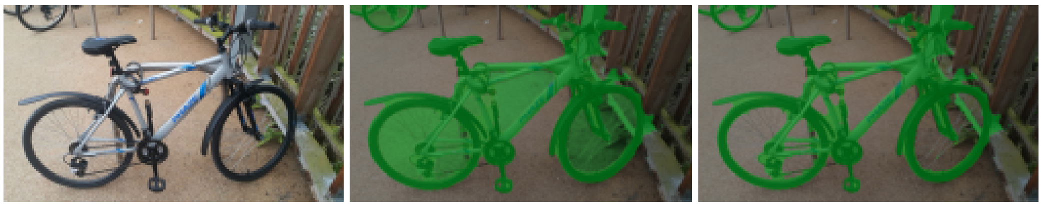

(a) Input Image (b) Segmented Image (c) Refined Result

Figure 2.2. – Effect of using graphical model based models as refinement on segmentation results. Credits to (Irem et al. 2019).

where x represents the segmentation mask of the image I, x_i is the prediction of pixels i and θ_u, θ_p are parameters of the potential functions. The unary potential $\varphi_u(x_i)$ is computed independently for each pixel by a classifier that produces a distribution over the label assignment x_i given image features. Since the outputs of the unary classifier for each pixel is produced independently, they are generally inconsistent. To this end, the pairwise potential $\varphi_p(x_i, x_j)$ encourages the prediction x_i, x_j to be consistent with the hand-crafted features f_i, f_j of pixels i, j. Depending on the applications, different pairwise potentials can be defined to capture the corresponding pixels dependency. Finally, maximum likelihood method is applied to determine the parameters θ_u, θ_p in the training phase. And in the inference phase, CRF aims to achieve a global optimization by finding a optimal x^* that minimizes the total energy E. CRFs are powerful models for segmentation, but they are mainly limited by the performance of the unary classifier, the choice of the potential functions and the optimization algorithm used in training and inference.

Although some of the traditional image segmentation methods mentioned above are no longer preferred for semantic segmentation in the Deep Learning (DL) era, the ideas behind them are still indicative for recent research. For example, CRFs are either integrated into deep learning methods or used as post processing for refinement with the purpose of improving segmentation performance as showed in Figure 2.2. The duality between region and edges is also considered in the recent deep learning approaches for semantic segmentation.

2.2 Fully Convolutional Network for Semantic Segmentation

In this section, we detail the Fully Convolutional Network (FCN) model (Long et al. 2015), which has been considered as the cornerstone of deep learning applied to segmentation semantics. It is one of the first works that demonstrates

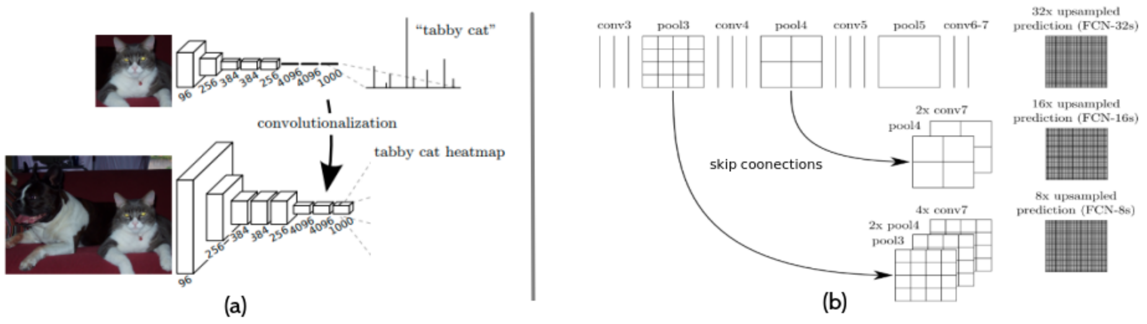

"tabby cat"
96 256 384 384 256 4096 4096 1000
convolutionalization
tabby cat heatmap
(a)
conv3 pool3 conv4 pool4 conv5 pool5 conv6-7
32x upsampled prediction (FCN-32s)
2x conv7
pool4
skip coonections
16x upsampled prediction (FCN-16s)
4x conv7
2x pool4
pool3
8x upsampled prediction (FCN-8s)
(b)

Figure 2.3. – (a): Removing flatten operation and transforming fully connected layers into convolution layers enables a dense classification. (b): Deconvolutions are used to upsample feature maps and skip connections can incorporate lower level information into the final prediction by adding them together. Illustration inspired by (Long et al. 2015).

how Convolutional Neural Network (CNN)s can effectively learn to make dense predictions for semantic segmentation in an end-to-end fashion. Many recent state-of-the-art methods can be seen as extensions and improvements of the FCN. Thus, a good understanding of the advantages and limitations of the FCN can help us to understand other works and provides insight into future research.

2.2.1 Principle and Model

After the huge success of CNNs on image classification, the application of CNNs to semantic segmentation is a natural step. However, due to differences between tasks, the CNN methods used for classification cannot be applied directly on semantic segmentation. For image classification problems, their goal is to recognize the object in the image regardless of location or boundaries. As a result, spatial information is generally discarded in classification CNNs, both to speed up computation and to obtain more robust representations. On the contrary, spatial information is the key for semantic segmentation since we need to classify each pixel in the image. The challenge, therefore, is how to retain more spatial information while maintaining discriminating features for classification.

Precisely, given an input image $I \in [\mathbf{0},\mathbf{1}]^{\mathbf{H}\times\mathbf{W}\times\mathbf{3}}$ and a pre-defined label set $L = 1, 2, ..., C$ where H (resp. W) is the height (resp. width) of the image and C is the total number of classes, a classification model is a function from $[\mathbf{0},\mathbf{1}]^{\mathbf{H}\times\mathbf{W}\times\mathbf{3}}$ to $\mathbf{R}^C$ while a semantic segmentation model is a function to $\mathbf{R}^{H\times W\times C}$. As can be seen, the classification problem requires the model to convert the volume of the 2d image into a single vector. To do this, a key operation in classification CNNs is the flatten operation, which transforms the 2d activation map into a 1d vector and allows the execution of fully connected operations. However, this kind of reshaping operation discards the spatial relations among pixels in the image. In addition, the fully connected layer requires a fixed input size and therefore the network can only be applied on images of a fixed size. With this in mind, the **first contribution** of the FCN is to remove the flatten operation and replacing fully connected layer by a 1×1 convolution layer. In this manner, the modified network maintains 2d activation maps to produce dense predictions, and can be applied to an image of any size. (see Figure 2.3 (a))

However, that is not enough to produce high quality segmentation masks. Apart from the flatten operation, classification CNNs progressively reduce the spatial resolution by subsampling operations such as stride, and usually the last feature map is 32 times smaller than the original size. For example, a 224×224 resolution image becomes a 7×7 feature map after all convolution layers. It is obvious that the segmentation mask provided on this 7×7 feature map will be very coarse and inaccurate. However, simply removing these subsampling operations within a network is a trade-off: the filters see finer information, but have smaller receptive fields and take longer to compute. **The second contribution** of FCN is to solve this problem by retaining the receptive field and adding skip connections and deconvolutions to refine the predictions. (see Figure 2.3 (b)). Skip connections can incorporate finer information (low level features) into the final prediction and the deconvolution is used as an up sampling method to perform higher resolution outputs.

Thanks to these two major contributions, the FCN can transform most popular classification CNNs such as VGG (Simonyan et al. 2015) and ResNet (K. He et al. 2016) into a fully convolutional version and then used them for semantic segmentation. In the following section, we describe the learning strategy proposed by FCN.

2.2.2 Learning Strategy

A supervised machine learning algorithm generally consists of three parts: the model (or the hypothesis space), the objective function and the optimization method over a dataset. We have already introduced the FCN model in Section 2.2.1, and in this section, we present the training strategy of FCN.

Classification models usually use the cross-entropy loss as objective function. This loss is computed between the predicted distribution $\hat{y} \in [\mathbf{0},\mathbf{1}]^C$ and the ground truth distribution $y^* \in [\mathbf{0},\mathbf{1}]^C$ by the formula:

$$L_{CrossEntropy}(\hat{y}, y^*) = -\sum_{i=1}^{C} y_i^* * log(\hat{y}_i) \tag{2.2}$$

where C is the total number of classes. The loss is then optimized by stochastic gradient descent methods and generally performs well for classification tasks. However, this objective function is not directly applicable to semantic segmentation since the predicted mask $\hat{S}$ and ground truth mask S^* are not distributions. These masks belong to the space $[\mathbf{0}, \mathbf{1}]^{H \times W \times C}$ where for each position (i, j), the vector $\hat{S}_{i,j}$ and $S^*_{i,j}$ are distributions. Therefore, FCN proposes to use the Per Pixel Cross Entropy (PPCE) loss which is a sum over the spatial dimensions of masks:

$$L_{PPCE}(\hat{S}, S^*) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{i=j}^{W} L_{CrossEntropy}(\hat{S}_{i,j}, S^*_{i,j}) \tag{2.3}$$

Gradients of this function is the sum over the gradients of each of its spatial components. Thus stochastic gradient descent on L_{PPCE} computed on whole images will be the same as stochastic gradient descent on $L_{CrossEntropy}$, taking all of the final layer receptive fields as a mini-batch. With this loss function, both feed-forward computation and back-propagation can be efficiently computed over an entire image.

As a result, a classification CNN be can transformed to a FCN version and then be efficiently trained in an end-to-end fashion with stochastic gradient descent methods. With minor modifications, a high-performance classification network can be easily used for semantic segmentation tasks. These advantages make the FCN a milestone in the application of deep learning methods for semantic segmentation.

2.2.3 Dataset and Evaluation

Datasets Since Machine Learning (ML)-based methods are data-dependent, open datasets are needed to evaluate these methods in the same configuration for a fair comparison. There are mainly two kind of datasets for semantic segmentation tasks: general purpose datasets and application-based datasets.

- General purpose datasets: These datasets contain generic class labels, typically created for research purposes. One of the most popular general purpose datasets is PASCAL VOC (Visual Object Classes) (Everingham et al. 2009), which is still active since its initial release in 2005. The image set and annotations are regularly updated and the leaderboard of the segmentation challenge contains hundreds of submissions, including all the latest algorithms. The image set covers a wide range of scenes from everyday life, both indoors and outdoors. It contains 20 foreground categories, including "TV" and "sofa" as well as "bird" and "car". The original data contains 1,464 images for training, 1,449 images for validation and 1,456 test images, and is then expanded by (Hariharan et al. 2011) to 10,582 fully annotated training images. Other popular general purpose datasets include MS COCO (T.-Y. Lin et al. 2014) and ADE20k (Zhou et al. 2019).

- Application-based datasets: These datasets are built for specific segmentation applications and contain only images of same domains, such as urban street scene images, satellite images or medical images. One of the most popular application-based datasets is Cityscapes (Cordts et al. 2016), which focuses on the semantic understanding of urban street scenes. It contains annotations for high-resolution images of 50 different cities, taken at different times of the day and in all seasons of the year, with variable background and scene layout. The annotations are made at two quality levels: fine for 5,000 images and coarse for 20,000 images. There are 19 foreground classes for semantic segmentation, including classes of scenes such as sky, road or trees, and classes of things such as different types of vehicles or traffic signs.

The performance of segmentation algorithms can be affected by the datasets in many ways such as the amount of data, the definition of semantic classes, the quality of the image and the annotation and the biases of the dataset. A segmentation algorithm can score high on one dataset and fail on another. Therefore, evaluations on multiple datasets are necessary to demonstrate the effectiveness of an algorithm.

Evaluation Measuring segmentation performance is not an easy task, as it is difficult to accurately measure the segmentation quality of an entire image with a single score. Although segmentation is often considered as a pixel-level classification, pixel-wise accuracy is not a good measure of segmentation because it is easily affected by class imbalance: in an image with a large background area and very small objects, predicting "background" for all pixels yields a very high pixel accuracy. To better evaluate the performance of segmentation, many measures have been proposed. The most widely used is the Intersection over Union (IoU) score, which is also known as the Jaccard index. The mean Intersection over Union (mIoU) score is the averaged IoU that measures the segmentation performance over all classes. It is the main index for ranking semantic segmentation methods in many datasets.

If we note k the total number of semantic classes, and S^* ground truth segmentation mask, the mIoU score of the prediction $\hat{S}$ is :

$$mIoU(\hat{S}, S^*) = \frac{1}{k}\sum_{j=1}^{k} IoU(\hat{S}_j, S_j^*) \tag{2.4}$$

$$= \frac{1}{k}\sum_{j=1}^{k} \frac{|\hat{S}_j \cap S_j^*|}{|\hat{S}_j \cup S_j^*|} \tag{2.5}$$

where $|.|$ is the cardinal function that refers to the number of pixels in the corresponding regions. An illustration of the IoU score is presented in Figure 2.4.

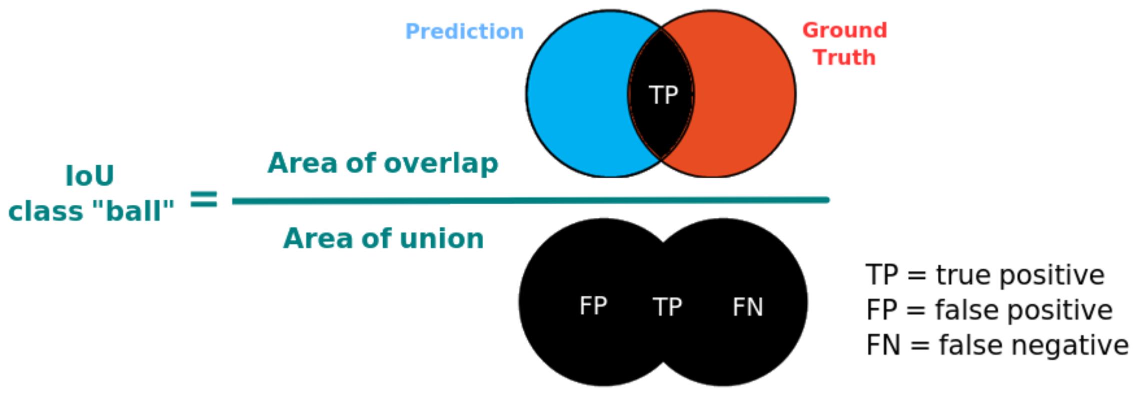


Figure 2.4. – Illustration of the IoU score for the class "ball". The IoU is simply the area of overlap divided by the area of union, where "area" refers to the number of pixels in the region. The mIoU is the average IoU score over all semantic classes.

It is obvious that $0 \leq mIoU \leq 1$ and that, the larger the value of mIoU, the better the segmentation results.

2.2.4 Challenges

FCNs were first implemented to perform semantic segmentation in the context of PASCAL VOC 2011 segmentation dataset (Everingham et al. 2015) and achieved a pixel accuracy of 90.3% and a mIoU of 62.7%, which were the state-of-the-art performance back then. However, as the mIoU score indicates, these results are far from perfect. Based on the three components of machine learning algorithms that we introduce in Section 2.2.2, we analyze the challenges that remain within the FCN framework in each of these three areas.

Models (Network architectures) In order to build a better segmentation model, it is essential to make good use of **spatial** and **contextual** information as described in $D1$.

The lack of sufficient spatial information results in loss of spatial details for segmentation. In FCN, the last feature map has a resolution 32 times smaller than the input image, which results in a large loss of spatial information. The segmentation mask predicted from this small feature map alone will of course be coarse, as a lot of the spatial information is lost in the process. However, maintaining a high resolution across networks is not straightforward for two main reasons. First, CNNs need a receptive field large enough to extract information from a region that covers objects of all sizes. The most efficient way to increase the receptive field is to apply sub sampling operations which, however, reduce the spatial resolution. Second, a higher resolution also leads to a higher memory usage. This will increase the complexity of the model and make the training process more difficult. Therefore, retaining as much spatial information as possible while having a large receptive field and keeping the model simple is the **first challenge** of architectural design.

The lack of corresponding contextual information leads to inconsistent and noisy pixel-wise predictions. Essentially, CNNs classify a pixel based on a region of pixels around it. This region is actually the receptive field of this pixel. Due to the translation invariant nature of CNNs, features at different spatial locations in the same layer have the same receptive field. This means that in a standard CNN, all pixels are classified according to the same context size. This is problematic and introduces local ambiguities in the pixel classification because different pixels have different contexts. On the one hand, for pixels belonging to a large object, a small context causes the network to predict without seeing the full object. On the other hand, for pixels describing small objects, a large context can make it harder for the network to focus on the corresponding object. As a result, how to effectively incorporate relevant contextual information into the network is the **second challenge**.

Objective function The objective function, also called loss function, is one of the most fundamental components of a ML problem, in that it provides the basic, formal specification of the problem. Essentially, an objective function is a measure of model performance, and it directs the training process to find the best models within that measure. In the context of semantic segmentation, the most common strategy, as done in FCN, is to consider segmentation as a pixel-level classification problem and to use PPCE loss function (see Equation 2.3). This is mainly motivated by its simplicity and efficiency of the calculation. However, this loss function has many drawbacks as discussed in *D*2. First, this function is not the metric used in the evaluation. This could results in the best model obtained through training not performing well according to the evaluation metric. Second, PPCE loss is the average of all pixels, which potentially suffers from the data imbalance problem. With PPCE, models tend to fit more on classes that have large objects, because the accuracy on small objects is less important. Last but not least, PPCE cannot measure the structural similarity between predicted masks and the ground truths. For example, the mask of a "football" can have a low PPCE loss value without being a circle, as described in Section 1.2. Considering these drawbacks, it is thus important to define some loss functions that are more adapted for semantic segmentation.

Optimization method over datasets The success of DL relies heavily on the Stochastic Gradient Descent (SGD) methods (Bottou et al. 2008) that can effectively solve the optimization problem defined by the ML problem, especially in the case of large-scale problems. With a differentiable objective function and sufficient training data, SGD methods can efficiently find the model parameters that result in a low loss value over the entire data set, even if there are millions of parameters in DL models. It is important to note that this process usually requires a lot of data for the parameters to go from a random initialization to an optimal (not necessarily global optimal) solution.

These optimization methods are also used for training semantic segmentation models. The most popular approach is currently full supervision, where the ground-truth segmentation mask is observable for training as in FCN. However, as described in *D*3, the data labeling process for the segmentation task is extremely time consuming, and it is not always easy to collect a fully annotated dataset. Hence, although fully-supervised methods achieve the best performance, in the case of limited annotation resources, Weakly Supervised Learning (WSL) approaches that solve the semantic segmentation problem with less informative annotations are important.

2.3 Optimizing Deep Architectures for Segmentation

Recently, in the spirit of Automated Machine Learning (AutoML), there has been significant interest in designing neural network architectures automatically, instead of relying heavily on expert experience and knowledge. Neural Architecture Search (NAS) has successfully identified architectures that exceed human-designed architectures on large-scale image classification problems (Zoph et al. 2018; Chenxi Liu et al. 2018; Real et al. 2019), and there exist recent efforts (L.-C. Chen et al. 2018; Chenxi Liu et al. 2019) of applying NAS on semantic segmentation that have shown strong experimental results.

However, NAS is usually time-consuming, and the obtained architectures are less intuitive and inspireful. To better understand the challenges of segmentation, in this section, we present the recent improvements of human designed networks based on the two challenges discussed above: spatial information and contextual information. It is worth noting that many works design their networks by taking into account both spatial and contextual information since they are the key to segmentation.

2.3.1 Spatial Information

Many works have been done to address the spatial information problem of CNNs for segmentation. As analyzed above, the main reason for the loss of spatial information is the receptive-field and the spatial resolution trade-off: sub sample

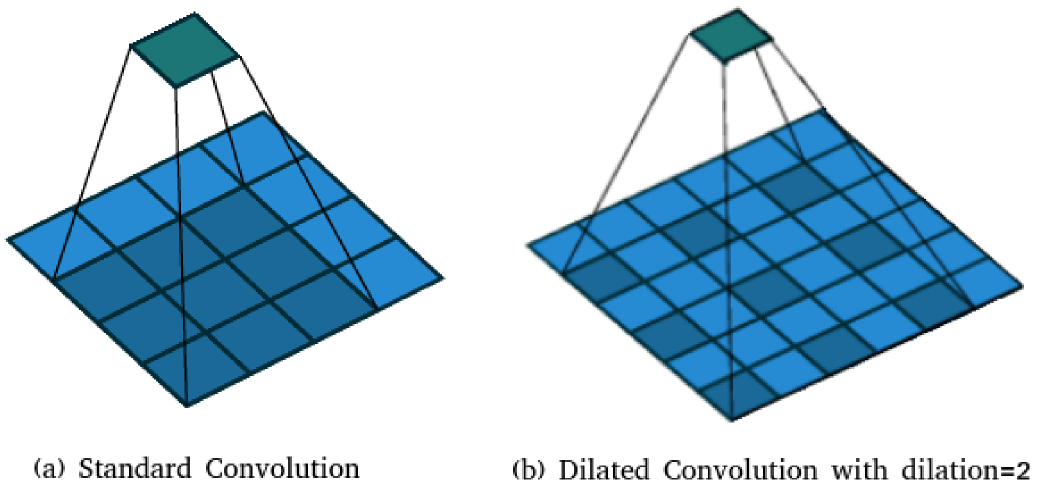
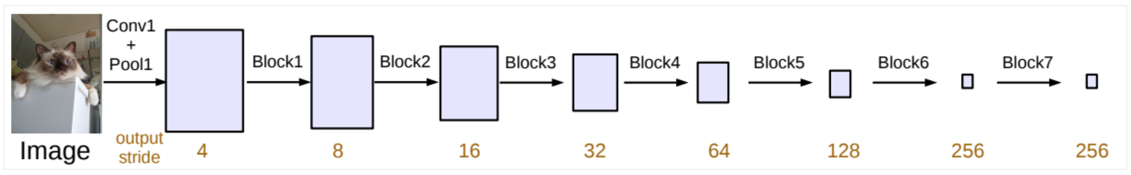


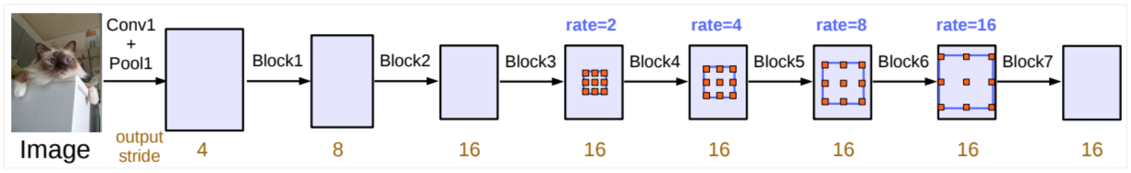
Figure 2.5. – (a): Standard Convolution. (b): Dilated Convolution with dilation 2. Dilated convolution corresponds to regular convolution with dilated filters and thus increase the receptive field without additional parameters.

operations are used to increase receptive field while these operations lead to decrease of spatial resolution. In order to maintain spatial information as much as possible while having a large receptive field, Deeplab (L.-C. Chen et al. 2014) proposes to use **dilated (atrous) convolution**. Dilated convolution is actually a regular convolution with dilated filters as showed in Figure 2.5. Image (a) represents a regular convolution with kernel size 3 and image (b) is a dilated convolution with kernel size 3 and dilation 2. Both operations have 9 parameters while dilated convolution has a larger receptive field (5) compared with regular convolution (3).

In this manner, dilated convolutions enlarge the receptive field without sub-sampling operations or additional parameters. By removing sub sampling operations in CNNs and insert dilated convolution layers, Deeplab (L.-C. Chen et al. 2014) effectively increases the spatial resolution by a factor of 16 while maintaining the same receptive field as the original CNNs, and yields better segmentation results (see Figure 2.6). This technique is very effective and can be applied on most of popular CNNs. Thus dilated CNNs are used as backbone networks by many state of the art methods (L.-C. Chen et al. 2016; L.-C. Chen et al. 2017; Liang-Chieh Chen et al. 2018; H. Zhao et al. 2017; Yuan et al. 2019; Zilong Huang et al. 2019; Fu et al. 2018). The negative aspect of dilated convolution concerns its higher demand for GPU storage and computation, since the feature map resolutions do not shrink within the feature hierarchy (K. He et al. 2016). In addition, dilated



(a) Standard convolution neural networks



(b) Convolution neural networks with dilated convolution

Figure 2.6. – (a): Standard CNNs. (b): CNNs with dilated convolution. Dilated convolution with *dilation* > 1 is applied after block3 in (b), allowing the modified CNNs to maintain a higher spatial resolution while keeping the same receptive field. Credits to (L.-C. Chen et al. 2017)

convolutions suffer from the gridding artifacts since their filters are padded with "0" (Zhengyang Wang et al. 2018; Mehta et al. 2018).

Another approach to handle the loss of spatial information is to reuse the lower-level features. The idea is that low-level feature maps have a higher resolution and contain more detailed spatial information. By incorporating these features into the final prediction, spatial information can be therefore recovered. One way consists in exploiting the encoder-decoder architecture. Typically, an encoder-decoder network contains (1) an encoder module that gradually reduces the feature maps and captures higher level semantic information, and (2) a decoder module that gradually recovers the spatial information, as shown in Figure 2.7. Methods based on encoder-decoder architecture generally differ in how low-level features are used and how they are upsampled. For example, (Long et al. 2015; Noh et al. 2015) employ deconvolution (Matthew D. Zeiler et al. 2011) to learn the upsampling of low resolution feature responses. SegNet (Badrinarayanan et al. 2017) reuses the pooling indices from the encoder and learn extra convolution layers to refine the feature responses, while U-Net (Ronneberger et al. 2015) adds skip connections from the encoder features to the corresponding decoder activations, and (Ghiasi et al. 2016a) employs a Laplacian pyramid reconstruction network. More recently, RefineNet (G. Lin et al. 2017) and (Brahimi et al. 2019; Pohlen et al. 2017; Peng et al. 2017; Amirul Islam et al. 2017) have demonstrated the effectiveness of models based on encoder-decoder structure on several semantic segmentation benchmarks.

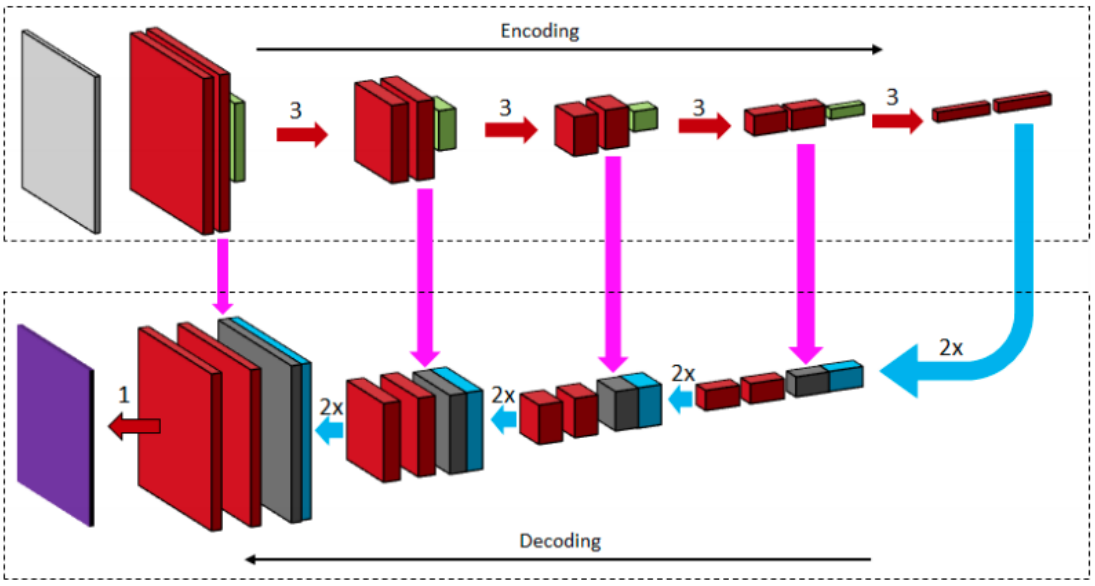


Figure 2.7. – An example of encoder-decoder architectures. During encoding, the network progressively captures the higher semantic information and during decoding, it gradually recovers the spatial information by concatenating encoder's features (purple arrows). Credits to (Ghosh et al. 2019).

2.3.2 Contextual Information

Contextual information is essential for semantic segmentation models to produce consistent predictions. The importance of contextual information has been showed in many works. (W. Liu et al. 2015) prove that simply adding global information can improve semantic segmentation results with FCN. DeepLabv2 (L.-C. Chen et al. 2016) proposes atrous spatial pyramid pooling (ASPP), where parallel atrous convolution layers with different dilation rates capture multi-scale information. PSPNet (H. Zhao et al. 2017) performs spatial pooling at several grid scales (see Figure 2.8) and demonstrates outstanding performance on several semantic segmentation datasets. Although these methods can effectively improve networks performance, they fuse different-level contexts for each pixel equally. (Fu et al. 2019) propose to select adaptive contexts for different pixels according the similarity between pixels and the global context.

The above-mentioned methods are all based on fixed-scale contextual information, while a more developed idea is to use pixel-sensitive contextual information. To this end, probabilistic graphical models, especially the CRFs, are again attracting attention as they are well suited for modeling pixel dependencies. As we described in Section 2.1, CRFs can be used as a post-processing method to

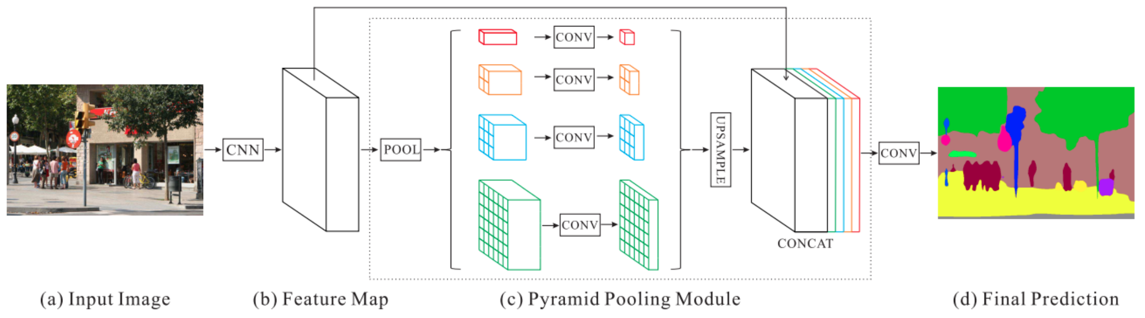


Figure 2.8. – Overview of PSPNet (H. Zhao et al. 2017). The pyramid pooling module (c) is used to capture contextual information at different scales. Different contextual information is then aggregated to improve the final prediction. Credits to (H. Zhao et al. 2017).

refine the segmentation results, as they constrain the pixel-level predictions to be consistent with the pixel dependencies. (L.-C. Chen et al. 2014) employ the fully connected CRF model by considering the segmentation CNNs as the unary classifier. However, since the segmentation CNN and the CRF are independent parts in this approach, it may not be the best way to incorporate the CRFs with the CNNs. To better integrate CRFs into DL-based method, (Zheng et al. 2015) show that the mean-field inference algorithm proposed by (Krähenbühl et al. 2011) can be reformulated as a Recurrent Neural Network (RNN): each step in the mean-field algorithm can be replaced by a CNN layer and the multiple mean-field iterations can be implemented by repeating those layers. Since these layers are differentiable, the parameters of CRFs (θ_u, θ_p in Equation 2.1) can also be jointly learned with the CNNs parameters. Therefore, (Krähenbühl et al. 2011) accommodate the CRF post-processing in a trainable module that can be integrated into the CNN in an end-to-end fashion. Since then, several methods have been proposed to either improve the potential function or speed up the inference process (G. Lin et al. 2016; Z. Liu et al. 2015; Jampani et al. 2016; Vemulapalli et al. 2016; Chandra et al. 2016). However, methods based on CRFs seem to be abandoned in recent research due to their slow nature. Notably, no significant studies published after 2018 have used a CRF- or graphical-model-based module to refine their segmentation results. Moreover, CRF-based works published in recent years show no significant leap in performance.

Recently, attention modules are applied in semantic segmentation to capture pixel-sensitive contextual information. The self-attention mechanism (Z. Lin et al. 2017; Vaswani et al. 2017) is originally proposed to better capture contextual information in Natural Language Processing (NLP) problems since contextual information is crucial for understand and translate a word in a sentence. It calculates the feature similarity between one position and all positions in a NLP frame, and defines the context of that position based on these feature similarity. Due to the success of attention modules in NLP, they are then adopted in segmentation by (Fu et al. 2018; Yuan et al. 2018; Hengshuang Zhao et al. 2018; Zilong Huang et al. 2019) and achieve state-of-the-art performance on several semantic segmentation benchmarks. Because the attention modules compute pairwise similarity, they are more costly in terms of calculation than fixed-scale contextual methods.

Somehow, the encoder-decoder-based strategies previously presented can also be considered as "context" methods because they exploit multi-scale local contexts from intermediate layers to bring more visual details. Although features of intermediate layers do represent contextual information at different scales, they contain less semantic information. When we talk about methods based on contextual information, we refer to methods that use contextual semantic information to reduce local ambiguity. Therefore, methods based on encoder-decoder architecture are more about recovering lost spatial information than adding contextual information.

2.4 Objective Function

In this section, we present several objective functions that are proposed to deal with class imbalance and evaluation metric problem of PPCE loss.

Class Imbalance A dataset is said to be imbalanced w.r.t. its classes if the number of samples varies a lot between the different classes. In the context of semantic segmentation, the number of samples of one class is the total numbers of pixels that belong to the this class. Therefore, the number of samples depends on the size of the objects and their occurrences. Taking a street scene image as an example, classes like road, building or sky usually contain much more pixels than class pedestrian or bicycle. Another special case is the "background" class where in many applications, we only annotate the objects that we are interested in, and all other classes are considered as background. In these applications, there are usually much more pixels belonging to the background class than to the foreground classes. Since PPCE loss is simply an average over all pixels, the loss value is largely affected by the class imbalance problem. Imagine a dataset where 90% of the pixels belong to the background, a model always predicting the background will reach a low value of PPCE loss. However, that is obviously not what we want. To address this issue, the main strategy consists in changing the contribution of each pixel in the final PPCE loss, either by introducing a pixel weight, or modifiying the pixel loss. In (Long et al. 2015), a weighted cross-entropy is proposed. Instead of averaging on all pixels, weighted cross-entropy loss ponder different classes differently in order to make all classes equally contribute to the total loss value. But this loss does not, in practice, enhance the accuracy (Long et al. 2015). Another idea is to decrease the importance of correctly predicted pixels and increase the importance of wrong predictions. Focal loss (T. Lin et al. 2020) is a popular example of this strategy.

$$Loss_{focal}(p_i) = -\alpha_i(1-p_i)^{\gamma}log(p_i) \tag{2.6}$$

As defined in Equation 2.6, the focal loss scales the cross-entropy loss by a factor $\alpha_i(1-p_i)^{\gamma}$. α_i is a weighting parameter that addresses the class imbalance problem and the parameter γ smoothly adjusts the rate at which easy examples are down-weighted (easy examples are those with a high probability in the ground truth class p_i). When $\gamma = 0$, focal loss is equivalent to a weighted cross-entropy loss, and increasing γ heavily concentrates the loss on hard examples, focusing nearly all attention away from easy negatives. This loss is simple and proved its effectiveness for imbalance problem (T. Lin et al. 2020).

Evaluation metric As we introduced in Section 2.2.3, the evaluation metric mostly used in tests is not cross entropy loss but the mIoU score. Therefore, models trained with PPCE are not directly optimized for mIoU, which might result in a fall in performance. The reason that we do not directly optimize the mIoU score during training is that mIoU is not differentiable due to the cardinal function in Equation 2.4. Some approaches employ approximated mIoU loss to train segmentation models. The goal is to minimize the gap between the actual mIoU value and its differentiable approximation. (Máttyus et al. 2017) use a soft version of mIoU and (Berman et al. 2018) provide a tractable surrogate based on the convex Lovasz extension of sub-modular functions. In the spirit of "learning everything with DL", (Nagendar et al. 2018) propose to automatically learn a surrogate loss function that approximates the mIoU with neural network. With these losses, the performances of several segmentation models is slightly improved compared to training with PPCE. However, experiments on different datasets with stronger segmentation models are need to better demonstrate their effectiveness.

Although mIoU is widely used in practice, many works have argued that mIoU is not a good measure of model performance in many cases. A major criticism of the mIoU score is that the mIoU does not accurately reflect whether the segmentation prediction follows the shape and structure of the ground truth map. In Figure 2.9, we show two predictions with the same IoU score, while Pred 2 retains much better the overall structure of the number "4". The main reason for this problem is that the IoU considers each pixel as independent. To this end, (Csurka et al. 2013) advocates for the use of per-image scores and proposes a boundary-aware metric to focus on contour pixels. (Margolin et al. 2014) generalize the $F_{\beta} - measure$ by assigning different weights to different errors, according to different location and neighborhood information. The Structural Similarity Measure (SSIM) index (Zhou Wang et al. 2004), which measures the similarity between two images with global statistics (such as mean, variance and covariance), is also adopted

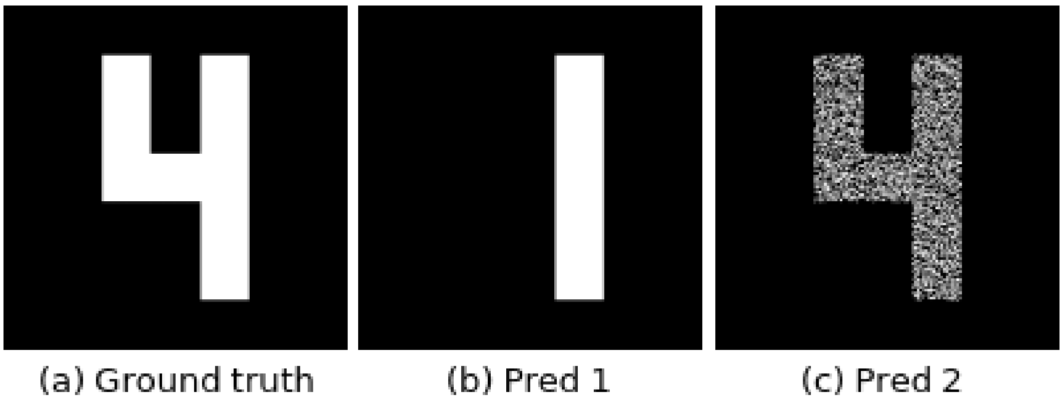


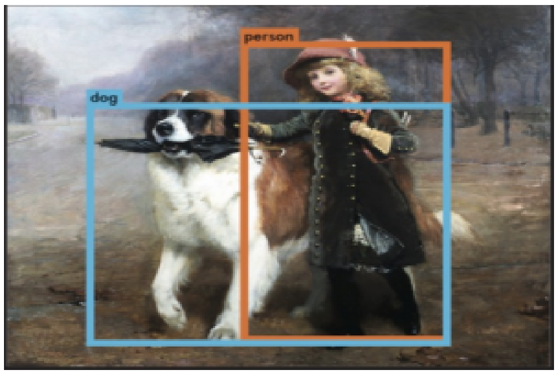
Figure 2.9. – (b) and (c) are two predicted segmentation masks with the same same IoU score. In (c), the false-negatives are sparsely spread within true-positive detection, thus offering a good sampling of the region. Credits to Margolin et al. 2014.

for segmentation by several works. (D.-P. Fan et al. 2017) extend the SSIM to capture both region-aware and object-aware structural similarities, and (D.-P. Fan et al. 2018) propose a luminance contrast based measure which simultaneously evaluates the pixel-level and image-level similarities. However, these metrics are not widely spread in the segmentation field, and the mIoU is still the main evaluation metric employed by most segmentation challenges.

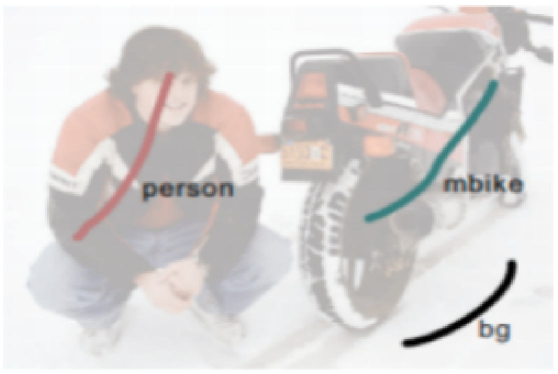
2.5 Weakly-Supervised Segmentation

WSL methods for semantic segmentation have attracted a lot of interest due to the lack of fully annotated data for segmentation. WSL approaches encompass a variety of training annotations less informative than the pixel level, such as (in order of decreasing informativeness): bounding box (Dai et al. 2015; Papandreou et al. 2015), scribble (D. Lin et al. 2016), point (A. L. Bearman et al. 2016), and image label (Papandreou et al. 2015; Xu et al. 2015) (see Figure 2.10). Image-level labels are the cheapest to provide and can be obtained easily from many resources, thanks to datasets for image recognition. However, due to the complete absence of spatial information, image-level labels are also the most challenging to use. In what follows, we review the literature in weakly-supervised semantic segmentation from image-level annotations.

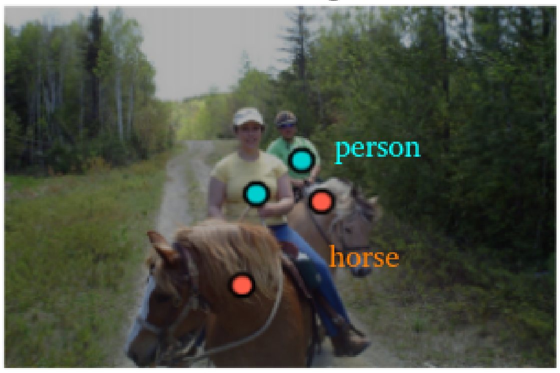
Fully-supervised methods can be considered as the "upper-bound" in performance for WSL because they are trained with theoretically the most informative supervisory data possible. Although, for the moment, the best fully supervised



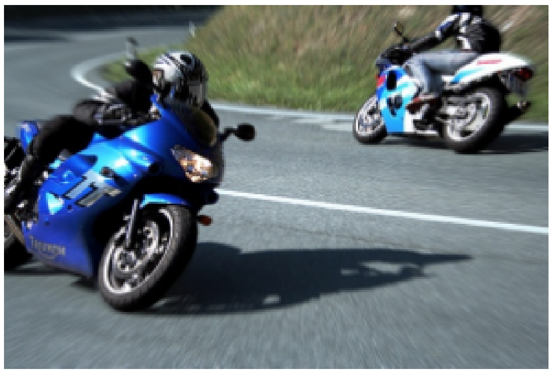
(a) Bounding box



(b) Scribble



(c) Point



(d) Image level with (person and motorbike)

Figure 2.10. – Different kind of weak labels. In order of decreasing informativeness: (a) Bounding box, (b) Scribble, (c) Point and (d) Image-level label.

segmentation model far out-performs the best WSL method, the quality of WSL methods is impressive, especially considering that they learn to segment without any location-specific supervision. To solve this problem, different WSL methods have been proposed and can be broadly categorized into two approaches. The first type of approach is based on Multi Instance Learning (MIL). More precisely, in MIL, the training set consists of labeled "bags", each of which is a collection of unlabeled instances. A bag is positively labeled if at least one instance in it is positive, and is negatively labeled if all instances in it are negative. The goal of the MIL is to predict the labels of new, unseen bags. In the context of WSL for segmentation, a bag is an image with image-level labels and each pixel is an unlabeled instance. That is, for a given image, the image-level labels tell us that at least one pixel of that class is present. In practice, this usually means training a CNN with an image-level loss and inferring the pixels responsible for each predicted class. MIL-FCN (Pathak et al. 2014) trains a FCN with a global max-pooling loss which selects the most informative region for the MIL prediction. WILDCAT

(Durand et al. 2017) proposes a weighted spatial average operation, which also takes negative evidences into account. To better locate objects, (Papandreou et al. 2015) incorporate an additional prior in the MIL framework in the form of an adaptive foreground/background bias. The notion of objectness priors is further developed by (Wei et al. 2016; A. L. Bearman et al. 2016), which provide each pixel with a probability of being an object.

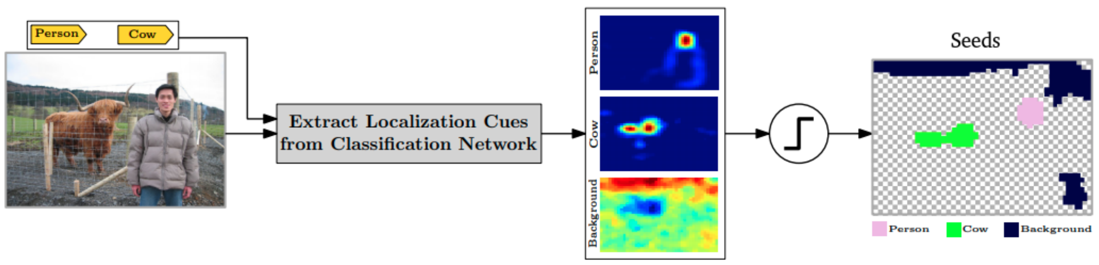


Figure 2.11. – Illustration of the seeds generation procedure. Localization cues are extracted from classification networks, and the resulting mask (seeds) is considered as pseudo ground truth for training traditional fully supervised segmentation models. Credits to (Kolesnikov et al. 2016).

Another important type of approach is to use the inferred pixel-level activations as pseudo ground-truth labels for training "fully" supervised segmentation models. In practice, it usually means extracting localization cues from a classification CNN to define a pseudo ground truth mask (or seeds), as showed in Figure 2.11, and training traditional segmentation models such as FCN with the resulting seeds. SEC (Kolesnikov et al. 2016) is the prototypical method of this approach. It applies CAM (Zhou et al. 2016) to generate seeds to train a Deeplab model (L.-C. Chen et al. 2014) with a composite of three loss functions, *i.e.* seeding, expansion and constrain-to-boundary. The main issue of this approach is the quality of the generated seeds. Since the localization cues are usually inaccurate, only the most discriminative regions are considered as seeds. Models trained with seeds tend to produce good segmentation results for only a small part rather than entire objects. Different methods have been proposed to solve this problem. AE-PSL (Wei et al. 2017) gradually mines discriminative object regions to obtain extended seeds. In each iteration, the previously found parts are erased from the image, and the new localization cues is extracted from the rest of the image. Instead of erasing images, which requires re-training the CNN, FickleNet (Lee et al. 2019) train a CNN at the image-level with centre-fixed spatial dropout in the last convolutional layers and aggregates different localization maps into a single map. Another way to obtain larger seeds is to propagate seeds to adjacent regions with similar visual appearance. (Zilong Huang et al. 2018) propose to grow the seeds to unlabeled pixels based on pixel homogeneity while (Ahn et al. 2018) use random walk to propagate the seeds to nearby and semantically identical areas. It is worth noting that, the seed-based approaches dominate the WSL semantic segmentation task on PASCAL VOC2012 dataset.

2.6 Positioning and Framework

In the previous part of this chapter, we reviewed the main methods of semantic segmentation, and highlighted the state-of-the-art methods that are based on DL. After a detailed analysis of the seminal approach FCN, we summarized the three major challenges of semantic segmentation algorithms based on DL and presented the corresponding improvements. We now go over some interesting questions that we address in this thesis to produce finer segmentation results and to alleviate the demand for annotated data.

The dependency between pixels plays an important role in semantic segmentation, as it defines which pixels in an image belong to the same object. As we introduced in Section 2.1, traditional segmentation methods, such as region-based and edge-based methods, employ human designed features to determine if they belong to the same segment. These features are mainly calculated using simple pixel variations and cannot capture the dependency at a semantic level. In the era of DL, thanks to the advances in hardware and algorithms, CNN-based models are able to learn more complicated dependencies from a large amount of annotated data. In order to better model and capture the pixel dependencies, most of the work has focused on improving the network architecture, such as encoder-decoder architecture, pyramid spatial pooling and attention mechanism that we presented in Section 2.3. However, the PPCE loss function in DL-based methods (see Section 2.4) does not guarantee that the modules added to the network will learn the pixels dependencies. Since the PPCE loss evaluates the pixels independently, it does not impose any constraint on the consistency of dependent pixel predictions. This means that the prediction of a pixel has no effect on the relevant pixels, and the predictions of highly dependent pixels need not be dependent as well. We are particularly interested in this important issue and attempt to address it by defining a new loss function, which called SEMEDA loss, that requires predicted masks to have the same spatial dependency as the ground truth. More specifically, when a pixel is inside a large object, its prediction should be uniform with its neighbors, and if the pixel is on the boundary of the object, its prediction should also contain information about the object next to it and retain the boundary information along with the neighboring pixels. In this manner, the mask produced by segmentation models can better conform to the boundary shape of objects and avoid holes inside the object.

The second concern is the lack of annotated data. As we described in Section 2.2.4, DL-based methods typically require a large amount of annotated data to learn from, while the data annotation process for segmentation is extremely expensive and time-consuming. Furthermore, even in a high-quality annotated data set, the data may still be noisy and imbalanced. Therefore, methods that can learn from partial or non-annotated data are well worth investigating. For example, WSL methods can learn from partially annotated data, making it possible to benefit from a large amount of weakly annotated data. Semi-supervised methods can learn from unlabeled data, when used in conjunction with a small amount of labeled data. And more recently, self-supervised learning has gained a lot of attention because it requires only unlabeled data to formulate a pretext learning task. In this thesis, we focus on how to use image-level annotated data for semantic segmentation because they are easy to obtain and contain semantic information. More specifically, we attempt to extract localization cues about semantic objects based on the mathematical properties of neural networks. The extracted localization cues can then be combined with the traditional WSL approach to enable the segmentation model to learn from the image-level labeled data.

Dataset Throughout this thesis, to evaluate the efficiency of proposed methods, almost all experiments are conducted on the PASCAL VOC 2012 (Everingham et al. 2015) and the Cityscapes (Cordts et al. 2016) datasets, which are the most used open datasets for evaluating semantic segmentation models. As we introduced in Section 2.2.3, PASCAL VOC is a general purpose dataset that includes all types of indoor and outdoor images, and Cityscapes is recently created large-scale dataset focused on urban street scene understanding. Since the quality of the datasets has a significant impact on the performance of the segmentation algorithm, we also conduct experiments on two additional datasets, namely HELEN (Le et al. 2012) for face parsing and ISIC2018 (Codella et al. 2019) for lesion segmentation. Figure 2.12 illustrates how images differ from different datasets and a consistency of performance across these datasets is an important demonstration of the effectiveness of the method.

Segmentation Network The segmentation backbone networks used in this thesis are different versions of the Deeplab model, mainly Deeplab-v2 (L.-C. Chen et al. 2016) and Deeplab-v3+ (Liang-Chieh Chen et al. 2018), that are built on powerful CNNs such as VGG and ResNet. Compared to the baseline model FCN, it has been improved in several ways. Deeplab-v2 uses atrous convolution to increase spatial information and proposes atrous spatial pyramid pooling to incorporate contextual information. Deeplab-v3+ further employs encoder-decoder structure to better recover spatial information, and adapts the model for more efficient backbone Xception (Chollet 2017). Deeplabs are one of the most efficient semantic

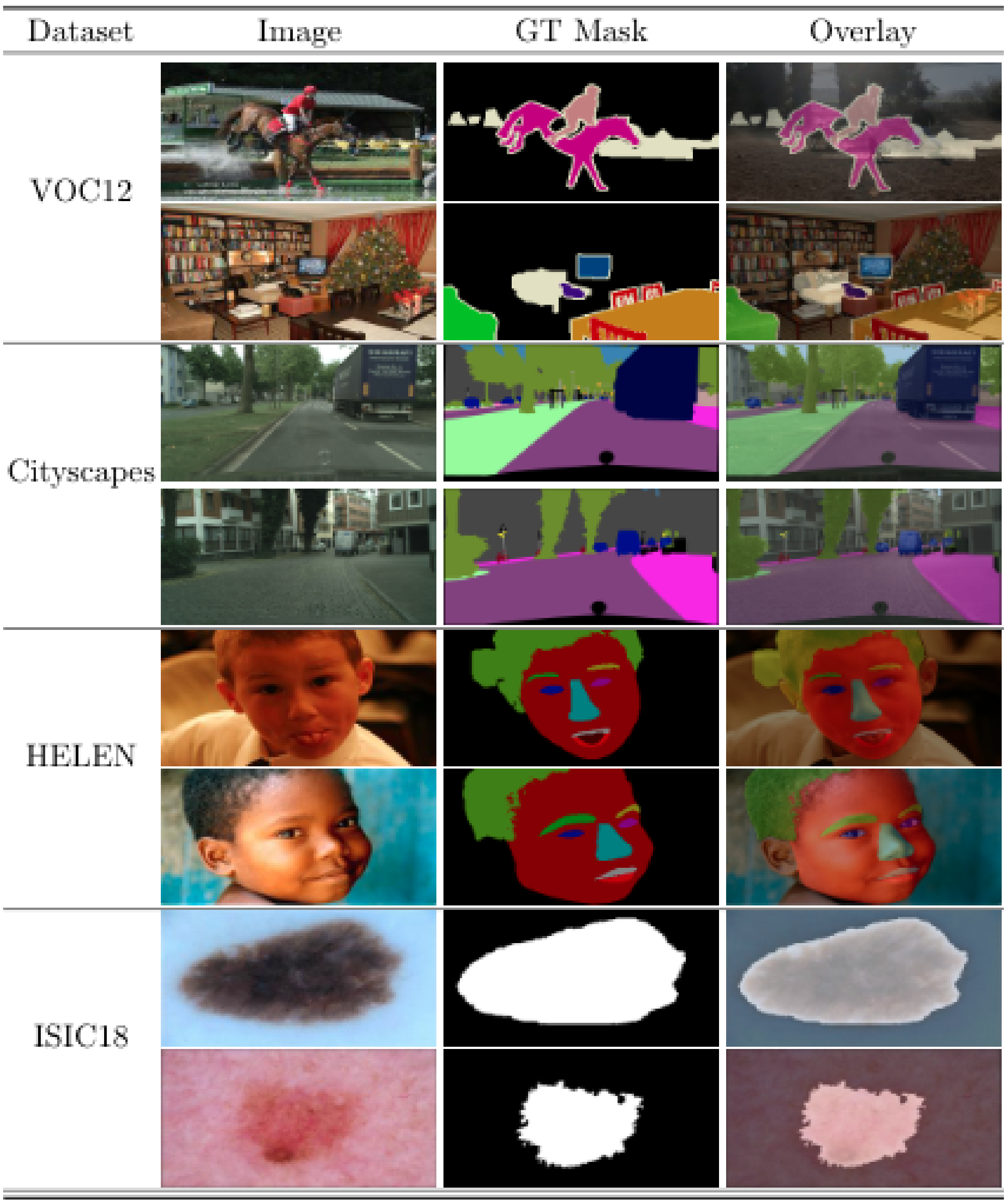


Figure 2.12. – Illustration of samples from different segmentation datasets used in course of this thesis. VOC12 and Cityscapes are the most used large-scale dataset for evaluating segmentation models. HELEN is a dataset for face parsing and ISIC18 is a dataset for lesion segmentation.

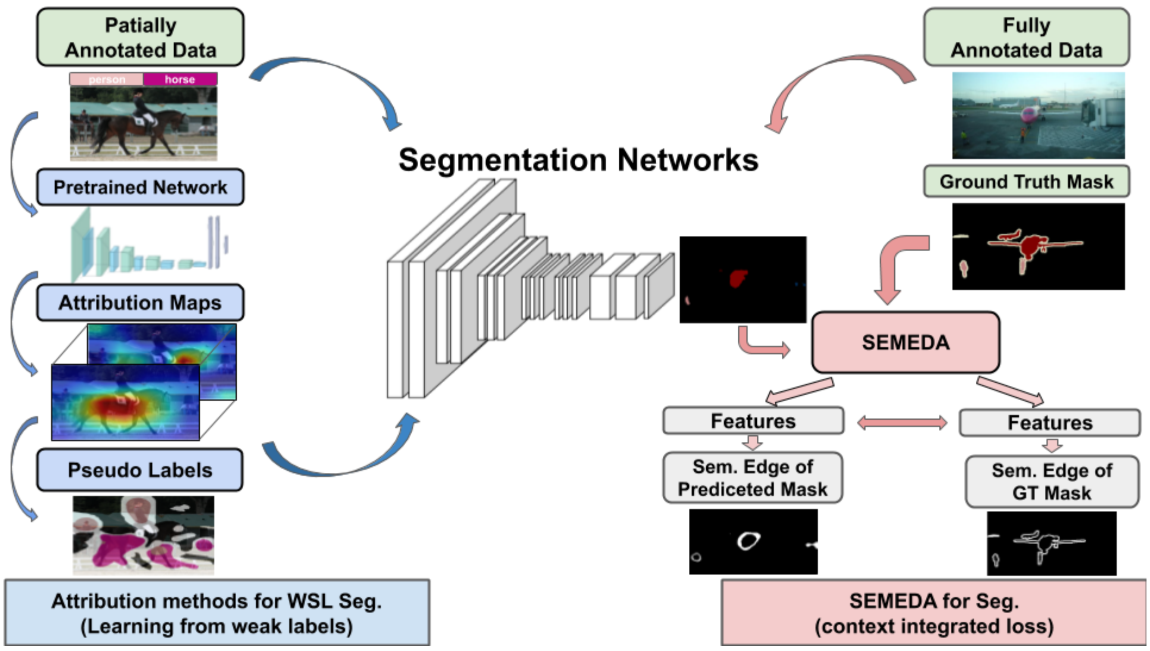


Figure 2.13. – Positioning of this thesis. One part of thesis addresses the semantic segmentation problem by a contextual information integrated loss function (red) and another part focus on attribution methods that allow segmentation models to learn from weakly annotated data (blue).

segmentation models, and they achieve the state-of-the-art performance on multi open datasets, including PASCAL VOC 2012 and Cityscapes.

The positioning of this thesis can thus be summarized in Figure 2.13. On the one hand, we address the semantic segmentation problem by a contextual information integrated loss function SEMEDA (red). The proposed SEMEDA loss constraints the prediction of a pixel to be consistent with its neighbors, whether it is inside or at the boundary of the object, and thus helps the networks to produce more shape-consistent results. On the other hand, we attempt to alleviate the demand of fully annotated data for semantic segmentation with WSL methods (blue). We develop a new attribution method based on the mathematical properties of CNNs, which can accurately identify the most relevant pixels for CNN predictions. This method can then be integrated in WSL methods and allow segmentation models to learn only from image-level annotated data.

In what follows, we present the two works that we carried out during the thesis. For these two works, we use the datasets and the backbone segmentation networks described above.

Chapter 3

SEMEDA: ENHANCING SEGMENTATION PRECISION WITH SEMANTIC EDGE AWARE LOSS

Contents

Chapter abstract

As mentioned earlier, loss functions play an important role in the learning process of machine learning problems, and the design of an appropriate loss function can be beneficial. Nowadays, PPCE is the most commonly used loss for semantic segmentation tasks. However, it suffers from a number of drawbacks. In this chapter, we focus on the loss function for semantic segmentation. After carefully analyzing the drawbacks of PPCE for segmentation, we present a SEMantic EDge-Aware strategy (SEMEDA) that solves these issues. Inspired by

perceptual losses, we propose to match the 'probability texture' of the predicted segmentation mask and ground truth through a proxy network trained for semantic edge detection on the ground truth masks. Through thorough experimental validation on several datasets, we show that SEMEDA steadily improves the segmentation accuracy with negligible additional computational overhead and can be added to any popular segmentation networks in an end-to-end training framework.

The work in this chapter has led to the submission of a journal paper:

- Yifu Chen, Arnaud Dapogny, and Matthieu Cord (2020). "SEMEDA: Enhancing Segmentation Precision with Semantic Edge Aware Loss". In: *Pattern Recognition (under review, major revision).*

3.1 Introduction

In this chapter, we mainly address the problem **D2** introduced in Section 1.2: the loss function for semantic segmentation. As we presented in Section 2.2, current state-of-the-art methods mainly rely on Fully Convolutional Network (FCN) architectures (Long et al. 2015) that are trained by optimizing a per-pixel loss between predictions and ground truth labels. Fully convolutional neural networks cleverly inherit the idea of classification Convolutional Neural Network (CNN)s, and classify each pixel by using a patch centered on it. In this manner, popular classification networks can be adapted into a fully convolutional network and their learned representations can be transferred to segmentation tasks by training them with the Per Pixel Cross Entropy (PPCE) loss. This loss is a natural choice as a spatial extension of the cross-entropy loss, which is the standard for classification. However, the PPCE loss has some drawbacks when applied to semantic segmentation.

First, PPCE loss is just an average over each pixel's accuracy and can not capture structural differences, such as the shape of objects, between output and ground-truth segmentation masks (**problem a**). Each pixel is treated equally and independently in PPCE loss. For example, a good segmentation mask should preserve the semantic boundary of each object. However, the PPCE loss is not designed for that. As a result, models trained with PPCE loss usually struggle to output geometrically correct predictions as presented in Figure 3.1. A similar issue has been found in image transformation problems, where an input image is transformed into an output image. A per-pixel loss such loss L1 and loss L2 is usually used in these problems (C.Dong et al. 2016; Z.Cheng et al. 2015; R.Zhang et al. 2016). As criticized by (Johnson et al. 2016), per-pixel losses can not measure perceptual differences between predicted and ground-truth images. An interesting solution provided by (Johnson et al. 2016) is called "perceptual loss", where generated and ground truth images are matched in the embedding spaces of the

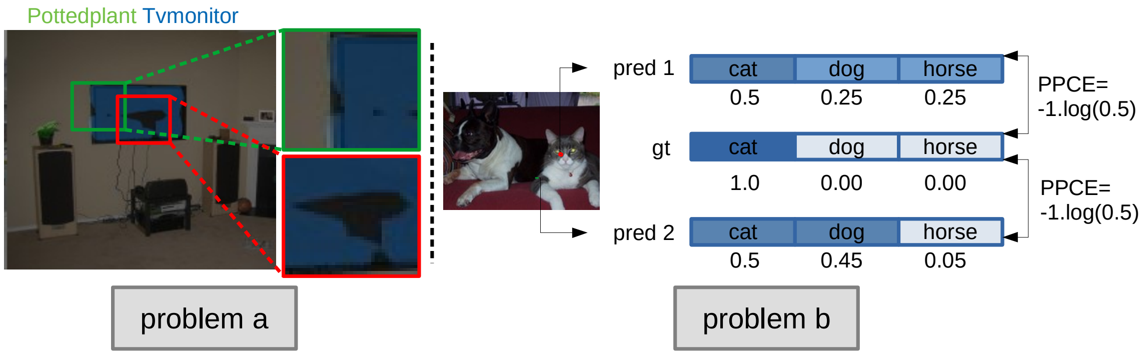


Figure 3.1. – Illustration of **problem a-b** of traditional PPCE loss for semantic segmentation. **problem a**: the segmentation maps predicted by a network trained with PPCE exhibit a lack of structure (green box: loose semantic boundaries, red box: holes in the predicted semantic regions). **problem b**: Two different predictions (pred 1 and pred 2) that are equivalent for cross entropy loss. However, we may prefer pred 2 for boundary pixels (green) and pred 1 for center pixels (red).

layers of a pre-trained classification network (usually VGG network (Simonyan et al. 2015)). Traditionally, more emphasis is put on the weights corresponding to the first layers: to a certain extent, perceptual losses lead to match higher-order moments (e.g. gradients) extracted by these layers, thus taking into account the neighborhood of each pixel. However, the perceptual loss cannot directly be used for semantic segmentation. On one hand, the perceptual loss is applied to RGB images while segmentation masks have C channels where C is the total number of classes. On the other hand, the distribution of segmentation masks is wildly different from natural images.

Secondly, PPCE only depends on the predicted value of the ground truth class but not the entire distribution over all classes (**problem b**). For instance, consider a case where there are three classes (cat, dog and horse) and a pixel that belongs to the region of a cat, as illustrated in Figure 3.1. Predictions (0.5, 0.25, 0.25) and (0.5, 0.45, 0.05) will have the same loss value since the cross entropy loss only depends on the value 0.5 (class cat). However, these two predictions are not equivalent. If the pixel is located in the center of the cat region, we may prefer the first prediction since it is more robust. If the pixel is located at the border where a cat and a dog overlap, the second prediction may be more reasonable.

To address these problems, we propose in this chapter a novel SEMantic EDge-Aware (SEMEDA) loss for training semantic segmentation networks. Instead of matching the predicted mask and the ground truth mask pixel by pixel, we propose to match them region by region. In other words, we propose a loss function, which forces the network to produce more structural outputs. In this way, we intend to offer a solution to the problem **D2** and also somehow to the problem **D1**. The rest of this chapter is organized as follows:

- We first review the recent improvements on the PPCE loss function for segmentation in Section 3.2. These loss functions can be generally grouped into two types. The first is based on explicit formulas designed by experts, and the second uses a network to learn a loss function from data. In order to provide a broader understanding, we present the idea behind these loss functions as well as their disadvantages.

- Then, we introduce the proposed SEMEDA loss as well as the training strategy for semantic segmentation networks in Section 3.3. Through detailed reasoning, we show that our approach is a viable solution to problems **a-b** of the PPCE loss.

- We then validate our method through thorough experiments on two commonly used segmentation datasets in Section 3.4. We show that with negligible additional computation and memory usage, our method consistently improves the segmentation results on both datasets. Our method also provides comparable results with other leading segmentation models on these datasets.

- To better demonstrate the effectiveness of our approach, we present two side applications of our method on noisy datasets in Section 3.5. Though the ground truth annotations are less accurate, our method still enhances the segmentation results compared to the baseline model, which further shows the robustness of the proposed method.

3.2 PPCE Loss Extension

As we presented in Chapter 2, the PPCE loss function is firstly used in FCN (Long et al. 2015) mainly because of its simplicity. Then, it's been widely inherited in other segmentation works. However, the choice of PPCE loss function has recently been criticized in many works. Specifically, we have presented two drawbacks (**Class Imbalance** and **Evaluation Metric**) and their related improvements in Section 2.4. Here, we focus on another important issue of PPCE, which is called **Structural Similarity**. Structural similarity includes similarities in shape, semantic edges, and geometries between the prediction mask and the ground truth mask. As many works (Csurka et al. 2013; Luc et al. 2016; S. Chen et al. 2018) argued, PPCE cannot properly measure structural similarities between predicted and ground truth segmentation masks. The problem **a-b** described in Section 3.1 are two main reasons for this issue. The PPCE loss considers the segmentation task as a pixel-wise classification problem where pixels are independent samples.

However, there are strong dependencies between pixels in an image, and these dependencies contain important information about the structure of objects (problem **a**). Moreover, PPCE only optimizes the prediction of the ground truth class but not the entire distribution across all classes (problem **b**), which makes it impossible to capture the finer relationship between the predictions of the different pixels. Due to these limitations, optimizing PPCE loss during training usually leads to errors on fine-scale structures, such as small object instances, instances with multiple connected components, and thin connections.

Although there have been many improvements in model design for encoding spatial and contextual information, such as Conditional Random Fields (CRF) (L.-C. Chen et al. 2014; Zheng et al. 2015), attention mechanisms (Z. Lin et al. 2017; Vaswani et al. 2017) and contour cues (Bertasius et al. 2016; L. Chen et al. 2016), there are only a few loss functions that ensure these pixel dependencies are really learned by the network. These losses can be roughly divided into two types: **explicit loss functions** and **network-based loss functions**. The **explicit loss functions** are usually based on specific structural features of the segmentation mask, such as edges, defined by certain explicit formulas. The most direct way to leverage edge information in loss functions is to weigh pixels differently in the traditional PPCE loss where edge pixels are emphasized more than other pixels, as described in Equation 3.1:

$$L(\hat{S}, S^*) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{i=j}^{W} w_{i,j} \times L_{CrossEntropy}(\hat{S}_{i,j}, S^*_{i,j}) \tag{3.1}$$

where $w_{i,j} = f(d(p_{i,j}))$ is a coefficient that shall be inversely proportional to the distance $d(p_{i,j})$ between pixel $p_{i,j}$ and its nearest edge. This coefficient can be defined differently: (Caliva et al. 2019) define $w_{i,j} = 1 + \frac{1}{d(p_{i,j})}$ while (Zhen et al. 2019) propose to group pixels according to the distance $d(p_{i,j})$ and each group of pixels has a different weight. The coefficient proposed by (Ronneberger et al. 2015) takes into account both the shortest and the second shortest distance:

$$w_{i,j} = w_0 * exp(-\frac{(d_1(p_{i,j}) + d_2(p_{i,j}))^2}{2\sigma^2})$$

Essentially, these methods assume that the closer the pixel lies from the boundary, the more important it is. However, this assumption is not always true and may provide discontinuities in the center of large objects since these pixels will have little impact on the total loss under this assumption.

Unlike previously introduced explicit loss functions, (Luc et al. 2016) adopts the Generative Adversarial Network (GAN) framework and proposes to use an adversarial network to learn a loss function for semantic segmentation. The segmentation network is considered as a mask generator, and the discriminator learns to determine whether a mask is a ground truth or whether it is generated

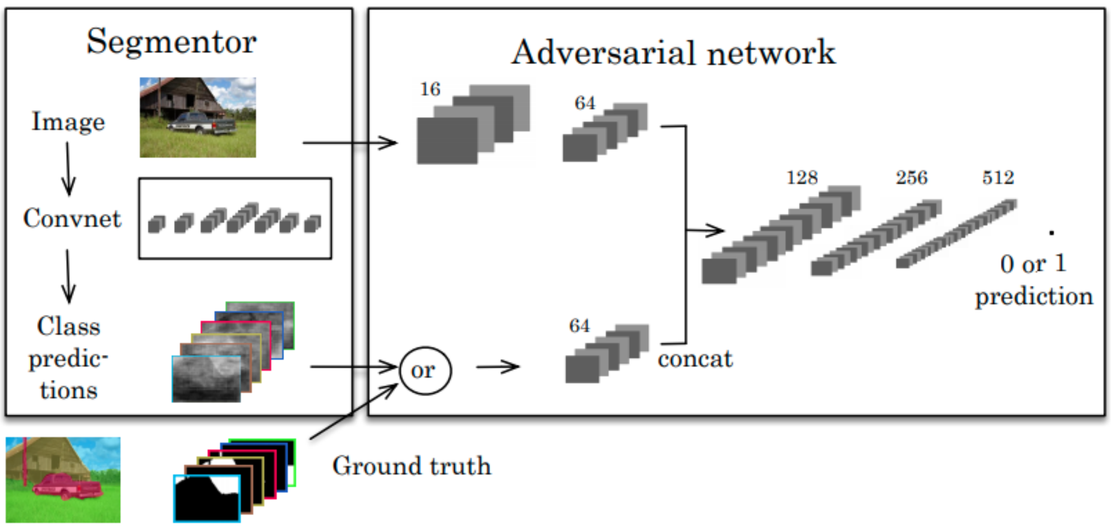


Figure 3.2. – Overview of the model proposed in (Luc et al. 2016). The adversarial network takes an RGB image and a mask as input and predicts whether the mask is the ground truth or is generated by the Segmentor. The adversarial network is used to define an adversarial loss which encourages the segmentation model to produce masks that cannot be distinguished from ground-truth ones (Luc et al. 2016).

(see Figure 3.2). They believe that the discriminator can detect and correct high-order inconsistencies between the ground truth map and the one produced by the generator. However, the shape of the semantic segments is too vague. A semantic segment does not separate two objects of the same class if they are overlapping. In addition to partial occlusion, object scaling, camera viewpoint change and small size of segmentation dataset, the possible shape of a semantic segment could be extremely complicated and difficult to model with generative adversarial networks. With the proposed adversarial loss, (Luc et al. 2016) get only about 0.25% improvement in mean Intersection over Union (mIoU) score on the PASCAL VOC 2012 dataset (Everingham et al. 2015).

3.3 SEMEDA Model

In this chapter, we propose to build upon the intuitions and interpretations of SEMEDA to address **problem a-b** of the PPCE loss introduced in Section 3.1. Figure 3.3 illustrates the proposed SEMEDA method to train a deep network for semantic segmentation problems. Classically, a semantic segmentation network (displayed in red in Figure 3.3) is trained to output a segmentation mask from an image. In SEMEDA, we use the embedding space of another network that we refer to as the SEMEDA network (blue in Figure 3.3) to match the predicted

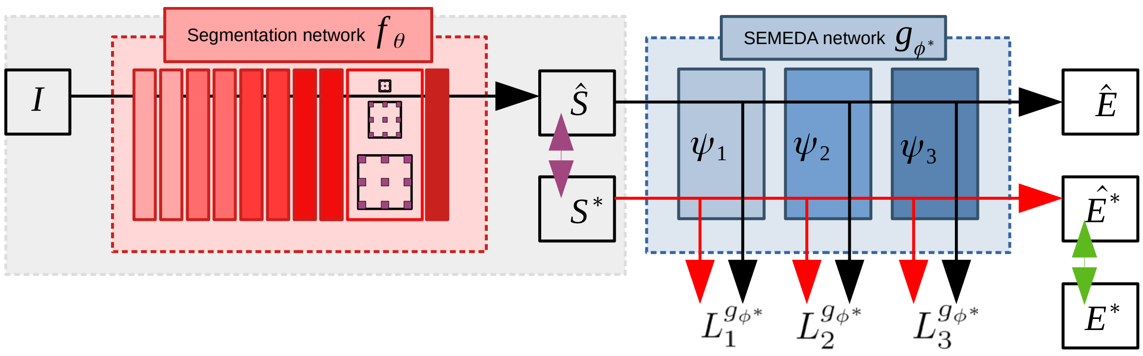


Figure 3.3. – Overview of our SEMEDA framework for semantic segmentation. A backbone segmentation network f_θ outputs a predicted mask $\hat{S}$. Both this predicted mask, as well as the corresponding ground truth mask are provided independently to a SEMEDA network g_ϕ that is trained for semantic edge detection (green arrow). We complete the traditional PPCE loss (purple arrow) by adding a novel SEMEDA loss term, which consists in matching the predicted segmentation mask with its corresponding ground truth within the embeddings of the SEMEDA network, each layer l downwards red and black arrows giving rise to a contribution $L_l^{g_{\phi^*}}$.

segmentation mask with the corresponding ground truth. In this section, we first introduce how traditional approaches for semantic segmentation can be formulated, and what the common pitfalls of such approaches are. Then, we present our approach to solve the problem which consists of matching a predicted segmentation mask with its corresponding ground truth in the embedding space of a pre-trained SEMEDA network.

3.3.1 The Pitfalls of Naive Segmentation Approaches

A standard approach for training semantic segmentation networks is illustrated in Figure 3.3 (gray box). Let's consider a semantic segmentation network f_θ parameterized by weights θ, mapping an RGB image $I \in \mathbb{R}^{H\times W\times 3}$ from a training dataset $\{I, S^*\}$ into a segmentation mask $\hat{S}$:

$$\hat{S} = f_\theta(I) \in [0,1]^{H\times W\times C} \tag{3.2}$$

where H (resp. W) is the height (resp. width) of the input image and C is the total number of semantic classes (background included). Such a network is usually trained upon optimization of the PPCE loss $\mathcal{L}^{PPCE}$ (purple arrow in Figure 3.3). PPCE measures the difference between the predicted label mask $\hat{S}$ and the ground truth mask S^*:

$$\mathcal{L}^{PPCE}(\hat{S}, S^*) = -\frac{1}{H * W} \sum_{i=1}^{H} \sum_{j=1}^{W} \sum_{c=1}^{C} S_{i,j}^{c*} \log(\hat{S}_{i,j}^{c}) \tag{3.3}$$

We argue that this loss function exhibits several drawbacks for semantic segmentation tasks. First, PPCE treats all pixels independently and does not take the local structure into account. As a result, the predicted segmentation masks usually contain holes in the structure of the segmented objects or inconsistencies at their boundaries (**problem a**). Secondly, for each pixel, since $\{S_{i,j}^{c*}\}_{c \in C}$ are usually one-hot encoded, PPCE only depends on the probability of the ground truth class. Therefore, several predictions are given the same penalty while the entropy of the predicted mask $\{\hat{S}_{i,j}^{c}\}_{c \in C}$ may vary a lot for that pixel (**problem b**). In the next section, we detail our proposed solution to these two problems.

3.3.2 Structure Learning Through Edge Embeddings

In the context of style transfer and image synthesis, the authors of (Johnson et al. 2016) obtained impressive results by defining high-level perceptual losses that involve a fixed pre-trained network, such as an ImageNet pre-trained VGG-19 (Simonyan et al. 2015) network. The idea of this method is to measure the semantic difference between two images as the difference between their feature representations as computed by the fixed network. However, this idea cannot be straightforwardly translated to semantic segmentation, as natural images and segmentation masks have different numbers of channels and belong to wildly different distributions. Now, suppose that we have access to another pre-trained network g_{ϕ^*} (that we refer to as the SEMEDA network) that maps any segmentation mask $S \in [0,1]^{H \times W \times C}$ into a binary semantic edge map $\hat{E}$:

$$\hat{E} = g_{\phi^*}(S) \in [0,1]^{H \times W \times 2} \tag{3.4}$$

In what follows, we respectively note $\hat{\psi}_1$,$\hat{\psi}_2$,...,$\hat{\psi}_L$ the embeddings of the L layers of $g_{\phi^*}(\hat{S})$, and ψ_1^*,ψ_2^*,...,ψ_L^* the embeddings of the L layers of $g_{\phi^*}(S^*)$. Similarly to (Johnson et al. 2016), we can thus match $\hat{S}$ and S^* within the embeddings of g_{ϕ^*} (black and red arrows in Figure 3.3). We thus define our SEMantic EDge-Aware (SEMEDA) loss as follows:

$$\mathcal{L}_{\text{SEMEDA}}^{g_{\phi^*}}(\hat{S}, S^*) = \sum_{l=1}^{L} \lambda_l \mathcal{L}_l^{g_{\phi^*}}(\hat{S}, S^*) \tag{3.5}$$

where for all layers l, the contribution of this layer of the SEMEDA network to the total loss is:

$$\mathcal{L}_l^{g_{\phi^*}}(\hat{S}, S^*) = ||\hat{\psi}_l - \psi_l^*||_1 \tag{3.6}$$

where λ_l is a hyperparameter representing the importance of this layer in the total loss. The final loss function for the segmentation network is a combination of the PPCE loss on the segmentation masks and the proposed SEMEDA loss:

$$\mathcal{L}_{tot}^{g_{\phi^*}}(\hat{S}, S^*) = \mathcal{L}^{PPCE}(\hat{S}, S^*) + \mathcal{L}_{\text{SEMEDA}}^{g_{\phi^*}}(\hat{S}, S^*) \tag{3.7}$$

Since g_{ϕ^*} is trained to detect inter-class boundaries, minimizing $\mathcal{L}_{\text{SEMEDA}}$ naturally penalizes high-entropy configurations where the contributions of several classes are important (addressing **problem b**). Furthermore, it also heavily penalizes discontinuities in the structure of objects (addressing **problem a**). Thus, semantic edge detection from segmentation masks is a particularly interesting candidate objective for g_{ϕ^*}. In what follows, we describe how it can be formalized.

3.3.3 Learning to Detect Semantic Edges

In this section, we detail the proposed SEMEDA strategy. Given an image I and its corresponding ground truth segmentation mask S^*, we generate a binary ground truth semantic edge map E^* by setting all pixels that do not have 8 identically-labeled neighbor pixels as 1, and other pixels as 0. These ground truth semantic edge maps are calculated beforehand and no further computation is needed afterwards.

We train the SEMEDA network g_ϕ to minimize PPCE loss between semantic edge maps $\hat{E}^* = g_\phi(S^*)$ predicted upon the ground truth segmentation masks S^*, and ground truth masks generated edge maps E^*, as indicated by the green arrow in Figure 3.3:

$$\mathcal{L}^{PPCE}(\hat{E}^*, E^*) = -\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{c=1}^{C} \hat{E}_{i,j}^{c*} \log(\hat{E}_{i,j}^{c}) \tag{3.8}$$

This network is depicted in Figure 3.3 (in blue). Once the SEMEDA network is trained with parameters ϕ^*, we train the segmentation network f_θ by minimizing loss $\mathcal{L}_{\text{SEMEDA}}^{g_{\phi^*}}$. The steps for training a segmentation network with SEMEDA are summarized in Algorithm 3.1.

3.4 Experiments

In this chapter, we validate the proposed SEMEDA strategy through experiments. First, we perform an ablation study and discuss the hyperparameter settings. Secondly, we validate our approach on two of the most popular semantic segmentation datasets, both quantitatively and qualitatively.

Algorithm 3.1 Train a segmentation network with SEMEDA

```
Input:
  $I$     RGB Images
  $S^*$   Ground truth segmentation masks
Output:
  $\theta^*$   Parameters of the segmentation net

  // pre-training the SEMEDA network $g_\phi$
  for all batches $\mathcal{B}$ do
    for all masks $S^*_k$,$k = 1, ..., K$ in $\mathcal{B}$ do
      Generate ground truth edge map $E^*_k$ from $S^*_k$ by
      examining neighboring pixels labels
      $\hat{E}^*_k = g_\phi(S^*_k)$
      $\phi \leftarrow \phi - \frac{1}{K}\frac{\partial}{\partial\phi}\mathcal{L}^{PPCE}(\hat{E}^*_k, E^*_k)$
    end for
  end for
  $\phi^* \leftarrow \phi$
  // training the segmentation network $f_\theta$
  for all batches $\mathcal{B}$ do
    for all labeled images $I_k$ in $\mathcal{B}$ do
      $\hat{S}_k = f_\theta(I_k)$
      $\theta \leftarrow \theta - \frac{1}{K}\frac{\partial}{\partial\theta}\mathcal{L}^{g_{\phi^*}}_{tot}(\hat{S}_k, S^*_k)$
    end for
  end for
  $\theta^* \leftarrow \theta$ =0
```

3.4.1 Implementation Details

Segmentation network: SEMEDA is agnostic to the architecture of the segmentation network f_θ, which can be any popular architecture. In what follows, we experiment with Deeplab-v2 (L.-C. Chen et al. 2016) as well as the recent Deeplab-v3+ (Liang-Chieh Chen et al. 2018). Both architectures are composed of a backbone feature extraction network and differ by the refinement portion of the network. For that matter, we also experiment with either an ImageNet pre-trained ResNet-101 (K. He et al. 2016) and a Xception-71 (Chollet 2017) backbone networks.

SEMEDA network: Because the task of detecting semantic edges from the segmentation mask is rather straightforward, we use a simple CNN composed of $L = 3$ layers with $16 \rightarrow 32 \rightarrow 2$ channels and Rectified Linear Unit (ReLU) activations (except for the last layer, which has a softmax activation) as g_ϕ. This network contains about 10k parameters which is far less than the number of parameters in backbone networks (e.g, about 44 millions for ResNet101 (K. He et al. 2016)).

In order to keep the runtime and memory footprint reasonable, we train our models by feeding the SEMEDA networks with 321×321 random crops for both datasets without multi-scale inputs. We augment the data with random scaling and random mirroring. Since we use mini-batches of 6 images, we fix parameters in batch norm layers and we set the initial learning rate to $5 \cdot 10^{-4}$. As is classically done in the literature, we report the mIoU metric over all the classes as our evaluation metric.

3.4.2 Experimental Setup

We conduct our experiments on two of the most used open segmentation datasets: **PASCAL VOC 2012** (Everingham et al. 2015) and **Cityscapes** (Cordts et al. 2016). As introduced in Section 2.2.3, PASCAL VOC 2012 is the most popular general purpose dataset for segmentation, which contains 20 foreground object classes as well as one background class. The original dataset contains 1,464 (*train*), 1,449 (*val*), and 1,456 (*test*) pixel-level labeled images for training, validation, and testing respectively. The dataset is augmented by the extra annotations provided by (Hariharan et al. 2011), resulting in 10,582 (*trainaug*) training images. We train our models on the augmented training set and report the performance on the validation set.

The Cityscapes dataset is the most widely used dataset for urban street scene understanding, which contains high quality pixel-level annotations of 5000 images (2975, 500, and 1525 for the training, validation, and test sets respectively). It contains 18 object classes and one background class for training. We report the performance on the validation set, after training on the train set. For memory reasons, we downsample the images to half resolution, *i.e.* 1024×512 for Deeplab-v2-based experiments. We keep the original image size for experiments with Deeplab-v3+ in order to achieve better results.

The evaluations on both datasets are mainly based on the mIoU score, which is a standard for semantic segmentation task. Since many new evaluation metrics have recently been proposed (see Section 2.4) and are expected to better measure the quality of segmentation results, we also use two additional metrics, namely $F_1 - score$ and $E - measure$ (enhanced-alignment measure) (D.-P. Fan et al. 2018) to evaluate our method. The $F_1 - score$ is another wildly used metric in segmentation, which is the harmonic mean of precision and recall ($F_1 = \frac{2}{recall^{-1} + precision^{-1}}$). The $E - measure$ is a metric recently proposed by (D.-P. Fan et al. 2018). The main idea of $E - measure$ is to align the bias matrix ϕ of the predicted and the ground truth mask at each position:

$$\xi_{align} = \frac{2\phi(GT) \circ \phi(pred)}{\phi(GT) \circ \phi(GT) + \phi(pred) \circ \phi(pred)}$$

where $\circ$ denotes the Hadamard product and $\phi(X) = X - \mu_X A$ is the bias matrix (A is a matrix in which all the element values are 1). This align matrix ξ_{align} is then enhanced by a convex $f(x) = \frac{(1+x)^2}{4}$, which suppresses decrease (which means having smaller derivative value) at negative value ($\xi_{align}(x, y) \leq 0$) regions and strengthens increase at positive value ($\xi_{align}(x, y) \geq 0$) regions. Finally, the $E_{measure}$ is defined as the mean value of the enhanced align matrix across all positions:

$$E_{mesure}(GT, pred) = \frac{1}{w \times h} \sum_{x=1}^{w} \sum_{y=1}^{h} f(\xi_{align}(x, y))$$

where w and h are the weight and height of the image. The E_{mesure} combines local pixel values with the image-level mean value in one term, and can jointly capture image-level statistics and local pixel matching information, as demonstrated in (D.-P. Fan et al. 2018).

We validate SEMEDA by showing that (a) introducing a structural term *via* semantic edge detection allows us to better capture the underlying structure of the objects and (b) that the SEMEDA network encodes richer embeddings compared to a naïve approach (e.g. using Sobel kernels). Most edge-based segmentation methods (L. Chen et al. 2016; Audebert et al. 2019) consist in performing edge detection as an auxiliary task and then incorporating the edge features into the segmentation prediction. Our method involves the use of an additional loss function which can better capture structural information, including edge information, from the segmentation masks. Thus, our method is complementary to the edge-based methods mentioned above, and can be used in conjunction

Method	λ_1	λ_2	λ_3	Cityscapes	VOC 2012
PPCE	-	-	-	75.5	77.1
Sobel	0.5	-	-	75.4	77.3
	1	-	-	76.1	77.5
	2	-	-	76.2	77.4
	4	-	-	76.3	77.6
SEMEDA	0.5	0.5	0.5	76.9	77.6
	1	0.5	0	76.2	77.9
	0.5	1	0	76.3	77.8
	1	0.5	0.25	76.3	78.1
	1	1	0	76.1	78.5
	2	4	0	76.9	**78.8**
	4	2	0	**77.3**	78.6
	4	4	0	77.1	78.4

Table 3.1. – Comparison of results (% mIoU) on Cityscapes and VOC 2012 validation sets with different hyperparameter (λ_l) values and an ImageNet pre-trained Deeplab-v3+/Xception 71 model. SEMEDA provides better results in all tested configurations.

with them. To this end, we do not compare our method with these edge-based methods, and we conduct our experiments with two baselines:

- **PPCE:** a segmentation network trained with a traditional per-pixel cross-entropy loss.

- **Sobel:** a setup similar to SEMEDA, except the SEMEDA network is replaced by Sobel kernels that independently process each class in the segmentation mask (i.e. matching the predicted segmentation mask with its corresponding ground truth in the embedding space generated by applying sobel kernels on both masks).

To better analyze the improvements that stem from applying SEMEDA, we do not use tricks such as adding multi-scale and flipped images during inference. In what follows, we show that SEMEDA significantly enhances the segmentation accuracy regardless of the architecture of the segmentation network, the underlying backbone, the pre-training strategy and on multiple datasets.

3.4.3 SEMEDA Parametrization

In this section, we provide insight into the behavior of SEMEDA depending on the only additional hyperparameters $\{\lambda_l\}_{l=1...L}$, which are the weights of each layer in the SEMEDA loss (see Equation 3.5). For Sobel, there is only one hyperparameter, the coefficient of the edge term λ_1. We perform ablation study on the Cityscapes and the VOC 2012 datasets with an ImageNet pre-trained Deeplab-v3+/Xception 71 model. Results under the standard evaluation metric mIoU are shown in Table 3.1. While the Sobel baseline model does improve the accuracy slightly, particularly with $\lambda_1 = 4$, SEMEDA allows a far more significant accuracy boost in all tested configurations. Likely, this is due to the fact that through its convolutional layers, SEMEDA mixes the class-wise segmentation channels in a one-vs-one manner, while the Sobel baseline model does not, separating classes in a one-vs-all manner. Thus, SEMEDA encodes richer embeddings that more efficiently capture structure in the segmentation masks. It is worth noting that matching the output of the last layer corresponding to semantic edges (λ_3) contributes less to the performance. The reason is that the output of SEMEDA is a binary mask, where no distinction is made between the semantic edges belonging to different classes. In other words, in such a case, the presence of an edge at a specific location simply implies that this pixel marks a boundary between different objects whose categories are unknown. Therefore, this last layer of the SEMEDA network contains much less information than the first ones. To draw a parallel, this echoes the results obtained in Johnson et al. 2016, where it is better to put more emphasis on the first layers of the SEMEDA network (which means λ_1 and $\lambda_2 > 1$).

With this ablation study of SEMEDA hyperparameters, we then fix them to the value that gives the best results for the rest of experiments.

3.4.4 Quantitative Validation

In this section, we quantitatively validate our method by showing that with our SEMEDA loss function, segmentation results are systematically improved in different settings such as different evaluation metrics, different pre-training strategies and different backbone networks.

As presented in Section 2.4, the mIoU metric has many drawbacks and does not necessarily fully reflect the quality of the segmentation results. To this end, we also evaluate our methods against another commonly used metric, *F1-score*, as well as the recently proposed *E-measure* (enhanced-alignment measure) (D.-P. Fan et al. 2018) described in Section 3.4.2. The comparison between the baseline model and the same model trained with SEMEDA on VOC 2012 val set (resp. Cityscapes val set) are shown in Table 3.2 (resp. Table 3.3). Our method is better than the baseline method for all these evaluation metrics on both datasets. On average, Semeda provides an increase of 1.7% on mIoU, 1.1% on $F_1 - score$ and 1.3% on $E - measure$. Table 3.2 and Table 3.3 also feature per-class comparison for each metric, showing that SEMEDA consistently improves the baseline accuracy for nearly every class and metric.

Metric	Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
IoU	Baseline	86.3	40.1	**89.5**	71.6	81.6	94.6	86.4	93.0	39.3	87.9	54.8	89.1	85.4	82.2	85.2	58.8	88.7	50.1	85.1	76.1	77.1
	Semeda	**88.3**	**41.0**	88.8	**73.1**	**82.8**	**95.1**	**87.1**	**93.1**	**42.9**	**90.3**	**56.7**	**89.2**	**88.7**	**85.4**	**86.2**	**59.6**	**90.1**	**51.5**	**89.6**	**80.9**	**78.8**
F1-score	Baseline	92.6	57.2	**94.5**	83.5	89.8	97.2	92.7	96.4	56.4	93.5	70.8	94.2	92.1	90.3	92.0	74.0	94.0	66.8	92.0	86.4	85.9
	Semeda	**93.8**	**58.1**	94.1	**84.5**	**90.6**	**97.5**	**93.1**	96.4	**60.0**	**94.9**	**72.4**	**94.3**	**94.0**	**92.1**	**92.6**	**74.7**	**94.8**	**68.0**	**94.5**	**89.4**	**87.0**
E-measure	Baseline	92.9	74.3	**94.3**	87.9	69.6	90.4	**81.5**	94.4	68.7	**92.4**	67.9	**95.7**	93.2	92.9	88.0	65.9	91.9	**73.4**	93.3	83.6	85.1
	Semeda	**94.8**	**75.9**	93.8	87.9	**73.6**	**91.4**	81.2	**94.9**	**76.1**	92.2	**69.5**	95.1	**94.6**	**94.0**	88.0	**70.6**	**95.5**	73.0	**95.1**	**86.1**	**86.6**

Table 3.2. – Different evaluation metrics (mIoU, F1-score and E-measure) on VOC 2012 val set. Our method is better than the baseline for all of these evaluation measures.

Metric	Method	road	sidewalk	building	wall	fence	pole	light	sight	veg	terrain	sky	person	rider	car	truch	bus	train	mbike	bike	mean
IoU	Baseline	98.1	84.2	92.0	57.7	**60.4**	59.6	60.0	72.3	91.9	63.7	94.4	77.8	55.6	94.1	83.1	83.8	73.5	59.5	72.4	75.5
	Semeda	**98.2**	**85.3**	**92.6**	**58.3**	59.9	**62.4**	**63.6**	**75.1**	**92.4**	**63.9**	**94.9**	**79.9**	**59.2**	**95.0**	**85.2**	**89.2**	**76.9**	**62.0**	**74.2**	**77.3**
F1-score	Baseline	99.0	91.4	95.8	73.2	**75.3**	74.7	75.0	83.9	95.8	77.8	97.1	87.5	71.4	96.9	90.8	91.2	84.7	74.6	84.0	85.3
	Semeda	**99.1**	**92.01**	**96.2**	**73.7**	74.9	**76.9**	**77.8**	**85.8**	**96.1**	**77.9**	**97.4**	**88.8**	**74.4**	**97.5**	**92.0**	**94.3**	**86.9**	**76.5**	**85.2**	**86.5**
E-measure	Baseline	99.2	96.6	97.8	77.4	**87.8**	**94.6**	85.7	94.2	**98.5**	80.4	97.3	92.2	79.0	98.2	75.9	87.7	86.7	73.7	88.3	89.0
	Semeda	99.2	**96.7**	**97.9**	**79.0**	87.4	94.5	**87.8**	**94.4**	98.4	**81.1**	**97.4**	**92.7**	**80.0**	**98.5**	**83.6**	**87.5**	**87.7**	**77.3**	**88.7**	**90.0**

Table 3.3. – Different evaluation metrics (mIoU, F1-score and E-measure) on Cityscapes val set. Our method is better than the baseline for all of these evaluation measures.

Table 3.4 shows results obtained on VOC 2012 and Cityscapes datasets, with Deeplab-v2 architecture and either Imagenet/MSCoco pre-training. MSCoco pre-training refers the process that further fine-tuning ImageNet pre-trained models on the MS-Coco dataset (T. Lin et al. 2014). For both pre-training strategies, SEMEDA improves the baseline about 1.3% on VOC 2012 and about 2% on Cityscapes. The improvements with ImageNet pre-training models are slightly higher than MSCoco fine-tuning (1.3% vs 1.2% on VOC 2012 and 2.5% vs 2.0% on Cityscapes), this may be because the MSCoco fine-tuning is done with the PPCE loss function.

Table 3.5 shows results obtained with and without SEMEDA with a stronger Deeplab-v3+ baseline and two different backbone networks: ResNet101 and Xception71. Both backbone networks are pretrained on the ImageNet dataset. In coordination with the results obtained in (Liang-Chieh Chen et al. 2018), Xception-71 as a backbone improves the segmentation results over ResNet101. For both backbones, SEMEDA further enhances the performance on both VOC 2012 and Cityscapes.

To sum up, in all these configurations, regardless of the dataset, evaluation metric, pre-training method, or backbone network, SEMEDA substantially enhances the overall accuracy. Likewise, this is due to the fact that SEMEDA better

Pre-train	Method	VOC 2012	Cityscapes
ImageNet	PPCE	72.9	64.1
	SEMEDA	**74.2**	**66.6**
ImageNet + COCO	PPCE	75.3	65.8
	SEMEDA	**76.5**	**67.8**

Table 3.4. – Results (%mIoU) on Cityscapes and VOC 2012 validation sets with Deeplab v2 and different pre-training strategies. For both pre-trained strategies, SEMEDA substantially enhances the performance.

Backbone	Method	VOC 2012	Cityscapes
ResNet-101	PPCE	74.4	72.4
	SEMEDA	**75.3**	**73.6**
Xception-71	PPCE	77.1	75.5
	SEMEDA	**78.8**	**77.3**

Table 3.5. – Results (%mIoU) on Cityscapes and VOC 2012 validation sets with Deeplab-v3+ pre-trained on ImageNet (with no additional pre-training on COCO or JFT). For both backbone networks, SEMEDA substantially enhances the performance.

captures the structure of objects, most notably at the boundary between different classes. To verify this intuition, we evaluate SEMEDA on boundary/non-boundary trimaps (a narrow band around boundary, illustrated in Figure 3.4), as it was done in (L. Chen et al. 2016; Csurka et al. 2013): at test time, we divide the pixels into two subsets depending on whether they belong to a boundary (trimap) or non-boundary region, as indicated by the semantic edge maps generated from the ground truth segmentation masks. To do so, we vary the width of a band centered on the boundary and count as positive all the pixels in the region defined by this band, negative otherwise. Thus, the more the width increases, the less precise the boundary definition is. Results are illustrated in Figure 3.5 with Deeplab-v3+ architecture. SEMEDA significantly enhances the mIoU on the boundary regions on both datasets, meanwhile, the Sobel baseline lies closer to the baseline performance. Particularly for strict boundaries (trimap width 2, 3), the mIoU improvement is 4 pts on VOC 2012 and 2.4 pts on Cityscapes, which is considerable. On non-boundary regions, the improvement is also very significant on both datasets. This is due to the fact that SEMEDA strongly penalizes the presence of holes or discontinuities in the internal structure of the predicted objects (which are tagged as non-boundary on the ground truth markups). Thus,

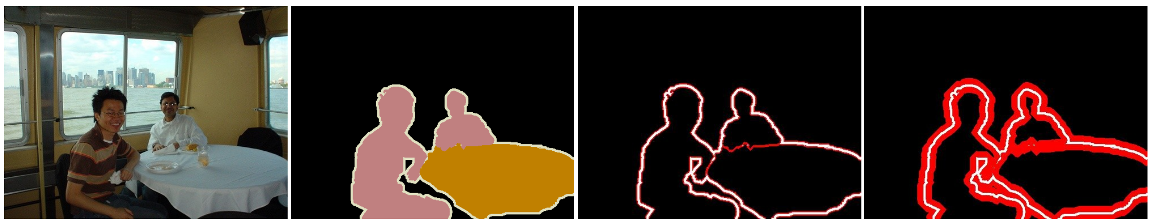


Figure 3.4. – Left to right: original image, ground truth segmentation and two boundary/non-boundary trimaps: one with 1 pixel width and the other with 10 pixels width. The red areas represent the boundary regions while black areas represent non-boundary regions. White areas depict 'void' pixels.

SEMEDA allows us to better capture the structure of objects, as well as to refine the inter-class boundaries in the segmentation masks.

3.4.5 Comparison with State-of-the-art Approaches

In this section, we compare our model with leading semantic segmentation methods on both the VOC 2012 *test* set and the Cityscapes *test* set. After finding the best model variant on val set, we then further fine-tune the model on the *validation* set. Our proposed SEMEDA attains the test set performance of 86.0% on VOC 2012 and 77.1% on Cityscapes, as shown in Table 3.6 and Table 3.7, which are comparable with other leading methods. In both datasets, SEMEDA is better than the reproduced Deeplab-v3+ model trained with PPCE loss.

Method	mIoU
LRR 4x ResNet-CRF (Ghiasi et al. 2016b)	79.3
Deeplab-v2 (L.-C. Chen et al. 2016)	79.7
SegModel (Shen et al. 2017)	81.8
Deep Layer Cascade (Xiaoxiao Li et al. 2017)	82.7
TuSimple (P. Wang et al. 2018)	83.1
Large Kernel Matters (Peng et al. 2017)	83.6
RefineNet (G. Lin et al. 2017)	84.2
PSPNet (H. Zhao et al. 2017)	85.4
Deeplab-v3 (L.-C. Chen et al. 2017)	85.7
EncNet (Zhang et al. 2018)	85.9
Deeplab-v3+(reproduced)	85.0
Semeda (ours)	**86.0**

Table 3.6. – Test set results on PASCAL VOC 2012 (mIoU)

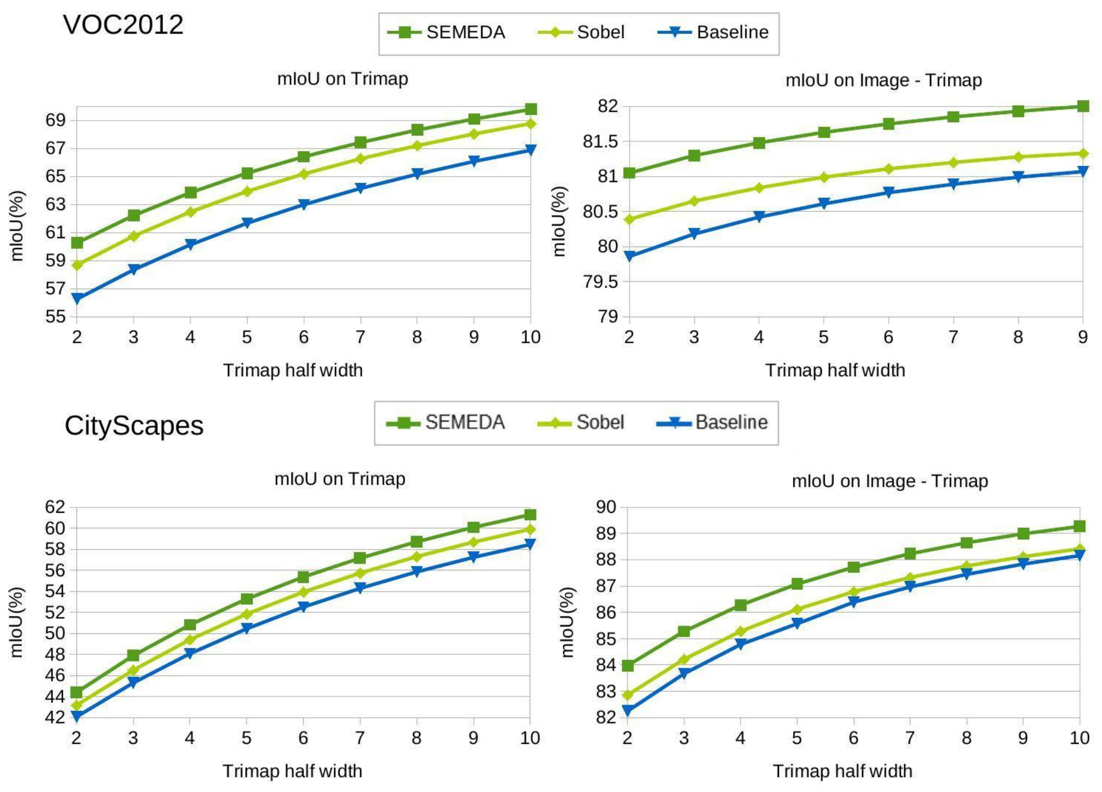


Figure 3.5. – mIoU for different models on boundary and non-boundary regions on the VOC 2012 and the Cityscapes datasets. SEMEDA consistently enhances the performance on both regions for all bandwidth.

Method	Coarse	mIoU
Deeplab-v2-CRF (L.-C. Chen et al. 2016)	✗	70.4
Deep Layer Cascade (Xiaoxiao Li et al. 2017)	✗	71.1
ML-CRNN (H. Fan et al. 2018)	✗	71.2
LRR-4x (Ghiasi et al. 2016b)	✓	71.8
RefineNet (G. Lin et al. 2017)	✗	73.6
FoveaNet (Xin Li et al. 2017)	✗	74.1
PSPNet (H. Zhao et al. 2017)	✗	76.3
Deeplab-v3+(reproduced)	✗	75.4
Semeda (ours)	✗	**77.1**

Table 3.7. – Test set results on Cityscapes (mIoU)

It is worth noting that, due to hardware limitations, we were unable to reproduce the results published in the Deeplab-v3+ paper (Liang-Chieh Chen et al. 2018). The biggest difference is that we have to use small batch size with fixed batch norm parameters during training. The importance of large batch sizes for training Deeplab models has been experimentally validated in (L.-C. Chen et al. 2017).

The original Deeplab-v3+ experiments (Liang-Chieh Chen et al. 2018) set batch size at 16 to train the batch norm parameters, while in our experiments we could only set batch size at 4 on VOC 2012 and at 2 on Cityscapes with fixed batch norm parameters. Other training strategies described in (L.-C. Chen et al. 2017; Liang-Chieh Chen et al. 2018) such as decreasing output stride to 4 for training, duplicating the images that contain hard classes (on VOC 2012) and fine-tuning with a coarse annotated dataset (Cityscapes) were also omitted for the same reason.

In the upcoming sections, we present qualitative results of the models trained with SEMEDA to better understand where these quantitative improvements come from.

3.4.6 Qualitative Assessment

To understand what SEMEDA network has learned, Figure 3.6 illustrates the semantic edge embeddings learned by the first two CNN layers of the SEMEDA network. These embeddings are visually similar to traditional edge maps (c1,c2,c3), except that the filters of the SEMEDA network encompasses inter-class relationships in a one-vs-one fashion (d1,d2,d3), instead of simply separating object and other classes in a one-vs-all setting, as it is the case with simpler edge detector such as Sobel kernels. These richer embeddings can more efficiently encompass the structure in the segmentation masks.

To better illustrate the effectiveness of our proposed method, we showcase the entropy maps of the predictions in Figure 3.7, illustrating the confidence of the predictions (the higher the entropy score, the less robust the classification is). Thus, with our method, high entropy activations (red points) only occur along the very boundaries of objects, whereas the baseline method produces high entropy scores both inside and at the boundary of objects. These results qualitatively prove that our approach do resolve the **problem a-b** described above: it not only refines the accuracy on the boundary pixels, but also makes the predictions within the object more certain and more uniform. These visualizations also validate the results obtained with Trimap experiments in the previous section (Section 3.4.4), namely that SEMEDA improves the performance on both boundary and non-boundary regions.

Figure 3.8 and Figure 3.9 show segmentation masks outputted using Deeplab-v3+/Xception-71 trained with SEMEDA and with PPCE only (baseline). For each image, the segmentation mask provided by the network is overlayed with the input image. As observed, predictions generally conform better geometric edges when SEMEDA is applied. For example, in the first image of VOC 2012, SEMEDA corrects wrong predictions (class dog (purple) confused with class horse (pink)) and produces better shaped predictions. This improvement is consistent on other

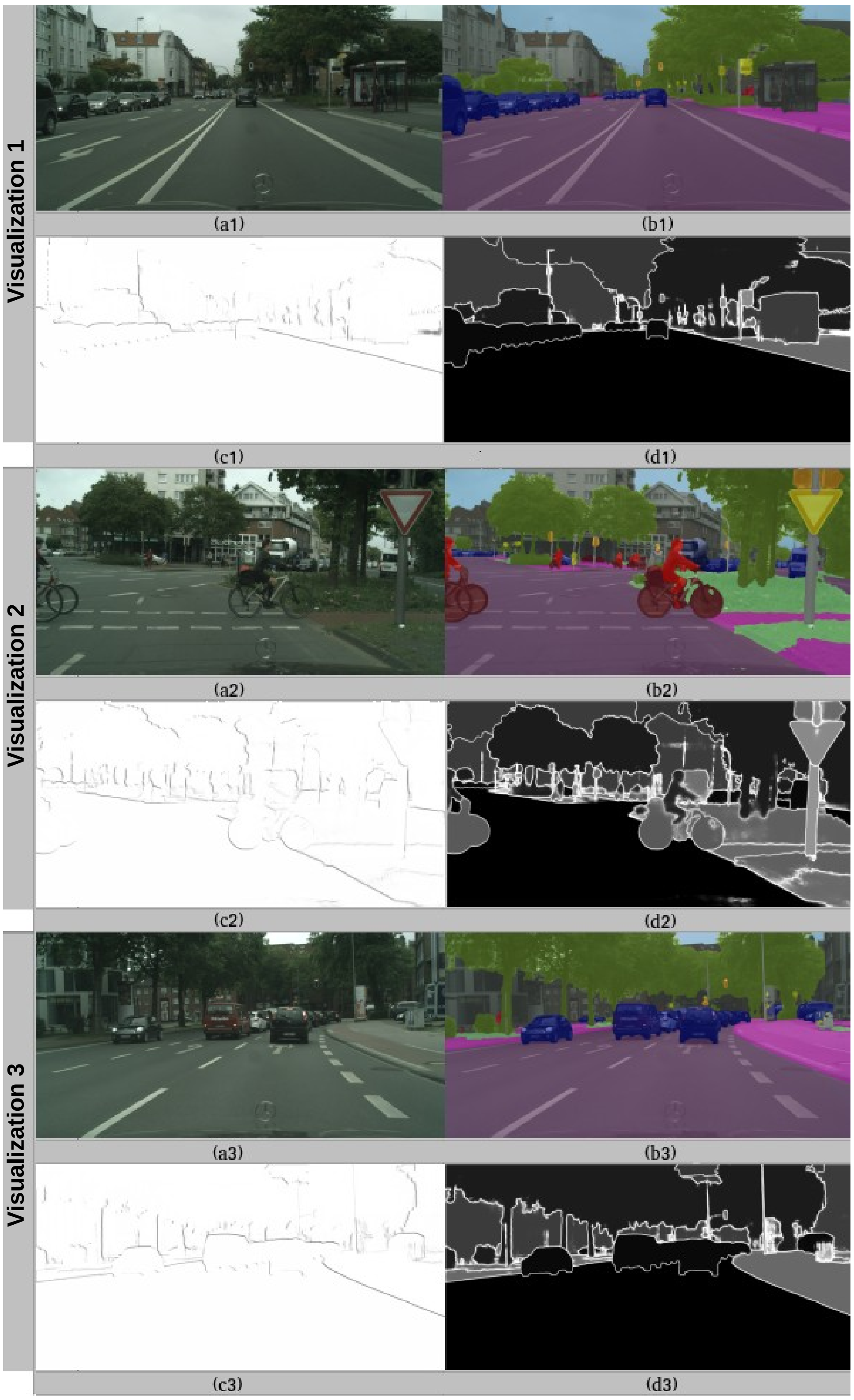


Figure 3.6. – Example of embeddings of the SEMEDA network: (a1,a2,a3) original images, (b1,b2,b3) predicted segmentation masks, (c1,c2,c3) and (d1,d2,d3) two feature maps of the SEMEDA network.

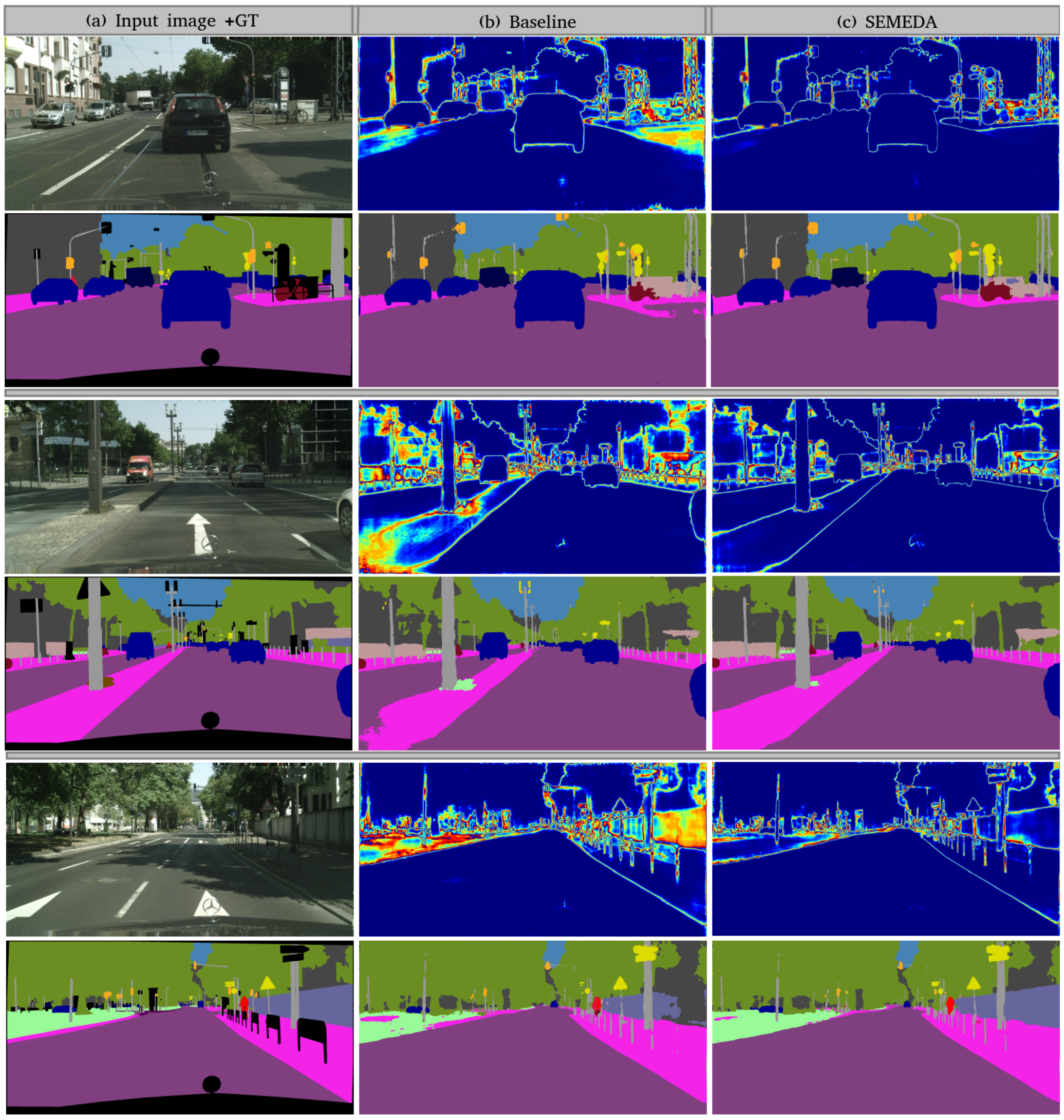


Figure 3.7. – Column (a) shows an input image and the corresponding semantic segmentation ground-truth. Column (b) and (c) show segmentation results (bottom) along with prediction entropy maps produced by different approaches (top). The baseline model produces noisy segmentation predictions as well as high entropy activations. Our models, on the other hand, manage to produce correct predictions at high level of confidence. (Red signifies a high entropy value.)

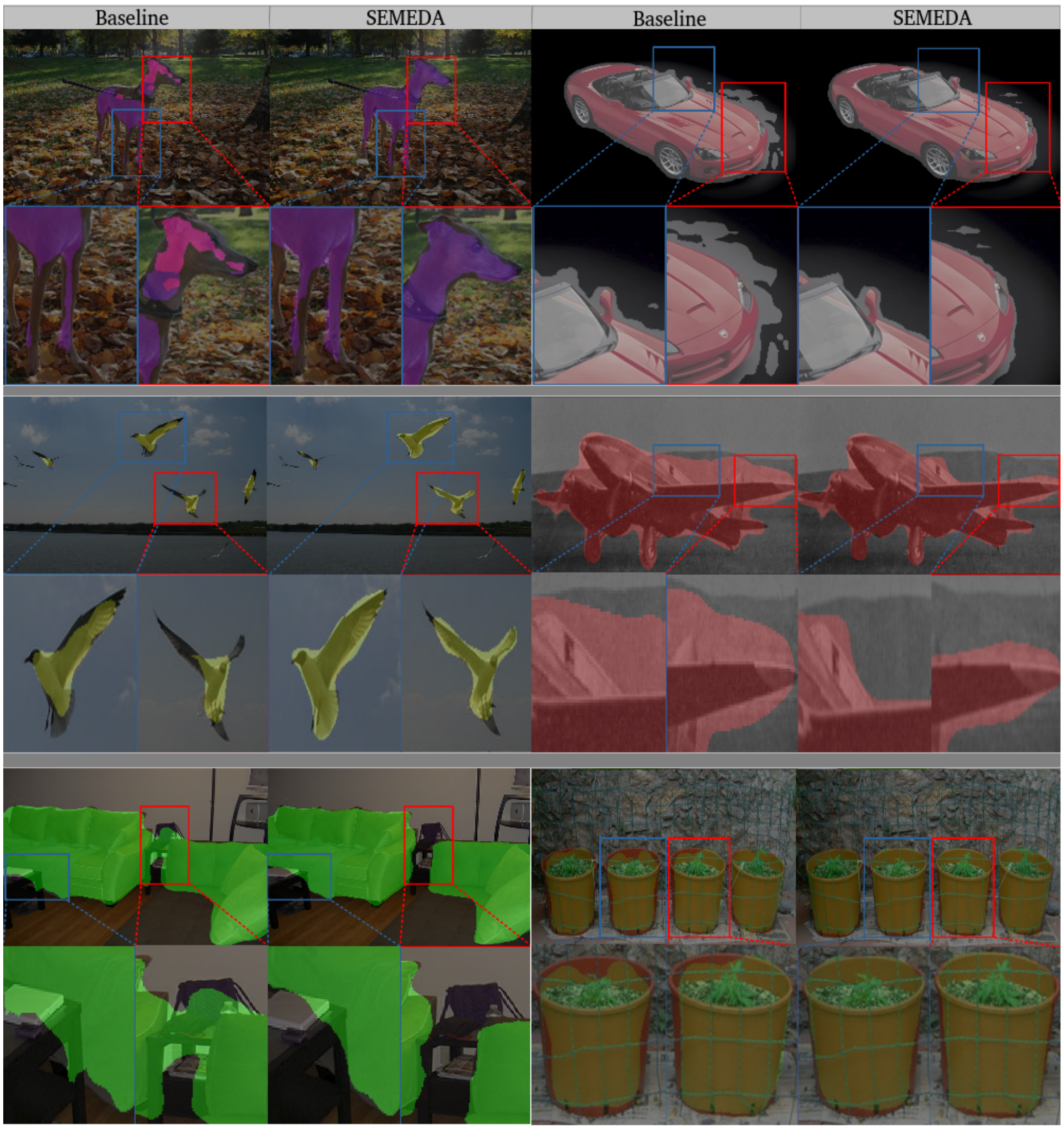


Figure 3.8. – Examples of predicted segmentation masks on the VOC 2012 dataset, and comparison between a baseline Deeplab-v3+ model trained with PPCE and SEMEDA. SEMEDA produces better shaped predictions compared to the baseline.

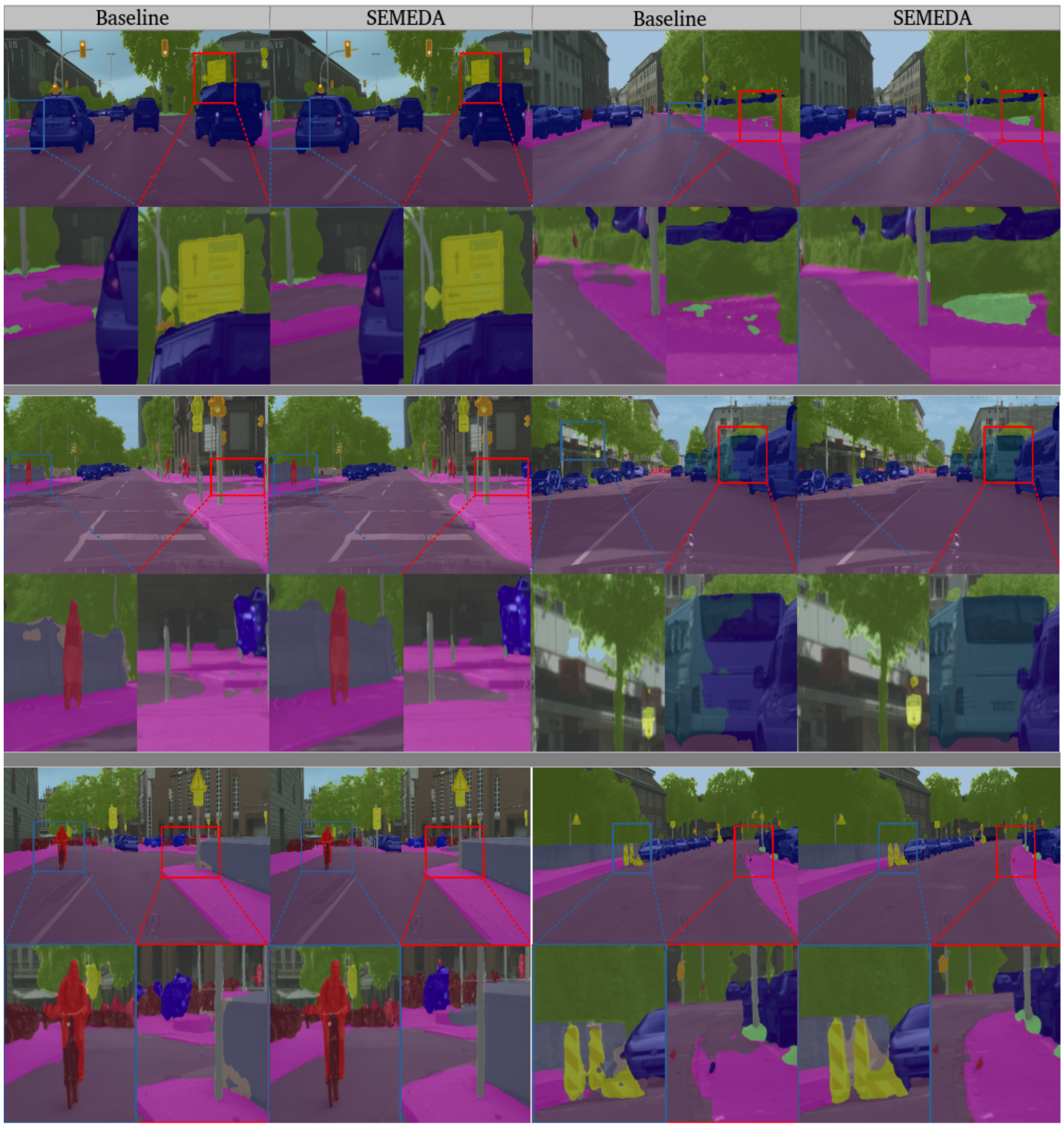


Figure 3.9. – Examples of predicted segmentation masks on the Cityscapes dataset, and comparison between a baseline Deeplab-v3+ model trained with PPCE and SEMEDA. SEMEDA produces better shaped predictions compared with the baseline.

images of VOC 2012 and also on Cityscapes where predictions of different classes such as traffic sign and sidewalk, are better conformed with geometric edges.

As stated above, SEMEDA allows the model to better capture the structure of the segmented objects, by putting more emphasis on the inter-class boundaries, as well as to avoid discontinuities (*e.g.* holes) inside the objects. With the proposed SEMEDA strategy, fine-grained elements such as traffic signs, tree leaves or people shapes are better segmented (see Figure 3.8 and Figure 3.9) and the ambiguity within large objects is also significantly reduced (see Figure 3.7). These qualitative results validate the quantitative improvement obtained in Section 3.4.4 and Section 3.4.5.

As mentioned in Section 2.2.3, the performance of segmentation algorithms can be affected by the quality of datasets. To demonstrate the robustness of our proposed method in what follows, we apply our method to two datasets where the annotations are noisier.

3.5 Applications

We have shown the effectiveness of our method on two of the most popular open datasets for semantic segmentation where the annotations are of relatively good quality. Since our method focuses on the local structure of segmentation masks, it is interesting to see how our method performs on noisier datasets. We present the quantitative and qualitative results of the application of our method to two other tasks: face parsing on **HELEN** dataset (Le et al. 2012) and lesion boundary segmentation on **ISIC2018** dataset (Codella et al. 2019). For both applications, we employ Deeplab-v3+ with Xception-71 as our backbone segmentation network and use the same SEMEDA architecture as the one described in Section 3.4.1.

3.5.1 Face Parsing

In face image analysis, one common task is to parse an input face image into facial parts. HELEN is a widely used face parsing dataset that containing 2,330 annotated images. It features high quality, real-world photographs of people with a more balanced proportion of genders, ages, and ethnicities than other face datasets. Each image is densely annotated with 11 facial components labels, including hair, skin, “left/right brows”, “left/right eyes", nose, “upper/lower lip”, mouth and background. It worth noting that only the most centered face in each image is annotated, and the labeling of HELEN’s training data is not very precise, especially for hair and skin as shown in Figure 3.10. These deficiencies limit the performance of the segmentation models trained on it.

We adopt the same dataset division setting as in (C. Liu et al. 2011; Smith et al. 2013; S. Liu et al. 2015) that uses 2,000 images for training, 230 images for

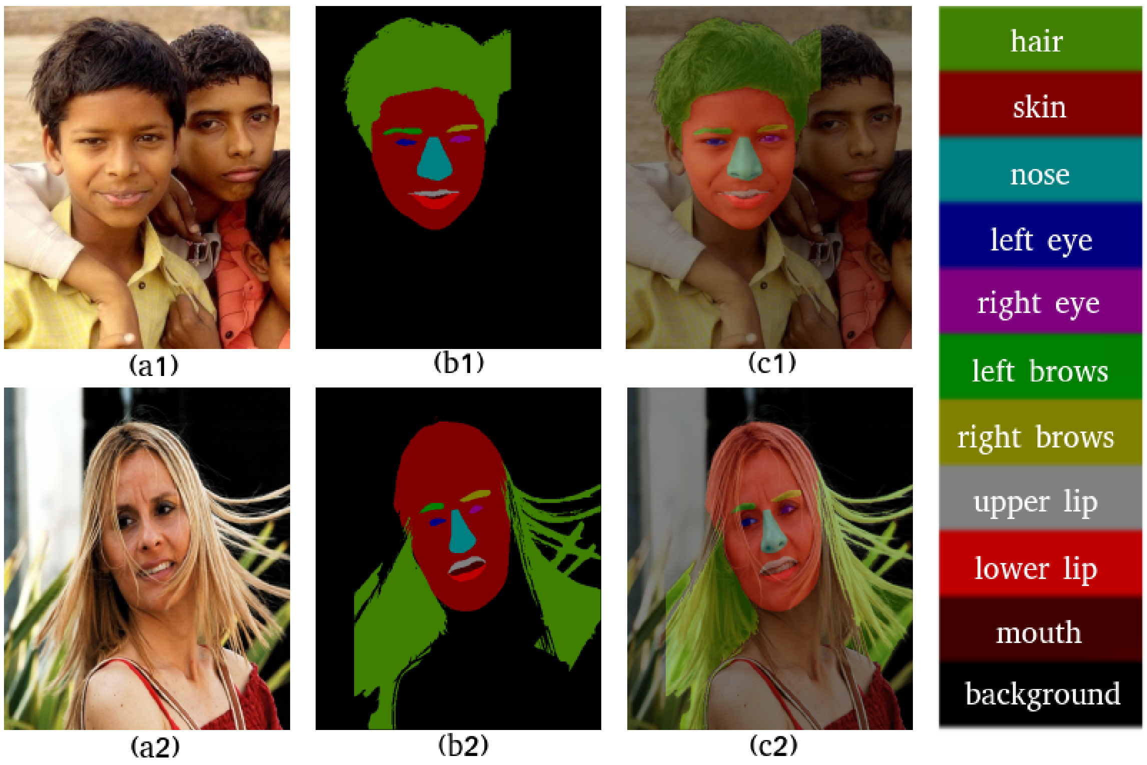


Figure 3.10. – Training samples of the HELEN dataset. (a1,a2): RGB images. (b1,b2): Ground truth masks. (c1,c2): Ground truth masks superimposed on the RGB image. Only the centered face is annotated (a1-c1). There are also quite a lot of inaccurate annotations in HELEN, especially for hair(green) and skin(red).

validation and 100 images for testing. We firstly find the best hyper-parameter values on *val* set, and then fine tune the model on the *train+val* set. We report the F_1-score on the *test* set which is commonly used in the existing face parsing literature. Due to the inaccuracy of the annotations, we do not evaluate models on class "hair" and "skin" (see Figure 3.10).

Method	eyes	brows	nose	I-mouth	U-lip	L-lip	mouth	Overall
C. Liu et al. 2011	77.0	64.0	84.3	60.1	65.0	61.8	74.2	73.8
Smith et al. 2013	78.5	72.2	**92.2**	71.3	65.1	70.0	85.7	80.4
S. Liu et al. 2015	76.8	71.3	90.9	80.8	62.3	69.4	84.1	84.7
PPCE	82.7	74.8	89.8	78.8	70.6	78.2	87.8	85.8
SEMEDA	**83.5**	**75.4**	91.7	**81.9**	**74.3**	**81.2**	**90.4**	**87.4**

Table 3.8. – Results (F_1-Score) for different face subparts on the HELEN test set with Deeplab-v3+/Xception-71 models. SEMEDA outperforms some of the leading methods and provides better results on all classes compared with PPCE baseline.

We show the comparison results on the HELEN *test* set in Table 3.8. Each column shows the F_1-score percentage corresponding to a specific face label. *I-mouth* is short for inner mouth, *U/L-lip* is short for upper/lower lip, and overall represents the union of all inner facial component (eyes/brows/nose/mouth) labels. As can be seen, SEMEDA performs favorably against some of the leading methods. It is worth noting that our method provides better results on all classes compared to PPCE baseline.

Figure 3.11 shows the visual parsing results on challenging images from the HELEN dataset. SEMEDA shows the ability to better handle small objects such as lips, brows and eyes. The results produced by SEMEDA not only better fit the contours of objects, but also avoid large areas of error. These quantitative and qualitative results demonstrate that our method is also effective for accurate and efficient face parsing.

3.5.2 Lesion Boundary Segmentation

The International Skin Imaging Collaboration (ISIC) is an international effort to improve melanoma diagnosis and the ISIC Archive contains the largest publicly available collection of quality controlled skin lesion images. The ISIC Challenge employs a subset of the ISIC Archive, which included a task for lesion segmentation. Th ISIC 2018 dataset for segmentation contains 2,549 image for training and 1,000 images for testing. The ground truth segmentation mask contains only 2 categories: lesion or background. They are annotated with three different methods (Codella et al. 2019), resulting in very different behavior at the boundary as shown in Figure 3.12. The extremely irregular shape of melanoma has already made this task very challenging, and the non-uniform labeling process of boundary pixels makes it even more difficult.

We randomly split the original training set into two parts: 2,000 images for training and 549 images for validation. Once we find the best hyper-parameters on validation, we further fine-tune the model 20 epochs on all 2,549 images to obtain the final model. The results on the *test* set are presented in Table 3.9. As mentioned in Section 2.4, the mIoU alone is not sufficient to accurately measure the performance of segmentation models. To this end, six different metrics are used to comprehensively evaluate the segmentation performance: Threshold Jaccard index (0 if Jaccard index is less than 0.65), Jaccard index (mIoU), Dice coefficient (F_1-score), Accuracy, Sensitivity (Recall) and Specificity (True negative rate). As can be seen, with the exception of Sensitivity, which scored slightly lower (0.3%), SEMEDA provides better results on all other scores compared to PPCE baseline.

Figure 3.13 shows the visual segmentation results on challenging images from the ISIC 2018 dataset. Because ground truth segmentation masks have extremely complicated contours, it is difficult to visually determine the quality of the segmentation results. To better illustrate the difference between predicted masks

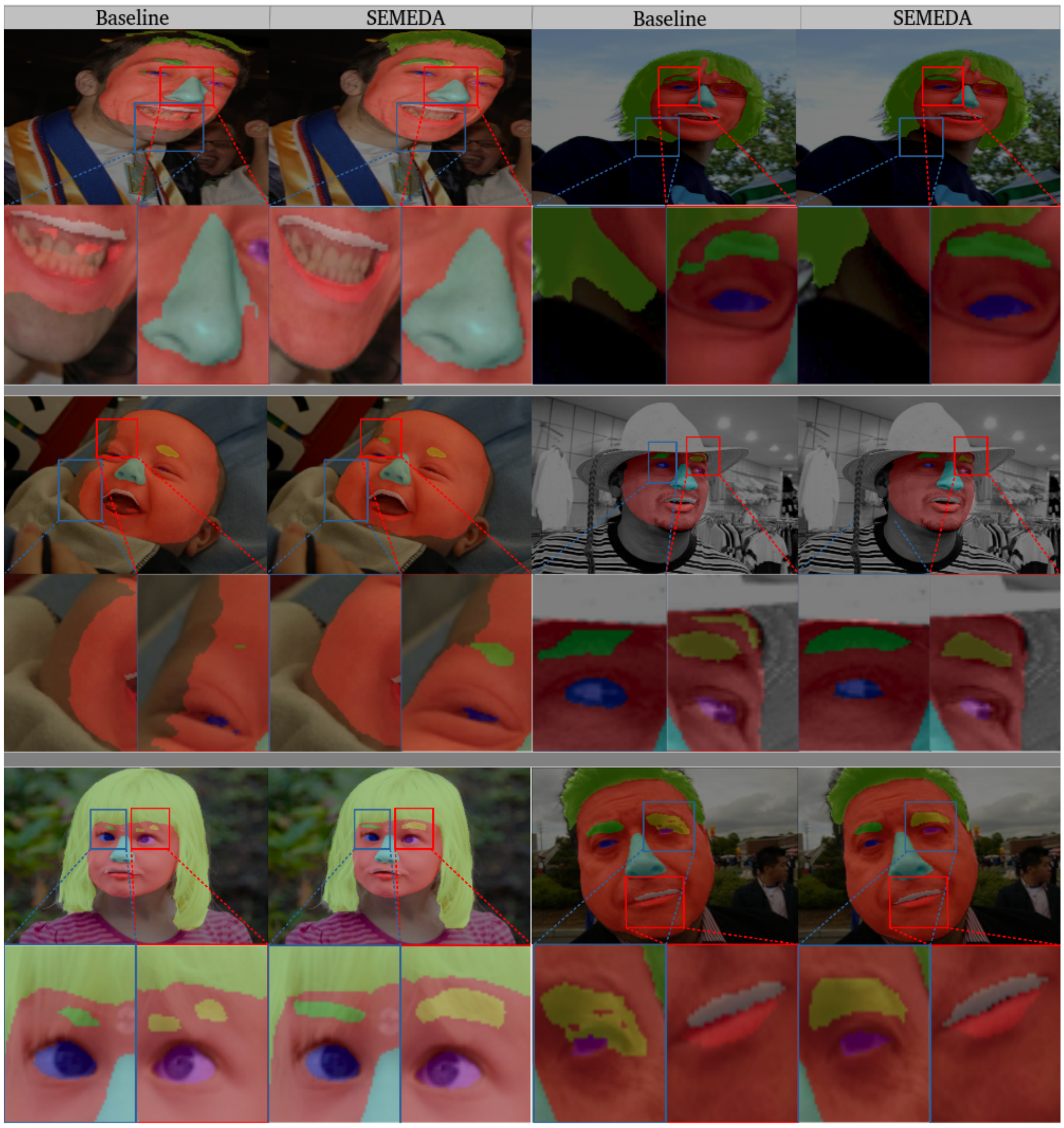


Figure 3.11. – Examples of parsing results on the HELEN dataset, and comparison between a baseline Deeplab-v3+ model trained with PPCE and SEMEDA. SEMEDA produces better shaped predictions compared to the baseline.

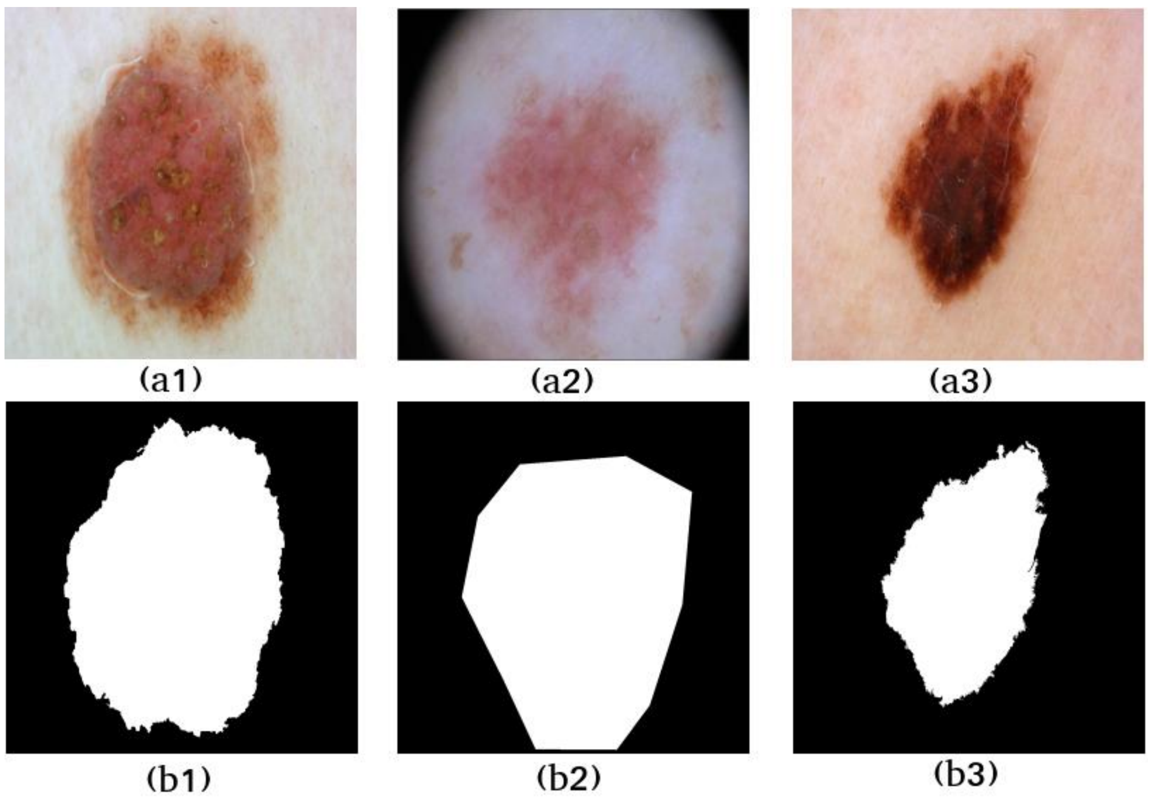


Figure 3.12. – Ground truth masks annotated with three different methods. (a1,b1): fully-automated algorithm, reviewed and accepted by a human expert. (a2,b2): manual polygon tracing by a human expert. (a3,b3): a semi-automated flood-fill algorithm, with parameters chosen by a human expert. The quality of the boundary pixels varies a lot.

Method	Threshold Jaccard	Jaccard	Dice	Accuracy	Sensitivity	Specificity
PPCE	71.9	78.7	87.1	92.8	**95.4**	90.4
SEMEDA	**75.0**	**80.6**	**88.4**	**93.5**	95.1	**91.5**

Table 3.9. – Different evaluation scores on ISIC 2018 *test* set with Deeplab-v3+/Xception-71 models. SEMEDA outperforms the PPCE baseline on almost all of these evaluation metrics.

and ground truth masks, we use two different colors in the visualization: the red regions represent False Positives and the green regions represent False Negatives. Therefore, the smaller the red and green areas, the better the results of segmentation. Since there is only one melanoma in each image, the only differences lie on the contours. As can be seen, although both methods fail to produce fine contours as the ground truth, SEMEDA still improves the quality of the contours compared to the baseline.

3.6 Conclusion

In this chapter, we mainly tackled the difficulty **D2** and introduced a new loss function for semantic segmentation which solves the drawbacks of the commonly used PPCE loss. Our approach leverages a semantic edge-aware loss to implicitly integrate structural information into segmentation predictions. It consists in training a semantic edge detection (SEMEDA) network to map segmentation masks to the corresponding edge maps. The predictions outputted by the segmentation network can then be optimized (via the proposed SEMEDA loss) in the embedding space of the semantic edge detection network, similarly to perceptual losses.

Through extensive evaluation on several datasets with very different application contexts, we showed that SEMEDA significantly enhances the overall performance of semantic segmentation networks in all tested hyperparameter configurations, segmentation network architectures, backbone networks and pre-training strategies. Furthermore, the improvements obtained with SEMEDA are consistent across different evaluation metrics. More precisely, we showed that SEMEDA works by enforcing inter-class boundary structure as well as avoiding holes in the segmented objects. In addition, SEMEDA does not require any additional annotation and only adds negligible computational overhead, and thus can be straightforwardly combined with traditional losses to improve the performance of any semantic segmentation network.

The proposed work leads us to rethink how structural information, such as semantic edge detection can be integrated into existing segmentation architectures for enhanced precision, beyond merely treating semantic edge detection and segmentation in a naive multi-task fashion. As such, it opens up a new perspectives for designing edge-enhanced semantic segmentation architectures. The experiments on HELEN and ISIC also show that SEMEDA could be adapted without bells and whistles to various applications of semantic segmentation such as face parsing and medical imaging systems.

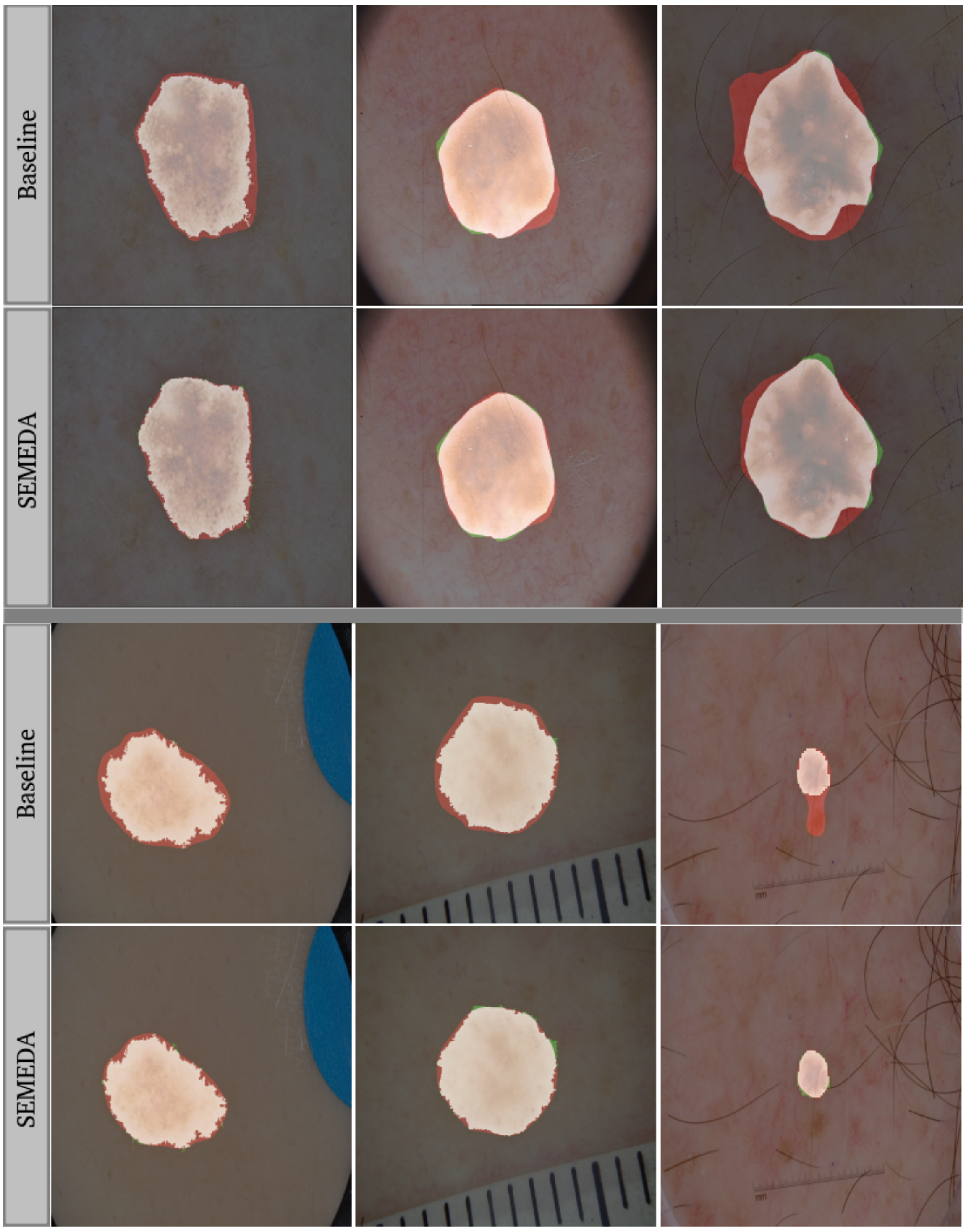


Figure 3.13. – Examples of segmentation results on the ISIC 2018 dataset, and comparison between a baseline Deeplab-v3+ model trained with PPCE and SEMEDA. The red regions represent False Positives and the green regions represent False Negatives. SEMEDA improves the contour quality compared to the baseline model.

Chapter 4

WEAKLY SUPERVISED SEGMENTATION WITH ATTRIBUTION METHODS

Contents

Chapter abstract

The success of deep learning for supervised tasks comes not only from advances in hardware as well as algorithms, but also heavily relies on large amounts of high-quality annotated data. However, for tasks like semantic segmentation, fully annotated data is very expensive and often very inaccurate. Therefore, weakly supervised approaches, which require only partially annotated data, are a good way to alleviate the burden of acquisition hand-labeled datasets. In this chapter, we first present how images annotated at the image level can be used for semantic segmentation by using an attribution method. Then, based on mathematical properties of ReLU based CNNs, we present a new attribution method **VGatt** (Value-Gradient Attribution) that allows to identify the most discriminating part of each object present in an image considered by the classification CNNs. Finally, we apply this attribution scheme to generate high quality seeds that can be used to enhance segmentation in a weakly supervised manner.

The work in this chapter has led to the publication of a conference paper:

- Yifu Chen, Antoine Saporta, Arnaud Dapogny, and Matthieu Cord (2019). "Delving Deep into Interpreting Neural Nets with Piece-Wise Affine Representation". In: *IEEE International Conference on Image Processing (ICIP)*.

4.1 Introduction

In this chapter, we mainly address the problem **D3** introduced in Section 1.2: the lack of annotated data for segmentation. As described above, many computer vision problems belong to supervised learning tasks that requires massive amounts of annotated data. For these problems, labeled data is somehow as important as the machine learning algorithm themselves: a large amount of very precisely annotated data is usually required for these algorithms to work, and their performance could be affected by the decrease in the amount of data and the quality of labels in the training set. Although tons of data are available and can be accessed freely on the Internet today, most of it is unlabeled data or weakly labeled data (data with labels that are not fully adapted to the task). In addition, the data labeling process is often inefficient and costly, requiring human beings to manually annotate the data. Therefore, developing methods that can learn from unlabeled or weakly labeled data becomes particularly important in machine learning today.

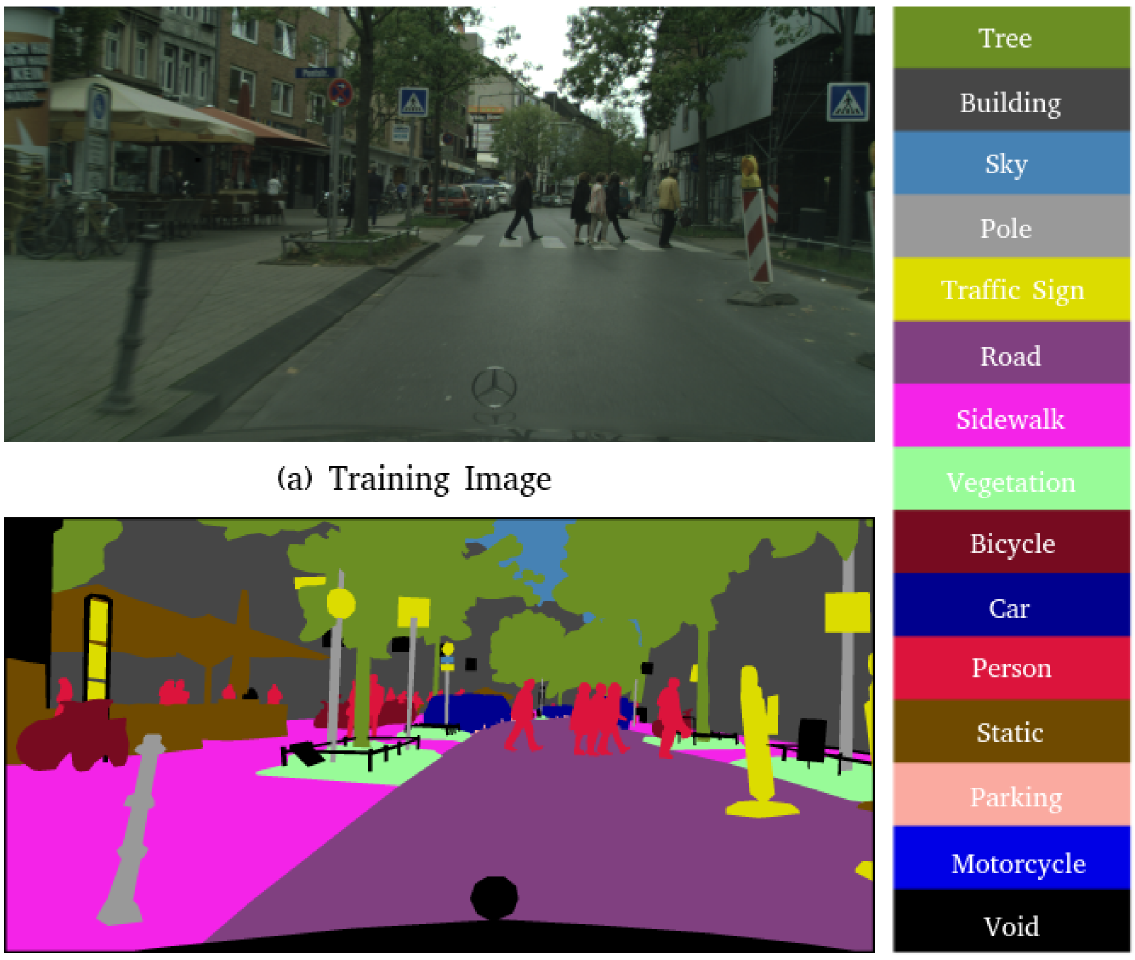
The rest of this chapter is organized as follows:

- In Section 4.2, we detail the problem of the lack of fully labeled data for the semantic segmentation task and the motivation of a Weakly Supervised Learning (WSL) framework that only uses images with image-level labels. Then, we introduce the principle of attribution methods, which identify the most relevant part to explain the prediction of Convolutional Neural Network (CNN)s. These methods therefore allow us to extract information about the location of objects from data with only image-level labels. Finally, we present the WSL pipeline that integrates the attribution methods to train segmentation models with image-level labeled data.

- Since attribution methods are the key component of the WSL pipeline described above, in Section 4.3, we focus on the attribution methods. First, we give an introduction to existing methods as well as their limitations. Then, we present our second contribution, a novel attribution method called VGatt, based on mathematical properties of a branch of state-of-the-art CNNs. By comparing with other popular attribution methods, we validate that our method can provide more accurate results.

- In Section 4.4, we employ our proposed attribution method to the weakly supervised semantic segmentation task. We carry out a wide range of experiments and analysis to show that when applied to weakly supervised semantic segmentation, the proposed attribution method enhances the accuracy.

4.2 Motivation and Context



(a) Training Image

(b) Fine annotations

Figure 4.1. – An example of fine annotated segmentation mask in Cityscapes dataset (Cordts et al. 2016). The training images have a resolution of 2048 × 1024, which means that a fine annotated mask contains over 2 million labeled pixels. The large number of different objects leads to complicated semantic boundaries, making the annotation process even more difficult.

Supervised learning methods construct predictive models by learning from a massive amount of training samples, where each training sample has one or more labels indicating its ground-truth output. This labeled data plays an important role in learning process. However, collecting annotated data usually costs a lot of time and money. For example, the ImageNet dataset (Russakovsky et al. 2015) for image recognition contains over 14 millions manually annotated images. The labeling process is carried out with Amazon Mechanical Turk, a service where hordes of humans sitting in front of computers around the world complete small online tasks for pennies. Even with Mechanical Turk, the first version of ImageNet dataset took two and a half years to complete. It consisted of 3.2 million labelled images, separated into 5,247 categories, sorted into 12 subtrees like "mammal", "vehicle" and "furniture" (J. Deng et al. 2009).

The data labeling process is even more complicated for semantic segmentation. For classification problems, only one label is needed for each image, while for semantic segmentation, we need to annotate each pixel of the image. Obtaining dense, high-quality annotations is very difficult and time-consuming, especially for pixels at the edges of objects. As a result, datasets for semantic segmentation generally contain far fewer fully annotated images compared with datasets for classification. For example, Pascal VOC 2012 (Everingham et al. 2015) contains 12 031 labeled images of size about 500×500 and Cityscapes (Cordts et al. 2016) only contains 5000 finely annotated images with a resolution of 2048×1024 as showed in Figure 4.1. In this case, WSL approaches become particularly interesting for semantic segmentation.

WSL is a branch of machine learning where incomplete, inexact, or inaccurate labels are used to provide supervision signal in a supervised learning setting. In the context of semantic segmentation, a particularly attractive framework is to train image segmentation models using training sets with only **image-level labels**, since this form of weak supervision is already provided in classification datasets and can be obtained very easily. As mentioned earlier, spatial information such as location and boundaries are the keys to segmentation models, whereas image-level labels contain only category information about the objects in the image. In order to use image-level labeled images for segmentation, we need to develop strategies to extract spatial information of objects present in the image.

As we know, the state-of-the-art CNNs can achieve high accuracy results in image classification problems, which means that these CNNs are good at finding class-discriminating information in the image to make correct predictions. Therefore, if we can identify the most discriminative regions for a given pre-trained CNN, we can then extract correspondent spatial information for segmentation. Actually, a similar problem is addressed by what we call **attribution methods** for deep neural networks. The goal of these methods is to understand how networks make decisions and interpret their predictions. In the context of Computer Vision (CV), these methods are also call visual attribution methods since they are

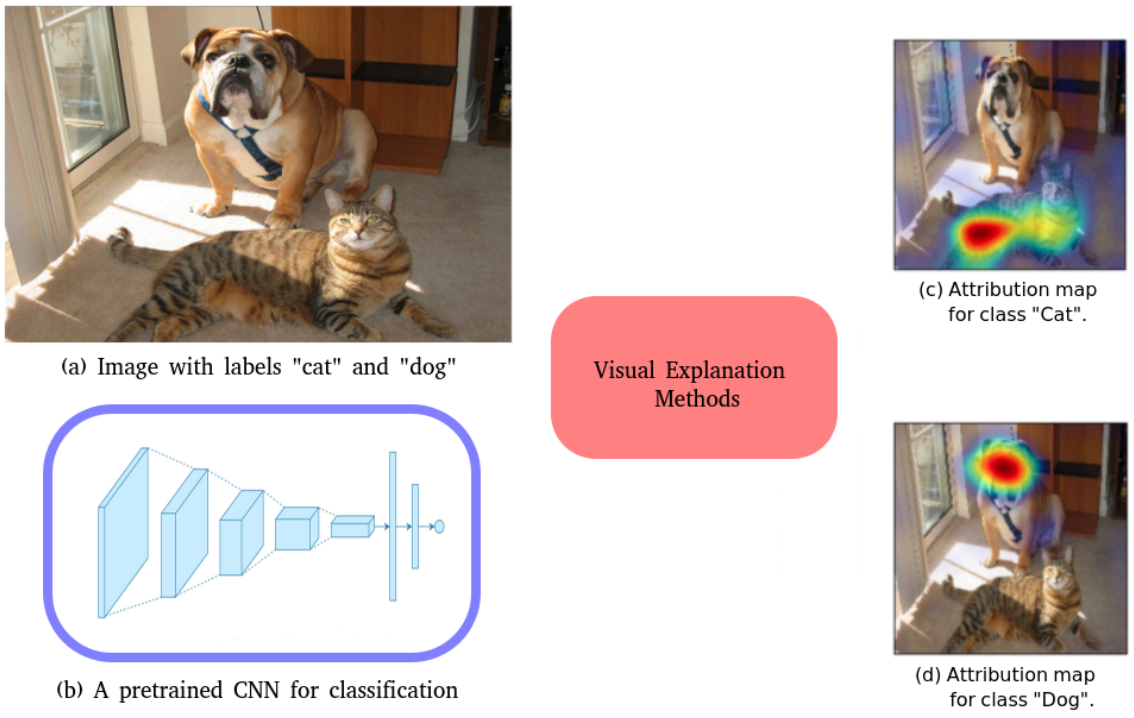

(a) Image with labels "cat" and "dog"
(b) A pretrained CNN for classification
Visual Explanation Methods
(c) Attribution map for class "Cat".
(d) Attribution map for class "Dog".

Figure 4.2. – An example of applying visual attribution methods to explain CNNs' decisions. Given a pre-trained CNN (image (b)) and an image with labels "dog" and "cat" (image (a)), attribution methods can identify the most discriminant regions of each class for this CNN. (image (c) for class "cat" and image (d) for class "dog").

used to visually interpret the predictions of CNNs. For example, let's consider a pre-trained classification CNN and an image with a cat and a dog, as shown in Figure 4.2 (a) and (b). Visual attribution methods output an "attribution map", which indicates the most discriminant regions relatively each category class given the pre-trained CNN. As shown in Figure 4.2 (c) and (d), although "attribution maps" cannot cover the entire object and usually contain some erroneous areas, they provide a good approximation of the location of the correspondent objects. Therefore, we can extract some spatial information from images labeled only at the image level, and make the weakly supervised semantic segmentation possible.

Concretely, training semantic segmentation models only with images labeled at the image level may be achieved by the following strategy. First, collecting an image dataset with image-level labels and training a classification CNN on this dataset. Second, applying attribution methods with this pre-trained CNN and images of the dataset. For example, let's consider an image I and its label set L. For each label $l \in L$, we can identify the most important region R_l of this class in the image. Pixels belonging to these regions, which are generally called as "Seeds", are then assigned the corresponding labels. Also, the least relevant

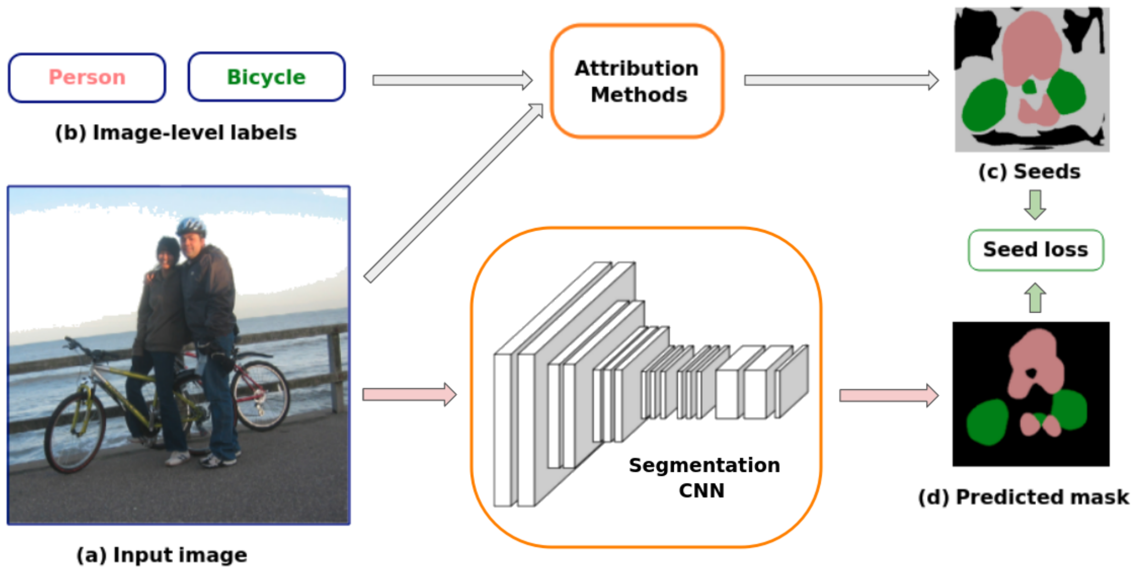


Figure 4.3. – A WSL pipeline for semantic segmentation only with image-level labeled images. By using attribution methods, we identify the most relevant parts of each class presented in the image. Furthermore, the least relevant parts for each possible objects are considered as the background class. The resulting mask, also called "Seeds", is then used as the ground truth mask for the training semantic segmentation networks with the seed loss. Thus, only the most certain parts of the object denoted as seeds, and highlighted by the attribution method, will be used for training.

regions obtained are considered as "Background". Regions that belong neither to "Seeds" nor to "background" are set as "unknown", and will not be used to train the segmentation network. In this way, we obtain a pseudo mask for each image in the dataset. Finally, these pseudo masks are considered as ground truth segmentation masks and then used to train semantic segmentation models with a per-pixel cross entropy loss that applies only to seed pixels (also known as seed loss). The entire pipeline is illustrated in Figure 4.3.

Visual attribution methods play an important role in this training strategy. Better "seeds" usually lead to better results for segmentation models. In the next section, we discuss existing methods for attribution and detail our approach. After this presentation, we explain in Section 4.4 how to use attribution methods in the WSL framework of Figure 4.3.

4.3 Attribution for Convolutional Neural Networks

We focus on the attribution problem for deep neural networks. Although nowadays CNNs are ubiquitous in computer vision, the lack of understanding behind these models makes them hard to interpret: it is however important to know why a CNN predicts what it does in order to trust the outcome of a Deep Learning (DL)-based intelligent system. Ideally, that means it shall be possible to justify a CNN's prediction by identifying relevant parts in an image as showed in Figure 4.2.

4.3.1 Related Works

A number of approaches adress the attribution problem for CNNs. A popular solution is to consider the partial derivation of the classification score w.r.t each pixel as the importance of this pixel. Specifically, Simonyan et al. (Simonyan et al. 2014) uses the absolute value of the gradient at pixel position as a relevance score, while Guided Backpropagation (Springenberg et al. 2015) and Deconvolution (Matthew D Zeiler et al. 2014) make some modifications to "raw" gradients that result in qualitative improvements. Despite producing fine-grained visualizations, these methods are not class discriminative, visualizations with respect to different classes are nearly identical. The problem of these methods is that the gradient measures how a small modification of this pixel can affect the classification score, but not the contribution of this pixel for classification. In what follows, we take two examples to explain this point.

Example 1 First, we consider a binary classifier $f(x_1, x_2) = x_1 x_2$, so that $x = (x_1, x_2) \in \mathbb{R}^+ \times \mathbb{R}^+$ belongs to class C_1 if $f(x) > 1$. For an input $x = (0.1, 20)$, the gradient of variable x_1 is equal to 20 ($\frac{\partial f}{\partial x_1} = 20$) and is bigger than the gradient of variable x_2 which is equal to 0.1 ($\frac{\partial f}{\partial x_2} = 0.1$). In such a case, if we define the gradient as the importance of variables, then x_1 is way more important than x_2. However, intuitively $x_2 = 20$ is the main reason that $(0.1, 20)$ is classed to C_1 and $x_1 = 0.1$ actually has a negative evidence for class C_1.

Example 2 Second, we consider a linear classifier $f(x_1, x_2) = 10x_1 + x_2$, and $x = (x_1, x_2) \in \mathbb{R} \times \mathbb{R}$ belongs to class C_1 if $f(x) > 0$. Since it is a linear function, the gradient of variable x_1 is always equal to 10 and is always bigger than the gradient of variable x_2 which is equal to 1. If we use gradients as the importance of variables, x_1 should always be more important than x_2 for any input. Considering the input $x = (-0.1, 20)$. This sample should be classified to class C_1 since $f(x) = 10 \times (-0.1) + 20 = 10 > 0$. However, to say that $x_1 = -0.1$ is the main reason why x is classified in class C_1 is somehow counter-intuitive.

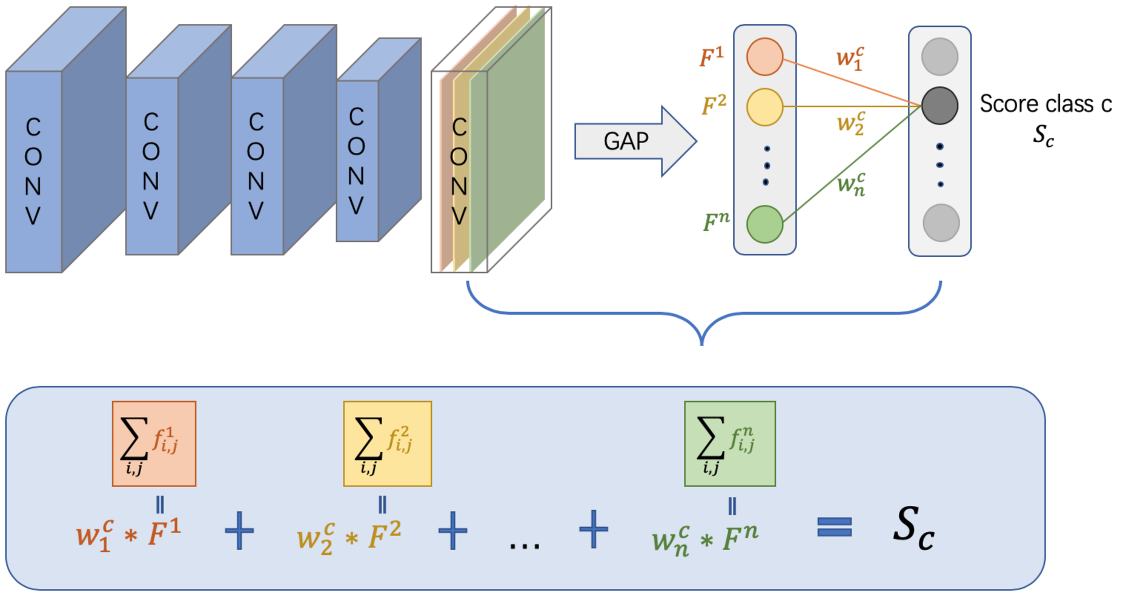

CONV
CONV
CONV
CONV
CONV
GAP
F^1
F^2
F^n
w_1^c
w_2^c
w_n^c
Score class c
S_c
$\sum_{i,j} f_{i,j}^1 = w_1^c * F^1$
$\sum_{i,j} f_{i,j}^2 = w_2^c * F^2$
$\sum_{i,j} f_{i,j}^n = w_n^c * F^n$
$w_1^c * F^1 + w_2^c * F^2 + \ldots + w_n^c * F^n = S_c$

Figure 4.4. – Explanation of the attribution method CAM (class activation maps) (Zhou et al. 2016). CAM allows to write the classification score explicitly on the last layer's features for a CNN, where the last convolution layer is followed by a global average pooling (GAP) layer.

These two simple examples show that the gradient itself is not a good indicator to define the importance of variables. Both the gradients and the values shall be used, which is a premise of our work. As we can observe in these two examples, defining the "contribution" of each variable is not always obvious, especially when variables are entangled with each other like in the Example 1. It is more logical to talk about the "contribution" of each variable if the function is in a summable form:

$$F(x_1, ..., x_n) = \sum_{i=1}^{n} f_i(x_i) \tag{4.1}$$

In this case, the contribution of each variable may be easily defined as the value of $f_i(x_i)$, and these contributions may be then used to build an attribution method. CAM (Zhou et al. 2016) is an attribution method for specific network architectures where the last convolution layer is followed by a global average pooling (GAP) and then a fully connected layer, with no bias as shown in Figure 4.4. Actually, by following these architectural modifications, CNNs are transformed into a summable form on last layer features as described in equation 4.1. If we note $f_{i,j}^k$ the activation of the k-th feature of the last convolutional layer at spatial location (i, j), then the activation after global average pooling is

$$F^k = \sum_{i,j} f^k_{i,j} \tag{4.2}$$

For a given class c, if we write w^c_k the weight corresponding to class c for unit k, then the classification score of class c is

$$\begin{aligned} S_c &= \sum_k w^c_k F_k \\ &= \sum_k \sum_{i,j} w^c_k f^k_{i,j} \\ &= \sum_{i,j} \sum_k w^c_k f^k_{i,j} \end{aligned} \tag{4.3}$$

In this case, it is thus possible to define the importance of the activation of pixel (i, j) in feature map k for predicting class c as $\sum_k w^c_k f^k_{i,j}$. However, this approach can only be used for specific architectures and lacks generality to describe many other popular deep networks.

Grad-CAM (Selvaraju et al. 2017) proposes a generalization of this notion to all kind of CNNs. In CAM, the coefficient w^c_k is defined as the importance of feature map k for predicting class c. Since for an arbitrary CNN, the above formulations (Equation 4.2 and Equation 4.3) do not hold, Grad-CAM uses an approximation of w^c_k by averaging partial derivations of the classification score for class c w.r.t the activation of $f^k_{i,j}$ as

$$\alpha^c_k = \frac{1}{Z} \sum_{i,j} \frac{\partial S_c}{\partial f^k_{i,j}} \tag{4.4}$$

where Z is the sum of activations of the feature map k. However, this approximation considers only the "importance" of different feature maps but ignores the spatial information inside the feature map since the average operation on all pixels in Equation 4.4 loses the spatial information. This leads to a harsh approximation, particularly when one wants to visualize the activations of upstream feature maps.

Based on this critique of gradient-based approaches, we propose a complete formalism to analyze more finely the relationship between the input pixels and the final decision of the network.

4.3.2 Delving Deep into Interpreting Neural Nets with Piece-Wise Affine Representation

In this section, we present our attribution method for all feed-forward deep neural networks with piece-wise linear activation such as the rectifier. In the context of neural network, a unit employing the rectifier is also called a Rectified

Linear Unit (ReLU) (Nair et al. 2010). The rectifier is an activation function defined as the positive part of its argument:

$$f(x) = x^+ = max(0, x) \tag{4.5}$$

where x is the input to a neuron. This activation function was first introduced to a dynamical network by Hahnloser et al. in 2000 with strong biological motivations and mathematical justifications (Hahnloser et al. 2000; Hahnloser et al. 2001). It was demonstrated for the first time in 2011 (Glorot et al. 2011) to enable better training of deeper networks, compared to the widely used activation functions prior to 2011, e.g., the logistic sigmoid (which is inspired by probability theory) and its more practical counterpart, the hyperbolic tangent. Some variants of ReLU such Leaky ReLU (Maas et al. 2013) and Parametric ReLU (K. He et al. 2015) are proposed to train non activated units as showed in Figure 4.5.

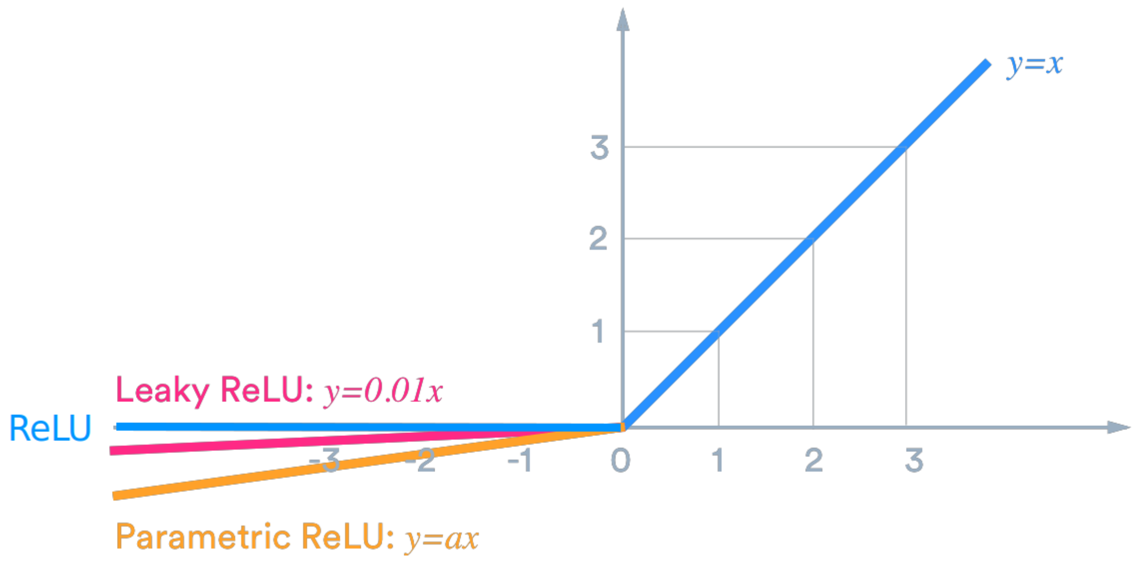


Figure 4.5. – Plots of ReLU (blue) and two variants: Leaky ReLU (red) and Parametric ReLU (orange). These two variants allow a small, positive gradient when the unit is not active. Parametric ReLU transforms the leakage coefficient into a parameter that is learned along with the other parameters of the neural network. These three activations are all piece wise affine fucntions.

ReLUs are widely used in modern networks, mainly because they offer good properties to avoid gradient vanishing while accelerating training process. For example VGG (Simonyan et al. 2015) and ResNet (K. He et al. 2016) all fall under this description. As of 2017, ReLU is the most popular activation function for deep neural networks (Ramachandran et al. 2018). In what follows, we will detail our explanation method for ReLU based feed-forward networks.

Structure of a feed-forward network In what follows, we consider a feed-forward deep neural network F. F can be expressed as a composition of elementary operations of its input X, *i.e.*:

$$F(X) = f_{out} \circ f_{L-1} \circ ... \circ f_1(X) \tag{4.6}$$

where f_l is an operation effected in the layer l. Operations commonly used in CNNs include linear transformation, ReLU, max/average pooling and batch normalization (Ioffe et al. 2015). An interesting remark is that all these operators, during evaluation, are actually piece-wise affine functions. As a consequence, if we ignore the final softmax layer, a ReLU-based network is actually a piece-wise affine function of its inputs since any composition of piece-wise affine functions is still a piece-wise affine function.

In other words, the input space is partitioned into a number of regions on which the network computes an affine function. In order to better illustrate the piece-wise affine nature of ReLU based network, we build a simple feed-forward binary classification network on synthetic 2D data. This network has 3 hidden layers and each layer contains 5 units with ReLU activation. The last layer contains two units which correspond to the output of class c_1 and c_2. After training on randomly generated 2d data, we plot the learned function of the network on Figure 4.6. One can see that the two components of the output are piece-wise affine functions, which means that the network effectively consists in different affine functions defined over different regions of the input space. This illustration gives us a intuition of the decomposition that will be presented in the next section.

Classification score decomposition Let's consider a ReLU-based neural network for a classification problem (the same consideration holds for a regression network). Given a input point $X \in \mathbb{R}^n$, we now explain how to define the contribution of each component x_i of X to the final output. Mathematically, the network defines a piece-wise affine function $F : \mathbb{R}^n \to \mathbb{R}^C$, where n is the input space dimension and C is the total number of classes. Several papers such as (Montufar et al. 2014) have studied the upper bound of the number of linear regions. Since this number is finite and the intersection of linear regions is negligible (subsets with dimension less than n), we can assume that the input X is not on the intersection of two linear regions. Therefore, the point X is located in the interior of a linear region $V(X)$ and the restriction of F to this region is an affine function $F_{|V(X)}$ defined as:

$$\begin{aligned} F_{|V(X)} : V(X) \subset \mathbb{R}^n &\to \mathbb{R}^C \\ Y &\mapsto W_{|V(X)}Y + b_{|V(X)} \end{aligned} \tag{4.7}$$

where $W_{|V(X)} = (w^c_{i|V(X)})_{c,i} \in \mathbb{R}^{C \times n}$ and $b_{|V(X)} = (b^c_{|V(X)})_c \in \mathbb{R}^C$ are constants. We note $F^c_{|V(X)}$ the c-th component of function F which represent the classification score of class c. According to Equation 4.7, $F^c_{|V(X)}$ is defined as :

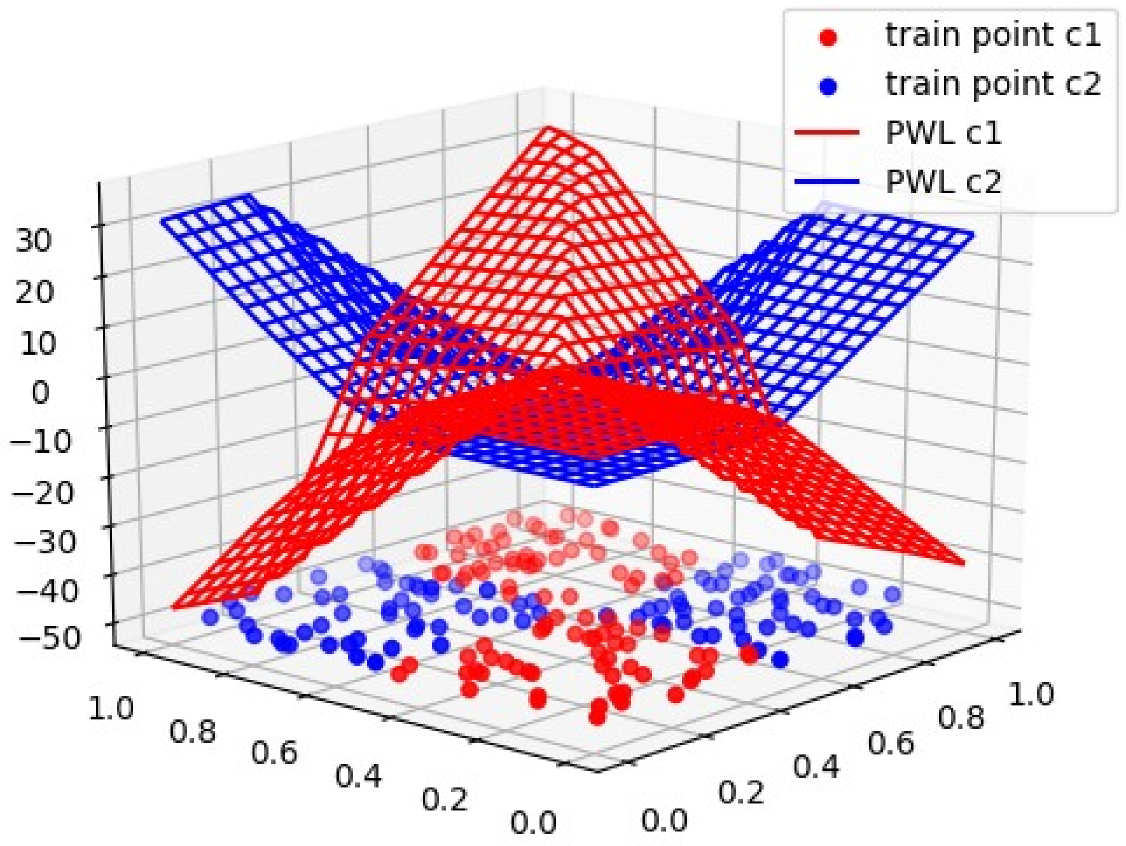


Figure 4.6. – Visualization of a simple neural network on a toy dataset of 2D points. The network has two outputs and three hidden layers with 5 rectified linear units each. PWL $c1$ (red) and PWL $c2$ (blue) are the plots of each output function. Both are piece-wise affine functions.

$$
\begin{aligned}
F^c_{|V(X)} : V(X) \subset \mathbb{R}^n &\to \mathbb{R} \\
Y &\mapsto W^c_{|V(X)} Y + b^c_{|V(X)} \\
&= \sum_i w^c_{i|V(X)} y_i + b^c_{|V(X)}
\end{aligned}
\tag{4.8}
$$

In this case, it makes sense to define the contribution of each variable as we describe in the equation 4.1. The contribution of variable x_i to the classification score $F^c_{|V(X)}$ is therefore

$$
S^c_{x_i} = w^c_{i|V(X)} x_i \tag{4.9}
$$

$S^c_{x_i} \geq 0$ implies that variable x_i has a positive contribution to class c while $S^c_{x_i} < 0$ means a negative contribution. Apart from the contributions of each variable, there is a bias term $b^c_{|V(X)}$ which define a "prior knowledge" of all points in $V(X)$. Imaging a small perturbation δX of X such that $X + \delta X$ still remain in the region $V(X)$. The importance of this perturbation can be measured by the ratio

$$\begin{aligned} \frac{\Delta F^c_{|V(X)}}{F^c_{|V(X)}(X)} &= \frac{F^c_{|V(X)}(X+\delta X)}{F^c_{|V(X)}(X)} \\ &= \frac{\sum_i w^c_{i|V(X)} \delta X_i}{\sum_i w^c_{i|V(X)} X_i + b^c_{|V(X)}} \end{aligned} \tag{4.10}$$

The larger the bias is, the lower the ratio and therefore the more robust the classification score will be w.r.t a small perturbation δX. Note at this point that $b^c_{|V(X)}$ is a constant that depends on $V(X)$, thus on the whole input X.

To sum it up, for any ReLU based feed-forward network and for any input X, the classification score of class c can be decomposed into contributions of each variable and a bias term. Moreover, this decomposition is exact and unique due to the piece-wise affine nature of the network.

Weights and bias computation We show now how to efficiently obtain the exact value of each variable's contribution for a given input X. According to equation 4.9, to compute the contribution of variable x_i for class c is back to the computation of $W_{|V(X)}$ and $b_{|V(X)}$.

Since for an affine function, the coefficient in front of x is equal to the gradient, $W^c_{|V(X)}$ can be easily calculated by a back-propagation:

$$W^c_{|V(X)} = \nabla F^c_{|V(X)}(X) \tag{4.11}$$

and $b^c_{|V(X)}$ can be obtained by:

$$b^c_{|V(X)} = F^c_{|V(X)}(X) - W^c_{|V(X)} X \tag{4.12}$$

Then, the contribution of the variable x_i to class c can be be easily obtained as described in equation 4.9:

$$S^c(x_i) = w^c_{i|V(X)} x_i \tag{4.13}$$

Thus, we show that the importance of variable x_i is $w^c_{i|V(X)} x_i$ rather than the gradient $w^c_{i|V(X)}$ as we discuss in Section 4.3.1. In other words, a pixel is important if both its value and gradient are large, which echoes the work of (Fong et al. 2017). Also note that this decomposition can be applied at the level of any feature map within the network because of its compositional structure. (eg. $f_{out} \circ f_{L-1} \circ ... \circ f_l \circ f_{l-1} \circ ... \circ f_1$ is also a piece-wise affine function of $f_{l-1} \circ ... \circ f_1$). CAM (Zhou et al. 2016) is actually a special case of our method applied on features of the last convolution layer.

This decomposition is exact for any ReLU based feed-forward networks including those popular CNNs. In the context of computer vision, X is an image and each component x_i is a pixel. By applying our method for CNNs, we can therefore highlight the most important part of each object presented in the image. In the upcoming section, we show the interest of this approach for visual explanation through experiments.

4.3.3 Experiment for Visual Explanation

In this section, we show the interest of our attribution method VGatt for visual explanation as well as a comparison with existing methods, in particular Grad-CAM (Selvaraju et al. 2017), a widely used method. We prove the effectiveness of VGatt through three experiments:

- **Exp1**: Visualization of VGatt with popular CNN architectures.
- **Exp2**: Visual comparison with gradient-based methods.
- **Exp3**: Visual comparison with Grad-CAM through different layers.

Exp1 As described above, our method is available for all ReLU based CNNs including most of state of the art networks. In this experiment, we present the visual explanation results for 4 popular networks: VGG19 (Simonyan et al. 2015), Resnet152 (K. He et al. 2016), Densenet161 (G. Huang et al. 2017) and Wide-ResNet101 (Zagoruyko et al. 2016), all pre-trained on ImageNet dataset. For this purpose, we randomly selected four images that do not belong to the training set and present the attribution map for a given class, as showed in Figure 4.7. Although these networks have different architectures, in particular the VGG19 which does not contain skip connections and ends up with a fully connected layer, the visual explanation defined by our method can focus on the most relevant regions as we expected. Since VGatt is an exact decomposition of the classification score, this result shows that the predictions made by these CNNs are indeed based on some of the most discriminate characteristics of the objects.

An interesting observation is that the VGG19 seems to provide poorer visual results than other networks: less object-centered and also focuses on some other areas such as image corners. There are two possible reasons for this. First, the 2d features of the VGG19 are flattened and followed by a fully connected layer. As a result, the different locations have a different weighting, unlike the other three networks, which all end with an global average pooling layer. As can be seen in Figure 4.7, the features of the four corners seem to play an important role in all these examples. Another reason could be that VGG19 contains dropout layers, which randomly drop features during training process. This might encourage the network to rely on more regions, that are more spread out throughout the

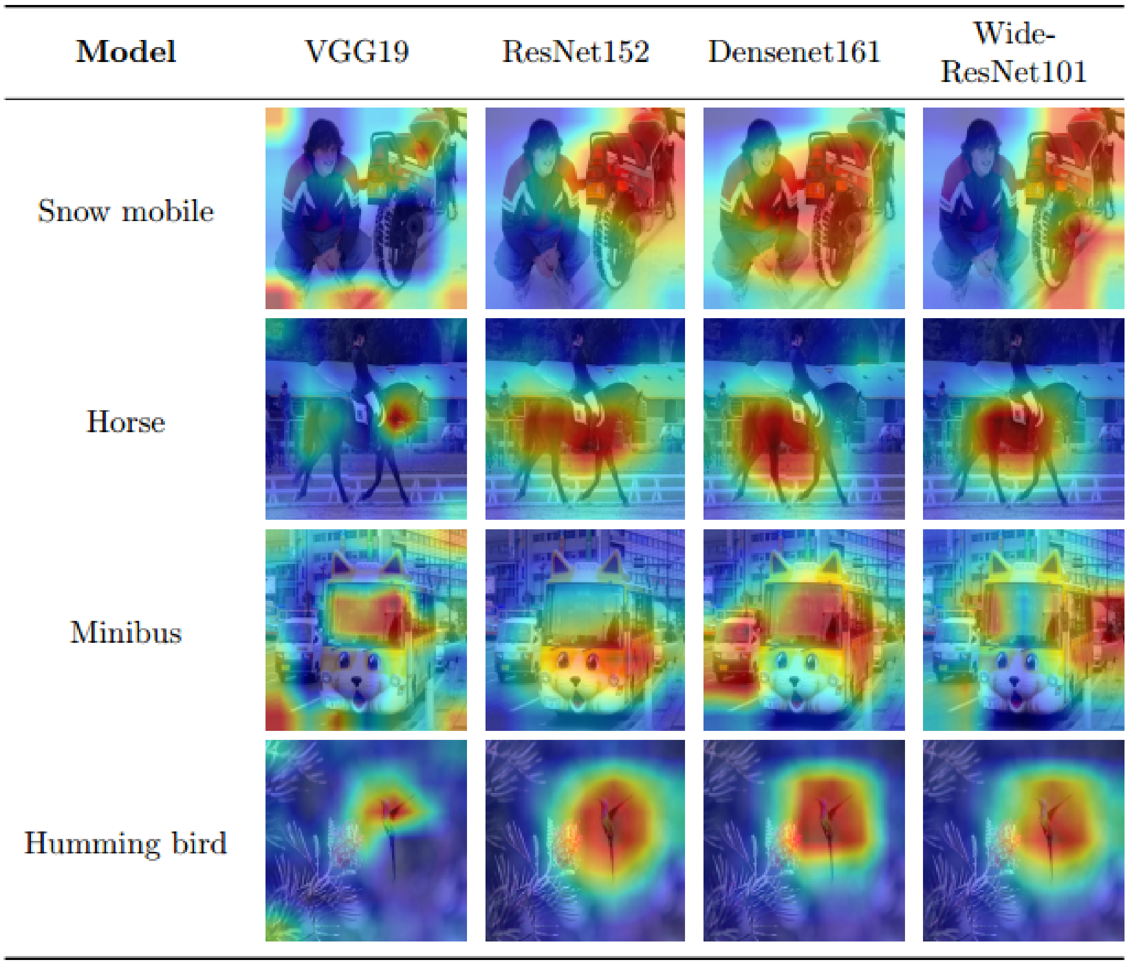


Figure 4.7. – Visualizations of VGatt applied on four popular CNNs: VGG19, ResNet152, Densenet161 and Wide-ResNet101. Although different networks have different visual explanation, they all of them more or less focus on the most relevant areas. Note that red regions corresponds to high score for the given class.

image when making decisions (VGG19 usually provides a sparser explanation map compared with other networks in Figure 4.7). Through this example, we have also shown that our method can be used not only for visual explanation, but also for understanding how networks work. Other applications will be detailed in Appendix A.

Exp2 In this experiment, we present the comparison between our method and three gradient-based methods: Deconvnet (Matthew D Zeiler et al. 2014), Guided Backpropagation (Springenberg et al. 2015) and Vanilla Backpropogation (Simonyan et al. 2014). We have explained through two examples that why the gradient is not a good indicator of the importance in Section 4.3.1. To validate our

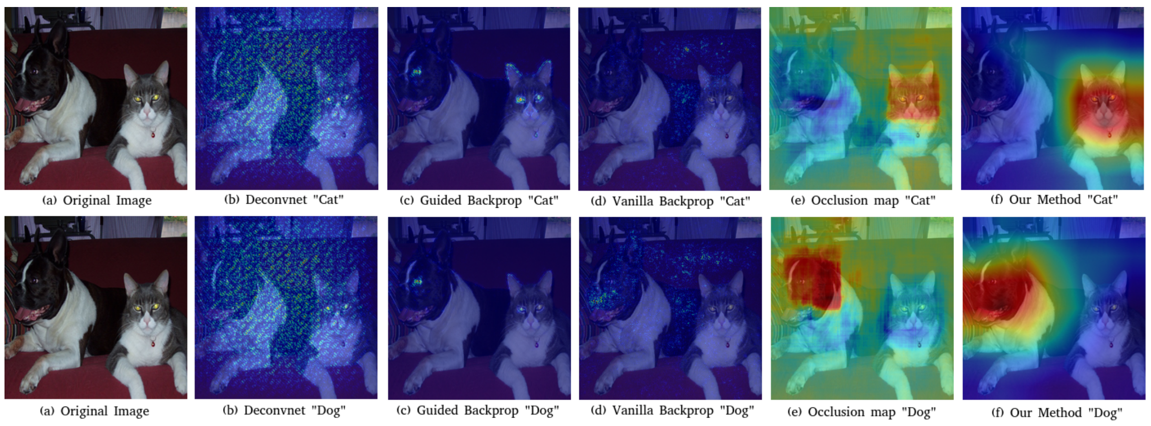


Figure 4.8. – Comparison between VGatt and gradient-based methods for visual explanation. (a) Original image with a cat and a dog. (b-f) Visual explanation of classes "Cat" and "Dog" according to various methods for ResNet152. (b1,b2): DeconvNet, (c1,c2): Guided Backpropagation, (d1-d2): Vanilla Backpropagation, (e1,e2): Occlusion, (f1,f2): Ours. Gradient-based methods (b-d) provide class discriminative visualizations. Our approach yields very similar results to the occlusion map, while is much cheaper to calculate. Note that red regions corresponds to high score for the given class.

argument, we choose an ImageNet pre-trained ResNet152 and an example image that contains a cat and a dog. We show the visualization of the most discriminative parts defined by gradient-based methods compared with our method in Figure 4.8. As can be observed, gradient-based methods are not class discriminative and provide similar visualizations (b-d) for both "Cat" and "Dog". On the contrary, our method concentrates precisely on the corresponding part (head of the cat and the dog). It is interesting to note that the results produced by our method is very similar to the occlusion sensitivity map, where we measure the difference in CNN scores when patches of the input image are masked (Matthew D Zeiler et al. 2014). To produce an occlusion sensitivity map, we need apply forward propagation through the CNN as many times as the total number of patches to be occluded, whereas our method requires only one forward and one backward pass, which is much cheaper to calculate.

Exp3 In the last experiment, we compare VGatt with a popular visual explanation method Grad-CAM (Selvaraju et al. 2017). As described above, Grad-CAM was proposed to generalize CAM(Zhou et al. 2016) since CAM can only be used in networks where the last layer is a global average pooling layer. Grad-CAM can be thus used at any layers as our method. Grad-CAM is an approximation, whereas our method can determine the exact attribution. In this experiment, we apply both our method and Grad-CAM on a ImageNet pre-trained VGG-19. This network consists of 5 blocks of 2 to 4 convolutional layers that will be denoted as conv1,...,conv5 in the following. And we compare the visual explanation at these 5 levels. Since the last layer of VGG19 is not a global average pooling layer, visual explanations obtained by these two methods should be different in all these levels. Figure Figure 4.9 shows visualization of pixel-wise relevance of pixels from features maps at different levels of VGG-19 network, relatively to the final classification (class "cauliflower" for the two upper rows, and class "bull mastiff" for the two lower rows). For both our approach and Grad-CAM, negative contributions are discarded for visualization. For the last layers (conv5), Grad-CAM and our method outputs similar results. But for lower layers (conv4 and below), Grad-CAM is not precise as it involves spatial averaging of the gradients as described in equation 4.4, which leads to irrelevant approximations. By contrast, our method is able to output more precise results, as the relevant regions are concentrated on the interesting objects in both cases.

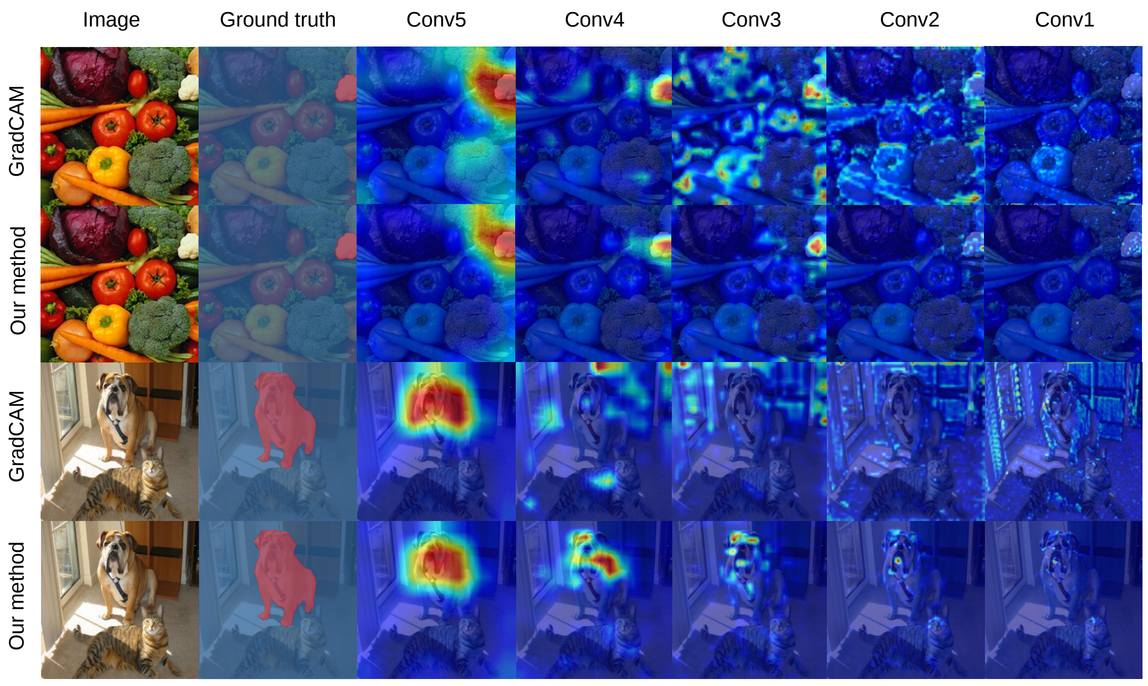


Figure 4.9. – Visualization of our method on two held-out images and VGG-19 network, and comparison with Grad-CAM. The closer to the input layer (Conv1 being the closest), the more spread out the pixels highlighted by Grad-CAM are. Our method, however, is more precise as pixels belonging to the objects are relevant to the classification score.

However, as we observe in Figure 4.9, the visualization of features at low-level provide highly sparse maps, especially for Conv1 and Conv2. Only few pixels are indicated as relevant. There may be two reasons for this problem. At first, features of low level layers have larger spatial resolution. For example, one feature in Conv1 represents for one pixels while one feature in Conv5 represents for 32 pixels. The visualization provided by deeper layers are therefore more dense. Secondly, the score decomposition formula Equation 4.7 has a bias term. A large bias term signifies that the total contribution of all pixels are small. Since we only visualize the pixels that contribute positively in Figure 4.9, large bias term can lead to small contribution of pixels and thus a sparse visual explanation map. An in-depth analysis of this issue can be found in Appendix A.

Let's now explain how to exploit attribution methods in the WSL framework for semantic segmentation.

4.4 Weakly Supervised Semantic Segmentation Using Attribution

In this section, we leverage weakly supervised semantic segmentation strategy by using attribution methods to define "seeds", as described in Section 4.2.

4.4.1 Experiment Setup

We conduct our experiments on the Pascal VOC 2012 (Everingham et al. 2015) dataset since this dataset provides both multi-label classification and semantic segmentation tasks. It means that for each image in Pascal VOC 2012, we have both its image-level labels and its segmentation mask. We only use their image-level labels in the weakly supervised setting and then validate the trained segmentation models with the ground truth segmentation masks. This dataset contains 20 object classes for multi-label classification where each image usually has about $2-3$ labels. We use the augmented version the dataset (Hariharan et al. 2011), which contains 10 582 images in train set and 1 449 images in validation set.

Weakly supervised semantic segmentation pipeline The weakly supervised semantic segmentation strategy we used in these experiments consists of three steps as shown in Figure 4.10:

1. Train a modified VGG19 (Simonyan et al. 2015) for multi-label classification problem on Pascal VOC 2012.
2. Apply VGatt to define "Seeds". We also employ Grad-CAM as the baseline attribution method.
3. Use obtained "Seeds" as pseudo labels to train Deeplab-v2 (L.-C. Chen et al. 2016) on Pascal VOC 2012 segmentation task.

In step 1, we modify an ImageNet pretrained VGG-19 for multi-label classification task. In order to increase the feature resolution, we omit two last pooling

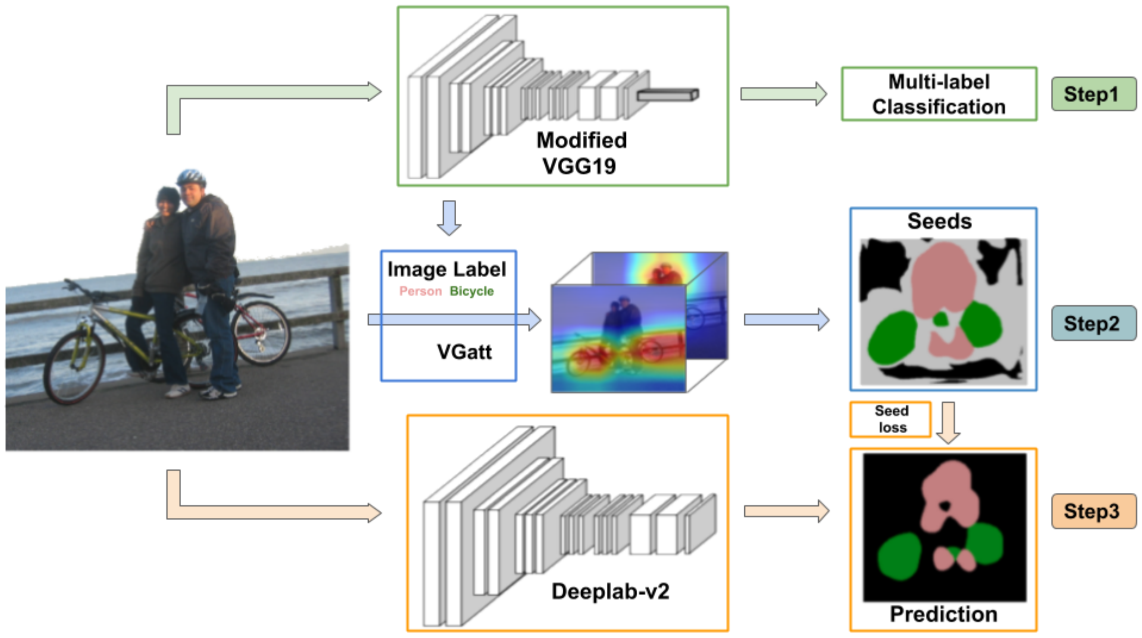


Figure 4.10. – Overview of our WSL semantic segmentation pipeline. We first train a modified VGG-19 for multi-label classification problem. Then, we apply VGatt on the obtained network to define seeds. Finally, we use the produced seeds to train the Deeplab-v2 model for semantic segmentation.

layers and replace fully connected layers with randomly initialized convolutional layers, which have 1024 output channels and kernel size of 3. The output of the last convolutional layer is followed by a global average pooling layer and then by a 1×1 convolutional layer with 20 outputs (the number of foreground semantic classes in Pascal VOC). The modified network is then fine tuned on Pascal VOC 2012 *trainaug* set with multi-label logistic loss. This network is then used to provide segmentation seeds with attribution methods (step 2), which will be detailed below. Finally, we train the Deeplab-v2 segmentation model with the obtained seeds in step 3.

Seed strategy Once the multi-label classification network is fine-tuned, we use the following strategy to define seeds:

1. For a given image I, we get its ground truth labels $L \in C$.
2. For each ground truth label l, we compute the attribution map of this class. We threshold the attribution map by $k\%$ of its maximum value to define the corresponding seeds.
3. If a pixel is a seed for more than one class, the class with the smallest seed area will be finally assigned.

4. Pixels that have negative contributions for all ground truth classes are defined as "background".
5. Other pixels are considered as "unknown".

k is the hyperparameter that we need to determinate. Intuitively, a large value of k means that only the pixels that contribute the most to the classification are considered as seeds. The resulting seeds may be more accurate but only occupy a small portion of the image. On the contrary, seeds obtained with a small value of k occupy a large part of the image but are less accurate. It is worth noting that this strategy can be used at any layer of the network.

4.4.2 Quantitative Results

In this section, we present the quantitative results of the WSL strategy for semantic segmentation. As we described above, the generation of seeds depends on both the hyper-parameter k and the layer on which we apply our method. Thus, we train Deeplab-v2 on the seeds obtained with different k and different layers, and employ the mean Intersection over Union (mIoU) score (see Section 2.2.3) on VOC 2012 *val* set to evaluate the performance of each model. Table 4.1 shows the results of Deeplab-v2 trained with different seeds. The best WSL model is trained on seeds generated at last layer with $k = 0.1$, and it reaches a mIoU of 39.6%. This means that, with only image-level labeled images, our method achieves 54% of the performance of the fully supervised Deeplab-v2, which is 73% on mIoU.

To better understand the limitations of these seeds, we use the ground truth masks to modify the best set of seeds (last layer with $k = 0.1$) in two ways: **Only True** set only contains the correct seeds obtained with our method and **All Corr.** set that corrects all the seeds. Thus, all seeds in these two sets are correct, and the difference is the number of pixels that are considered as seeds: **Only True** considers 57% of pixels as seed while **All Corr.** covers 73% of the total pixels. It is important to note that these two experiments are not in the context of WSL since the ground truth masks are used to correct the seeds, their purpose is to analyze the limitation of the generated seeds. Surprisingly, Deeplab-v2 achieves very high mIoU score (66.7% for **Only True** and 73% with **All Corr.**) when trained on these two sets of seeds. We note that the result of Deeplab-v2 trained in fully supervised setting is 73%, which equals to result of **All Corr.**. This means that whole-image labeling is not necessary: with 70% of well-located labeled pixels, segmentation models can achieve performance similar to fully supervised case. This further demonstrates the potential of the WSL methods.

To further validate the effectiveness of the use of VGatt, we employ Grad-CAM in the same way as our approach to define seeds, and compare the results of Deeplab-v2 trained on both seeds. As in these experiments, the classification network used for defining seeds ends with a global average pooling layer, Grad-

Seeds	Conv4		Conv5		Last Layer				Corrected Seeds		Fully Supervised
	k=0.1	k=0.2	k=0.1	k=0.2	k=0.1	k=0.15	k=0.2	k=0.25	Only True	All Corr.	
Bg	0.63	0.533	0.768	0.784	0.766	0.774	0.777	0.779	0.904	0.931	0.931
aero	0.283	0.231	0.495	0.442	0.386	0.407	0.416	0.423	0.719	0.857	0.858
bike	0.196	0.154	0.307	0.273	0.256	0.262	0.267	0.267	0.362	0.391	0.394
bird	0.199	0.168	0.358	0.446	0.382	0.401	0.413	0.423	0.784	0.835	0.836
boat	0.193	0.178	0.279	0.249	0.245	0.255	0.262	0.267	0.549	0.661	0.647
bottle	0.401	0.408	0.404	0.311	0.38	0.357	0.335	0.313	0.719	0.789	0.794
bus	0.477	0.459	0.487	0.449	0.527	0.506	0.478	0.449	0.848	0.909	0.911
car	0.347	0.347	0.405	0.381	0.457	0.44	0.421	0.398	0.785	0.83	0.829
cat	0.25	0.273	0.411	0.342	0.425	0.378	0.339	0.306	0.845	0.898	0.896
chair	0.122	0.131	0.149	0.175	0.16	0.17	0.176	0.178	0.319	0.342	0.331
cow	0.265	0.358	0.484	0.293	0.401	0.365	0.33	0.292	0.723	0.782	0.786
table	0.203	0.209	0.23	0.217	0.267	0.248	0.224	0.199	0.427	0.543	0.535
dog	0.233	0.265	0.314	0.321	0.433	0.386	0.345	0.308	0.758	0.839	0.839
horse	0.274	0.278	0.469	0.348	0.396	0.375	0.348	0.319	0.717	0.797	0.809
mbike	0.435	0.415	0.486	0.429	0.507	0.487	0.465	0.436	0.751	0.79	0.797
person	0.358	0.318	0.439	0.498	0.538	0.523	0.505	0.482	0.776	0.832	0.831
plant	0.172	0.155	0.221	0.32	0.356	0.351	0.346	0.342	0.499	0.516	0.522
sheep	0.302	0.243	0.549	0.423	0.48	0.46	0.449	0.433	0.735	0.803	0.799
sofa	0.21	0.199	0.219	0.218	0.237	0.234	0.229	0.22	0.394	0.441	0.441
train	0.392	0.353	0.365	0.314	0.352	0.334	0.311	0.291	0.729	0.825	0.822
tv	0.179	0.195	0.301	0.319	0.365	0.366	0.36	0.347	0.676	0.713	0.712
mean	0.291	0.279	0.388	0.359	**0.396**	0.385	0.371	0.356	0.667	0.73	**0.73**

Table 4.1. – The mIoU score on VOC 2012 *val* set of Deeplab v2 trained on different seeds.

CAM is equivalent to our method at the last layer. We thus only compare the best results for seeds generated at Conv4 and Conv5. Results are reported in Table 4.2. Seeds generated by VGatt achieve better results than Grad-CAM with an improvement of 1.9% on Conv4 and 2.6% on Conv5. These results show that the our method provides better seeds on both Conv4 and Conv5.

Layer	Seed Method	mIoU
Conv4	Grad-CAM	0.273
	VGatt (ours)	**0.291**
Conv5	Grad-CAM	0.362
	VGatt (ours)	**0.388**

Table 4.2. – Comparison of the results of Deeplab v2 trained on seeds defined by our method and Grad-CAM. (VOC 2012 *val* set)

To complete the comparison with Grad-CAM, we show some illustrations of seeds defined by our method and Grad-CAM. We take the first image of the *trainaug* set as an example and present seeds defined by both methods at Conv4 and Conv5 with different *k* in Figure 4.11. This image contains objects of class "person" (pink) and "airplane" (red), and the black and white regions represent "background" and "unknown" respectively. It is obvious that Grad-CAM has difficulty in providing good localization cues of objects: it assigns the "person" label to huge area, even for pixels of the airplane. On the contrary, VGatt produces much more accurate seeds than Grad-CAM. In addition to the foreground classes, seeds obtained by our approach also have a smaller "unknown" region (white area) and a larger "background" region (black area). These illustrations therefore validate the qualitative improvement obtained with our attribution method.

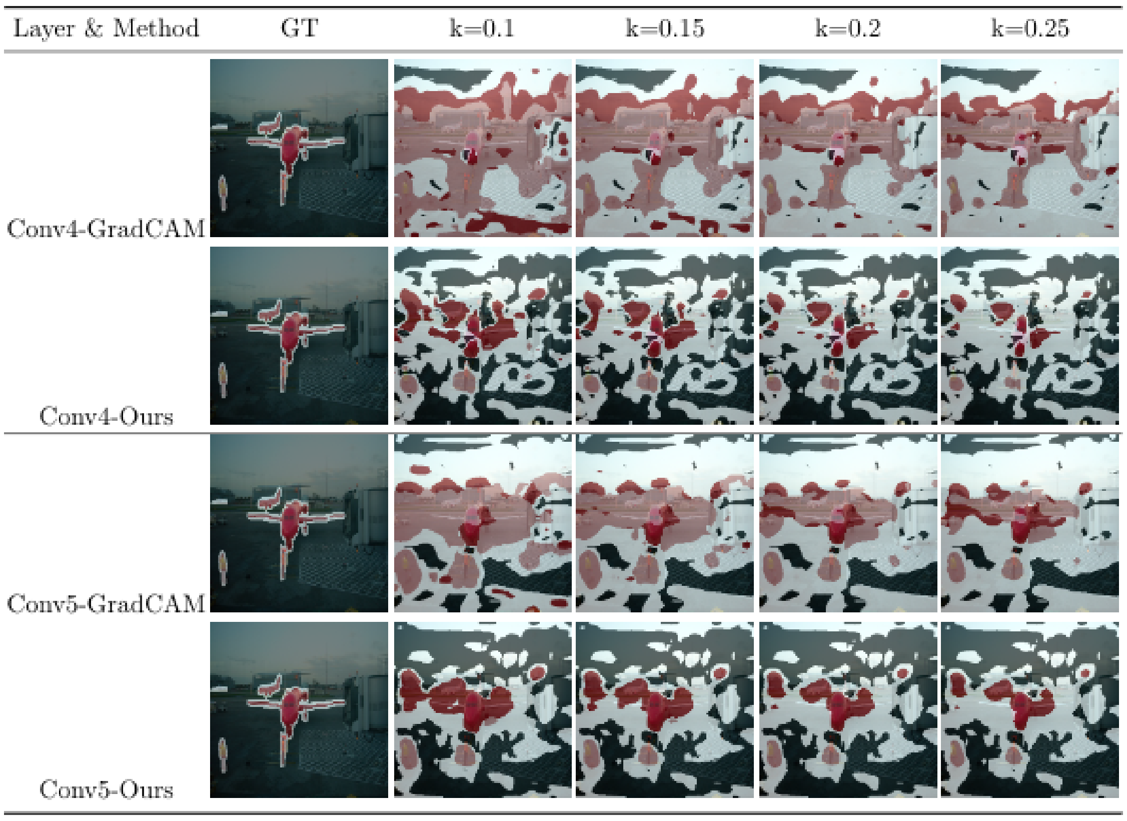


Figure 4.11. – Comparison of seeds defined by our method and Grad-CAM at Conv4 and Conv5. The red, pink, black and white regions represent "airplane", "person", "background" and "unknown" respectively.

4.4.3 Ablation Study

In this section, we provide insight into the quality of seeds depending on the hyper parameter k and the layer on which we apply our method. We take the same image as example.

Qualitative Figure 4.12 shows the different seeds obtained with different k and different layers. As can be seen, although accurate information on the contours of objects is lost, the seeds defined by our method provide correct information on the location of objects at all layers. It is interesting to note that

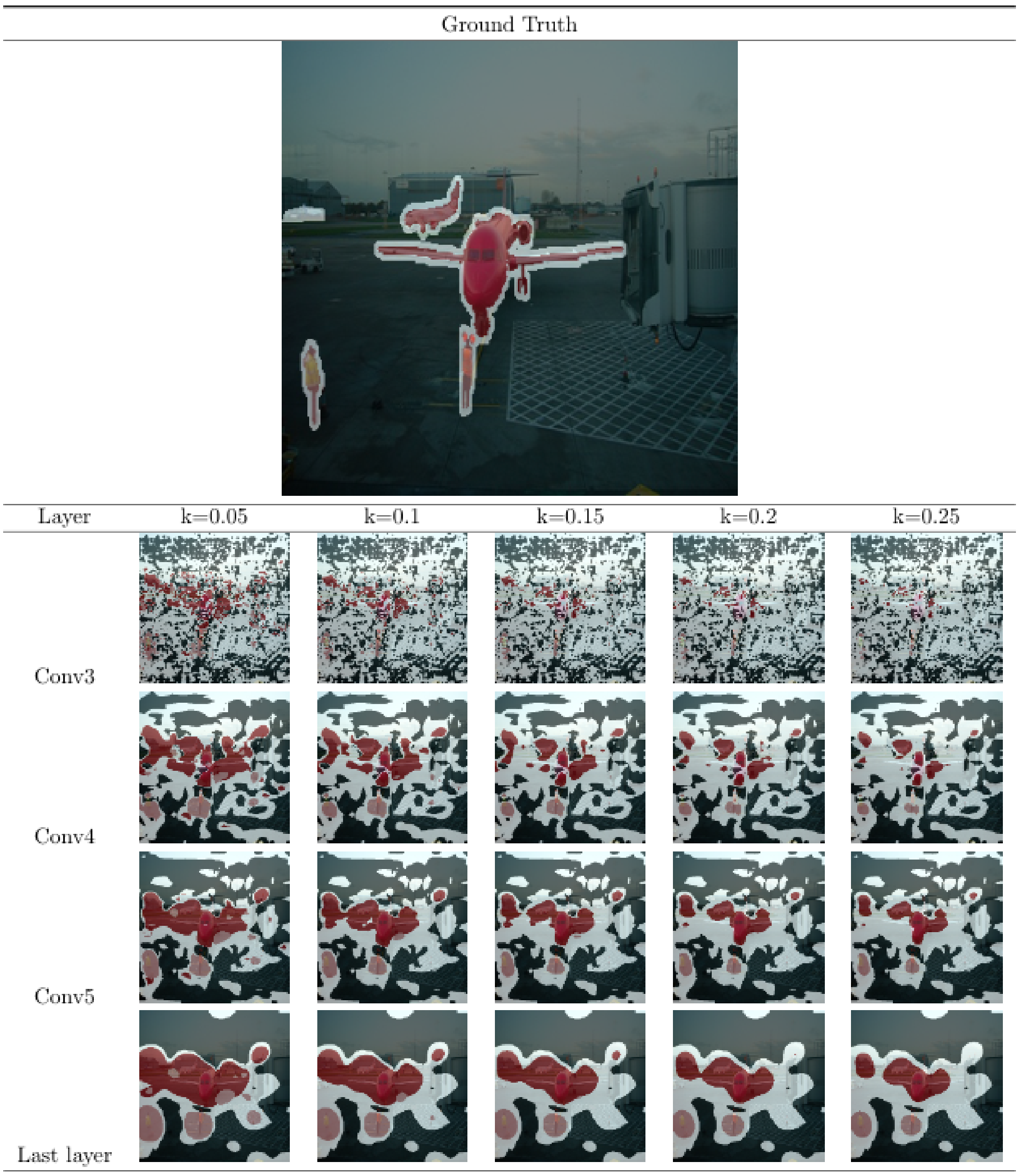


Figure 4.12. – Example of visualization of "Seeds" obtained by our method with different *k* and at different layer. The red, pink, black and white regions represent "airplane", "person", "background" and "unknown" respectively.

seeds obtained at shallow layers (Conv3 and Conv4) are much sparser and define less continuous regions than that obtained at deeper layers (Conv5 and Last layer). This phenomenon is consistent with the visualization results showed in Figure 4.9, where deeper features provide denser and better localization information of objects.

Quantitative In order to better analyze the overall quality of the seeds generated by our method, we compare the seeds with ground truth masks under two measures: **pixel accuracy** and **area ratio**. Pixel accuracy measures the average precision of seeds and the area ratio, which is the ratio between the total number of seeds and the total number of pixels in an image, measures how many pixels are considered seeds. Both evaluations are performed on the entire *trainaug* set, and the higher the score of both measures, the better the seeds. It worth noting that these measurements are only evaluated on pixels belonging to "seed" or "background", but not on "unknown" pixels. For example, if our method defines 70 pixels as seeds for an image of 100 pixels (thus 30 pixels "unknown"), the area ratio is then 70% and the pixel accuracy is calculated only on these 70 pixels.

Name	Pixel Accuracy	Area ratio
Conv4, k=0.1	74.5%	60%
Conv4, k=0.2	76.9%	53.2%
Conv5, k=0.1	79.4%	67.1%
Conv5, k=0.2	81.6%	60%
last layer, k=0.1	78.5%	**73.2%**
last layer, k=0.15	80.6%	68.4%
last layer, k=0.2	82.1%	64.8%
last layer, k=0.25	**83.2%**	61.9%

Table 4.3. – Evaluations of seeds obtained with different *k* and at different layer. Both evaluations are performed on seed pixels ("unknown" pixels are omitted).

The results are shown in Table 4.3. Seeds obtained with deeper layers generally provide higher scores for both measurements. For example, with $k = 0.1$, seeds generated at last layer occupy 73.2% of the image with a pixel accuracy of 78.5%, while seeds generated at Conv4 only occupy 60% of the image with an accuracy of 74.5%. Moreover, comparing the scores for different *k* when the layer is fixed also validates our intuition for this hyper parameter, so that a large value of *k* provides larger (area ratio) but less accurate (pixel precision) seeds. These results suggest that, in practice, it is preferable to apply attribution on the last layers of the classification CNN to obtain better seeds.

Relationship with mIoU Finally, we examine the relationship between seeds and the resulting segmentation performance. As we described above, the area ratio represents the portion of pixels that have been considered as seeds and the pixel accuracy measures how many seeds have been assigned with the correct label.

Actually, there are three types of pixels in the seeds: correct seeds, wrong seeds and "unknowns", and the sum of these three types of seeds is equal to the total number of pixels. Thus, the area ratio is the sum of correct seeds and wrong seeds and the pixel accuracy is the ratio between correct seeds and total number of seeds. The number of correct seeds can be obtained simply by the product of these two measurements. Intuitively, the larger the correct seeds and the total seeds, the better the segmentation results. To this end, we plot the mIoU as a function of correct seeds and area ratio in Figure 4.13 (the size of the dot in the figure is proportional to the mIoU score). As it can be seen, the mIoU score in the top/right region of the plot are bigger than those in the bottom/left region, which validates our intuition.

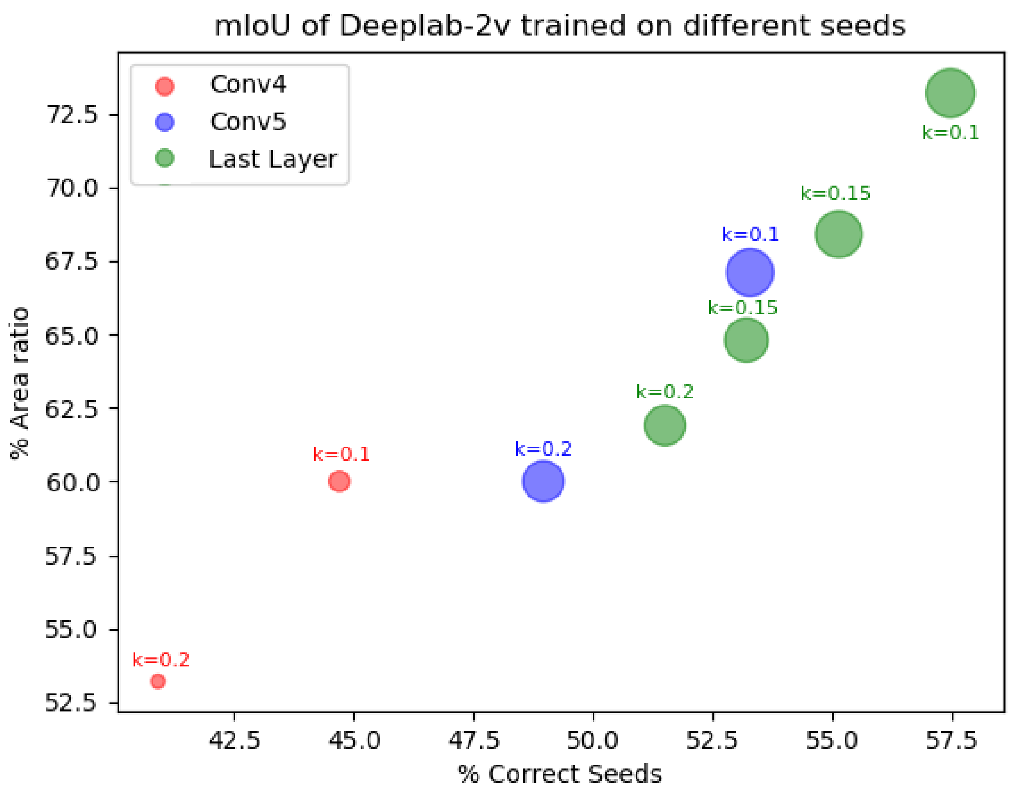


Figure 4.13. – Bubble plot of WSL segmentation results (mIoU) according to the "area ratio" and the "correct seeds". The size of the dot is proportional to the value of mIoU.

4.4.4 Discussion

We have demonstrated in this WSL segmentation context how to successfully use our attribution method into a simple seed generation strategy and obtain promising results in the context of WSL segmentation. Through a qualitative and quantitative evaluation, we have shown that the seeds defined by our method provide relatively good localization cues. However, there are two main limitations of these naive strategies. First, the quality of the seeds depends largely on the network that is used for classification. If the prediction of the classification network is based on only a small part of the object (which is generally the case), seeds defined with this network will have a low area ratio. As shown in Figure 4.14, pixels assigned as seeds are mostly concentrated on the bird's head since this small portion of the image is sufficient for the network to make good predictions. Secondly, the contour information of the object is completely lost in the seeds, and lots of pixels outside the contours are also assigned as seeds. This issue results in low pixel accuracy of the seeds. For example, there are only two coarse regions defined as seeds in Figure 4.14. Many of the pixels outside the bird's head are also considered as seeds, while some pixels on the beak are missed.

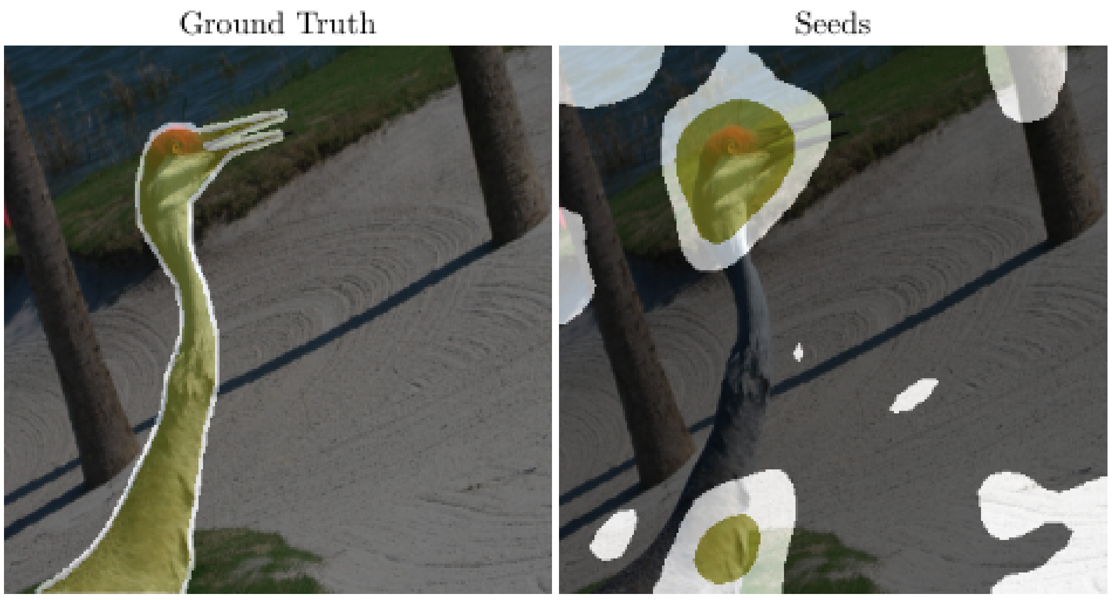


Figure 4.14. – An example where only the most discriminating part is assigned as a seed. The yellow, black and white regions represent "bird", "background" and "unknown" respectively.

To address these two issues, one possibility is to incorporate low-level information such as intensity or color of pixels into to seed generation strategy. Although it seems impossible to obtain the semantic edges of objects with only image-level labels, the discontinuities of pixels provide an indication of contours. Extending the most accurate seed regions under pixel continuity constraints, we could theoretically obtain larger seeds without loss of precision.

4.5 Conclusion

In this chapter, we discussed WSL strategies for semantic segmentation, addressing the lack of fully annotated data. We identified a particular interesting setting of training segmentation models only with image-level labeled data. In this context, we have proposed a new attribution method VGatt to better extract spatial information from pre-trained CNNs only with image-level labels. We showed that ReLU-based CNNs are piece-wise affine functions of its input. Given an image, this function can be calculated explicitly through gradient computation, providing a pixel-wise contribution to the classification score, as well as a global bias term. Unlike other intuition-based approaches, we provide a mathematical proof of our method. Our method is therefore more accurate and reliable. The similar idea is recently explored in (Srinivas et al. 2019), which proves the validity of our approach. By incorporating our method into WSL scheme for segmentation, we showed that VGatt can provide better seeds and improve segmentation results on Pascal VOC than the popular attribution method Grad-CAM.

Although our method can generate better seeds for WSL segmentation, they are still very coarse and inaccurate. To further solve the lack of data in semantic segmentation, a process of seeds cleaning or seeds extension can be considered. For example, to use another classification network to generate seed, to extent seeds based on super-pixel algorithms or to incorporate object priors such as size and shape of objects. Given the ease of collecting weak annotations and the growing demand of semantic segmentation applications, we believe that WSL segmentation is a promising approach and further works are imperative.

Finally, since VGatt describes the exact function of ReLU based CNNs, it can be used for many other problems as well. Through side applications in Appendix A, we show that our method can be used for visual explanation, network diagnosis as well as generating adversarial examples. Other applications such as explaining decisions made by deep networks in language or visual question answering models are also worth trying.

Chapter 5

CONCLUSION

Contents

We first summarize the contributions that we propose in this thesis before discussing interesting directions for future work.

5.1 Summary of Contributions

In this thesis, we tackle the challenging problem of semantic segmentation. We took an approach based on Deep Learning (DL), and identified several challenges and limitations of classical models. We placed a particular focus on the loss function and the lack of fully annotated data for optimization process. In response to these two issues, our contributions can be organized in two points detailed below.

Semantic-edge-aware loss for segmentation The loss function is a fundamental element of Machine Learning (ML) methods, and a well-constructed loss can lead models to produce results that are more consistent with the task. In the context of semantic segmentation, it is especially important to build an appropriate loss, since the predictions of each pixel should have strong dependencies. However, the commonly used Per Pixel Cross Entropy (PPCE) loss completely ignores these dependencies.

To resolve this issue, we propose in Chapter 3, a SEMantic EDge Aware (SEMEDA) loss to better constraint the dependency of pixel-level predictions. By using the semantic edge information, SEMEDA loss imposes the prediction of pixels inside a given object to be uniform with the surrounding pixels, and imposes the prediction of pixels at the semantic contour of objects to contain the contour information as well as the semantic classes of neighboring objects. SEMEDA loss is defined using an additional network, whose parameters are fixed during the training. This learning strategy does not require any additional annotation and only adds negligible computational overhead, and can thus be straightforwardly combined with all popular segmentation networks. Through thorough experiments, we demonstrate that SEMEDA loss consistently improves segmentation results, regardless of the segmentation backbones, pre-training strategies, hyperparameters settings, datasets and evaluation metrics. As a result, SEMEDA loss shows the importance of the loss function for semantic segmentation, and provides an indication of how to constrain the dependency of predictions at the pixel level.

Weakly supervised segmentation with attribution method While SEMEDA addresses the loss function for semantic segmentation, in Chapter 4, we tackle another major challenge of the DL-based methods: the lack of fully annotated data. A particularly interesting approach to alleviate the demand for fully annotated data is to train the segmentation model using image-level labeled data, as it is easy to collect and still provides semantic information. The key process in this approach is to attribute the image-level labels to the relevant pixels, as segmentation models are trained with pixel-level annotations.

To this end, we propose VGatt, a new attribution method that decomposes the output score of a Convolutional Neural Network (CNN) into contributions of each pixel. With a highly accurate classification CNN, VGatt allows us to identify the most relevant pixels for the ground truth labels considered by this CNN. Unlike other intuition-driven approaches, VGatt is based on the mathematical properties of Rectified Linear Unit (ReLU)-based CNNs, which provides the exact contribution of each pixel. In fact, we notice that ReLU-based CNNs are actually piece-wise affine functions. Thanks to this property, the output score can be naturally decomposed into a sum of the contributions of each pixel, which can be easily calculated by a forward and a backward pass. Thus, by comparing the contributions of different pixels, VGatt can identify the pixels that contribute most to the output of CNN.

We then integrate VGatt into the previously introduced Weakly Supervised Learning (WSL) segmentation framework. Specifically, the pixels identified by the attribution method are assigned with the corresponding labels, and are considered as the ground truth labels to train segmentation networks. Through experiments on VOC 2012, we validate the effectiveness of the use of VGatt for defining accurate seeds. Our experiments also show that the WSL methods are promising for resolving the lack of fully annotated data for semantic segmentation. In addition to the application on the WSL segmentation, VGatt can also be applied in a broader context. Several other interesting applications of VGatt, such as visual explanation, network diagnosis, and generation of adversarial examples, are presented.

5.2 Perspectives for Future Work

At the end of our work, it is important to step back and summarize existing methods, thereby identifying promising research directions for future work.

In general, there are two ways to improve existing methods. The first approach is to identify the framework within which the existing algorithms are based and to improve the shortcomings of existing methods within this framework. The second is to identify problems that are not solved by the existing framework, and then make the appropriate changes in the modelization of the problem.

In the context of semantic segmentation, almost all DL-based approaches consider the semantic segmentation as a **pixel-wise classification problem**. The goal is to build a CNN-based network to learn a mapping from a RGB image to a label map, where each pixel is annotated with single label. We have identified three key components of methods under this modelization: the segmentation network, the loss function and the optimization process over datasets. In what follows, we first discuss what can be improved in each component under the "pixel-wise classification" modelization. Then, we present two limitations of this modelization and attempt to propose two modifications at the model level.

Pixel-wise classification problem In this paragraph, we stay within the existing modelization and present some interesting research directions with respect to the three components mentioned above.

1. **Network Architecture**

   - **Texture Invariance** The segmentation network should be invariant to the texture of objects. No matter what color or material a T-shirt is, the network should always output "T-shirt" for these pixels. However, none of the existing networks explicitly verify this property. A model trained on Beijing street scenes may not be able to accurately segment Paris street scenes, due to the change in texture of the objects (Geirhos et al. 2019). One possible reason for this is that existing methods rely almost entirely on the texture of the image. To solve this problem, we should better integrate the **semantic shape information** into the network architecture to alleviate the dependency on texture.

   - **Neural Architecture Search** Recent advances in Neural Architecture Search (NAS) show that automatically designed networks can surpass the performance of expert-designed models in many computer vision tasks such as image classification. One of the first attempts to apply NAS to semantic segmentation is carried out by Auto-Deeplab (Chenxi Liu et al. 2019), which achieves comparable performance with methods designed by experts. Although NAS usually requires large amounts of material resources, it is still well worth investigating. As hardware evolves, it may one day be possible for NAS to be more efficient than expert-designed networks, just as CNNs have become much more efficient than SIFT (Lowe 1999).

- **Complexity** Segmentation networks typically require a lot of memory usage because they need to hold as much spatial information as possible. However, limited by hardware resources, some methods are difficult to apply for large images as well as in certain application contexts such as on mobile. Therefore, techniques to reduce network size and memory usage as well as image-scale invariant methods may be important in these contexts.

2. **Loss function**

   - **Differentiable Evaluation Metric** As extensively discussed, the standard evaluation metric mean Intersection over Union (mIoU) has many drawbacks and cannot be directly minimized during training. Thus, it is compelling to build a new metric, which not only allows a better assessment of structure similarities (such as shape and size) between predicted segmentation masks and ground truth masks, but which is also differentiable, so that it can be directly optimized during training.

3. **Optimization process**

   - **Learning strategies** Generally, there are no specific learning strategies for training semantic segmentation networks. However, the numerous findings on the benefits of learning strategies for classification problems suggest that efficient learning strategies could also be useful for segmentation. For example, curriculum learning strategies start with easier sub tasks and then gradually increase the difficulty level. These learning strategies could not only improve the performance of existing models, but also address the problem of data imbalance.

   - **Lack of data** The lack of finely annotated data for semantic segmentation is a fundamental problem, especially when building a specific application with segmentation models. This problem can be addressed in many ways. For example, active learning algorithms are usually used to speed up the labeling process when collecting new datasets. These algorithms can interactively query a user to label new data points with the desired outputs. In addition, weakly supervised, semi-supervised and unsupervised methods, which make it possible to train segmentation models on partial or unlabeled data, are also particularly worth studying.

Limitations and new modelization Although, methods based on the "pixel-wise classification" modelization achieve satisfactory performance in many cases, this modeling has several limitations.

- **Multi-label annotations** In the classical pixel-wise classification setting, each pixel belongs to a single semantic class. However, this annotation can lead to some dilemmas. For example, let's consider an image where a person drives a car, and both "person" and "car" are semantic classes defined in the dataset. On the one hand, the pixels of the "person" should clearly belong to class "person". However, on the other hand, the whole set of pixels should also belong to the "car" class since the person is inside the car. To this end, using multi-label (non-exclusive) annotations for each pixel might reduce this kind of ambiguity.

- **Extra information** There are several cases that cannot be handled with 2D image semantic segmentation algorithms. For example, imagine a self-driving car driving down the road and observes via its camera a scene full of blue. Existing algorithms cannot tell whether this is the sky or a big blue truck, since both scenes provide the same 2D image. We may need additional information such as the depth map, multi-viewpoint data or a video input to solve these situations. These types of additional information are particularly important for many real applications, such as self-driving driving systems or robot vision systems.

The recently proposed task, panoptic segmentation (Kirillov et al. 2019), combines semantic segmentation and instance segmentation to achieve a complete understanding of 2d scenes. We believe that this new context is an excellent playground to explore the different paths of research mentioned here to extend our work.

BIBLIOGRAPHY

Ahn, Jiwoon and Suha Kwak (2018). “Learning Pixel-Level Semantic Affinity With Image-Level Supervision for Weakly Supervised Semantic Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 33).

Alex, Krizhevsky, Ilya Sutskever, and Geoffrey.E Hinton (2012). “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 3, 4).

Amirul Islam, Md, Mrigank Rochan, Neil D. B. Bruce, and Yang Wang (2017). “Gated Feedback Refinement Network for Dense Image Labeling”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 25).

Audebert, Nicolas, Alexandre Boulch, Bertrand Le Saux, and Sébastien Lefèvre (2019). “Distance transform regression for spatially-aware deep semantic segmentation”. In: *Computer Vision and Image Understanding* 189 (cit. on p. 48).

Badrinarayanan, V., A. Kendall, and R. Cipolla (2017). “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 39.12, pp. 2481–2495 (cit. on p. 25).

Bearman, Amy L., Olga Russakovsky, Vittorio Ferrari, and Fei-Fei Li (2016). “What’s the Point: Semantic Segmentation with Point Supervision”. In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 30, 32).

Bearman, Amy, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei (2016). “What’s the Point: Semantic Segmentation with Point Supervision”. In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 9).

Berman, M., A. Triki, and M. Blaschko (2018). “The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 29).

Bertasius, Gedas, Jianbo Shi, and Lorenzo Torresani (2016). “Semantic Segmentation with Boundary Neural Fields”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 41).

Bottou, Léon and Olivier Bousquet (2008). “The Tradeoffs of Large Scale Learning”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 22).

Brahimi, Sourour, Najib Ben Aoun, Alexandre Benoıt, Patrick Lambert, and Chokri Ben Amar (2019). “Semantic segmentation using reinforced fully convolutional densenet with multiscale kernel”. In: *Multimedia Tools and Applications* 78.15, pp. 22077–22098 (cit. on p. 25).

C.Dong, C.C.Loy, K.He, and X.Tang (2016). “Image super-resolution using deep convolutional networks.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 38.2 (cit. on p. 38).

Caliva, Francesco, Claudia Iriondo, Alejandro Morales Martinez, Sharmila Majumdar, and Valentina Pedoia (2019). “Distance Map Loss Penalty Term for Semantic Segmentation”. In: *International Conference on Medical Imaging with Deep Learning* (cit. on p. 41).

Chandra, Siddhartha and Iasonas Kokkinos (2016). “Fast, Exact and Multi-Scale Inference for Semantic Image Segmentation with Deep Gaussian CRFs”. In: (cit. on p. 27).

Chen, L., J. T. Barron, G. Papandreou, K. Murphy, and A. L. Yuille (2016). “Semantic Image Segmentation with Task-Specific Edge Detection Using CNNs and a Discriminatively Trained Domain Transform”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 41, 48, 52).

Chen, Liang-Chieh, Maxwell D. Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam, and Jonathon Shlens (2018). “Searching for Efficient Multi-Scale Architectures for Dense Image Prediction”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 23).

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille (2014). “Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs”. In: *CoRR* (cit. on pp. 24, 27, 32, 41).

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille (2016). “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* (cit. on pp. 24, 26, 34, 47, 53, 54, 84).

Chen, Liang-Chieh, George Papandreou, Florian Schroff, and Hartwig Adam (2017). “Rethinking Atrous Convolution for Semantic Image Segmentation”. In: *CoRR* abs/1706.05587 (cit. on pp. 24, 25, 53–55).

Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018). “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation”. In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 24, 34, 47, 51, 54, 55).

Chen, S.C, C.X Ding, and M.F Liu (2018). “Dual-force convolutional neural networks for accurate brain tumor segmentation”. In: *Pattern Recognition* 88, pp. 90–100 (cit. on p. 40).

Chen, Yifu, Arnaud Dapogny, and Matthieu Cord (2020). “SEMEDA: Enhancing Segmentation Precision with Semantic Edge Aware Loss”. In: *Pattern Recognition (under review, major revision)* (cit. on pp. 11, 38).

Chen, Yifu, Antoine Saporta, Arnaud Dapogny, and Matthieu Cord (2019). “Delving Deep into Interpreting Neural Nets with Piece-Wise Affine Representation". In: *IEEE International Conference on Image Processing (ICIP)* (cit. on pp. 11, 68).

Chollet, F. (2017). "Xception: Deep Learning with Depthwise Separable Convolutions". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807 (cit. on pp. 34, 47).

CloudFactory (2019). *The Ultimate Guide to Data Labeling for Machine Learning*. URL: https://www.cloudfactory.com/data-labeling-guide (cit. on p. 8).

Cocquerez, Jean-Pierre and Sylvie Philipp (1995). *Analyse d'images: filtrage et segmentation*. Masson (cit. on p. 15).

Codella, Noel C. F., Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen W. Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael A. Marchetti, Harald Kittler, and Allan Halpern (2019). "Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)". In: *CoRR* abs/1902.03368 (cit. on pp. 34, 60, 62).

Cordts, M., M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele (2016). "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 20, 34, 47, 69, 70).

Csurka, Gabriela and Diane Larlus (2013). "What is a good evaluation measure for semantic segmentation?" In: vol. 26 (cit. on pp. 29, 40, 52).

Dai, Jifeng, Kaiming He, and Jian Sun (2015). "BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 30).

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). "ImageNet: A Large-Scale Hierarchical Image Database". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 3, 70).

Durand, Thibaut, Taylor Mordan, Nicolas Thome, and Matthieu Cord (2017). "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 32).

Everingham, Mark, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman (2015). "The pascal visual object classes challenge: A retrospective". In: *International Journal of Computer Vision (IJCV)* 111.1, pp. 98–136 (cit. on pp. 21, 34, 42, 47, 70, 84).

Everingham, Mark, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman (2009). "The PASCAL Visual Object Classes (VOC) Challenges". In: *International Journal of Computer Vision (IJCV)* (cit. on p. 19).

Fan, Deng-Ping, Ming-Ming Cheng, Yun Liu, Tao Li, and Ali Borji (2017). "Structure-measure: A New Way to Evaluate Foreground Maps". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 30).

Fan, Deng-Ping, Cheng Gong, Yang Cao, Bo Ren, Ming-Ming Cheng, and Ali Borji (2018). "Enhanced-Alignment Measure for Binary Foreground Map Evaluation". In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (cit. on pp. 30, 48, 50).

Fan, H., X. Mei, D. Prokhorov, and H. Ling (2018). "Multi-Level Contextual RNNs With Attention Model for Scene Labeling". In: *IEEE Transactions on Intelligent Transportation Systems* 19.11, pp. 3475–3485 (cit. on p. 54).

Fong, R. and A. Vedaldi (2017). "Interpretable Explanations of Black Boxes by Meaningful Perturbation". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 79).

Fu, Jun, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu (2018). "Dual Attention Network for Scene Segmentation". In: (cit. on pp. 24, 28).

Fu, Jun, Jing Liu, Yuhang Wang, Yong Li, Yongjun Bao, Jinhui Tang, and Hanqing Lu (2019). "Adaptive Context Network for Scene Parsing". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 26).

Geirhos, Robert, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel (2019). "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on p. 97).

Ghiasi, Golnaz and Charless C. Fowlkes (2016a). "Laplacian Pyramid Reconstruction and Refinement for Semantic Segmentation". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 25).

Ghiasi, Golnaz and Charless C. Fowlkes (2016b). "Laplacian Pyramid Reconstruction and Refinement for Semantic Segmentation". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 53, 54).

Ghosh, Swarnendu, Nibaran Das, Ishita Das, and Ujjwal Maulik (2019). "Understanding Deep Learning Techniques for Image Segmentation". In: *CoRR* abs/1907.06119 (cit. on p. 26).

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the International Conference on Artificial Intelligence and Statistics* (cit. on p. 76).

Goodfellow, Ian, Jonathon Shlens, and Christian Szegedy (2015). "Explaining and Harnessing Adversarial Examples". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on p. 115).

Gordon, Cooper (2019). *New Vision Technologies For Real-World Applications*. URL: `https://semiengineering.com/new-vision-technologies-for-real-world-applications/` (cit. on p. 4).

Hahnloser, Richard H. R., Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas, and H. Sebastian Seung (2000). "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit". In: *Nature* 405, pp. 947–951 (cit. on p. 76).

Hahnloser, Richard H. R. and H. Sebastian Seung (2001). “Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 76).

Hale and Loke James (2019). *More Than 500 Hours Of Content Are Now Being Uploaded To YouTube Every Minute*. URL: https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/ (cit. on p. 1).

Hariharan, Bharath, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik (2011). “Semantic Contours from Inverse Detectors”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 19, 47, 84).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 76).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 4, 18, 24, 47, 76, 80).

He, Xuming and Richard S. Zemel (2009). “Learning Hybrid Models for Image Annotation with Partially Labeled Data”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 15).

Huang, G., Z. Liu, L. Van Der Maaten, and K. Q. Weinberger (2017). “Densely Connected Convolutional Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 80).

Huang, Zilong, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu (2019). “CCNet: Criss-Cross Attention for Semantic Segmentation”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 24, 28).

Huang, Zilong, Xinggang Wang, Jiasi Wang, Wenyu Liu, and Jingdong Wang (2018). “Weakly-Supervised Semantic Segmentation Network With Deep Seeded Region Growing”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 33).

Ioffe, Sergey and Christian Szegedy (2015). “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *International Conference on Machine Learning (ICML)* (cit. on p. 77).

Irem, Ulku and Akagunduz Erdem (2019). “A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D images”. In: *CoRR* abs/1912.10230 (cit. on p. 16).

Jampani, Varun, Martin Kiefel, and Peter V. Gehler (2016). “Learning Sparse High Dimensional Filters: Image Filtering, Dense CRFs and Bilateral Neural Networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 27).

Jeremy, Jordan (2019). *An overview of semantic image segmentation.* URL: https://www.jeremyjordan.me/semantic-segmentation/ (cit. on p. 6).

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). "Perceptual losses for real-time style transfer and super-resolution". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 38, 44, 50).

Kass, Michael, Andrew Witkin, and Demetri Terzopoulos (1988). "Snakes: Active contour models". In: *International Journal of Computer Vision (IJCV)* 1.4, pp. 321–331 (cit. on p. 15).

Kirillov, Alexander, Kaiming He, Ross B. Girshick, Carsten Rother, and Piotr Dollár (2019). "Panoptic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 99).

Kolesnikov, Alexander and Christoph H. Lampert (2016). "Seed, Expand and Constrain: Three Principles for Weakly-Supervised Image Segmentation". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (cit. on p. 32).

Krähenbühl, Philipp and Vladlen Koltun (2011). "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 15, 27).

Lafferty, John D., Andrew McCallum, and Fernando C. N. Pereira (2001). "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data". In: *International Conference on Machine Learning (ICML)* (cit. on p. 15).

Le, Vuong, Jonathan Brandt, Zhe Lin, Lubomir Bourdev, and Thomas S. Huang (2012). "Interactive Facial Feature Localization". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 34, 60).

LeCun, Yann and Corinna Cortes (2010). "MNIST handwritten digit database". In: URL: http://yann.lecun.com/exdb/mnist/ (cit. on p. 115).

Lee, Jungbeom, Eunji Kim, Sungmin Lee, Jangho Lee, and Sungroh Yoon (2019). "FickleNet: Weakly and Semi-Supervised Semantic Image Segmentation Using Stochastic Inference". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 32).

Lempitsky, Victor, Andrea Vedaldi, and Andrew Zisserman (2011). "Pylon Model for Semantic Segmentation". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 15).

Li, Xiaoxiao, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang (2017). "Not All Pixels Are Equal: Difficulty-Aware Semantic Segmentation via Deep Layer Cascade". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 53, 54).

Li, Xin, Zequn Jie, Wei Wang, Changsong Liu, Jimei Yang, Xiaohui Shen, Zhe Lin, Qiang Chen, Shuicheng Yan, and Jiashi Feng (2017). "FoveaNet: Perspective-Aware Urban Scene Parsing". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 54).

Lin, Di, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun (2016). "ScribbleSup: Scribble-Supervised Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 30).

Lin, Guosheng, Anton Milan, Chunhua Shen, and Ian Reid (2017). "RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 25, 53, 54).

Lin, Guosheng, Chunhua Shen, Anton Hengel, and Ian Reid (2016). "Efficient Piecewise Training of Deep Structured Models for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 27).

Lin, Tsung-Yi, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár (2020). "Focal Loss for Dense Object Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 42.2, pp. 318–327 (cit. on p. 29).

Lin, Tsung-Yi, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (2014). "Microsoft COCO: Common Objects in Context". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 51).

Lin, Tsung-Yi, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár (2014). "Microsoft COCO: Common Objects in Context". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 19).

Lin, Zhouhan, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio (2017). "A Structured Self-attentive Sentence Embedding". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on pp. 27, 41).

Liu, C., J. Yuen, and A. Torralba (2011). "Nonparametric Scene Parsing via Label Transfer". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 33.12, pp. 2368–2382 (cit. on pp. 60, 61).

Liu, Chenxi, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L. Yuille, and Fei-Fei Li (2019). "Auto-DeepLab: Hierarchical Neural Architecture Search for Semantic Image Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 23, 97).

Liu, Chenxi, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy (2018). "Progressive Neural Architecture Search". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 23).

Liu, Sifei, Jimei Yang, Chang Huang, and Ming-Hsuan Yang (2015). "Multi-Objective Convolutional Learning for Face Labeling". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 60, 61).

Liu, Wei, Andrew Rabinovich, and Alexander C. Berg (2015). "Parsenet: Looking wider to see better". In: *CoRR* abs/1506.04579 (cit. on p. 26).

Liu, Ziwei, Xiaoxiao Li, Ping Luo, Chen-Change Loy, and Xiaoou Tang (2015). "Semantic Image Segmentation via Deep Parsing Network". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 27).

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 9, 13, 16, 17, 25, 28, 38, 40).

Lowe, David G. (1999). "Object Recognition from Local Scale-Invariant Features". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 98).

Luc, Pauline, Camille Couprie, Soumith Chintala, and Jakob Verbeek (2016). "Semantic Segmentation using Adversarial Networks". In: *Advances in Neural Information Processing Systems Workshop (NIPS-W)* (cit. on pp. 40–42).

Maas, Andrew L., Awni Y. Hannun, and Andrew Y. Ng (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *International Conference on Machine Learning Workshop (ICML-W)* (cit. on p. 76).

Margolin, R., L. Zelnik-Manor, and A. Tal (2014). "How to Evaluate Foreground Maps". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 29, 30).

Máttyus, G., W. Luo, and R. Urtasun (2017). "DeepRoadMapper: Extracting Road Topology from Aerial Images". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 29).

Mcculloch, Warren and Walter Pitts (1943). "A Logical Calculus of Ideas Immanent in Nervous Activity". In: *Bulletin of Mathematical Biophysics* 5, pp. 127–147 (cit. on p. 4).

Mehta, Sachin, Mohammad Rastegari, Anat Caspi, Linda Shapiro, and Hannaneh Hajishirzi (2018). "ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 25).

Montufar, Guido F, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio (2014). "On the number of linear regions of deep neural networks". In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on p. 77).

Nagendar, G., D. Singh, V. Balasubramanian, and C.V. Jawahar (2018). "NeuroIoU: Learning a Surrogate Loss for Semantic Segmentation". In: *Proceedings of the British Machine Vision Conference (BMVC)* (cit. on p. 29).

Nair, Vinod and Geoffrey E. Hinton (2010). "Rectified Linear Units Improve Restricted Boltzmann Machines." In: *International Conference on Machine Learning (ICML)* (cit. on p. 76).

Noh, Hyeonwoo, Seunghoon Hong, and Bohyung Han (2015). "Learning Deconvolution Network for Semantic Segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 25).

Noyes, Dan (2019). *The Top 20 Valuable Facebook Statistics*. URL: https://zephoria.com/top-15-valuable-facebook-statistics/ (cit. on p. 1).

Papandreou, George, Liang-Chieh Chen, Kevin P. Murphy, and Alan L. Yuille (2015). "Weakly-and Semi-Supervised Learning of a Deep Convolutional Network for Semantic Image Segmentation". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 30, 32).

Pathak, Deepak, Evan Shelhamer, Jonathan Long, and Trevor Darrell (2014). "Fully Convolutional Multi-Class Multiple Instance Learning". In: *CoRR* abs/1412.7144 (cit. on p. 31).

Pavlidis, Theodosios (1972). "Segmentation of pictures and maps through functional approximation". In: *Computer Graphics and Image Processing* 1, pp. 360–372 (cit. on p. 14).

Peng, Chao, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun (2017). "Large Kernel Matters – Improve Semantic Segmentation by Global Convolutional Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 25, 53).

Pohlen, Tobias, Alexander Hermans, Markus Mathias, and Bastian Leibe (2017). "Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 25).

Prewitt, Judith M. S. (1970). "Object enhancement and extraction". In: (cit. on p. 14).

R.Zhang, P.Isola, and A.A. Efross (2016). "Colorful image colorization". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 38).

Ramachandran, Prajit, Barret Zoph, and Quoc Le (2018). "Searching for Activation Functions". In: *Proceedings of the International Conference on Learning Representations Workshop (ICLR-W)* (cit. on p. 76).

Real, Esteban, Alok Aggarwal, Yanping Huang, and Quoc V. Le (2019). "Regularized Evolution for Image Classifier Architecture Search". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on p. 23).

Richard, Szeliski (2010). *Computer Vision: Algorithms and Applications*. 1st. Berlin, Heidelberg: Springer-Verlag (cit. on p. 1).

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (cit. on pp. 25, 41).

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al.

(2015). “Imagenet large scale visual recognition challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252 (cit. on p. 70).

Selvaraju, Ramprasaath R., Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra (2017). “Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 75, 80, 82).

Shen, F., R. Gan, S. Yan, and G. Zeng (2017). “Semantic Segmentation via Structured Patch Prediction, Context CRF and Guidance CRF”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 53).

Shotton, Jamie, Matthew Johnson, and Roberto Cipolla (2008). “Semantic Texton Forests for Image Categorization and Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 15).

Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman (2014). “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”. In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on pp. 73, 81).

Simonyan, Karen and Andrew Zisserman (2015). “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on pp. 18, 39, 44, 76, 80, 84).

Smith, B. M., L. Zhang, J. Brandt, Z. Lin, and J. Yang (2013). “Exemplar-Based Face Parsing”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 60, 61).

Sobel, Irwin (2014). “An Isotropic 3x3 Image Gradient Operator”. In: *Presentation at Stanford A.I. Project 1968* (cit. on p. 14).

Springenberg, J.T., A. Dosovitskiy, T. Brox, and M. Riedmiller (2015). “Striving for Simplicity: The All Convolutional Net”. In: *Proceedings of the International Conference on Learning Representations Workshop (ICLR-W)* (cit. on pp. 73, 81).

Srinivas, Suraj and François Fleuret (2019). “Full-Gradient Representation for Neural Network Visualization”. In: *Advances in Neural Information Processing Systems (NeurIPS)* (cit. on p. 93).

Ulusoy, Ilkay and Christopher M. Bishop (2006). “Comparison of Generative and Discriminative Techniques for Object Detection and Classification”. In: *Toward Category-Level Object Recognition*. Springer Berlin Heidelberg, pp. 173–195 (cit. on p. 15).

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems (NIPS)* (cit. on pp. 27, 41).

Vemulapalli, R., O. Tuzel, M. Liu, and R. Chellappa (2016). “Gaussian Conditional Random Field Network for Semantic Segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 27).

Wang, P., P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell (2018). "Understanding Convolution for Semantic Segmentation". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)* (cit. on p. 53).

Wang, Zhengyang and Shuiwang Ji (2018). "Smoothed Dilated Convolutions for Improved Dense Prediction". In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery Data Mining* (cit. on p. 25).

Wang, Zhou, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli (2004). "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13.4, pp. 600–612 (cit. on p. 29).

Wei, Yunchao, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan (2017). "Object Region Mining with Adversarial Erasing: A Simple Classification to Semantic Segmentation Approach." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 32).

Wei, Yunchao, Xiaodan Liang, Yunpeng Chen, Zequn Jie, Yanhui Xiao, Yao Zhao, and Shuicheng Yan (2016). "Learning to segment with image-level annotations". In: *Pattern Recognition* 59, pp. 234–244 (cit. on p. 32).

Xu, J., A. G. Schwing, and R. Urtasun (2015). "Learning to segment under various forms of weak supervision". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 30).

Yuan, Yuhui, Xilin Chen, and Jingdong Wang (2019). "Object-Contextual Representations for Semantic Segmentation". In: *CoRR* (cit. on p. 24).

Yuan, Yuhui and Jingdong Wang (2018). "OCNet: Object Context Network for Scene Parsing". In: *CoRR* abs/1809.00916 (cit. on p. 28).

Z.Cheng, Q.Yang, and B.Sheng (2015). "Deep colorization". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 38).

Zagoruyko, Sergey and Nikos Komodakis (2016). "Wide Residual Networks". In: *Proceedings of the British Machine Vision Conference (BMVC)* (cit. on p. 80).

Zeiler, Matthew D., Graham W. Taylor, and Rob Fergus (2011). "Adaptive Deconvolutional Networks for Mid and High Level Feature Learning". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 25).

Zeiler, Matthew D and Rob Fergus (2014). "Visualizing and understanding convolutional networks". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 73, 81, 82).

Zhang, Hang, Kristin J. Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal (2018). "Context Encoding for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 53).

Zhao, H., J. Shi, X. Qi, X. Wang, and J. Jia (2017). "Pyramid Scene Parsing Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 24, 26, 27, 53, 54).

Zhao, Hengshuang, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia (2018). “PSANet: Point-wise Spatial Attention Network for Scene Parsing”. In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on p. 28).

Zhen, M., J. Wang, L. Zhou, T. Fang, and L. Quan (2019). “Learning fully dense neural networks for image semantic segmentation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on p. 41).

Zheng, Shuai, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr (2015). “Conditional Random Fields as Recurrent Neural Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 27, 41).

Zhou, Bolei, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba (2016). “Learning Deep Features for Discriminative Localization”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 32, 74, 79, 82).

Zhou, Bolei, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba (2019). “Semantic Understanding of Scenes Through the ADE20K Dataset”. In: *International Journal of Computer Vision (IJCV)* 127.3 (cit. on p. 19).

Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le (2018). “Learning Transferable Architectures for Scalable Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 23).

Appendix A

SIDE APPLICATIONS OF VGATT

Since the VGatt proposed in Chapter 4 gives the exact decomposition formula for Rectified Linear Unit (ReLU) based Convolutional Neural Network (CNN)s, this method can be used in broader contexts. In this appendix, we present two side applications of our proposed attribution method. In Section A.1, we show that VGatt can be used to diagnose the behavior of the CNNs and in Section A.2 we use the method to find adversarial examples.

a.1 Network Diagnosis

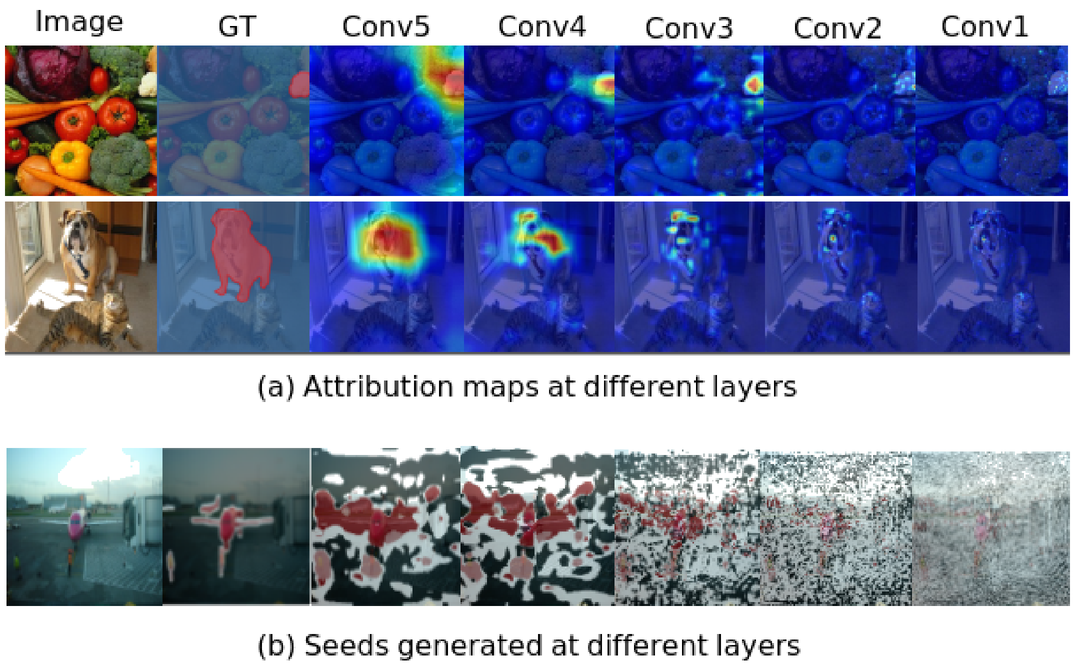


Image
GT
Conv5
Conv4
Conv3
Conv2
Conv1
(a) Attribution maps at different layers
(b) Seeds generated at different layers

Figure A.1. – Attribution maps (a) and seeds (b) generated at different layers with VGatt.

We have observed in both Section 4.4.1 and Section 4.3.3 that attribution maps (or seeds) provided by low-level features are very sparse where only a few pixels

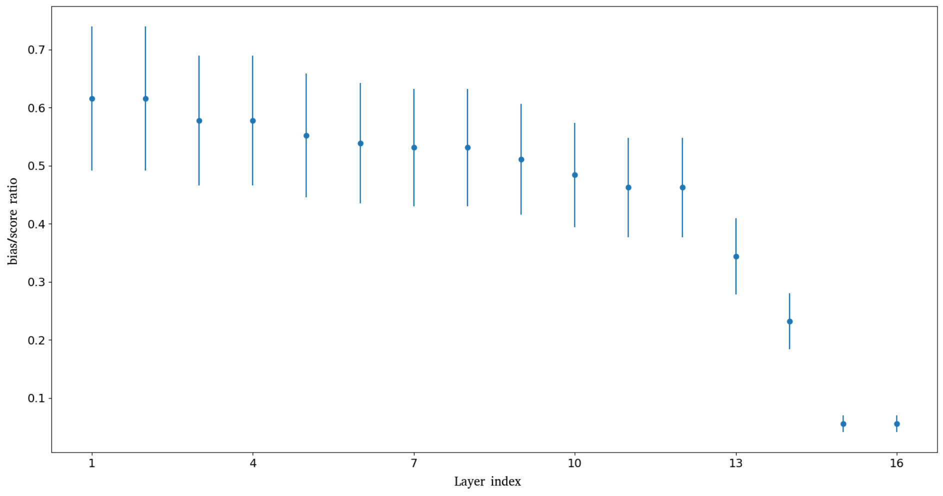


Figure A.2. – Bias/score ratio as a function of the layer depth for VGG19. For each layer, we show the average ratio (blue point) and its confidence interval (blue line) estimated on 300 randomly selected images from Pascal VOC dataset. The estimated average ratio decreases to zero as going deeper into the network.

are indicated as relevant (see Figure A.1). In order to better understand this phenomenon, we provide an in-depth analysis in this section.

As demonstrated in Section 4.3.2, ReLU based CNNs are actually piece-wise affine functions. It means that for a given image I, the output of the network on I is $F_{|V(I)}(I) = WI + b_{|V(I)}$, where WI represents the pixels contributions. Obviously, if the output is fixed, a large bias $b_{|V(I)}$ signifies that the contributions of pixels are small. In this case, the output is mainly explained by the bias term rather than by the contribution of each pixel. Thus the bias/score ratio could be used as an interpretability measure of pixel contributions: the larger the ratio, the less the output is explained by pixel contributions.

We randomly select 300 images from Pascal VOC 2012 validation set and compute the bias/score ratio at different layers of a ImageNet pre-trained VGG network. These ratios are then averaged among the all classes and presented in Figure A.2. We can see that the bias term is relatively important (from approximately 0.5 to 0.7 times the classification score) in the early stage. As going deeper, the bias term decreases and approaching zero at the last layers. This is perfectly consistent with the results shown in Figure A.1, where lower-level features provide sparse and inaccurate attribution maps (and seeds), while higher-level features provide relatively accurate and dense attribution maps (and seeds). This analysis of the bias/score ratio also suggests that introducing constraints on the bias term during training may improve the interpretability of the network.

a.2 Adversarial Attack

(Goodfellow et al. 2015) has demonstrated the vulnerability of current deep networks to adversarial examples: a slight, imperceptible perturbation of the input image can easily lead the network to misclassify the image with high confidence. Particularly, they slightly modify an image in the same direction of its gradients (∇f_I) defined by the classification networks and show the resulting image has a large chance to fool a network trained on MNIST dataset (LeCun et al. 2010). In this section, we propose another method to generate adversarial examples which consists in removing the k most relevant pixels defined by VGatt. The intuition is that good attribution algorithms identify the most important pixels for the classification CNN. Thus, by removing these pixels from the image, the result of the network will be significantly modified as well.

Specifically, our procedure is as follows: for a given image, we replace the most relevant k pixels for the predicted class with black pixels. And then we measure the neural network output for this class, before and after perturbation, and plot the absolute value of the fractional difference. We employed this strategy on Pascal VOC 2012 validation set (1,449 images) for a VGG-19 network trained on VOC training set, and we vary k from 1% to 8%. To better measure the quality of the generated adversarial samples, we also evaluate the accuracy of the network on the generated val set. We use Grad-CAM as the baseline method.

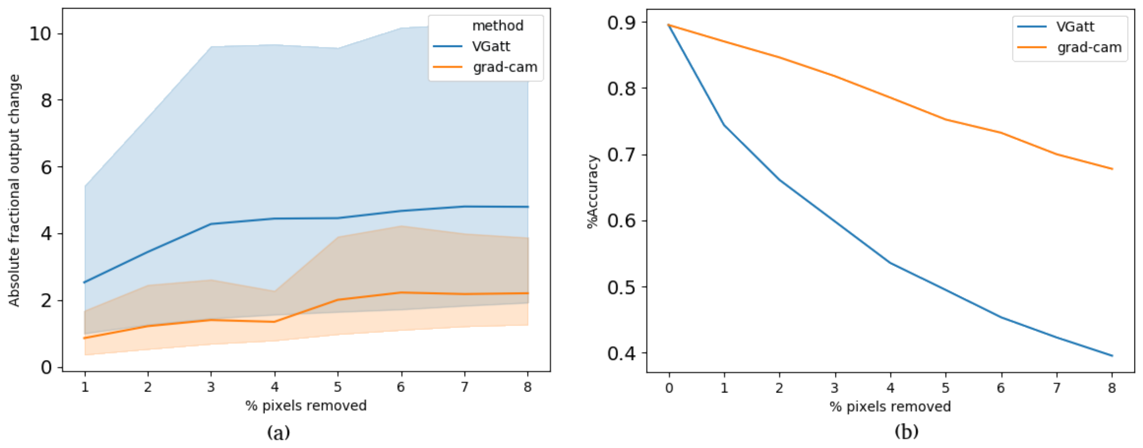


Figure A.3. – (a) Pixel perturbation on VOC 2012 val set where we remove k% most relevant pixels and measure absolute value of fractional output change. Higher curves mean that removed pixels are more relevant. (b) Classification accuracy evaluated on VOC 2012 val set where k% most relevant pixels are removed. The lower the accuracy, the better.

The quantitative results are shown in Figure A.3. As can be seen, our method provide a higher absolute fractional output change curve as well as a lower accuracy curve, as compared to the baseline method. These curves show that by removing the k% most relevant pixels defined by our method, the output of the same class is significantly changed and lead to misclassifications. By removing only 1% of the pixels, the accuracy drops from 90% to 74%, which is largely ahead of Grad-CAM (from 90% to 87%). By removing 8% of pixels with VGatt, half of the images of val set fool the network. These results demonstrate that our method is indeed able to find most relevant pixels and is much better than Grad-CAM.

Finally, we show some qualitative results to complete this application. We present the 1%, 4% and 8% most relevant pixels for three images in Figure A.4 (the removed pixels are displayed in red for better illustration). We can see that VGatt is able to focus on the most prominent parts of the object (face of the dog and sheep, and body of the airplane), while Grad-CAM puts a lot of attention on background pixels. It is worth noting that the difficulty of the three examples is different: the first one fools the network with only 1% of pixels (from 96% to 0.1%), the second one needs 4% of pixels (from 99% to 45%), and the last one requires 8% (from 99% to 24%). This may be determined by the size of the object, or the size of the discriminating parts.

With this application, we show that VGatt can be used to generate adversarial examples for classification networks since it can identify the most relevant pixels for the predictions. Furthermore, it allows us to diagnose the network in such a way that it can be easily fooled by removing only a few pixels from the images.

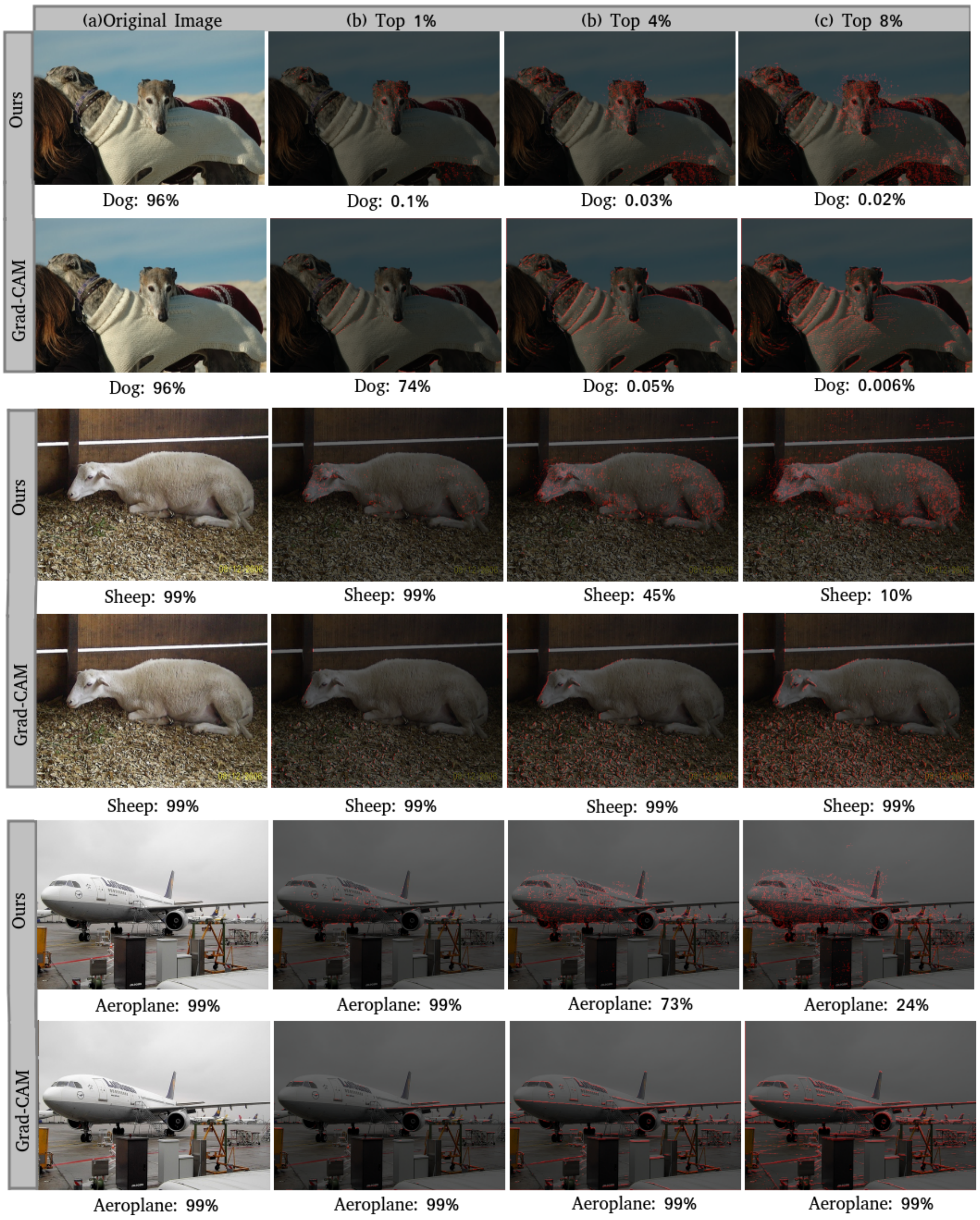


Figure A.4. – Adversarial examples generated by removing top 1%, 4% and 8% pixels. Since only a few pixels have been removed, we display the removed pixels in red to better visualize the ones that are considered most relevant.