VISUAL SALIENCY DETECTION AND SEMANTIC SEGMENTATION

by

NASIM SOULY
M.S. Amirkabir University of Technology (Tehran Polytechnic), Iran 2009

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2017

Major Professor: Mubarak Shah

ABSTRACT

Visual saliency is the ability of a vision system to promptly select the most relevant data in the scene and reduce the amount of visual data that needs to be processed. Due to its ability to reduce the amount of processing data and its applications in computer vision tasks, visual saliency has gained interest in computer vision studies. We pro-pose a novel unsupervised approach to detect visual saliency in videos of natural scenes. For this, we employ a hierarchical segmentation technique to obtain super-voxels of a video and simultaneously we build a dictionary from cuboids of the video. Then we create a feature matrix from coefficients of dictionary elements. Next, we decompose this matrix into sparse and redundant parts and obtain salient regions using group lasso. The applicability of our method is examined on four video data sets of natural scenes. Our experiments provide promising results in terms of predicting eye movement using standard evaluation methods. Moreover, we apply our video saliency on human action recognition task in a standard dataset and achieve better results. Saliency detection only highlights important regions, and there is no notion of classes in saliency. In Semantic Segmentation, the aim is to assign a semantic label to each pixel in the image. Even though semantic segmentation can be achieved by simply applying classifiers (which are trained using supervised learning), to each pixel or a region in the image, the results may not be desirable due to the fact that general context information beyond the simple smoothness is not considered. In this dissertation,, two supervised approaches to address this problem are proposed. First, an approach to discover interactions between labels and regions using a sparse estimation of precision matrix, which is the inverse of covariance matrix of data obtained by graphical lasso. In this context, we find a graph over labels as well as regions in the image which en-codes significant interactions and also it is able to capture the long-distance associations. Second, a knowledge-based method to incorporate dependencies among regions in the image during inference. High level knowledge rules - such as co-occurrence, spatial relations and

mutual exclusivity - are extracted from training data and trans-formed into constraints in Integer Programming formulation. Competitive experimental results on three benchmark datasets including SIFTflow MSRC2 and LMSun are obtained used these two methods. A difficulty which most supervised semantic segmentation approaches are confronted with is the lack of enough training data. Annotated data should be at the pixel-level, which is highly expensive to achieve. To address this limitation, next a semi super-vised learning approach to exploit the plentiful amount of available unlabeled as well as synthetic images generated via Generative Adversarial Networks (GAN) is present-ed. Furthermore, an extension of the proposed model to use additional weakly labeled data to solve the problem in a weakly supervised manner is proposed. The basic idea here is by providing these fake data from the Generator and the competition between real/fake data (discriminator/generator networks), true samples are encouraged to be close in the feature space. Therefore, the model learns more discriminative features, which lead to better classification results for semantic segmentation. We demonstrate our approaches on three challenging benchmarking datasets: PASCAL, SiftFLow, Stanford and CamVid.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Mubarak Shah, and committee members, Dr. Ulas Bagci, Dr. Pensky, Dr. Guo-Jun Qi and Dr. Rahnavard for their guidance throughout the research process. I feel very fortunate for having been part of Center for Research in Computer Vision (CRCV) and for being given the opportunity to be involved in and learn from a wide range of projects. I would like to thank my parents, my sister and my brother for their endless support. Last but not least, I want to thank my husband Roozbeh for all his help and encouragement during my PhD.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Psychology researches (e.g. [160]) have shown that perceptual grouping plays an important role in human visual perception, meaning that the human visual system can extract high level information (e.g. structure) from low level primitives. The hierarchical grouping is expressed by proximity, similarity, continuation, closure, and symmetry. This representation of images can aid in the visual indexing , retrieving and processing. In Computer Vision, perceptual grouping has been employed in segmentation, which is generally the first step in analyzing or interpreting an image automatically in many application. A segmented image could also be used as an initial step in in a wide variety of computer vision problems such as, optical character recognition (OCR), object tracking in a sequence of images, object detection, boundary detection, occlusion detection, image compression, matching, recognition, etc. (e.g. [49], [75]). Formaly, image segmentation is defined as dividing an image into different homogeneous disjoint sub-regions. These regions may correspond to objects or parts of objects. The level to which this division is implemented depends on the problem. For instance, in some applications the process should stop when the objects of interest have been segmented, while in some other applications, simple and meaningful regions, which may correspond to parts of the objects and scene may be sufficient.

The segmentation techniques employ pixel features such as color, gray level, texture, as well as similarity between pixels and spatial coherence to find homogeneous regions. Image segmentation, can also be considered as clustering of image-pixels using visual characteristics e.g. intensity. Image segmentation can be categorized generally into two main classes: object-based segmentation methods and Region-based segmentation approaches. Object-based methods employ object detectors or a bank of object proposals to define shape masks, they attempt to segment classes or instances of objects. Latter methods (e.g. [49], [46]), which have been more explored and used in generic image segmentation, are applied on features of the images. Another form of segmentation

is super-pixel segmentation, which has become popular in computer vision studies lately. Super-pixel is a group of pixels, with homogeneous intensity; several super-pixels based segmentation methods which partition an image into hundreds of presumably small regions have been proposed.

Video segmentation generalizes the idea of image segmentation to the grouping of pixels into spatio-temporal regions. These regions must be coherent in motion in addition to appearance homogeneity. Video segmentation is exploited in several higher-level vision tasks such as salient object segmentation [66], action recognition [90], object tracking [104], content-based retrieval [71], etc. Video segmentation is a more challenging task due to frame to frame changes, which make finding consistent regions difficult. In order to overcome this issue, some methods enforce consistency of region boundaries over time using spatial region matching among frames. Volumetric approaches, though, generalize image segmentation to 3D-volume and achieve coherence among few frames. However, long-term consistency needs more complex approach. Hierarchical approaches [52], [164] have been more successful in modeling spatio-temporal coherency and scalability to longer videos with more pixels to process.

A functional and practical variety of segmentation is foreground-background segmentation or basically extracting the important regions of the scene. For instance, moving objects segmentation is typically the first step in many visual surveillance applications. Therefore, in the first part of this thesis we aim to leverage the low-level segmentation to obtain the distinctive and salient parts of the scene which pop-out throughout a video. To address this problem, many existing techniques rely on segmenting primary regions, interactively or via detectors, in beginning frames or a set of key frames followed by finding corresponding regions via tracking or matching. However, this assumption limits the applicability and feasibility of the method. Therefore, we propose an end to end approach in which the salient areas of a video are obtained using an unsupervised approach.

While in segmentation there is no attempt to understand what each region represents, in semantics

segmentation one aims to understand the role of individual parts in the scene, thus a semantically meaningful label from a determined set of categories is assigned to each segment. Semantic segmentation has many direct applications in computer vision. The ultimate goal of breaking an image into regions which represents semantic classes is to provide computers with an ability to understand and perceive the visual world. For instance, in [95] authors use semantic segmentation to detect and recognize road signs. The autonomous driving car is one of the subjects of broad and current interest, and segmenting and labeling the environment is a necessary task to interact with the world. Several datasets have been gathered to explore this topic implying the importance of the subject [11], [176]. Recognizing, understanding the image in pixel level is required in robot vision as well. Also, semantic segmentation has several applications in medical imaging such as tumor segmentation, automatic lung segmentation, instruments detecting in operations. In addition, in remote sensing in order to detect lands and roads semantic segmentation can be employed.

In this thesis, we utilize grouping information and segmentation in two important manners. First, in visual saliency detection in videos, in which we single out important regions in a video via an unsupervised method (chapter 3). Second, alongside with label information of pixels in images, we segment an image into semantically meaningful regions yield to better understanding of the image (chapters 4-5). First two methods for semantic segmentation employ supervised learning and use labeled data. However, we leverage from unlabeled data in the last proposed approach (chapter 6) to improve the segmentation results. To this end, we introduce a semi-supervised approach using generative adversarial networks for the semantic segmentation application. In semi-supervised learning one aims to promote and boost learning process by employing both labelled, presumably a small set, and available unlabeled data.

We make the following contributions in this dissertation to address visual saliency detection in videos and semantic segmentation problem in images:

- We introduce a novel unsupervised method for detecting visual saliency in videos of natural scenes. For this purpose, we divide a video into non-overlapping cuboids and create a matrix whose columns correspond to intensity values of these cuboids. Simultaneously, we segment the video using hierarchical segmentation method and obtain super-voxels. Then, the video is represented as coefficients of atoms from a dictionary learned from the feature data matrix of a video, and decomposed into salient and non-salient parts. We use group lasso regularization to find the sparse representation of a video, which benefits from grouping information provided by super-voxels and extracted features from the cuboids. We find saliency regions by decomposing the feature matrix of a video into low-rank and sparse matrices by using Robust Principal Component Analysis (RPCA) matrix recovery method. We show the applicability of our method by testing it on three video data sets of natural scenes and our experiments provide promising results in terms of predicting eye movement using standard evaluation methods. Moreover, we apply our video saliency on human action recognition task on a standard dataset and achieve better results.

- For solving semantic segmentation problem, we employ a sparse estimation of precision matrix (also called concentration matrix), which is the inverse of covariance matrix of data obtained by graphical lasso to find significant interactions between labels and regions. To do this, we formulate the problem as an energy minimization over a graph, whose structure is captured by applying sparsity constraint on the elements of the precision matrix. This graph encodes only significant interactions and avoids a fully connected graph, which is typically used to reflect the long distance associations. We use local and global information to achieve better labeling. Experimental results on three challenging datasets and competitive baselines indicate the advantage of our approach.

- We propose to use high level knowledge regarding rules, such as presence, implication and mutual exclusivity, to incorporate dependencies among regions in the image to improve

scores of scene labeling or semantic segmentation. Towards this aim, we extract knowledge-based rules from training data and transform them into constraints in Integer Programming to optimize the structured problem of assigning labels to super-pixels (consequently pixels) in an image. In addition, we use soft-constraints, which permit violation of hard constraints by imposing penalties, and can be solved through slack variables; thereby yielding a more flexible model. Furthermore, we learn scene-label association weights in order to employ global context to improve label confidences. We assess our approach on three datasets and obtaine promising results.

- We present a semi-supervised approach to address the shortage of available fully-labeled data for semantic segmentation. We leverage, on one hand, a massive amount of available unlabeled or weakly labeled data, and on the other hand, synthetic images created through Generative Adversarial Networks (GAN) [47]. The underlying idea is that adding large fake visual data forces real samples to be close in the feature space, which, in turn, improves multiclass pixel classification. To ensure a higher quality of generated images for GANs with consequence of improved pixel classification, we extend the above framework by adding weakly annotated data, i.e., we provide class level information to the generator. We test our approaches on several challenging bench-marking visual data sets, i.e. PASCAL, SiftFLow, Stanford and CamVid, achieving competitive performance compared to state-of-the-art semantic segmentation methods.

In the next Sections, we introduce the different approaches to tackle the scene understanding using grouping and segmentation which we explore in this dissertation: visual saliency detection using group lasso regularization in videos in Section 1.1, scene labeling using sparse precision matrix in Section 1.2.1, scene labeling through knowledge-based rules in Section 1.2.2, and semi-supervised semantic segmentation using GAN in Section 1.2.3.

1.1 Visual Saliency Detection

Although images of natural scenes contain large amounts of data which need to be processed, significant portion of scenes are redundant and visual system has limitations to fully process a complex scene. Therefore, a process of selecting informative data (saliency detection) is needed.

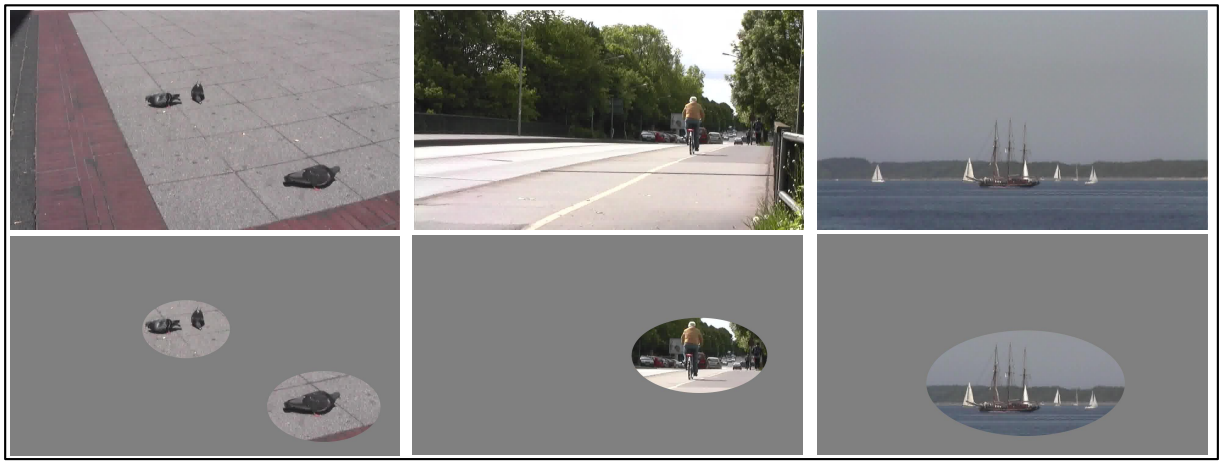


Figure 1.1: High-saliency areas of a natural scene are the small portions that hold most important information and can be identified easily by human vision system. In this figure the top row shows the frames from videos of natural scenes and the bottom row shows parts of scene which are most relevant to understand the scenes.

The human vision system is equipped with a cognitive mechanism that points out relevant parts and regions of interest from complex scenes, and naturally one's gaze is directed to important aspects of a scene. The part of an image or a video that captures human attention is said to be "salient"; that is where people look when watching a scene. For instance, in figure 1.1, we are likely to be attracted to the pigeons and not the ground, person riding a bike not the road or boats and the ship not the sea.

Despite the fact that extensive psychological and neurophysiological studies have investigated the

human visual system (e.g. [117], [154]), it is not completely understood how one's gaze easily is guided by some salient stimulus. Therefore, saliency detection remains a challenging problem. In determining saliency, the computational vision model seeks to find the part of an image or video which stands out from the rest of the scene. Changes in the scene such as color variation, spatial contrast or sudden movement are important factors since they redirect the observer's gaze. A variety of methods exist to indicate what exactly captures the eye [99].

In general, saliency detection methods are divided into two types: top-down [9, 152], which deals with task-driven methods involving high-level cognitive process that model attention by task; and bottom-up [128, 70] which are stimuli-driven and they extract eye-catching regions from image or video without any prior knowledge. **Top-down** models employ a biased selection process considering the expectation, "*will*" of a target, and they are the subject of interactive studies such as driving and game playing [8]. On the other hand, **bottom-up** approaches try to find novel parts of a scene using low-level features without prior knowledge about the scene. The latter methods mostly have been investigated using eye movement prediction in free-viewing of videos. Bottom-up methods are usually faster because they use low-level features characterized by stimuli driven factors [128]. In this thesis, we leverage from bottom-up segmentation method to find super-voxels in the videos to be used in salient region segmentation.

In spite of much attention to saliency detection in images (e.g. [13], [44], [8]), few methods have been proposed for videos. We live in a dynamic world and videos capture more realistic models of the environments in which one's vision system is exposed to. In this approach, we use spatiotemporal visual features to develop a method for detecting saliency in videos.mThe goal of this method is to find salient objects and actions with no presumption about the target in free-viewing videos. Therefore, we propose a bottom-up approach to find visual saliency and predict gaze based on visual features. In addition, in this approach there is no need for a training process in which similar videos have to be available.

The proposed model focuses on the concept of a saliency map, which indicates the saliency of a specific location over the entire scene. The task of saliency detection consists of three major steps: extraction of the features that could be used to find salient areas effectively, then determining salient regions based on those features and finally assigning saliency value or score to each part.

Since the salient regions in videos are only a small part of a video, we use a sparse-signal analysis technique to represent the information as redundant part plus salient parts. In this way, non-salient areas, such as background, are expressed by low-dimensional subspace and salient parts are specified by sparse ones [122]. The idea is that, perceptually, saliency is related to homogeneity, in a manner such that when homogeneity increases saliency decreases [39]. In this approach, we use Robust Principal Component Analysis (RPCA) low-rank matrix recovery [162] method in order to decompose the obtained feature matrix.

The essential task here is to determine a feature (descriptor) matrix that determines a space in which the non-salient regions stay in a low-dimensional subspace. For this reason, the main part of our work involves providing an appropriate feature matrix as input for the decomposition step. Since sparse coding representation, inspired by neuroscience studies, has been successful in modeling natural scenes, and because psychological studies, e.g. [39], show heterogeneous surfaces are more salient than homogeneous ones, we use sparse representation in our model.

Sparse coding suppresses slight changes in a scene so that the strong variations stand out. In this new representation redundant data lies in a low rank space. A video is represented as a collection of spatiotemporal cuboids expressed in terms of an over-complete dictionary. In doing so, a dictionary is created whose atoms are learned based on the feature data matrix of a video. By using L_1-minimization approach, a coefficients matrix is obtained and then is divided into salient and non-salient parts. However, the coefficients could be noisy and possibly due to salient regions not being sparse, inasmuch as a large area divided into small patches. In order to address this problem, we

propose to use super-voxels and sparsity among super-voxels rather than cuboids. Consequently, in addition to decomposing a video into cuboids, we group these cuboids in super-voxels.

Next, a group lasso regularization method –which uses $L_{1,2}$-norm minimization to encourage the columns within a group to be zero– is used to transform the video feature matrix into sparse coding space that is used by Robust PCA. The final step of our method is to find a saliency map via the sparse matrix that was found by the decomposition step. Each vector in the acquired sparse matrix belongs to a cuboid in the original video. Thus by computing L_1-norm of these vectors, the saliency values of super-voxels are achieved. Furthermore, by applying a threshold sailent regions in the video are obtained. In Chapter 3 the details of our approach and experimental results are presented. In addition, we explore the idea of applying saliency detection as a pre-processing to prune features for action recognition in videos.

1.2 Semantic Segmentation in Images

In saliency detection, salient and non-salient parts of the image and videos are detected where people pay attention to. However, in semantic segmentation each pixel is assigned a label. Semantic image segmentation, is a classic and long standing challenging task in computer vision, due to the efforts needed to simultaneously segment and recognize the image regions. While segmentation is a task to partition an image into coherent parts without any notion of the representation of each part, the aim of semantic segmentation is to divide an image into semantically meaningful regions, and to label (classify) each segment with one of the classes from a pre-defined set of categories.

The same goal can be achieved by classifying each pixel (pixel-wise classification instead of each region), which results in the same output. “semantic segmentation”, “scene labeling” and “pixelwise classification” essentially attempt to address the same problem: assigning a label to each

Figure 1.2: An Illustration for Image Semantic Segmentation. Given sample images (first row) and a set of labels, semantic segmentation method should generate an image (second row) in which each pixel is classified.

pixel in the image, Figure 1.2 depicts the semantic segmentation task. Recognizing the categories of objects present in the scene and the knowledge of their placement yield to perceive information about the 3D world. The constraints imposed by the associations and connections of different type of objects and their spatial arrangement in the scene can be helpful in extracting the information and understanding images.

Many approaches to address semantic segmentation problem have been proposed recently, including estimating labels pixel by pixel, or using features of segmented regions. Random field methods are a widely used approach of modelling spatial regularities in images. Generative modeling using Markov Random Fields (MRFs) were used initially, but recently, because of their ability to directly predict the labels given the image, Conditional Random Field (CRF) [79] models have

become more popular. In both Markov Random Field or Conditional Random Field approaches the scene labeling task is formulated as finding the most probable labeling.

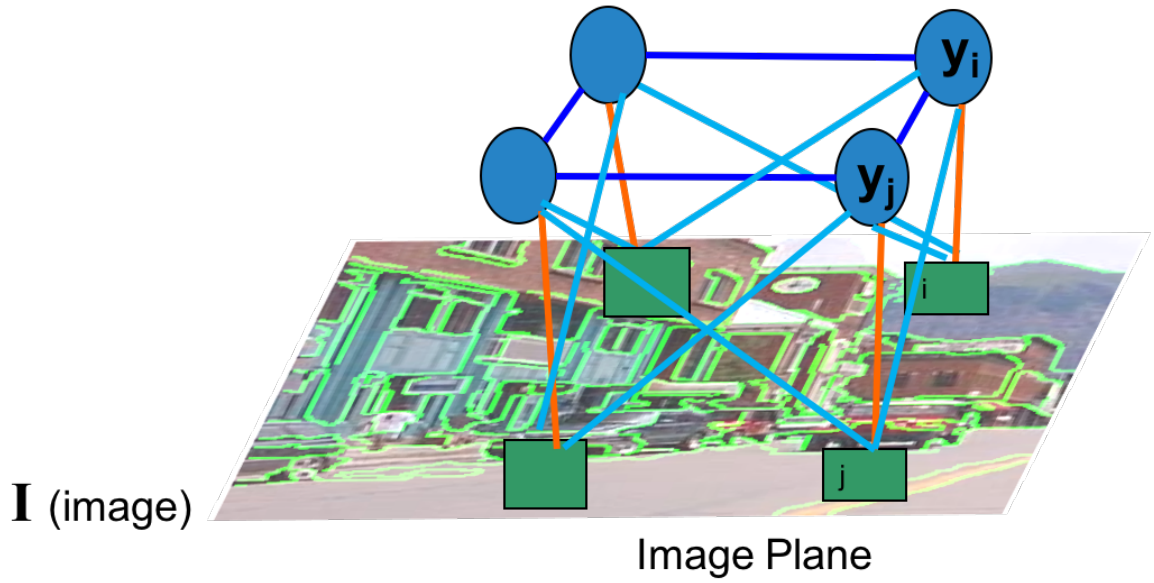


Figure 1.3: The CRF model: the output labeling y_i, y_j conditioned on observed features via an energy function.

These models generally comprise of the unary or association potential, which measures how likely a pixel (or a super-pixel) can be assigned a particular label without taking into account the properties of other parts of the image, and the smoothing or interaction potential, which assesses (evaluates) how the labels of the other connected nodes (pixels or super-pixels) interact to maximize the assignment agreement.

The CRF model defines a Gibbs distribution of the output labeling y conditioned on observed features I via an energy function E as follows: $p(\mathrm{y}|I,\theta) = \frac{1}{Z(I,\theta)}[exp(-E(\mathrm{y};I,\theta)]$. Z is the normalization factor to ensure that the distribution is summed to 1. The energy function can be factorized into unary potential ϕ which is associated with each pixel (or other primitive e.g. su-

perpixel) and pairwise potential ψ between a pair of primitives, then maximizing a posterior of p is formulated as $p(y|I,\theta) = \frac{1}{Z(I,\theta)}[\prod_i \phi_i(y_i, I|\theta_i) \prod_{ij} \psi_{ij}(y_i y_j, I|\theta_{ij})]$, which is equivalent to minimizing the energy function (figure 1.3).

Semantic segmentation approaches which follow this formulation, generally consist of the following steps: the features from primitives (pixels, patches or super-pixels) are extracted, using these features and a model (usually a classifier) is trained to compute the score (i.e. confidence) for data samples. During testing, the model is applied on test images unary and pairwise potentials are computed and finally MAP inference is applied on the graph. The different methods vary depending on how they perform these steps.

Commonly, the structure of the CRF is specified manually; in images typically a 2D lattice is used to build an adjacency CRF using the neighboring pixels. However, this model has two important limitations. First, it is unable to incorporate long-range (long-distance) connections between different regions of the image. Second, it may not be able to model the contextual relationships among labels and may not be capable of capturing the complexity in the labels.

One of the approaches to overcome this problem is to use a *fully connected CRF*, in which pairwise potentials are defined between all pairs of the nodes (pixels or super-pixels). However, the main limitation of this method is the complexity of the inference. The overwhelming number of edges in the model makes the problem difficult to be solved in an acceptable time. Furthermore, since the optimization solution to the multi-label CRF is not exact, the complexity in the structure leads to reduced accuracy. Figure 1.4 shows the comparison between a fully-connected graph and the its sparse version.

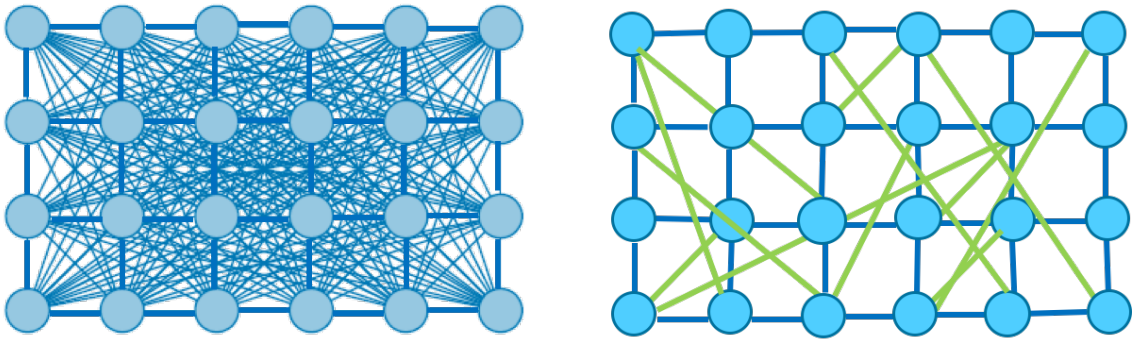


Figure 1.4: On the left a fully connections of the graph is shown and on the right a sample of sparse connectivity of the same nodes.

1.2.1 Scene Labeling using Sparse Precision Matrix

We propose [138] to use a sparse estimation of precision matrix (also called concentration matrix), which is the inverse of covariance matrix of data obtained by graphical lasso to find interaction between labels and regions. To do this, we formulate the problem of scene labeling as an energy minimization over a graph, whose structure is captured by applying sparsity constraint on the elements of the precision matrix. This graph encodes (or represents) only significant interactions and avoids a fully connected graph, which is typically used to reflect the long distance associations.

We use local and global information to achieve better labeling, and propose to learn the label graph (the correlation graph between labels in the dataset) and find the structure of the super-pixels within the image using the sparse *precision matrix* (also called concentration matrix) estimated using graphical lasso. We aim to infuse the relations between labels in the model, without expensive learning of parameters in training the CRF.

By using precision matrix, we find the compatibility among labels instead of using the Potts model

for all pairs of labels, thus the cost for different combination of labels would be dependent on their correlation and the way they influence each other. However, we consider only nodes (regions) in the image, which have interactions with other regions and which are not limited to spatial smoothness only. Also, our model facilitates using smaller elements (smaller super-pixels or pixels) of the image. Because we do not need to encode all interactions between these elements, we can find finer and more accurate boundaries using smaller super-pixels. In our approach, in addition to utilizing the scene semantics by employing the structure and dependency among labels and regions, we also exploit global context by refining local probabilities achieved by classifiers using a retrieval set, which is obtained based on k nearest neighbors of image employing GIST features.

In Chapter 4, we demonstrate our experimental results which show that relations between labels, which obtained via Graphical lasso, are meaningful and more importantly these dependencies improve the labeling performance in inference on test data.

1.2.2 Scene Labeling Through Knowledge-Based Rules

As mentioned in the previous section, many learning methods have been proposed for structured modeling, which mostly attempt to model dependency among labels during the learning process by optimizing a global objective function (as in using MRF and CRFs). However, efficiency and tractability confine to encode only local relationships. Although non-local dependencies can be encoded in such models as well, the models then need to learn more parameters (e.g., in graphical models, more edges and weights need to be included to model long-term dependencies). While this can be achieved by infusing the knowledge in the model as constraints, rather than using a fully connected graph to capture the all possible interactions or by employing higher order representation, where, in both cases the complexity of method during learning and inference increases significantly.

In order to interpret and analyze an image; objects types and their placement in the scene provide substantial information about the visual world captured in the image. The constraints implied by the relations between different type of objects and their spatial arrangement in the scene can aid in processing and understanding images. The effectiveness and advantage of using world knowledge to understand images have been explored and assessed in early computer vision research works [112], and in this thesis we pursue this line of work to semantically label the pixels in an image using semantic knowledge automatically derived from training images.

In this approach, we propose to use the knowledge of some relevant interactions between labels *directly* in the model, instead of *indirectly* via learning. In doing so, we benefit from the prior knowledge during the inference stage, by adding some constraints that must be satisfied. As a result, we apply *inference only during testing*, and do not require solving any inference problem during the training process. Therefore, we can use any training algorithm, and apply the constrained inference model; that way the features and constraints are separated and distinguished. We use a data-driven approach to extract rules to form the training data which is later used to generate expressive constraints. Since the constraints are formed as Boolean functions, they can be represented as logical expressions. Some constraints are not always valid and applicable in some of the cases; thus, we model soft-constraints in addition to the hard constraints. For instance, most of the time mountain and building are not seen together, but this is not always the case. Thus, we use slack variables in the objective function to model the soft constraints. To solve the inference problem with expressive constraints, we propose to use an Integer Programming formulation in this work.

By exploiting declarative constraints, we eliminate the need for re-training the model for new incoming data; because we can simply add more constraints in the inference part without changing the prediction problem. Furthermore, we exploit the global context of the image by learning scene-label association weights to increase the probabilities of the most confident labels, and limit the

number of labels that need to be explored.

1.2.3 Semi and Weakly Supervised Semantic Segmentation Using GAN

Although modeling dependency between regions as well as labels and taking into account the context are important to improve the results in scene labeling, nevertheless pixel (local) classification is essential to achieve high performance in semantic segmentation and recent methods aim to benefit from the deep neural networks to this end. Even though recent deep methods have been demonstrated to be a valuable tool to classify image pixels [17], [88], semantic segmentation is still not fully solved. Deep networks require large annotated visual data that, in this case, should be at the pixel-level (i.e., each *pixel* of training images must be annotated), which is highly prohibitive to obtain.

An alternative to supervised learning is unsupervised learning that exploits a large amount of available unlabeled visual data. Unfortunately *unsupervised* learning methods have not been very successful for semantic segmentation, because they lack the notion of classes and merely try to identify consistent regions and/or region boundaries [155].

Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning, where in addition to unlabeled data, some supervision is also given, e.g., some of the samples are labeled. In semi-supervised learning, the idea is to identify some specific hidden structure – $p(x)$ from unlabeled data x –under certain assumptions - that can support classification $p(y|x)$, with y class label. In this thesis, we aim to leverage unlabeled data to find a data structure that can support the semantic segmentation phase. In particular, we exploit the assumption that if two data points, x_1, x_2, are close in the input feature space, then the corresponding outputs (classifications), y_1, y_2, should also be close (smoothness constraint) [15]. This concept can be applied to semantic segmentation, i.e., pixels lying on the same manifold should be close in the label, thus should be classified in the same

class. This means that unsupervised data acts as a regularizer in deep networks, thus improving their generalization capabilities.

Under the above assumption, in this approach [139], we employ GAN [47] to support semi-supervised segmentation by generating additional information useful for the classification task. GANs have recently gained a lot of popularity because of their ability in generating high-quality realistic images with several, documented, advantages over other traditional generative models [61].

In our GAN-based semi-supervised semantic segmentation method, the generator creates large realistic visual data that, in turn, forces the discriminator to learn better features for more accurate pixel classification. Furthermore, to speed up and improve the quality of generated samples for better classification, we also condition the GANs with additional information – weak labels – on image classes.

In our formulation of GAN, we employ a generator network similar to [115], which, given a noise vector as an input, generates an image to be semantically segmented by b) a multiclass classifier (our discriminator) that, in addition to classifying the pixels into different semantic categories, determines whether a given image belongs to training data distribution or is coming from generated data.

In Chapter 6 we extend the traditional GAN, wherein the discriminator distinguishes real images from fake images, to a framework in which discriminator is a fully convolutional multi-class pixel classifier. This network assigns a label y from the K semantic classes to each image pixel or marks it as a fake sample ($K+1$ class). Our experimental results validates the idea that generative model can improve the classification performance on several benchmark datasets.

1.3 Dissertation Organization

This dissertation is organized as follows: In Chapter 2, we introduce the related works for saliency detection, semantic segmentation in images and semi-supervised learning; in Chapter 3 we describe our proposed method to acquire saliency maps and salient regions in video using group lasso regularization. In Chapter 4, we present the proposed method for using sparse precision matrix in scene labeling and report improved results on publicly available datasets; in Chapter 5, we describe our constrained knowledge based method for semantic segmentation and report the obtained results. And lastly in Chapter 6, we present a proposed semi-supervised method for semantic segmentation using GAN framework and demonstrate the results on public available datasets; and we conclude and discuss future works in Chapter 7.

CHAPTER 2: LITERATURE REVIEW

In this chapter, we present a summary of related work on visual saliency detection and semantic segmentation, two parts of this thesis. Through visual saliency detection important parts of images and spatio-temporal regions of video can be identified. These *regions and tubes are where we humans will pay most attention to, when viewing the images or videos. Generic segmentation methods also partition an image into meaningful* regions, which may correspond to background and foreground (objects) present in the scene. Each region is a connected group of pixels and the same ID is assigned to all pixels in a region. However, object or semantic labels are not assigned to the regions. On the other hand, semantic segmentation methods assign a *semantic* label e.g. sky, car, tree etc to each pixel. Semantic segmentation has been widely investigated in past years. Different methods proposed in literature differ in terms of the features and descriptors they use, primitive elements (pixels, patches or regions) they apply the method on, classifiers they employ to assign labels to these primitive elements, and how they utilize context. In addition, deep learning methods have been vastly exploited to classify pixels in the images. In the following sections, we introduce these techniques concisely.

Saliecny detection methods, based on how they define salient objects, can be categorized into two groups: bottom-up and top down approaches. Bottom-up models mainly employ low-level cues such as intensity, color, texture, and try to single out regions which show distinct characteristics from their surroundings. While, top down approaches are task-oriented and attempt to locate a target object from a specific category; they rely on features of the object of interest. In this chapter, we briefly review these methods.

2.1 Visual Saliency

2.1.1 Bottom-up Approaches

Most saliency detection models in the bottom-up category are biologically inspired and follow the Feature Integration Theory of Treisman [151]. This suggests that when perceiving a stimulus, features are registered early, automatically, and in parallel, while objects are identified separately and at a later stage during processing. These bottom-up methods decompose visual input into separate low-level feature maps, e.g. orientation, contrast, and color. For every individual feature, a different map is computed and normalized. Then, a saliency map is formed by the weighted combination of them. Peaks in the map reflect the attention (saliency) [65]. Recently, some new mathematical and statistical tools (e.g [65], [13]) have been used in order to obtain precise results, and these methods have been mostly evaluated on eye movement data provided by gaze location of viewers.

On the other hand, top-down approaches are mostly investigated by cueing experiments, in which a "cue" brings one's notice to the target. The cue could be what the target is or where it will be [38]. Note that while bottom-up approaches are mainly driven by the visual characteristics of a scene, top-down models mostly define attention models according to the task of interest. Gao [43], [44], [42] proposed top-down approaches which used decision-theoretic models. They introduced the concept of discriminant saliency, which is based on the definition of the target and null hypotheses. They defined top-down saliency as a classification task with which locations where a target could be distinguished from a non-target, with minimum error, is categorized as salient.

Saliency detection approaches can also be categorized on the basis of the techniques that they use to obtain saliency maps. For instance, different computational principles such as Information theoretic models and Bayesian models have been employed in bottom-up saliency methods

to define the concept of saliency. Some approaches use information theory to determine "distinctiveness". Bruce and Tsotsos [12] proposed a model, which tries to find the most informative locations by maximizing Shannon's self-information from local visual feature vectors. To find these features, Independent Component Analysis (ICA) is applied on small RGB patches from the image. The probability of detecting RGB values for a particular patch is determined using ICA bases likelihood. The same authors in [13] further elucidate saliency as self-information of the visual features, by extending the method to find a joint-likelihood. In doing so, each ICA coefficient captures a probability based on its likelihood from the probability distribution of surrounding patches coefficients. The joint likelihood for a particular region is found by the product of all comprised likelihoods. To find saliency map, the joint likelihood is converted to Shannons measure of Self-Information. The attention model and eye movement prediction on complex scenes have been formulated using Bayesian methods as well. Using these methods, prior knowledge about the scene, such as visual attribute statistics or descriptions, can be combined with layout. Itti and Baldi [63], developed a metric for surprise by calculating the mismatches between viewer expectations and perceived reality. This method finds the saliency map by applying center-surround linear filters on different feature channels, such as color and intensity. This approach is only advantageous in pin pointing the focus of the scene if one of the features is distinct, and not so if all the features perform evenly.

Likewise, the Bayesian framework has been employed by model SUN [172] and [128] to study fixations. The SUN model attempts to detect saliency by estimating the probability of presenting a target given visual features at every location in the scene. In a free viewing condition, where there is no notion of target, this model also finds bottom-up saliency using a maximum information approach. Unlike [12] it obtains self-information by finding differences between a particular image's statistics and natural images' statistics. The challenge here is the cluttered background. Consider the case when the salient parts have simpler context in comparison with non-salient parts,

the entropy of the former would be lower, due to the fact that they have been obtained locally. Seo and Milanfar in [128] and [129] also proposed to compute some local descriptors, called local regression kernels, from images or videos to measure the similarities of a pixel or voxel to its surroundings. Visual saliency is estimated using "self resemblance" measures. Therefore, a saliency map is attained, wherein salient regions are determined by dissimilarity (using matrix cosine similarity) compared to their surroundings. This method only compares local neighborhoods and so it suffers from the aforementioned problem of local estimation.

2.1.2 Learning Based and Top Down Methods

Learning techniques which infer the model structure from the data have also been employed in visual saliency modeling. Kienzle developed operators to detect saliency from human eye movement data using machine learning techniques, employing the pixel intensities of static scenes [70] and Hollywood movies [69]. They showed that learned discriminative features have a center-surround pattern. Judd [68] also proposed a top-down method, in which a model of saliency based on low, middle and high-level image features (computed by some saliency methods) is learned from eye tracking data on static scenes. Liu [87] proposed a supervised method that uses learning to detect salient objects. Databases of manually labeled images and video segments were used for the learning task. Learning based methods are unfeasible as they not only rely on eye tracking data and manual labeling, but are also heavily dependent on training data.

2.1.3 Saliency Detection in Videos

Several recent works also deal with the extensions and applications of image saliency detection methods to videos. Guo [53] proposed spatiotemporal saliency detection in frequency domain by extending a two-dimensional Fourier Transform to a quaternion Fourier Transform. Zhang in [171]

extended their model to videos by applying spatio-temporal filters on video frames and computing the features. The bottom-up saliency map is then computed using these features. In [91] spatio-temporal cuboids are modeled by using dynamic textures based on the center-surround contrast hypothesis. In [170] a spatiotemporal video attention detection technique was proposed to detect attention regions and interesting actions in video sequences. Interest-point correspondences and geometric transformations between images are used to compute the motion contrast in the scene. For the spatial attention model, a pixel-level saliency map is computed using color histograms.
Some more current methods attempt to learn a model from gaze data, with the aim of detecting saliency in videos [122] or using obtained saliency maps to accomplish action recognition tasks [102]. These methods are mostly dependent on gaze points, and it is well known that cumbersome amount of effort goes into capturing data from different subjects. Also, the authors in [174] proposed a dynamic consistent optical flow model based on human visual dynamic continuity assumption. They exploit a face detector and spatial saliency models (e.g. [65]) to find a spatio-temporal attention model. Many methods (e.g. [172]) place emphasis on object boundaries and assign high saliency to borders rather than salient regions. In contrast, saliency maps obtained by gaze locations show that the object regions are most frequently the target of interest. To address this issue, we incorporate super-voxels and early video segmentation to saliency detection.

Previous methods (e.g. [157]) mostly depended on training videos and learning features for saliency from these videos. However, the visual features for a region need to be distinctive, irregular and infrequent for a region to be salient. Toward this end, we detect saliency by finding irregularities in videos via sparse representation. Furthermore, our method, which does not require any training videos, is able to deal with cluttered background and videos with noise due to the fact that it does not merely consider local contrast or saliency in small areas. Also, our method is not limited to only capturing moving objects as a salient region; the spatial-temporal information via cuboids and group information from super-voxels yield to obtain salient area more precisely.

2.2 Image Segmentation Techniques

Even though image segmentation has been the subject of many studies for a long time, however, it has not been fully solved due to its challenges such as proper features selection, grouping or homogeneity and scale and granularity. Edge based image segmentation is one of the earliest methods employed in segmentation, in which boundaries between two regions are determined using edges. Threshold based methods are also one of the popular and simple image segmentation methods in this approach based on the difference in pixel intensities of different regions, objects and background are separated; one or more thresholds are defined and the image is segmented into different brightness regions. Region based methods including *region growing* are another category of image segmentation. In the region growing method, starting with seed points, based on some similar properties such as color, texture, etc. , pixels are grouped into larger regions.

Bottom up methods are vastly used in image segmentation, these methods usually cluster nearby pixels using features homogeneity. *K-means* is one of the simplest methods in this category, assuming the number of cluster K is known, K-means selects K random centers, then each pixel is assigned to one of the centers (clusters) based on the feature similarities (distance from the center), finally centers are updated accordingly. *Mixture of Gaussian* is another clustering approach that has been used in segmentation. This method is similar to K-means, except that here each center is a represented by a Gaussian mean and covariance matrix and the parameters are obtained via an EM algorithm [7]. In [166], image segmentation is addressed as a clustering problem using texture features. The distribution of the texture features is modeled using a mixture of Gaussian distributions.

Mean-Shift is another clustering method which unlike previous ones is not parametric and the number of clusters is not needed to be determined beforehand. In this approach, the maxima of a density function, from which presumably data points are drawn, is located and iteratively the

modes of this density are detected. When the method is stabilized and reaches the convergence point, meaning that there are no more changes in the modes, the result is the segmented image [18].

Graph based methods, are more recent and popular methods. Many segmentation methods merely use local features which leads to insignificant segments which make the segmentation noisy due to sensitivity to small changes. However, via spectral graph theory, graph based methods, such as *Normalized Cut* [131], [96] partition the affinity graph which takes the global information into account in grouping.

Graph Based Region Mapping [34] encodes an image into a graph whose nodes are pixels and the edge weights are dissimilarities between pixels. In the beginning each node is a segment, afterward iteratively using the region differences, sub-regions are merged until the difference between regions is more than internal differences. This approach is one of the Minimal spanning tree based methods. The authors present a model to convert over-segmented images (even starts from pixels) to high level segmentation while keeping the cluster features. Hierarchical image segmentation and multi-scale segmentation have been proposed to address the limitation of graph cut and normalized cut approaches. These approaches [19] [130] segment the image using multiple scales graphs from fine to to coarse representation. Iteratively, a subset of nodes in finer graph is selected and joined to one node in the coarser graph. In [3] using local and global information, contours are detected in a given image, then these contours are transformed into a hierarchy of regions.

2.2.1 Object Proposals

Object segmentation is still one of the challenging problems in computer vision. Thus, some studies recently relaxed the object segmentation problem to object proposals, in which a set of regions with high probability of being object is obtained. One can leverage class-specific object

detectors such as Deformable Part Model (DPM) [32] or Poselets [10] to find objects. Also, some researches proposed to use object detection with segmentation to achieve object segmentation. In [82] CRF is applied to refine the bounding boxes obtained from detectors results to generate object segmentation. Borrowing the idea of objectness from [2] some approaches present category-independent object proposal and region proposals [27]. Semantic image segmentation, as a specific problem in image segmentation, has attracted attention of many studies recently. In the following section, we present its related works.

2.3 Semantic Segmentation Approaches

We divide this sections into two subsections; first we review the related work using non deep learning method, which is followed by a review of recent deep learning methods for solving semantic segmentation problem.

2.3.1 Traditional Methods for Semantic Image Parsing

Semantic segmentation methods, depending on their applications, use different type of data. In medical imaging, for instance, CT, magnetic resonance (MR), or ultrasound images, gray level images are utilized, while RGB images are commonly used in most computer vision applications. In recent years, due to the availability of depth images using Microsoft Kinect, RGB-D images also are used.

Typical approaches in this category, start with extracting a fixed size feature vector for each element (pixels, patches, super-pixels or larger segments) of the image. The sliding-window approach has been popular for a certain time, where a classifier is trained on fixed size images then windows (rectangular regions of the image) are given to the classifier [126], [22]. Since this type of ap-

proaches (path classifiers) are applied to a large number of images(patches), some methods such as using stride or interpolation were proposed to make them computationally tolerable. Commonly used approaches, however, use an energy function via MRF or CRF as an alternative. We introduce these methods based on the type of features and the classifiers they use, how they incorporate context and global information and the graphical model they use in the following sections.

2.3.1.1 Features in semantic segmentation

Pixel level features such as **color** are widely used in most of the approaches, even though RGB is very common, some approaches use other color spaces. For instance, HSI which makes the model robust to illumination changes [144] or CIE-L*a*b* color space is used following the idea that human perception of brightness can be estimated by this color space. Region and image level features were used in [56]. RGB color mean and standard deviation and color histogram for each super-pixel are used in [147]. Appearance features including Histogram of Oriented Gradients (HOG), Scale-invariant feature transform (SIFT) and Bag-of-visual-words (BOV) have been used alongside other features in various approaches [147], [20]. **Textons**, which are named after human textural perception and are shown to be beneficial in material classification, have been used in several semantic segmentation approaches [147], [148], [133], [132]. In [133] authors proposed to represent local features using texture-layout filters which model patterns of texture and their spatial layout. In [132], the idea of bag of words is extended to the bag of semantic textons, a histogram of the semantic textons is computed for each region and is combined with a region prior category distribution. Also, features from deep neural networks have been used in recent approaches. Filters from early layers in Convolution Neuronal Networks (CNNs) can be considered as textons as well. High resolution images, or images with higher dimensionality, typically are down-sampled or via a dimension reduction method, for example PCA, their dimensionality is reduced [16].

2.3.1.2 Classifiers in semantic segmentation

Various classifiers have been used in semantic segmentation, either solely to label pixels or patches or in MRF/CRF models to find the unary potentials. **Random Forests** [83] is one of the widely used classifiers for semantic segmentation. In [132] and [67], Random Decision Forest with texton features have been used for semantic image segmentation. In addition, Boosted Decision Trees(BDT) have been used in scene labeling as well. In [147], the authors applied BDT on semantic and geometric labels for regions in images, and concluded that this classifier perform better in geometric labeling. In [45] using boosted decision trees multiple classifiers are trained and combined to achieve a better decision for image parsing.

SVM is another well known classifier which has been employed in semantic segmentation. The authors in [167] attempt to leverage object detectors trained by a support vector machine (SVM) to define a probabilistic model to estimate pixels class labels and object instance labels. Also, SVM with HOG features was used in the 2010 PASCAL segmentation challenge [28], [33]. Tighe *et al.* [148] applied exemplar-SVMs (per-exemplar detectors) [97], which are claimed to perform better on classes with small available training data and high intra-class variance, in order to attain detectors for all categories, including things (objects) and stuff (e.g. sky) and showed that fusion of region-based parsing and detectors lead to improved results. Also, the method in [78] in addition to local features, benefits from object detectors and combines the results from detectors and context information. Moreover, in [54], the authors use detectors to find the bounding boxes of the objects and label regions using information from detectors and surface occlusions. In addition, they use RGB-depth to understand the scene.

In approaches based on **Nearest Neighbors and Retrieval sets**, the image segments are labeled by transferring the labels from a dataset of known labels. To do so, for a given image, similar images are retrieved from a sample data using a nearest neighbor algorithm, then by using Markov random

field model, pixels (or super-pixels) in the image are labeled [85], [147] and [149].

There are many extensions of this type of labeling, for instance, in [25], the authors propose to learn the weights of descriptors in an off-line manner to reduce the impact of incorrect retrieved super-pixels. Also, authors in [135] proposed to use a locally adaptive distance metric to find the relevance of features for small patches in the image and to transfer the labels from retrieved candidates to small patches of the image. In [50], instead of using a retrieval set to transfer the labels, a graph of dense overlapping patch correspondences is constructed; and the query image is labeled by using established patch correspondences.

In [62] authors proposed to represent an image as a collage of warped, layered objects which are sampled from reference images. For a given image, they retrieve a dictionary of object segment candidates that match the image, then represent the image by combining these matched segments. For this purpose, they need a dataset of label exemplars.

2.3.1.3 Context Information

Some of the existing methods aim at finding a graph structure over the image, by using Markov Random Field (MRF) or Conditional Random Field (CRF), to capture the context of an image as well as using classifiers to label different entities (pixels, super pixels or patches) [149] [54] [137].

A majority of methods employ CRFs [78]. These methods use mainly appearance (local features) as unary potential and smoothness between neighboring elements as the pairwise term [133]. In order to integrate potentials of features at different levels (pixels and superpixels), higher order CRF have also been explored [124], [48].

In some other papers, authors incorporate context information in their modeling, using global features of the image or applying co-occurrence of the labels [158]. Authors in [41] exploit different

forms of context based on co-occurrence, spatial adjacency and appearance. While, authors in [40] proposed to combine different levels of local context interactions (at pixel, region and object) and learn a model to integrates appearance features with pixel and region interaction data. Then a conditional random field (CRF) is used to incorporate object level interactions. The authors in [76] propose to incorporate global context efficiently by penalizing unlikely pairs of labels in the Graph Cut model. Additional information, such as long range connections, to refine further the segmentation results have also been proposed [137]. Nonetheless, these methods employ hand crafted features for classification, which makes them hardly generalizable. Deep learning approaches have shown a huge success in semantic segmentation applications recently.

2.3.2 Deep Learning for Semantic Segmentation

Recently, deep learning techniques have also been used in scene labeling. In [30], [136] for each pixel of the image, multi-scaled features are obtained and a neural network is trained to aggregate feature maps and to label the regions with highest scores. Note that these models need a large amount of data for training. In [106], borrowing the early ideas of neural networks, super-pixel proposals (a sequence of nested regions zooming out from super pixels to image-level resolution) are classified using a feed-forward multilayer network.

Convolutional Neural Networks (CNNs) have been very popular recently in many computer vision applications including semantic segmentation. For instance, [106] and [31] leverage deep networks to classify super-pixels and label the segments. More recent methods such as [88] apply per-pixel classification using a fully convolutional network. This is achieved by transforming fully-connected layers of CNN (VGG16) into convolutional layers and using the pre-trained ImageNet model to initialize the weights of the network. In this approach, by implementing skip connection, the model learns to combine coarse, high layer information with fine, low layer infor-

mation to avoid reduction in the accuracy due to down sampling (pooling) in the network. Multiple deconvolution layers [108] have been also employed to enhance pixel classification accuracy.

Post-processing based on MRF or CRF on top of deep network framework has been adopted, as in [17], to refine pixel label predictions. For example, in [127] the error of MRF inference is passed backward into CNN in order to train jointly CNN and MRF. However, this kind of post-processing is rather expensive since for each image during training, iterative inference should be performed.

The authors in [173] proposed an end-to-end training model which interprets Conditional Random Fields as Recurrent Neural Networks. The CRF is formulated by Gaussian pairwise potentials and mean-field approximate inference. Then, the network (CRF-RNN) is connected to a CNN to form a deep network that exploits properties of both CNNs and CRFs.

Segmentation from natural language expressions [60] is a relatively new problem, which is different from traditional semantic segmentation over a predefined set of labels. In order to generate pixelwise segmentation for a language expression (text), this approach proposed to use a recurrent Long Short-Term Memory (LSTM) network to encode text to vector representation and a fully convolutional network to extract a spatial feature map from the image. This model is trained jointly to handle visual and linguistic data. The output of the model (CNN part) is a spatial response map reflecting the target objects.

2.4 Semi Supervised Learning

The aforementioned methods are based on supervised learning and rely strongly on large annotated data, which is often unavailable. Semi-supervised learning initially focused on approaches, where inputs are first assigned to clusters, such that each cluster reflects a specific category [113], [21]. Unlabeled data then can affect the shapes and sizes of these clusters and consequently change

the classification results. Due to the popularity of discriminative methods, in which unlabeled data cannot help, a method proposed to assign probabilistic labels to unlabeled data and train the discriminative model ($P(y|x)$) using the estimated labels. Label propagation techniques are based on the assumption that the labels are uniform locally, and the nearest neighbors are likely to have the same label. Therefore, they propagate labels through regions by choosing the label with highest probability for each instance based on its similarity to its neighbours [143]. In [161], the authors extended the nonlinear semi-supervised embedding algorithms to employ label propagation in deep networks.

Unlabeled data was used in [24] to pre-train a convolutional network. In doing so, in the beginning a class label is assigned to each input image. Afterward, a number of transformation, including scaling, rotation, etc. are applied on the image. Therefore, features that are invariant to the transformations can be extracted. Then, the last classification layer is replaced with a new classifier trained on real labeled data yields in improvement in experimental results.

Deep generative models based on variational autoencoders were proposed in [72] for semi-supervised learning, the authors also presented the stacked version using multiple autoencoders. The authors in [116] following the idea of Ladder network [155], proposed a network whose structure is an autoencoder with skip connections from the encoder to decoder and it denoises representations at every level of the model. Combining the Ladder network model with supervision has shown promising results in semi-supervised classification [116].

Despite the recent progress in semi-supervised learning for classification task, the application of this type of learning has not been fully explored in other computer vision fields. To cope with the annotated data limitation in semantic segmentation, a few number of weakly or semi-supervised semantic segmentation methods have been proposed [111]. These approaches assume that weak annotations (bounding boxes or image level labels) are available during training and that such

annotations, combined with limited pixel-level labels, force deep networks to learn better visual features for classification. In [58], the authors address the semantic segmentation as two separate tasks of classification and segmentation, and assume image level labels for all images in data set and a limited number of fully pixel-level labeled data are available.

While generative methods have been largely employed in unsupervised and semi-supervised learning for visual classification tasks [140], [125], very little has been done for semantic segmentation, e.g., [89]. In [89], authors aim at generating probability maps for each class for a given image, then the discriminator has to distinguish between generated maps and ground truth maps. We propose a method in which we use a semantic segmentation network to find the semantic labels of pixels. We leverage unlabeled data along side generated data, in an adversarial way, to compete in getting realistic labels, and provide the classifier more example to aid it to learn robust features. We use conditional GAN to enhance the quality of generated samples for better segmentation performance as well as to make GAN training more stable.

2.5 Summary

In this chapter, we briefly presented commonly used approaches for saliency detection, including bottom up and top-down methods, in images and videos. Also, we reviewed recent important approaches introduced for image segmentation, such as methods based on minimum spanning trees and hierarchical techniques. Moreover, a review on semantic segmentation approaches is presented. In this context, we reviewed some recent approaches proposed for modeling the structure of image regions and relations between labels. Furthermore, we introduced some methods which use different tools (e.g. context information) to improve the scene labeling. In addition, we also summarized recent methods for semantic segmentation leveraging deep neural networks, which have been very successful. Finally, we reviewed some available semi and weakly-supervised methods which leverage unlabeled data or weakly labeled data to tackle semantic segmentation problem. In the next chapter, we describe our proposed method to find salient regions in a video using an unsupervised approach. In subsequent three chapters, our three different approaches to semantic segmentation are presented.

CHAPTER 3: VISUAL SALIENCY DETECTION USING GROUP LASSO REGULARIZATION IN VIDEOS

Visual saliency is the ability of a vision system to promptly select the most relevant data in the scene and reduce the amount of visual data that needs to be processed. Thus, its applications for complex tasks such as object detection, object recognition and video compression have attained interest in computer vision studies. In this chapter, we introduce a novel unsupervised method for detecting visual saliency in videos of natural scenes. For this, we divide a video into non-overlapping cuboids and create a matrix whose columns correspond to intensity values of these cuboids. Simultaneously, we segment the video using a hierarchical segmentation method and obtain super-voxels. A dictionary of atoms learned from the feature data matrix of the video is subsequently used to represent the video as coefficients of atoms. Then, these coefficients are decomposed into salient and nonsalient parts.

We propose to use group lasso regularization to find the sparse representation of a video, which benefits from grouping information provided by super-voxels and extracted features from the cuboids. We find saliency regions by decomposing the feature matrix of a video into low-rank and sparse matrices by using robust principal component analysis matrix recovery method. The applicability of our method is tested on four video data sets of natural scenes. Our experiments provide promising results in terms of predicting eye movement using standard evaluation methods. In addition, we show our video saliency can be used to improve the performance of human action recognition on a standard dataset. An overview of our proposed model is shown in figure 5.1. We use Robust Principal Component Analysis (RPCA) low-rank matrix recovery [162] method in order to decompose the obtained feature matrix.

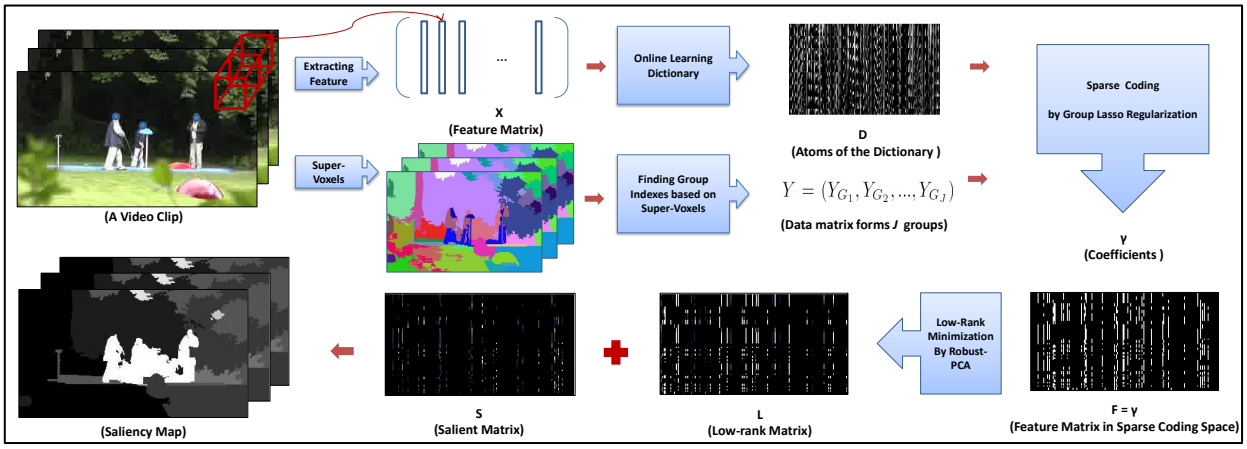


Figure 3.1: An overview of the proposed approach: We begin with extracting the feature matrix, X, of a video, and segmenting the video into super-voxels. A dictionary, D, is learned online. The video is then represented by F in terms of coefficients γ obtained from group lasso regularization over the dictionary. Salient parts, represented by Sparse matrix (S), and non-salient parts (L) are recovered via low-rank minimization technique (Robust PCA). Finally, a saliency map is generated based on the L_1 norm of columns of the matrix S belonging to super-voxels.

3.1 Overview of Proposed Approach

We decomposed a video into salient and redundant parts, where the salient parts are sparse and the redundant parts correspond to homogeneous and highly regular portions of videos. Let F represent a features matrix, whose columns correspond to features from frames of a video. Our aim is to decompose F into low rank matrix L, and sparse matrix S, as follows

$$F = L + S. \tag{3.1}$$

Thus, the problem can be formulated as low-rank and sparse recovery, for which Robust PCA (RPCA) [162] can be used to solve. RPCA attempts to decompose the given matrix F, into the

low-rank matrix and the sparse matrix by solving the following optimization problem

$$\begin{aligned} &\min_{L,S} \ rank(L) + \lambda \|S\|_0 \ , \\ &s.t. \quad L + S = F \ and \ \|S\|_0 \leqslant k. \end{aligned} \tag{3.2}$$

If this problem can be solved for appropriate λ, L and S may be recovered exactly to generate the data F. However, (3.2) is a highly nonconvex optimization problem, and there is no known efficient solution for it. The low rank matrix computation problem and the L_0-minimization problem are both NP-hard and difficult to approximate. Since the formal hardness result for (3.2) is not known, the reasonable guess is that it is NP-hard [162]. By using the relaxed convex alternative, in which L_0-norm is replaced with L_1-norm and the rank with the nuclear norm, a tractable optimization problem is obtained,

$$\begin{aligned} &\min_{L,S} \ \|L\|_* + \lambda \|S\|_1 \ , \\ &\quad s.t. \ L + S = F, \end{aligned} \tag{3.3}$$

where $\|L\|_*$ is the nuclear norm of L and $\|S\|_1$ is L_1-norm. The rank of a matrix is the number of nonzero singular values, so an alternative for the rank function in (3.2) could be a nuclear norm, which denotes the trace norm of the matrix, then (3.3) minimizes the sum of the singular values over the constraint set.

The main objective here is to find a feature space in which the assumption of non-salient parts being low-rank and salient parts being sparse remains valid. The connection between sparsity and saliency is due to the fact that the human vision system is attracted to informative rare scene regions and processes merely a small amount of the entire observed information [73, 8]. Hence, we use sparse representation as mid-level features. In addition, correlation between redundant parts is retained via group lasso regularization (section 3.2.2). In the following sections, we describe

different steps of the proposed approach to obtain the appropriate features, and finally find the saliency maps.

3.2 Feature Space Selection

In this section we explain our method to obtain the feature matrix in order to decompose it into low-rank and salient matrices.

3.2.1 Low-level features

We divide a given video into non-overlapping cuboids of size $p \times q \times t$ and construct matrix $X = [\mathrm{x}_1, ..., \mathrm{x}_n] \in R^{m \times n}$, where x_i is the visual feature vector (e.g. intensity) from cuboid i.

Motivated by neuroscience studies which show that sparse coding successfully simulates the V1 population responses to natural stimuli (e.g. [109], [110]), we propose to model videos of natural scenes as sparse representation. The idea is to represent observed data, i.e., vectorized cuboids, in terms of a linear combination of bases of a known dictionary. Assume $D = [\mathrm{d}_1, ..., \mathrm{d}_k] \in R^{m \times k}$ is a dictionary matrix. We can represent x_i as follows

$$\mathrm{x}_i = \sum_{j=1}^{k} \mathrm{d}_j \beta_{ji} + \varepsilon, \tag{3.4}$$

where d_j is an atom of the dictionary, β_{ji} is the corresponding coefficient, a scalar value, that needs to be found, and ε is a Gaussian noise. We can rewrite (3.4) as

$$\mathrm{x}_i = D\beta_i + \varepsilon. \tag{3.5}$$

Therefore, x_i is represented by $\beta_i = [\beta_{1i}, ..., \beta_{ki}] \in R^{k\times 1}$ in the sparse coding space. In other words, each data point is represented as a sparse linear combination of the atom vectors in the dictionary.

Although the popular loss function used for regression problems is the Least Squares Error (minimization of residual sum of squared errors) with a penalty on the $L2$-norm regularization as follows,

$$\min_{\beta} \|X - D\beta\|_2^2 + \lambda\|\beta\|_2, \tag{3.6}$$

it does not impose sparsity, and the resulting coefficients have non-zero values.

To address this issue, we use lasso, proposed by [145], replacing $L2$-norm regularization with L_1-norm and formulate it as follows

$$\min_{\beta} \|X - D\beta\|_2^2 + \lambda\|\beta\|_1, \tag{3.7}$$

where X is a matrix of observed data, D is a given dictionary of bases, and $\beta = [\beta_1, ..., \beta_n]$ is a $k \times n$ coefficient matrix, where each column is a sparse representation for a data point. In (3.7) $\|.\|_1$ denotes the entry-wise matrix L_1-norm ($\|\beta\|_1 = \sum_{i=1}^{n} \|\beta_i\|_1$), and λ is a regularization parameter that controls the sparsity level.

However, if the salient object or region is large, the number of cuboids belonging to the region will be enormous; the cuboids, which we expect to be outliers and indicate saliency, cannot be considered as sparse. It has been shown that the lasso tends to select only one data point (feature vector) from a group of highly correlated data points, and is not concerned with which one is selected [177]. In order to overcome this, we use instead group lasso regularization to find the coefficients, in which group structure of coefficients is determined by super-voxels in a video.

Note that the goal is to select an important subset of variables imposing sparsity among the groups. Intuitively, this should drive all the weights in one group to zero together. With this approach, not only would the noise be suppressed, but also the variation in the features for finding saliency would not be as large as the sparse representation based on individual cuboids.

3.2.2 Group Lasso Regularization

We can formulate our problem as a general regression,

$$Y = D\gamma + \varepsilon, \tag{3.8}$$

in which, Y is a low-level feature matrix the columns of which are vectorized cuboids from the video. Y is constructed from the X matrix in a way that each division of Y consists several columns of X. D is the dictionary and γ is a coefficient matrix. Assume that Y, the feature matrix, is structured in J disjointed groups $\{G_1, G_2, ..., G_J\},\ G_i \cap Gj = \varnothing$, and is represented as $Y = (Y_{G_1}, Y_{G_2}, ..., Y_{G_J})$ where $Y_{G_j} = (X_1, X_2, ..., X_{n_j})$, in which group indices are determined by the super-voxels in the video.

The group lasso is an extension of the lasso which assumes covariates are clustered in groups [36]. It aims to obtain a regularization of the empirical error that finds a sparse solution to preserve the groups of variables together. It solves the optimization problem via $L_{1,2}$-regularization, which imposes sparsity on groups by using the sum of Euclidean norms of coefficients in each group instead of L_1-normof each single coefficient. This could drive all the coefficients in one group to zero together, and can result in group selection [168]. The group lasso regularization problem

would be as follows

$$\min_{\gamma} \|Y - D\gamma\|_2^2 + \lambda\|\gamma\|_{1,2}, \tag{3.9}$$

where $\gamma = [\gamma_{G_1}, \gamma_{G_2}, ..., \gamma_{G_J}]$ is the matrix of coefficients that must be obtained, γ_{G_j} is a division of γ that corresponds to the j_{th} group of coefficients and consists of several columns of the coefficients matrix, and $\|\gamma\|_{1,2} = \sum_{j=1}^{J} \|\gamma_{G_j}\|_2$. The parameter λ determines the level of group sparsity to be imposed in the solution. This model assumes group structure is given. In our case, the structure is provided by super-voxels in a way that cuboids indicate feature vectors and each super-voxels consists of a group of cuboids.

3.2.3 Dictionary Learning

In sparse coding, we want to approximate a signal **x** over a *dictionary* **D** (which has k columns referred to as *atoms*) in such a way that the obtained signal as a linear combination of a *few* atoms is as close as possible to **x**.

Various types of dictionaries have been used for this task, for example, a predefined dictionary which is based on different wavelets for natural images ([120], [98]). An alternative approach determines the dictionary from the training samples using techniques such as Principal Component Analysis (PCA) and Generalized PCA. These algorithms, nevertheless, generate unstructured dictionaries which are computationally expensive to apply and limit the size of the learning dictionary because of its complexity [121]. Therefore, sparse dictionaries, which are structured based on a sparsity model, have been proposed to be used in sparse signal approximation. These dictionaries perform with significantly more efficiency and function better for larger dictionaries and higher-dimensional data [121]. It has also been shown that learning a structured dictionary improves

signal reconstruction and results in a better representation [26].

Most algorithms for dictionary learning are batch-based, which access the whole data at each iteration and cannot handle large data efficiently. We resolve this by using an *online* approach that processes mini batches and uses sparse coding in the optimization procedure to find atoms. This method reduces memory consumption and lowers computational cost, hence it could be advantageous for image and video processing.

For learning dictionary on a given set of signals, in our case the cuboids of a given video, $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n] \in \mathrm{R}^{m \times n}$, the classic approach is to optimize a cost function

$$f_n(D) \triangleq \frac{1}{n} \sum_{i=1}^{n} l(\mathbf{x}_i, D), \tag{3.10}$$

where matrix $\mathbf{D}$ in $\mathrm{R}^{m \times k}$ is the dictionary whose columns are bases (atoms), and $l(\mathbf{x}, \mathbf{D})$ is the loss function that shows how "good" D is in representing $\mathbf{x}$ via a sparse representation. In the online learning method [94], $l(\mathbf{x}, \mathbf{D})$ is defined as the result of L_1-sparse representation problem

$$l(\mathbf{x}, \mathbf{D}) \triangleq \min_{\alpha} \frac{1}{2} \|\mathbf{x} - \mathbf{D}\alpha\|_2^2 + \lambda \|\alpha\|_1. \tag{3.11}$$

There is a common constraint, call it C, on the dictionary's atoms $\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_k$ having an L2-norm less or equal to one, which prevents atoms from having large values and consequently, coefficients having arbitrarily small values. A convex set of matrices validates this constraint:

$$\mathfrak{C} \triangleq \{\mathbf{D} \in \mathrm{R}^{m \times k} \; s.t. \; \forall j = 1, ..., k, \; \mathbf{d}_j^T \mathbf{d}_j \leq 1\}. \tag{3.12}$$

Since the cost function $f_n(D)$ is not convex with respect to D, it is rewritten as a joint optimization problem with respect to the dictionary D and the coefficients α of the sparse decomposition. While

the function in equation (3.11) is not jointly convex, when one of the two variables D or α are fixed it becomes convex with respect to the other:

$$\min_{D\in\mathfrak{C},\alpha\in\mathbb{R}^{k\times n}} \sum_{i=1}^{n} (\frac{1}{2}\|\mathbf{x}_i - \mathbf{D}\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1). \tag{3.13}$$

To solve this problem, the common approach is alternatively minimizing one variable while keeping the other one fixed. We use SPAMS open source toolbox [92], which implements the aforementioned online dictionary learning method.

3.2.4 Representing Feature Space by Sparse Coding

Once we determine the dictionary, we need to find coefficients by solving the objective function (3.9). Group lasso regularization has been the subject of many studies recently, and several methods have been proposed for solving (3.9) (see [4], [103], [119]). In this section, we use a block-coordinate descent (BCD) approach that is an extension of the classic method to the group lasso [168], where minimization is performed over each group of variables. The BCD method utilizes an objective function that can be efficiently optimized over one group of variables. Each group subproblem can be solved in closed form. Another category of methods is gradient-based methods, in which gradient information is used to optimize the objective function [86]. This also generates subproblems that have closed form solutions. However, [114] has shown that the BCD approach often outperforms the existing gradient-based approaches. In our implementation, we use simultaneous signal decomposition methods based on block coordinate descent, which efficiently solves (3.9) by computing the covariance matrix DD^T first and then $D^T Y_{G_i}$. We then compute a matrix of coefficients using a Cholesky-based decomposition method [92].

3.3 Finding the Saliency Map

Once we transform the data matrix to feature space with sparse coding, low-rank and salient parts are recovered by using Robust PCA. The feature matrix is considered as a combination of non-salient parts in a low dimensional space, and salient objects or motion as sparse portions. Thus, given the feature matrix the augmented Lagrange multiplier method is used for recovering low-rank matrices via optimization equation(3.3), where λ balances rank and sparsity. For an appropriate λ, the F matrix, which is the coefficient matrix computed from group lasso regularization, is estimated properly by obtained L and S matrices. There exist various methods to extract low-rank and sparse matrices by this optimization problem. We use a technique of augmented Lagrange multiplier, named ALM. This method can handle large matrices and has Q-linear convergence speed which makes it suitable for image and video processing applications. This simple implementation iteratively computes a partial SVD of a matrix and converges to the solution in a small number of iterations. The algorithm also has a faster version, i.e the inexact ALM algorithm, which requires a smaller number of partial SVDs [84].

Final step of our approach is in computing the saliency map using the sparse matrix values found in the previous step. The L_1-norm of columns of S matrix, corresponding to cuboids, indicates the saliency value. Then, saliency value of the super-voxels covering these cuboids is obtained by counting salient cuboids and normalizing them based on the super-voxel size. The higher the norm, the more salient the corresponding region.

3.4 Experiments and Evaluation

For evaluating the proposed method, we first generate a saliency map for all regions in each video using the proposed approach. Each saliency map acts like a maximum likelihood binary classifier

for each video, and determines the salient and non-salient regions. After thresholding, regions in the saliency maps that have value greater than the threshold are considered as to belong to the salient class.

3.4.1 Data sets

We have evaluated our method on four different data sets. The first is INB by Dorr *et al.* [23], which consists of 18 high-resolution movie clips of natural outdoor scenes. Each video is 1280 by 720 pixels in size, has 30 frames per second and is about 20 seconds in length. The gaze data of 54 human subjects freely viewing these videos is available. About 40,000 saccades have been extracted from the gaze data using a dual-threshold velocity based procedure. Salient locations are labeled positive by using these saccade points. Because of the latency of the oculomotor system, the gaze response to a salient event is not necessarily matched with the time of the event. Hence, some methods consider a temporal offset. However, [156] have shown the average lag in natural scenes to be near zero, and so there is no need to consider any offset, therefore we do not consider temporal offset.

The second data set is the UCF Sports Action data set [118], which consists of 150 videos from 9 different types of actions such as *Diving*, *Horseback riding* and *Swinging*. The gaze data for this data set, including eye fixation information from 16 subjects viewing the videos, is available via [102, 101].

Third data set is our own, UCF Saliency data set, which is a more challenging data set. In this data set the resolution of videos, unlike INB and UCF Sports, which have high resolution videos, is low and camera motion could be problematic. This data set consists of 6 different videos from different events, such as Person Running, Moving Car, Jumping and Sailing. In figure 9, a set of still frames and their corresponding results are shown. In order to find ground-truth saliency maps,

we asked 4 subjects to mark freely some points (the average is 6), on regions in each frame of the video which they believe are important in understanding the scene or are interesting and capture the attention. This was done by using an annotation tool which was developed in our group, and subjects did not have any prior knowledge about the video.

The last data set that we have tested our method on is Hollywood2 Actions dataset [100]. This is a large scale dataset with camera motion and clutter, which consists of 2517 videos of which 884 are selected as a test subset. Human fixations from 16 subjects are also available for this data set [102, 101].

3.4.2 Evaluation Methods

In order to compare our method with [157], we perform the same experiments, which they regard as *Bias-Free*. In doing so, we consider the set of saccade landing points in a video as a positive class, and randomly selected gaze locations from different videos are considered as a negative class. Since this labeling method leads to overlap between positive and negative samples, another labeling model has been proposed called *Default-Labeling*.

In *Default-Labeling*, for each video an empirical saliency map is generated using gaze locations. These maps specify the density of the gaze points via all subjects. At each gaze point a spatiotemporal Gaussian is placed, and for all subjects these Gaussian filters are superimposed. We use the same Gaussian filter with a spatial support of 2.4 degrees of the visual angle, of 0.17 seconds temporal support, and standard deviations of 0.6 degrees (spatial) and 600 ms (temporal). In this case, positive samples are selected from the highest density of the eye movement data in the empirical saliency map and negative class samples are picked from the lowest density. After thresholding, these saliency maps are treated as ground truth data, and for quantitative analysis we report ROC Scores (AUC: area under curve). Our results are compared with ground truth data and AUC is

reported for each video.

Since studies show the probability of directing attention in the center of a scene is higher, as a post-processing step we apply a Gaussian filter to smooth the map and emphasize the center in terms of saliency values. Generally, this step leads to better results in terms of predicting eye movement locations among all videos, even though for some videos AUC scores get slightly reduced.

3.4.3 Implementation Details and Computational Complexity

One of the primary steps in our method is grouping similar voxels in videos into meaningful segments called super-voxels. For finding super-voxels in a video, we use the Efficient Hierarchical Graph-Based Video Segmentation method. Basically, it is a spatiotemporal segmentation approach that uses hierarchical graph-based algorithm [52]. This method is chosen based on [164], which using existing benchmarks, evaluates several video segmentation methods and concludes that the a hierarchical graph-based method is one of the best in terms of accuracy and efficiency.

In parallel, by finding super-voxels, we extract intensity feature vectors from the video cuboids. The size of the cuboids in our experiment is $4 \times 4 \times 4$. Afterward, a dictionary is created on the video feature vectors via Online Dictionary Learning for Sparse Coding method. We use SPAMS (SPArse Modeling Software) optimization toolbox for this purpose. The parameter that needs to be tuned in this phase is the number of dictionary atoms. Since the dictionary is over-complete, the number of bases must be greater than the vector size, which is 64 in our case. Therefore, we tried different numbers such as 100, 300, 640 and 1000 and found empirically that 640 is the most proper choice as a trade-off between efficiency and effectiveness. The mentioned toolbox also is used for obtaining sparse coding through group lasso regularization.

Computational Complexity: As shown in figure 5.1, our method consists of several steps iden-

tified by different blocks. Therefore we analyze the computational complexity of each step separately. The initial step is segmentation, which is linear in terms of n number of pixels in the whole clip. For learning a dictionary, we used the online learning method which solves the problem by optimizing dictionary atoms and coefficients iteratively. When the dictionary is fixed, k lasso problems are required to be optimized, where $k = n/64$ is the number of cuboids in our case, which is a fraction of number of pixels. And for a fixed coefficient matrix, optimizing the dictionary is a least squares problem of pm variables and m constraints, where p and m are respectively dimensions of data matrix and number of atoms in dictionary which are constants. For instance, in our experiments with cuboids of size $4 \times 4 \times 4$, p is 64 and m, the size of the dictionary is 640, therefore this part has linear time complexity as well.

In the last part of method, which is matrix decomposition, IALM method is used. This is a fast implementation of Robust PCA which has the complexity of $O[min(nm^2, mn^2)]$ where, in our method $m \ll n$, so the algorithm has linear complexity.

In our implementation, the dictionary and super-voxels are created in parallel then the saliency map is obtained. We have written our program in MATLAB code and have used a system with Intel(R) Xeon(R) CPU which has 6 cores and 12 threads with Windows 7 operating system. The memory of our system is 24 GB, and regardless, the program uses at most 12 GB for 12 threads. Using this configuration, finding the dictionary and generating the saliency map takes 0.83 s per frames, in other words the rate of generating the saliency map is 1.2 frame of size 320×240 per second. Rewriting the code in C++ or a faster platform than MATLAB could aid in the performance of the program.

Table 3.1: Our results in comparison to state of the art methods in bias-free configuration: This table summarizes the performance of our method on INB data set, in terms of AUC, comparing with the Bayesian surprise [64], SUNDAy [171] and Intrinsic dimensionality methods [157]. GL is our method and GL-S[mooth] shows the results after smoothing.

Video	Surp	SUN	Intr.K	GL	GL-S
Beach	0.61	0.65	0.71	0.63	**0.78**
Breite strasse	0.70	0.70	**0.76**	0.60	0.75
Bridge1	0.52	0.50	0.59	0.70	**0.75**
Bridge2	0.64	0.60	0.53	0.72	**0.75**
Bumblebee	0.54	0.56	**0.63**	0.54	0.57
Doves	0.71	0.72	**0.83**	0.77	0.80
Ducks Boat	0.65	0.63	**0.70**	0.62	0.54
Ducks Children	0.56	0.70	**0.78**	0.65	0.66
Golf	0.67	0.77	0.77	0.81	**0.82**
Holsten Gate	0.51	0.61	0.66	**0.79**	0.75
Koenigstrasse	0.60	**0.62**	0.60	0.61	**0.62**
Puppies	0.71	0.65	**0.75**	0.68	0.70
Roundabout	0.62	0.63	0.70	0.62	**0.72**
Sea	0.83	0.84	**0.86**	0.74	0.74
St Petri Gate	0.56	0.51	0.60	0.58	**0.66**
St Petri Market	0.52	0.58	0.63	0.74	**0.82**
St Petri McDonald	0.51	0.57	0.50	**0.60**	0.57
Street	0.58	0.68	0.77	0.78	**0.81**
Average	0.61	0.64	0.69	0.67	**0.71**

3.4.4 Results

In table 3.1, the results of our method and comparison with other methods using Bias-Free labeling, are reported separately for each video. As we can see, after smoothing, our final results outperform the state of the art. Also, even with no post processing (smoothing) our results are reasonable and encouraging and better than other two unsupervised methods. Moreover in some videos, on which other methods work poorly, such as St petri market and golf videos, we obtained better performance. It should be noted that average AUC value for empirical saliency maps using Bias-

Free labeling for determining saliency locations is 0.79.

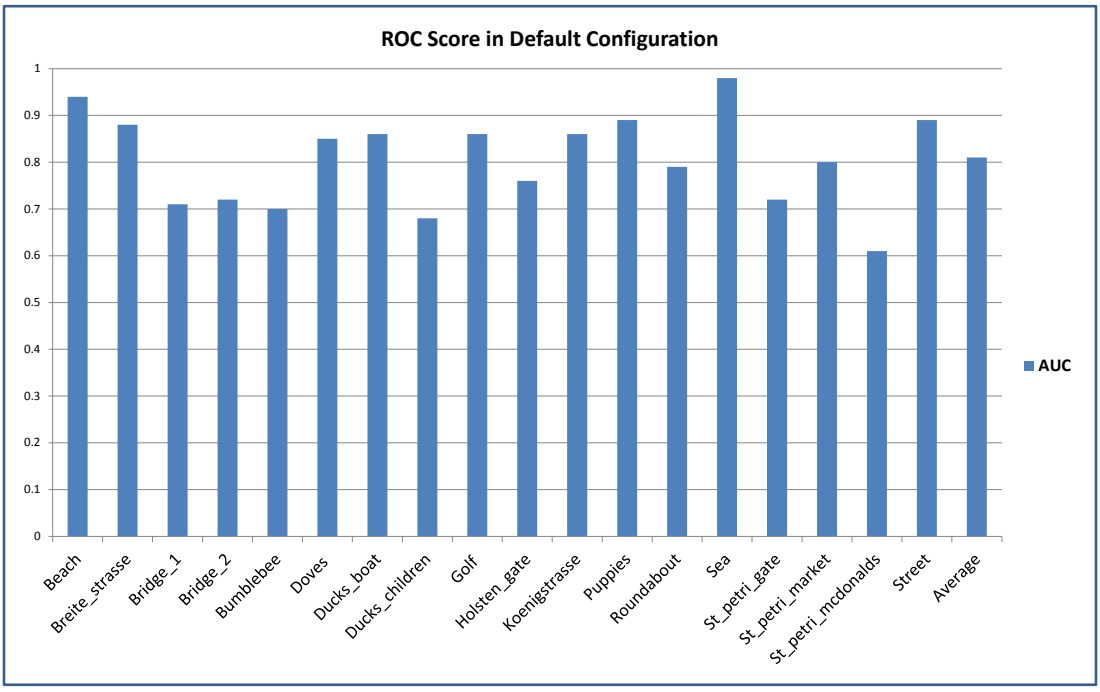


Figure 3.2: The results of Default labeling for each video using our method and smooth version. The performance improvement over bias-free labeling is remarkable.

Furthermore, the AUC score obtained for each video via the Default labeling model is reported in figure 3.2. Most of the videos perform noticeably better in terms of ROC scores. Also, figure 3.3 shows baseline methods, and results obtained by our method using Default labeling. In this case, the empirical average of saliency is the upper-bound with the value of AUC being 1. Some example of qualitative results, sample frames and their saliency maps, are depicted in figure 3.4, 3.5, 3.6, and 3.7. We have also tested our method by performing experiments on UCF Sports Action dataset [118]. We have used the same experiments setup as aforementioned in the configuration for INB data set. In doing so, for Bias-Free experiments, we have used gaze points from current videos as the positive samples, and randomly selected fixations from different classes of videos as

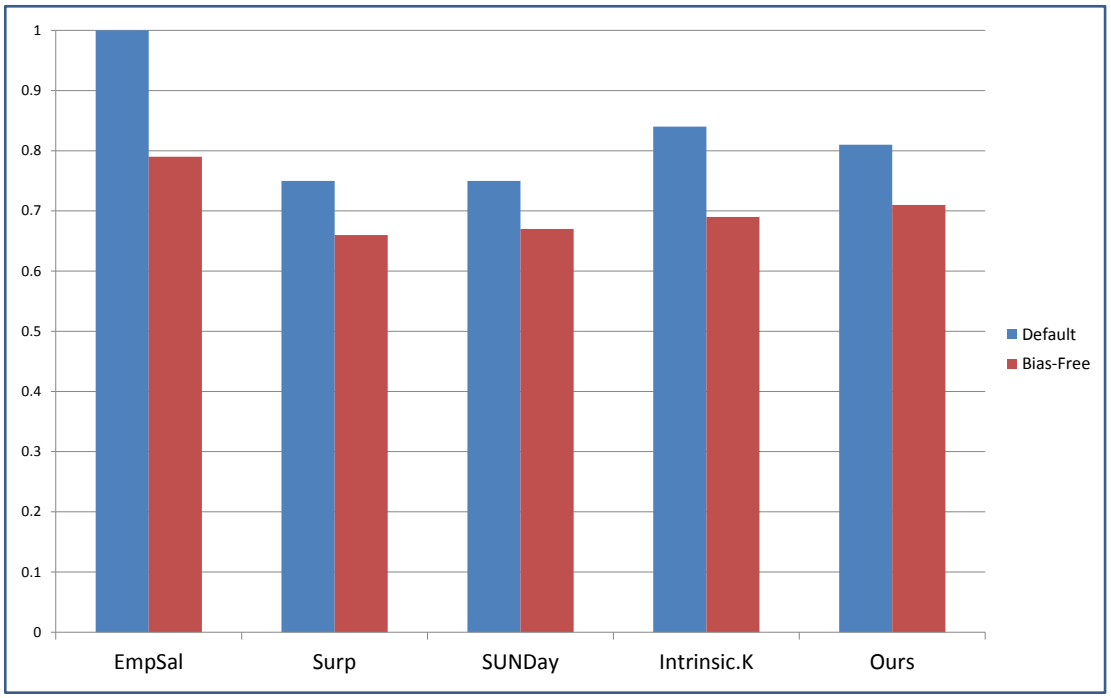


Figure 3.3: Average AUC of the empirical saliency for the baseline methods: Bayesian "surprise" [64], SUNDAy [171], Intrinsic dimensionality methods [157] and our model.
the negative ones.

Since the measurements have some errors and calibration errors are provided to ensure that the data is accurate, and to get the likely positions of point-of-regard, we have used gaze samples where the calibration error is less than 0.5. For evaluation based on the Default-labeling method, a probabilistic distribution of the gaze point is required.

Therefore, we create a Gaussian model with sigma equal to the calibration error for each point, then the top 10 percent of the mixture of Gaussian are considered as salient parts. The quantitative results for these experiments for our method as well as Bayesian-surprise [64] and SUNDay [171] are presented in figures 3.9 and 3.10.

In figures 3.11 and 3.12 some frames from sample videos, including Swinging, Walking, Diving

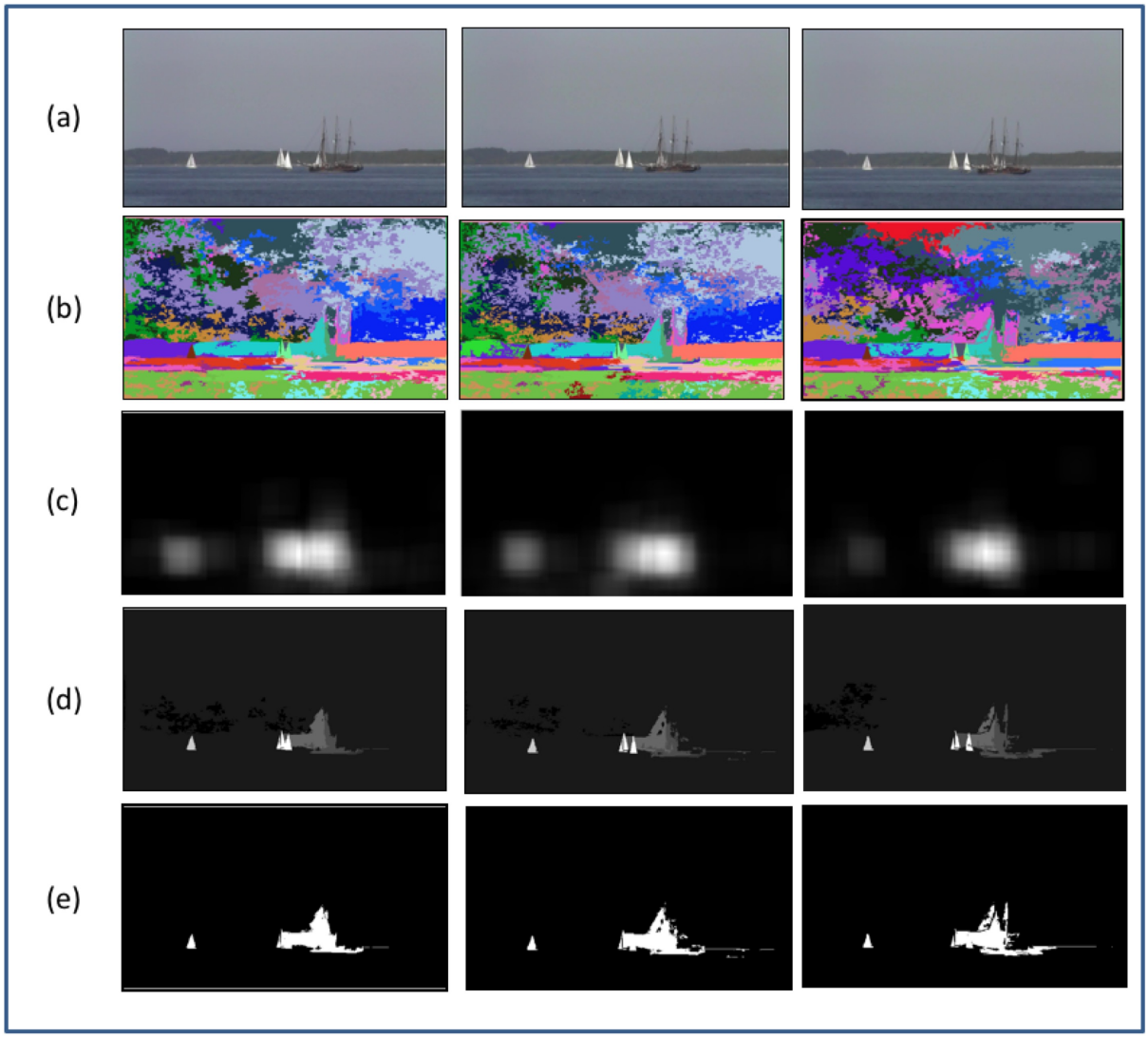


Figure 3.4: Examples of frames from (a) data set videos, (b) super-voxels, (c) empirical saliency maps obtained by gaze data, (d) our saliency map results and (e) binary maps showing the most salient regions. Comparing empirical saliency maps and our results illustrated that the maxima in saliency maps is matched.

and Horse-Riding, the corresponding super-voxels and saliency maps, in which gaze locations are indicated, are shown. Similarly, we have tested our method on the Holywood2 dataset, since our method is unsupervised, we have used only test subset of the data.

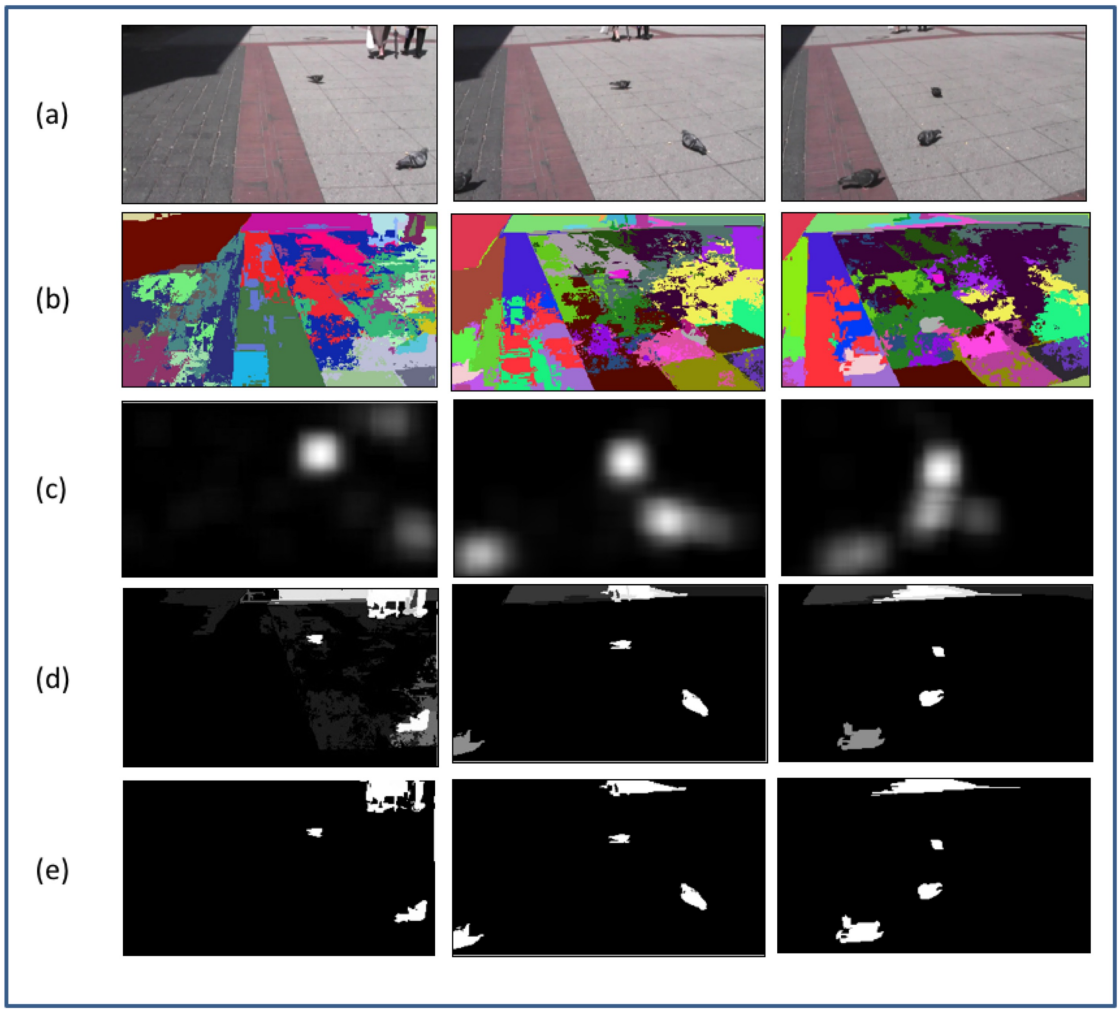


Figure 3.5: More examples of frames from (a) data set videos, (b) super-voxels, (c) empirical saliency maps obtained by gaze data, (d) our saliency map results and (e) binary maps showing the most salient regions. Comparing empirical saliency maps and our results illustrated that the maxima in saliency maps are matched.

For the sake of comparison, we have applied the SUNDay and Bayes Surprise method on this data set. As figure 3.8 indicates, the proposed method has higher performance in terms of AUC scores, and the SUNDay method has the lowest due to more emphasis on the edges and borders of regions, which is misleading in a cluttered background.

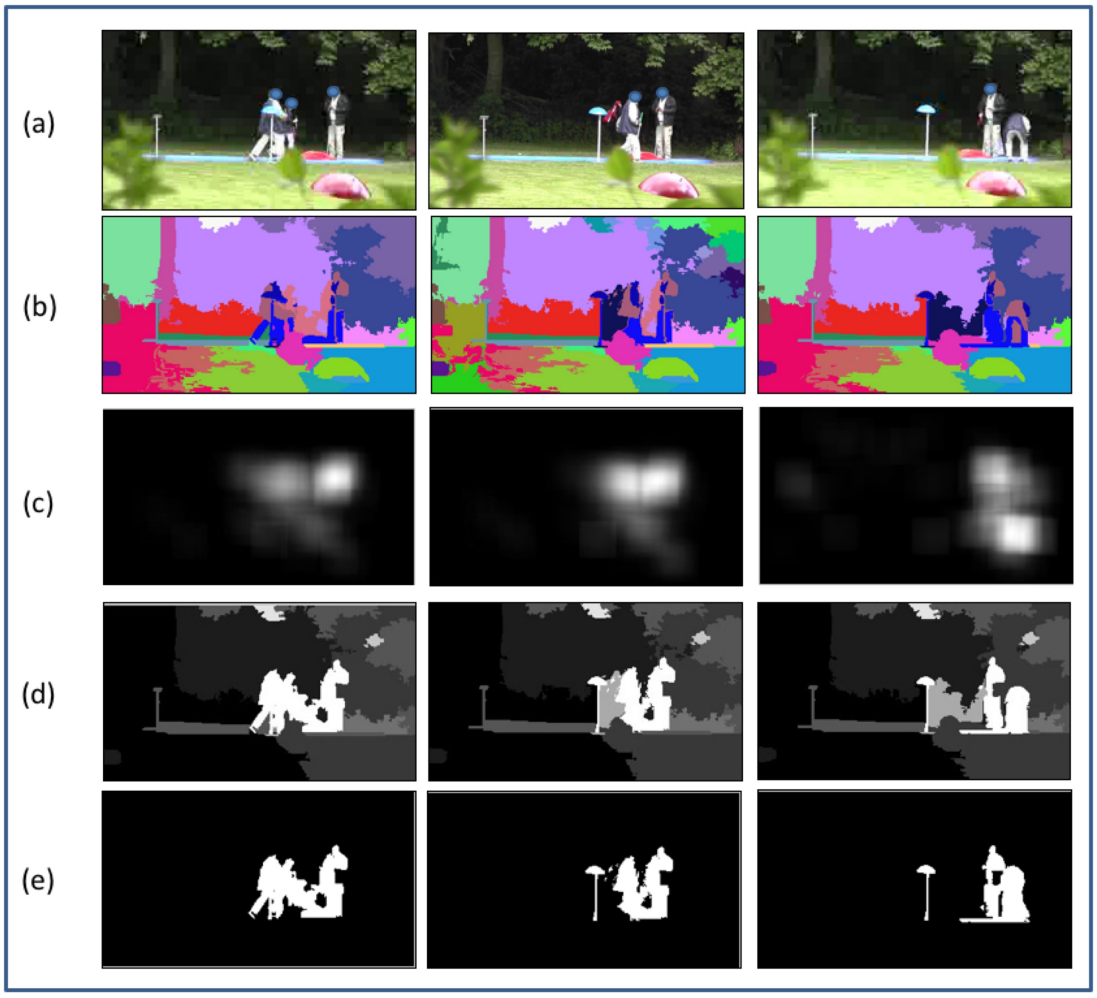


Figure 3.6: More examples of frames from (a) data set videos, (b) super-voxels, (c) empirical saliency maps obtained by gaze data, (d) our saliency map results and (e) binary maps showing the most salient regions.

In figs. 3.13–3.16, a set of still frames and their corresponding results from the UCF Saliency data set is shown. The results show the saliency maps are in accordance with the saliency distribution obtained by thesalient points. Additionally, quantitative results in terms of AUC scores are reported in figure 3.17.

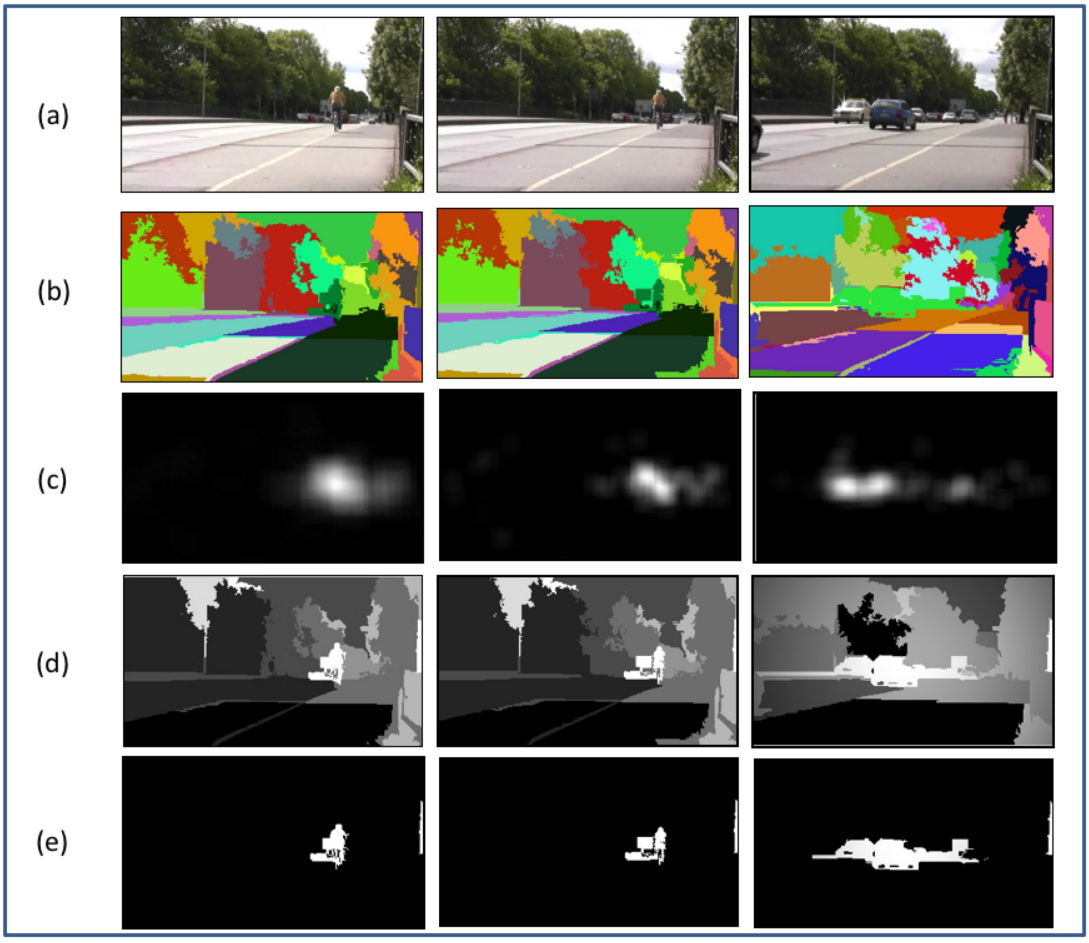
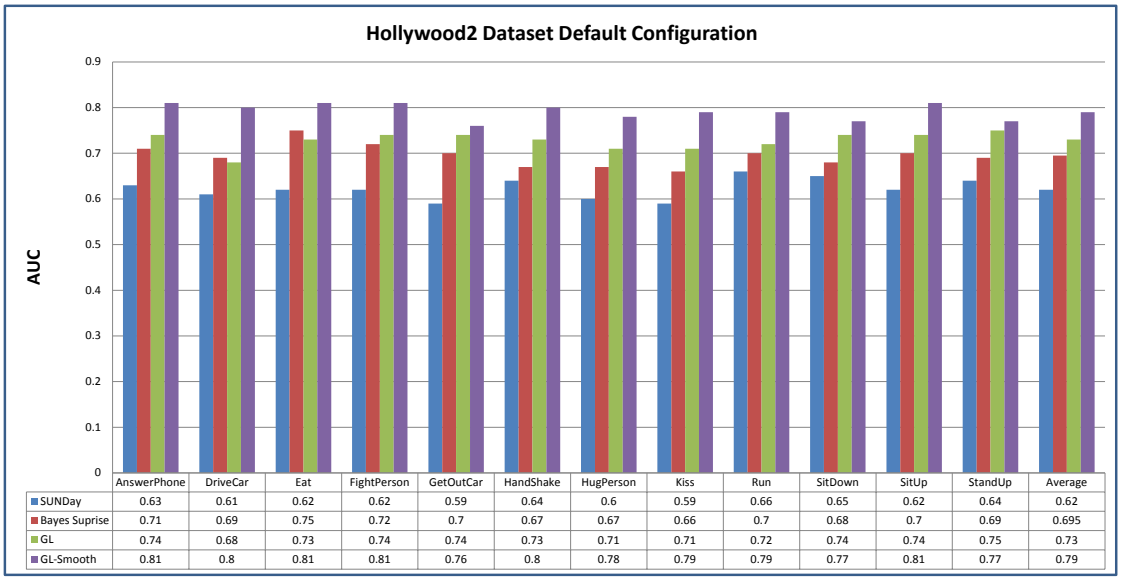


Figure 3.7: More examples of frames from (a) data set videos, (b) super-voxels, (c) empirical saliency maps obtained by gaze data, (d) our saliency map results and (e) binary maps showing the most salient regions.

3.5 Visual Action Recognition

Next we present an application for saliency in an action recognition problem in the UCF Sports data set. Local spatio-temporal descriptors are being widely used for action recognition in videos. In these experiments we use saliency maps to prune these features, and we show that even after discarding roughly 30 percent of descriptors, the method still outperforms the baseline. We use



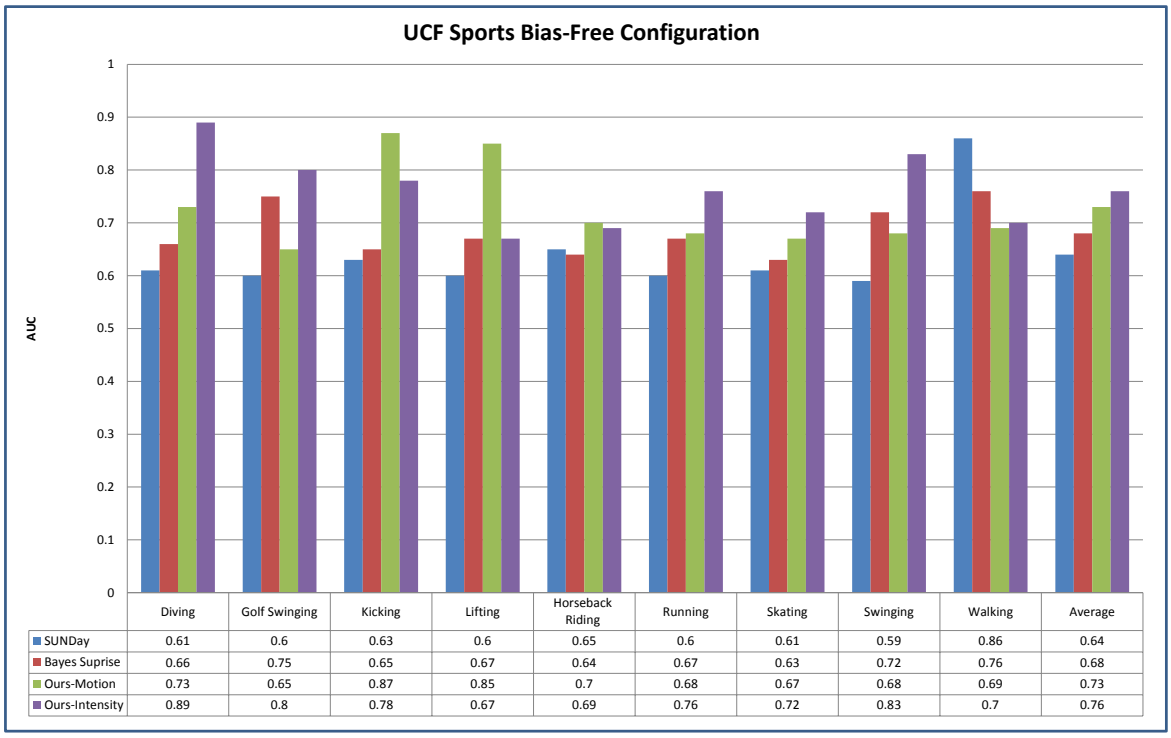
	AnswerPhone	DriveCar	Eat	FightPerson	GetOutCar	HandShake	HugPerson	Kiss	Run	SitDown	SitUp	StandUp	Average
SUNDay	0.63	0.61	0.62	0.62	0.59	0.64	0.6	0.59	0.66	0.65	0.62	0.64	0.62
Bayes Suprise	0.71	0.69	0.75	0.72	0.7	0.67	0.67	0.66	0.7	0.68	0.7	0.69	0.695
GL	0.74	0.68	0.73	0.74	0.74	0.73	0.71	0.71	0.72	0.74	0.74	0.75	0.73
GL-Smooth	0.81	0.8	0.81	0.81	0.76	0.8	0.78	0.79	0.79	0.77	0.81	0.77	0.79

Figure 3.8: AUC scores for videos in Hollywood2 data set based on Default configuration.

the bag of visual words framework for action recognition, which consists of obtaining features by descriptor extraction, K-means clustering and codebook generation, feature quantization and classification using SVM classifier.

For the first experiment, we extract dense space-time interest point descriptors [81], with a 50 percent spatiotemporal overlap, using a single spatial and temporal scale. We use HOG and MBH descriptors for the first experiment. Afterward, we remove the descriptors which do not belong to the salient areas, then we generate a codebook of size 1000, and for each video we compute a histogram using the codebook. For classification, we use a non-linear SVM with a chi-squared kernel. In order to divide the data into a train set and a test set, we use a training-testing split provided in [80]. In this, 103 of 150 videos in the UCF Sports data set are used for training and the 47 remaining videos for testing.

We reproduced the baseline using the same framework, except the pruning part, where we use all

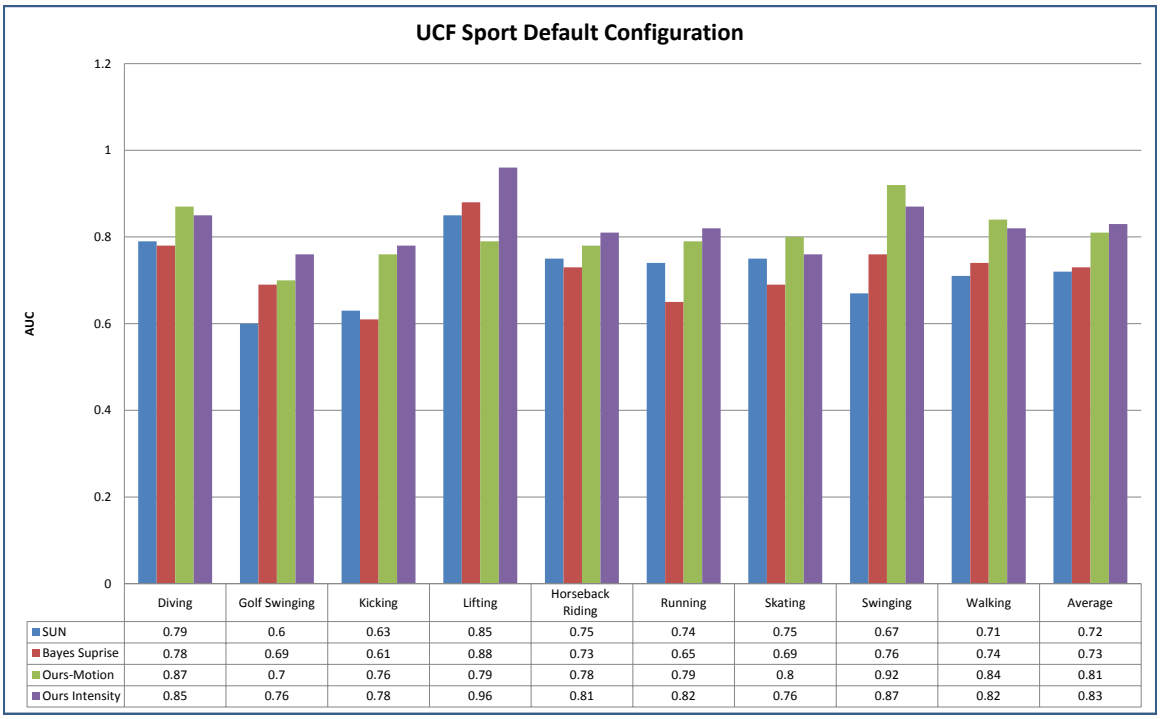


	Diving	Golf Swinging	Kicking	Lifting	Horseback Riding	Running	Skating	Swinging	Walking	Average
SUNDay	0.61	0.6	0.63	0.6	0.65	0.6	0.61	0.59	0.86	0.64
Bayes Suprise	0.66	0.75	0.65	0.67	0.64	0.67	0.63	0.72	0.76	0.68
Ours-Motion	0.73	0.65	0.87	0.85	0.7	0.68	0.67	0.68	0.69	0.73
Ours-Intensity	0.89	0.8	0.78	0.67	0.69	0.76	0.72	0.83	0.7	0.76

Figure 3.9: AUC scores for videos in UCF Sports data set using Bias-Free labeling configuration.

the descriptors. In table 3.2 the results are shown, as one can see using saliency, the results have been improved. This can be justified that by using the saliency map redundant features from the background, which are not discriminative and are common between classes, for example features from the sky, are removed. However, the approach is different from using merely foreground, the context of action is also captured by the saliency mask.

We also repeat the experiment using Dense Trajectory Features [159], with the same configuration as mentioned. Visibly more features are pruned in this experiment by the saliency mask. The results in table 3.3 indicate that even though in some classes the accuracy drops, on average the method outperforms the baseline.



	Diving	Golf Swinging	Kicking	Lifting	Horseback Riding	Running	Skating	Swinging	Walking	Average
SUN	0.79	0.6	0.63	0.85	0.75	0.74	0.75	0.67	0.71	0.72
Bayes Suprise	0.78	0.69	0.61	0.88	0.73	0.65	0.69	0.76	0.74	0.73
Ours-Motion	0.87	0.7	0.76	0.79	0.78	0.79	0.8	0.92	0.84	0.81
Ours Intensity	0.85	0.76	0.78	0.96	0.81	0.82	0.76	0.87	0.82	0.83

Figure 3.10: AUC scores for videos in UCF Sports data set based on Default configuration.

3.6 Comparison

In order to show the effectiveness of the proposed method, which uses super-voxels as the basic elements to find saliency in videos, and group-lasso regularization to provide appropriate feature space for decomposing via low-rank minimization, we have also implemented a saliency detection method using cuboids only and L_1-minimization (lasso, not group lasso). We applied and tested the latter method on some samples, which we used in our experiments, and compared the results with ours. As figure 3.18 shows, decomposition based only on the results of the referenced baseline [165] methods is noisy and vague.

The reason is, if the salient object or region is large, the number of cuboids would be enormous;

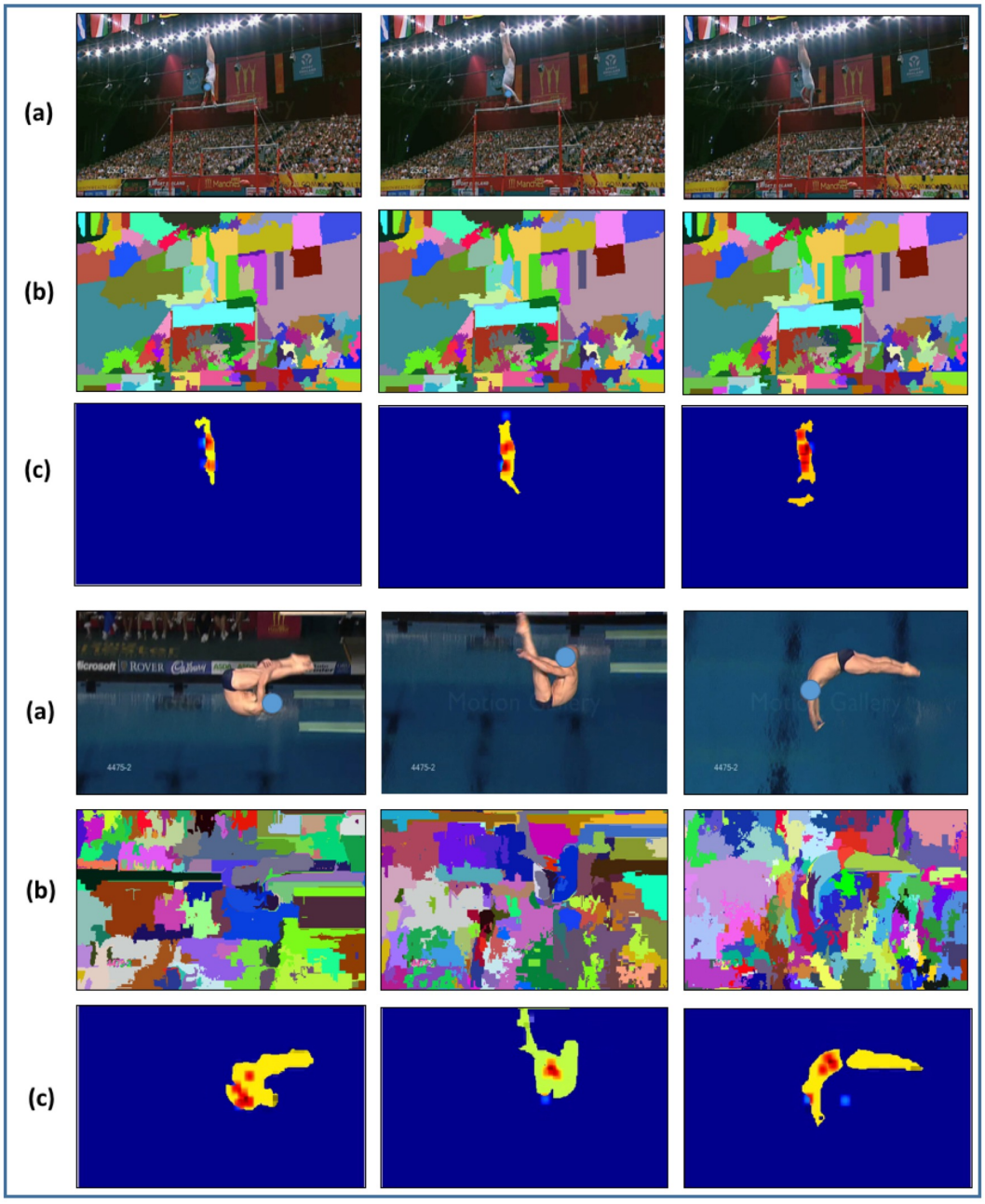


Figure 3.11: Examples of frames from (a) UCF Sports data set videos, (b) super-voxels, (c) our results showing most salient regions plus gaze points shown in red considering calibration errors.

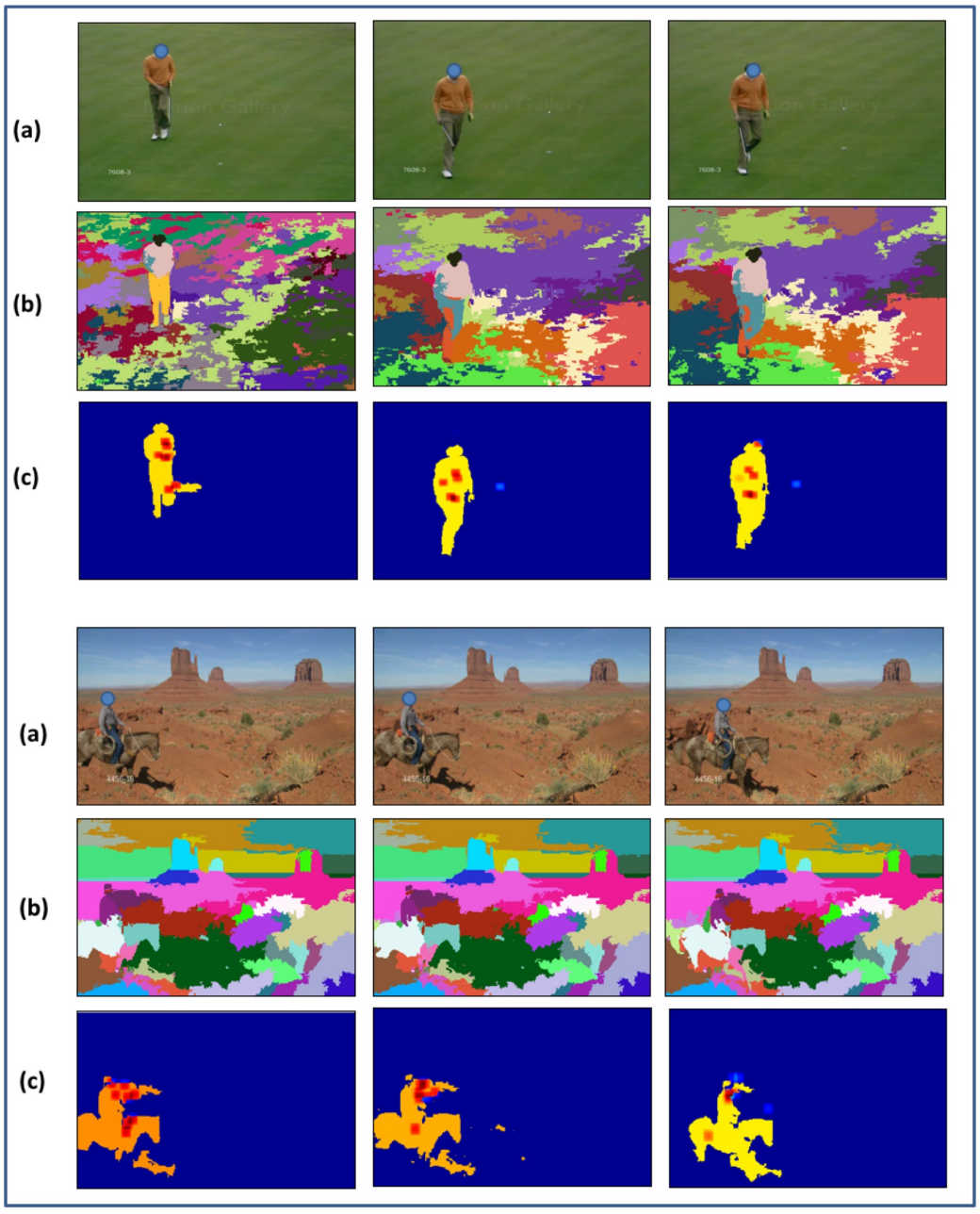


Figure 3.12: More examples of frames from (a) UCF Sports data set videos, (b) super-voxels, (c) our results showing most salient regions plus gaze points shown in red considering calibration errors.

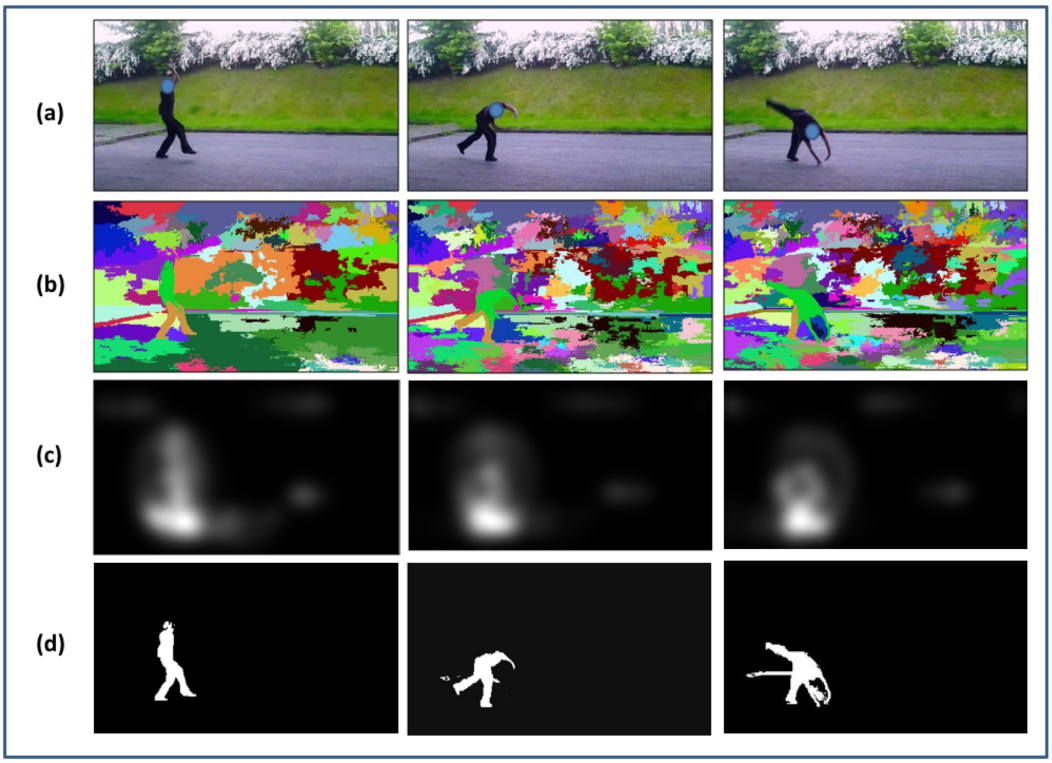


Figure 3.13: Examples of frames from (a) UCF Saliency data set videos, (b) super-voxels, (c) empirical saliency maps and (d) our results showing most salient regions. These salient regions correspond to meaningful objects such as person filliping, a person walking with kids, boat and car.

and they could not be considered sparse. The assumption of the salient parts being sparse would not be valid anymore. Furthermore, this approach does not consider the correlation between variables, therefore it does not enforce that the non-salient part should have a low rank. On the other hand, by using grouping of cuboids and utilizing group lasso regularization, as we have done, highly correlated variables are selected together and sparsity is applied among groups. Our approach does not need object detection methods or training. It is able to fairly accurately detect most dominant objects, which by using only lasso regularization and cuboids is not feasible. One of the key aspects of our approach is that it does not use any gaze locations or labeled data to train the system, and we do not need to adjust our method to specific type of videos or objects.

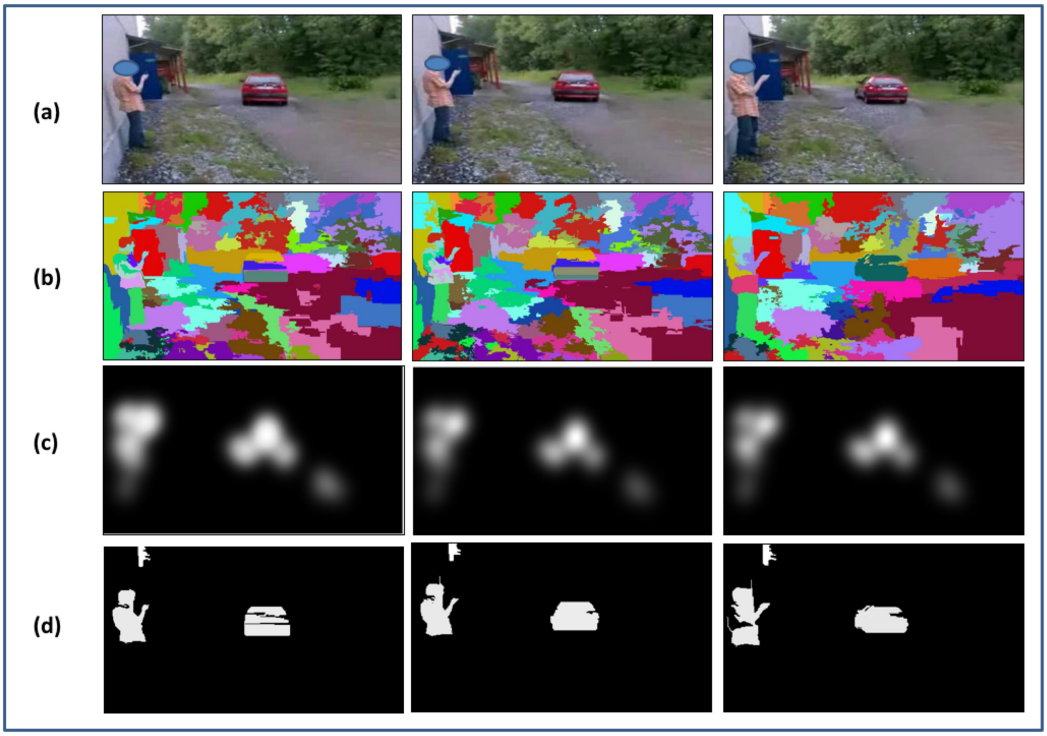


Figure 3.14: Examples of frames from (a) UCF Saliency data set videos, (b) super-voxels, (c) empirical saliency maps and (d) our results showing most salient regions.

For more demonstration on effectiveness of group lasso to impose sparsity and make non-salient parts low-rank, in table 3.4 rank and sparsity percentage of data after decomposition using RPCA are shown. The first column shows using only intensity, and the second one indicates the data after applying group lasso, which shows dramatic reduction in rank as well as number of non-zero values.

We have also experimented with different initial features including intensity, RGB, luminance channel (Y), YUV and temporal gradients. As figure 3.19 shows, intensity and luminance channel have the best results, and the other features lead to slightly lower performance. In this case, the temporal gradient has the lowest AUC score. It can be explained as some videos like the bumblebee

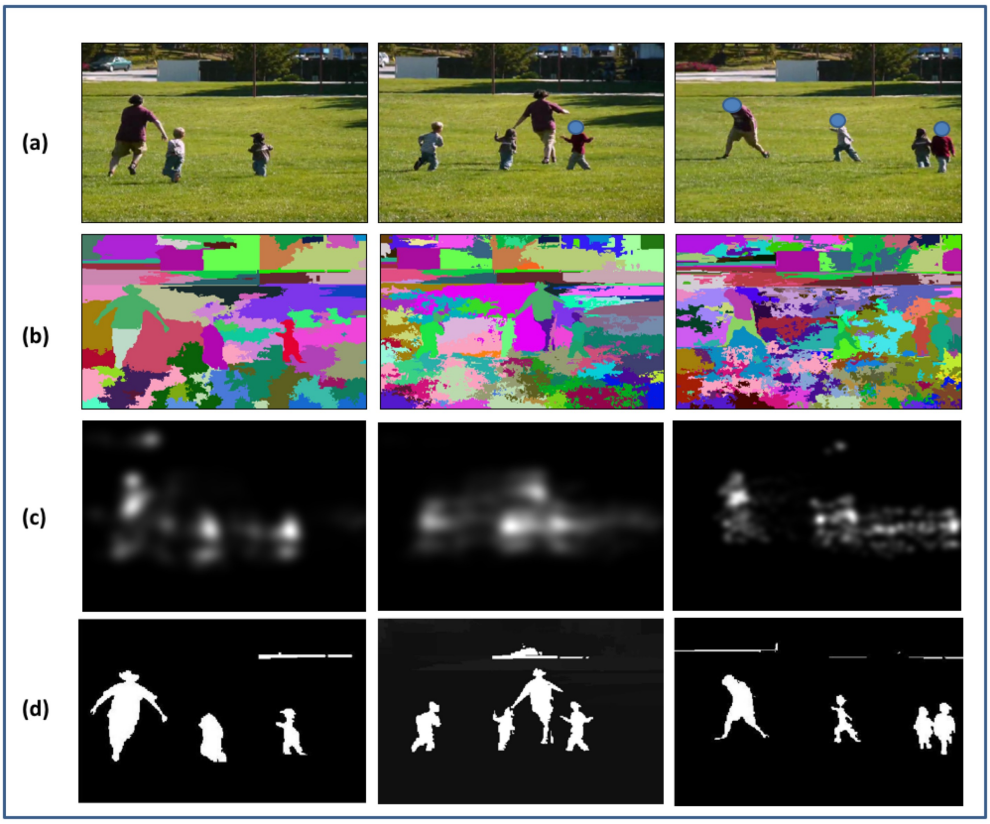


Figure 3.15: Examples of frames from (a) UCF Saliency data set videos, (b) super-voxels, (c) empirical saliency maps and (d) our results showing most salient regions.

has no dominant and meaningful action in them. We also repeated this experiment for the UCF Sports data set since actions and motion are the main focus in this data set, the temporal gradient performs better than the intensity in some videos. However, the difference is not that remarkable and on average the intensity performs slightly better, therefore for the sake of consistency we have reported results for other data sets using intensity.

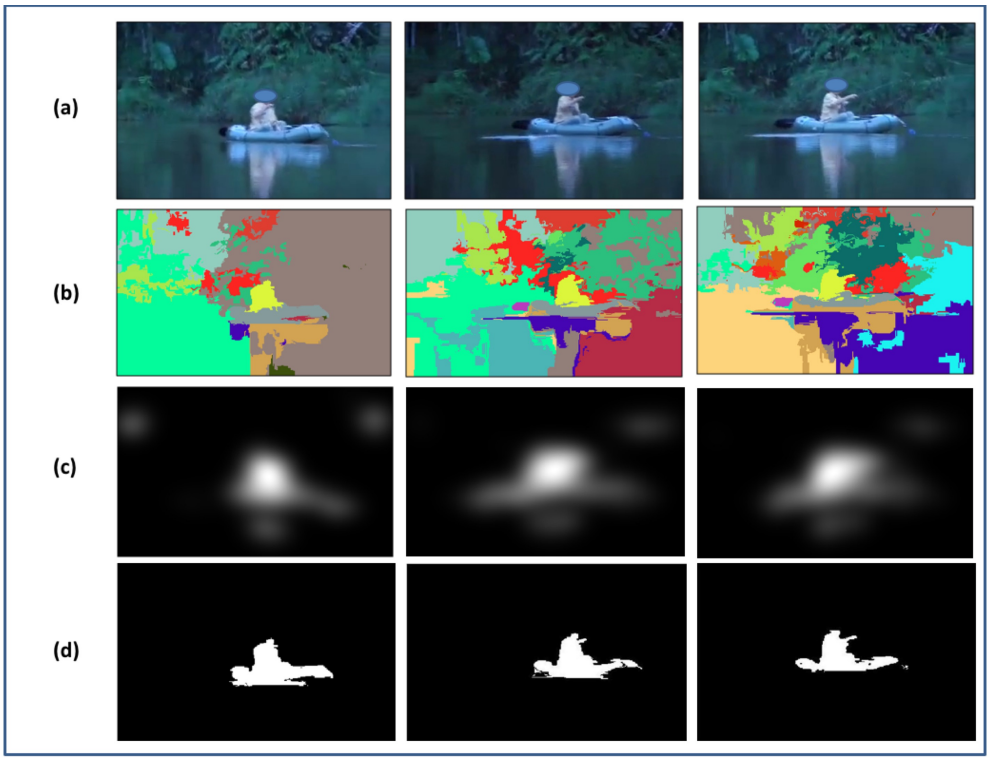


Figure 3.16: More examples of frames from (a) UCF Saliency data set videos, (b) super-voxels, (c) empirical saliency maps and (d) our results showing most salient regions. These salient regions correspond to meaningful objects such as boat and person.

Table 3.2: Accuracy results Using HOG+MBH descriptor for action recognition in UCF Sports data set.

Action	baseline(reproduced)	saliency sampling
Diving	100%	100%
Golf	100%	100%
Kicking	100%	100%
Lifting	50%	100%
Horse Riding	100%	100%
Running	25%	75%
Skating	50%	50%
Swing Bench	83.33%	50%
Swing Side	100%	100%
Walking	71.43%	85.71%
Average (per video)	80.85%	**85.10**%

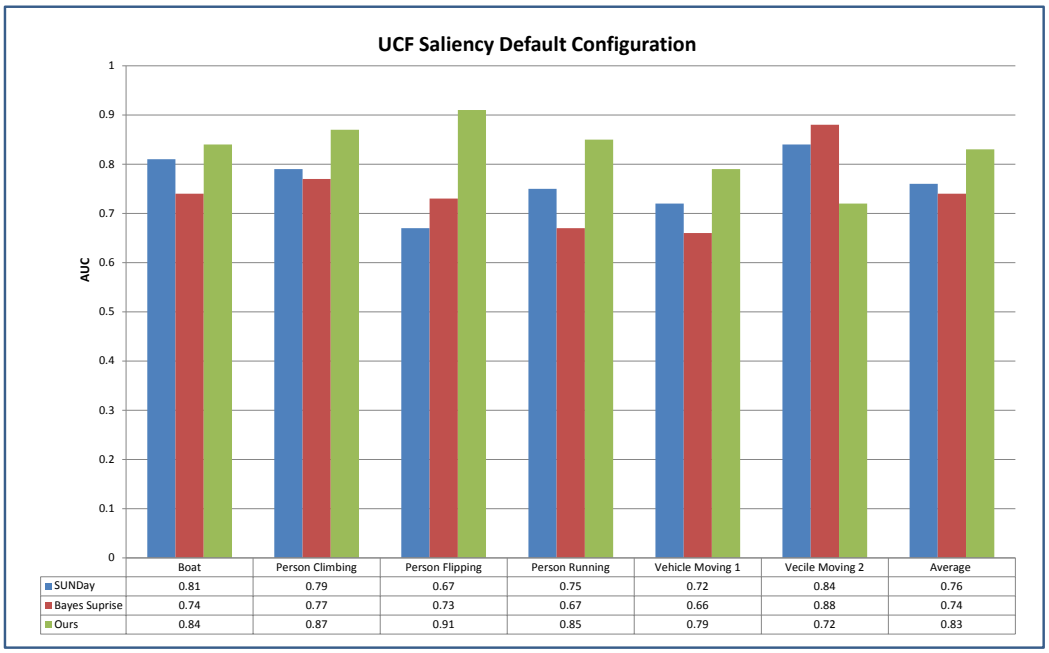


Figure 3.17: AUC score for videos in UCF Saliency data set based on Default-Labeling configuration.

Table 3.3: Accuracy results Using DTF descriptor for action recognition in UCF Sports data set.

Action	baseline(reproduced)	saliency sampling
Diving	100%	100%
Golf	100%	66.67%
Kicking	50%	83.33%
Lifting	100%	50%
Horse Riding	75%	100%
Running	75%	75%
Skating	0%	50%
Swing Bench	100%	100%
Swing Side	75%	75%
Walking	71.43%	71.43%
Average (per video)	76.59%	**78.72**%

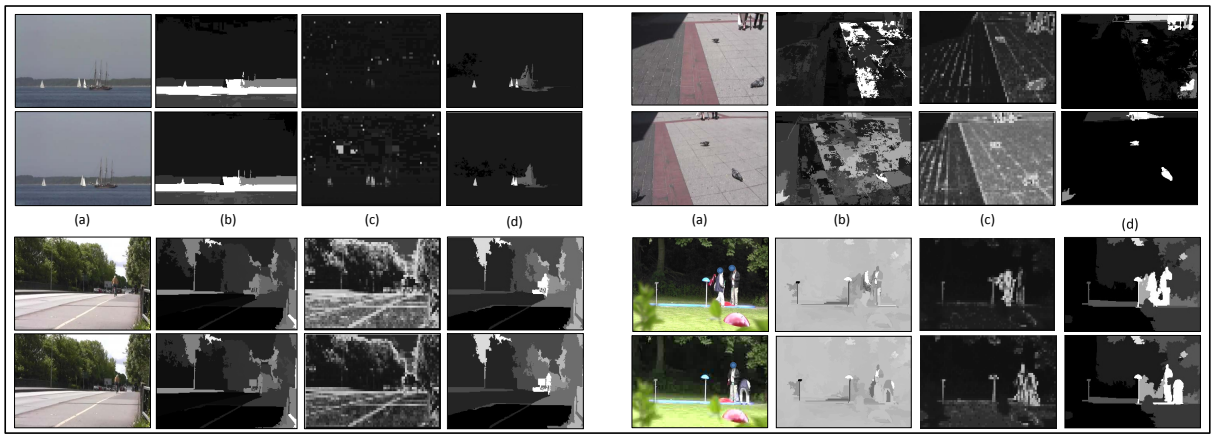


Figure 3.18: Examples of results for street, sea, doves and golf video scenes from INB dataset. (a) video sample frames set, (b) saliency map using low rank decomposition on intensity data (c) saliency maps via L_1-minimization with no grouping and (d) results of our method. The AUC scores obtained by low rank decomposition are respectively 0.68, 0.56, 0.51 and 0.59. For L_1-minimization they are 0.63, 0.52, 0.54 and 0.71, which are noticeably lower in accuracy than our results.

Table 3.4: This table shows some examples of rank reduction and imposing sparsity before and after using group lasso. Non-zero represents the percentage of non-zero elements in the sparse matrix after decomposition by RPCA, and Rank shows the rank of the low-rank matrix after decomposition.

	Rank-Sparsity	
Video	Non-Zero w/o GL	Non-Zero w/ GL
Diving 001	81 %	12 %
Walking 022	85 %	9 %
Skating 004	28 %	6 %
Kicking side 001	82 %	4.5 %
Swing Side 009	72 %	19 %
	Rank w/o GL	Rank w/ GL
Diving 001	35	9
Walking 022	37	4
Skating 004	16	2
Kicking side 001	39	12
Swing Side 009	33	6

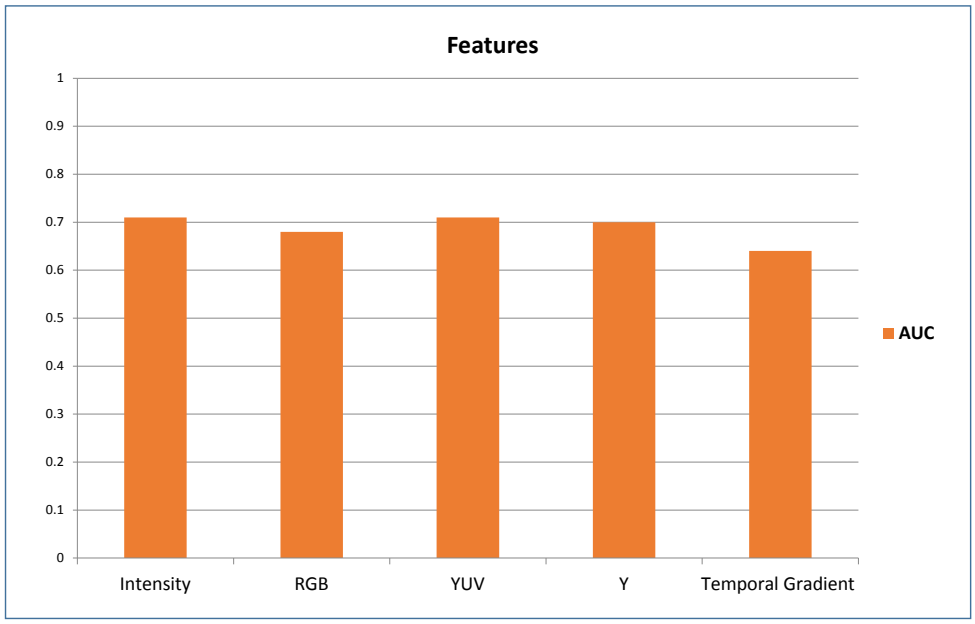


Figure 3.19: AUC scores using different features: intensity, RGB, luminance channel (Y), YUV and temporal gradients features for Bias-Free labeling configuration from INB data set.

3.7 Summary

In this chapter, we presented a completely unsupervised bottom-up method to detect the regions of videos to which people's eyes are drawn to. We used spatiotemporal information to represent a video as a matrix and by using super-voxels, we grouped the columns into the matrix to cluster related data together. We proposed to utilize group lasso regularization to transform data into a sparse representation, in which redundant parts remain low rank and salient part would be sparse. Moreover, the correlation between data is retained and non-salient parts tend to be of low rank. We have shown that without using data labeling and learning techniques requiring eye movement data, we are able to find salient regions in videos properly.

In the next three chapters, we address problem of semantic segmentation, where we label each pixel in an image with a semantic label e.g building, sea, grass etc. This kind of analysis provide more understanding of an image compared to saliency detection.

CHAPTER 4: SCENE LABELING USING SPARSE PRECISION MATRIX

Semantic segmentation task is to segment the image into meaningful regions and categorize them into classes of objects or scenes which comprised the image. Commonly used methods typically find the local features, such as color histogram, texture and etc. for each image segment and label them using classifiers. Afterward, labeling is smoothed locally in order to make sure that neighboring regions receive similar labels. However, these methods ignore expressive and non-local dependencies among regions as they require expensive training and inference. One of the widely used approaches to address this problem is to exploit MAP (maximum a posterior) inference in a multi-class conditional random field (CRF). This is the extension of the binary CRF, which has been widely used to find foreground-background in images. CRF provides a probabilistic framework to model interactions between output variables and observed features. Nonetheless, its structure is fixed either limited to only instant neighbors or connected to all other nodes. Thus, the long-range dependencies are missed or hard to achieve.

In this chapter, we propose to use a sparse estimation of precision matrix (also called concentration matrix), which is the inverse of covariance matrix of data obtained by graphical lasso to find interaction between labels and regions. To do this, we formulate the problem as an energy minimization over a graph, whose structure is captured by applying sparse constraint on the elements of the precision matrix. This graph encodes (or represents) only significant interactions and avoids a fully connected graph, which is typically used to reflect the long distance associations. We use local and global information to achieve better labeling. An example of how our method performs versus spatial smoothing is shown in Figure 4.1. We assess our approach on three datasets and obtain promising results.

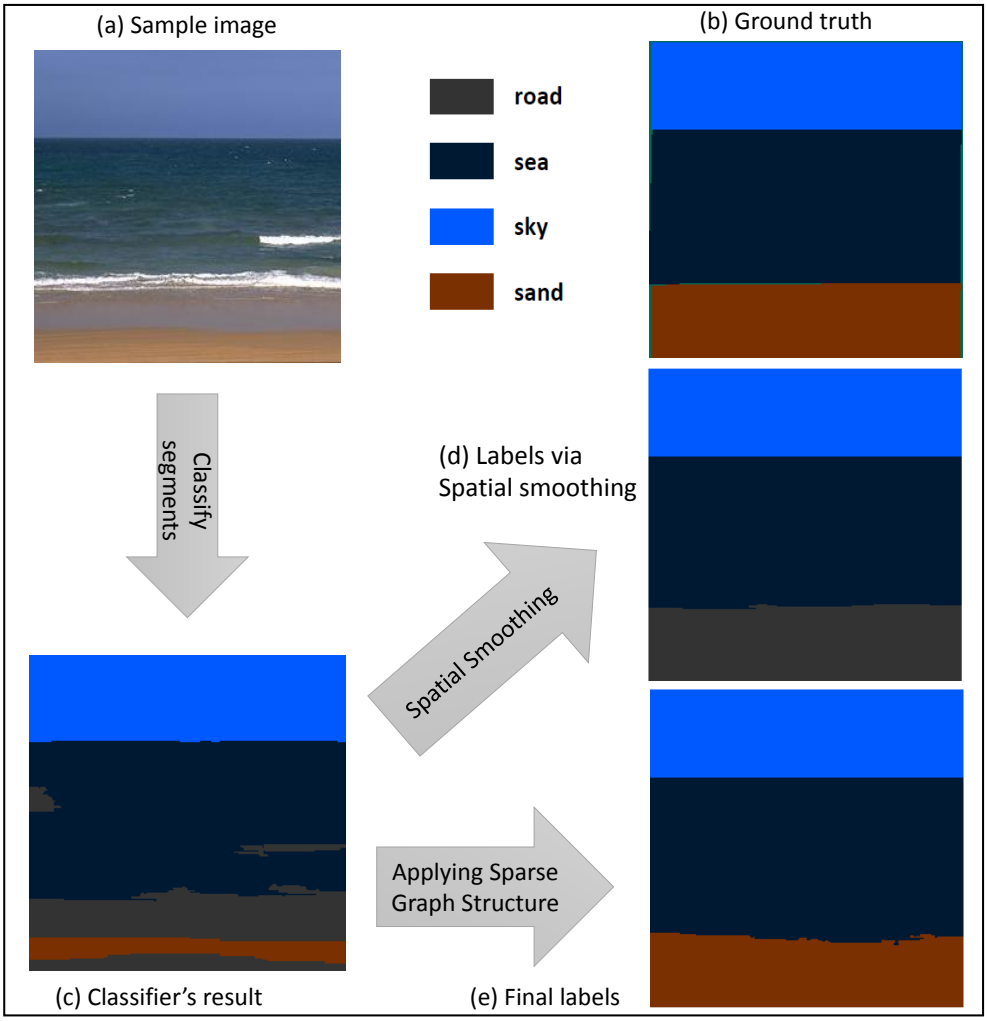


Figure 4.1: Given an image, we aim to improve the semantic labels of regions, originally miss-labeled by classifiers. (a) shows a query image, (b) shows the human annotated image (ground truth), (c) shows labels obtained by classifiers, (d) shows labels via spatial smoothing and (e) shows our results.

4.1 Proposed Approach

An overview of our proposed model is shown in figure 4.2.Our approach consists of two main steps. The first step consists of off-the-shelf parts including feature extraction and classifier training

based on local features of the sample training images. Also, in this phase using the training data, we capture the structure of semantic label interactions graph to be later employed in the pair-wise cost computation. In the second step, which is the inference, for a given query image, using scores computed by the classifiers for each possible label, and the pair-wise costs obtained by label correlations and appearance features of the image, the MAP inference in CRF framework is applied and each super-pixel is assigned a label. Next, for each super-pixel, local features, including SIFT, color histogram, mean and standard deviation of color, area and texture, are extracted. Given these local features, classifiers (random forest) are trained to label super-pixels using their local features. In training, first we segment images using efficient graph based segmentation [34]. Also, in training phase we build the sparse precision matrix based on the sample data to highlight the important relations (positive or negative correlations) between labels.

In testing, for a query image we find the unary terms, for its segments, using scores from local classifiers refined with the probabilities obtained from a retrieval set based on global features. Then, we use a fast implementation of graphical lasso to find the structure of the dependency graph between super-pixels and assign weights to edges based on correlation values. Finally, we use α expansion to optimize the energy function and assign a label to each super-pixel.

4.1.1 Image Segmentation

If we represent a graph on image primitives as $G(V, E)$, V is a set of nodes and E is the set of edges. Even though in segmentation the regions need to be homogeneous, the variation of intensity should not be the only measure for dividing a region. In addition, some non-local criteria must be employed in order to address cases such as intensity difference across the boundary between areas with high variations. The intuition behind this algorithm is that "the intensity differences across the boundary of two regions are perceptually important if they are large relative to the intensity

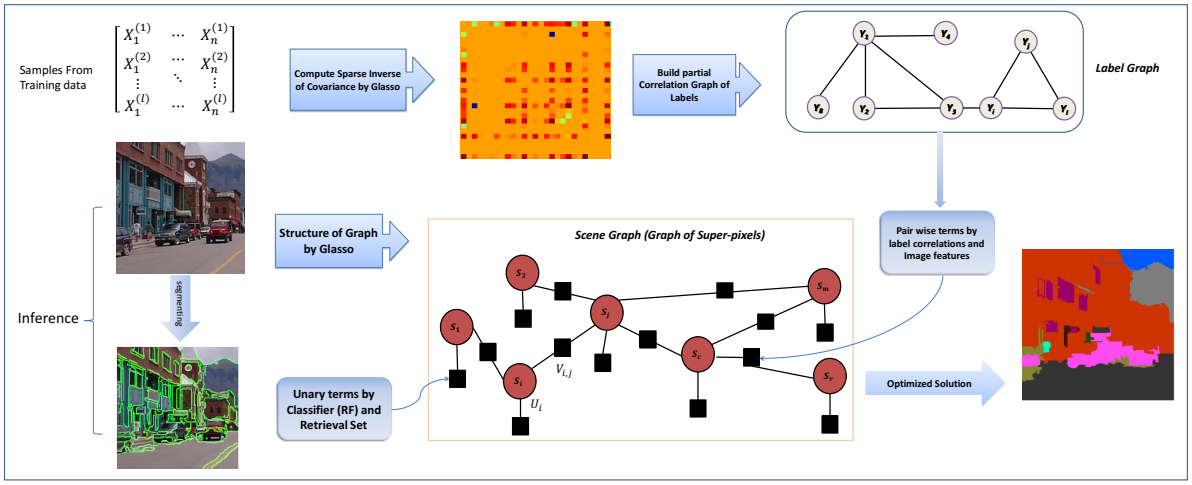


Figure 4.2: The overview of our approach: We begin by extracting the feature matrix, and segmenting the image into super-pixels. Then classifiers (random forest) are trained. We detect the relations between labels using the sparse estimated partial correlation matrix of training data. In the inference part, for a given image the label scores are obtained via the classifiers, then the energy function of a sparse graphical model on super-pixels is optimized to label each super-pixel.

differences inside at least one of the regions" [34].

In this graph-based approach [34], S is a segmentation that partitions V into segments (regions) such that each component (or region) $C \in S$ is a connected component in a graph G. A pair-wise region comparison predicate D is defined to assess whether a boundary is needed between two regions or not. This predicate compares the inter-component dissimilarities to within(intra) component differences.

To do so, using the minimum spanning tree of the component $MST(C, E)$, the internal difference of a component $C \subset V$ is defined $Int(C) = \max_{e \in MST(C,E)} w(e)$ which is the largest weight in the MST, meaning that as long as the edges with weight at least $Int(C)$ are taken into account C can

remain connected.

The difference between two components $C_1, C_2 \subset V$, which finds the minimum weight edge connecting the two components, is defined as $Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j)\ inE} w((v_i, v_j))$, and if there is no edge the $Dif(C_1, C_2) = \infty$. If $Dif(C_1, C_2)$, is large relative to at least one of the internal differences $Int(C_1)$ and $Int(C2)$, there should be a boundary between the regions. To this aim, a threshold function controls the ratio of the difference between components and minimum internal difference. Some sample images and their segmentation are shown in Figure 4.3.

4.1.2 Background on Graphical Lasso

In this section, we review some statical concepts which are helpful to understand the rest of the section.

- **Covariance** is defined using expectation and measures how random variables co-vary or change together, $\Sigma = E([X - E(X)][Y - E(Y)])$.
- **Correlation** is covariance normalized by standard deviation. Correlation values are between -1 and 1 making it a more proper measure for comparing random variables and finding out how they are related. $C(X, Y) = \frac{\Sigma(X,Y)}{sd(X)sd(Y)}$, sd is standard deviation.
- **Precision** matrix or concentration matrix is the inverse of covariance matrix, $\Omega = \Sigma^{-1}$ and indicates how tightly variables are clustered around the mean and they are independent.
- **Partial correlation** measures the degree of association between two random variables given other variables , $\rho(X, Y|Z)$, here ρ is reflecting how X and Y are correlated given Z. Precision matrix and partial correlation matrix $R\{\rho_j k\}$ have a useful relation as $R(j, k) =$

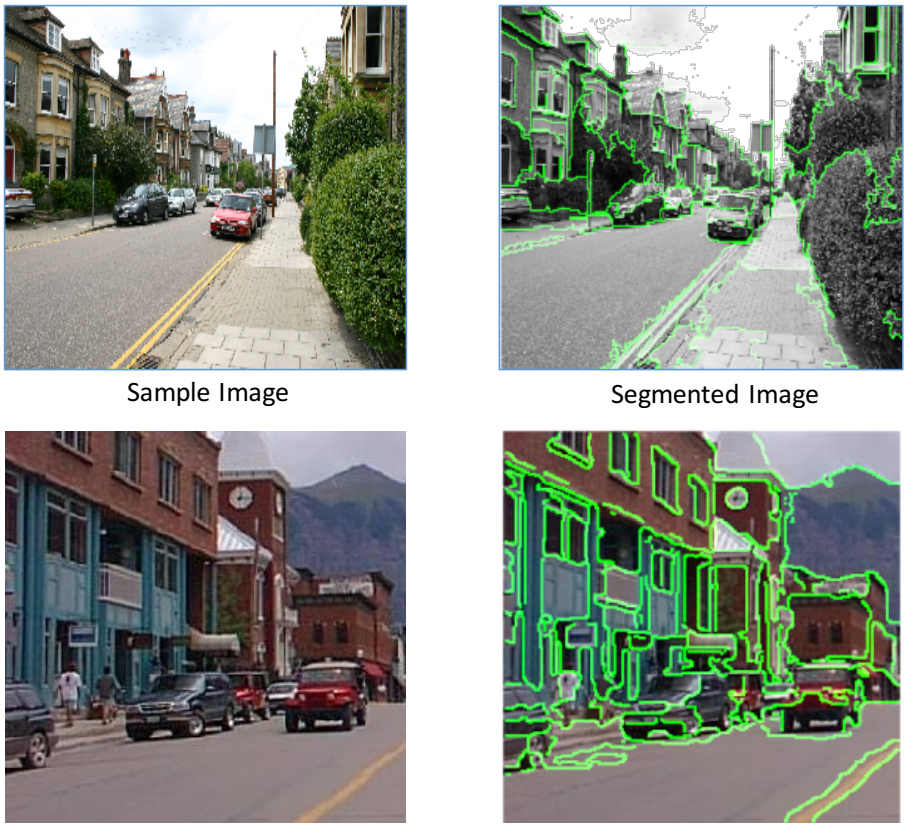


Figure 4.3: Sample images and corresponding segmented images using *Efficient Graph-Based Image Segmentation* method.

$\frac{-\Omega_{jk}}{\sqrt{\Omega_{jj}\Omega_{kk}}}$ indicating that if Ω_{jk} is zero consequently $R(j,k)$ is zero, and j and k are not correlated given other variables.

Graphical model (Dependency graph) and partial correlation graph are the same under the Gaussian assumption. Even if data is not normal, since finding a conditional independent graph is complex, partial correlation is a decent approximation for finding dependency between variables.

4.2 Graphical Lasso and Sparse Precision Matrix

In order to find the structure of the graph of our model, we employ the precision matrix (the inverse of covariance matrix) to capture the dependency between variables. The partial correlation between two variables, X and Y, given other variable Z, measures the association between X and Y after regressing X and Y on Z. If the partial correlation between two variables given all other variables is zero, there will be no edge between them in the corresponding partial correlation graph.

The matrix of partial correlations between variables can be defined using the inverse of covariance matrix Ω. Therefore, zeros in the inverse covariance indicate that there is no edge in the graph. Even though empirical covariance of the data is a decent approximation of the true covariance, this is not valid for the precision matrix. Furthermore, when the dimension of the data increases, the covariance matrix may not be invertible. We assume the data follows Gaussian distribution.

Let $X = (X^{(1)}, ..., X^{(p)})$ be a *p*-dimensional random vector. Assume, we have a set of n random samples $X_1, ..., X_n$, we are interested in identifying conditional independence between the pair of variables (features) $X^{(i)}$ and $X^{(j)}$, given other variables. In doing so, X can be represented by a graph $G = (V, E)$, where vertices correspond to p variables and the edges represent the correlations between variables. In the Gaussian (Normal) distribution, the correlation and dependency graph are equivalent.

Even though the data may not have a normal distribution, since conditional independence graphs are hard to estimate, employing partial correlation is a reasonable alternative to find the structure of the interactions between the variables. Let the matrix $C = \{\rho_{i,j}\} \in \mathbb{R}^{p \times p}$ be a partial correlation matrix, where $\rho_{i,j}$ captures the partial correlations between variables $X^{(i)}$ and $X^{(j)}$, and

$$\rho_{i,j} = -\Omega_{i,j} / \sqrt{\Omega_{i,i}, \Omega_{j,j}}, \tag{4.1}$$

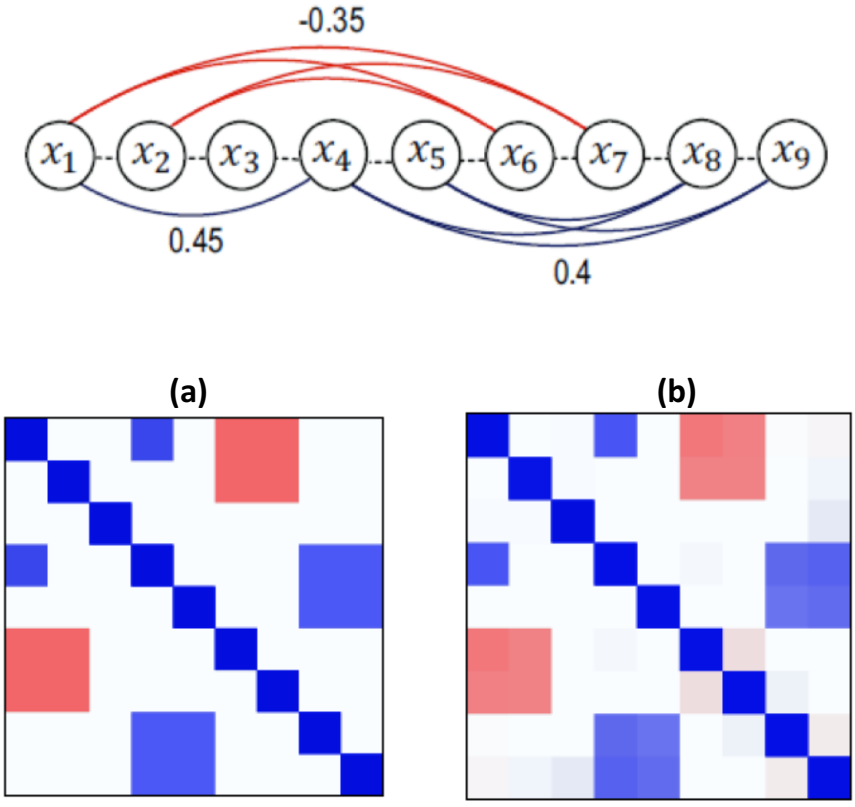


Figure 4.4: An example of using graphical lasso to discover dependency between variables. The first row is a graph dependency between 9 variables: red lines shows negative correlation, blue indicates positive correlation and dotted lines mean local dependency (adjacency). (a) is precision matrix of the ground truth and (b) is learned structure from data [59], even though (b) is not exactly as (a) and has some noise, using graphical lasso we can recover true dependencies.

where $\Omega = \Sigma^{-1}$ is the inverse of the covariance matrix of the data with covariance Σ. Using sample covariance matrix to estimate the matrix C is not proper for high dimensional data, due to the limited number of samples, the covariance matrix may not be invertible. Also, more importantly, the inverse of empirical covariance matrix may not be sparse and consequently not resulting in a sparse graph. In order to find the structure of the graph and obtain a certain number of influential edges, it is desirable to have zeros in the precision matrix, since zeros determine the independent (uncorrelated) variables. Therefore, imposing sparsity constraint on the elements of precision

matrix enforces that insignificant and noisy relations are discarded and meaningful dependencies are persevered. To achieve sparsity, [169] proposed to use a lasso (Least Absolute Shrinkage and Selection Operator) model [146] to estimate each variables using others as predictor and by applying L_1 regularization on coefficients to enforce sparsity. Therefore, the edges in the graph are removed for the variables for which corresponding coefficients are zero.

In [35], an algorithm, named graphical lasso (glasso), was proposed to maximize the Gaussian log-likelihood of the data with L_1 penalty on precision matrix elements to impose sparsity. This approach uses block coordinate gradient to solve the optimization problem, which is fast and suitable for our application. Let S be the empirical covariance matrix of the data, then Ω can be obtained by,

$$\arg\max_{\Omega} \ \text{logdet}\,\Omega - \text{tr}(S\Omega) - \lambda \left\|\Omega\right\|_1 , \tag{4.2}$$

where tr is the trace of the matrix and $\|\|_1$ is the L_1norm (sum of the absolute values) of the matrix. In brief, one can model the dependency between variables using their partial correlation graph. The partial correlation graph has an edge between j and k when $\rho_{j,k} \neq 0$. Furthermore, as mentioned above, partial correlation has a direct relation with inverse of covariance of the data (equation 4.1). Therefore, by estimating a sparse precision matrix (inverse of covariance), one could obtain the structure of the dependency graph between variables, where zeros in the precision matrix mean there is no edge between corresponding variables. In following sections we explain each part of our approach in detail.

4.3 Local Classifiers

In this section, we explain the first step of the model. In training, we start with segmenting each sample image into super-pixels using efficient graph-based segmentation method [34], followed

Table 4.1: Super-pixel features from [147]

Shape	Mask of superpixel shape over its bounding box (8×8)	64
	Bounding box width/height relative to image width/height	2
	Superpixel area relative to the area of the image	1
Location	Mask of superpixel shape over the image	64
	Top height of bounding box relative to image height	1
Texture/SIFT	Texton histogram, dilated texton histogram	100×2
	SIFT histogram, dilated SIFT histogram	100×2
	Left/right/top/bottom boundary SIFT histogram	100×4
Color	RGB color mean and std. dev.	3×2
	Color histogram (RGB, 11 bins per channel), dilated hist.	33×2
Appearance	Color thumbnail (8×8)	192
	Masked color thumbnail	192
	Grayscale GIST over superpixel bounding box	320

by computing a feature vector(including, SIFT, color mean) for each super-pixel in the image , thus each super pixel would be represented by a feature vector for the classifier. Since the ground truth for each image is pixel based, each super-pixel is assigned a label which corresponds to the majority of its pixels. We use the same features as used in [147], the details of the features are given in table 4.1.

We use random forest classifiers [83] to classify each super-pixel in a test image. Due to the fact that super-pixels may break the structure of the data, since training data inevitably is noisy, the bagging using subset of training examples and subsets of features is used to reduce the effects of the noisy data. In order to rescale the classifier scores and give chance to other classes to compete during optimization phase, we use a sigmoid function. By doing so, if the classifier mislabels a super-pixel, there is more chance that the label would be changed during the inference phase. We adapt the parameters of the sigmoid function using the validation data.

Unlike some of the other methods, which train object detectors in addition to the region classifiers, we only use region features and small scale classifiers to obtain the initial label scores for each

super-pixel. In our experiments, random forest achieved better results in terms of average accuracy among all the classes, even though we randomly discard some of the samples during the training, due to large number of super-pixels.

4.4 Global Retrieval

Since the local classifiers treat each super-pixel individually, the context information may be missed, therefore we propose to refine the scores obtained from the classifiers by leveraging the global feature extracted from the data. By doing this, we enforce that global information of the scene and spatial features play a role in labeling the data. We use GIST features to retrieve a subset of the nearest neighbors of the query image from the training data.

We use the method proposed in [107] to speedup the retrieval process and make it scalable for large databases. Next, we compute the probability of assigning each label, l, at a specific location by counting the number of super-pixels with the label l in the retrieval set, and normalize it with respect to the total number of labels. Thus, we have a probability as $p_g(label = l_i|location = (x, y))$. Finally, for each label we modify the obtained scores from the classifiers with these probabilities (corresponding to the super-pixels) using the following late fusion formulation:

$$\mathrm{w}_g(i,j) = \mathrm{w}(i,j)^{\gamma} \times p_g(i,j)^{1-\gamma}, \tag{4.3}$$

where γ is the combination coefficient and $\mathrm{w}_g(i,j)$ is the new probability of i_{th} label for the super-pixel j.

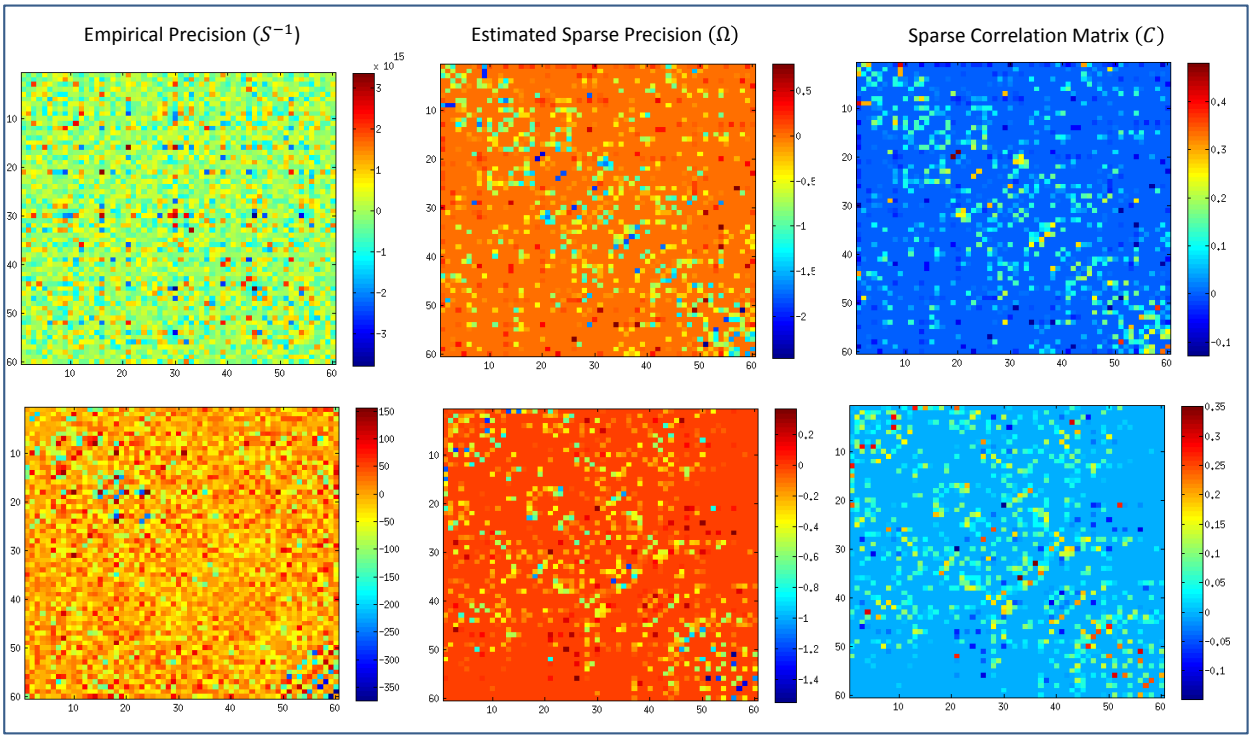


Figure 4.5: The first row is obtained by using the output (scores) of each classifier and treating it as a random sample. The second row is obtained using the features of the super-pixels to find the correlation between them. The first column corresponds to empirical inverse of covariance matrix of the data, as shown the entries are very noisy and finding true interactions among the super-pixels is difficult. However, the estimated sparse precision matrix provides fewer and more meaningful interactions.

4.5 Graphs Structures

In order to capture the structure of the label graph, we start by building a matrix comprising of the sample data. Each image in the training set is represented by a vector of size equal to the number of classes; we want to discover the influences of labels, therefore our random variables (features) are labels in the dataset. For example, in the SIFTflow data set we have 33 labels, therefore each vector has dimension of 33. The value of a particular variable (specific label) in this representation

is the probability of seeing that label in the image. These probabilities are obtained by counting the pixels belonging to the class and normalizing them by the image size. Then, using equation 4.2 the precision (concentration) matrix is estimated.

The degree of the sparsity is handled by parameter λ. Partial correlations of labels are used to find the interaction between labels, which is used for pairwise-cost (interaction potential) in the CRF formulation instead of using Potts model which applies a penalty for nearby similar pixels that are assigned different labels, and it is ignorant to compatibility between labels. Therefore, in the proposed approach, if two connected nodes do not have the same label, assigning different pair of labels contribute differently in finding the conditional probability of the assignment.

In addition, in order to capture the structure of the graph (scene graph) for the image elements (here super-pixels) in the inference step, we use the graphical lasso to obtain the relationships between the super-pixels. By doing this, if two super-pixels are related but assigned irrelevant labels, the cost of the assignment is increased.

To do so, each super-pixel is treated as a random variable, and by using the classifiers which are trained for class labels, we generate samples for these variables. Thus, the length of each vector is equal to the number of super-pixels and we will have L vectors, where L is number of the classes. Then, we again use graphical lasso and estimate a sparse precision matrix (inverse of covariance), and subsequently obtain the partial correlation graph, where the zero indicates no edge between super-pixels. Note that since the number of super-pixels can be large, the covariance matrix may be singular and not invertible, due to the fact that the number of available samples (e.g. scores from classifiers) is limited. Therefore, in such cases using the sparse estimation can be beneficial.

Not only do we find the structure and the connections between regions, we also use these values to incorporate relevancy of super-pixels in pair-wise potentials. As it is shown in figure 4.5, the inverse of the sample covariance matrix is very noisy due to the fact that the covariance matrix

can be singular (or close to singular). By using the graphical lasso we can capture the structure of the graph, (as shown in the figure) and also preserve the correlation between spatial neighbors of super-pixels. The alternative for finding the relations between super-pixels is to use their features and try to find dependency between super-pixels, by sparse representation of each super-pixels using other super-pixels as predictors. The example is shown in the bottom row of the figure 4.5. We use features of super-pixels after reducing the dimension by PCA. In our experiments, we use the scores from classifiers as sample data since they are more efficient.

4.6 Energy Function Optimization

As we obtain graph structure for the query image, we build a CRF over the super-pixels given the features of the image, and formulate an energy function E as follows:

$$E(y, x) = \sum_{s_i} U(y_i, x) + \tau \sum_{i,j \in rel\_set(i,j)} V(y_i, y_j, x), \tag{4.4}$$

where the goal is to assign a label $y_i \in \mathfrak{L} = 1, 2, ..., l$ to each super-pixel i, while leveraging correlations between labels to refine the individual labeling. Also, we aim to incorporate local smoothness between relevant super-pixels as well. $rel\_set(i, j)$ represents the set of the edges, which correspond to non-zero entires in the precision matrix. And, τ is a weight to control the balance of smoothness. The unary term, U, here is defined as the cost of assigning a label c to a super-pixel s_i, which we obtain by using scores provided by the classifiers $\mathrm{w}_g(c, i)$ for a particular super-pixel:

$$U(y_i = c|x_{s_i}) = 1 - \frac{1}{1 + e^{-\mathrm{w}_g(c,i)}}. \tag{4.5}$$

The pairwise term considers both appearance similarity between super-pixel i and j as well as correlation between labels, as follows:

$$V(y_i = l, y_j = k | x_{s_i}, x_{s_j}) = \delta(l,k) \times F(s_i, s_j), \tag{4.6}$$

$$\delta(l,k) = -\log(\sigma(\rho_{l,k})), \tag{4.7}$$

where $\rho_{l,k}$ is the correlation between labels which is found in the training step, σ is a sigmoid function, and F is the measure of similarity between super-pixels based on color and position features and relevancy of two super pixels obtained from scene graph of super pixels. This term adds cost to the energy in cases where related or similar super-pixels are given irrelevant labels. However, it also applies different costs to different combinations of labels.

It should be noted that here that the edges are not limited to spatial neighbors of the super-pixels only, we also include significant (relevant) long interactions. However, despite fully connected configurations, we do not consider all interactions, thus only significant relations are taken into account. In this structure irrelevant and noisy interactions are avoided. Moreover, we incorporate the partial correlations between super-pixels in the function F given below. This provides the notion of dependency between super-pixels apart from only the appearance similarity.

$$F(s_i, s_j) = (w_1 \, e^{-\|I_i - Ij\|} + w_2 \, e^{-\|p_i - p_j\|}) R(s_i, s_j), \tag{4.8}$$

where I_i is the feature for super-pixel i, namely color mean , p_i is the center position of the super-pixel i, and R measures the relevancy of two super-pixels. This can be computed as $exp(\sigma(\rho_{s_i,s_j}))$, where σ is a sigmoid function.

4.7 Experiments and Results

We evaluate our method on three benchmark datasets. The first dataset is Stanford-background [48], which has 8 classes and 715 images, and following [134] data is randomly split into 80% for training and the rest for testing with 5-fold cross validation. As shown in Table 4.2 we compare our results with state-of-the-art methods, and we achieve better results.

Table 4.2: Accuracy on StandfordBG dataset

Method	Avg Accuracy
Farabet natural [30]	81.4
Gould [51]	77.1
Shauai [134]	80.1
Local Classifier	72.8
Local Classifier + Global	78.9
Local + Global + Spatial smoothing	82.2
Ours Final (sparse structure)	**84.6**

The second dataset that we assess our approach with is SIFTflow dataset [85], which consists of 2,488 training images and 200 testing images from 33 classes collected from LabelMe [123] The quantitative results of our approach are reported in table 5.3 and qualitative results are shown in Figure 4.7. As it is shown, our method is able to achieve promising results without using computationally expensive features or object detectors as required by other methods (e.g. [45]). Note that the main aim of our method is to improve the local labeling via capturing the proper interactions among labels and super-pixels in addition to leverage from context information. Thus, improving the initial labeling using classifiers will lead to better final results.

We also applied our method on third dataset, MSRCV2 [133], which has 591 images of 23 classes. We use the provided split, 276 images in training and 255 images. Here again our method improves

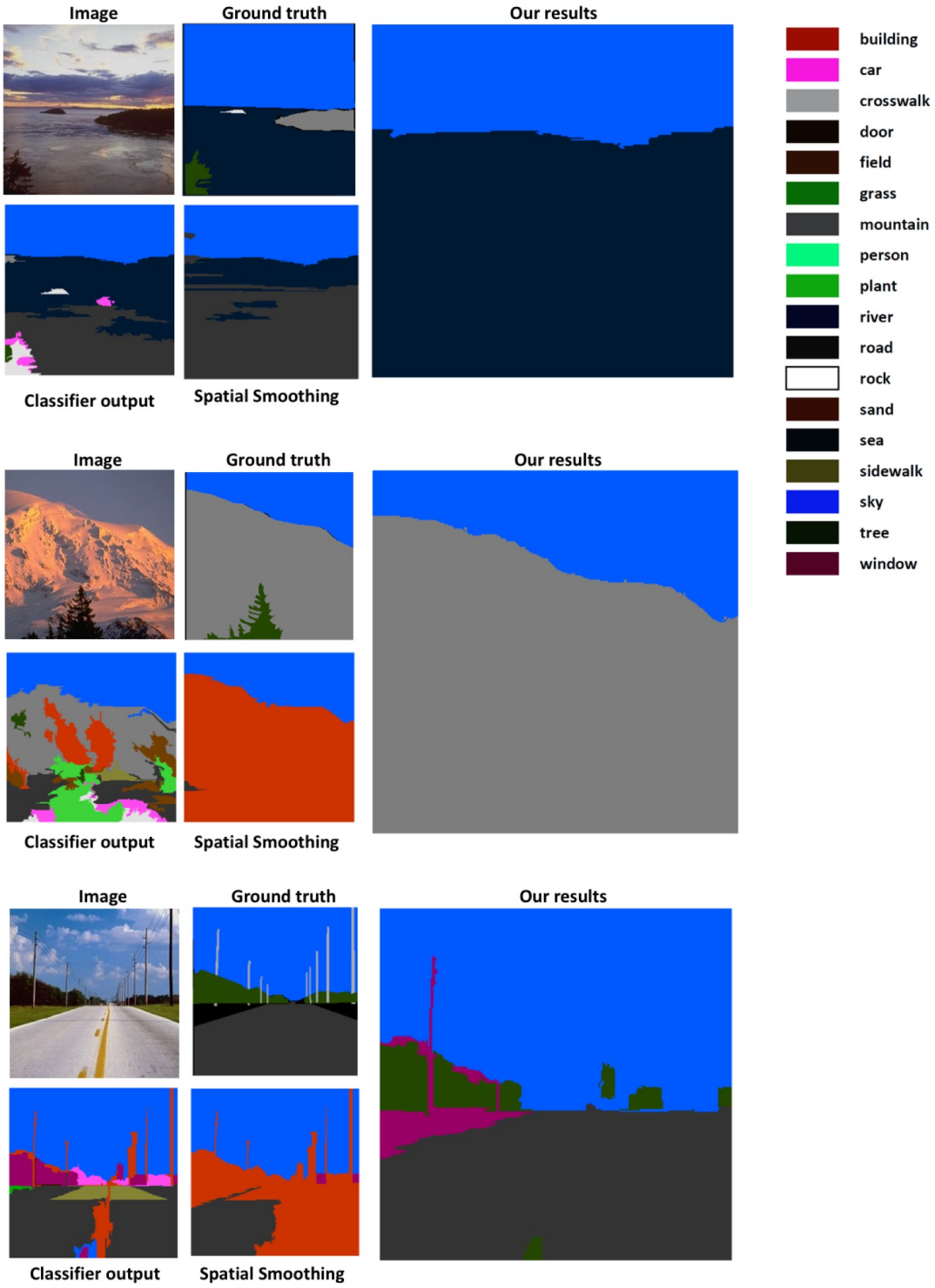


Figure 4.6: Some samples from SIFTflow data set: We show the image, the labels based on classifier scores, results after smoothing using spatial neighborhood and Potts model, and results using our method employing super-pixel correlation graphs (our results are shown in an enlarged image in order to highlight the differences) .

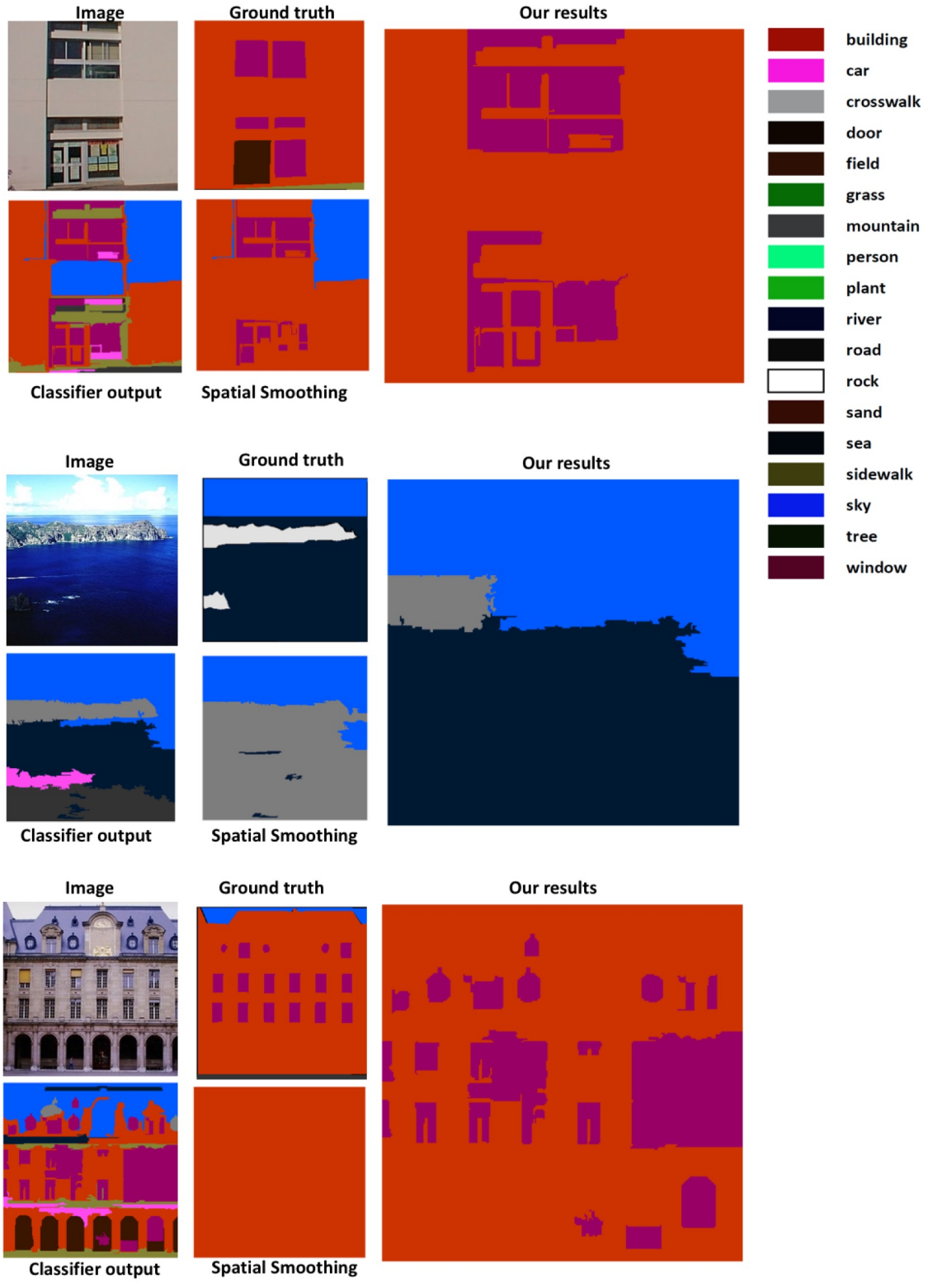


Figure 4.7: More results from SIFTflow data set.

Table 4.3: Accuracy on SIFTflow dataset

Method	Avg Accuracy
Farabet [30]	78.5
Tighe [148]	78.6
Collage Parsing [153]	77.1
Shauai [134]	80.1
Gerorge without Fisher Vectors [45]	77.5
Gerorge Full [45]	81.7
Local Classifiers	71.2
Local Classifiers + Global	75.3
Local +Global + Spatial smoothing	77.7
Ours Final (sparse structure)	**80.6**

the classifiers results and achieves comparable results to the other methods which use different features. For instance, [74] extract features for each pixel, and builds a fully connected graph on pixel levels, where the unary classifier (pixel classifier) gives 84% accuracy, they improve the results by 2%, while our improvement is about 8%, which is significant. If the classifiers are improved, our results can be improved even more.

Table 4.4: Accuracy on MSRC2 dataset

Method	Avg Accuracy
Harmony Potentials [1]	83
Fully Connected CRF [74]	86
Segment CRF with Co-Occurrence [77]	80
Local Classifier	76.6
Local Classifier + Global	77.1
Local +Global + Spatial smoothing	81.7
Ours Final (sparse structure)	**84.1**

In table 4.5 the average accuracy results per class are reported. As it is shown, our method does not compromise the per class accuracy for smoothing.

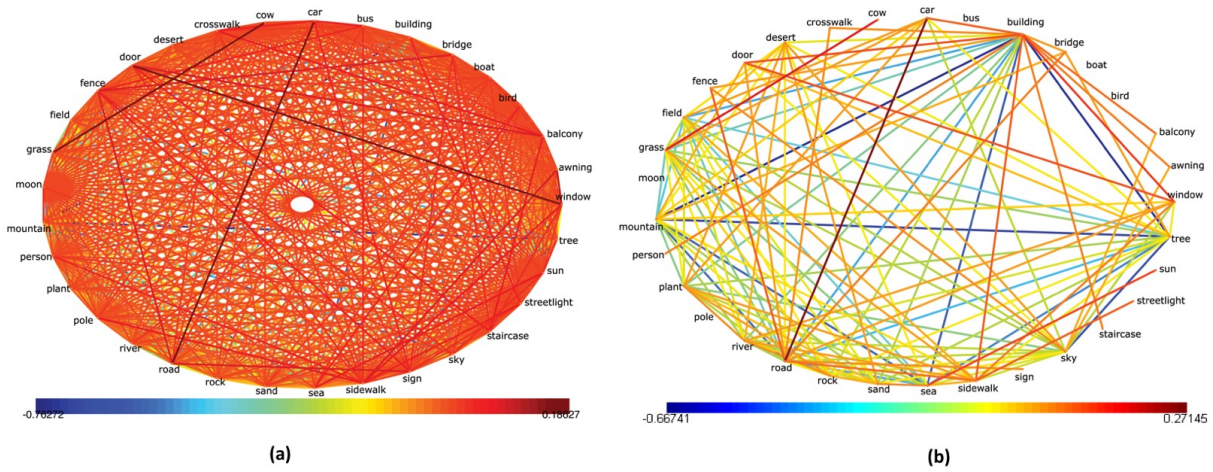


Figure 4.8: On top row we show two graphs: (a) obtained using an empirical inverse of covariance matrix, and (b) obtained by the sparse partial correlation matrix. In bottom row we show the color bar representing the scores obtained from precision matrix. As it is clear, more relevant relations are maintained and irrelevant edges are removed.

Table 4.5: Avg Accuracy Per Class

Method	StanfordBG	SIFTflow	MSRC21
Local Classifier	53.8	37.6	71.3
Our Result	**77.3**	**45.8**	**76.8**

4.7.1 Discussion on Experimental Results

Our method improves results obtained from the classifiers in two folds. First, by imposing some constraints on label graph, more meaningful pairwise costs are applied for scene labeling.

For example, in the label graph as shown in figure 4.8, building and mountain have negative partial correlation, on the other hand, building and windows have high positive correlation. Therefore, as shown in the top row of examples in figure 4.9, the mountain segments are refined. Also, since

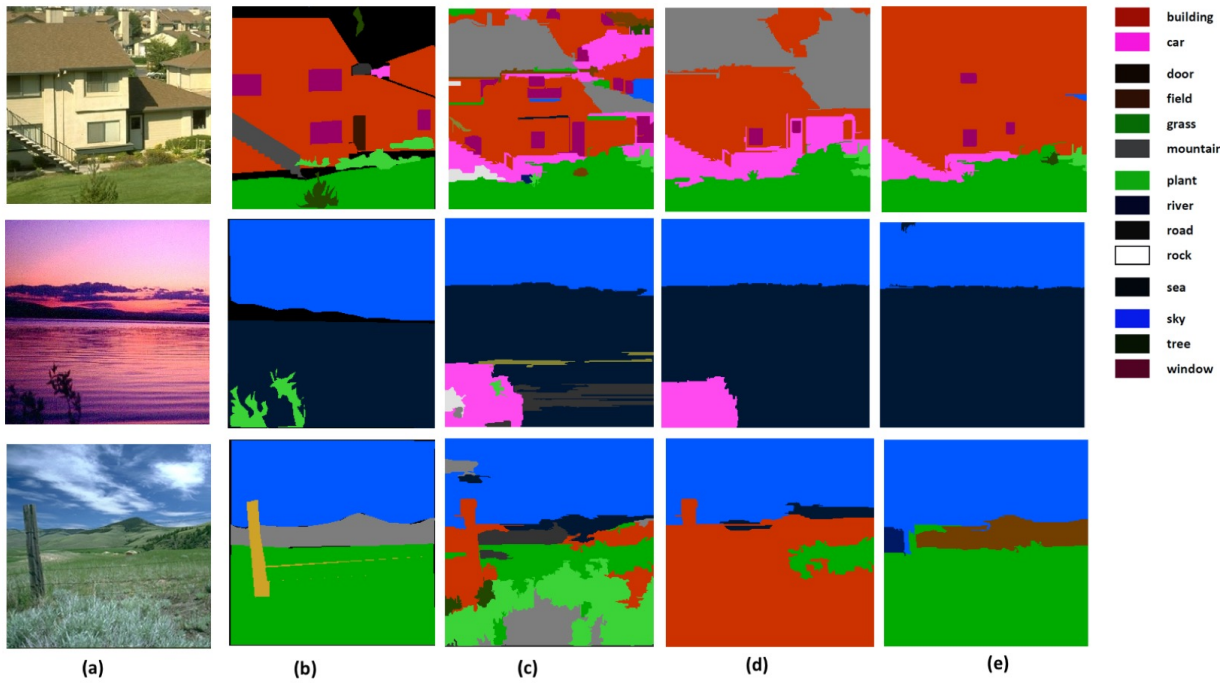
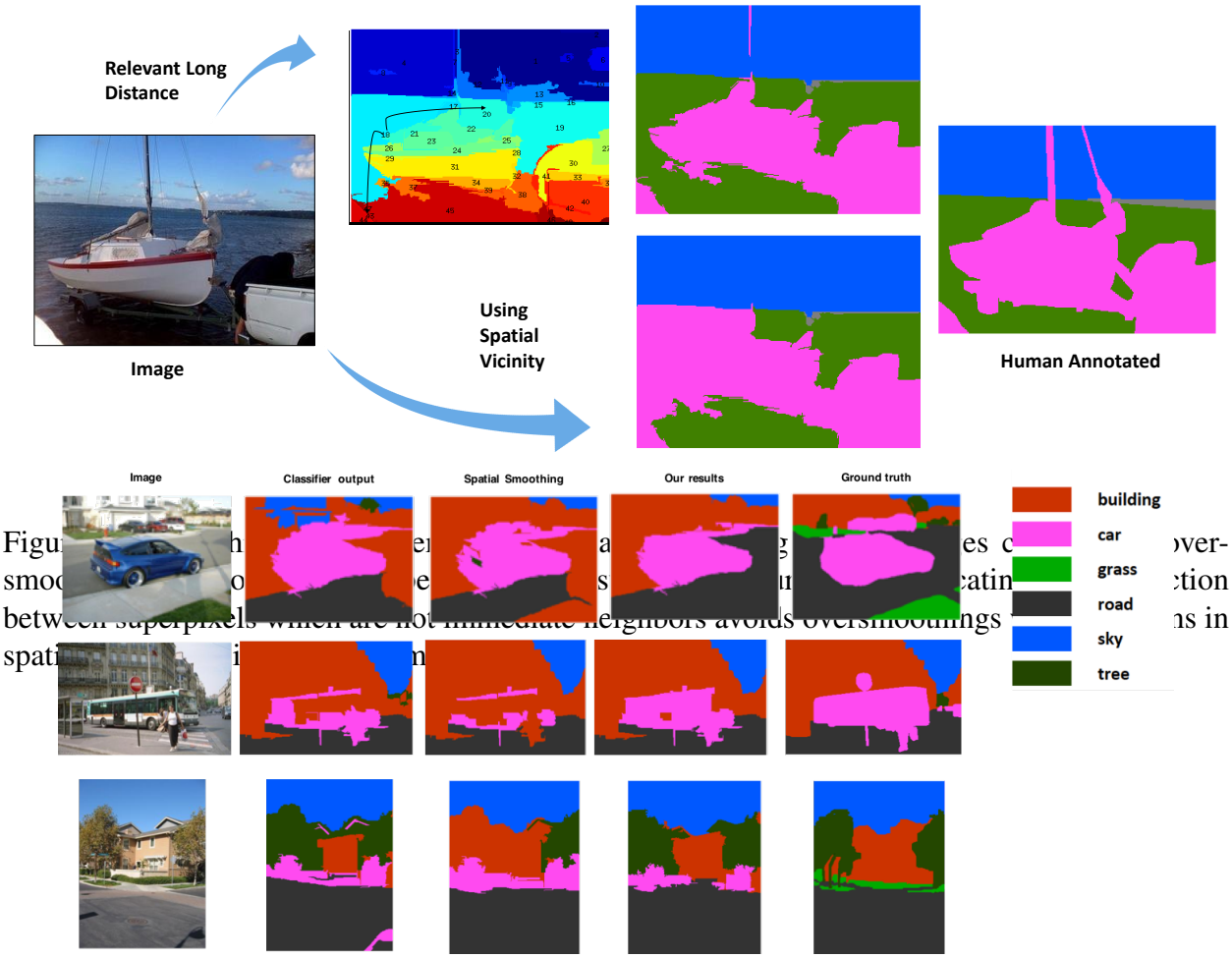


Figure 4.9: We show some sample images from SiftFlow dataset which have been properly labeled using the positive or negative correlation between labels. (a) sample image, (b) ground truth, (c) classifier results, (d) spatial neighborhood smoothing with Potts model, (e) results obtained by our approach.

windows-building have less pairwise-cost, the windows super-pixels are not smoothed out as it was the case in column (d) of Figure 4.9.

In addition, expanding the connectivities beyond immediate vicinities boosts the strength of the model. Selective edges based on partial correlation between segments prevent model from over-smoothing and enforces that correlated segments are assigned the relevant labels. For instance, in the image shown in figure 4.10 super-pixel 18 and 15 are not immediately adjacent, however in the sparse correlation matrix they are positively correlated, thus there is an edge between them and consequently, since their similarity and correlation is high, they are labeled correctly. More examples are shown in Figure 4.11.



Figu h er a g es c over-
smoc o e s u catin ction
betwe n superpixels which are not immediate neighbors avoids oversmoothings s in
spati i n

Figure 4.11: Some examples from StandfordBG dataset on the effectiveness of the long range connections in improving the labeling.

4.8 Summary

In this chapter, we proposed to incorporate context information in both label space and observation space (super-pixels) to boost local classifier results in order to better semantically label segments in an image. We used graphical lasso to estimate the sparse precision matrix of data to find relevant long distance interactions in addition to spatial smoothness.

We have shown that, this model can refine label assignment using the correlation between labels as well as segments. Also, our model does not smooth out foreground labels as can be seen in spatial labeling. We reported improved experimental results on the SIFTflow, Standford background and MSRC2 benchmark datasets.

In addition to attempt to learn and find dependency among labels, one can use constraints to further limit the space of possible assignments of labels. These constraints can be obtained from an expert, for instance human derived rules on a data set. In the following chapter, we explore using knowledge-based rules to find these constraints and apply them in a linear programming framework to solve scene labeling problem.

CHAPTER 5: SCENE LABELING THROUGH KNOWLEDGE-BASED RULES EMPLOYING CONSTRAINED INTEGER LINEAR PROGRAMMING

In this chapter, we present a novel approach to leverage knowledge based rules for pruning the search space of probable assignments of labeling in images. Our system consists of two main phases. The first phase consists of feature extraction and classifier training based on extracted local features of the sample training images. The second phase is the inference, in which for a given query image, using scores computed by the classifiers for each possible label, an objective function is maximized such that the constraints learned from knwoledge-based rules are enforced. An overview of our proposed approach is shown in Figure 5.1.

In training, first we segment images using efficient graph-based segmentation [34]. We use this specific method to be consistent with other approaches. Next, for each super-pixel, local features, including SIFT, color histogram, mean and standard derivation of color, area and texture, are extracted. Given these local features, classifiers are trained to label super-pixels using their local features. We use XGBoost (extreme gradient boosting trees forest), which is based on weak learners (trees), and tries to add new trees iteratively to improve the already built ones. Boosted Trees can be distributed easily, and they are efficient in terms of time complexity.

In the inference part, we update the classifiers scores by reducing the scores of the assignments which conflict with the defined constraints. We use the product of experts ([57], [14]) for combining the probabilities of the label assignments given by the classifiers and the degree of inconsistency obtained through constraints.

We formulate label prediction for each segment in the image by constrained optimization prob-

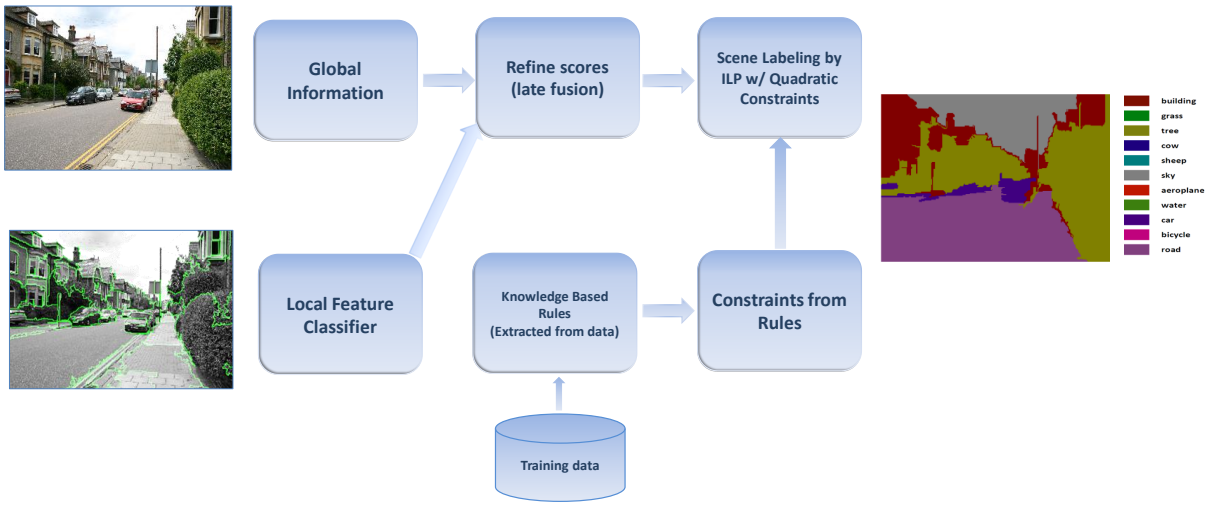


Figure 5.1: An overview of the proposed approach. For training, we begin by segmenting images into super-pixels and extracting the feature matrix. Then local classifiers, extreme gradient boosting trees, are trained. Also we find the scene-labels association matrix to capture global context. In the inference part during testing, for a given image the label scores are obtained via the classifiers results. Finally, labels scores are updated by applying constraints learned from the knowledge-based rules through the optimization of the objective function by Integer Programming.

lem, and employ Integer Linear Programing (ILP) with quadratic constraints as an optimization framework. In doing so, we assign a binary label $y_i^j \in \{0, 1\}$ to a super-pixel i, in order to find the best assignment $\mathcal{Y} = \{y_1^{j_1}, y_2^{j_2}, ..., y_n^{j_n}\}$ from different assignments y, where n is the number of super-pixels and j_n belongs to labels $1, ..., l$. Hence, given the classifier score for each super-pixel, we formulate the inference function as

$$\mathrm{y}^* = \arg\max_{\mathrm{y}} \sum_{i,j} \phi_i^j \; y_i^j, \tag{5.1}$$

where ϕ_i^j is the confidence of assigning label j to super-pixel i, provided by our local feature based classifiers which plays the role of the first expert. This objective function maximize the number of

correct predictions for the super-pixels in the image. Toward exploiting context and prior knowledge (another expert), we introduce different types of constraints such as non-coexistence, presence etc, which are explained in Section 5.3 . The general constraint, applicable to all instances, enforces that each super pixel gets exactly one label.

$$\forall i \quad \sum_{j=1}^{n} y_i^j = 1. \tag{5.2}$$

In following sections, we explain each part of the approach in detail.

5.1 Features and Local Classifiers

In this section, we explain the first step of our method. In training, we start with segmenting each sample image into super-pixels using an efficient graph-based segmentation method [34], followed by computing a feature vector (including SIFT, color mean) for each super-pixel in the image. We use the same features as used in [147].

We use a sigmoid function to rescale the classifier scores in order to give a chance to other classes, beside the one the classifier with a maximum score, to compete during the optimization phase. By doing so, if the classifier mislabels a super-pixel there is a chance that the label may get changed during the inference phase by applying the constraints. We adapt the parameters of the function using the validation sample data. Also, the sizes (areas) of super pixels are multiplied by the scores to make larger super-pixels more important during the optimization.

We use Extreme Gradient Boosting [37] with softmax objective function to categorize each super-pixel in an image. Since the training data inevitably is noisy, super-pixels may break the structure

of the data, the bagging using a subset of training examples and subsets of features are used to reduce the effects of the noisy data.

Unlike some of the other methods, which train object detectors in addition to the region classifiers, we only use region features and simple classifiers to obtain the initial label scores for super-pixels. In our experiments, boosting trees achieved better results in terms of average accuracy among all the classes, even though we discard some of the samples randomly during the training. We discard some samples since the number of sample data for some classes is enormous.

5.2 Global Context Information

In performing scene labeling, incorporating scene information, such as the places, can be helpful in order to improve classifier confidences (e.g. ϕ in Equ. 5.1). For instance, *if the probability of being a desert is high for an image presumably the chance of seeing chair or river would be low*. Therefore, we use image level categories to refine the scores of the local classifiers. We use Places CNN model [175] to find the most probable scene semantics in a given image.

Unlike pixel-level annotation, image level annotation is more feasible, thus training a deep network is doable. This deep network is trained on scene labeled images and includes 205 places categories. Using our training data, we learn a mapping between these categories and the label set. To do so, we employ non-negative sparse regression formulation, and extract a weight matrix $U \in \mathbb{R}^{L \times G}$ for scene-label association as follow:

$$\min_{U \geq 0} \|V - US\|_F^2 + \lambda \|U\|_1, \tag{5.3}$$

where $S \in \mathbb{R}^{G \times M}$ is the confidence score matrix for scene-categories of training images, G is the number of scene categories, here 205, and M is the number of training images. $V \in \mathbb{R}^{L \times M}$ is a

matrix of labels occurrence in the corresponding images where L is the number of labels in the dataset. In the interest of putting emphasis on learning the mapping between scene categories and smaller super-pixels and rare classes, if label l_i presents, i_{th} element of V is determined as follows,

$$V_i = 1 - n(l_i)/\sum_j n(l_j). \tag{5.4}$$

,

where $n(l)$ is the number of pixels belonging to class l. The solution of this problem can be efficiently obtained by FISTA algorithm [6], which is implemented in SPArse Modeling Software(SPAMS) [93].

Then during testing time, for a given image, using scene categories scores for the image and the weight matrix U found in training, the most confident labels are obtained in a vector of size L and used to refine the label assignment obtained earlier from classifiers (ϕ in Equ. 5.1), via late fusion.

5.3 Extracting Rules and Creating Constraints

In this section, we describe the types of constraints that we add to the aforementioned optimization problem (5.1). Note that even though these constraints can be obtained by common sense knowledge, we explore the training data to discover plausible rules in the context of semantic labeling of images. For instance, the common sense about labels *sky* and *building* would be: *sky is always above the building*; however, since images are 2D projections of a 3D world this is not always the case. Therefore, we capture these types of constraints as relative constraints with some penalty if they are violated. We call them soft-constraints and describe them in Section 5.4.

The first type of constraints are **spatial constraints**; whether two labels can have a particular

spatial relationship or not. For example, *sky* is always seen above the *sea*. For each super-pixel, we keep the bounding boxe information, then we apply a rule: *If the label for a super-pixel is sea, super-pixels below it cannot get label sky*. These types of constraints can be formulated as follows:

$$y_i^a \sum_{j=i+d}^{n} y_j^b \leqslant 0, \tag{5.5}$$

where a and b respectively are labels of the lower and upper parts of the image, d is the displacement of super-pixel j from super-pixel i and n is the number of super-pixels. By exploring the data, we can find the hard constraints belonging to this group (e.g (sky, field), (sea, sand), (ceiling, floor)...).

The second type of constraints are **mutually-exclusive**, which represent cases *when labels a and b, never occur together in an image*. These can be formulated as follows

$$\sum_i y_i^a \sum_j y_j^b \leqslant 0. \tag{5.6}$$

Presence (existence) is another type of constraints which can be applied. According to this constraint, *if a certain label a appears in the image, then there should be at least one super-pixel with label b*. We can express these constraints through either one of the following equations:

$$\sum_i y_i^a \sum_i y_i^b \geqslant 1, \tag{5.7}$$

$$\sum_i y_i^a - \sum_{i,j} y_i^a y_j^b \leqslant 0. \tag{5.8}$$

For example, we have found that whenever the label *balcony* appears in the image there exists, at

least, one region labeled as a *building*, similarly whenever the label *pole* appears in the image, at least, one region is labeled as a *road*. Note that; this constraint is different from "co-occurrence". In co-occurrence two labels frequently appear together in images; the constraint is not necessarily always valid, thus, it can suitably be formulated as a soft-constraint. These rules are expendable, for example, we add adjacency constraints to enforce the neighboring super-pixels receive the same labels, or apply some constraints on the other features of the specific labels. In Table 5.1 a summary of rules is shown. In "adjacency" rule, $\neg(y_i \oplus y_j)$ is XNOR and can be implemented in Binary Integer Programming by $y_i - y_j = 0$ constraint.

We define the rules as described above, and then we use sample data to find relations between classes which follow these rules. In order to force the model to follow these rules, we formulate them as constraints and add them to the optimization.

We create a tensor with the size of $L \times L \times N_R$, where L is number of labels and N_R is number of relations. Each matrix of this tensor shows the frequency of a relation occurring between classes, and each cell indicates the frequency of a particular relationship between two categories (class labels). For example, to find labels that have a specific relation, we count how many times that relation occurs in sample data for each pair of classes. If the value is higher than a threshold, we consider it as a constraint. In addition, we distinguish between soft and hard constraints whether or not the converse or alternative occurs in the sample data.

For instance, about the spatial relation like above, for a pair semantic labels (ceiling, floor) the relations is always true. Therefore, we consider it as a hard constraint. While, for (sky, building) there are some images in which the building super-pixels are above the sky, thus we consider this relation as soft constraint, that way our formulation can tolerate violating this constraint with a penalty.

5.4 Integer Linear Programming with Soft Constraints

Our proposed approach for extraction of rules not only helps us in deriving the hard constraints, but it also contributes to providing the soft constraints such as *co-occurrence* and *relative spatial* constraints. Since some of the rules may not necessarily be satisfied by all the data, we use 0-1 soft-constraint modeling [141] and define soft constraints by introducing a new binary variable, z_k, for each constraint k and an associated penalty, c_k, which indicates the degree of violation (how the confidence of assignment should be reduced when the constraint is not satisfied). Then objective function in Equation 5.1 becomes:

$$\arg\max_{y} \sum_{i,j} \phi(i,j)\ y_i^j - \sum_{k} c_k(1 - z_k), \tag{5.9}$$

which implies that if constraint k is violated z_k would be zero and consequently a penalty will be imposed to the optimization function. Moreover, we need to connect the z_k to the constraint C_k as $z_k \leftrightarrow C_k$. We achieve that by using logical representation and adding these constraints into the objective function. For example, *sky is often above the building*; however, based on the rules that we have extracted from the database, this is not always true. Therefore, we change the constraint to soft constraint as:

$$y_i^{building} \sum_{j=i+d}^{n} y_j^{sky} = 0 \leftrightarrow z_k, \tag{5.10}$$

here $\leftrightarrow$ is an equivalence for the conditional constraint, that is if constraint k is violated, z_k will be zero, d is displacement. Consequently, in the Equation 5.9, $c_k(1 - z_k)$ will be a positive number, which is subtracted from the score. In order to formulate if-else conditions in the Integer

Table 5.1: Summary of the rules extracted from the sample data.

Name	**Logic representation**	**Examples**
non-coexistence	$\mathbf{y}^{l_1} \wedge \mathbf{y}^{l_2} = 0,\ \mathbf{y}^{l} = (y_1^l \vee y_2^l \vee ... \vee y_n^l)$	(desert, sidewalk)
spatial	$y_i^{l_1} \wedge \mathbf{y}_{i+d}^{l_2} = 0,\ \mathbf{y}^{l} = (y_{i+d}^l \vee ... \vee y_n^l)$	(road below window)
presence	$y_i^{l_1} \Rightarrow \mathbf{y}^{l_2},\ \mathbf{y}^{l} = (y_1^l \vee y_2^l \vee ... \vee y_n^l)$	(balcony, building)
co-occurrence	$\mathbf{y}^{l_1} \wedge \mathbf{y}^{l_2} = 1,\ \mathbf{y}^{l} = (y_1^l \vee y_2^l \vee ... \vee y_n^l)$	(car, road)
adjacency	$y_i^{l_1} \equiv y_j^{l_2} = \neg(y_i^{l_1} \oplus y_j^{l_2}) = 1$	(sky, sun)

Programming formulation we use the binary variables and add the constraints as inequalities.

The penalty values are obtained statistically from the dataset by finding the probability of each case divided by the number of its appearances. For example, *sky* and *building* most of the time occur together, but not all the time. Therefore, we find a penalty using the following formula, which takes into account the frequency of the constraint violations in the data,

$$c_k = -\log \frac{P(C_k = 0)}{P(C_k = 1)}, \tag{5.11}$$

where $P(C_k = 0)$ and $P(C_k = 1)$ are respectively the probability of violating and satisfying the constraint k.

5.4.1 Solving Integer Programming

We are able to solve the Integer Programming problem of our size (equ. 5.1 with constraints from Table 5.1), in a short time using Gurobi toolkit [55], which can solve 70 IQP per second on a desktop computer. Our task is to formulate the constraints in ILP format, as though each binary

variable is assigning label i to superpixel j, then we give the model with constraints to GUROBI to optimize.

GUROBI solver employs piece-wise linear optimization and relaxes LP to solve the problem. Our constraints reduces the search space in comparison to fully connected graph, e.g mutual exclusions constraints or geometric constraints decrease the number of possible solutions, therefore the dependencies and interactions are much fewer than those in fully connected graph. The solver uses a piece-wise linear optimization and relaxed LP to solve the integer programming. Also, it is feasible to convert the quadratic constraints to linear ones by adding slack variables due to the binary nature of our variables.

5.5 Experiments and Results

We use the SIFTFlow dataset [85], which consists of 2,488 train and 200 test images collected from LabelMe [123]. These images are from broad range of categories such as natural images (coasts, country, etc.) and urban images. Ground truth images are available, in which each pixel of the image is labeled from 33 classes including different objects (e.g., cars, windows) and stuff (e.g., sky, sea).

In Table 5.3, the accuracy results of our method are reported, and compared to the state of the art methods, which employ similar features. Our method obtains comparable per-pixel accuracy; even though we do not use object detectors or massive training algorithms. Note that slight improvement by [45] over our method is due to use of Fisher vectors; which if used in our method will also boost our performance. Also in Table 5.2, we show the accuracy gained in different steps of our method, indicating the improvement due to addition of constraints.

In Figure 5.2, we show some qualitative results from SIFTflow data set including sample images,

ground truth labeling, results from [148], and results obtained by our method before optimization and after applying constraints based optimization. As it is clear, we are able to improve the results without any side-effects such as over smoothing. Also, our method achieves promising results in terms of per-class accuracy. For instance, in the third-row super-pixels below the sea are labeled as mountain by local classifiers, yet constraints such as *mountains are not below sea*, are able to handle the miss-labeling and change it to *tree*. Note that plant and tree are very similar in terms of the appearance.

Table 5.2: Detailed Results on SIFTFlow dataset

Method	Avg Accuracy
Local Features + Classifiers	72.8
Local Features + Global Context	77.3
Local + Global + Rules	80.9

Table 5.3: Comparison on SIFTflow dataset

Method	Per-Pixel	Per-Class
Farabet natural [30]	78.5	29.6
Farabet balanced [30]	74.2	46.0
Tighe [148]	78.6	39.2
Collage Parsing [153]	77.1	41.1
Gerorge w/out Fisher Vectors[45]	77.5	47.0
Gerorge Full [45]	81.7	50.1
Ours	80.9	50.3

The second data set which we assessed our approach on is LMSun [148]. This data set contains 45,676 training images and 500 test images including indoor and outdoor scenes, with different image sizes. However, the number of available samples for different labels are not balanced, and some labels are rare.

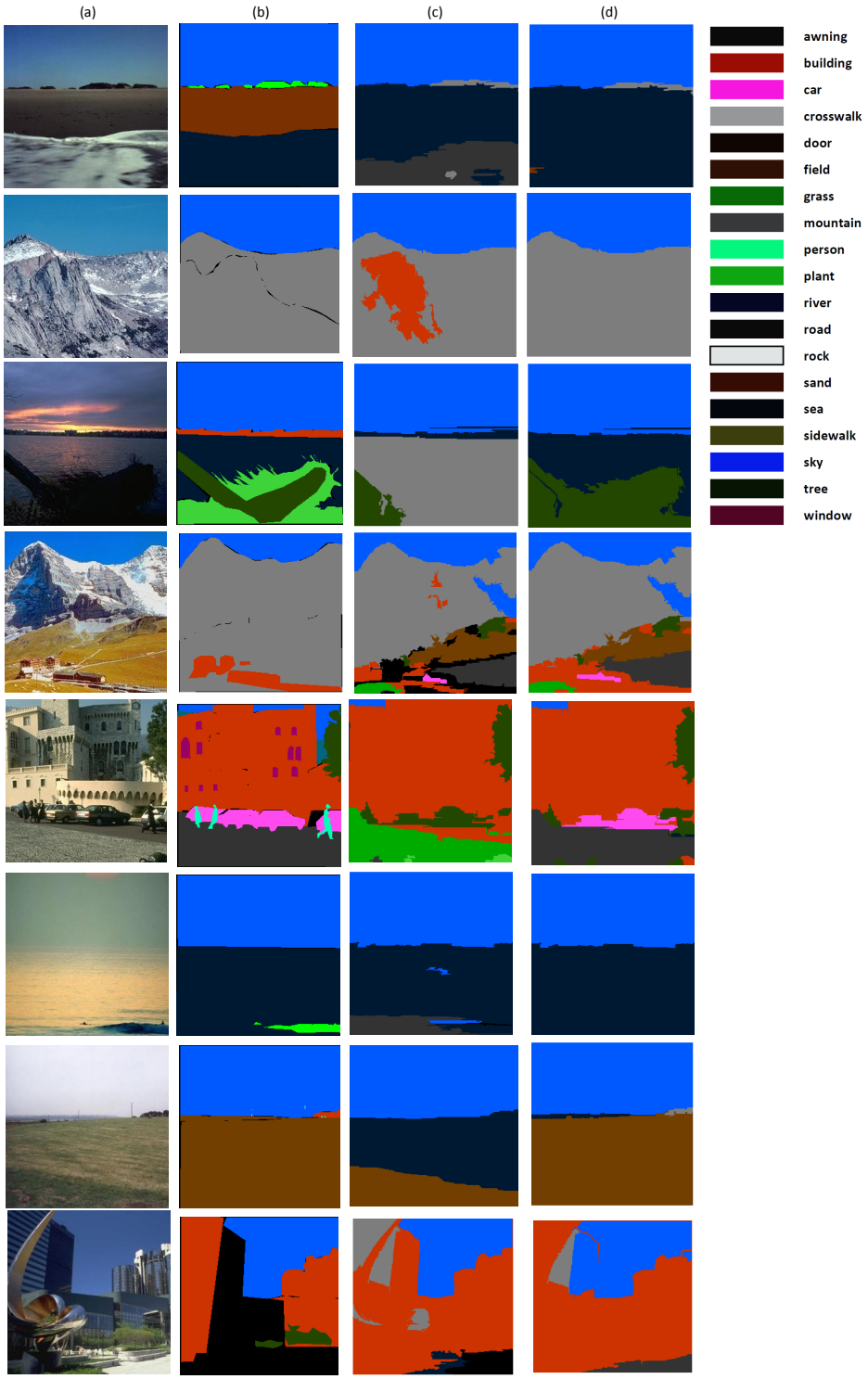


Figure 5.2: Examples results obtained by our method on SIFTFlow dataset, (a) query images, (b) ground truths, (c) initial classifier outputs and (d) our final results.

While some labels, for instance, sky and building, have a large number of samples, some others have fewer samples. To make the training more feasible, we use all samples from rare classes and only 25% of samples from common classes for training our extreme gradient boosting classifiers.

Also, in learning as well as while using the scene-label associations we assign more weights to smaller super-pixels and rare classes to avoid the influence of common labels such as *sky* or *building*. As shown in 5.4, with simple features, we can improve the per-pixels accuracy results of classifiers by about 9% and more importantly our method obtains better results in terms of per class accuracy due to the fact that in our model context information aids rare classes and, therefore, small super-pixels receive correct labels. Even though using Fisher vectors in [45] results in per-pixel accuracy close to our approach, our approach achieves better accuracy in terms of per-class accuracy.

Table 5.4: Comparison on LMSun [148] dataset

Method	Per-Pixel	Per-Class
Tighe [148]	61.4	15.2
Gerorge w/out Fisher Vectors [45]	58.2	13.6
Gerorge Full [45]	61.2	16.0
Ours Local Classifiers	53.1	13.4
Ours Local Classifiers +Global Context	57.3	14.6
Ours Local + Global + Rules	61.5	17.3

We also applied our method on MSRCV2 data set [133], which has 591 images from 23 classes. We use the provided split, 276 images for training and 255 images for testing. The qualitative results are shown in Figure 5.3 and our results are presented in Table 5.5. Similar to previous experiments, in this dataset using context and rules improves the classifier results. As shown in Figure 5.3, for instance, rules that *car* cannot be above airplane refines the labels in the last row of the figure.

Table 5.5: Detailed results on MSRCV2 [133] dataset

Method	Per Pixel Accuracy
Harmony Potentials [1]	83
Segment CRF with Co-Occurrence [77]	80
Local Features + Classifiers	76.7
Local Features + Global Context	78.3
Local + Global + Rules	**84.4**

5.6 Discussion

The proposed method can get similar results compared to other models such as CRF while having fewer edges i.e., not considering fully connected graphs. In our approach, two sources of information, including visual features and high-level knowledge-based constraints, are combined to obtain better results. Some relevant information about the data, such as spatial relationships or non-coexistence, which is hard to learn from the features automatically, are easily captured by our method and used to refine the labels. Our approach can easily scale and generalize.

In our approach, we add constraints to the trained model, so without retraining, we can add constraints which are declarative and hard to model using only features. Hence, in cases when sufficient training data for some categories (class labels) is not available, the constraints help us to obtain better results. Also, when new classes are added to the data set, the proposed method can model the dependencies between classes without learning the pairwise terms, since we keep the features (classifier learning) and constraints separate.

As shown in experiments on different data sets, with a various number of labels, our method obtains promising results. The constraints are assumed to be trusted, and penalties are obtained from data by simply finding the frequency in contrast to learning from the features. It should be noted that our primary contribution is improving the labeling on top of classifier scores; therefore,

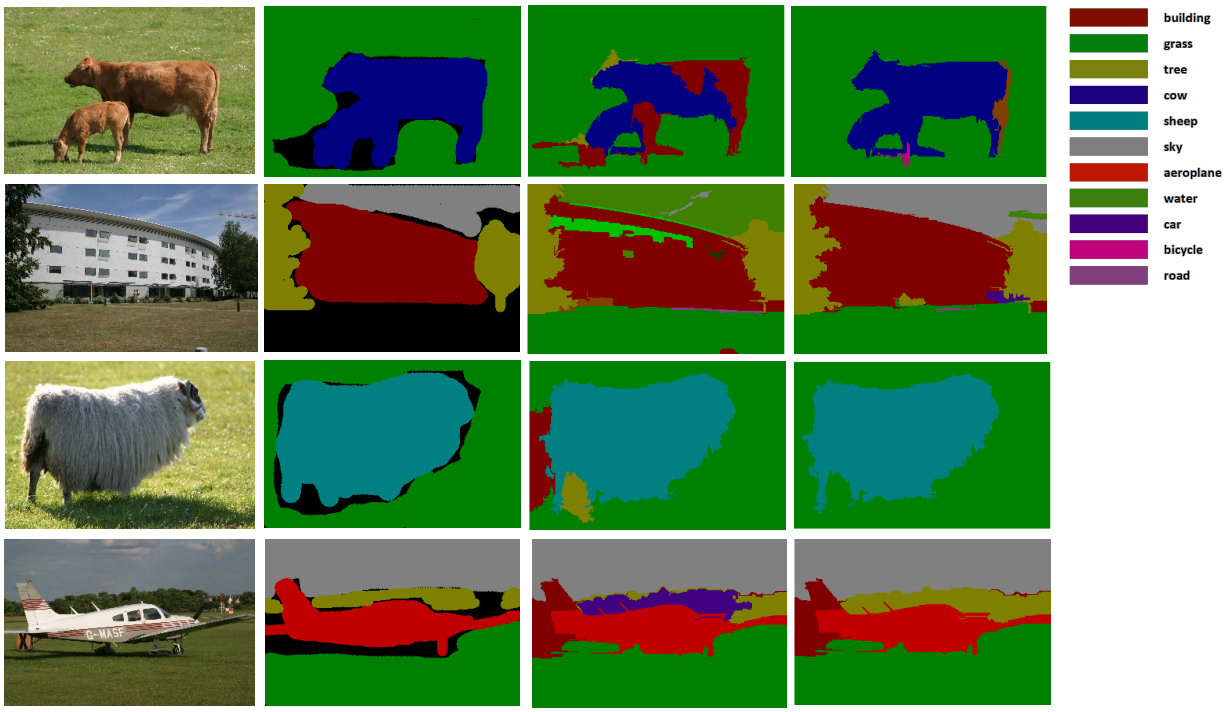


Figure 5.3: Example results obtained by our method on MSRC dataset, (a) query images, (b) ground truths, (c) initial classifier output and (d) our final results.

using extensive classifiers, such as deep learning, can further boost the final results.

5.7 Summary

In this chapter, we proposed a novel scene labeling approach, in which we use an enhanced inference method that enables the model to incorporate general constraints structure. We use an integer programming formulation, which can be solved by linear programming relaxation to address the problem. We also proposed to use soft constraints in addition to hard constraints to make the model more flexible. Experimental results on three data sets show the effectiveness of our method.

In addition to pairwise costs and relations between regions in the image, features and local classifiers play a crucial role in semantic segmentation. Deep neural networks have revolutionized machine learning approaches in particular in computer vision applications. Hand crafted features are no longer popular and the proposed deep methods have achieved significant improvements in classification.

However, training deep networks require enormous amount of annotated data. In particular, for semantic segmentation methods, annotation should be at pixel level, however obtaining these annotations are expensive and time-consuming. Therefore, in next chapter we present a new approach usging generative adversarial networks to leverage from unlabeled data or weakly-labeled data (e.g. image level annotation) to label pixels in images.

CHAPTER 6: SEMI SUPERVISED SEMANTIC SEGMENTATION USING GENERATIVE ADVERSARIAL NETWORK

Semantic segmentation has been a long standing challenging task in computer vision and needs a significant number of pixel-level annotated data, which is often unavailable. To address this lack of annotations, in this chapter we leverage, on one hand, a massive amount of available unlabeled or weakly labeled data, and on the other hand, synthetic (fake) images created through Generative Adversarial Networks. In particular, we propose a semi-supervised framework based on Generative Adversarial Networks (GANs), which consists of a generator network to provide extra training examples to a multi-class classifier, acting as discriminator in the GAN framework, that assigns sample a label y from the K possible classes or marks it as a fake sample (extra class). The underlying idea is that adding large fake visual data forces real samples to be close in the feature space, which, in turn, improves multiclass pixel classification. Our goal here is to exploit unlabeled data as well as generated data to find a structure that can support the semantic segmentation task as shown in Figure 6.1.

Since labeled data can be hard to obtain, and unlabeled data is inexpensive, semi-supervised learning has gained attentions among researchers in many studies. The goal of semi-supervised learning is to exploit labeled and unlabeled data to achieve better performance than using each one alone. Let $p(x)$ represents some knowledge distribution about unlabeled data x, which could lead to some useful information about $p(y|x)$; classification of data x to a class y. One of the constraints, which we will use in this chapter, is the smoothness constraint: If two data points x_1, x_2 are close in the input feature space, then the corresponding outputs (classifications) y_1, y_2 should also be close [15]. Therefore, data points lying on the same manifold or belonging to the same cluster are more expected to be classified to the same class. Thus, the end goal here is to leverage the unlabeled

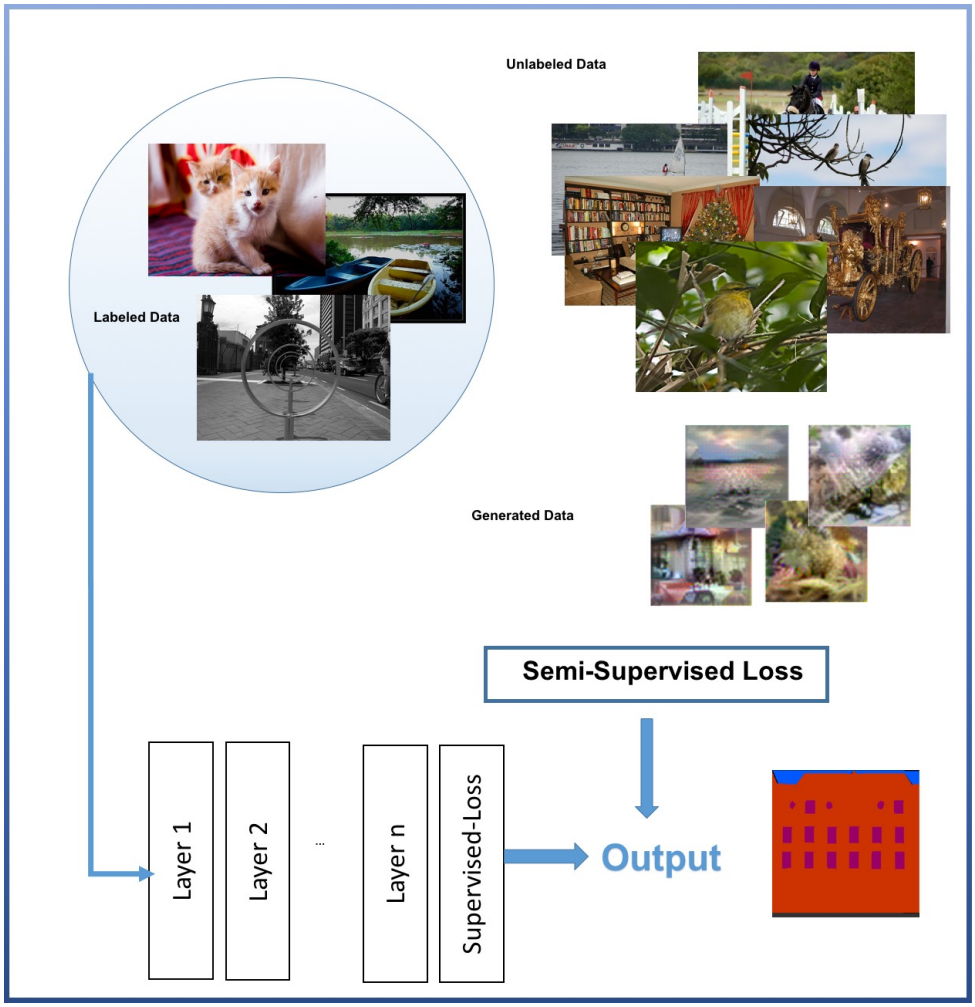


Figure 6.1: Given a small set of labeled data and available unlabeled data and generated data, our aim is to train a deep neural network in semi-supervised fashion. The total loss of the framework is the summation of unlabeled, labeled and and generated data losses.

data to find this structure.

Given a small set of labeled data, N_l, and unlabeled data, N_u, we can formulate the problem as

general cost minimization as given below:

$$L = \sum_{n=1}^{N_l} L_l(y_n, x_n) + \lambda \sum_{n=1}^{N_u} L_u(x_n), \tag{6.1}$$

where L_l is the loss function for the labeled data (supervised part), L_u is loss estimation for unlabeled data and K is the number of classes. In supervised learning, the main goal is to derive a function $F : X \rightarrow Y$ for each training pair (x_i, y_j) such that $y_i = F(x_i)$. Neural networks solve this problem using back-propagation to find a local optimum. For the semi-supervised training, we use training samples with pixel-level annotation $(x_i, y_i); i = 1...N_l$, and unlabeled data $x_i, i = 1...N_u$, assuming (x, y) belongs to a data distribution, p_{data}. We assume that we have access to a limited number of labeled pairs (x, y) and plentiful amount of unlabeled data points. In the case of unlabeled data, where y_is are not available, we use the following constraint: if there is a valid F for all training data, data distribution of $F(x)$ and y should be equal [142]. Therefore, we employ a generative model to impose this constraint to these two distributions. Generative Adversarial Networks (GANs) have been used succesfully to train a generative model. We use GAN to impose our unsupervised objective.

6.1 Generative Adversarial Network (GAN)

Generative Adversarial Network (GAN) is a framework introduced by [47] to train deep generative models. It consists of a generator network, G, whose goal is to learn a distribution, p_z matching the data, and a discriminator network D, which tries to distinguish between real data (from true distribution $p_{data}(x)$) and fake data (generated by the generator). G and D are competitors in a minmax game with the following formulation:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[log(D(x))] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z))], \tag{6.2}$$

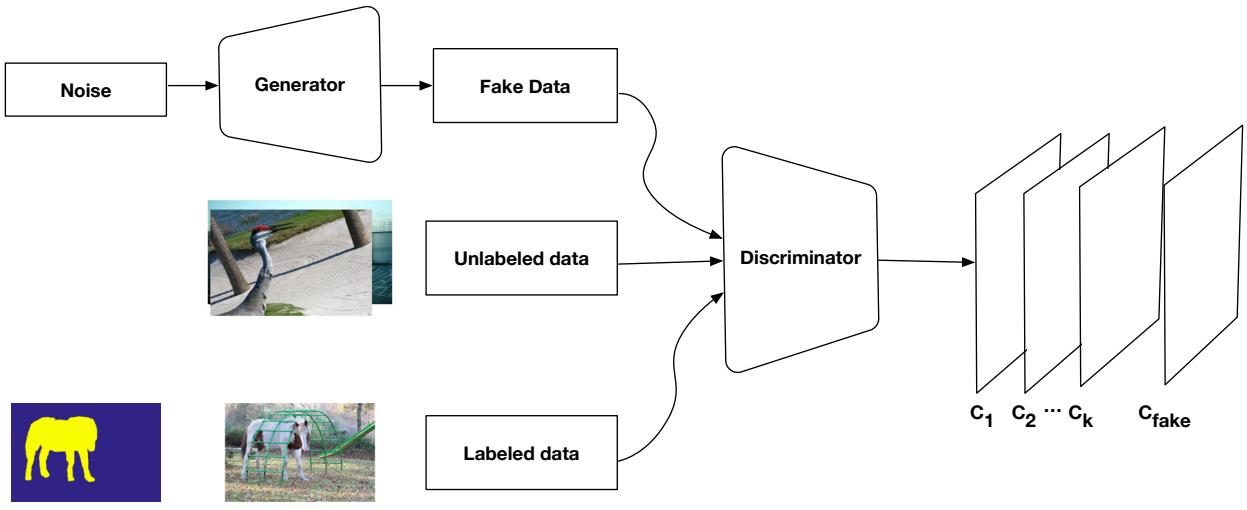


Figure 6.2: Proposed semi-supervised convolutional GAN architecture. Noise is used by the Generator to generate an image. The Discriminator uses generated data, unlabeled data and labeled data to learn class confidences and produces confidence maps for each class as well as a label for a fake data.

where $\mathbb{E}$ is the empirical estimate of expected value of the probability. G transforms a noise variable z into $G(z)$, which basically is a sample from distribution p_z, and ideally distribution p_z should converge to distribution p_{data}. Minimizing $\log(1 - D(G(z))$ is equivalent to maximizing $\log(D(G(z))$, and it has been shown that it would lead to better performance, so we follow the latter formulation.

6.2 Semi Supervised Learning using Generative Adversarial Networks

In semi-supervised learning, where class labels (in our case pixel-wise annotations) are not available for all training images, it is convenient to leverage unlabeled data for estimating a proper prior to be used by a classifier for enhancing performance. In this approach we adopt and extend GANs, to learn the prior fitting the data, by replacing the traditional discriminator D with a fully convolutional multiclass classifier, which, instead, of predicting whether a sample x belongs to the data

distribution (it is real or not), it assigns to each input image pixel a label y from the K semantic classes or mark it as a fake sample (extra $K+1$ class). More specifically, our discriminator $D(x)$ is a function parametrized as a network predicting the confidences for K classes of image pixels and softmax is employed to obtain the probability of sample x belonging to each class. In order to be consistent with GAN terminology and to simplify notations we will not use D_k and use D to represent pixel-wise multi-class classifier. Generator network, G, of our approach maps a random noise z to a sample $G(z)$ trying to make it similar to training data, such that the output of D on that sample corresponds to one of the real categories. D, instead, is trained to label the generated samples $G(z)$ as fake. Figure 6.2 provides a schematic description of our semi-supervised convolutional GAN architecture and shows that we feed three inputs to the discriminator: labelled data, unlabelled data and fake data. Accordingly, we minimize a pixel-wise discriminator loss, $\mathcal{L}_D$, in order to account for the three kind of input data, as follows:

$$\mathcal{L}_D = -\mathbb{E}_{x\sim p_{data}(x)} log(D(x)) + \gamma\, \mathbb{E}_{x,y\sim p(y,x)}[\text{CE}(y, P(y|x, D))] - \mathbb{E}_{z\sim p_z(z)} log(1 - D(G(z)), \quad (6.3)$$

where

$$D(x) = [1 - P(y = fake|x)], \quad (6.4)$$

with $y = 1 \cdots K$ being the semantic class label, $p(x, y)$ the joint probability of labels (y) and data (x), CE is the cross entropy loss between labels and probabilities predicted by $D(x)$. The first term of $\mathcal{L}_D$ is devised for unlabeled data and aims at decreasing the probability of pixels belonging to the fake class and increasing the probability of real semantic classes. The second term accounts for all pixels in labeled data to be correctly classified in one of the K available classes. While the third loss term aims at driving the discriminator in distinguishing real samples from fake ones generated by G. γ is a parameter used for balancing generator and discriminator (segmentation) tasks; decreasing gamma gives more emphasis to the generator rather than discriminator (segmentation). We empirically set $\gamma = 2$.

The generator loss, $\mathcal{L}_G$ is defined as follows:

$$\mathcal{L}_G = \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z))]. \tag{6.5}$$

Note that our GAN formulation is different from typical GANs, where the discriminator is a binary classifier for discriminating real/fake images, while our discriminator performs multiclass pixel categorization.

6.3 Semi Supervised Learning with Additional Weakly labeled data using Conditional GANs

The recent extension of GANs is conditional GANs [105], where generator and discriminator are provided with extra information, e.g., image class labeles. The traditional loss function, in this case, becomes:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x,l \sim p_{data}(x,l)}[log(D(x,l))] + \mathbb{E}_{z \sim p_z(z,l), l \sim p_l(l)}[log(1 - D(G(z,l),l)], \tag{6.6}$$

where $p_l(l)$ is the prior distribution over class labels, $p_{data}(x,l)$ is joint distribution of data, x, and labels l, and $p_z(z,l)$ is joint distributions of generator noise z and labels l indicating that labels l control the conditional distribution of $p_z(z|l)$ of the generator.

Semantic segmentation can naturally fit into this model, as long as additional information on training data is available, e.g., image level labels (whose annotation is much less expensive than pixel level annotation). We use this side-information on image classes to train our GAN network with weak supervision. The rationale of exploiting weak supervision in our framework lies on the assumption that when image classes are provided to the generator, it is encouraged to learn co-occurrences between labels and images resulting in higher quality generated images, which, in

turn, help our multiclassifier to learn more meaningful features for pixel-level classification and true relationships between labels.

Our proposed GAN network architecture for semi supervised semantic segmentation using additional weakly labeled data is shown in Figure 6.3. The discriminator is fed with unlabeled images together with class level information, generated images coming from G and pixel-level labeled images. Thus, the discriminator loss, $\mathcal{L}_D$, is comprised of three terms: the term for weakly labeled sample data belonging to data distribution $p_{data}(x, l)$, the term for loss of generated samples not belonging to the true distribution, and the term for the loss of pixels in labeled data classified correctly. Hence, the discriminator loss $\mathcal{L}_D$ is as follows:

$$\begin{aligned}\mathcal{L}_D = &-\mathbb{E}_{x,l\sim p_{z,l}(x,l)} \log[p(y = fake|x)] - \mathbb{E}_{x,l\sim p_{data(x,l)}} \log[p(y \in K_i \subset 1...K|x)]+ \\ &\gamma\, \mathbb{E}_{x,y\sim p(y,x)}[\text{CE}(y, P(y|x, D))],\end{aligned} \tag{6.7}$$

where K_i indicates the classes present in the image. Here, we have modified the notations for probability distributions and expectation to include label l. Conditioning space l (labeled) in loss L_D aims at controlling the generated samples, i.e., given image classes along with the noise vector, the generator attempts to maximize the probability of seeing labels in the generated images, while the goal of discriminator is to suppress the probability of real classes for generated data and to encourage high confidence of image level labels for unlabeled data. The generator loss is similar to the one used for semi-supervised case (see Eq. 6.5), and aims at enforcing the image-level labels to be present in the generated images. For unlabeled data, we use negative log-likelihood of confidences, favoring the labels occuring in the image, meaning that we add a fixed value to pixel confidences for image-level labels.

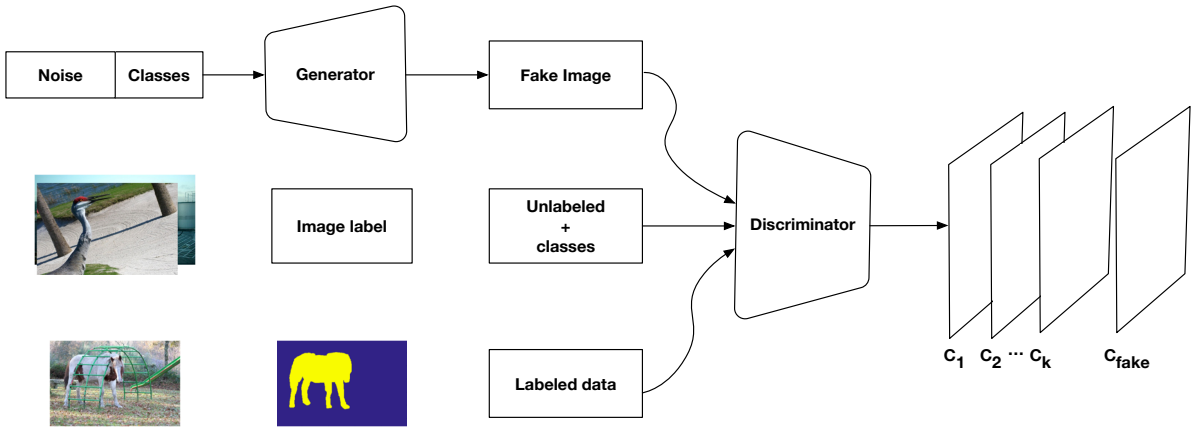


Figure 6.3: Our semi-supervised with additional weakly-labeled data convolutional GAN architecture. In addition to noise, class label information is used by the Generator to generate a fake image. The Discriminator uses generated data, unlabeled data plus image-level labels and pixel-level labeled data to learn class confidences and produces confidence maps $C_1, C_2, \ldots, C_k$ for each semantic class as well as a label C_{fake} for the fake data.

6.4 System Overview

In this section, we present the details of our deep networks, including the discriminator (classifier) and the generator. In both settings, i.e., semi-supervised and weakly-supervised approaches, the discriminator is a fully convolutional network [88] using VGG16 convolutional layers plus 1 or 3 deconvolution layers, which generates $K + 1$ confidence maps. The generator, instead, consists of 4 deconvolution layers transforming noise (and noise plus image class information) into an image (see Figure 6.4).

The generator network, shown in Figure 6.4, starts with noise, followed by a series of deconvolution filters and generates a synthetic image resembling samples from real data distribution. The generator loss enforces the network to minimize the distance between $D(G(z_i))$ and $y_i \in l_i ... l_K$, as shown in Equation 6.3.

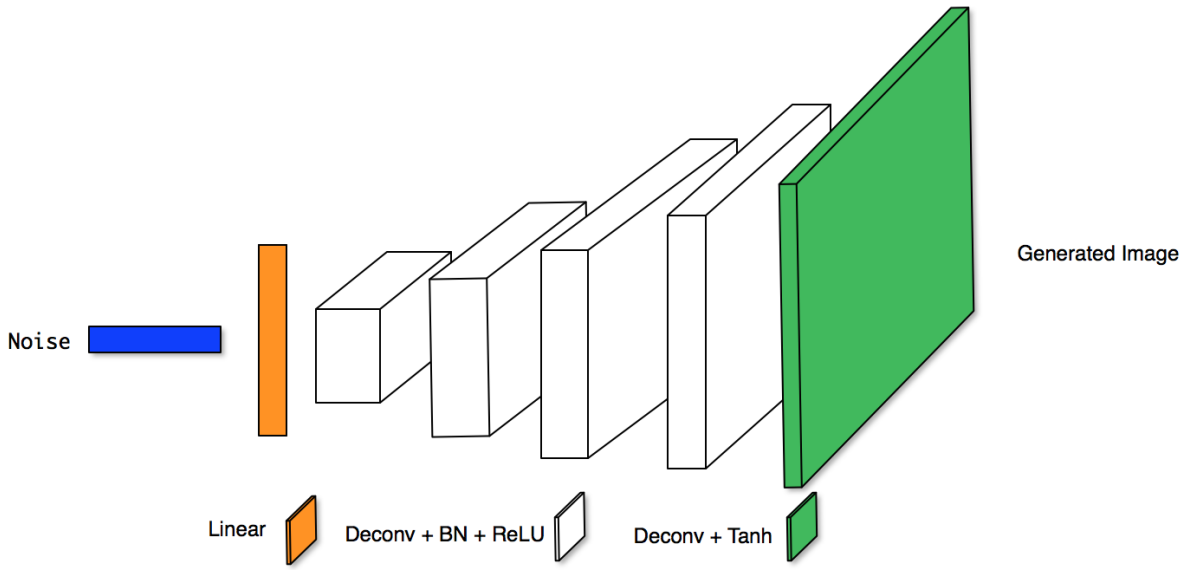


Figure 6.4: The generator network of our GAN architecture. The noise is a vector of size 100 sampled from a uniform distribution. The number of feature maps in the five different convolutional layers, respectively, are 769, 384, 256, 192 and 3.

In a typical GAN model, the discriminator's objective is to distinguish between the true data distribution and the distribution of the generated (fake) samples. In semi-supervised learning, this definition is extended to determine whether the data is from one of the classes or is generated from a noise. The loss of discriminator network shown in (figure 6.5), is the sum of cross entropy between labeled data and the output of classifiers. This enforces that the discriminator should classify pixels from the generated image (data) into the fake class and unlabeled data to the true classes.

In weakly supervised training, we impose the constraint on the generator that, instead, of generating generic images from data distribution, it produces samples belonging to specific visual classes provided as input to it. To do that, a one-hot image classes vector is concatenated to the noise sampled from the noise distribution. Afterward, the deconvolution layers are applied similar to the typical generator network and a syntactic image conditioned on image classes is generated.

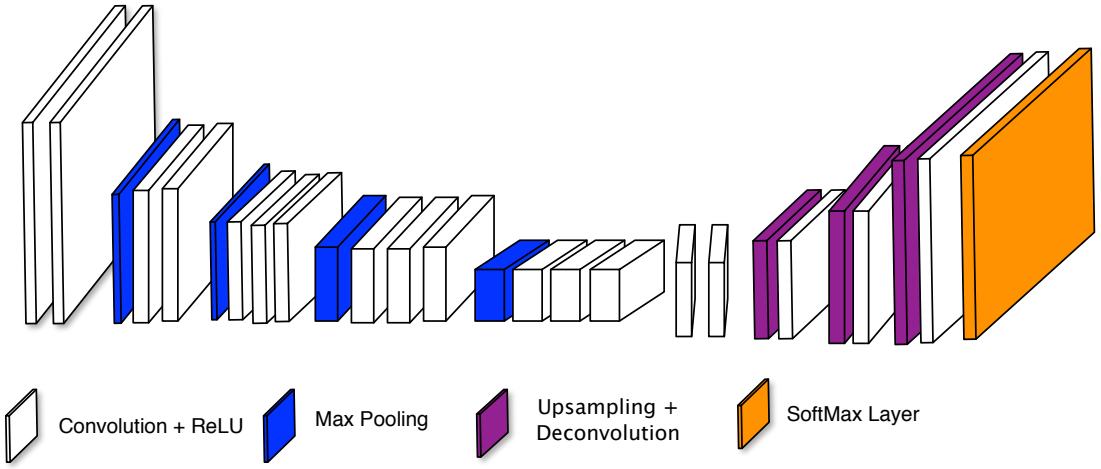


Figure 6.5: Our discriminator network in GAN architecture. The network is based on a fully convolutional VGG 16 network with deconvolution layers.

All the networks are implemented in chainer framework [150]. The standard Adam optimizer with momentum is used for discriminator optimization, and the classifier network's convolutional layers weights are initialized using VGG 16-layer net pre-trained on ILSVRC dataset. For training the generator, we use Adam optimizer with isotropic Gaussian weights. Due to memory limitations, we use a batch of size 2; however, since the loss is computed for every pixel of training images and the final loss is averaged over those values, the batch-size is not that small. We do not use any data augmentation or post-processing (e.g. CRF) in these experiments.

6.4.1 Inference

During testing, we only use discriminator network as our semantic segmentation labeling network. Given a test image, the softmax layer of the discriminator outputs a set of probabilities of each pixel belonging to semantic classes, and accordingly, the label with the highest probability is assigned

Table 6.1: The results on val set of VOC 2012 using all fully labeled and unlabeled data in train set.

method	pixel acc	mean acc	mean IU
Full - our baseline	89.9	69.2	59.5
Semi Supervised	90.5	80.7	64.1
Weak Supervised	91.3	80.0	65.8
FCN [88]	90.3	75.9	62.7
EM-Fixed [111]	-	-	64.6

to the pixel. One can use a post processing algorithms, for instance dense CRF, to improve further the results.

6.5 Experimental Results

We evaluate our method on PASCAL VOC 2012 [29], SiftFlow [85], [163], StanfordBG [48] and CamVid [11] datasets. In the first experiment for Pascal dataset, we use all training data (1400 images) for which the pixel-level labels are provided as well as about 10k additional images with image-level class labels, i.e., for each image its semantic classes are known, but not the pixel-level annotations. These images are used in the weakly supervised setting. In the second experiment on Pascal dataset, for semi-supervised training, we use about 30% (about 20 samples per class) of pixel-wise annotated data and the rest of images without pixel-wise annotations. As metrics, we employ *pixel accuracy*, which is per-pixel classification accuracy, *mean accuracy*, i.e, average of pixels classification accuracies on number of classes and *mean IU*, average of region intersection over union (IU).

Quantitative results of our method on VOC 2012 validations set are shown in Tables 6.1 and 6.2, and the qualitative results on some sample images are depicted in Figure 6.6. As shown in Table 6.2, the semi-supervised method notably improves mean accuracy about 5% to 7%. The pixel

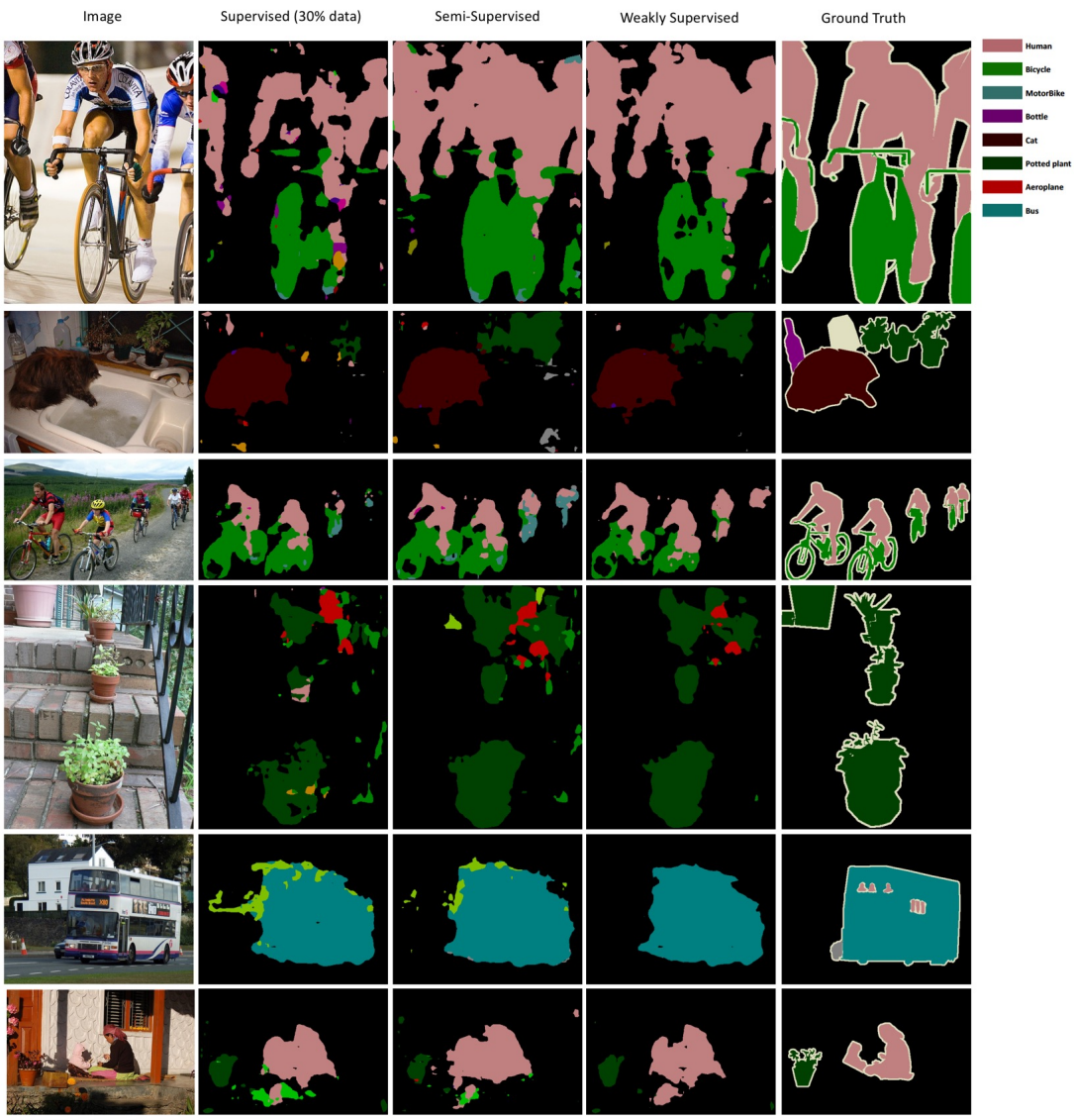


Figure 6.6: Qualitative segmentation results on VOC 2012 validation set. The first to fifth columns, respectively, show: the original images, the results of supervised learning using only 30% of labeled data, the results of the proposed semi-supervised learning using 30% labeled and about 400 unlabelled images, the results obtained using proposed weakly supervised learning with 30% of labeled data and additional 10k images with image level class labels, and the Ground Truth. Both semi-supervised and weakly-supervised learning methods outperform the fully-supervised method. Weakly-supervised approach is more successful in suppressing false positives (background pixels misclassified as part of one of the K available classes).

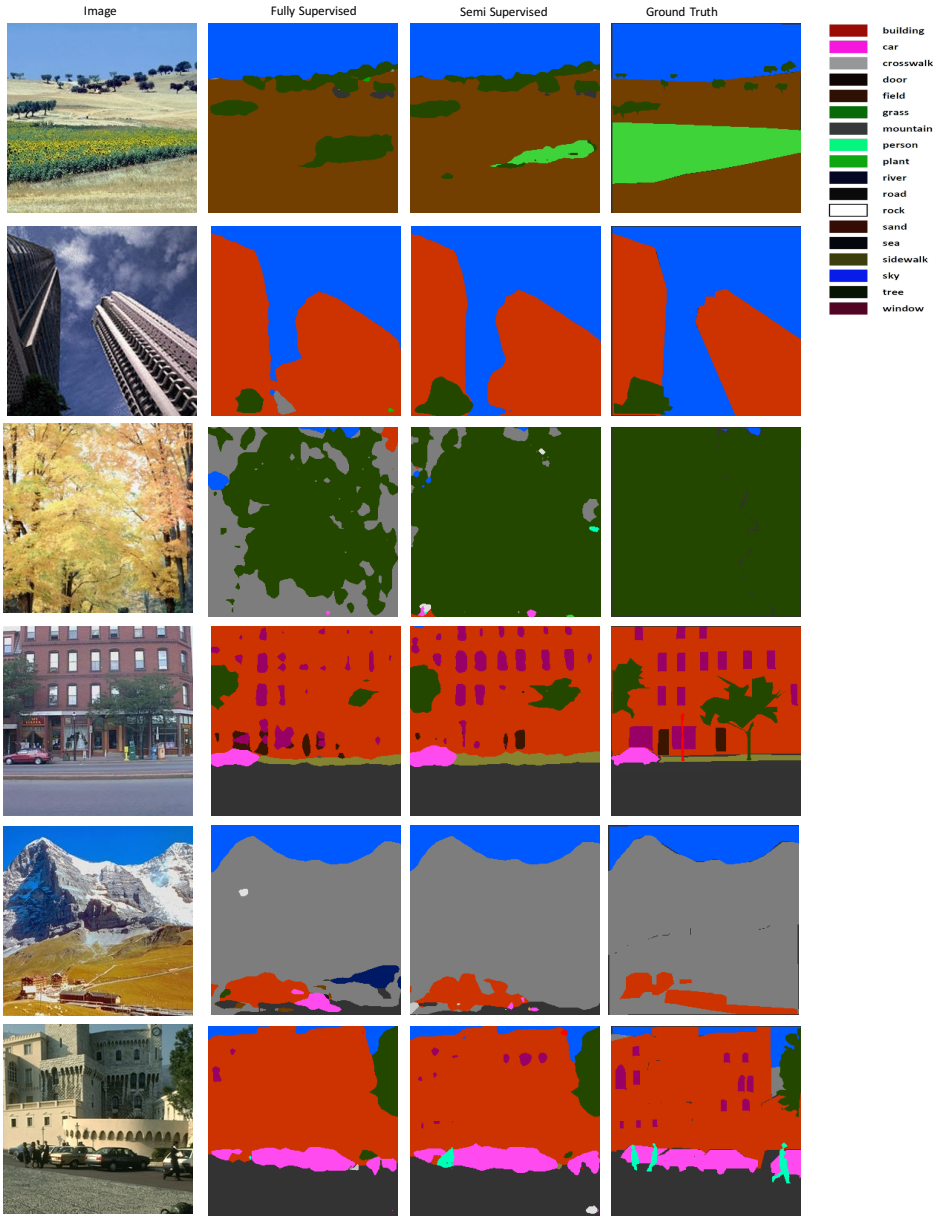


Figure 6.7: Qualitative results on SiftFlow dataset, using unlabeled data results in more accurate semantic segmentation, unlikely classes in the image are removed in semi-supervised approach.

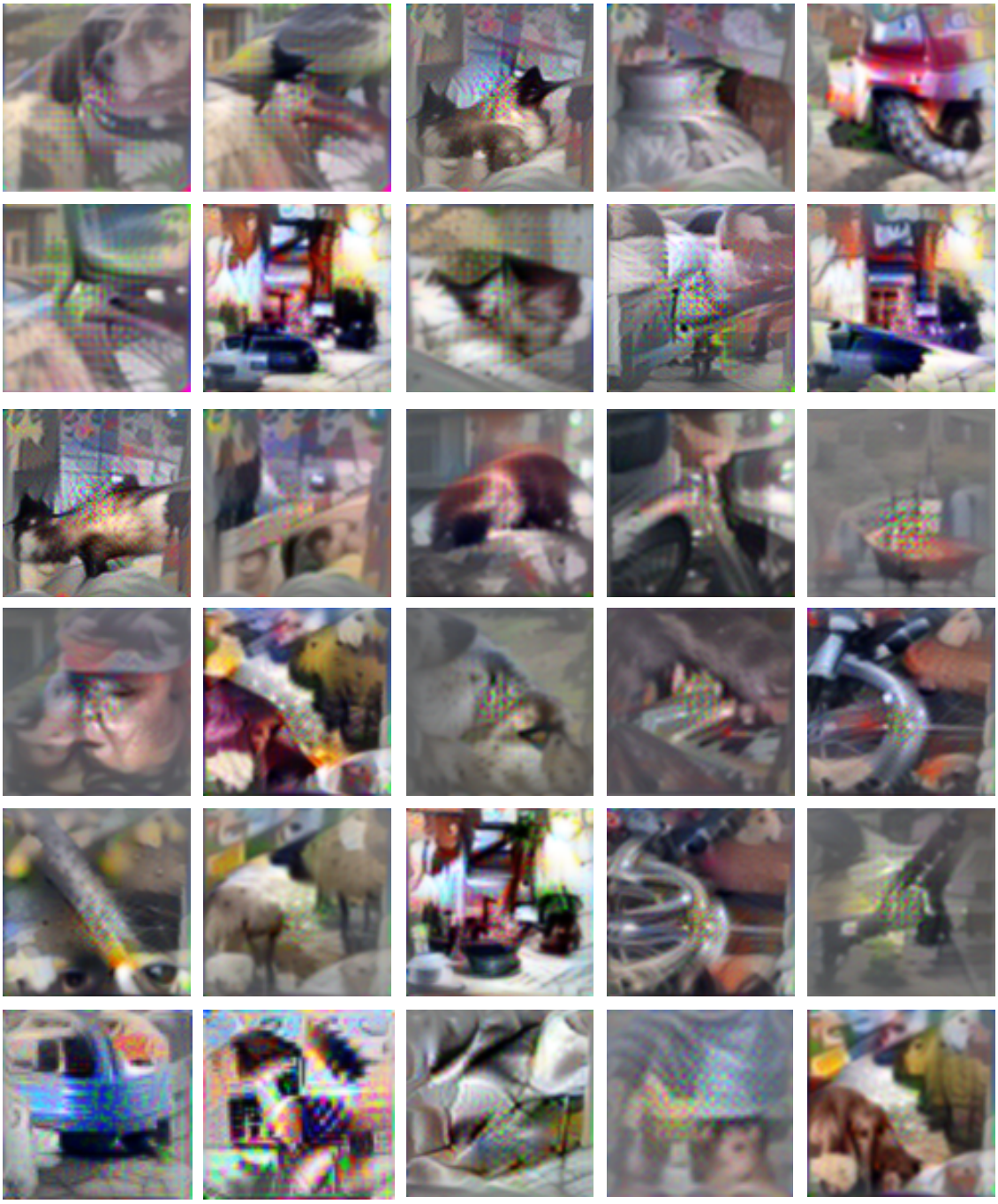


Figure 6.8: Images generated by the generator of our conditional GAN on the Pascal dataset. Interestingly, patterns about dogs, cars, plants and cats have been automatically discovered. This highlights the effectiveness of our approach, indeed, the generator identifies automatically visual clusters that are then employed by the discriminator as pixel-level annotated data.

Figure 6.9: Images generated by the generator during our GAN training on the SiftFlow dataset. Patterns about forests, beaches and slies can be observed.

Table 6.2: The results on VOC 2012 validation set using 30% of fully labeled data and all unlabeled data in training set.

method	pixel acc	mean acc	mean IU
Fully supervised	83.15	53.1	38.9
Semi supervised	83.6	60.0	42.2
Weak Supervised	84.6	58.6	44.6

Table 6.3: The results on SiftFlow using fully labeled data and 2000 unlabeled images from SUN2012

method	pixel acc	mean acc	mean IU
Fully supervised	83.4	46.7	34.4
Semi supervised	86.3	50.8	35.1

accuracy is not significantly improved due to some false positives, which correspond to background pixels promoted by unlabeled data belonging to one of the classes in the training set.

False positives are reduced in the weakly supervised framework, due to the fact that the unsupervised loss encourages only labels occurring in the image and assigns them high confidences. This effect can be observed in qualitative results in Figure 6.6. Thus, even though the semi-supervised method labels most of objects properly, it sometime assigns semantic classes to background pixels, while the weakly supervised method is able to reduce false positive detections. Furthermore, as shown in the same Tabel 6.1, our weakly approach also outperforms state of the art semi-supervised semantic segmentation methods, such as [111], adopting a similar strategy to our weakly-supervised one. Table 6.3 shows the results achieved by our approaches over the SiftFlow dataset [85]. Since in this dataset, background pixels are also labeled, the pixel accuracy is improved compared to the results obtained on PASCAL VOC 2012 dataset.

Since images with class level labels are not available in the SiftFlow dataset, we only test semi-supervised learning. Figure 6.7 shows qualitative results on the SiftFlow dataset. In this case, unlabeled data allows us to refine the classification that initially are labeled with incorrect classes. For instance, in the fifth row the pixels which are mistakenly labeled as car or river are corrected in the semi-supervised results. Moreover, some small objects, such as the person or windows in the last row of Figure 6.7, which are not detected before, can be labeled correctly by employing additional data.

Table 6.4: The results using different percentages of fully labeled data and all unlabeled data in train set.

method	pixel acc	mean acc	mean IU
VOC 20% Full	73.15	23.2	16.0
VOC 20% Semi	79.6	27.1	19.8
VOC 50% Full	88.5	63.6	51.6
VOC 50% Semi	88.4	66.6	54.0
SiftFlow 50% Full	79.0	28.3	21.0
SiftFlow 50% Semi	81.0	33.0	23.2

Table 6.5: The results on StanfordBG using fully labeled data and 10k unlabeled images from PASCAL dataset

method	pixel acc	mean acc	mean IU
Sem Seg Standard [89]	73.3	66.5	51.3
Sem Seg Adv [89]	75.2	68.7	54.3
Fully supervised	77.5	65.1	53.1
Semi supervised	82.3	77.6	63.3

We repeated the semi-supervised experiments with different training set sizes e.g. 20% and 50% of labeled data, and the results are presented in Table 6.4. This results suggest that the extra data, in conjunction with the way the loss is formulated, act as regularizer. Also, using more labeled data increases the overall performances, and the gap between the two settings is reduced.

For the third experiment, we evaluated our method on StanfordBG [48] data set. This is a small data set including 720 labeled images, therefore we use Pasacal images as unlabeled data, since these images collected from pascal or similar datasets. Table 6.5 shows our performance over the test images of StanfordBG data set compared to [89]. Note that our approach, again, outperforms significantly state of the art methods, e.g., [89], besides improving our fully-supervised method used as baseline.

Finally, we applied our proposed method to CamVid [11] dataset. This dataset consists of 10

Figure 6.10: Qualitative results on StanfordBG dataset, using unlabeled data results in more accurate semantic segmentation, unlikely classes in the image are removed in semi-supervised approach.

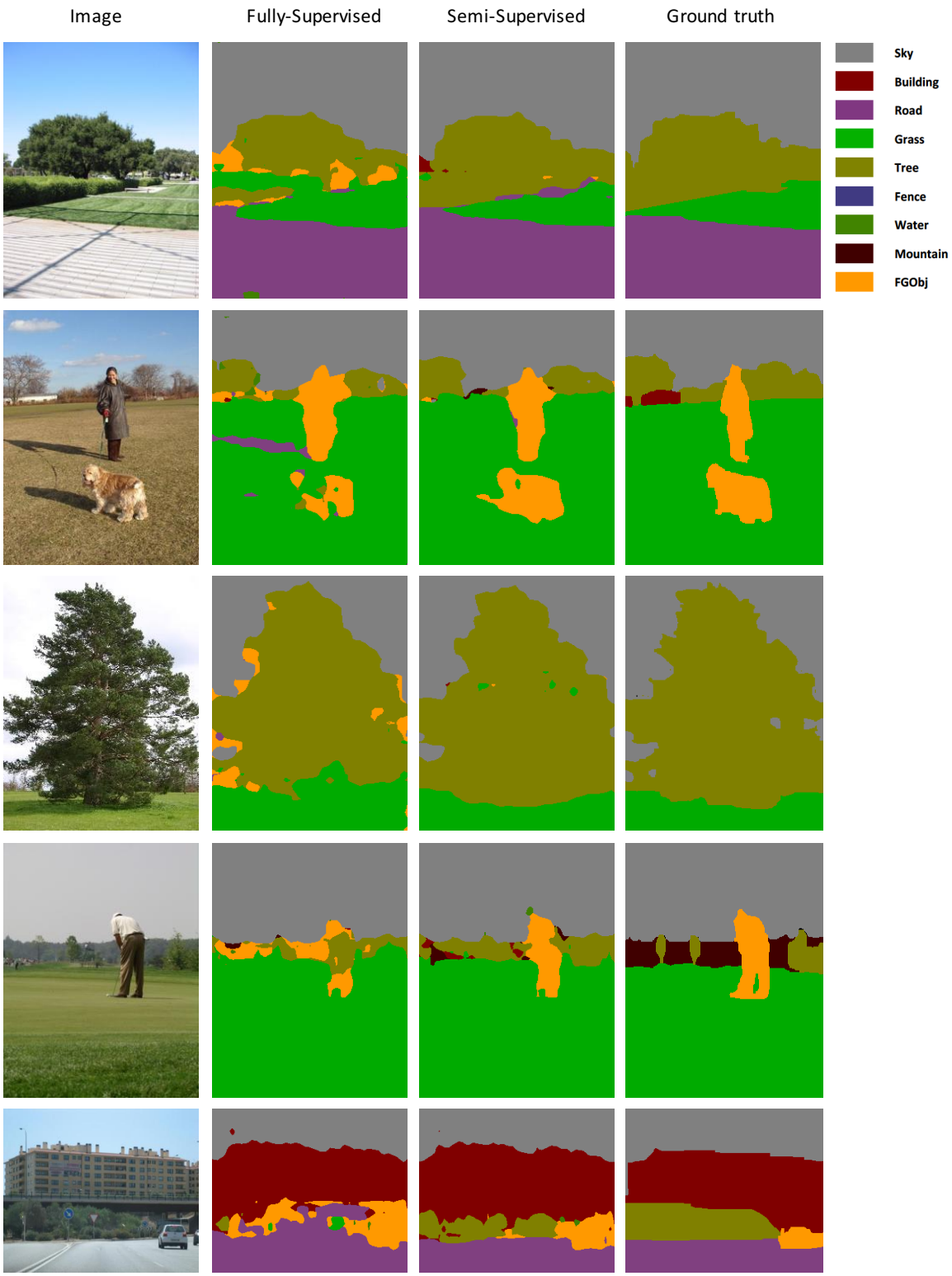


Figure 6.11: More qualitative results on StanfordBG dataset.

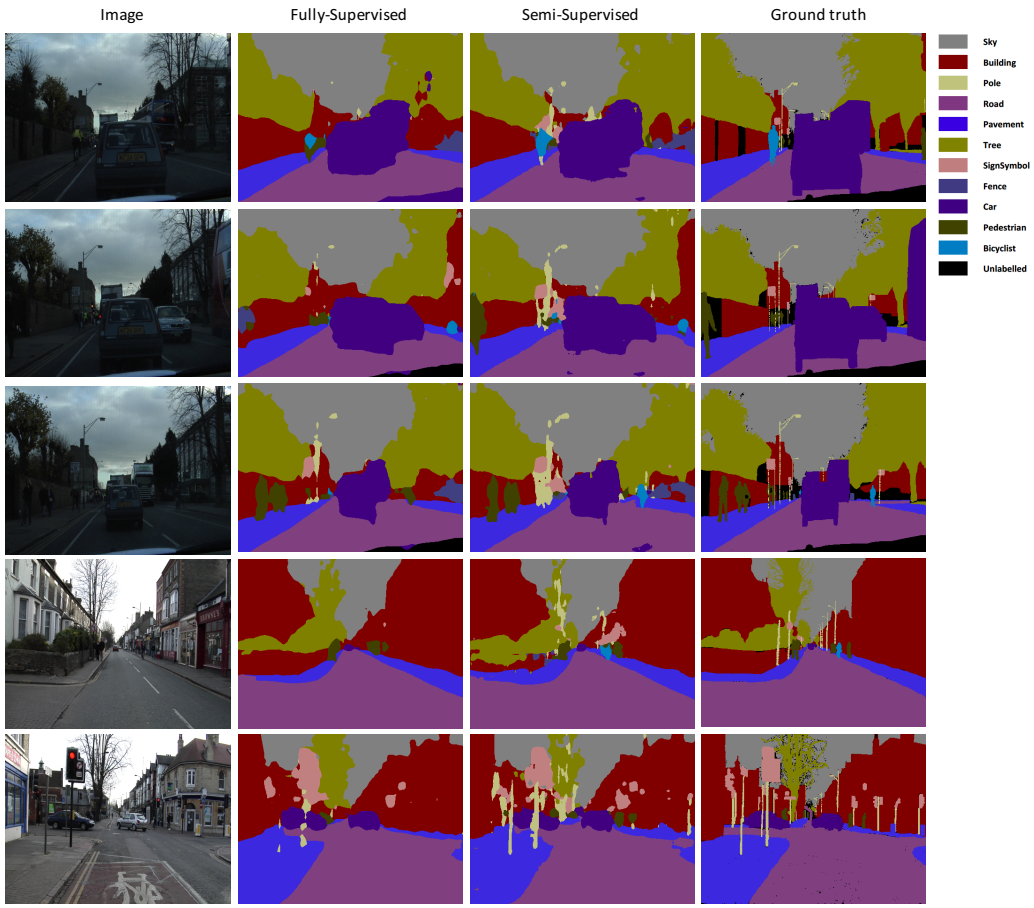


Figure 6.12: Samples of qualitative results from CamVid dataset. More classes are captured in semi-supervised learning approach.

minutes videos (about 11k frames), for 700 images of which per-pixel annotations are provided. We use the training set of fully-labeled (11 semantic classes) data and all frames as unlabeled data, and we perform the evaluation on the test set.

We compare our results to SegNet [5] method in addition to our baseline (i.e., the fully-supervised method). The results are reported in Tab. 6.6 and show that our semi-supervised method notably improves per-class accuracy, which indicates that classes, which are present in the images are identified correctly.

Samples of images generated by our GAN during training over the employed datasets are shown

Figure 6.13: Images generated by the generator for the CamVid dataset. Patterns about mountains, cars and building can be observed.

in Figures 6.9, 6.13 and 6.8. These images clearly indicate that our network is able to learn hidden structures (specific of each dataset), that are then used to enhance the performance of our GAN discriminator as they can be considered as additional pixel-level annotated data.

Moreover, interestingly, our GAN framework is also able to learn spatial object distributions, for example, roads are at the bottom of images, sky and mountains are at the top, etc. In Figures 6.10 and 6.11 examples from qualitative results for StanfordBG dataset are depicted; by using unlabeled data via our proposed approach some pixels, which fully-supervised method labeled incorrectly,

Table 6.6: The results on CamVid using fully labeled training data and 11k unlabeled frames from its videos.

method	pixel acc	mean acc	mean IU
Segnet-Basic [5]	82.2	62.3	46.3
SegNet (Pretrained) [5]	88.6	65.9	50.2
Ours Fully supervised	88.4	66.7	57.0
Ours Semi supervised	87.0	72.4	58.2

can be refined. For instance, in the second row parts from Cow which are mistakenly labeled as building or tree are corrected in the semi-supervised result.

Samples of qualitative results from CamVid dataset are shown in Figure 6.12. As before, some pixels are refined using unlabeled data. Moreover, some small objects, for example the pole, pedestrian or bicyclist in the figure 6.12, which are not detected can be labeled correctly by employing additional data.

In summary, the results achieved over different experiments indicate that the extra data provided through adversarial loss boosts the performance (outperforming both fully-supervised and state-of-the-art semi-supervised methods) of semantic segmentation, especially in terms of mean accuracy measure. The competitiveness of the discriminator and the generator yields not only in generating images, but, most importantly, to learn more meaningful features for pixel classification.

6.6 Summary

In this chapter, we developed a novel semi-supervised semantic segmentation approach employing Generative Adversarial Networks. We also investigated GANs conditioned by class-level labels, which are easier to obtain, to train our fully-convolutional network in a weakly supervised manner. We demonstrated that this approach outperforms fully-supervised methods trained with a limited amount of labeled data as well as state of the art semi-supervised methods over several benchmarking datasets.

Beside, our model generates plausible synthetic images, which show some meaningful image features such as edges and class labels, that supports the discriminator in the pixel-classification step. The discriminator can be replaced by any better classifier suitable for semantic segmentation for further improvements. We tested our model on VOC 2012, SiftFlow, StanfordBG and CamVid datasets and reported the improved results over the standard classifier results.

CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this dissertation, we address two fundamental problems of visual scene understanding: Visual Saliency Detection and Semantic Segmentation. In the first part, we discuss that salient regions in a video can be represented as sparse parts which are different from the rest of the video. We utilize super-voxel segmentation to obtain coherent segments of the video, then we emply this group information in group lasso regularization to acquire a feature matrix for each clip of a video, which is subsequently decomposed into sparse and low rank parts via applying Robust PCA method. In chapter 3, we experimentally demonstrate the effectiveness of our unsupervised saliency detection method to predict the areas of the video that captures human attention. In addition, we showed the application of the saliency detection in action recognition task in videos, in which discarding descriptors from non-salient part improves the accuracy.

In the second part, we address the problem of semantic segmentation in images. In computer vision many tasks have been introduced to help understand scenes; among them semantic segmentation provides a full pixel level labeling of the image. In this thesis, we present three different methods to address this problem. In the first and second method, the image is divided into super-pixels, and each superpixel is described by a set of features, such as color histogram, SIFT, etc., representing appearance and geometry of the superpixel. Then labeling is performed on each super-pixel descriptor using classifiers; we used Random Forest and XGBoost classifiers. We propose to incorporate context information to refine assignment of labels to each segment of the image. In order to tackle the limitation of neighboring smoothing approaches, in chapter 4 we use graphical lasso algorithm to find the interactions between labels as well as segments to refine classifier results. The method demonstrates that the graphical lasso is an efficient way to capture and incorporate

significant relationships between labels and segments.

In chapter 5, we use a knowledge based (rules) method which applies the rules as constraints in binary integer programming. In this approach, we are allowed to integrate different aspects of context information via an expressive method employing rules extracted from training images. In addition, we benefit from Places CNN deep network to include more context information through scene category- semantic label associations which are learned in training.

The traditional approaches to semantic labeling commonly focus on the fully-labeled data; however, with the neural network models which require a large amount of data to train, it is crucial that our models do not only use the limited fully annotated available data, but also be able to leverage from plentiful unlabeled or weakly labeled images. To this aim, in chapter 6 we propose a semi-supervised learning method using Generative Adversarial Network utilizing unlabeled data and weakly labeled data to improve the classifier (Discriminator) results. In doing so, we introduce a GAN framework in which Discriminator is a fully convolutional neural network whose task is to perform semantic segmentation by labeling each pixel, and the Generator produces synthetic images presenting the classes in database enforcing Discriminator learn robust features.

In order to support our ideas and verify the applicability and effectiveness of proposed techniques, we performed extensive experiments on different challenging data sets and reported the comparison of our approaches with several competitive baselines.

7.2 Future Work

There are some possible ways to improve the results and performance of the proposed approaches in future work; we highlight some of these extensions in this section.

Improving the super-pixel or super-voxel segmentation, in both saliency detection and semantic segmentation, can lead to better final performance in terms of time efficiency and accuracy. Since more accurate segmentation reduce the errors cause by overlapping classes in a segment or splitting incorrectly an object or part of objects to different segments. Moreover, using recent approaches to obtain feature vectors results in better overall performance. For instance, deep networks based features are acceptable alternatives to as known hand-crafted features in many computer vision tasks. Bag of Word representations can be replaced with convolution features leading to more robust and higher quality representation of visual features.

In chapter 5, for solving our optimization problem, we employ Gurobi which is a simplex based solver, thus we rely on LP relaxation of the solver to optimize the binary integer programming. One can leverage from other methods of relaxation or changing the quadratic inequalities into multiple linear inequalities in order to achive more adequate performance. Another extension for chapter 5 could be using the idea of the rule-based constraints in a weakly supervised manner, in which only image-level annotation is available. The constraints can guide the optimization problem and prune the search space.

Since recently most of the semantic segmentation methods exploit deep learning approaches, one future direction could be integrating the label interactions or rule-based constraints in the structure of the network. By using these ideas, context information can be more utilized in deep networks, considering that pooling layers discard the global context information. Even the receptive field of CNNs without pooling layers are limited, and can only grow linearly with the number of layers.

Finally, in the Generative Adversarial Network framework, we used a basic semantic segmentation network; however, this part can be easily replaced with a more recent and robust network, as long as the resources allow, to achive finer and more effective results.

LIST OF REFERENCES

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

[2] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2189–2202, 2012.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011.

[4] F. R. Bach. Consistency of the group lasso and multiple kernel learning. *The Journal of Machine Learning Research*, 9:1179–1225, 2008.

[5] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[7] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color-and texture-based image segmentation using em and its application to content-based image retrieval. In *Computer Vision, 1998. Sixth International Conference on*, pages 675–682. IEEE, 1998.

[8] A. Borji and L. Itti. State-of-the-art in visual attention modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[9] A. Borji, D. N. Sihite, and L. Itti. Computational modeling of top-down visual attention in interactive environments. In *British Machine Vision Conference*, pages 1–12, 2011.

[10] L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. *Computer Vision–ECCV 2010*, pages 168–181, 2010.

[11] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV (1)*, pages 44–57, 2008.

[12] N. Bruce and J. Tsotsos. Saliency based on information maximization. In *Advances in Neural Information Processing Systems*, pages 155–162, 2005.

[13] N. D. Bruce and J. K. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of Vision*, 9(3), 2009.

[14] M.-W. Chang, L. Ratinov, and D. Roth. Structured learning with constrained conditional models. *Machine learning*, 88(3):399–431, 2012.

[15] O. Chapelle, B. Scholkopf, and A. Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[16] C. Chen, J. A. Ozolek, W. Wang, and G. K. Rohde. A pixel classification system for segmenting biomedical images using intensity neighborhoods and dimension reduction. In *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pages 1649–1652. IEEE, 2011.

[17] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[18] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[19] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1124–1131. IEEE, 2005.

[20] G. Csurka and F. Perronnin. A simple high performance approach to semantic segmentation. In *BMVC*, pages 1–10, 2008.

[21] J. Dai, K. He, and J. Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1635–1643, 2015.

[22] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[23] M. Dorr, T. Martinetz, K. R. Gegenfurtner, and E. Barth. Variability of eye movements when viewing dynamic natural scenes. *Journal of Vision*, 10(10), 2010.

[24] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 766–774, 2014.

[25] D. Eigen and R. Fergus. Nonparametric image parsing using adaptive neighbor sets. In *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*, pages 2799–2806. IEEE, 2012.

[26] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15, 2006.

[27] I. Endres and D. Hoiem. Category-independent object proposals with diverse ranking. *IEEE transactions on pattern analysis and machine intelligence*, 36(2):222–234, 2014.

[28] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[29] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[30] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Scene parsing with multiscale feature learning, purity trees, and optimal covers. *arXiv preprint arXiv:1202.2160*, 2012.

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

[32] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Cascade object detection with deformable part models. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 2241–2248. IEEE, 2010.

[33] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[34] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[35] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[36] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

[37] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[38] S. Frintrop, E. Rome, and H. I. Christensen. Computational visual attention systems and their cognitive foundations: A survey. *ACM Transactions on Applied Perception (TAP)*, 7(1):6, 2010.

[39] F. G. Frdric J. A. M. Poirier and M. Arguin. Perceptive fields of saliency. *Journal of Vision*, 8, November 2008.

[40] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 113–120. IEEE, 2010.

[41] C. Galleguillos, A. Rabinovich, and S. Belongie. Object categorization using co-occurrence, location and appearance. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[42] D. Gao, S. Han, and N. Vasconcelos. Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):989–1005, 2009.

[43] D. Gao and N. Vasconcelos. Discriminant saliency for visual recognition from cluttered scenes. In *Advances in Neural Information Processing Systems*, pages 481–488, 2004.

[44] D. Gao and N. Vasconcelos. Decision-theoretic saliency: Computational principles, biological plausibility, and implications for neurophysiology and psychophysics. *Neural Computation*, 21(1):239–271, 2009.

[45] M. George. Image parsing with a wide range of classes and scene-level context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3622–3630, 2015.

[46] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):142–158, 2016.

[47] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

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

[49] S. Gould, T. Gao, and D. Koller. Region-based segmentation and object detection. In *Advances in neural information processing systems*, pages 655–663, 2009.

[50] S. Gould and Y. Zhang. Patchmatchgraph: Building a graph of dense patch correspondences for label transfer. In *Computer Vision–ECCV 2012*, pages 439–452. Springer, 2012.

[51] S. Gould, J. Zhao, X. He, and Y. Zhang. Superpixel graph label transfer with learned distance metric. In *Computer Vision–ECCV 2014*, pages 632–647. Springer, 2014.

[52] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2141–2148. IEEE, 2010.

[53] C. Guo, Q. Ma, and L. Zhang. Spatio-temporal saliency detection using phase spectrum of quaternion fourier transform. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[54] R. Guo and D. Hoiem. Labeling complete surfaces in scene understanding. *International Journal of Computer Vision*, pages 1–16, 2014.

[55] I. Gurobi Optimization. Gurobi optimizer reference manual, 2015.

[56] X. He, R. S. Zemel, and M. Á. Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *Computer vision and pattern recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE computer society conference on*, volume 2, pages II–II. IEEE, 2004.

[57] G. E. Hinton. Products of experts. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, pages 1–6. IET, 1999.

[58] S. Hong, H. Noh, and B. Han. Decoupled deep neural network for semi-supervised semantic segmentation. In *Advances in Neural Information Processing Systems*, pages 1495–1503, 2015.

[59] J. Honorio, D. Samaras, N. Paragios, R. Goldstein, and L. E. Ortiz. Sparse and locally constant gaussian graphical models. In *Advances in Neural Information Processing Systems*, pages 745–753, 2009.

[60] R. Hu, M. Rohrbach, and T. Darrell. Segmentation from natural language expressions. In *European Conference on Computer Vision*, pages 108–124. Springer, 2016.

[61] F. Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

[62] P. Isola and C. Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3048–3055. IEEE, 2013.

[63] L. Itti and P. Baldi. A principled approach to detecting surprising events in video. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[64] L. Itti and P. Baldi. Bayesian surprise attracts human attention. *Vision Research*, 49, 2009.

[65] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[66] S. D. Jain and K. Grauman. Supervoxel-consistent foreground propagation in video. In *European Conference on Computer Vision*, pages 656–671. Springer, 2014.

[67] M. Johnson, J. Shotton, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Decision Forests for Computer Vision and Medical Image Analysis*, pages 211–227. Springer, 2013.

[68] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *IEEE International Conference on Computer Vision*, pages 2106–2113, 2009.

[69] W. Kienzle, B. Schölkopf, F. A. Wichmann, and M. O. Franz. How to find interesting locations in video: a spatiotemporal interest point detector learned from human eye movements. In *Pattern Recognition*, pages 405–414. Springer, 2007.

[70] W. Kienzle, F. Wichmann, B. Schölkopf, and M. Franz. A nonparametric approach to bottom-up visual saliency. In *Advances in Neural Information Processing Systems*, 2007.

[71] C. Kim and J.-N. Hwang. Fast and automatic video object segmentation and tracking for content-based applications. *IEEE transactions on circuits and systems for video technology*, 12(2):122–129, 2002.

[72] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[73] K. Koch, J. McLean, R. Segev, M. A. Freed, M. J. Berry II, V. Balasubramanian, and P. Sterling. How much the eye tells the brain. *Current Biology, PubMed Central (PMC)*, 16(14):1428–1434, 2006.

[74] V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural Inf. Process. Syst*, 2011.

[75] J. Konrad and M. Ristivojevic. Video segmentation and occlusion detection over multiple frames.

[76] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Graph cut based inference with co-occurrence statistics. In *Computer Vision–ECCV 2010*, pages 239–253. Springer, 2010.

[77] L. Ladickỳ, C. Russell, P. Kohli, and P. H. Torr. Inference methods for crfs with co-occurrence statistics. *International journal of computer vision*, 103(2):213–225, 2013.

[78] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and crfs. In *Computer Vision–ECCV 2010*, pages 424–437. Springer, 2010.

[79] J. Lafferty, A. McCallum, F. Pereira, et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, volume 1, pages 282–289, 2001.

[80] T. Lan, Y. Wang, and G. Mori. Discriminative figure-centric models for joint action localization and recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2003–2010. IEEE, 2011.

[81] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[82] D. Larlus and F. Jurie. Combining appearance models and markov random fields for category level object segmentation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE, 2008.

[83] A. Liaw and M. Wiener. Classification and regression by randomforest.

[84] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

[85] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011.

[86] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.

[87] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):353–367, 2011.

[88] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[89] P. Luc, C. Couprie, S. Chintala, and J. Verbeek. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*, 2016.

[90] S. Ma, J. Zhang, N. Ikizler-Cinbis, and S. Sclaroff. Action recognition and localization by hierarchical space-time segments. In *Proceedings of the IEEE international conference on computer vision*, pages 2744–2751, 2013.

[91] V. Mahadevan and N. Vasconcelos. Spatiotemporal saliency in dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):171–177, 2010.

[92] J. Mairal. Spams: a sparse modeling software [online], available: http://spams-devel.gforge.inria.fr. 2012.

[93] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. *arXiv preprint arXiv:1411.3230*, 2014.

[94] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

[95] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferreras. Road-sign detection and recognition based on support vector machines. *Trans. Intell. Transport. Sys.*, 8(2):264–278, June 2007.

[96] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International journal of computer vision*, 43(1):7–27, 2001.

[97] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 89–96. IEEE, 2011.

[98] S. Mallat. *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[99] S. Marat, T. H. Phuoc, L. Granjon, N. Guyader, D. Pellerin, and A. Guérin-Dugué. Modelling spatio-temporal saliency to predict gaze direction for short videos. *International Journal of Computer Vision*, 82(3):231–243, 2009.

[100] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936. IEEE, 2009.

[101] S. Mathe and C. Sminchisescu. Actions in the eye: Dynamic gaze datasets and learnt saliency models for visual recognition. Technical report, Institute of Mathematics of the Romanian Academy and University of Bonn, February 2012.

[102] S. Mathe and C. Sminchisescu. Dynamic eye movement datasets and learnt saliency models for visual action recognition. In *IEEE European Conference on Computer Vision*, 2012.

[103] L. Meier, S. Van De Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

[104] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid. Joint tracking and segmentation of multiple targets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5397–5406, 2015.

[105] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[106] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3376–3385, 2015.

[107] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36, 2014.

[108] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

[109] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

[110] B. A. Olshausen and D. J. Field. Sparse coding of sensory inputs. *Current opinion in neurobiology*, 14(4):481–487, 2004.

[111] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. *arXiv preprint arXiv:1502.02734*, 2015.

[112] C. C. Parma, A. R. Hanson, and E. M. Riseman. *Experiments in schema-driven interpretation of a natural scene*. Springer, 1981.

[113] D. Pathak, E. Shelhamer, J. Long, and T. Darrell. Fully convolutional multi-class multiple instance learning. *arXiv preprint arXiv:1412.7144*, 2014.

[114] Z. Qin, K. Scheinberg, and D. Goldfarb. Efficient block-coordinate descent algorithms for the group lasso. *Mathematical Programming Computation*, pages 1–27, 2010.

[115] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[116] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

[117] R. A. Rensink, J. K. O'Regan, and J. J. Clark. To see or not to see: The need for attention to perceive changes in scenes. *Psychological Science*, 8(5):368–373, 1997.

[118] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2008.

[119] V. Roth and B. Fischer. The group-lasso for generalized linear models: uniqueness of solutions and efficient algorithms. In *International Conference on Machine Learning*, volume 104, 2008.

[120] R. Rubinstein, A. M. Bruckstein, and M. Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[121] R. Rubinstein, M. Zibulevsky, and M. Elad. Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on Signal Processing*, 58(3):1553–1564, 2010.

[122] D. Rudoy, D. B. Goldman, E. Shechtman, and L. Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1147–1154. IEEE, 2013.

[123] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[124] C. Russell, P. H. Torr, and P. Kohli. Associative hierarchical crfs for object class image segmentation. In *in Proc. ICCV*. Citeseer, 2009.

[125] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

[126] F. Schroff, A. Criminisi, and A. Zisserman. Object class segmentation using random forests. In *BMVC*, pages 1–10, 2008.

[127] A. G. Schwing and R. Urtasun. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.

[128] H. J. Seo and P. Milanfar. Nonparametric bottom-up saliency detection by self-resemblance. In *Computer Vision and Pattern Recognition Workshops*, pages 45–52, 2009.

[129] H. J. Seo and P. Milanfar. Static and space-time visual saliency detection by self-resemblance. *Journal of vision*, 9(12), 2009.

[130] E. Sharon, M. Galun, D. Sharon, R. Basri, and A. Brandt. Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):810–813, 2006.

[131] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[132] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Computer vision and pattern recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

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

[134] B. Shuai, G. Wang, Z. Zuo, B. Wang, and L. Zhao. Integrating parametric and non-parametric models for scene labeling.

[135] G. Singh and J. Kosecka. Nonparametric scene parsing with adaptive feature relevance and semantic context. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3151–3157. IEEE, 2013.

[136] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

[137] N. Souly and M. Shah. Scene labeling through knowledge-based rules employing constrained integer linear programing. *arXiv preprint arXiv:1608.05104*, 2016.

[138] N. Souly and M. Shah. Scene labeling using sparse precision matrix. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[139] N. Souly, C. Spampinato, and M. Shah. Semi and weakly supervised semantic segmentation using generative adversarial network. *arXiv preprint arXiv:1703.09695*, 2017.

[140] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[141] V. Srikumar. Soft constraints in integer linear programs. 2013.

[142] I. Sutskever, R. Jozefowicz, K. Gregor, D. Rezende, T. Lillicrap, and O. Vinyals. Towards principled unsupervised learning. *arXiv preprint arXiv:1511.06440*, 2015.

[143] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *nips*, volume 14, 2001.

[144] M. Thoma. A survey of semantic segmentation. *arXiv preprint arXiv:1602.06541*, 2016.

[145] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58, 1996.

[146] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[147] J. Tighe and S. Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *Computer Vision–ECCV 2010*, pages 352–365. Springer, 2010.

[148] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3001–3008. IEEE, 2013.

[149] J. Tighe, M. Niethammer, and S. Lazebnik. Scene parsing with object instances and occlusion ordering. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3748–3755. IEEE, 2014.

[150] S. Tokui, K. Oono, S. Hido, and J. Clayton. Chainer: a next-generation open source framework for deep learning. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[151] A. M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

[152] J. Triesch, D. H. Ballard, M. M. Hayhoe, and B. T. Sullivan. What you see is what you need. *Journal of Vision*, 3(1), 2003.

[153] F. Tung and J. J. Little. Collageparsing: Nonparametric scene parsing by adaptive overlapping windows. In *Computer Vision–ECCV 2014*, pages 511–525. Springer, 2014.

[154] S. K. Ungerleider and L. G. Mechanisms of visual attention in the human cortex. *Annual review of neuroscience*, 23(1):315–341, 2000.

[155] H. Valpola. From neural pca to deep unsupervised learning. *Advances in Independent Component Analysis and Learning Machines*, pages 143–171, 2015.

[156] E. Vig, M. Dorr, T. Martinetz, and E. Barth. Eye movements show optimal average anticipation with natural dynamic scenes. *Cognitive Computation*, 3(1):79–88, 2011.

[157] E. Vig, M. Dorr, T. Martinetz, and E. Barth. Intrinsic dimensionality predicts the saliency of natural dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1080–1091, 2012.

[158] T. L. Vu, S.-W. Choi, and C. H. Lee. Improving accuracy for image parsing using spatial context and mutual information. In *Neural Information Processing*, pages 176–183. Springer, 2013.

[159] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[160] M. Wertheimer. Laws of organization in perceptual forms (partia1 translation). *A Sourcebook of Gestalt Psycychology*, pages 71–88.

[161] J. Weston, F. Ratle, H. Mobahi, and R. Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

[162] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in Neural Information Processing Systems*, pages 2080–2088, 2009.

[163] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE, 2010.

[164] C. Xu and J. J. Corso. Evaluation of super-voxel methods for early video processing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1202–1209, 2012.

[165] J. Yan, M. Zhu, H. Liu, and Y. Liu. Visual saliency detection via sparsity pursuit. *IEEE Signal Processing Letters*, 17(8):739–742, 2010.

[166] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2):212–225, 2008.

[167] Y. Yang, S. Hallman, D. Ramanan, and C. C. Fowlkes. Layered object models for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1731–1743, 2012.

[168] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68, 2006.

[169] M. Yuan and Y. Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

[170] Y. Zhai and M. Shah. Visual attention detection in video sequences using spatiotemporal cues. In *ACM international conference on Multimedia*, pages 815–824, 2006.

[171] L. Zhang, M. H. Tong, and G. W. Cottrell. Sunday: Saliency using natural statistics for dynamic analysis of scenes. In *Annual Cognitive Science Conference*, pages 2944–2949, 2009.

[172] L. Zhang, M. H. Tong, T. K. Marks, H. Shan, and G. W. Cottrell. Sun: A bayesian framework for saliency using natural statistics. *Journal of Vision*, 8(7), 2008.

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

[174] S.-h. Zhong, Y. Liu, F. Ren, J. Zhang, and T. Ren. Video saliency detection via dynamic consistent spatio-temporal attention modelling. In *AAAI*, pages 1063–1069, 2013.

[175] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

[176] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *arXiv preprint arXiv:1608.05442*, 2016.

[177] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.