MULTILEVEL CLUSTER ENSEMBLING FOR HISTOPATHOLOGICAL IMAGE SEGMENTATION

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Ahmet Çağrı Şimşek

August, 2011

ABSTRACT

MULTILEVEL CLUSTER ENSEMBLING FOR HISTOPATHOLOGICAL IMAGE SEGMENTATION

Ahmet Çağrı Şimşek
M.S. in Computer Engineering
Supervisors: Prof. Dr. Cevdet Aykanat and
Assist. Prof. Dr. Çiğdem Gündüz Demir
August, 2011

In cancer diagnosis and grading, histopathological examination of tissues by pathologists is accepted as the gold standard. However, this procedure has observer variability and leads to subjectivity in diagnosis. In order to overcome such problems, computational methods which use quantitative measures are proposed. These methods extract mathematical features from tissue images assuming they are composed of homogeneous regions and classify images. This assumption is not always true and segmentation of images before classification is necessary. There are methods to segment images but most of them are proposed for generic images and work on the pixel-level. Recently few algorithms incorporated medical background knowledge into segmentation. Their high level feature definitions are very promising. However, in the segmentation step, they use region growing approaches which are not very stable and may lead to local optima.

In this thesis, we present an efficient and stable method for the segmentation of histopathological images which produces high quality results. We use existing high level feature definitions to segment tissue images. Our segmentation method significantly improves the segmentation accuracy and stability, compared to existing methods which use the same feature definition. We tackle image segmentation problem as a clustering problem. To improve the quality and the stability of the clustering results, we combine different clustering solutions. This approach is also known as cluster ensembles. We formulate the clustering problem as a graph partitioning problem. In order to obtain diverse and high quality clustering results quickly, we made modifications and improvements on the well-known multilevel graph partitioning scheme. Our method clusters medically meaningful components in tissue images into regions and obtains the final segmentation.

Experiments showed that our multilevel cluster ensembling approach performed significantly better than existing segmentation algorithms used for generic and tissue images. Although most of the images used in experiments, contain noise and artifacts, the proposed algorithm produced high quality results.

Keywords: Histopathological image segmentation, cluster ensembles, multilevel graph partitioning, unsupervised segmentation.

ÖZET

HİSTOPATOLOJİK GÖRÜNTÜ BÖLÜTLEMESİ İÇİN ÇOK SEVİYELİ KÜMELEME BİLEŞİMİ

Ahmet Çağrı Şimşek
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticileri: Prof. Dr. Cevdet Aykanat ve
Yrd. Doç. Dr. Çiğdem Gündüz Demir
Ağustos, 2011

Dokuların patologlar tarafından histopatolojik incelemesinin yapılması, kanser tanı ve derecelendirmesinde altın standart olarak kabul edilir. Bu işlemde gözlemcilerin değişkenlik göstermesi, tanı sonuçlarında öznelliğe sebep olur. Bu tarz sorunların üstesinden gelebilmek için, nicel veriler kullanan hesaplamasal teknikler ileri sürülmüştür. Bu teknikler, doku resimlerinin homojen bölgelerden oluştuğunu varsayarak bu resimlerden matematiksel özellikler çıkarır ve resimleri sınıflandırır. Fakat bu varsayım her zaman doğru değildir ve sınıflandırmadan önce resimlerin bölütlenmesi gerekir. Resimleri bölütlemek için çeşitli teknikler ileri sürülmüştür, fakat bu tekniklerin çoğu imgecikler üzerinde çalışır ve genel resimler için geliştirilmiştir. Son zamanlarda birkaç algoritma doku resimleri bölütlemede tıbbi bilgileri kullanmıştır. Bu tekniklerin yüksek seviye özellik tanımları çok ümit vericidir. Ancak, bu teknikler bölütleme safhalarında, çok kararlı olmayan ve yerel çözümlere kaçabilen bölge büyütme yaklaşımını kullanmıştır.

Bu tezde, histopatolojik resimlerin bölütlenmesi için yüksek kalite sonuçlar üreten, verimli ve kararlı bir yöntem sunuyoruz. Doku resimlerini bölütlemek için var olan yüksek seviye özellik tanımlarını kullandık. Bölütleme yöntemimiz, bizimle aynı özellik tanımını kullanan diğer yöntemlerin bölütleme başarısını ve kararlılığını önemli derecede arttırıyor. Resim bölütleme problemini bir kümeleme problemi olarak kabul ettik. Kümeleme sonuçlarının kalitesini ve kararlılığını arttırmak için farklı kümeleme sonuçlarını bir araya getirip birleştirdik. Bu teknik, kümeleme bileşimi olarak da bilinir. Biz ayrıca kümeleme problemini çizge bölümleme problemine dönüştürdük. Birbirinden farklı ve yüksek kaliteli kümeleme sonuçları elde etmek için, iyi bilinen çok seviyeli çizge bölümleme tekniği üzerinde değişiklikler ve iyileştirmeler yaptık. Yöntemimiz tıbbi olarak bir anlamı olan nesneleri ayrı bölgelere toplayarak sonuç bölütlemeyi elde eder.

Yaptığımız deneyler, önerdiğimiz çok seviyeli kümeleme bileşimi tekniğinin, genel resimler ve doku resimleri için daha önceden önerilmiş bölütleme tekniklerinden çok daha iyi sonuçlar ürettiğini gösterdi. Deneylerde kullandığımız doku resimlerinin çoğu resim elde etme aşamasında ortaya çıkan bozulmalar içermesine rağmen, önerdiğimiz yöntem yüksek kaliteli sonuçlar üretti.

Anahtar sözcükler: Histopatolojik görüntü bölütleme, kümeleme bileşimi, çok seviyeli çizge bölümleme, güdümsüz bölütleme .

Acknowledgement

I am grateful to my supervisors Prof. Dr. Cevdet Aykanat and Assist. Prof. Dr. Çiğdem Gündüz Demir for their guidance and support. I thank Prof. Dr. Fatoş Tünay Yarman-Vural and Assoc. Prof. Dr. Hakan Ferhatosmanoğlu for accepting to participate in my thesis committee.

I am also grateful to my colleagues Akif Burak Tosun, Erdem Özdemir, Ata Türk, Tayfun Küçükyılmaz, Enver Kayaaslan, Reha Oğuz Selvitopi, Kadir Akbudak, and Fahreddin Şükrü Torun for their contributions and comments.

I present my thanks to Bilkent University and TÜBİTAK for supporting me financially and physically.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Cancer is a type of disease described by uncontrolled growth of abnormal cells. Damaged and abnormal cells reproduce uncontrollably and create masses of a tissue called tumors. Tumors can grow, distort, and change the cellular and organizational structure of tissues from which they originate [1, 34, 45]. Metastasis occurs when a tumor spreads to different parts of the body, grows, invades, and destroys other healthy tissues. The result of metastasis is a serious condition which is very difficult to treat.

There are many types of cancer depending on the tissue it originates. Colon cancer is one of the most common cancers in the world [60]. Huge portion of cancer deaths are caused by colon cancer in the western world and in the countries which adapted western diets. Although colon cancer has a high prevalence, the survival rate is high if it is diagnosed early and treated correctly. For the diagnosis of colon cancer, there are various types of screening tests such as digital rectal exam, MRI, endoscopy, and colonoscopy. These tests mainly look for symptoms and polyps. If they locate polyps or find any other indicative symptoms, histopathological examination should be conducted to confirm the cancer and its grade [31, 54]. In this examination, a small part of a tissue is extracted from a patient by surgery and examined under a microscope by pathologists.

Early detection and correct grading of cancer affect the success of the selected treatment method and increase the chance of survival [2]. Hence, using procedures that provide reliable information is very crucial. Histopathological examination is the most reliable procedure and considered as the gold standard for diagnosis and grading. In this examination, pathologists should be able to identify the changes in cellular structures and the deformations in tissue distribution. This relies on visual interpretation of a tissue, and hence, is affected by the experience and expertise of pathologists [20, 77]. Moreover, tissue preparation procedures such as staining and sectioning operations may introduce noise and artifacts to the image, which makes the image hard to interpret [35]. Therefore, histopathological examination is subject to a considerable amount of intra- and inter-observer variability [43, 46, 12, 15]. In order to reduce the effect of observer variability, it is very important to standardize diagnosis and grading processes based on quantitative measures. One of the most reliable ways of doing this is to develop computational methods and build tools and programs.

1.1 Motivation

There are plenty of computational studies developed for histopathological image analysis. Most of these studies quantify histopathological images extracting their mathematical features and classify them based on the extracted features. These features include textural [21, 25, 26, 44, 66, 87], morphological [72, 73, 88], and structural [4, 86, 75, 22, 37, 38, 6] descriptors of the images. Although these descriptors are more or less successful to quantify homogeneous tissue images, they may fail to characterize heterogeneous tissue images, which consist of different homogeneous regions. As shown in Figure 1.1, colon tissue images may contain such regions that show very different characteristics in shape, color, and texture. Heterogeneity affects the representation power of the feature descriptors, and thus, the performance of classifiers. Segmenting heterogeneous tissue images into their homogeneous regions and then extracting the descriptors of these regions greatly improve the classification performance.

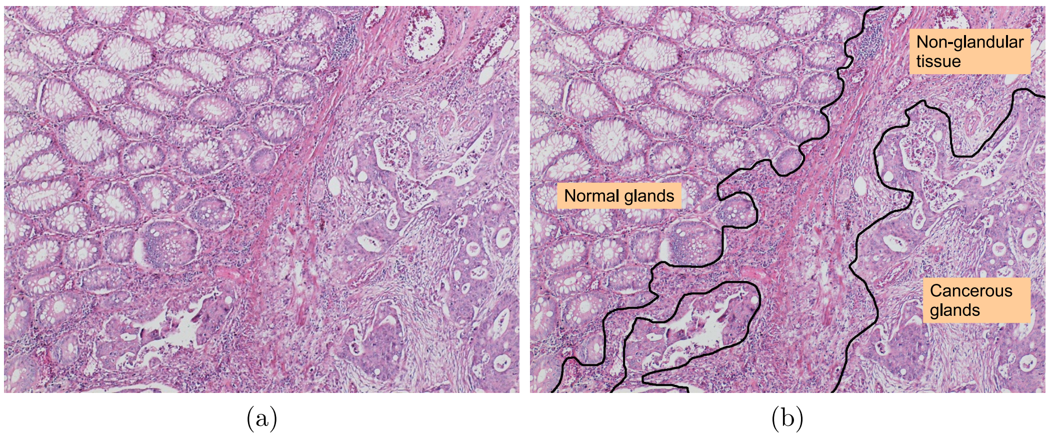


Figure 1.1: A colon tissue may consist of different types of regions: (a) a colon tissue image and (b) its manual segmentation.

In histopathological images, regions are characterized with the organization of their components. A colon tissue is composed of glands. In a normal tissue, these glands follow a regular structure and colon cancer causes deformations in these structures. In addition to the regions containing normal and cancerous glands, there may also exist regions that do not contain any glandular structure. These types of regions are shown in Figure 1.2. In literature, there are many techniques for unsupervised segmentation of generic images. However, there are only few studies that have been proposed for histopathological image segmentation [71, 85]. These studies segment images dividing them into grids and classifying each grid based on its feature descriptors. These features are commonly extracted making use of pixel-level information, without considering the domain specific knowledge.

Recently Tosun *et al.* [82, 81] introduced new sets of high level feature descriptors, which take medically meaningful objects into account. For that, they identify the approximate locations of cytological components in a tissue and define the texture descriptors on these components instead of defining them on pixel values. Using these new descriptors, they achieve segmentation by a region growing algorithm. These studies [82, 81] aim to improve the segmentation performance by mainly focusing on the feature extraction part. On the other hand, there is a room of improvement in its segmentation part. Indeed, like all of its kinds, a region growing algorithm has a risk of obtaining a local optimal solution,

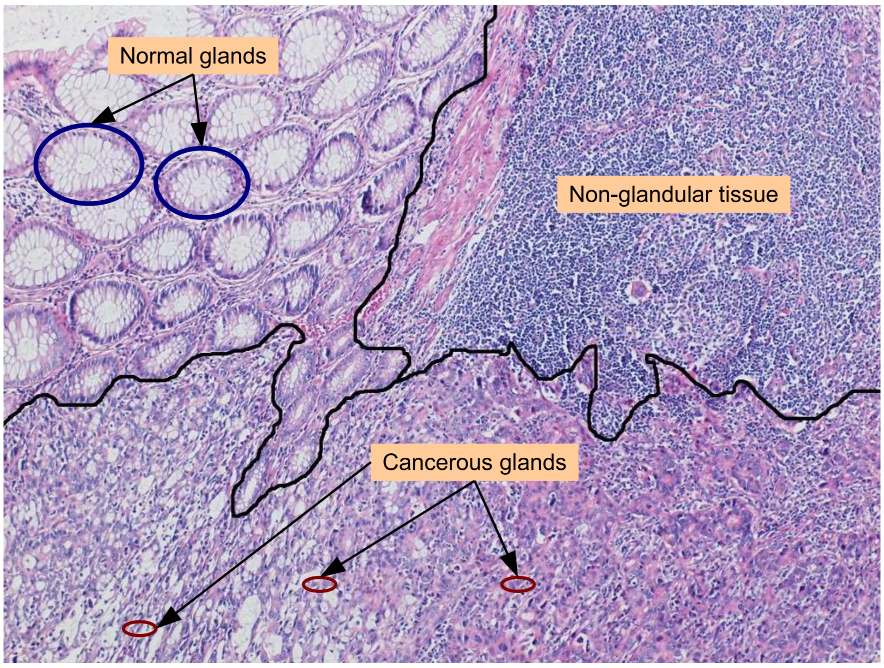


Figure 1.2: Heterogeneous histopathological images are composed of different regions.

especially when initial regions (seeds) are not carefully selected [90]. In this thesis, we focus on improving the segmentation part. To this end, we propose a new algorithm that ensembles multiple segmentations. The experiments indicate that the proposed method is more effective to reduce the negative effects of finding local optimal solutions.

It is known for many years now that ensembles of different classifiers perform better than a single classifier. Each classifier recognizes a different aspect of data and combining those multiple different points of views yields better accuracy. More recently, ensembling algorithms have been started to be used for clustering as well [74]. Single runs of clusterings may not be stable and may not yield accurate results for algorithms that require initialization points or that are randomized. Moreover, there may be cases, in which algorithms that can capture global optimum do not perform well under the conditions of noise and insufficient data representation. In such cases, the use of cluster ensembling has a potential to improve the results.

In ensembling, two most important factors that affect the final clustering result are the diversity and quality of individual clusterings [28, 52, 53]. In order to get an accurate and stable final clustering, each clusterer should yield an at least slightly different result to increase the diversity. There are several techniques to introduce diversity to clusterers. For that, it is possible to use a randomized version of the same algorithm. Alternatively, random subsets of data points or random subsets of features can be used to obtain each clusterer.

The quality of each clustering result should also be acceptably high. To this end, effective clustering algorithms should be used. Spectral methods are the examples of such algorithms [70, 57, 89]. In these methods, data points are considered as graph vertices and similarity of these points correspond to weights of the graph edges defined between these data objects. The objective is to divide the graph into a predefined number of parts by minimizing the sum of the weights of the cut edges. This directly corresponds to the goal of the clustering problem where "inter-cluster distance is maximized and intra-cluster distance is minimized". These spectral methods perform considerably well in finding global optimum. However, as these algorithms need eigen decomposition, they are very demanding in both CPU time and memory space. Moreover cluster ensembling requires running these algorithms multiple times, which further increases the computational costs. To overcome the computational burdens of spectral methods, Dhillon *et al.* propose to use multilevel graph partitioning. They show that similar results can be obtained with multilevel graph partitioning by breaking the balance criterion [18, 47]. However their proposed algorithm is not a good choice for cluster ensembling since it usually yields lower diversity.

1.2 Contribution

In this thesis, our main contribution is as follows: We present a new multilevel cluster ensembling algorithm to be used in histopathological image segmentation. The proposed method is efficient and stable. It yields accurate segmentations avoiding local optimum and over-segmentation. It achieves this by producing diverse and high quality clusterings and combining them efficiently. In this algorithm, we work on the objects described in [81] and use a set of features defined on these objects. After defining the objects and their features, the algorithm considers each object as a data point in a clustering problem and clusters the objects into the desired number of clusters. It runs a predefined number of iterations to get different clustering results and combines them into a final clustering solution. In each iteration, it takes a random subset of objects and clusters them using a multilevel graph clustering algorithm whose refinement phase is redesigned to obtain diverse and high quality clustering solutions. The algorithm combines these clustering solutions with a consensus function. The consensus function constructs a bipartite graph, making clusters and objects two groups of vertices and connecting each cluster to the objects it contains by a unit weight edge [29]. This bipartite graph actually holds the objects' frequency of being together in the same cluster. After that, it partitions the bipartite graph to get the final clustering solution. Our experiments showed that the proposed multilevel cluster ensembling provides an effective image segmentation tool for histopathological tissue images and significantly increases segmentation accuracies of the previous approaches.

1.3 Outline of Thesis

The outline of this thesis is as follows. Chapter 2 summarizes previous computational methods for generic and histopathological image segmentation. Chapter 3 provides detailed description of the proposed multilevel cluster ensembling algorithm. Chapter 4 gives the experimental setup and reports the segmentation results of the proposed algorithm. It also gives the comparison of the proposed algorithm with other algorithms. Chapter 5 includes concluding remarks and discussions.

Chapter 2

Background

This chapter presents previous studies on image segmentation and its applications to histopathological images. In the first section, we explain previous studies on generic and histopathological image segmentation. In the second one, we explain multilevel framework. In the last section, we explain clustering and cluster ensembling.

2.1 Image Segmentation

Image segmentation is described as the operation of dividing an image into non-overlapping and connected pixel groups or regions that are semantically coherent in a particular context. Image segmentation aims to transform the representation of images and make them more meaningful and easier to analyze [68]. Pixels sharing certain visual characteristics are classified into regions. Pixels in each region are similar in an attribute or a computed feature, like intensity, color, and texture. Bordering regions are quite different according to the same characteristics [68]. Image segmentation is a well studied subject in computer vision and attracted a great deal of attention by many researchers. As a result, there are lots of different algorithms and approaches for image segmentation. These can be grouped into four: pixel-based methods, graph-based methods, region-based methods, and statistical methods. In the following subsections, we briefly mention these methods.

2.1.1 Pixel-based Methods

Pixel-based methods consider pixels as the smallest informative part of the image and groups the pixels according to their intensity or color values using different techniques.

2.1.1.1 Thresholding

Thresholding is the oldest and simplest segmentation method. It classifies pixels as foreground and background according to their intensities. For example, for detecting lighter foreground objects in darker background, a pixel is labeled as foreground object if its pixel value is greater than some threshold and as background otherwise. Thresholding methods are generally the most efficient methods in terms of computational requirements. Otsu [58] has a seminal work, in which he proposed a statistical threshold determination method for grayscale images. Thresholding can also be used for color images. A separate threshold can be defined for every RGB component and then the thresholding result of each component can be combined with an AND operation. The HSL, HSV, and CMYK color models can also be used [61].

2.1.1.2 Edge Detection

Another method that works on pixel values is edge detection. Here segmentation is achieved by finding the region boundaries with an edge detection algorithm. In order to locate edges, changes in gray-level pixel values can be detected using first order derivative operators like Sobel and Prewitt and second order derivative operators like Laplacian [36]. As these operators are more sensitive to noise, it is also possible to use operators such as Laplacian of Gaussian and difference of Gaussians. For example, the Laplacian of Gaussian operator first smooths an image with a Gaussian filter to reduce noise and then applies the Laplacian operator [55]. Another way is to use the Canny edge detector [10], which is a multi stage algorithm for edge detection.

2.1.1.3 Clustering Methods

Image segmentation is very similar to clustering in a sense that both try to group similar pixels or data points into groups according to a distance criterion. The simplest and the well known clustering technique is the k-means algorithm [32]. K-means first initializes the centroids of k clusters, randomly or using a heuristic. It then updates these centroids iteratively, until there is no significant change in the centroids. In each iteration, every data point is assigned to its nearest centroid and the new centroids are computed by averaging the data points that are assigned to those centroids. The k-means algorithm does not guarantee global optimum but good centroid initializations may lead to good results. An image can be segmented into regions by extracting intensity, color, and texture descriptors for each of its pixels and using the k-means algorithm [14, 9].

Fuzzy c-means [24] is another clustering algorithm used for image segmentation [65, 13]. The fuzzy c-means algorithm introduces fuzziness to memberships of data points. Each data point belongs to every cluster with a weight coefficient, which gives the degree of that object being in a cluster. The fuzzy c-means algorithm is a slightly modified version of the k-means algorithm with fuzziness. Just like in k-means, initial c cluster centroids are selected and then updated iteratively. In each iteration c weight coefficients are computed for each point. The coefficients are defined using a function of the distance between the point and the corresponding centroid. Once coefficients are found, cluster centroids are updated averaging the data points according to their weight coefficients.

The mixture of Gaussian model can also be used for clustering. In that case, maximum likelihood estimations of the covariances, means, and coefficients of the model are computed iteratively. This model is more sensitive to initialization [16]. Clustering algorithms can be sensitive to noise and intensity heterogeneities, because they do not incorporate spatial information.

2.1.2 Region-based Methods

Region-based methods group image pixels into regions preserving spatial connectivity among the pixels in the same region.

2.1.2.1 Watersheds

Gray level intensity images can be segmented by watershed algorithms [5, 84]. These algorithms consider a gray level image as a topographic map and a pixel's intensity value as its altitude in the map. A local minimum is the place where a drop of water falling on the topography flows down and finally reaches. A local minimum is the base for a catchment basin. Watersheds are the meeting points for the waters of adjacent catchment basins. In image segmentation, watershed lines correspond to the region boundaries. There are several approaches to segment images using watershed representation. In one of them, a downstream path is first found from each image pixel to a local minimum point of surface altitude of image. The set of pixels whose downstream paths meet in the same minimum altitude, is then defined as a catchment basin. Another approach uses flooding, in which the catchment basins are filled from the bottom, instead of characterizing the downstream paths. The barriers where water from different catchment basins meet are the watershed boundaries. Watershed methods are usually applied to gray level images and they suffer from the over-segmentation problem, which occurs when the number of regions in segmentation is higher than expected. Marker controlled watersheds are effective to alleviate the over-segmentation problem. These watersheds determine markers (flooding points), which correspond to local minima,at the beginning and allow rising the water only from these points.

2.1.2.2 Region Growing

Region growing methods extract regions that are composed of connected primitives with respect to some criteria [50, 17, 78, 3, 67, 79]. It is based on the assumption that neighbouring pixels have similar values. Seeded region growing [3] is a common form of region growing methods. Seeded region growing is a semi-supervised method because it takes a group of initial seeds as an input. The seeds determine different regions to be segmented. Each region (seed) is grown iteratively by checking unlabeled neighboring pixels. Decision to include a neighbor pixel in the growing region is made on the similarity between the feature value of the neighbor pixel and the average of pixels in the region. The most similar pixel is included in the region in each iteration. Iterations continue until all pixels are labeled. One disadvantage of the method is the requirement of supervised inputs for the seed points. Therefore, for every region that is to be segmented, a seed point is necessary. There are also methods, in which seeds are automatically determined. The JSEG algorithm [17] is one of them. It defines J value for image pixels and identifies pixel groups with smaller J values as seeds. It is also possible to use approaches start with a single seed and add new seeds if necessary. These approaches apply the same operations on the neighboring pixels with the seeded region growing approaches. Differently, they include the neighboring pixel into the region if the similarity is above a threshold. If not, a new region is created with this neighboring pixel being a seed. Region growing techniques are similar to greedy algorithms that consider the best local choices at a given time. For this reason, they may lead to local optima. Region growing techniques are also computationally expensive.

2.1.2.3 Split-and-merge

Split-and-merge methods segment an image by recursive partitioning. The image is represented as a quadtree and each segment is partitioned into four equally sized squares. Split-and-merge methods start from the root of the tree. If it finds a heterogeneous region, it splits that region into four equal squares. If four equal squares are homogeneous, it can merge them as a connected component. This operation is carried out recursively until no further splits or merges are possible.

2.1.3 Graph-based Methods

Graph-based methods construct a graph from a given image, where vertices represent pixels and the edge weights represent the similarity between two connected vertices. They then consider the image segmentation as graph partitioning. There are algorithms solving this problem using different similarity measures, different cost functions, and different optimization methods.

Graph partitioning, clustering, and image segmentation are all similar problems; they all partition the data into uniform groups. The first step to consider in solving these kind of problems is to define a criterion to optimize. The second step is to find an algorithm to carry out the optimization. Most of the time, the second step is more challenging and many attractive criteria suffer from the lack of an effective algorithm that finds the global optimum. Greedy or gradient descent based approaches usually fail to find global optimum for high-dimensional, non-linear problems. Therefore, algorithms that guarantee the global optimum are important as well as optimization criterion.

In graph-based approaches, the constructed graph can be divided into two separate parts by removing edges that connect the two parts. The distance between these two parts is defined as the sum of the edge weights that are to be removed. This is called the *cut* in graph theory literature. The optimal bisection of a graph, minimizes the cut value. Graph partitioning schemes partition the graphs into a predefined number of vertex groups by optimizing the minimum cut criterion. This can be achieved by recursively computing the minimum cuts bisecting the current regions. The minimum cut criterion may produce very small sized sets of vertices in the partition. This situation is expected, because the cut value increases with the increasing number of edges of boundary vertices. To avoid the unnatural bias for dividing small groups of points, Shi and Malik [70] propose a new measure. They find the cut as a ratio of the total edge connections to all graph vertices, instead of looking at the value of the sum of the edge weights that connect the two parts. This is called the normalized cut. With their definition of the association between the different parts, the cut that divides small isolated points will not have a low normalized cut value. Because the cut will be a large fraction of the total edge connections from that little group to all other vertices. Minimizing normalized cut exactly is NP-complete. However, Shi and Malik propose an approximate discrete solution, which uses eigenvector decomposition. They construct the affinity matrix, which is the adjacency matrix of the graph, and solve the generalized eigenvalue problem for the normalized Laplacian of the affinity matrix. They partition the graph into two parts by using the eigenvector with the second smallest eigenvalue. They recursively bipartition the graph in this way. The normalized cut yields results that are very close to the global optimum because the normalized cut criterion measures "both the total dissimilarity between the different groups as well as the total similarity within the groups".

Although its high quality results, its computational requirements are very high both in memory and CPU usage. These computational burdens make normalized cuts impractical and prohibitive in the case of large graphs and high resolution images. Felzenszwalb and Huttenlocher [27], propose a faster algorithm, for which segmentations satisfy global properties although it makes greedy choices. Its region comparison criterion uses the minimum spanning tree approach. The algorithm iterates through the graph edges deciding whether or not to merge components. Although the measures for under- and over-segmentation are defined, the algorithm cannot fully optimize these measures and often results in over-segmentation. Boykov and Funka-Lea [7] also propose a faster graph-based segmentation algorithm, in which they formulate the problem as a min-cut/max-flow problem and solve it using a fast graph cut algorithm. They report high quality results but the process is not fully unsupervised and requires supervised user inputs.

2.1.4 Statistical Methods

Statistical methods consider image segmentation as a probabilistic optimization problem. They model the image probability distributions directly, using parametric and non-parametric estimation or by using graphical models.

Markov random field modeling is a statistical model, which is used in image segmentation. Markov random fields model spatial relationships between adjacent or nearby pixels. They have the assumption that most of the pixels tend to be together in the same cluster with their adjacent pixels. This means that any region containing only one pixel has a very low probability of occurring under a Markov random field assumption. Panjwani and Healy [59] use Markov random fields to characterize a texture by interaction between different color planes and spatial interaction within each color plane. They then perform agglomerative hierarchical clustering on these models.

2.1.5 Histopathological Image Segmentation

A large portion of the available image segmentation methods are proposed for generic images for object or scene segmentation. There are very few methods specifically proposed for histopathological image segmentation. There are studies [71, 85] that use color and texture features to segment tissue images. These studies perform grid analysis on the images. For that, they divide the images into fixed sized square grids and extract color and texture features from the pixels of the grids. Then they classify each grid in a supervised way. These studies use the features defined on pixels but they do not consider the background knowledge of tissue organization to define them. Actually, it is difficult to express background knowledge in terms of pixels.

Most recent segmentation algorithms have proposed to incorporate medical knowledge of a pathologist into the feature definition [82, 81]. These studies approximately locate tissue components and define texture descriptors on these components. Such representations provide good descriptors for histopathological tissue images. However their segmentation parts, which use seeded region growing algorithms need improvement.

2.2 Multilevel Framework

Multilevel framework was introduced to be used in the graph partitioning problem. Graph partitioning problem shows its significancy in a variety of subjects such as VLSI design, scientific computing, and task scheduling. Graph partitioning divides the vertices in graphs into p approximately equal parts by minimizing the sum of weights of edges between vertices in different parts. To solve linear equations like $Ax = b$, using iterative techniques using parallel processing, one has to deal with the graph partitioning problem. In these kind of techniques, multiplying a sparse matrix with a dense vector is an important step. If the related matrix A is partitioned well, then a considerable amount of decrease in the communication volume in sparse matrix-vector multiplication for parallel processing.

Graph partitioning is an NP-complete problem. However, there are algorithms that can produce fairly good partitions. It is known that spectral graph partitioning techniques produce good quality results for a wide range of problems [41, 63, 62]. However, these techniques are computationally inefficient, because they need to compute the eigenvector corresponding to the second smallest eigenvalue, also known as Fiedler vector. In order to overcome the computational burdens of spectral graph partitioning methods, multilevel graph partitioning algorithms are proposed [42, 40, 48, 11, 76]. It is seen that multilevel algorithms produced high quality partitions extremely fast compared to the spectral methods.

Multilevel graph partitioning algorithms consist of three phases called coarsening, partitioning, and uncoarsening. They basically decreases the size of the original graph before partitioning. Partitioning the small sized graph takes very little time. They then uncoarsen the small graph by refining the partition at each level. In the coarsening phase, vertices of the graph are visited and merged with their neighbors to form multinodes. The original graph is repeatedly coarsened level by level until a small number of multinodes remain. An initial division of the coarsest graph is performed. Then, this partition is improved as the small graph is uncoarsened level by level. They make use of iterative improvement heuristics [49, 30] to refine the coarse graph in the uncoarsening phase. In the Kernighan-Lin [49] heuristic, pairs of vertices from the adjacent parts are swapped in each step whereas in the Fiduccia-Mattheyses [30] heuristic, a single vertex is moved from one part to another.

The objective is to minimize the sum of the edge weights that are incident to vertices on the boundary of the partition. It is an expected situation that the method put a single vertex with the minimum sum of edge weights into one part and other vertices to the other part to minimize the cut value. For this reason, these heuristics compute the gain of each vertex move, to the cut value considering the balance of the parts. This balance constraint can be limiting and lead to poor results in specific areas of applications like clustering and image segmentation.

2.3 Cluster Ensembling

Clustering is the process of grouping unlabeled data objects into clusters with respect to a similarity definition to ”maximize the intra-cluster similarity and to minimize the inter-cluster similarity” at the same time [23]. Clustering is an important subject in the machine learning research. Ensemble learning also became very popular and attracted more attention recently. Ensemble learning combines the results of different methods or the same method with different parameters settings to obtain a superior result than the single runs of other learners [56]. Ensemble learners have a better generalization ability. They are more robust and produce high quality results. Ensemble learning was extensively used with supervised methods in the past [8, 64, 51, 83, 69] . Recently, ensemble learning is started to be used with unsupervised methods. Strehl and Ghosh [74] proposed to use ensembles of different clustering algorithms. Topchy *et al.* [80]

showed that cluster ensembles can do better than the typical single clustering algorithms in terms of robustness, stability, and scalability.

A certain clustering method which has a specific view of the data is defined as a *clusterer*. Each clusterer produces cluster labels for some or all data objects. Cluster ensembling is the problem of combining many different clustering of data objects using only cluster labels without accessing the original features. Each *clusterer* can use different feature descriptions and different grouping techniques. Cluster ensembling is a good way of using different feature spaces together to get a better view of the data.

Previous studies showed that, in classification or regression problems, performance improvements from using ensemble techniques are directly related to the amount of diversity among the individual component models [51, 83]. The ideal ensemble should contain models that are powerful and have different inductive biases to be able to make distinct generalizations [19]. Therefore, ensembles are mostly used for integrating relatively unstable models such as decision trees and multi-layered perceptrons. Recent studies [28, 52] showed that diversity and quality of individual clustering results increase the cluster ensemble performance as in supervised ensembling.

To increase the diversity of individual clustering results, different clustering algorithms can be used. Also a single clustering algorithm can be modified to produce diverse results by means of randomization and other techniques. Random sub-sampling is a way of increasing the diversity of a single algorithm. For each clustering run, actual dataset is sub-sampled with a predefined percentage of sub-sampling. Then clustering is performed with the sub-sampled data objects and each data object omitted from the current sub-sample is assigned to its nearest cluster center to ensure that all the data objects are clustered. Another way of increasing the diversity is to use random projection. In random projection, for each clustering run, data objects are projected to a lower dimension feature space randomly. Then clustering is performed on the low-dimensional data set. This is actually effective in the case of high-dimensional data sets. Different runs of diverse clusterings recover different parts of the structure of the data and the increased number of diverse clustering solutions approach to capturing almost perfect structure of the data.

Good quality partitionings are also necessary to increase the performance of cluster ensembles. Using *k-means* with random initializations is not very effective. Because algorithms like *k-means* are based on the convex spherical sample space and most of the time the sample space is not convex trapping the algorithms into local minimum.

The last step is to combine the clustering solutions using a consensus function to get the final cluster labels. Creating a similarty matrix based on the frequency of being in the same cluster for pairs of data and applying agglomerative clustering on the new similarity matrix yields good results. However such an approach is computationally inefficient. Strehl and Ghosh [74] propose two approaches that use graph partitioning techniques in cluster ensembling. The first technique they propose is an instance based technique. In this technique, a similarity matrix, which contains the pairwise information of instances' frequency of being clustered together is constructed. This similarity matrix is considered as the adjacency matrix of the graph and the graph is partitioned. The second technique they propose is a cluster based technique, in which clusters are modeled as vertices. The weights of edges are defined as the ratio of instances that the incident clusters share. The original cluster ensemble cannot be reconstructed from a graph that is constructed by the instance based or the cluster based technique. Therefore, both techniques lead to information loss from an ensemble. Fern and Brodley [29] propose a graph formulation which represents both instances and clusters as vertices in a bipartite graph. This kind of graph preserves all of the information of an ensemble. It allows both the similarity among clusters and the instance to be taken into account collectively to produce the final clusters. The resulting graph partitioning problem can be solved efficiently.

Chapter 3

Methodology

In this thesis, we propose a segmentation algorithm to achieve high quality and stable results. The proposed method relies on obtaining several clusterings each of which produces diverse and high quality results, and effectively combining the results of these clusterings by a consensus function. In order to obtain high quality and diverse clusterings at the same time, we proposed a modified version of the multilevel graph partitioning scheme. For that, we first removed the balance constraint because the segments in tissue images can be in any arbitrary shape. We then removed the initial partitioning phase and randomized the boundary refinement step in the uncoarsening phase of the multilevel graph partitioning scheme.

The proposed method consists of the following three steps: (1) feature extraction and graph construction, (2) clustering, and (3) ensembling with a consensus function. In the feature extraction and graph construction step, the object graph of image is constructed by detecting medically meaningful objects as vertices and Delaunay triangulation is applied on these objects to define edges between the vertices. Then, a set of features is defined and the distance between the features of two adjacent vertices is assigned as the weight of the graph edge between these two vertices [81]. Once the input object graph is constructed, it is clustered by the clustering algorithm which uses multilevel graph partitioning several times to produce different clusterings. Produced clusterings are then combined using a

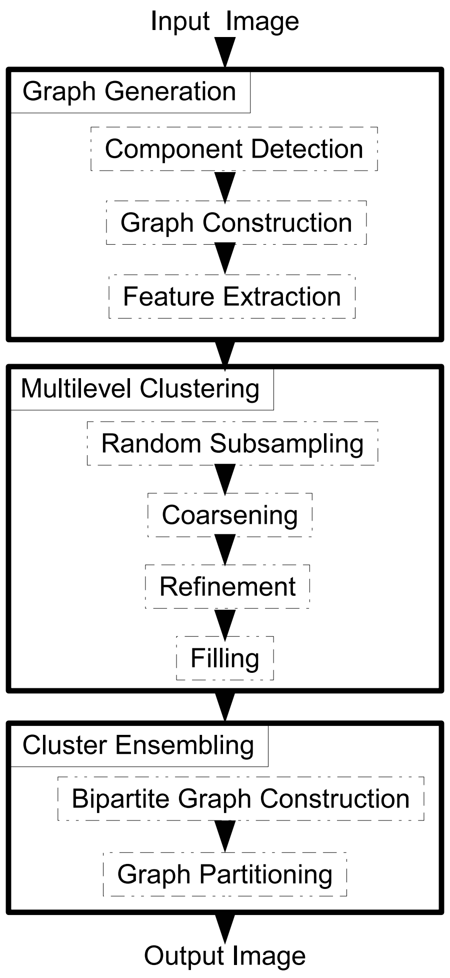


Figure 3.1: Overview of the proposed method

consensus function. The consensus function constructs a bipartite graph between the clusterings and the objects and obtains the final cluster labels of the objects using a graph partitioning algorithm.

3.1 Graph Construction and Feature Extraction

To segment histopathological images, Tosun *et al.* proposed a method that incorporates domain specific knowledge of a pathologist into segmentation [81]. In this method, they represented histopathological objects as object graphs and defined high-level textural features on these graphs. This gives a powerful representation

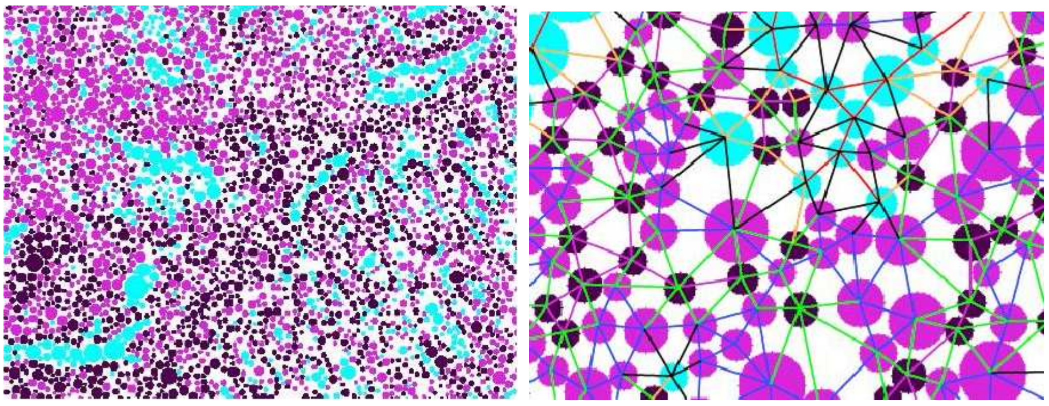

(a) Components (b) Edges

Figure 3.2: Detected tissue components(a). A close up view of some of the edges found by Delaunay triangulation (b).

method that yields promising segmentation results. However, the segmentation part of their method uses a standard region growing algorithm, which has a risk of obtaining local optimal results. Therefore it may lead to inaccurate and unstable results.

In this thesis, our main focus is to design and implement a new segmentation algorithm that yields more accurate and stable results. For this purpose, we use the features previously defined by Tosun *et al*. [81] and focus on the segmentation part rather than the feature extraction part. In this section, we briefly mention the feature extraction. The reader is referred to the previous work [81] for comprehensive explanation.

Image pixels are clustered into three clusters using *k-means* algorithm on their color information. Because after the staining procedure, tissue images get three colors and their variations. These colors are purple, pink, and white, which typically correspond to nuclear, stromal, and luminal components, respectively. After clustering, circle fitting heuristic [82, 39] is applied on the clustered pixels to locate different type of tissue components. Each detected tissue component is considered as a vertex of the graph. Connectivity of vertices is determined by the Delaunay edges that are found on the cartesian coordinates of the centers of the tissue components. After producing the graph representation of the image, edge weights are defined by the feature extraction process. The tissue components or the graph vertices are considered as primitive objects instead of pixels. A modified version of the gray level run-length features [33] which are proposed for pixels are defined on the image graph vertices [81] . For each vertex (component) a 16 dimensional feature vector is computed. The weights of the edges between two tissue components are computed as the euclidean similarity between the feature vectors of the components. The resultant graph is an undirected graph with weights on its edges.

3.2 Clustering

We solve image segmentation problem through clustering. In both problems, there is a similar objective in which primitive elements are clustered into uniform groups. Our clustering scheme utilizes the cluster ensembling technique. By ensembling multiple different clustering solutions of the same data, much better and stable clusterings can be obtained. The performance of cluster ensembles are affected by two criteria which are diversity and quality. Each clustering solution in the ensemble should be different than other solutions to improve the ensemble performance. Each clustering solution should contribute to the final solution by capturing a different aspect of the structure of the data. Also individual clustering solutions should be of high quality. One should not sacrifice quality for the sake of diversity. There should be a balance between the two criteria. So we proposed an efficient algorithm that satisfies diversity and quality constraints.

Partitioning of a graph also corresponds to clustering the vertices of that graph. For that reason, we solve the clustering problem by graph partitioning. We use the terms clustering and partitioning interchangeably in this work. Graph partitioning problem is described as follows: Given a graph $G = (V, E)$ with $|V| = n$, partition V into k subsets, $V_1, V_2, ..., V_k$ such that $\bigcup_i V_i = V$ and $V_i \cap V_j = \emptyset$ for $i \neq j$, the partitioning objective is to minimize the number of edges whose incident vertices are in different parts. The partitioning constraint is to maintain a given balance on the part weights, where the weight of a part, is defined as the number of vertices in that part. For edge weighted graphs, the partitioning objective becomes minimizing the sum of weights of the edges whose connected vertices are in different parts. A partition result of vertex set V is shown by a vector Π of length n. All vertices $v \in V$, $\Pi[v]$ is a number between 1 and k. This number indicates the partition of the vertex v. The *edge cut* of a partition Π is the number of edges whose connected vertices are in different parts.

Algorithm 1 Segmentation Algorithm

```
Input: image $I$, integer $K$, integer $K'$, float $prc$, integer $nCls$
Output: image $R$
 1: $V = \{<x_1, y_1>, <x_2, y_2>, ..., <x_n, y_n>\} \leftarrow$ ObjDetect($I$)
 2: $E = \{e_{a,b} = (v_a, v_b) \mid v_a \in V and\ v_b \in V\} \leftarrow$ DelTriEdges($V$)
 3: $V = \{<\mathbf{v_1}, x_1, y_1>, <\mathbf{v_2}, x_2, y_2>, ..., <\mathbf{v_n}, x_n, y_n>\} \leftarrow$ FeatExt($V, E$)
 4: $W = \{w_{a,b} = sim(v_a, v_b) \mid e_{a,b} \in E\ and\ v_a \in V and\ v_b \in V\}$
 5: $G = (V, E, W)$
 6: $\Psi = \emptyset$
 7: for $i = 1 \rightarrow nCls$ do
 8:     $V_0^s = \{v_1^s, v_2^s, ..., v_p^s\} \leftarrow$ RandSubSample($V, prc$)
 9:     $G_0^s \leftarrow$ ConstructObjectGraph($V_0^s$)
10:     $\Pi' \leftarrow$MultilevelGraphPartition($G_0^s, K'$)
11:     $\Pi \leftarrow$ Fill($\Pi', G$)
12:     $\Psi = \Psi \cup \{\Pi\}$
13: end for
14: $R \leftarrow$ ClusterEnsembling($\Psi, K$)
```

Algorithm 2 Multilevel Graph Partitioning

```
Input: graph $G_0^s$, integer $K'$
Output: set $\Pi'$
 1: $[G^s, numLvls] \leftarrow$ Coarsening($G_0^s, K'$)
 2: $\Pi' \leftarrow$ Refinement($G^s, numLvls$)
```

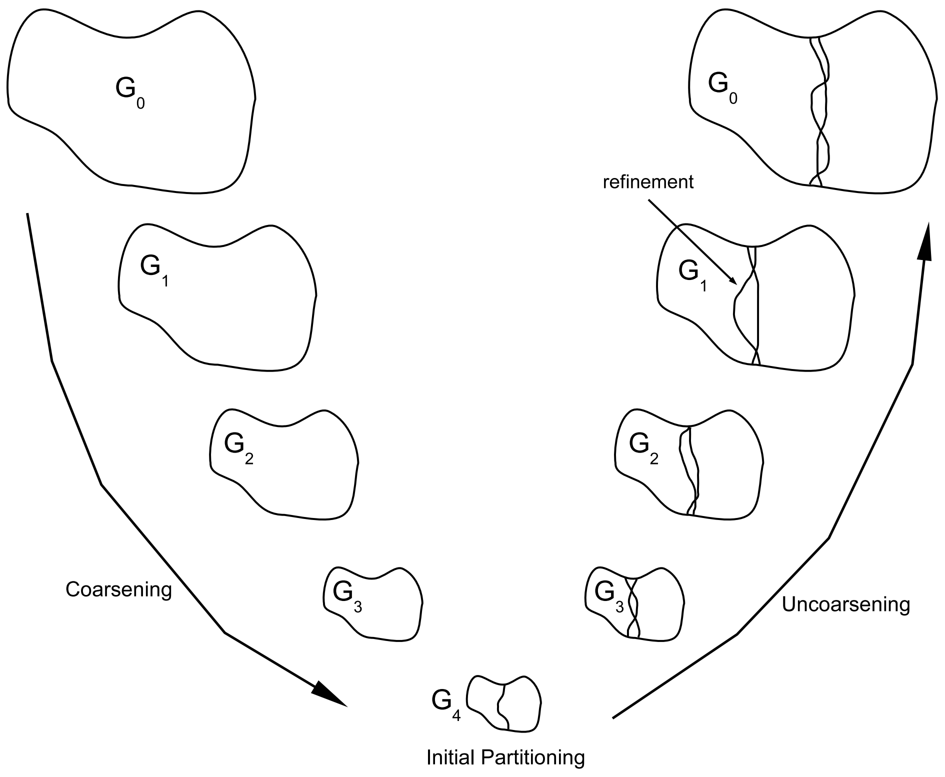
Algorithm 3 Construct Object Graph

```
Input: set $V$
Output: graph $G$
 1: $E = \{e_{a,b} = (v_a, v_b) \mid v_a \in V and\ v_b \in V\} \leftarrow$ DelTriEdges($V$)
 2: $W = \{w_{a,b} = sim(v_a, v_b) \mid e_{a,b} \in E\ and\ v_a \in V and\ v_b \in V\}$
 3: $G = (V, E, W)$
```

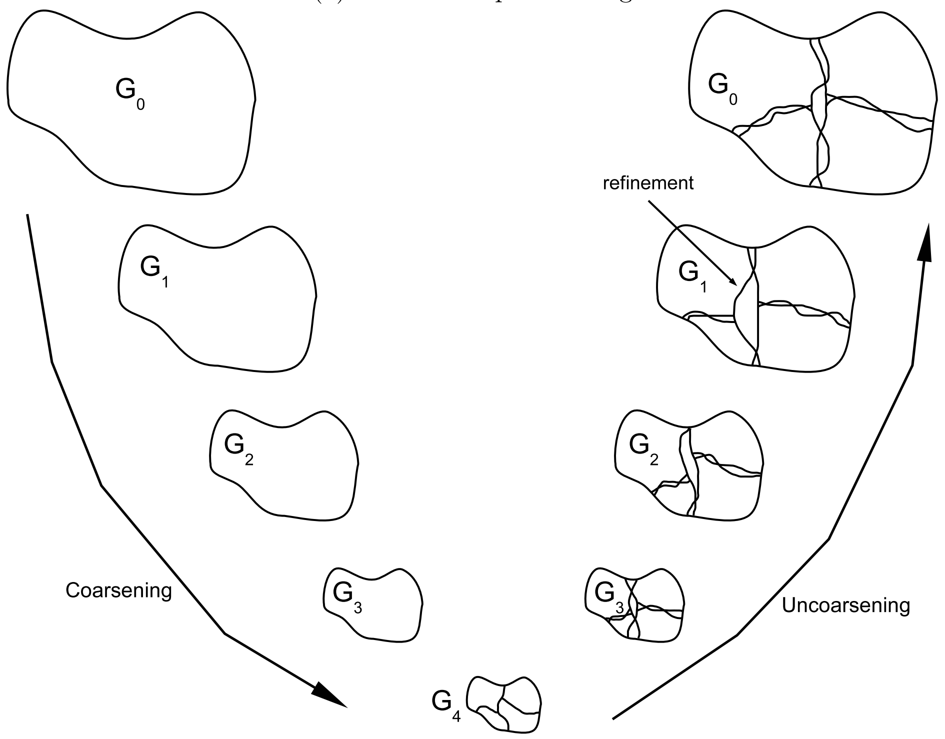
Multilevel graph partitioning algorithms produce good partitionings of graphs efficiently. Our method partitions the input graph using a multilevel scheme which produces high quality results efficiently. In order to obtain different partitionings each time the algorithm is run, we made some modifications in the multilevel scheme. Multilevel graph partitioning algorithms coarsen the original graph by clustering its vertices. They partition the resultant smaller graph much more faster than the original graph. They then uncoarsen the smaller graph by refining the partitions level by level. The partitioning of the smaller graph has an important effect on the final partitioning result. In our method, we omit the initial partitioning of the smaller graph for diversity. We also randomize the coarsening and the uncoarsening steps. But, we do not fully randomize everything, we still optimize some local criteria. Multilevel graph partitioning algorithms stop the coarsening step when approximately a hundred vertices remain, they then perform initial partitioning on the coarsest graph. In our case in which the initial partitioning step is omitted, the coarsening of the original graph stops when a few vertices remain.

Our segmentation method is described in Algorithm 1. The first five lines of the algorithm show the component detection, graph construction and the feature extraction steps as described in [81] . In each iteration of the for loop between the lines 7 and 13, a different clustering solution is produced to be used in the ensemble. In the last line of the algorithm, multiple clustering results produced in the for loop are combined using a consensus function to get the final clustering result.

Our multilevel method is composed of two main phases which are the coarsening and the refinement phases as described in Algorithm 2, while traditional multilevel schemes also have an initial partitioning step between those two. Other parts in the for loop are just pre-processing and post-processing steps that improve the performance of the ensemble. In line 8, a random subset of the vertices is selected from the input graph with a predefined percentage. In line 9, a new Delaunay triangulation is computed and the new edges between the vertices of the selected subset is defined as the computed Delaunay edges (Algorithm 3) . This is necessary because the selected subset of the vertices have different spatial relationships with each other. After defining the connectivity of the selected vertices, edge weights are computed as the euclidean similarity between the feature



(a) with Initial partitioning



(b) without Initial partitioning

Figure 3.3: Multilevel Graph Bipartitioning with Initial partitioning (a). Multi-level Graph Partitioning without Initial partitioning (b).

vectors of adjacent vertices.

In the subsequent coarsening and refinement phases, the sample graph is partitioned. The resulting partition vector in line 10 only reports the cluster labels of the vertices that were in the selected subset. In order to obtain a complete labeling for all vertices in the graph, a filling operation is performed in line 11. What is done in this operation is to assign the vertices, that are absent in the selected subset, the label of the closest vertex in the selected subset in terms of spatial proximity.

3.2.1 Random Subsampling

We mentioned that diversity and the quality of the individual clusterings affect the ensemble performance. Increasing the diversity helps improving the clustering performance. There are various techniques to increase the diversity of a clustering algorithm such as randomization of clustering steps, randomization of the data and the randomization of features. Random projection is the process of randomizing the features. In random projection, in each clustering run, a random projection of the features are used to cluster data. This technique is especially useful in the case of high dimensional data and act as a dimensionality reduction scheme. But in our case, the feature definition we use is not very high dimensional (there are 16 dimensions) and no significant improvement can be obtained by random projection. Another technique is to randomize the steps of the clustering algorithm. We randomized some key steps in our clustering algorithm.

There is also the random sub-sampling technique. In random sub-sampling, in each clustering run, a random subset of the data objects are selected and only these objects are used in the clustering process. This technique also provides different views of the data to the clustering algorithm and is immune to the noise and variations in the data. In our method, each clustering solution is produced using a different random subset of vertices. Before the multilevel graph partitioning step, we select the sample vertices with a uniform random distribution.

Then we redefine the edges between the selected vertices by Delaunay triangulation and computing the euclidean similarity of adjacent vertices. We call this resultant graph the sample graph. Multilevel graph partitioning is performed on this sample graph.

3.2.2 Multilevel Graph Partitioning

3.2.2.1 Coarsening

Aim of the coarsening phase is to produce a smaller version of the input graph. The input graph G_0 is transformed into a number of small graphs $G_1, G_2, ..., G_m$ such that $|V_0| > |V_1| > |V_2| > ... > |V_m|$. In the coarsening step, a combination of smaller graphs each having lesser number of vertices, is constructed. In most multilevel schemes, a group of vertices of G_i are merged to create a coarser graph's single super-vertex for the next level G_{i+1}. In traditional multilevel schemes, to retain the connectivity in the next level graph, the edges incident to a super-vertex are the union of the edges of its constituent vertices.

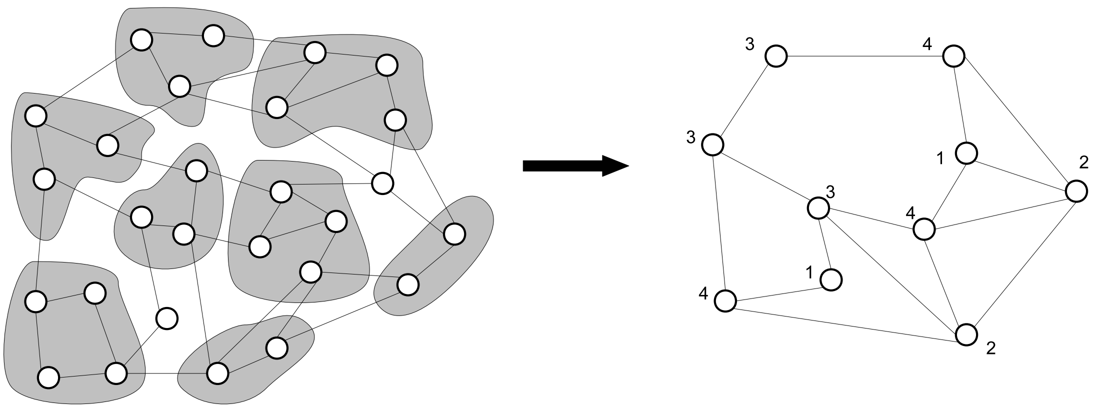


Figure 3.4: Coarsening a graph

A coarser version of a graph G_i can be produced by merging its adjacent vertices. A super-vertex composed of these two adjacent vertices is produced by collapsing the edge between them. The formal definition of collapsing of edges can be made using matchings. A subset, where no two edges are incident to a common vertex, is a *matching* of a graph. G_{i+1} which is a coarser version of the graph G_i, is constructed by computing a matching of G_i and merging each pair of vertices into super-vertices. There will be unmatched vertices and those are preserved in the G_{i+1}. The matching should be composed of many edges. Because the objective of merging vertices using matchings is to construct a smaller version of the graph G_i. To obtain each next level coarser graph, *Maximal matchings* are used. In a maximal matching, any edge in the graph that is not in the matching has one of its endpoints matched. Based on the type of method used for finding matchings, the number of edges in maximal matchings can be different. *Maximum matching* is the maximal matching with the maximum number of edges. But maximal matching is preferred because of its computational complexity. Using matchings in the coarsening phase, conserves many features of the original graph which is desirable.

Algorithm 4 Coarsening

```
Input: graph $G_0^s$, integer $K'$
Output: set $G^s$, integer $numLvls$
1: $numLvls \leftarrow 1$
2: $currNumObjs \leftarrow n$
3: $r \leftarrow 0$
4: while $currNumObjs > K'$ do
5:     for each randomly visited vertex $v$ in $V_r^s$ do
6:         if $v$ is not $merged$ with any other vertex then
7:             $T = \{t \mid e_{t,v} = (t,v) \in E \ and \ v \in V and \ t \in V\}$
8:             $t \leftarrow \underset{t \in T}{\operatorname{argmax}} \ sim(t,v)$
9:             merge vertices $v$ and $t$
10:            update $E_r^s$ and $W_r^s$
11:            mark $v$ and $t$ as $merged$
12:            $currNumObjs = currNumObjs - 1$
13:        end if
14:    end for
15:    $G_{r+1}^s = G_r^s$
16:    $G^s = G^s \cup \{G_r^s\}$
17:    $numLvls = numLvls + 1$
18:    $r = r + 1$
19: end while
```

There are different ways to generate a matching of a graph to coarsen it. Using a randomized algorithm, a maximal matching can be found. In the random maximal matching, vertices of a graph are randomly visited. If there is a vertex u which is not matched, then one of its unmatched neighboring vertices is randomly selected. Two vertices are said to be adjacent if there exists an edge that is incident to those two vertices. If there exists such a vertex v, the edge (u, v) is included in the matching and the vertices u and v are marked as matched. Vertex u remains unmatched in the random matching if there is no unmatched adjacent vertex v.

The goal in the graph partitioning is to minimize the sum of the weights of the edges between the vertices on the boundary of the parts of the graph. So a randomized matching method may not always produce satisfactory results for every graph. In order to decrease the edge cut value, *heavy edge matching* [48] can be used. In heavy edge matching, vertices are again visited randomly but the visited vertex is matched with its adjacent vertex with the greatest edge weight. This helps decreasing the final edge cut value.

The coarsening phase of our algorithm which is invoked in line 1 of Algorithm 2 is described in Algorithm 4. There are two inputs to the algorithm. First one is the sample graph which is a random sub-sample of the original image graph. Second one is the number of parts. In traditional multilevel schemes, the graph is partitioned after the coarsening phase. Partitioning is required before the refinement phase because it will refine the boundaries of the parts. The coarsening phase of our algorithm continues until a few vertices remain. The resultant coarsest graph is considered as an initial partitioning for the refinement phase.

The *while* loop between lines 4 and 19 shows the steps of a single level coarsening operation. After each level of coarsening a smaller graph in the size of vertices is produced. This phase goes on by further coarsening the output of the previous level coarsened graph. The coarsest graph is obtained after the final level of coarsening. Number of levels of coarsening depends on the desired number of vertices of the coarsest graph. As described between lines 4 and 19, a level of coarsening is as follows. Each vertex of the graph is randomly visited. This corresponds to the $for\ each$ loop between lines 5 and 14. The visited vertex is checked if it is already merged with another vertex or super-vertex in this level. If it is not merged with any other vertex or super-vertex, then the vertices or the super-vertices which are incident to the visited vertex are considered. The one with the greatest edge weight is merged with the visited vertex.

The merging process in our coarsening phase is different from traditional coarsenings. First of all we keep the feature vectors of the vertices. When we produce a super-vertex consisting of many vertices, the feature vector for the produced super-vertex is computed as the average of the vertices it contains. Keeping the feature information of the vertices also affects the edge collapsing process. The weight of a collapsed edge between two super-vertices, is computed as the Euclidean similarity of the super-vertices considering the finest level vertices constituting these two super-vertices. In line 10 connectivity of the vertices and the edge weights are updated. In line 11, merged vertices are marked to prevent them merging again in the current level. In lines 15 and 16 every coarse graph produced in each level are saved to be used in the refinement phase. The resultant coarsest graph is a smaller version of the sample graph which preserves its properties.

3.2.2.2 Refinement

The refinement phase uncoarsens the small graph to its original size by improving the partition. The partition Π_m of the coarser graph G_m is uncoarsened into the input graph. Original graph is obtained by using the graphs $G_{m-1}, G_{m-2}, ..., G_1$. Obtaining Π_i from Π_{i+1} is done by putting the vertex group V_i^v merged into $v \in G_{i+1}$ to the partition $\Pi_{i+1}[v]$. Because each vertex of G_{i+1} is composed of a different group of vertices of G_i.

Π_{i+1} is a local optimum partition of G_{i+1}. But the uncoarsened partition Π_i may not be at a local optimum according to G_i. Because G_i is finer than G_{i+1}, G_i has more degrees of freedom which can be utilized to refine Π_i, and reduce the cut value. Uncoarsened partition of G_{i-1} can also be improved by local improvement heuristics. Therefore, after uncoarsening a partition, a refinement heuristic is used. Partition refinement heuristics aim to find two vertex subsets, a set from each different part which minimizes the edge cut when swapped. If X and Y are the two parts of a partition, a refinement heuristic selects $X' \subset X$ and $Y' \subset Y$ such that $X \setminus X' \cup Y'$ and $Y \setminus Y' \cup X'$ is a partitioning with a smaller edge cut.

There are algorithms producing high quality results based on Kernighan-Lin [49] and Fiduccia-Mattheyses [30] heuristics. Kernighan-Lin heuristic swaps pairs of vertices from the adjacent parts in each step whereas in Fiduccia-Mattheyses heuristic, a single vertex is moved from one part to another part. This kind of algorithms compute the best possible swap that decreases the edge cut the most, before moving any vertex. They also consider the balance of the parts of the graph. They prevent making swaps that will distort the balance of the parts of the graph. In our algorithm we use a similar heuristic but modify some steps. We remove the balance criterion and we make greedy vertex moves, instead of best gain swaps.

Algorithm 5 Refinement

```
Input: set $G^s$, integer $numLvls$
Output: set $\Pi'$
 1: for $r = numLvls - 1 \rightarrow 1$ do
 2:    $B_r \leftarrow boundary\ vertices\ of\ V_r^s$
 3:    repeat
 4:       {This $foreach$ loop is called a $pass$}
 5:       for each randomly visited vertex $b$ in $B_r$ do
 6:          if $b$ is not $moved$ in this $pass$ then
 7:             move vertex $b$ into the most similar region
 8:             update $\Pi'$, $B_r$
 9:          end if
10:       end for
11:       find newly emerged $regions$
12:    until $numRegions$ is not changing
13: end for
```

The refinement phase of our algorithm which is invoked in line 2 of the Algorithm 2 is described in Algorithm 5. The input to the algorithm is a set of graphs that are produced by the coarsening phase. This set contains a coarser graph for each level. The for loop between the lines 1 and 13 uncoarsens the coarsest graph level by level. Refinement starts from the coarsest graph. Each vertex in a coarser level contains vertices of the next finer level. In our algorithm we omitted the initial partitioning phase and our coarsening phase produces a graph with few vertices where in the refinement step each super-vertex of the coarsest graph is treated as a part. First, super-vertices of the coarsest graph are uncoarsened and the vertex sets coming from different super-vertices are considered as parts. Then boundary refinement heuristic is run on the vertices that are on the boundaries. This process goes on level by level until the original graph is obtained.

We made some modifications on the refinement phase and in the boundary refinement step. In each level, we first uncoarsen the vertices of super-vertices of the coarser graph. Then, in line 2, vertices on the boundaries of the parts are detected. A vertex is a boundary vertex if one of its incident vertices are on a different part. Our boundary refinement method can cut off vertices from other parts and create new parts. In each level, we repeat the boundary refinement pass until it converges to a constant number of parts. This step is described in the *repeat until* loop between lines 3 and 12. We call the $for\ each$ loop between the lines 5 and 10 a *pass*.

In a pass, boundary vertices are randomly visited. The visited vertex is checked if it is already moved in this pass. This is important because we may get stuck in some local minimum and end up moving the same vertex repeatedly. In order to prevent this *thrashing* process, moved vertices are locked. There are two things to do if a vertex is not moved in that pass. Move the vertex to the adjacent part, or leave it in its current part. Since we construct our input graphs from images by Delaunay triangulation, our graphs are planar and the boundary vertices can only be moved to one adjacent region. Decision to move the vertex is made as follows. The visited vertex is considered as a single part. Then its euclidean similarity to the adjacent region and its own region without the visited vertex, is computed. If it is more similar to the adjacent region then it is moved to that region otherwise it is left in its own region. If the vertex is moved, feature values of the regions are recalculated incrementally. After each vertex move, some vertices can loose their property of being a boundary vertex and some vertices can become a boundary vertex. We take this issue into account and update the boundary vertices incrementally after each vertex move. A *pass* stops when a number of vertices are visited. We take it as the number of boundary vertices at the start of the pass.

Our experiments showed that, in this kind of a refinement environment, some vertices can loose their connectivity with the region they belong to. It means that they are different from the region they are currently in and also different from the adjacent region. In such a case, they should be considered as new regions. When this kind of vertices are allowed to form new regions, we observed that they are cut off and merged with some other vertices from their neighbouring regions. So after each pass, we check the connectivity of the vertices with their regions. If we detect any loss of connectivity, we take those vertex groups as new parts or regions. We do this at the end of each pass because of computational requirements. Forming new regions or parts in the refinement step is actually useful in two ways. First, it helps produce better partitions by capturing the structure of the data. Second, it improves the cluster ensembling performance providing a finer grain information about the object pairs' frequency of being clustered together.

With our refinement heuristic, number of regions or parts of the graph, changes after a pass forming new regions. Experiments also showed that, after a number of passes, the number of parts of the graph stays the same. Number of regions in the image actually converges to a constant number. So in each level, the number of refinement passes are not static in our method. We stop the refinement passes in a level, when there is no change in the number of parts of the graph happens.

3.2.3 Filling

As mentioned before, we use random sub-sampling to improve the ensemble performance. In cluster ensembling, the data objects' frequency of being in the same cluster defines a new feature over the data objects. Then the final clustering result is obtained from these features. To increase the diversity of the individual clustering solutions, we used random sub-sampling. We generated a sample graph from the original input graph and performed the clustering on this sample graph. After the clustering, we only have the labels of the vertices that were in the random subset. But in order to define a new similarity between every pair of data object, we need the cluster labels for all objects. For this reason, we estimate the cluster labels of the vertices that were not in the random subset as shown in line 11 of Algorithm 1. We label every unlabeled vertex with its most similar and spatially close labeled vertex.

3.3 Consensus Function

After producing multiple clustering solutions, we should solve the problem of combining these results into a final clustering. This is a difficult problem since clustering is an unsupervised process where the number of classes in the data is unknown. Different clustering solutions can be coming from different feature space based clustering algorithms. So the final clustering should be obtained independent of the individual feature definitions of different clustering algorithms that are in the ensemble. Only the cluster labels should be used to combine different clusterings.

Approaches like majority voting in supervised ensembling can be used. One approach is to define a new similarity based on the cluster labels of objects in different clusterings. This approach considers the objects' frequency of being clustered together. The assumption is, if an object tends to be in the same cluster with another object in most of the different clustering results, then they will most likely be in the same cluster in the final clustering solution. With this assumption a new similarity matrix is defined. The similarity between two objects in this matrix is their frequency of being clustered together. After the construction of a new feature of the objects, clustering can be performed on the new similarity matrix.

Using agglomerative clustering on the new similarity matrix, yields good results. But agglomerative clustering is not an efficient way of partitioning the data when the data set is large. To solve this problem efficiently, clustering the

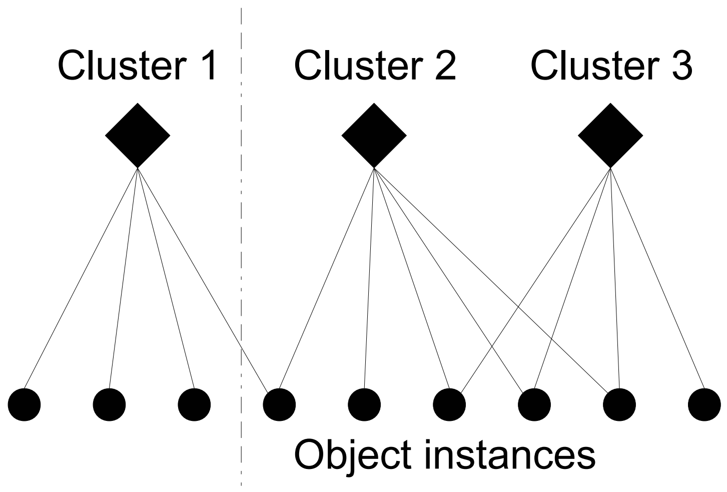


Figure 3.5: Bipartite graph representation of vertices and clusters

similarity matrix can be reduced to partitioning the graph constructed from the similarity matrix. We construct a weighted undirected graph and partition it efficiently using the state-of-the-art graph partitioning techniques.

There are techniques that use graph partitioning in cluster ensembling. First one is an instance based technique. This is actually the graph version of the similarity matrix which is constructed with the cluster labels from the ensemble. Each edge weight is the similarity between the pair of vertices it is incident to. The second one is a cluster based technique. It represents clusters as vertices of a graph and computes the edge weights as the ratio of instances they share. These approaches can cause information loss. The actual graph based cluster ensemble, constructed by an instance based or a cluster based technique, cannot be reconstructed. Fern and Brodley [29] proposed a new hybrid method that models both clusters and instances as vertices of a bipartite graph. This kind of bipartite graph preserves all of the information in the ensemble. It allows both the similarity among clusters and the instances to be taken into account collectively in the construction of the final clusters.

A cluster ensemble contains multiple different clustering solutions and is defined as $C = \{C^1, C^2, ..., C^t\}$. A bipartite graph $G = (V, E)$ is constructed where $V = V^C \cup V^I$. There are t vertices in V^C and each vertex represents a cluster from the ensemble. V^I consists of n vertices and each of them represents an instance of the data set. $E(i, j)$ is equal to zero, if the vertices j and i are both instances or both clusters. Otherwise if instance j is in to cluster i, both $E(i, j)$ and $E(j, i)$ are equal to 1. If not, they are equal to 0. Figure 3.5 shows the constructed bipartite graph where diamond vertices are for clusters and round vertices are for instances. A vertex is connected with an edge to the cluster it is contained. All weights of the edges of the graph are unit weights. Instance vertices are only connected to the cluster vertices. The first advantage of this kind of representation is that the actual cluster ensemble can be easily recovered from the bipartite graph. The second advantage is that, it does not treat similarity of clusters and similarity of instances independently. Further justifications can be found in [29].

After the bipartite graph is constructed, it is partitioned. We can use numerous graph partitioning algorithms here. We do not use the graph partitioning algorithms with the balance constraint because of the sizes of the expected clusters. The Normalized Cut [70] produces good results here but its computational requirements are very high. So we use the multilevel graph partitioning algorithm proposed by Dhillon *et al.* [18]. This algorithms produces very close results to the Normalized Cut algorithm by removing the computational requirements and the balance constraint on the cluster sizes. The partition includes both cluster vertices and instance vertices. We only take instance vertices into account and we get the final cluster labels of the data objects.

```
Algorithm 6 Cluster Ensembling
Input: set $\Psi$, integer $K$
Output: partition $R$
1: $bg \leftarrow$ ConstructBipartiteGraph($\Psi$)
2: $R \leftarrow$ GraphPartition($K, bg$)
```

3.4 Post processing

We detected medically meaningful tissue components in the input tissue image and generated a graph representation of it. Then we partitioned the graph and obtained the cluster label of each component. We also need the cluster label of each pixel of the image. For this reason, the Voronoi diagram of the components is computed. There is a component at the center of each Voronoi cell. Finally the pixels in each Voronoi cell are labeled with the label of the component at the center of that Voronoi cell. By this the final segmentation of the image is obtained.

Chapter 4

Experiments and Results

In this chapter, we describe our dataset, explain our experimental setup, and present the results of the proposed segmentation algorithm. We also describe the validation method we used in the experiments and provide comparisons with other existing segmentation algorithms proposed for generic and tissue images.

4.1 Experimental Setup

This section provides information about the images used in the experiments and the method that we use to validate segmentation results.

4.1.1 Dataset

Our dataset is composed of 200 images. These images are obtained from the colon biopsy samples that are selected randomly from the Pathology Department archives of Hacettepe School of Medicine. The samples are 5-6 μm thick tissue sections and they are stained with the routine *hematoxylin and eosin* technique. A Nikon Coolscope Digital Microscope is used to obtain images from biopsy samples with 5$\times$ microscope objective lens and 1920 $\times$ 2560 image resolution.

Fifty of the images are used in the training set to estimate the parameters of the algorithm. The remaining 150 images, which are not used in parameter estimation, are used in the test set to measure the performance of the algorithm.

The experimental system is developed in MATLAB 7.11, then deployed on a 64 bit UNIX-based server with four 2.1 GHz 6-core AMD Opteron processors and 128 GB of memory. The segmentation of an image containing approximately 5500 medically meaningful components (a graph with 5500 vertices), takes approximately 60 seconds when the ensemble size is 100 and 125 seconds when the ensemble size is 250.

4.1.2 Validation

Segmentation performance can be evaluated visually by examining the result of our algorithm with manual segmentations (gold standards) provided by a domain expert. However, the validation based on quantitative measures is also necessary, especially for comparisons. Our algorithm and those that we use in comparisons, are all unsupervised methods. Therefore, they do not report the class labels of regions; they just define the separate pixel groups as regions or clusters. However, we need to assign a label to each region (cluster) since we use the true positive(TP), true negative(TN), false positive(FP), and false negative(FN) pixels for computing sensitivity, specificity, and accuracy. For this purpose, each segmented region is compared with the gold standard and labeled with the class of the region in the gold standard that this segmented region overlaps the most (i.e., with the class of the dominant region in the gold standard). With this definition, the pixels that are in the non-overlapping parts, are considered as false negative or false positive, depending on the class of their dominant regions.

In this thesis, our objective is to separate regions that contain cancerous and non-cancerous glands. Therefore, we do not consider non-glandular regions in the validation scheme. In other words, the pixels in these regions are not considered in TP, FP, TN, and FN computations.

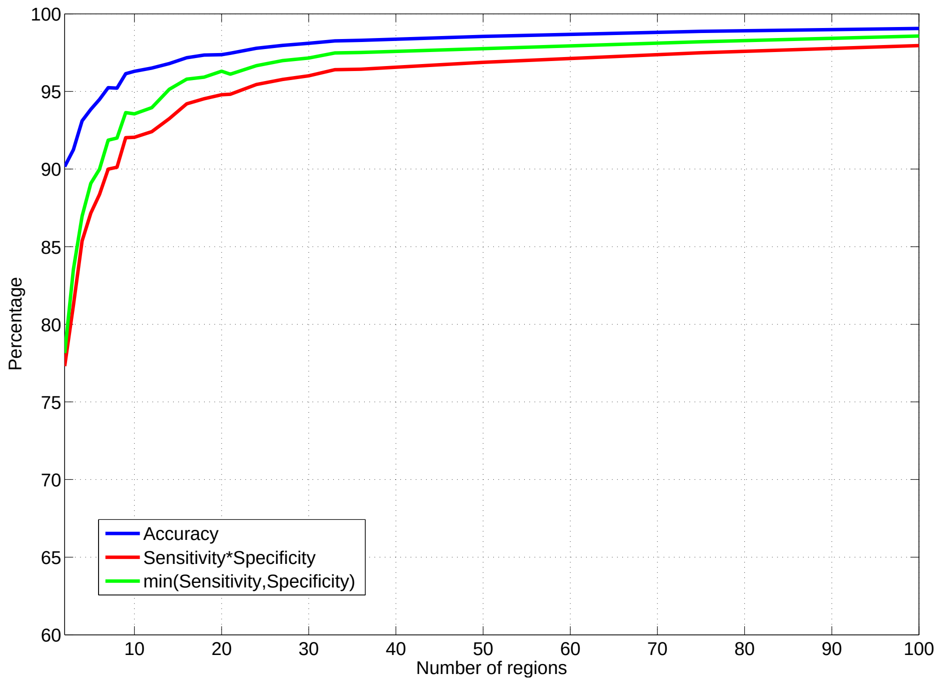


Figure 4.1: Segmentation performance increases as the number of regions increases

For the evaluation, in addition to accuracy, we use sensitivity and specificity. *Sensitivity* measures the rate of actual cancerous pixels that are correctly identified as cancerous pixels. *Specificity* measures the rate of non-cancerous pixels that are correctly identified as non-cancerous pixels. In our experiments, we observe that accuracy for some images could be very high although either sensitivity or specificity is quite low. These images usually correspond to ones that have larger normal regions and smaller cancerous smaller regions, or vice versa. For this reason, we define two more criteria that use sensitivity and specificity. First one is the *multiplication* of sensitivity and specificity and the second one is the *minimum* of sensitivity and specificity.

The aforementioned criteria consider the quality of a segmentation result. However, they do not penalize the over-segmented results. Thus, one should keep in mind that an increase in the number of segmented regions will usually increase the accuracy (Figure 4.1). In a histopathological image, there typically exist 2-3 regions. Therefore, in our experiments, we focus on the results that yield a small number of segmented regions.

4.2 Results

The proposed multilevel cluster ensembling (MLCE) method has three parameters: sub-sampling ratio (*ssRatio*), initial K (*iK*), and ensemble size (*eSize*). In our experiments, we select these parameters on the training set. For that, we consider every combination of the following candidate sets $ssRatio = \{0.1, 0.2, ..., 0.9, 1.0\}$, $eSize = \{1, 2, 5, 10, 15, ..., 450, 500\}$, and $iK = \{2, 3, ..., 99, 100\}$ and select the one that leads to the best performance on the training samples. We separately consider accuracy, sensitivity×specificity, and min(sensitivity,specificity) performance measures. All of them give the same best parameter set, which is ssRatio = 0.1, iK = 15, and eSize = 250.

Table 4.1: The training results obtained by our multilevel cluster ensembling (MLCE) algorithm

Accuracy	Sensitivity	Specifity	Sens×Spec	*min*(Sens,Spec)	♯ of regions
91.52 ±9.13	92.43 ±15.17	87.06 ±24.92	80.03 ±26.82	80.97 ±27.08	2
92.54 ±6.16	94.36 ±9.21	88.23 ±16.02	82.74 ±16.14	84.12 ±15.47	3
93.82 ±5.26	94.92 ±7.35	91.13 ±11.72	85.61 ±12.75	87.45 ±11.94	4
94.75 ±4.54	96.15 ±6.87	91.32 ±9.38	87.33 ±10.86	89.25 ±9.79	5

Table 4.2: The test results obtained by our multilevel cluster ensembling (MLCE) algorithm

Accuracy	Sensitivity	Specifity	Sens×Spec	*min*(Sens,Spec)	♯ of regions
91.40 ±9.24	92.34 ±15.54	86.09 ±25.72	79.10 ±27.91	80.64 ±27.39	2
92.33 ±6.22	93.72 ±9.64	87.76 ±16.54	81.98 ±16.95	83.54 ±16.25	3
93.11 ±5.67	94.21 ±7.96	90.02 ±12.81	84.70 ±13.67	86.73 ±12.65	4
93.96 ±4.95	95.26 ±7.05	90.94 ±9.79	86.62 ±11.27	88.67 ±10.17	5

In the experiments, since we want to avoid over-segmentation, we focus on the results where an image is segmented in to less than 5 regions. Therefore, we fix the number of the final clusters (segmented regions) as 2, 3, 4, and 5. When we report the parameter selection for each of these numbers, the same best parameter set is selected. The training and the test results obtained with this parameter set are reported in Tables 4.1 and 4.2.

In these tables, we observe that the proposed MLCE algorithm yields greater than 91% for all number of selected regions. Moreover, it gives high sensitivity and specificity values at the same time, resulting in high Sens$*$Spec and min(Sens,Spec) values. We will compare these results with other algorithms in the next subsection.

4.2.1 Parameters

In this section, we investigate the effects of parameter selection to the segmentation performance. For that, we fix two of the three parameters and observe the performance as a function of the other.

4.2.1.1 Sub-sampling ratio

In order to generate each clustering result, the proposed method randomly selects a subset of the objects, cluster them, and project the result onto the whole set of objects. Random sub-sampling is effective in increasing the diversity of the ensemble, which improves the final clustering performance. Moreover, it decreases the computational time since the multilevel clustering algorithm works on a less number of objects. The ratio of the selected subset to the entire set is considered as a model parameter.

In the analysis, we consider the ratio from 1 to 100 percent. We also obtain results for different number of segmented regions (for region no: 2, 3, 4, 5) and average those results. Figure 4.2 shows the average of the accuracy, sensitivity$\times$specificity, and min(sensitivity,specificity). This figure shows that even very small ratios (10 percent) are sufficient to obtain good segmentation results. Here we also observe that there is a slight decrease when larger ratios are

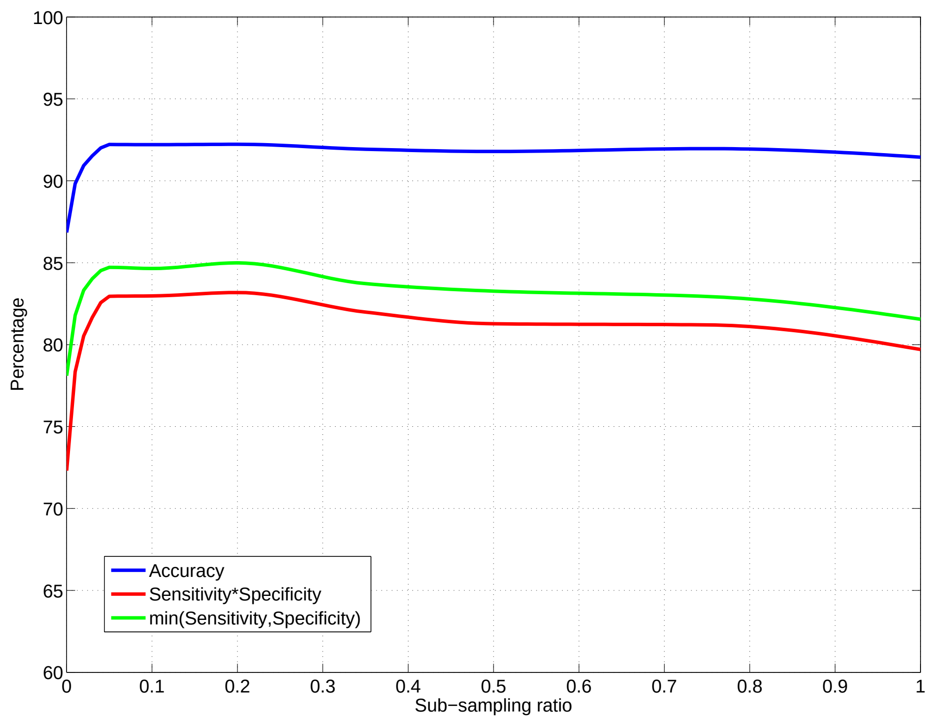


Figure 4.2: The segmentation performance of the test set as a function of the sub-sampling ratio

selected. This is attributed to the decrease in the diversity. The use of smaller values of this parameter leads to more diverse results that are to be ensembled.

4.2.1.2 Initial K

Traditional multilevel graph partitioning algorithms coarsen the original graph until there remain a few hundred vertices and perform an initial partitioning on the coarsest graph. They then perform the refinement on this coarse partition. These algorithms partition the smaller version of the graph for efficiency and the initial partitioning of this small graph should be of good quality. However, in our algorithm, we require different clustering results which increase the diversity. Therefore, we omit the initial partitioning phase. The original graph is coarsened until it has a few vertices instead of few hundred vertices.

The number of vertices until which the coarsening phase partition the graph is considered as another parameter; we call it the initial K. To understand the effect

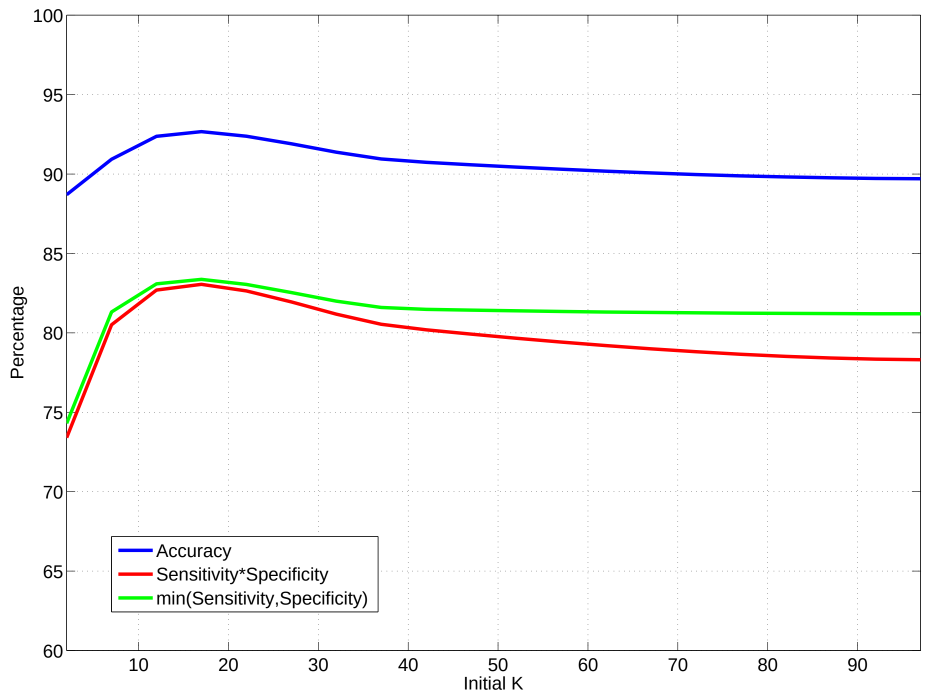


Figure 4.3: The segmentation performance of the test set as a function of the initial K

of this parameter to the performance, we fix the other parameters and select initial K ranging from 2 to 100. Likewise, we conduct the experiment for region no: 2, 3, 4, and 5 and average their results. These results are presented in Figure 4.3. This figure shows that smaller values this parameter yield worse results. We attribute this to the following property: smaller values decrease the degrees of freedom of graph vertices in the refinement phase such that the vertices cannot move to a better part. Larger values of this parameter also decrease the performance, leading to over-segmented results. They also increase the computational time of the algorithm because the number of vertices on the boundaries increase greatly.

4.2.1.3 Ensemble size

The ensemble size ($eSize$) parameter determines the number of clustering results that are to be combined in an ensembling scheme. Fern and Brodley [28] show that ensemble performance can be improved by increasing the ensemble size, provided that individual clusterings are diverse and of high quality. We also see

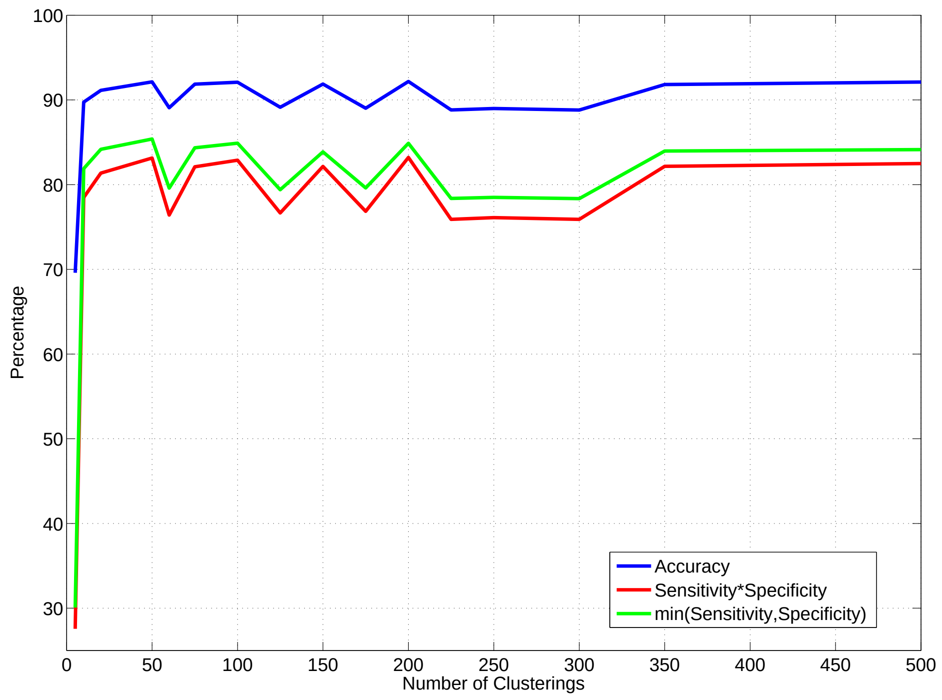


Figure 4.4: The segmentation performance of the test set as a function of the ensemble size

this effect in our experiments. Figure 4.4 reports the average performance results (avg. of reg. no: 2, 3, 4, 5) of this parameter. In this figure, we observe a considerable improvement of the segmentation performance up to an ensemble size of 50. Further increase in the ensemble size improves the performance but it is not stable.

4.2.2 Comparisons

In the experiments, we compare our results with the GraphRLM algorithm [81] and those that are used for comparisons in [81]. We use the GraphRLM algorithm to understand the effect of the proposed segmentation algorithm. Remember that the proposed algorithm and GraphRLM use the same set of features to characterize the objects bur they differ in their segmentation parts. Besides GraphRLM, we make use of four more algorithms. The parameters of these algorithms are also selected on the training set. The details of these algorithms and their parameters are explained in [81].

- GraphRLM is used to understand the effect of the new segmentation algorithm proposed by this thesis. We use the same set of features with GraphRLM. These are textural features defined on the objects that correspond to tissue components. GraphRLM uses a region growing algorithm in its segmentation part.
- GrayRLM [33] is the pixel-based counterpart of GraphRLM. Its features are textural features defined on image pixels. Similarly, it uses a region growing algorithm in its segmentation.
- objectSEG is the algorithm that also uses texture features defined on objects [82]. It also employs a region growing algorithm. It is defined for histopathological images as well.
- JSEG is a well known segmentation algorithm proposed for generic images, not for histopathological images [17]. It relies on a new texture definition called J values. It can be considered in a way that it uses a kind of region growing algorithm.
- GBS is another algorithm also proposed for generic images [27]. It is a graph-based algorithm that construct a graph from the image pixels for segmentation.

The parameter selection of these methods does not rely on finding the parameter combination that gives the best accuracy results. If they consider just the accuracy, they always favour the over-segmented results since the number of segmented regions is automatically determined by these algorithms. Therefore, in the parameter selection, only the parameter combinations that lead to at most 5 segmented regions are considered. The detailed explanation of this parameter selection is explained in [81]. Tables 4.3 and 4.4 report the results of these five algorithms for the training and test sets respectively. Moreover, they also report the number of segmented regions given by these algorithms.

As opposed to these previous algorithms, our proposed algorithm does not determine the number of segmented regions but takes this as an input. In Tables 4.3 and 4.4, we present the results obtained when this number is selected as 2, 3, 4, and 5. These tables show that the proposed algorithm improves sensitivity and specificity results of the other algorithms. The t-test with a significance level of 0.05 shows that this is a statistically significant improvement. Note that, as also discussed in [81], the over-segmentation is an issue for these algorithms. For this reason, in [81], the results are also reported when the maximum number of segmented regions is allowed to be 10. In that case, the accuracy results are shown, to be greatly improved.

For more fair comparison, we also change the algorithms, if that is possible, such that the number of region is given as an input. For the GraphRLM and GrayRLM algorithms, that is easily implemented whereas for the others simple modifications is not possible. For those algorithms more complex modification should be made, which will be considered as future work. Providing the region number as an input, we run the parameter selection of the GraphRLM and GrayRLM algorithms; this selection is still done on the training samples. The training results when the region number is fixed to 2, 3, 4, and 5 are given in Tables 4.5, 4.6, 4.7, and 4.8 respectively. Similarly the test results for these region numbers are given in Tables 4.9, 4.10, 4.11, and 4.12 respectively. For better comparing the algorithms, the test results are also given in bar charts, in Figures 4.5, 4.6, 4.7, and 4.8. The charts indicate that the performance of GraphRLM and GrayRLM increases when the region number is provided as an input. This is related to better selection of the parameters according to a particular region number. These plots also show that the proposed segmentation algorithm improves the results especially for smaller number of regions. We also give the visual segmentation results of some images in Figures 4.9 and 4.10.

Table 4.3: Training results obtained by the algorithms. The results of the proposed MLCE algorithm are reported for region no: 2, 3, 4, 5. For other algorithms, the results obtained when maximum 5 regions are selected.

Method	Accuracy	Sensitivity	Specifity	Sens×Spec	min(Sens,Spec)	♯ of regions
MLCE	91.52 ±9.13	92.43 ±15.17	87.06 ±24.92	80.03 ±26.82	80.97 ±27.08	2
MLCE	92.54 ±6.16	94.36 ±9.21	88.23 ±16.02	82.74 ±16.14	84.12 ±15.47	3
MLCE	93.82 ±5.26	94.92 ±7.35	91.13 ±11.72	85.61 ±12.75	87.45 ±11.94	4
MLCE	94.75 ±4.54	96.15 ±6.87	91.32 ±9.38	87.33 ±10.86	89.25 ±9.79	5
objectSEG ($r < 5$)	81.43 ±14.34	80.07 ±30.46	76.45 ±32.78	57.18 ±33.99	58.23 ±34.27	3.24 ±0.87
GraphRLM ($r < 5$)	87.06 ±13.62	90.72 ±18.69	79.21 ±33.66	70.22 ±33.89	71.75 ±34.41	2.75 ±1.09
GrayRLM ($r < 5$)	77.19 ±14.31	74.37 ±35.80	71.10 ±38.48	45.85 ±36.36	46.49 ±36.84	2.96 ±1.14
GBS ($r < 5$)	72.61 ±8.83	63.55 ±34.87	74.47 ±22.07	41.91 ±24.43	48.08 ±27.77	3.26 ±0.92
JSEG ($r < 5$)	69.03 ±12.28	46.75 ±48.01	72.62 ±40.20	19.55 ±31.54	19.55 ±32.69	2.38 ±1.29

Table 4.4: Test results obtained by the algorithms. The results of the proposed MLCE algorithm are reported for region no: 2, 3, 4, 5. For other algorithms, the results obtained when maximum 5 regions are selected.

Method	Accuracy	Sensitivity	Specifity	Sens×Spec	min(Sens,Spec)	♯ of regions
MLCE	91.40 ±9.24	92.34 ±15.54	86.09 ±25.72	79.10 ±27.91	80.64 ±27.39	2
MLCE	92.33 ±6.22	93.72 ±9.64	87.76 ±16.54	81.98 ±16.95	83.54 ±16.25	3
MLCE	93.11 ±5.67	94.21 ±7.96	90.02 ±12.81	84.70 ±13.67	86.73 ±12.65	4
MLCE	93.96 ±4.95	95.26 ±7.05	90.94 ±9.79	86.62 ±11.27	88.67 ±10.17	5
objectSEG ($r < 5$)	86.91 ±11.44	90.38 ±21.79	77.19 ±28.52	68.09 ±30.42	69.35 ±30.67	4.06 ±1.31
GraphRLM ($r < 5$)	84.78 ±14.42	85.83 ±26.16	76.15 ±35.71	62.73 ±37.50	64.41 ±37.64	2.75 ±1.09
GrayRLM ($r < 5$)	75.67 ±14.48	77.15 ±35.56	62.58 ±40.95	40.22 ±36.34	41.01 ±36.89	3.03 ±1.30
GBS ($r < 5$)	73.43 ±8.96	64.54 ±33.81	72.20 ±30.33	39.63 ±26.14	44.35 ±29.20	3.67 ±1.25
JSEG ($r < 5$)	69.40 ±12.14	62.67 ±45.53	62.24 ±39.78	25.16 ±29.54	25.43 ±29.88	2.96 ±1.32

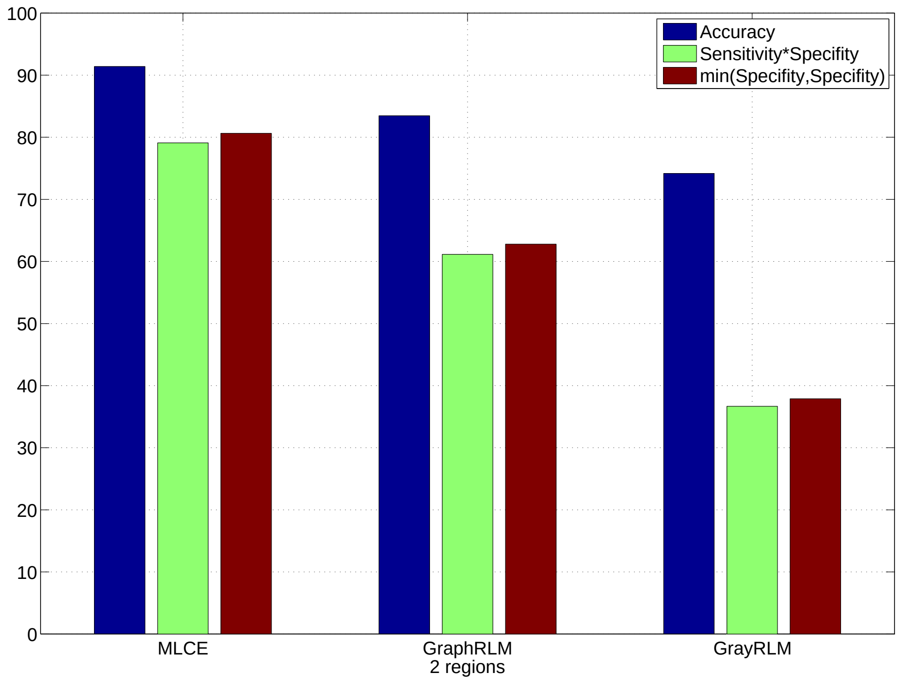


Figure 4.5: Test results of the algorithms when the number of segmented regions is fixed to 2

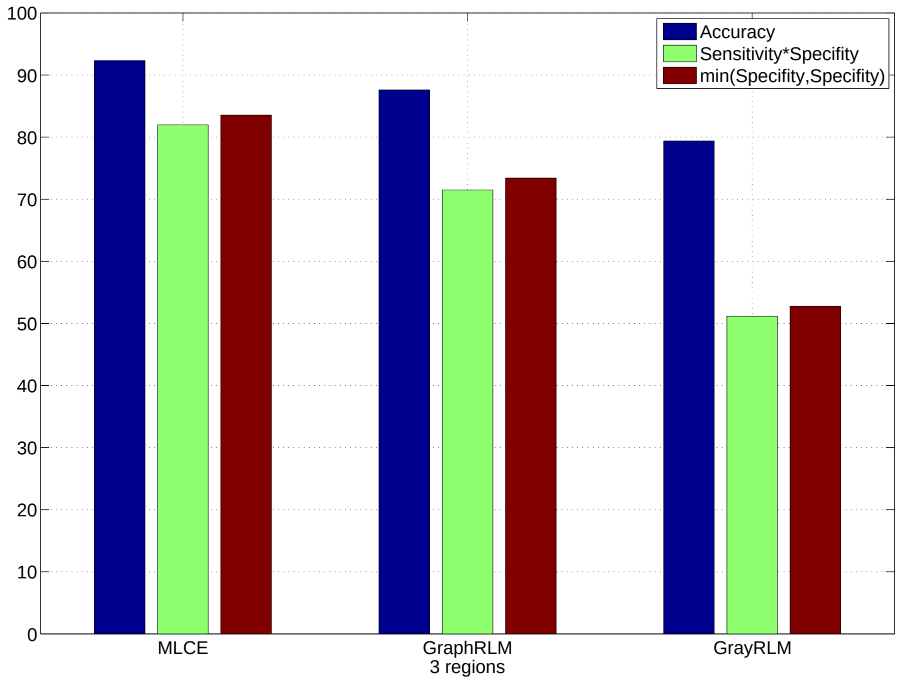


Figure 4.6: Test results of the algorithms when the number of segmented regions is fixed to 3

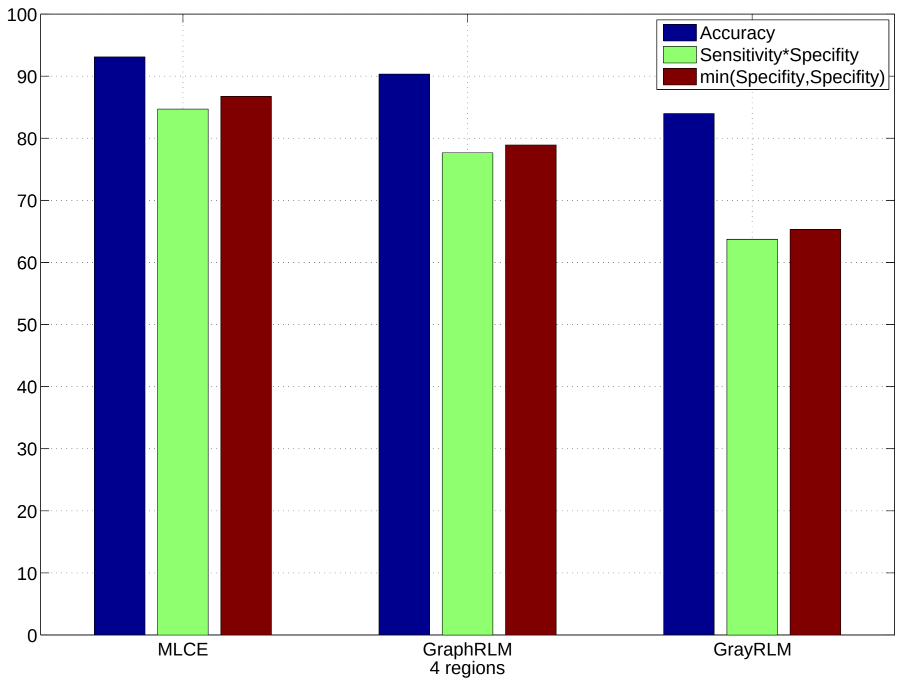


Figure 4.7: Test results of the algorithms when the number of segmented regions is fixed to 4

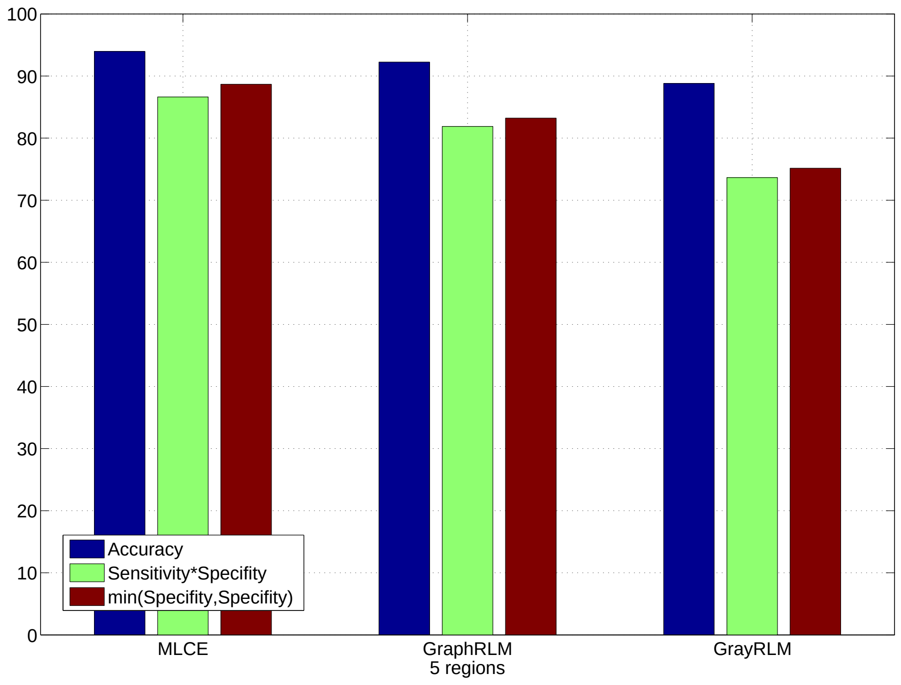


Figure 4.8: Test results of the algorithms when the number of segmented regions is fixed to 5

Table 4.5: Training results of the algorithms when the number of segmented regions is fixed to 2

Method	Accuracy	Sensitivity	Specifity	Sens$*$Spec	min(Sens,Spec)	♯ of regions
MLCE	91.52 $\pm$9.13	92.43 $\pm$15.17	87.06 $\pm$24.92	80.03 $\pm$26.82	80.97 $\pm$27.08	2
GraphRLM	82.62 $\pm$13.03	81.09 $\pm$27.05	78.05 $\pm$32.61	59.78 $\pm$33.84	61.17 $\pm$34.01	2
GrayRLM	75.48 $\pm$14.42	74.03 $\pm$34.92	67.34 $\pm$40.04	42.55 $\pm$36.03	43.76 $\pm$35.99	2

Table 4.6: Training results of the algorithms when the number of segmented regions is fixed to 3

Method	Accuracy	Sensitivity	Specifity	Sens$*$Spec	min(Sens,Spec)	♯ of regions
MLCE	92.54 $\pm$6.16	94.36 $\pm$9.21	88.23 $\pm$16.02	82.74 $\pm$16.14	84.12 $\pm$15.47	3
GraphRLM	88.22 $\pm$10.60	93.14 $\pm$14.67	80.38 $\pm$24.98	73.84 $\pm$24.02	74.84 $\pm$24.51	3
GrayRLM	83.61 $\pm$14.15	83.38 $\pm$28.22	78.47 $\pm$31.56	62.41 $\pm$34.17	63.50 $\pm$33.99	3

Table 4.7: Training results of the algorithms when the number of segmented regions is fixed to 4

Method	Accuracy	Sensitivity	Specifity	Sens$*$Spec	min(Sens,Spec)	♯ of regions
MLCE	93.82 $\pm$5.26	94.92 $\pm$7.35	91.13 $\pm$11.72	85.61 $\pm$12.75	87.45 $\pm$11.94	4
GraphRLM	91.34 $\pm$6.64	90.18 $\pm$13.51	91.21 $\pm$13.49	81.74 $\pm$15.19	83.16 $\pm$14.95	4
GrayRLM	84.91 $\pm$13.54	88.55 $\pm$20.62	76.89 $\pm$32.57	66.17 $\pm$32.63	67.37 $\pm$33.08	4

Table 4.8: Training results of the algorithms when the number of segmented regions is fixed to 5

Method	Accuracy	Sensitivity	Specifity	Sens$*$Spec	min(Sens,Spec)	♯ of regions
MLCE	94.75 $\pm$4.54	96.15 $\pm$6.87	91.32 $\pm$9.38	87.33 $\pm$10.86	89.25 $\pm$9.79	5
GraphRLM	92.75 $\pm$7.71	92.55 $\pm$12.37	92.06 $\pm$12.81	84.84 $\pm$15.20	86.06 $\pm$14.83	5
GrayRLM	89.22 $\pm$11.07	89.94 $\pm$18.81	84.16 $\pm$26.40	74.59 $\pm$28.64	76.09 $\pm$28.85	5

Table 4.9: Test results of the algorithms when the number of segmented regions is fixed to 2

Method	Accuracy	Sensitivity	Specifity	Sens∗Spec	min(Sens,Spec)	♯ of regions
MLCE	91.40 ±9.24	92.34 ±15.54	86.09 ±25.72	79.10 ±27.91	80.64 ±27.39	2
GraphRLM	83.47 ±13.05	81.12 ±27.06	79.13 ±32.62	61.14 ±33.85	62.79 ±34.02	2
GrayRLM	74.17 ±13.42	78.17 ±32.92	57.68 ±42.04	36.67 ±34.03	37.89 ±34.99	2

Table 4.10: Test results of the algorithms when the number of segmented regions is fixed to 3

Method	Accuracy	Sensitivity	Specifity	Sens∗Spec	min(Sens,Spec)	♯ of regions
MLCE	92.33 ±6.22	93.72 ±9.64	87.76 ±16.54	81.98 ±16.95	83.54 ±16.25	3
GraphRLM	87.59 ±10.60	90.44 ±15.67	80.11 ±26.98	71.49 ±27.02	73.40 ±26.51	3
GrayRLM	79.38 ±13.15	80.18 ±28.22	70.19 ±37.56	51.16 ±34.17	52.78 ±34.99	3

Table 4.11: Test results of the algorithms when the number of segmented regions is fixed to 4

Method	Accuracy	Sensitivity	Specifity	Sens∗Spec	min(Sens,Spec)	♯ of regions
MLCE	93.11 ±5.67	94.21 ±7.96	90.02 ±12.81	84.70 ±13.67	86.73 ±12.65	4
GraphRLM	90.34 ±9.64	91.20 ±14.51	85.99 ±25.49	77.67 ±26.19	78.93 ±25.95	4
GrayRLM	83.98 ±11.54	84.70 ±23.62	78.41 ±29.57	63.73 ±29.63	65.30 ±30.08	4

Table 4.12: Test results of the algorithms when the number of segmented regions is fixed to 5

Method	Accuracy	Sensitivity	Specifity	Sens∗Spec	min(Sens,Spec)	♯ of regions
MLCE	93.96 ±4.95	95.26 ±7.05	90.94 ±9.79	86.62 ±11.27	88.67 ±10.17	5
GraphRLM	92.24 ±7.71	93.82 ±10.37	87.67 ±20.81	81.88 ±21.20	83.23 ±20.83	5
GrayRLM	88.79 ±10.07	89.13 ±16.81	84.01 ±26.40	73.64 ±26.64	75.15 ±26.85	5

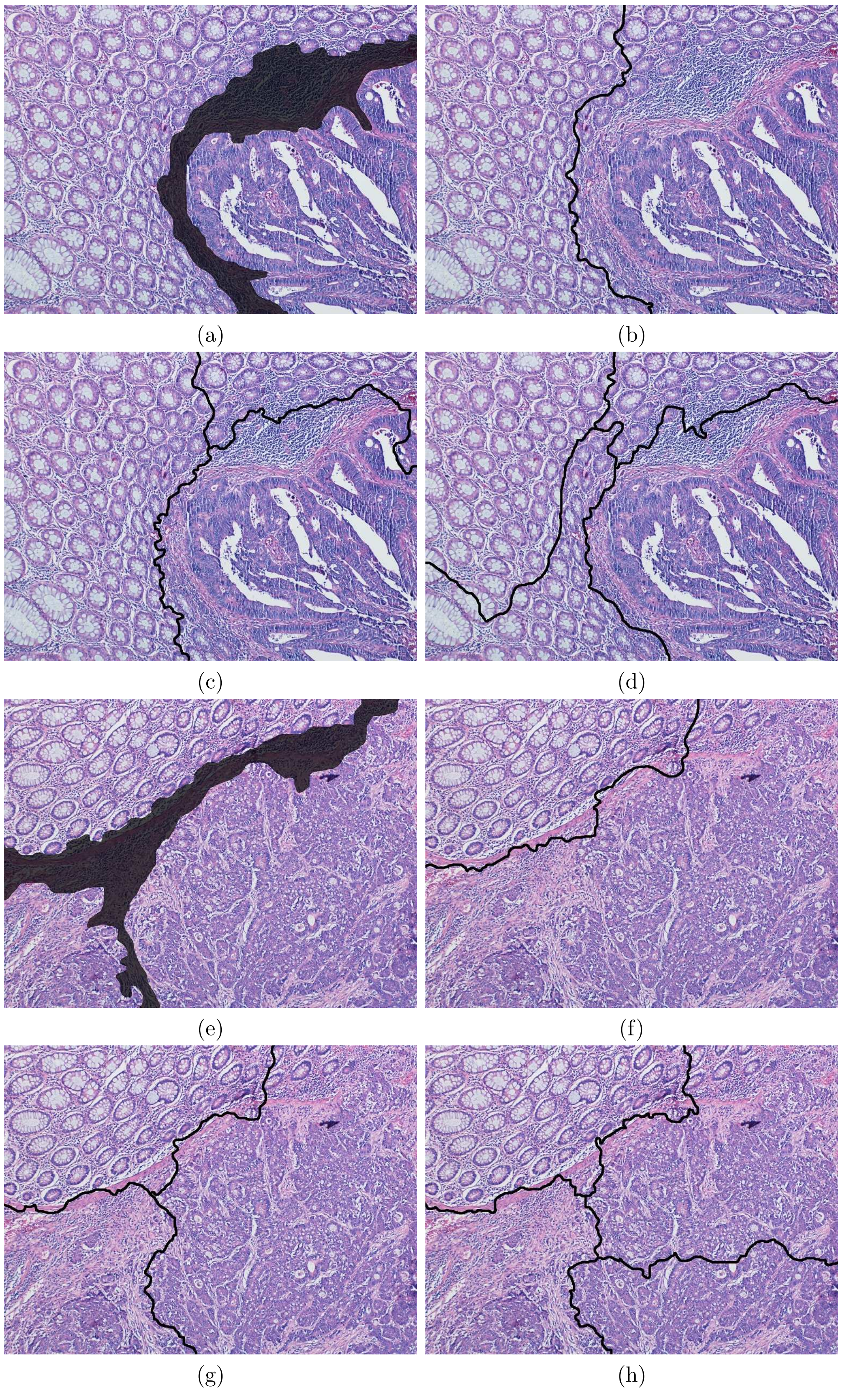


Figure 4.9: Visual results for 2 example images: (a) gold standard for the first image and it is segmented into (b) two, (c) three, (d) four regions. (e) Gold standard for the second image and it is segmented into (f) two, (g) three, (h) four regions

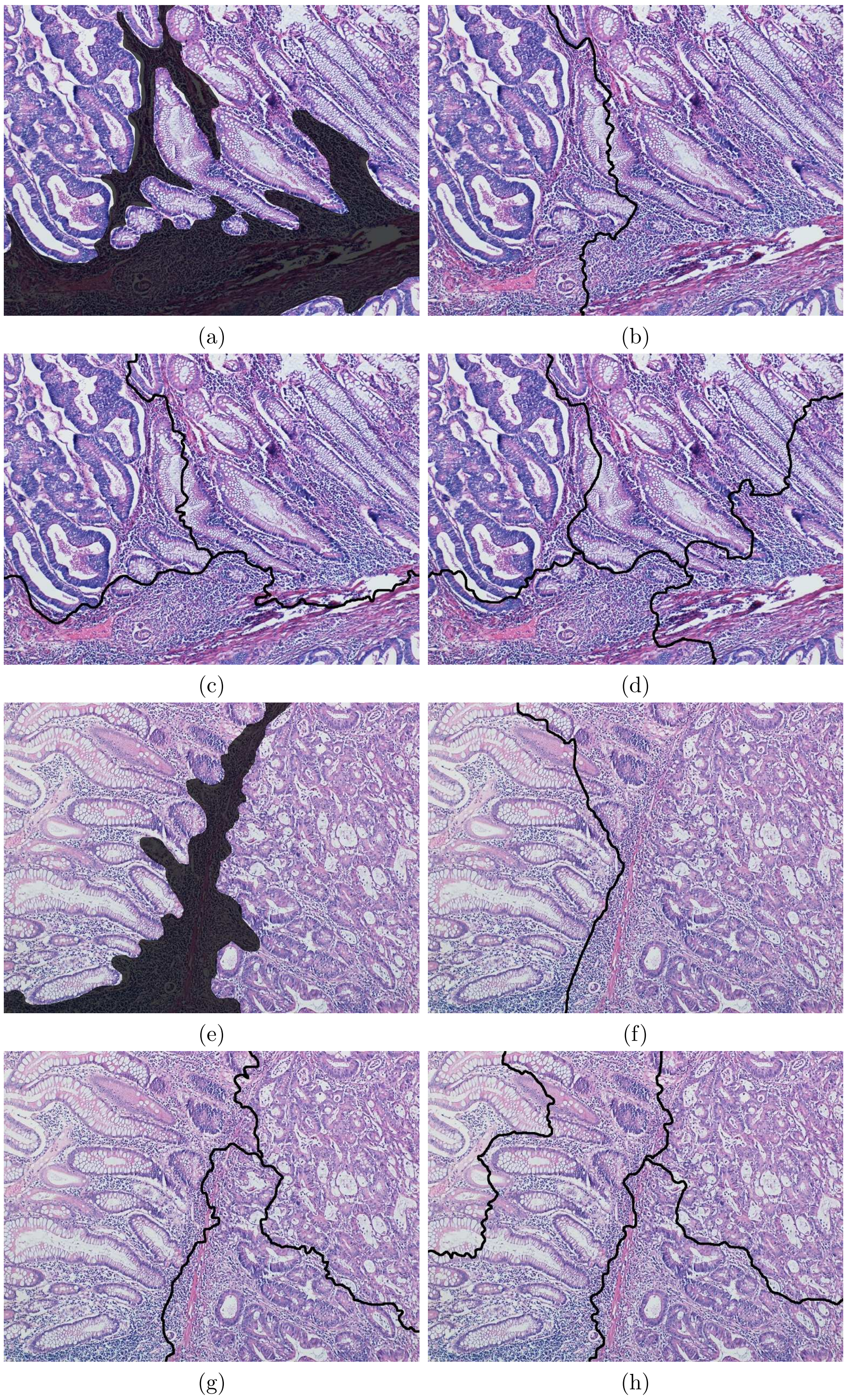


Figure 4.10: Visual results for 2 example images: (a) gold standard for the first image and it is segmented into (b) two, (c) three, (d) four regions. (e) Gold standard for the second image and it is segmented into (f) two, (g) three, (h) four regions

Chapter 5

Conclusion and Discussion

In this thesis, we proposed a new clustering algorithm to be used in segmentation of histopathological images. We use existing high level feature descriptors designed for histopathological images. These feature descriptors have a good representation power because they incorporate the background knowledge of a pathologist into feature extraction by defining image primitives as medically meaningful tissue components. The proposed method focuses on the segmentation part rather than feature extraction. It formulates the image segmentation problem as a clustering problem using the cluster ensembling approach in which different clustering solutions are combined to obtain the final cluster labels. Cluster ensembles produce high quality results when the individual clustering solutions in the ensemble are diverse and accurate. To maximize the diversity and the accuracy of each clusterer, the proposed algorithm formulates the clustering problem as a graph partitioning problem. It modifies the well known multilevel graph partitioning scheme to produce diverse and good quality partitions. Graph partitioning fits very well to the problem since the high level features used in segmentation are graph-based features. After producing many different clusterings of image components, these results are combined using a consensus function producing the final cluster labels of tissue components of an image. Computing Voronoi diagram of components as a post-processing step, it outputs the pixel level segmentation.

We compared our algorithm with five other segmentation algorithms. Some of them are proposed for generic images and some of them are proposed for tissue images. The proposed algorithm performed significantly better than all five algorithms in accuracy, sensitivity, and specificity. Our algorithm used the same feature definition with the GraphRLM algorithm which performed better than other four algorithms. Test results showed that our algorithm improves the segmentation performance significantly even it uses the same features. It also has the lowest standard deviations in validation criteria indicating its stability and generalization power. Another advantage of the proposed algorithm is that it produces high quality segmentations in smaller number of regions overcoming the over-segmentation problem. Because the segmented regions should be large enough to be used in classification and grading of cancer.

Our algorithm expects the number of segments from the outside as a parameter. Most of the time the number of regions or clusters is unknown since this is an unsupervised process. Detection of natural number of clusters or regions in images can be listed as a future work. An increase in the size of ensembles up to a certain point, increases the clustering performance. But there may not be a further significant improvement after that point. Another future work is to develop a method for choosing clustering solutions which will make an actual contribution to the ensemble to increase the clustering performance.

Bibliography

[1] Cancer facts and figures 2009. Technical report, American Cancer Society, Atlanta, GA, 2009.

[2] Cancer prevention and early detection facts and figures 2009. Technical report, American Cancer Society, Atlanta, GA, 2009.

[3] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.

[4] D. Altunbay, C. Cigir, C. Sokmensuer, and C. Gunduz-Demir. Color graphs for automated cancer diagnosis and grading. *IEEE Transactions on Biomedical Engineering*, 57(3):665–674, 2010.

[5] S. Beucher and C. Lantuejoul. Use of watersheds in contour detection. In *International Conference on Image Processing*, 1979.

[6] C. Bilgin, C. Gunduz-Demir, C. Nagi, and B. Yener. Cell-graph mining for breast tissue modeling and classification. In *IEEE International Conference on Engineering in Medicine and Biology Society*, pages 5311–5314, 2007.

[7] Y. Boykov and G. Funka-Lea. Graph cuts and efficient nd image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

[8] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[9] J. Bryant. On the clustering of multidimensional pictorial data. *Pattern Recognition*, 11(2):115–125, 1979.

[10] J. Canny. A computational approach to edge detection. *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, 184(87-116):86, 1987.

[11] U. V. Catalyurek and C. Aykanat. Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. *IEEE Transactions on Parallel and Distributed Systems*, 10(7):673–693, 1999.

[12] W. Chan and K. H. Fu. Value of routine histopathological examination of appendices in Hong Kong. *Journal of Clinical Pathology*, 40(4):429–433, 1987.

[13] T. Q. Chen and Y. Lu. Color image segmentation–an innovative approach. *Pattern Recognition*, 35(2):395–405, 2002.

[14] G. B. Coleman and H. C. Andrews. Image segmentation by clustering. *Proceedings of the IEEE*, 67(5):773–785, 1979.

[15] I. S. Cook and C. E. Fuller. Does histopathological examination of breast reduction specimens affect patient management and clinical follow up? *Journal of Clinical Pathology*, 57(3):286–289, 2004.

[16] J. W. Davenport, J. C. Bezdek, and R. J. Hathaway. Parameter estimation for finite mixture distributions. *Computers & Mathematics with Applications*, 15(10):819–828, 1988.

[17] Y. Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 800–810, 2001.

[18] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1944–1957, 2007.

[19] T. Dietterich. Ensemble methods in machine learning. *Multiple Classifier Systems*, pages 1–15, 2000.

[20] M. F. Dixon, L. J. R. Brown, H. M. Gilmour, A. B. Price, N. C. Smeeton, I. C. Talbot, and G. T. Williams. Observer variation in the assessment of dysplasia in ulcerative colitis. *Histopathology*, 13(4):385–397, 1988.

[21] R. Dobrescu, F. Talos, and C. Vasilescu. Using fractal dimension for cancer diagnosis. In *IEEE International Symposium on Video/Image Processing and Multimedia Communications*, pages 173–176. IEEE, 2002.

[22] S. Doyle, S. Agner, A. Madabhushi, M. Feldman, and J. Tomaszewski. Automated grading of breast cancer histopathology using spectral clustering with textural and architectural image features. In *IEEE International Symposium on Biomedical Imaging*, pages 496–499, 2008.

[23] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*, volume 2. Wiley New York, 2001.

[24] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Cybernetics and Systems*, 3(3):32–57, 1973.

[25] A. N. Esgiar, R. N. G. Naguib, B. S. Sharif, M. K. Bennett, and A. Murray. Microscopic image analysis for quantitative measurement and feature identification of normal and cancerous colonic mucosa. *IEEE Transactions on Information Technology in Biomedicine*, 2(3):197–203, 1998.

[26] A. N. Esgiar, R. N. G. Naguib, B. S. Sharif, M. K. Bennett, and A. Murray. Fractal analysis in the detection of colonic cancer images. *IEEE Transactions on Information Technology in Biomedicine*, 6(1):54–58, 2002.

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

[28] X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proceedings of International Conference on Machine Learning*, volume 20, page 186, 2003.

[29] X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the ACM International Conference on Machine Learning*, page 36. ACM, 2004.

[30] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *Papers on Twenty-five Years of Electronic Design Automation*, pages 241–247. ACM, 1988.

[31] A. H. Fischer, K. A. Jacobson, J. Rose, and R. Zeller. Hematoxylin and eosin staining of tissue and cell sections. *Cold Spring Harbor Protocols*, 2008(6), 2008.

[32] E. W. Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[33] M. M. Galloway. Texture analysis using gray level run lengths. *Computer Graphics and Image Processing*, 4(2):172–179, 1975.

[34] M. Garcia, A. Jemal, E. M. Ward, M. M. Center, Y. Hao, R. L. Siegel, and M. J. Thun. Global cancer facts and figures 2007. Technical report, American Cancer Society, Atlanta, GA, 2007.

[35] J. Gil, H. Wu, and B. Y. Wang. Image analysis and morphometry in the diagnosis of breast cancer. *Microscopy Research and Technique*, 59(2):109–118, 2002.

[36] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Addison-Wesley, 1993.

[37] C. Gunduz, B. Yener, and S. H. Gultekin. The cell graphs of cancer. *Bioinformatics*, 20(1):145–151, 2004.

[38] C. Gunduz, B. Yener, and S. H. Gultekin. Learning the topological properties of brain tumors. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 262–270, 2005.

[39] C. Gunduz-Demir, M. Kandemir, A. B. Tosun, and C. Sokmensuer. Automatic segmentation of colon glands using object-graphs. *Medical Image Analysis*, 14(1):1–12, 2010.

[40] L. Hagen and A. B. Kahng. A new approach to effective circuit clustering. In *Proceedings of the IEEE/ACM International Conference on Computer-aided Design*, pages 422–427. IEEE Computer Society Press, 1992.

[41] B. Hendrickson and R. Leland. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM Journal on Scientific Computing*, 16(2):452–469, 1995.

[42] B. Hendrickson and R. Leland. A multilevel algorithm for partitioning graphs. In *Proceedings of the ACM/IEEE Conference on Supercomputing*. ACM, 1995.

[43] D. R. Hinton, E. Dolan, and A. A. Sima. The value of histopathological examination of surgically removed blood clot in determining the etiology of spontaneous intracerebral hemorrhage. *Stroke*, 15(3):517–520, 1984.

[44] K. Jafari-Khouzani and H. Soltanian-Zadeh. Multiwavelet grading of pathological images of prostate. *IEEE Transactions on Biomedical Engineering*, 50(6):697–704, 2003.

[45] A. Jemal, R. Siegel, and E. Ward. Colorectal cancer facts and figures 2008-2010. Technical report, American Cancer Society, Atlanta, GA, 2008.

[46] A. E. Jones, A. W. Phillips, J. R. Jarvis, and K. Sargen. The value of routine histopathological examination of appendicectomy specimens. *BMC Surgery*, 7(1):17, 2007.

[47] G. Karypis and V. Kumar. Analysis of multilevel graph partitioning. In *Proceedings of the ACM/IEEE Conference on Supercomputing*, page 29. ACM, 1995.

[48] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359, 1999.

[49] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, 49(2):291–307, 1970.

[50] W. Kim, Y. Chen, M. M. Crawford, J. C. Tilton, and J. Ghosh. Multiresolution manifold learning for classification of hyperspectral data. In *IEEE International Geoscience and Remote Sensing Symposium 2007*, pages 3785–3788. IEEE, 2007.

[51] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems*, pages 231–238, 1995.

[52] L. I. Kuncheva and S. T. Hadjitodorov. Using diversity in cluster ensembles. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1214–1219, 2004.

[53] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.

[54] R. Lillie. *Histopathologic Technic and Practical Histochemistry: 3d Ed.* Blakiston Division, Mcgraw-Hill, 1965.

[55] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207:187–217, 1980.

[56] B. Minaei-Bidgoli, A. Topchy, and W. F. Punch. Ensembles of partitions via data resampling. In *Proceedings of the International Conference on Information Technology: Coding and Computing*, page 188. IEEE Computer Society, 2004.

[57] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14: Proceeding of the 2001 Conference*, pages 849–856, 2001.

[58] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 1(9):62–66, 1979.

[59] D. K. Panjwani and G. Healey. Markov random field models for unsupervised segmentation of textured color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(10):939–954, 1995.

[60] D. M. Parkin, F. Bray, J. Ferlay, and P. Pisani. Estimating the world cancer burden: Globocan 2000. *International Journal of Cancer*, 94(2):153–156, 2001.

[61] N. A. Pham, A. Morrison, J. Schwock, S. Aviel-Ronen, V. Iakovlev, M. S. Tsao, J. Ho, and D. W. Hedley. Quantitative image analysis of immunohistochemical stains using a cmyk color model. *Diagnostic Pathology*, 2(1):8, 2007.

[62] A. Pothen, H. D. Simon, and K. P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal of Matrix Analysis and Applications*, 11(3):430–452, 1990.

[63] A. Pothen, H. D. Simon, L. Wang, and S. T. Barnard. Towards a fast implementation of spectral nested dissection. In *Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 42–51. IEEE Computer Society Press, 1992.

[64] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[65] P. Scheunders. A genetic c-means clustering algorithm applied to color image quantization. *Pattern Recognition*, 30(6):859–866, 1997.

[66] O. Sertel, J. Kong, H. Shimada, U. V. Catalyurek, J. H. Saltz, and M. N. Gurcan. Computer-aided prognosis of neuroblastoma on whole-slide images: Classification of stromal development. *Pattern Recognition*, 42(6):1093–1103, 2009.

[67] S. Shafer and T. Kanade. Recursive region segmentation by analysis of histograms. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 1166–1171. IEEE, 1982.

[68] L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice Hall, 2001.

[69] P. S. Shelokar, V. K. Jayaraman, and B. D. Kulkarni. An ant colony approach for clustering. *Analytica Chimica Acta*, 509(2):187–195, 2004.

[70] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[71] J. Smolle. Computer recognition of skin structures using discriminant and cluster analysis. *Skin Research and Technology*, 6(2):58–63, 2000.

[72] P. Spyridonos, P. Ravazoula, D. Cavouras, K. Berberidis, and G. Nikiforidis. Computer-based grading of haematoxylin-eosin stained tissue sections of urinary bladder carcinomas. *Informatics for Health and Social Care*, 26(3):179–190, 2001.

[73] W. N. Street, W. H. Wolberg, and O. L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Proc. Int. Symp. on Electronic Imaging: Science and Technology*, volume 1905, pages 861–870, 1993.

[74] A. Strehl and J. Ghosh. Cluster ensembles – A knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[75] J. Sudbo, R. Marcelpoil, and A. Reith. New algorithms based on the voronoi diagram applied in a pilot study on normal mucosa and carcinomas. *Analytical Cellular Pathology*, 21(2):71–86, 2000.

[76] L. Sun and M. Leng. An effective multi-level algorithm based on simulated annealing for bisecting graph. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 1–12. Springer, 2007.

[77] G. D. Thomas, M. F. Dixon, N. C. Smeeton, and N. S. Williams. Observer variation in the histological grading of rectal carcinoma. *Journal of Clinical Pathology*, 36(4):385–391, 1983.

[78] J. C. Tilton. Analysis of hierarchically related image segmentations. In *IEEE Workshop on Advances in Techniques for Analysis of Remotely Sensed Data*, pages 60–69. IEEE.

[79] J. C. Tilton. Image segmentation by region growing and spectral clustering with a natural convergence criterion. In *IEEE International Geoscience and*

Remote Sensing Symposium Proceedings 1998, volume 4, pages 1766–1768. IEEE, 1998.

[80] A. Topchy, B. Minaei-Bidgoli, A. K. Jain, and W. F. Punch. Adaptive clustering ensembles. *Pattern Recognition*, 1:272–275, 2004.

[81] A. B. Tosun and C. Gunduz-Demir. Graph run-length matrices for histopathological image segmentation. *IEEE Transactions on Medical Imaging*, 30(3):721–732, 2011.

[82] A. B. Tosun, M. Kandemir, C. Sokmensuer, and C. Gunduz-Demir. Object-oriented texture analysis for the unsupervised segmentation of biopsy images for cancer detection. *Pattern Recognition*, 42(6):1104–1112, 2009.

[83] K. Tumer and J. Ghosh. Linear and order statistics combiners for pattern classification. *Combining Artificial Neural Nets*, pages 127–162, 1999.

[84] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.

[85] Y. Wang, D. Crookes, O. S. Eldin, S. Wang, P. Hamilton, and J. Diamond. Assisted diagnosis of cervical intraepithelial neoplasia (cin). *IEEE Journal of Selected Topics in Signal Processing*, 3(1):112–121, 2009.

[86] B. Weyn, G. V. de Wouwer, G. K. Samir, A. V. Daele, P. Scheunders, E. V. Marck, and W. Jacob. Computer-assisted differential diagnosis of malignant mesothelioma based on syntactic structure analysis. *Cytometry*, 35(1):23–29, 1999.

[87] B. Weyn, G. van de Wouver, M. Koprowski, A. van Daele, K. Dhaene, P. Scheunders, W. Jacob, and E. van Marck. Value of morphometry, texture analysis, densitometry, and histometry in the differential diagnosis and prognosis of malignant mesothelioma. *The Journal of Pathology*, 189(4):581–589, 1999.

[88] W. H. Wolberg, W. N. Street, D. M. Heisey, and O. L. Mangasarian. Computer-derived nuclear features distinguish malignant from benign breast cytology. *Human Pathology*, 26(7):792–796, 1995.

[89] S. X. Yu and J. Shi. Multiclass spectral clustering. In *IEEE International Conference on Computer Vision*, volume 1, pages 313–319, 2003.

[90] S. C. Zhu and A. Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996.