M.Sc. Thesis
Master of Science in Engineering

DTU Compute
Department of Applied Mathematics and Computer Science

Support Vector Machines for Pixel Classification

Application in Microscopy Images

Jakob Busk Sørensen (s061356)

Kongens Lyngby 2014

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Matematiktorvet, building 303B,
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Short contents

Summary

Visiopharm currently uses *Bayesian classification* and *K-means clustering* for pixel classification in their software. The purpose of this project was to investigate if *Support Vector Machines* (SVMs) would be a good additional classifier. It will be shown that a quantitative improvement (increase in accuracy) is indeed possible compared to existing methods, but that this is not the only thing to take into consideration. Overall SVMs does seem like a good addition to Visiopharms software, but more projects should follow this, to answer some of the new questions which this project has raised.

Preface

This project was carried out in collaboration with Technical University of Denmark (DTU) and Visiopharm A/S and is written as a master's thesis. It balances academic research and functional use of the theory. The project is worth 30 points on the ECTS scale.

Kongens Lyngby, June 19, 2014

J. Busk

Jakob Busk Sørensen (s061356)

Acknowledgements

A huge thank you goes out to everyone who made this project possible. To Visiopharm for offering the project as well as providing data, facilities and guidance. To Professor Lars Kai Hansen, my supervisor at the project and also to the Technical University of Denmark, where I have enjoyed five great years. And finally to family and friends for proofreading this report and for supporting me throughout the project.

Figure 1: The lobby at the Technical University of Denmark

Contents

CHAPTER 1

Introduction

1.1 Motivation

At some point, everyone has probably fantasized of a robot or computer, taking over the trivial tasks of their job. This is likely also the case when it comes to highly educated specialists who spend hours looking into a microscope (or at a computer monitor), in order to perform manual analysis of a tissue sample. Fortunately this is one trivial task which can actually be performed by a computer, though making a computer performing the task is anything but trivial.

One of the non-trivial parts, is the pixel classification, which acts as a simplification of the image, to ease the remaining of the automation. Hence to improve the outcome of this step will impact the analysis from beginning to end, making this small task, a task of great importance.

Figure 1.1: Specialists are very skilful when it comes to manual analysis. However these specialists are often very busy, so any task of theirs which can be automated, will free their time for other assignments.

1.2 Problem Description

Current standpoint

Visopmorph is a module in *Visiopharm Integrator System* (VIS), a software which is used in both hospitals and laboratories for automatic analysis of microscopy images. One of the tasks, and the first step in the automatic image analysis, is a pixelwise classification, where each pixel in the image is assigned to a class. This simplifies the image tremendously, making further image analysis possible (morphology and calculations [25]). Currently this classification is done primarily using *Bayesian classification* or *K-Means clustering* [25]. Despite being rather effective, these methods are often beaten by more flexible classifiers, such as *Support Vector Machine (SVM) classification*. This to a degree which has lead to *Bayesian classification* being referred to as the "favourite punching bag of new classification techniques" [15]. In Fig. 1.2 an example is shown of an image, which has been classified using Visiomorph's K-Means clustering.

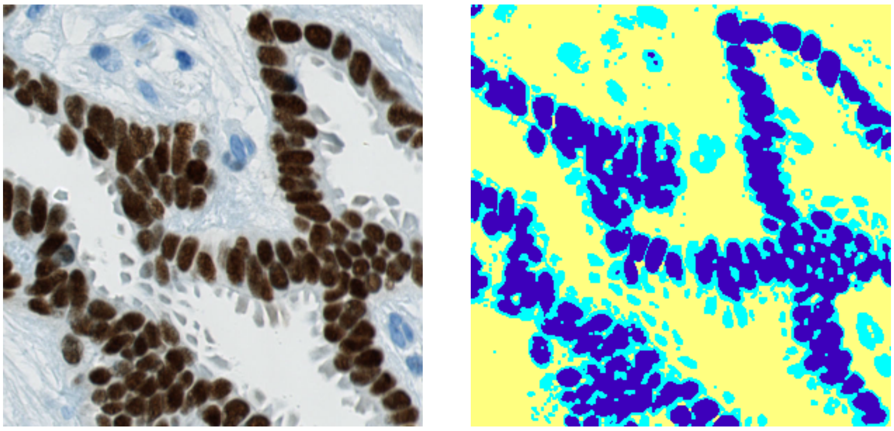

(a) Before Classification (b) After Classification

Figure 1.2: An example of classification, where the two different nuclei has been separated from the background. The reduction to only three colors (classes), eases the remaining process.

The First Step Towards SVM

While the term "favourite punching bag" might not be completely fair, it is a reasonable hypothesis, that SVMs (at least in certain areas) can be an improvement over the existing methods. It is the purpose of this project to act as a *proof of concept* study, exploring the viability of implementing an SVM classifier in Visiomorph. Hence the primary question to be answered is *if* the implementation is possible. Only to a lesser degree will the question of *how* to implement it, be answered. In other words this project can be considered the first in a series of steps, towards the use of SVM in Visiomorph. Depending on the outcome of the project, Visiopharm may chose to take additional steps in the shape of follow-up projects, or not to.

1.3 Focus Areas

SVMs is a broad topic, with many subtopics. Some subtopics are general, but most are to some degree dependant on the subject of classification, or the choice of settings for the SVM. For this project, focus will be on the subtopics which is of relevance to the implementation in Visiomorph.

Implementation with LibSVM

LibSVM is an open source library for SVMs. It is written and maintained by Chih-Chung Chang and Chih-Jen Lin from National Taiwan University [4]. LibSVM was created back in the year 2000, so it is a very well tested library. This should make it ideal for a quick implementation of an SVM classifier in Visiomorph. The first task of this project will be to make a basic SVM classifier in MATLAB, which uses the LibSVM library.

Radial Basis Function Parameters

The kernel trick is what makes it possible to do non-linear classification with SVMs, depending on the choice of kernel. For this project is the *Radial Basis Function* (RBF) is used. This is the most widely used kernel in field of SVMs. This kernel has two free variables (hyper parameters), often notated as γ and C. These will be described in greater details in the theory chapter, but briefly γ can be described as the kernel width whereas C as the penalty term for the error [3]. The value for γ and C can have great influence on the accuracy of the SVM. But as they are often calculated by the use of brute force, it is advantageous to obtain some sort of prior knowledge of their influence. This may in best case lead to a fixed value of the γ and C. But even a reduction of the 2-dimensional space they form can be valuable.

Viability of Implementation

Even if it turns out that the use of SVMs can improve the accuracy, this is no guarantee for success. There are variables which should be examined, to determine if implementation in Visiomorph, is viable. Besides an improvement in accuracy, for at least some data, it also requires reasonable computation time as well as the implementation should be possible with the resources available to Visiopharm. The latter can be difficult to prove until tested (which is not a part of this project), but an estimate of the resource requirements should be made.

CHAPTER 2

Theory of Support Vector Machines

As mentioned this project is using an existing implementation of SVMs (LibSVM), which means that strictly speaking only little knowledge of SVMs are required. However a basic understanding of the SVM theory, will increase the chances of achieving this goal. Much like having a basic knowledge of car mechanics, will increase a race drivers chance of winning the race.

The theory in this report will cover the basics of SVMs. For a greater insight in the theory, it is recommended to read *Pattern Recognition and Machine Learning by Christopher M. Bishop* [1].

2.1 The Classification Problem - Separable Classes

SVMs are so called *maximum margin classifiers* [1]. To explain this we will consider a simple example of two linearly separable clusters in a two-dimensional space, as shown in Fig. 2.1. While there exists an infinite amount of decision boundaries which will correctly classify the data, most people would agree that the solution in Fig. 2.1c intuitively is the most correct.

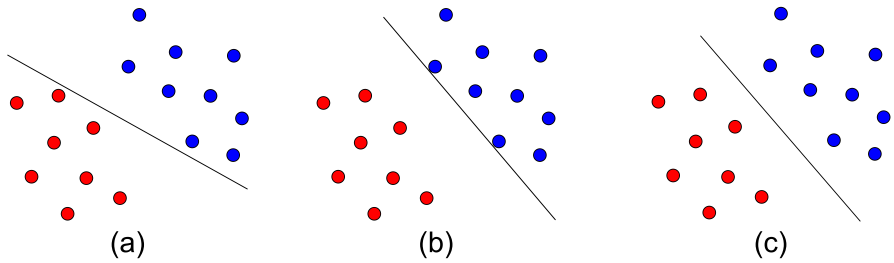


Figure 2.1: While all three classification boundaries are valid, only the boundary in (c) will maximize the margin (illustration of the margin is shown in Fig. 2.2). This is what SVMs does.

This also happens to be the solution, which maximizes the margin. The margin is the distance distance between the black and the gray lines in Fig. 2.2. To understand how this is done, consider the equation for a hyperplane which is given as [19]

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{2.1}$$

Where $\mathbf{w}$ is a non-zero vector normal to the hyperplane, $\mathbf{x}$ is any point in the same space as the hyperplane and b is a scalar. This means that the equation remains identical, no matter the dimensionality. Now consider two additional hyperplanes, which are canonical (normalized), given as

$$\mathbf{w} \cdot \mathbf{x} + b = \pm 1 \tag{2.2}$$

These hyperplanes are shown in Fig. 2.2 as the dashed gray lines. The distance from any given point $\mathbf{x}_i$ is known to be

$$d(\mathbf{w}, b; \mathbf{x}_i) = \frac{|\mathbf{w} \cdot \mathbf{x}_i + b|}{||\mathbf{w}||} \tag{2.3}$$

The support vectors are defined as the points on the canonical hyper plane described in Eq. 2.2, which means that the numerator in Eq. 2.3 can be set to 1. Points which are not support vectors have no influence on the classification. This means that the expression in Eq. 2.3 can be simplified to

$$d(\mathbf{w}, b; |\mathbf{w} \cdot \mathbf{x} + b| = 1) = \frac{1}{||\mathbf{w}||} \tag{2.4}$$

Remembering that this distance is the margin, we need to maximize this term, in order to maximize the distance.

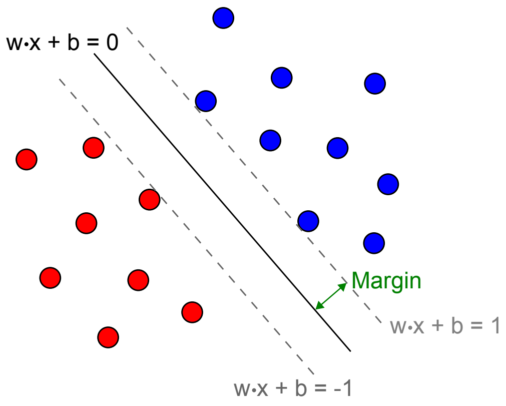


Figure 2.2: The hyperplane separating two classes, and its equation. Additionally two canonical hyperplanes are showed (the dashed lines) with equations. The distance from these hyperplanes to the separating hyperplane, is the margin.

Maximizing the term in Eq. 2.4 is not a very friendly optimization problem. However maximizing $||\mathbf{w}||^{-1}$ is the same as minimizing $||\mathbf{w}||^2$ [1]. So the new optimization problem becomes

$$\min_{\mathbf{w},b}\left(\frac{1}{2}||\mathbf{w}||^2\right) \tag{2.5}$$

The constant 1/2 is added for later convenience [1], when the optimization problem is to be solved. Being a constant scalar it does not affect the optimization problem. Furthermore the optimization problem is subject to the constraint [19]

$$y(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1, \forall i \in [1, m] \tag{2.6}$$

Or in other words, the distance from any given point $\mathbf{x}_i$ to the hyperplane, must be equal to or greater than one. This makes sense, as without constraints $\mathbf{w} \to \mathbf{0}$ would always be the optimal solution.

The margin hyper planes are defined only by the few point which lie on them (the *support sectors*). Only a change in the support vectors will lead to a change in the classification, all other points are indifferent to the classification.

2.2 Overlapping Classes

The above method for solving the classification problem, only works if the classes are perfectly separable. An example of classes than cannot be completely separated, can be seen in Fig. 2.3, where four points are outside their respective margin boundaries. In this case no hyperplane would be able to separate the classes completely.

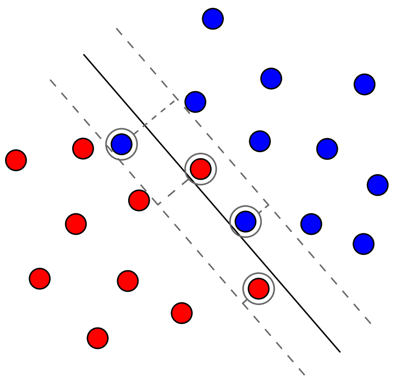


Figure 2.3: Classes can, for many reasons, not always be completely separated. In SVMs, this issue is dealt with by adding slack. The slack allows certain points to lie outside their classes margin hyperplane. These points are marked by a gray circle and a line connecting them to their respective hyperplane. As two of the points are on the "wrong" side of the hyperplane, these points will be classified incorrectly.

For points outside their margin boundary, there are two options. If they are still on the correct side of the separating hyperplane, they will be classified correctly, but still increase the penalty term. If they are on the wrong side of the separating hyperplane (e.g. a blue point in the red class) they will be classified incorrectly and add to the penalty term as well. The penalty term, is a term added to Eq. 2.5, allowing some slack for non-separable classification problems. With the penalty term the optimization problem becomes [19]

$$\min_{\mathbf{w},b}\left(\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i\right) \tag{2.7}$$

Where C is a constant which controls the trade-off between margin maximization and error minimization [5], and $\xi_i \geq 0$ is a function of the the distance from the margin hyperplane to the points which has slack. If the points are between their margin hyperplane and the separating hyperplane then $0 < \xi_i \leq 1$. If the points are on the incorrect side of the separating hyperplane then $1 < \xi_i$ [1]. The function that ξ_i describes is sometimes referred to as the hinge loss function. The type of function which is used for hinge loss may vary [19], but common for all of them is, that ξ_i increases with the slack, and that $\xi_i = 0$ when there is no slack (i.e. the points are inside their margin boundary).

The hyper parameter C is a variable scalar which has a large influence on the model. For very large values of C, any error will have a dramatic effect on Eq. 2.7, and for the extreme case of $C \to \infty$ we will end up with the term in Eq. 2.5 as no error will be tolerated. For very low values of C, errors are almost ignored causing the model to be very prone to misclassification.

2.3 The Kernel Trick

Sometimes linear classification is not a possibility, not even with a fair amount of slack. This issue can be handled with something called a kernel trick [1]. The kernel trick basically takes all the points and map them into a higher dimensional space. To do so, a kernel K is defined such that points $\mathbf{x}$ and $\mathbf{x}'$ have a kernel value $K(\mathbf{x}, \mathbf{x}')$ which is equal to an inner product of $\Phi(\mathbf{x})$ and $\Phi(\mathbf{x}')$ [19]. That is

$$K(\mathbf{x}, \mathbf{x}') = (\Phi(\mathbf{x}), \Phi(\mathbf{x}')) \tag{2.8}$$

In practice this means that instead of using $\mathbf{x}$ in Eq. 2.7, we will use $\Phi(\mathbf{x})$. In this project we will use a RBF kernel, sometimes also known as a Gaussian kernel. It is described in greater details in the next section, but it can be thought of simply as a transformation of $\mathbf{x}$ into an infinite dimensional space, allowing the linear classification which is the basis of SVMs.

2.4 The Radial Basis Function

Whilst there theoretically exists infinitely many kernels which can be used in SVMs, we will only be using the RBF in this project. It a very commonly used kernel and resembles a Gaussian function [19]. It is given as

$$K(\mathbf{x}, \mathbf{x}') = e^{-\frac{||\mathbf{x}'-\mathbf{x}||^2}{2\sigma^2}} \tag{2.9}$$

However in some cases, including *LibSVM*, the $1/2\sigma^2$ is referred to as γ which is then inversely proportional to σ^2 [4]. This will change the kernel equation to be

$$K(\mathbf{x}, \mathbf{x}') = e^{-\gamma||\mathbf{x}'-\mathbf{x}||^2} \tag{2.10}$$

The value of γ is inversely proportional to the squared width of the Gaussian function, as it is seen if Fig. 2.4. We will use this knowledge later, when we have to select the value range, in which we search for the optimal value of γ.

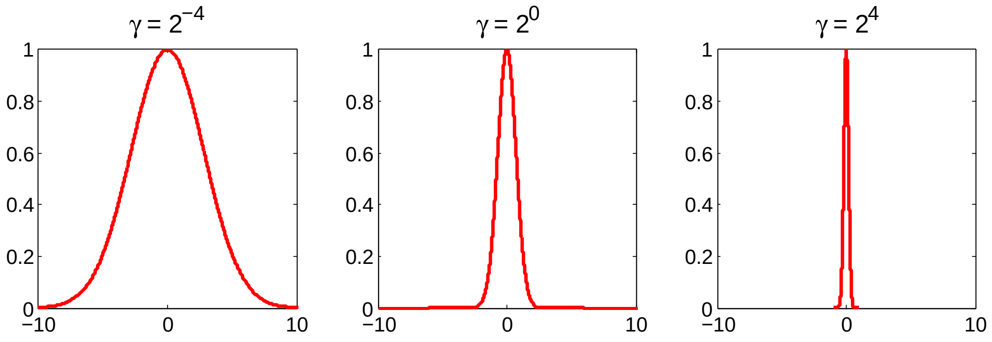


Figure 2.4: The RBF shown in one dimension, for three different values of γ. The larger the value of γ, the narrower the RBF becomes.

The use of the radial basis function also means that we will have another free variable, γ, in addition to C. So the SVM used in this project will be a function of (C, γ) [4].

2.5 Accuracy Estimation

One method of testing accuracy (or error) in machine learning, is cross-validation [1]. It is the method used in this project, and it is important to the understanding of the accuracy estimates, which will be presented in chapter 5. Cross-validation is an enhanced version of validation, so in order to understand cross-validation, one must first understand basic validation.

Validation

In order to do validation, some data is needed for which the correct result is known. This is also what defines training data, which means that the training data can be used for the validation. Let us denote the full set of training data D. We then split D into two smaller sets, D_{Train} and $D_{Validate}$. Now only D_{Train} is used for the training (the process of using labelled data to create a model used to classify the unlabelled data), which gives a model f^{-}. The minus indicates that the model is not complete, as it is made with only part of the training data. This model, f^{-}, is then used to classify the remaining data, $D_{Validate}$. This classification is then compared to the true result, to create a measure of the accuracy. This process is also shown in Fig. 2.5.

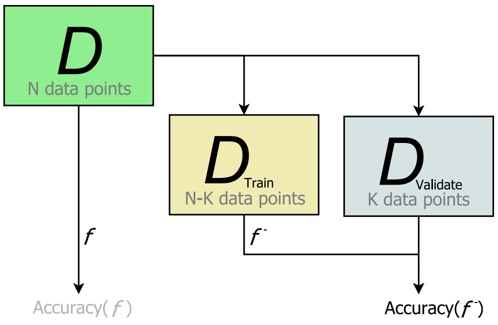


Figure 2.5: The complete known data denoted D is split into two smaller sets denoted D_{Train} and $D_{Validate}$. The training set D_{Train}, is used to train the model f^{-}, and the remaining data $D_{Validate}$, is then classified using that model. As the result is already known for this data, an estimate of the accuracy can be made. Only the accuracy of f^{-} can be calculated, so this is used as an estimate of the accuracy of the full model f.

The accuracy is given as the fraction of points which have been correctly classified. Hence if the accuracy is denoted $A(f^{-})$, it is given as

$$A(f^{-}) = \frac{N_{correct}}{N_{total}} \tag{2.11}$$

Where $A(f^-)$ is the accuracy of the reduced model, $N_{correct}$ is the number of correctly classified points in the validation set and N_{total} is the total number of points in the validation set. It is important to notice that $A(f^-)$ is only an *estimate* of the accuracy, since only a partial model, f^- is used. This means that a leap of faith is taken, when returning from the validation and back to using the full data set (D) to train the full model f.

Besides having a reasonable training set, a ratio between the data used for training and validation also has to be decided. A good rule of thumb, is that this ratio should be approximately 80/20. I.e. the part of the data used for validation, should be around one fifth of the total training data [8].

Cross-Validation

Intuitively a good argument against the validation process discussed in chapter 2.5, would be the risk of getting an unlucky split of D. Cross-validation can solve this, with the small cost of additional computation time. In cross-validation the data is split into M equally sized subsets [1]. One of the subsets acts as the validation set, $D_{Validate}$, while the remaining subsets are used as one set, equivalent of D_{Train}. In the next iteration a different subset is used as $D_{Validate}$, and the rest as D_{Train}. This process continues until M accuracy estimates have been calculated. The mean value of these estimates is then used as the combined estimate (cross-validation estimate).

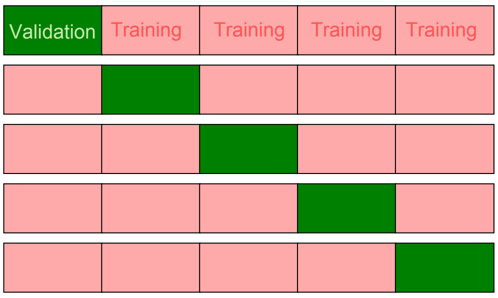


Figure 2.6: An example of 5-fold cross-validation. The complete data is split into five equally sized subsets. Four of the subsets (red) are used as training, while the last subset (green) is used for validation. This is done five times, such that all five subsets are eventually used as validation set. The mean value of the individual accuracy estimates, is the cross-validation estimate of the accuracy.

In Fig. 2.6 the split of the data is shown, where $M = 5$. Since each of the subsets has to be used for validation, the training has to be done equally many times. This means that the computation time for the cross-validation is slightly less[1] than M times longer than the basic validation (five times longer in Fig. 2.6).

[1]Since each training set is only 4/5 of the total data, each training will be slightly faster.

2.6 State of the Art

Similar Experiments

The most common methods for pixel classification in microscopy images, is probably *Bayesian classification* and *K-Means clustering*. The same methods which already exists in Visiomorph. The reason for this could be that these are very decent classifier, as it has been shown by *Khutlang et al.* in 2010. Reaching close to 90% accuracy on the pixel classification proved sufficient for the further process of automatic screening for *mycobacterial tuberculosis* [11]. Furthermore there are ways to improve these simple classifiers, which might be quicker than implementing a new type of classifier. An example of this is proven by *Lezoray* and *Cardot* back in 2002. By simply changing the colorspace (i.e. changing the features) they noticeably increased the accuracy of both the *Bayesian classifier* and the *K-Means clustering* [16].

Despite not be the most commonly used method in pixel classification, SVMs have been used for the purpose. For example in 2007 an article was written by *Lenseigne et al.* on SVMs for automatic detection of tuberculosis [14]. Using a very simple approach, only the green band is used as a feature and the final result is given as the percentage of pixels classified as bacilli. If the percentage reaches a certain threshold, then tuberculosis is considered present. The conclusion of the article is, that even the simple approach used, outperformed existing methods such as *direct fluorescence measure*.

In another article from 2013, *Giannakeas et al.* describes how SVMs can be used for *segmentation of microarray images* [7]. While this is a slightly different issue than the one in this project, it is not irrelevant as Visiopharm also work with microarray images. In the article they use a three class SVM to separate *background*, *signal* and *artefacts*. Besides concluding that SVMs can indeed be used for the desired segmentation, the results from the article also indicates that SVM is a more accurate and more stable classifier than both *Bayesian classification* and *K-Means clustering*, which are the methods currently used in Visiomorph [25].

Semi-supervised Classification

Another possible use of SVM is for *semi-supervised classification*, sometimes also referred to as S^{3}VM (*Semi-Supervised Support Vector Macine*). One way of doing this is by selecting the classes by unsupervised clustering, rather than manually. This approach has been used in an article from 2010 on *color image segmentation* [26]. The training data is selected all together, after which it is clustered using *Fuzzy C-Means Clustering* (FCM). FCM is a clustering method very similar to K-Means, with the main difference being that FCM provides a membership value (a scalar indicating how likely it is that the data point belongs to a certain class) to each data point, whereas K-Means only provides the class a data point belongs to [17]. The data points with high membership (those close to the cluster center) is then used as training for the SVM. Being a maximum margin classifier, the decision boundary will be placed with maximum margin between the cluster centres. A flow diagram of the semi-supervised process is shown in Fig. 2.7.

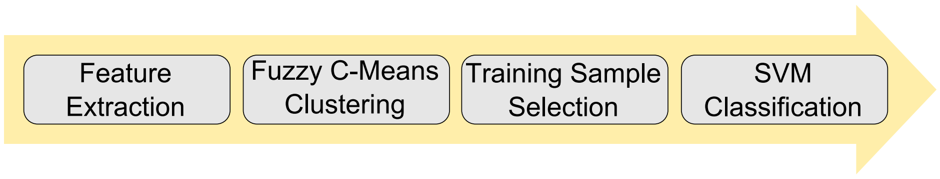


Figure 2.7: Fuzzy C-Means clustering can group unlabelled data, and score it depending on its membership to different groups. Points with high membership to groups are used as training points for that group, training points which are then used in the SVM. This can either be used as an addition to labelled points, or as the only training data.

Another article from 2010 describes the use of S^{3}VM for *pixel classification in remote sensing imagery* [18]. Despite being a different type of imagery, than what this project is about, the pixel classification problem is remarkably similar (without spatial context the complexity of an image is greatly reduced). Furthermore the article also describes how to use an *ensemble approach*, where multiple SVMs are trained and majority voting is used to decide the class. They compare the results with a conventional (supervised) SVM, with the conclusion that S^{3}VM can provide a noticeable increase in accuracy, and even greater increase when using the ensemble approach.

Neighbourhood Pixels as Features

Besides good classification results, the article on *automatic detection of tuberculosis* [14] also describes an interesting aspect of how to think of features. In Visiomorph only values directly related to the pixel is being used as features (filters can add some additional context). However in the article they present an approach where values from the 8 nearest neighbour pixels are also used as features. This allows for some spatial information, relative to the surrounding pixels and may help mitigate classification errors caused by noise artefacts.

CHAPTER 3

Data Acquisition & Structure

This chapter deals with data acquisition. It will describe the process starting when the data is still tissue on a glass plate, and all the way to the point where it is features which can be used as input in the SVM (or any other machine learning method). The data acquisition can be boiled down to a three step process, which is shown in Fig. 3.1.

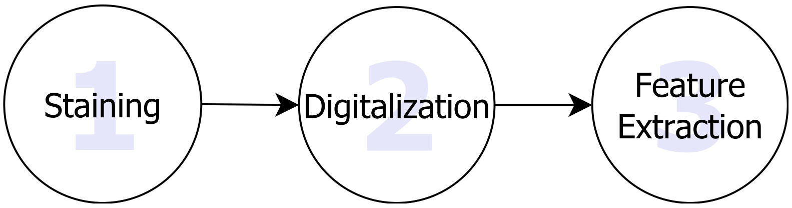
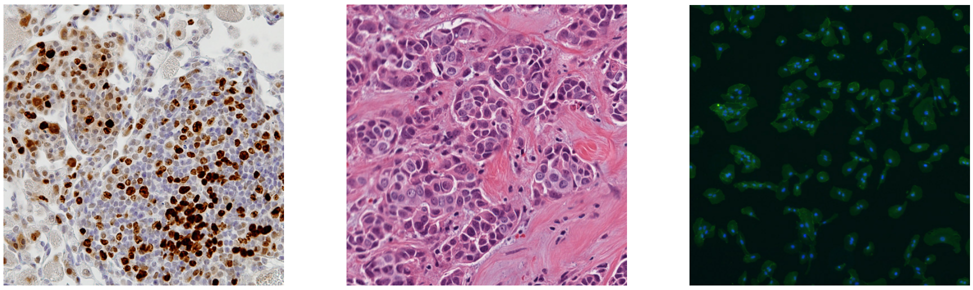


Figure 3.1: Obtaining the data is a three step process. First the Tissue samples are stained to enhance contrasts, second the data is digitalized as an image and finally features are extracted from that image.

The data used in this project is already digital images, meaning that it has passed the first two steps in Fig. 3.1. However it is still important to understand these steps, as they are the foundation of the data dealt with in the project.

3.1 Staining

Tissue samples have to be stained, in order for microscopy images to be useful in image analysis. In this project we will focus on images stained by three different methods. Examples from the three methods are shown in Fig. 3.2. The next three sections describes briefly how each method works and why the images looks so different, depending on staining method.



(a) Ki-67 Immunostaining (b) H&E Staining (c) Fluorescence lighting

Figure 3.2: The three different image staining techniques, used to generate the images used in this project. While there is a large variation between the groups, images stained using the same technique are very similar, from a machine learning point of view.

Immunostaining with Ki-67

Immunostaining is often used (amongst other things) to enhance images of potential tumours. One example, and the method used in this project, is Ki-67[1] which enhances cells with a high proliferation (growth) rate [20]. Antibodies targeting specific antigens are added to the tissue sample [23]. Then an additive is added, which targets the antibodies, starting a series of chemical reactions resulting in the staining of areas with high concentration of antibodies (and hence antigens) [29]. Or in other words, potential cancer cells will shown as brown, when looking at the sample in a microscope, as it can be seen in Fig. 3.2a. A simplified model of the immunostaining process is illustrated in Fig. 3.3.

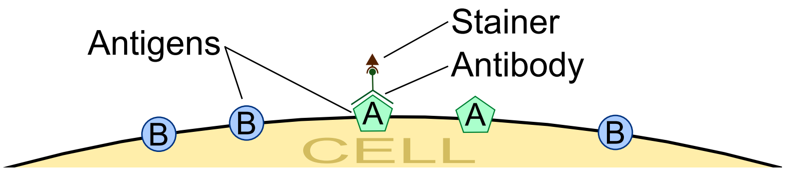


Figure 3.3: Immunostaining is done by adding antibodies which targets antigens, specific to the cell targeted for the staining. Another chemical is then added, which binds to the antibody. All cells with a high amount of the antigens, will then become a different color in the image.

[1]The name Ki-67 is derived from Kiel, the city in which it was discovered, and the fact that it was found in the 67th well on the 96-well plate.

Hematoxylin and Eosin Staining

Hematoxylin and Eosin staining in pathology (often referred to simply as H&E staining [2]) is likely the most common staining method. It uses two separate dyes, *hematoxylin* and *eosin* [2]. The hematoxylin will stain the nucleus material of the cell, to become a dark blue color. The eosin will stain the cytoplasm material to become a pink/red color. An example of a H&E staining can be seen in Fig. 3.2b.

Fluorescence Microscopy

The process of fluorescence microscopy starts like the other staining methods, by applying an agent that binds to a certain component. An example could be 4',6-diamidino-2-phenylindole, more commonly known as DAPI, which attaches to DNA in the cells [9].

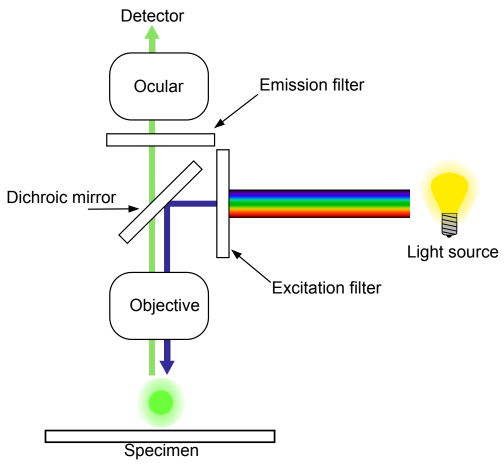


Figure 3.4: The principles of a fluorescence microscope. Light with a single wavelength is absorbed by the fluorescence agent, causing it to enter an excited state. After a while the excitation wears off, releasing energy in form of photons (light). *Image from [28] with minor modifications.*

But rather than staining the tissue sample, fluorescent stainers like DAPI, has fluorescent abilities (i.e. they "'glow in the dark"). Fluorescence agents works by absorbing light from specific wavelengths (the absorption band), which causes the electrons to become excited. After a while this excitement is released as photons (light) with different wavelengths (the emission band) [13]. An example of how this process is achieved in a microscope is shown in Fig. 3.4.

3.2 Digitalization of Data

The stained tissue samples are digitalized, using a technique called *Whole Slide Imaging* (WSI), which scans the entire sample simulating the same view as a light microscope [21]. WSI can create very high resolution images, which means that data in Visiomorph often has to be split in smaller pieces, called Field of Views [25]. An example of a typical image in this project, could be an image of the *Tagged Image File Format* (TIFF) with a resolution of 1024×1024 and a 24-bit RGB colorspace.

3.3 Classes

To continue the example from the previous section, let's look at a small part of an immunostained image, like the one in Fig. 3.2a. In Fig. 3.5 there are two different colours of cell nuclei, a brown and a dark blue. The remaining image is considered to be background, despite actually containing both cytoplasm (light blue) as well as the actual background (white).

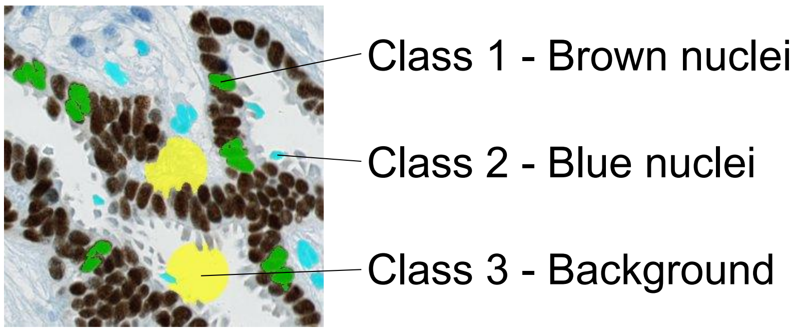


Figure 3.5: The different classes are selected by manually drawing the training areas on to a microscopy image. Different colors represent different classes. In this example three different classes are present: *Brown nuclei (dark green)*, *blue nuclei (teal)* and *background (yellow)*

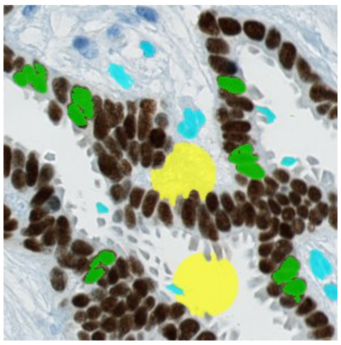
There is no clear answer to exactly whether something is considered a class of its own, or not. This is decided by the user, and depends on the purpose of the analysis. This means that areas that would be considered different classes in machine learning, could be considered a single class in a real life use case. This is not necessarily a problem, but it is worth to remember when designing new classification algorithms.

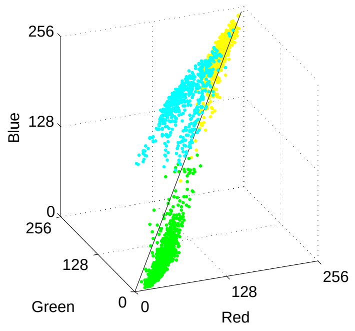
3.4 Feature Selection

Any information from a pixel which can be quantified, can be used as a feature. In this project however, focus will be on features which already exist in the Visiomorph. While there are many different features, we will try to simplify and consider only three different groups of features. These groups actually covers all the different features used in Visiomorph.

Basic Features

As the microscopy images uses the RGB colorspace, the basic features are simply the intensities for *red*, *green* and *blue*. If all three basic features are used, then the data points (pixels) are simply points in the 3-dimensional RGB space.


(a)



(b)

Figure 3.6: Areas to be used for training are drawn on a microscopy image (a). Different colors represent different classes. Data points from each class are plotted in RGB-space (b) which is the most basic feature space in pixel classification.

In Fig.3.6 an example data using RGB features is shown. Each class is located around an area according to the color of the class. The dark pixels of the brown nuclei are located close to $(0, 0, 0)$, while the bright background is in the opposite corner, close to $(255, 255, 255)$. The variety in the clusters are equivalent of the variety in pixel colors. Pixels from the same class does not have the exact same color. We will, unless otherwise mentioned, treat this phenomenon as noise.

Combined Basic Features

The second group of features which are used in VIS, is mathematical combinations of the basic features (red, green and blue). An example of such a combination could be the red chromaticity, which is included in VIS [25]. It is the value of the red color intensity of a pixel, compared to the total intensity of the pixel. Mathematically it is defined as:

$$r(x,y) = \frac{R(x,y)}{R(x,y) + G(x,y) + B(x,y)} \tag{3.1}$$

Where $r(x,y)$ is the red chromaticity of the pixel and $R(x,y)$, $G(x,y)$ and $B(x,y)$ is the red, green and blue color intensities of the pixel. In Fig. 3.7 the red chromaticity of an Ki-67 immunostained microscopy image is shown.

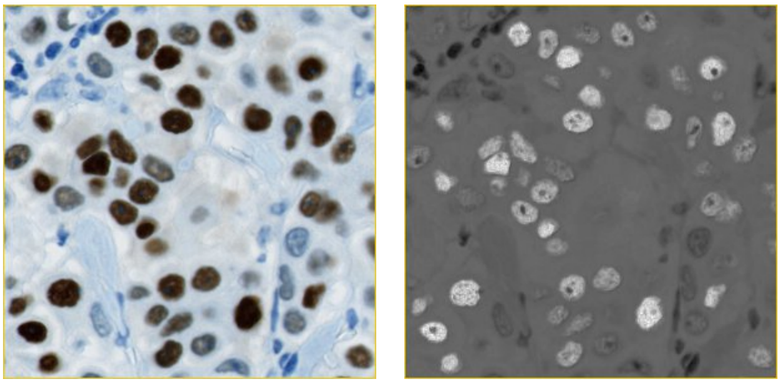

(a) Original image (b) Red chromaticity

Figure 3.7: An example of combinations of basic features. The red, green and blue intensities of the original image, has been combined into the red chromaticity, using Eq. 3.1.

Mathematical transformations of existing values do not provide any new information, they are simply transformations of existing informations. These transformations do however have their place in machine learning, transforming complex data to simpler data. Transformations therefore only have an effect on the result, if the data are complex, and the transformation can add simplicity. If not, the transformation may have no effect, or in some cases even make the result worse. Later on, we will take a look at how transformations affects the result of the SVM in this project.

Filtering Features

The last group of features, are features which are calculated by using multiple pixel values. This is done by the use of filtering [25]. A kernel of size $M \times N$ is convoluted with one of the basic features, resulting in a value which depends on all the values covered by the kernel. A simple example of a filtering operation is median filtering. It is used to reduce noise, and works by taking the median value of the pixels inside the kernel area. In Fig. 3.8 an example of a 3×3 median filter is shown. This also illustrates why the output value is dependant on every value inside the kernel area.

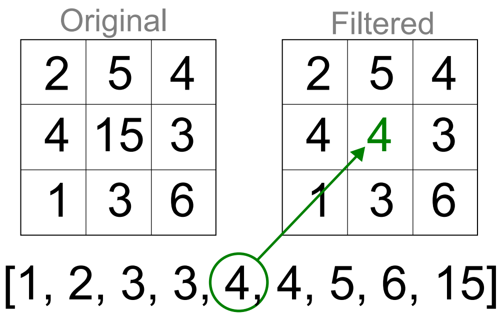


Figure 3.8: A simple example of how a median filter works. All the values covered by the 3×3 kernel is listed, and the median value is the output value.

Using filter values to include values from neighbouring pixels, should not be confused with using the neighbouring pixel values as separate features. Filtered pixel values are a calculated on the values of all pixel values covered by the filter, however it does not provide much information regarding how each pixel affected the result. Hence a lot of information is lost in the calculation.

CHAPTER 4

Design of Experiments

In the determination of whether or not SVMs would be an improvement to Visiomorph, there are two key aspects to consider. First of all it should give a higher accuracy than the existing methods. If not always, then at least in some cases, for instance for images stained by a specific method. Secondly it also needs to be considered, if SVMs are even viable for implementation. For example, even a great increase in accuracy cannot be justified, if the training time is several minutes. To address these two aspects in a systematic way, the experiments are sorted in two sections. The "Stand-alone Experiments" section contains all experiments regarding the viability of the implementation, while the "Comparison to Visiomorph" section considers the aspect of comparison to the existing methods.

4.1 Stand-alone Experiments

Hyper Parameter Values

As described in chapter 2, the value of the hyper parameters C and γ can have a large impact on the accuracy of the model. We also learned that there are no values of C and γ which are always the best. It depends on the problem that is to be solved, and should therefore be optimized for every problem. This is done by brute force, testing each combination of values for C and γ within a certain range. However this is a very slow process and not viable for implementation, as it would cause the SVM method to be much slower than any existing methods.
Instead we will try to find fixed values of C and γ, which delivers satisfying results for one of two options:

1. Images can be grouped by type, each group will have the same fixed values.

2. All images uses the same fixed values.

While option 2 is the most desirable, option 1 might be a more realistic. This also allows for ignoring some of the groups. As described in the introduction, the SVM classification do not have to be the best for all image types, in order to be an improvement. Superiority in only certain image types might also be acceptable.

From a strictly theoretical point of view, there is no reason that the choice of C and γ should affect the training time. However as we are using LibSVM as a black box tool, we will test both the accuracy and training time for each combination of C

Variable	**Value**
Minimum value of C	2^{-2}
Maximum value of C	2^{12}
Minimum value of γ	2^{-12}
Maximum value of γ	2^{0}
Step size (resolution)	6.02 dB
Repetitions of each step	3

Table 4.1: The parameter values used in the test of the variables C and γ.

and γ. The parameters used in the test can be seen in Tab. 4.1. This will be done for a series of different images, and for each image two maps will be generated. One map with the accuracy as a function of every combination of C and γ, which we denote $A(C, \gamma)$, and a second map with the training time as a function of C and γ, which we denote $T(C, \gamma)$. Finally for each value of C and γ we will find the *minimum* accuracy and *maximum* time for across all the maps. These values will be stored in two new maps, $A_{min}(C, \gamma)$ and $T_{max}(C, \gamma)$. These maps will be used to determine the fixed values of C and γ.

Minimum and Maximum Values

The larger the range of values to search is, the longer the optimization will take. While the time this experiment takes is not all that important (it will be a one time experiment), it is not completely without influence. Obviously there is a maximum number of runs which can be done, lets call this number N. Hence if we increase the maximum and minimum values of γ and/or C, we will have to increase the step size in order to fit the value range in N runs. In other words, an increase in range will mean a decrease in resolution, lowering out chances of finding the sweet spot where the accuracy is highest. In Fig. 4.2 this has been illustrated by having two grids with the same number of runs. The black grid will search a large range of values, but with a low resolution. The green grid will search a narrower range of value, but with a higher resolution. As long as we make sure that the maximum is inside the grid, the high resolution (green grid) is the better choice.

So how to decide the value range for γ and C? Starting with a huge range and then narrowing in, is an obvious approach. However at least for γ there is a less random approach. Since γ is inversely proportional to the width of the Gaussian distributions in the radial basis function [4], the maximum and minimum value of γ should relate to the maximum and minimum distance between the points that are to be classified. The data type used in Visiomorph is unsigned 8-bit integers, meaning every integer from 0 to 255. Hence the smallest distance possible is 1, independent of dimensionality. The largest possible distance is $255\sqrt{D}$, where D is the dimensionality (number of features). Looking at Fig. 2.4 we can see that at $\gamma = 2^4$ the width of the Gaussian distribution is ≈ 1, equivalent of the minimum possible distance between

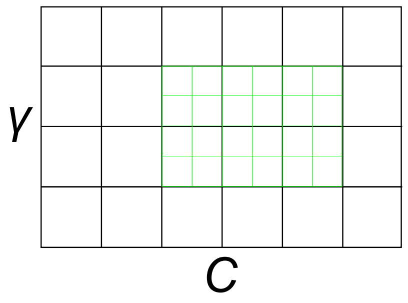


Figure 4.2: Two grids different ways to search for the optimal values of γ and C. Either have a large range of values and a low resolution (black grid) or have a lower range of values, but a high resolution (green grid).

two data points. In the other end at $\gamma = 2^{-12}$ the width is slightly less than 400 which is approximately the maximum distance in three dimensions. While the range of γ should be found in this interval, initial experiments suggested that lower values of γ usually results in the highest accuracy, so we will limit the range to be $\gamma = 2^{-12}$ as minimum, and $\gamma = 2^0$ as maximum.

For the range of C there is no intuitive answer to what the range should be. Often a rather large value is used [10], an approach that will also be used in this project. We will still keep a relatively broad range, with a minimum of $C = 2^{-2}$ and a maximum of $C = 2^{12}$.

Number of Training Data Points

Often the data available for training is a limited resource in machine learning. This is not the case in this project (as described in the chapter 3). Instead there is a risk of having too much training data, which makes the training slow. On the other hand, using too few data for training, may reduce the accuracy. This was discussed in the chapter 2, but how do we decide exactly how much data to use for the training?

To investigate the optimal number of data points for training, we will test the accuracy and training time as a function of the number of training data points. The parameter values used in the test can be seen in Tab. 4.3. The values of C and γ will be a fixed number, equivalent to the optimal value found in the first experiment.

Accuracy by Features

The accuracy as a function of features, will be evaluated using a practical approach. Visiopharm has selected 13 features which is used for the test, these features are shown in Tab. 4.4, listed by the order in which they were selected by Visiopharm.

Using one feature at a time, a model will be trained and accuracy estimated by 5-fold cross-validation. This accuracy will be used as a score for each feature, to

Variable	**Value**
Value of γ	2^{-11}
Value of C	2^{9}
Minimum number of data points	500
Maximum number of data points	10,000
Step size (resolution)	500
Repetitions of each step	3

Table 4.3: The parameter values for the test of the impact of the number of training data points. The values of γ and C has been determined in the previous experiment.

Feature number	Feature name	Notation
1	Red channel	$I(r)$
2	Green channel	$I(g)$
3	Blue channel	$I(b)$
4	Intensity	$\frac{1}{3}(I(r)+I(g)+I(b))$
5	Red chromaticity	$\frac{I(r)}{I(r)+I(g)+I(b)}$
6	Green chromaticity	$\frac{I(g)}{I(r)+I(g)+I(b)}$
7	Blue chromaticity	$\frac{I(b)}{I(r)+I(g)+I(b)}$
8	Red-Green contrast	$I(r)-I(g)$
9	Red-Blue contrast	$I(r)-I(b)$
10	Green-Blue contrast	$I(g)-I(b)$
11	HDAB - DAB	*Colour de-convolution*
12	HDAB - Haematoxylin	*Colour de-convolution*
13	Intensity gradient (polynomial)	*Not available*

Table 4.4: A list of all the features used in the experiment of testing accuracy by features. *Colour de-convolution* is described in [12].

estimate the importance of the feature. The higher the score, the more important the feature is considered (though strictly speaking this is not necessarily the case). The features are then sorted descending (from high to low), by their score. Once again a model is trained and accuracy is estimated, for the first feature on the sorted list. Next this is done for using the *two* best features, then the *three* best feature, and so on. This is done until all 13 models have been made, and the accuracy for each model estimated.

This method is based on the assumption that inter-relationships between features has only little effect. This is a quite rough assumption, but it is a necessary one, since it reduced the number of experiments from $2^{13} = 8192$ to $2 \cdot 13 = 26$, or more than a factor of 300. This is what makes this approach viable for real-time software, and not just as a one-time experiment. Depending on the results, further investigation of automatic feature selection, might be advantageous as a follow-up project.

4.2 Comparison to Visiomorph

While LibSVM has a built-in method for cross-validation, this is not the case for Visiomorph. Instead, we will use a custom made cross-validation algorithm, using 2-fold cross-validation. In order to simplify the explanation of how it works, let us forget for a second that we are working with images. Instead we will just just consider a data set containing all the data. From this set, two subsets are selected. In Fig. 4.5 those two subsets are labelled “Training Set A” and “Training Set B”.

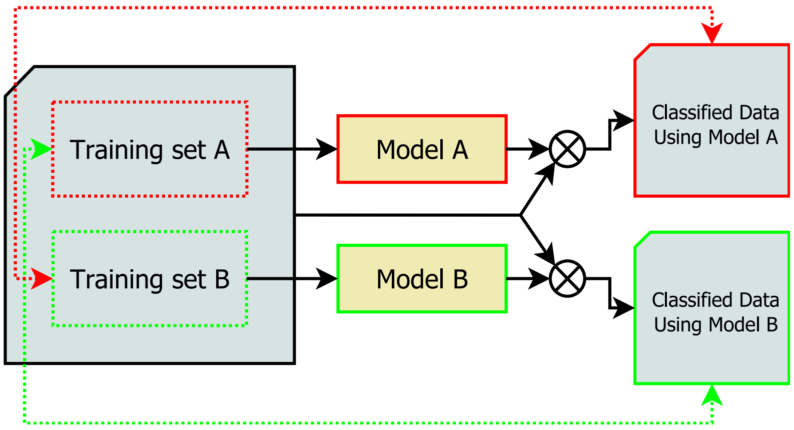
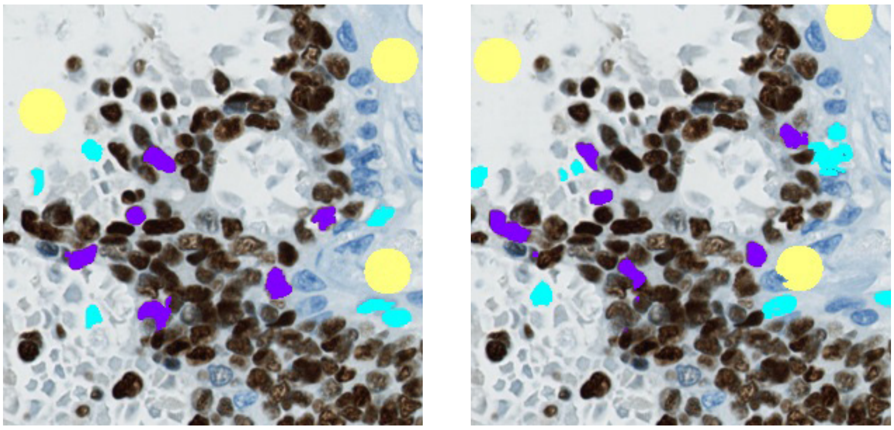


Figure 4.5: From the original data set two subsets are selected, “Training set A” and “Training Set B”. From those two models, “Model A” and “Model B”, are created. These models are used to classify the original data. The two different classification are now compared to the opposite training set, so classified data A is compared to training set B, and vice versa. The accuracy is estimated as the percentage of data points which are the same in the classified data and the compared training set.

From each training set, a model is created. Similarly these are named “Model A” and “Model B” in Fig. 4.5. Using each models to classify the full data set will give two slightly different classifications. Now to estimate the accuracy, each of the training sets are now used as a test set for the data classified using the opposite model. That means that the data classified using model A, will be compared to training set B, and vice versa. Obviously the training sets contains less data than the full set (which is the meaning of subset), so comparison will only happen for the data points that exists in the training sets. All other data points are simply ignored.

In Fig. 4.6 an example of the training set selection is shown. The brightly coloured areas are the pixels (data points) which will be used for training, each color, represents a class. When inspected carefully it is clear that while the areas selected for training are similar, they are not identical. In fact there is no overlap in the two different selections of training data.



(a) First set of training data (b) Second set of training data

Figure 4.6: The two sets of training data. The classification done with a training set (a), will be matched with the opposite training data (b), and vice versa.

In Visiomorph the user often select classification method (Bayes, K-means, etc.) depending on the input data. Therefore it makes sense to test the accuracy of each method on images stained with different methods, rather than considering all images similar. This test is however very resource dependant when it comes to manual work time. So the test set will be limited to six different images. Two from each of the three most common groups: *Ki-67 immunostained*, *H&E stained* and *fluorescent microscopy*. This way we will be able to estimate how big the potential of SVM's are, as well as where that potential lies when it comes to image groups.

CHAPTER 5

Experiment Results

This chapter summarizes the results of the experiments described in chapter 4. Some of the results may have been used to decide the set up of other experiments, or to re-design the experiment that generated the results.

5.1 Stand-alone Results

Hyper Parameter Values

In Fig. 5.1 the minimum accuracy, estimated by cross-validation, for each combination of γ and C is shown in a contour plot.

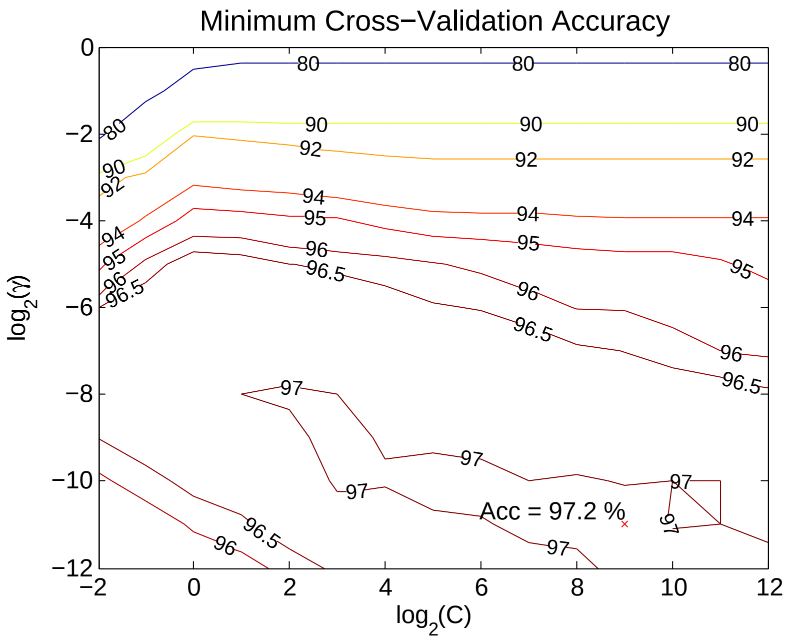


Figure 5.1: The accuracy found using values of γ ranging from 2^{-12} to 2^{0} and C ranging from 2^{-2} to 2^{12}. The maximum value is indicated by the red cross.

As it shows in Fig. 5.1, there is only one maximum of 97.2%, found at $(\gamma, C) = (2^{-11}, 2^{9})$. However a large area of the map has accuracies above 97%, and an even larger area with accuracies above 96.5%.
Another factor which has to be considered is the training times. In Fig. 5.2 a contour plot shows the time it took to run the cross-validation for each of the combinations of γ and C. While this is not necessarily identical to the actual training time, it is reasonable to assume that the cross-validation time and training times are proportional.

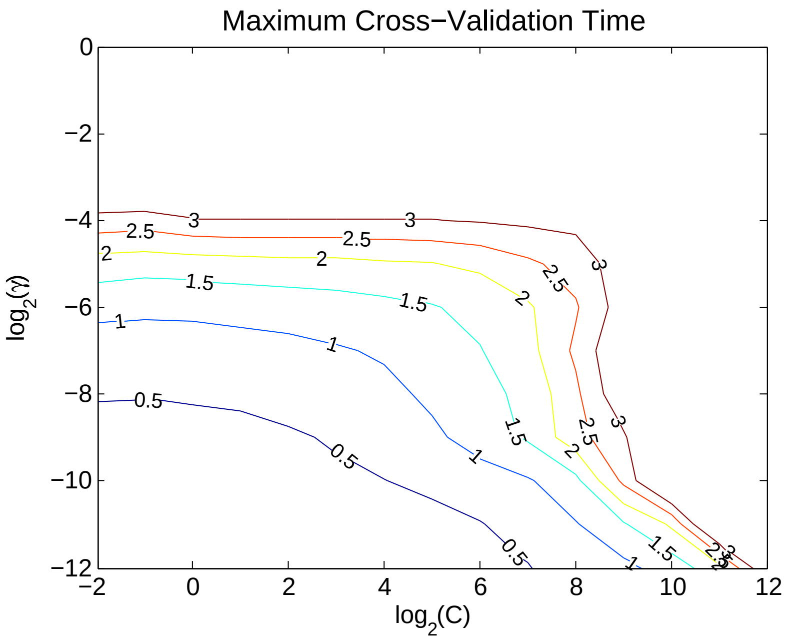


Figure 5.2: The time it took to run the cross-validations, shown in seconds. Cross-validation is proportional (but not identical) to training time, so the numbers should be considered a relative measurement of how long the training time will take.

The point of training time is not to find a single (or a few) optimal values, like it is for the accuracy. The training time acts in the way of a constraint to γ and C, limiting the potential choice of value, for the hyper parameters.

Number of Training Data Points

The impact of the training set, on training time and accuracy, is tested on a H&E stained image, with four different classes. The image is shown in Fig. 5.3.

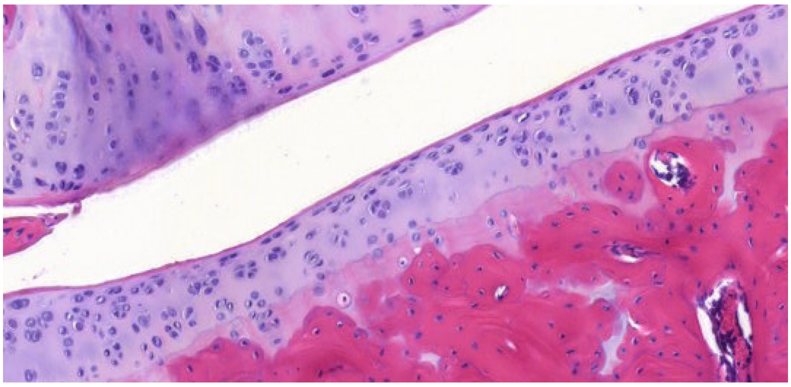


Figure 5.3: The image used to test the scaling with data size has up to 10.000 data points available for training.

The training time and accuracy is plotted as a graph in Fig. 5.4 (*left*), where a polynomial estimation is also done (*right*). The polynomial estimation indicates that the training time scaled somewhere between linear and quadratic, which can be noted as $O(N^p)$ with $1 < p < 2$.

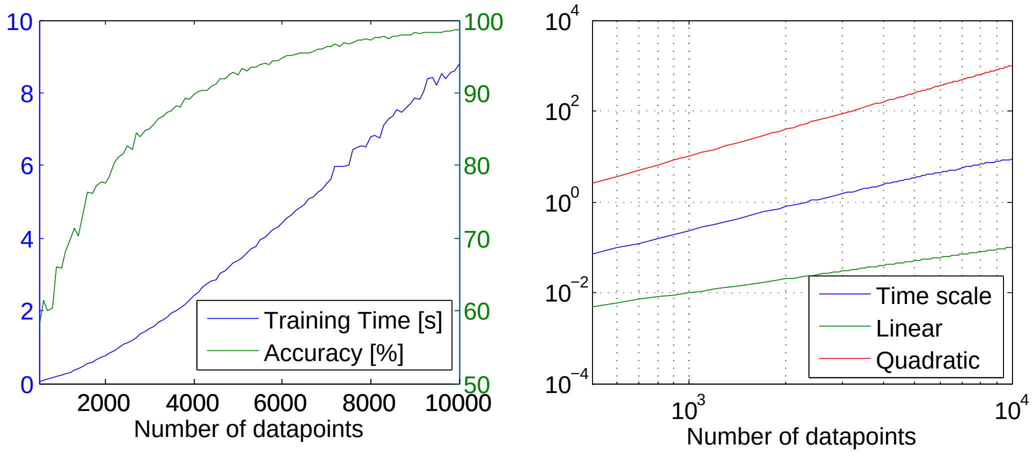


Figure 5.4: Both training time and accuracy increases with the number of data points used in the training (*left*). The training time scales somewhere between linear and quadratic (*right*).

This scaling fits well with the fact that LibSVM makes use of *Sequential Minimum Optimization* (SMO) for solving the quadratic problem [4]. SMO is an algorithm which decreases the computation time of the SVM, causing the scaling with data size to be somewhere in between linear and quadratic [22].

Accuracy by Features

The first part of the results is the accuracy of the model, using single features only. The result of this experiment is shown in Fig. 5.5, first in numerical order (*left*), and then sorted by accuracy (*right*). The order in which the features are sorted is also the order in which they are added in Fig. 5.6.

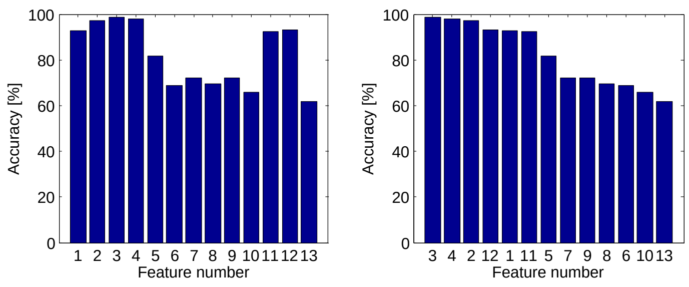


Figure 5.5: The chart shows the accuracy, estimated by cross-validation, when training is made with a single feature. The accuracy is used as a score, used to sort the features (high score is considered a more important feature).

In Fig. 5.6 the accuracy is shown depending on the number of features used. Features are added one at a time, in the order in which they are shown in the sorted plot in Fig. 5.5 (*right*). The best combination of features is that with the highest accuracy.

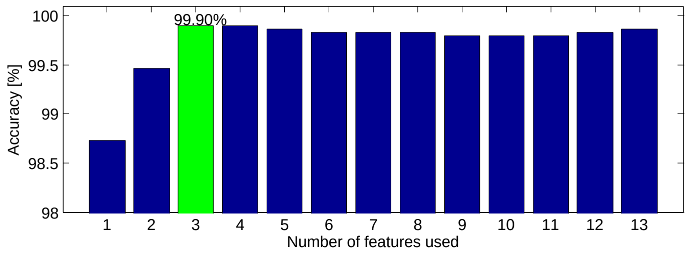


Figure 5.6: Using the sorted list from Fig. 5.5, features are added one at a time, to the model training. The accuracy is then estimated, and shown as a function of the number of features which has been used.

.

5.2 Comparison to Visiomorph

Due to the rather small number of pictures which have been tested, the results will be shown for each picture, before looking at the overall score. For each image the cross-validation accuracy A, is shown for *Bayes*, *K-Means* and *SVM* classification. The methods are sorted from best to worst, based on the accuracy. Finally the improvement is calculated by

$$A_{improve} = \frac{A_{svm} - A_{prev}}{1 - A_{prev}} \cdot 100\% \tag{5.1}$$

Where $A_{improve}$ is the improvement in accuracy, A_{svm} is the accuracy with SVM classification and A_{prev} is the best non-SVM accuracy. The scores are listed for each of the six images, and summarized in Tab. 5.7, where $A_{improve}$ is also shown. Since the maximum theoretical accuracy is 100% (equivalent of all points being correctly classified), the maximum improvement is

$$A_{improve} = \frac{1 - A_{prev}}{1 - A_{prev}} \cdot 100\% = 100\% \tag{5.2}$$

This improvement is however only achievable if the data is perfectly separable. However, this is rarely the case in real life problems. The more overlap there are between classes, the lower the maximum improvement will become.

In this chapter only the original image is shown along with its result. The classified images, along with their original, can be found in Appendix A - Full Result Set.

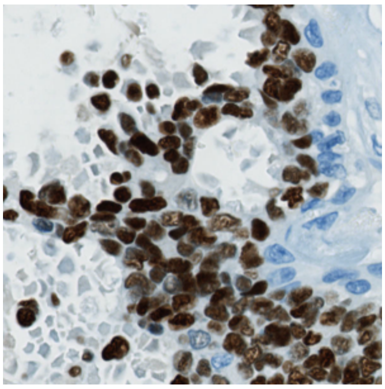
Confidence Interval

For each result the 95% confidence interval (CI) has been calculated using the Wald method, which is a method estimating binomial confidence intervals [6]. The confidence interval is calculated as

$$\hat{p} \pm z_{0.025}\sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \tag{5.3}$$

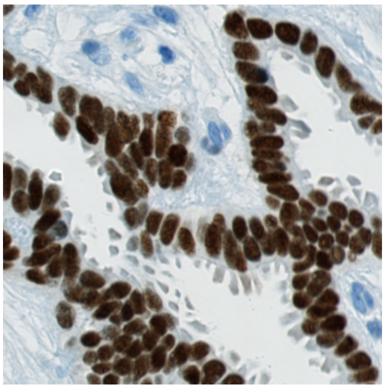
Where $\hat{p}$ is the estimated accuracy of the classification method, $z_{0.025}$ is the z-value equivalent of a 95% confidence level and $n = 3000$ is the number of data points (pixels) used in the test. It should be noticed that overlapping confidence intervals does not mean that the confidence is less that 95%. Only if the estimated accuracy of one classification method, is inside the interval of another classification method, will the 95% confidence be violated.

Ki67 - Image 1



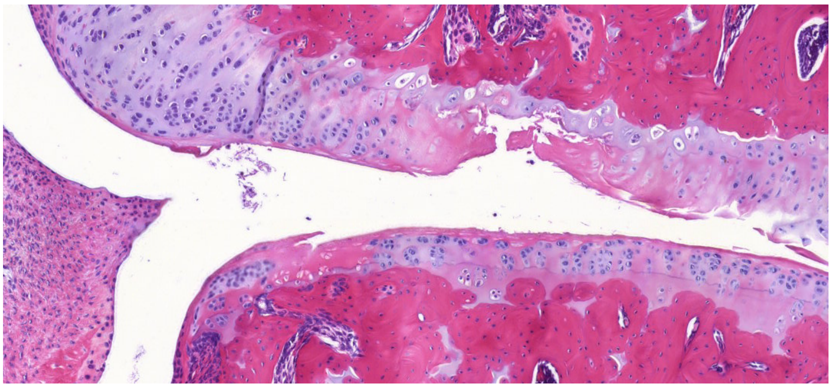
Method	**Accuracy**	95% **confidence interval**
Support Vector Machine	93.9%	$93.0\% < A < 95.8\%$
Bayesian	91.5%	$90.5\% < A < 92.5\%$
K-Means	91%	$90\% < A < 92\%$

Ki67 - Image 2



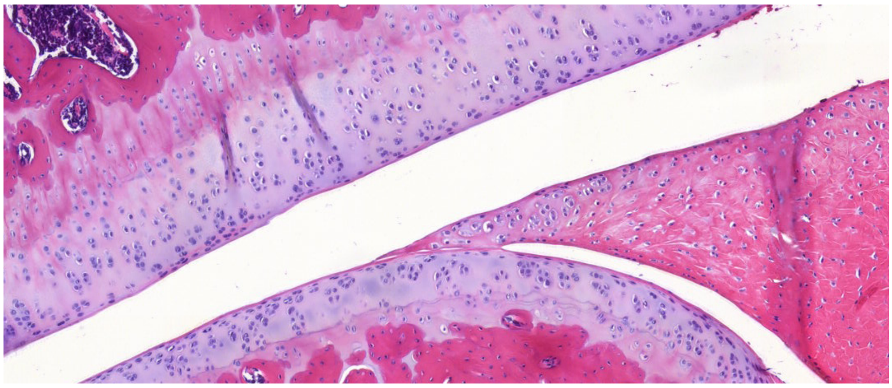
Method	**Accuracy**	95% **confidence interval**
Support Vector Machine	98.2%	$97.7\% < A < 98.7\%$
Bayesian	95.8%	$95.1\% < A < 96.5\%$
K-Means	94.4%	$93.6\% < A < 95.2\%$

H&E - Image 1



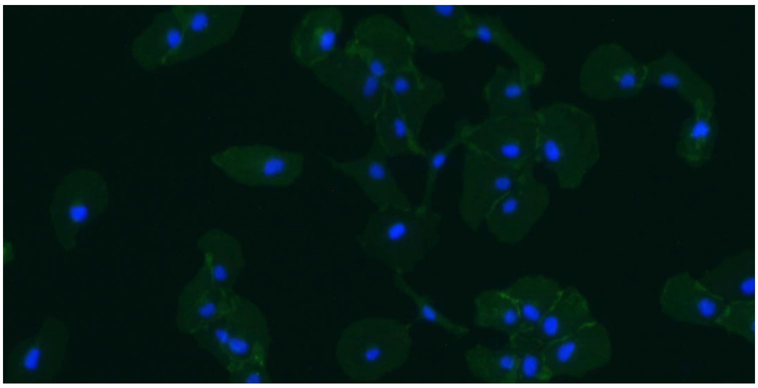
Method	**Accuracy**	**95% confidence interval**
Support Vector Machine	98.7%	$98.3\% < A < 99.1\%$
K-Means	97.3%	$96.7\% < A < 97.9\%$
Bayesian	96.7%	$96.1\% < A < 97.3\%$

H&E - Image 2



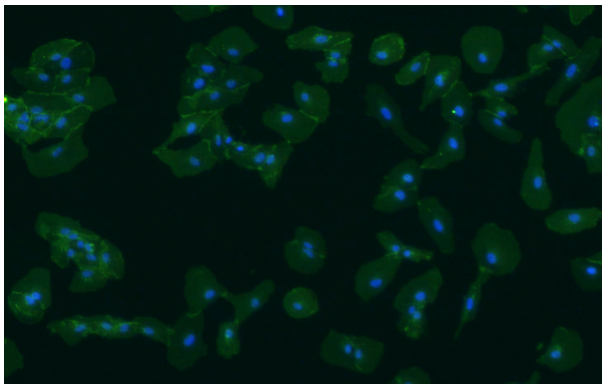
Method	**Accuracy**	**95% confidence interval**
Support Vector Machine	99.2%	$98.9\% < A < 99.5\%$
K-Means	98.6%	$98.2\% < A < 99.0\%$
Bayesian	98.6%	$92.2\% < A < 99.0\%$

Fluorescence - Image 1



Method	**Accuracy**	95% **confidence interval**
Support Vector Machine	99.8%	$99.6\% < A < 100\%$
K-Means	94.3%	$93.5\% < A < 95.1\%$
Bayesian	92%	$91\% < A < 93\%$

Fluorescence - Image 2



Method	**Accuracy**	95% **confidence interval**
Support Vector Machine	98.7%	$98.3\% < A < 99.1\%$
Bayesian	97.9%	$97.4\% < A < 98.4\%$
K-Means	96.9%	$96.3\% < A < 97.5\%$

Summary

A summary of the results from the six images, is shown in Tab. 5.7. Each method of classification is a column, while each of the six images is a row. In the fourth row the reduction in error $A_{Improve}$ (Eq. 5.1) is shown. For an easy comparison the mean value of all four measurements is calculated.

	LibSVM	**Bayesian**	**K-Means**	$A_{Improve}$
Immunostaining - Image 1	93.9%	91.4%	91.0%	28.2%
Immunostaining - Image 2	98.2%	95.8%	94.4%	56.3%
H&E Staining - Image 1	98.7%	96.7%	97.3%	51.3%
H&E Staining - Image 2	99.2%	98.6%	98.6%	41.1%
Fluorescence - Image 1	99.9%	91.7%	94.3%	95.7%
Fluorescence - Image 2	98.7%	97.9%	96.9%	39.1%
Mean value	98.1%	95.3%	95.4%	52%

Table 5.7: Summary of the accuracy estimate for the six test images. Though *Baysian* and *K-Means* are very similar in mean accuracy, it still makes a difference which one is used, depending on staining method.

Deviation in Accuracy

Another way of investigating the precision of the SVM is by brute force, an approach that is not possible in Visiomorph. The accuracy is estimated by cross-validation, this is repeated 40 times on the same data, containing $\approx 41,000$ data points and using the three basic features (RGB-values). For each iteration a new random subset, containing $3,000$ data points is selected, and accuracy is estimated by five-fold cross-validation. The results for all 40 iterations is shown in Fig. 5.8.

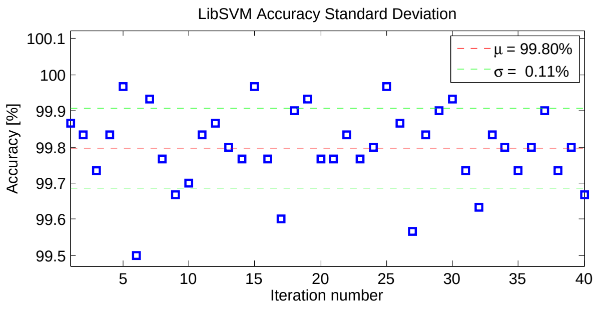


Figure 5.8: Accuracy estimated on 40 iterations of training with LibSVM. Only difference is that a new random subset of $3,000$ data points is selected for each iteration, causing slightly different results.

5.3 VisSVM - A Demo Tool

In order to get a feel of how the classification process is from beginning to end, a demo software has been created in Matlab, including a graphical user interface for improved usability. As the software makes use of the LibSVM library, the copyright terms from appendix B applies. The software is found in an public folder on Dropbox.com.

Link to VisSVM Demo Tool

Please keep in mind that this software is made for demonstration purposes, which means that there is no guard against user caused errors, such as choosing the wrong file as input. For use of the software, please read the `readme.txt`, located in the same folder.

CHAPTER 6

Discussion

6.1 Accuracy Compared to Existing Methods

From a quantitative perspective (i.e. when comparing accuracy) the SVM is superior in all six cases, as it can be seen in Tab. 5.7. On average the error has been reduced by 53%, compared to *Bayesian classification* and *K-Means clustering* currently available in Visiomorph. By looking at the 95% confidence intervals for each result in chapter 5.2, it can also be seen that the SVM in all cases is a significant improvement.

However the question of *if* the SVM was superior is only part of the comparison to Visiomorph. The other part is *how* the SVM differs from the existing methods. Looking at Fig. 6.1 and 6.2, will help answering the latter question.

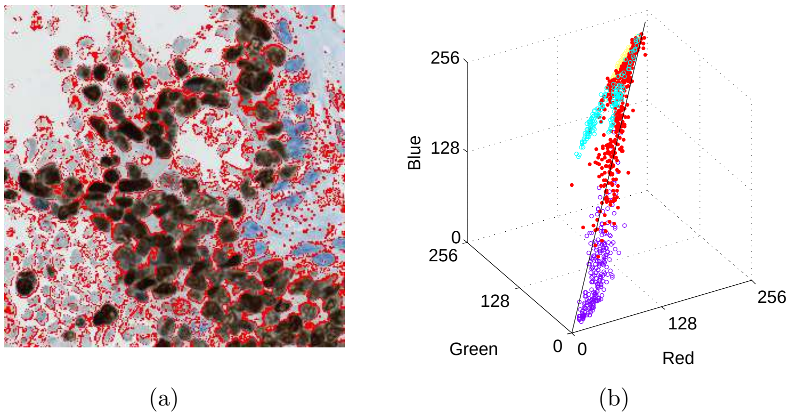


Figure 6.1: The pixels which are differently classified illustrated as the red pixels in the image (a) and in the RGB-space (b). The difference in the classification methods are mainly on the borders between classes.

The differences in classification between *SVM* and *Bayes*, has been illustrated in two different ways. In Fig.6.1a the differences are shown on the raw image, showing that most of the difference is around the borders of the nuclei. The other way is shown in Fig. 6.1b, where a subset of the different classified pixels are shown in as red scatters in the RGB-space. It shows that most of the difference is found in the regions where the classes overlap. This means that despite the rather large difference in classification accuracy, the impact of this difference should be investigated further in a possible follow-up project.

Another thing which can be analysed is where the different methods misclassifies. In Fig. 6.2 the first Ki-67 immunostained image has been analysed in this way. The errors has been marked in such a way that errors made *only* by the SVM are shown as pink, errors made *only* by the Bayesian classification are shown as dark green and errors made by *both* classifiers are shown as red. This applies to both figures.

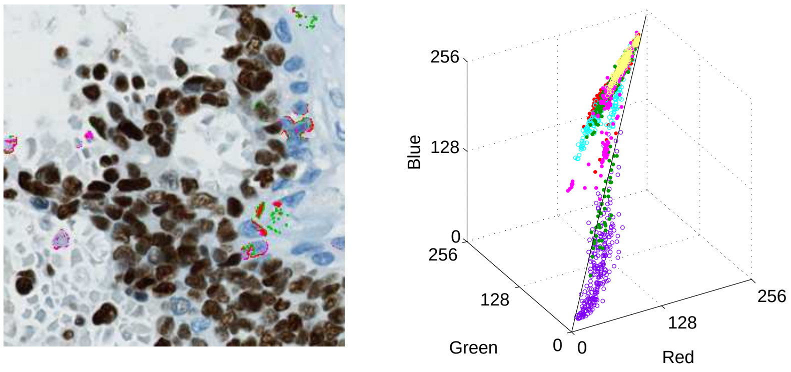


Figure 6.2: The pixels which are differently classified illustrated as the red pixels in the image (a) and in the RGB-space (b). The difference in the classification methods are mainly on the borders between classes.

It is important to remember that in Fig. 6.2a, not all pixels have been pre-labelled. Hence the true answer is only known for the training areas, limiting errors to be shown in these areas. This explains the relative low amount of pixels shown as misclassified. However it is interesting that Fig. 6.2b shows a great deal of pixels effectively inside the *brown nuclei cluster*, which have been misclassified by the Bayesian classifier. In the same area, not a single pixel is misclassified by the SVM. This means that there is not only a difference in amount of misclassifications by the methods, but also a difference in which areas are prone to misclassification.

6.2 High Dimensionality using SVM

In section 5.1 the influence of multiple features was tested. The improvement gained from using features beyond the basic RGB-values was hardly existing, however it did show that the three best features was *blue*, *green* and *HSI instesity*. This was also the combination of features which gave the highest overall accuracy, however the difference was to small to conclude anything. As mentioned in chapter 2.6, other methods involve the use of neighbouring pixels as features. It is reasonable to assume that this approach could work in Visiomorph as well, since there is some correlation between the neighbouring pixels and class. For instance in the Ki-67 immunostained images, if all the neighbouring pixels are brown as well as the pixel itself, it is likely to increase the possibility that the pixel is of the brown nuclei class.

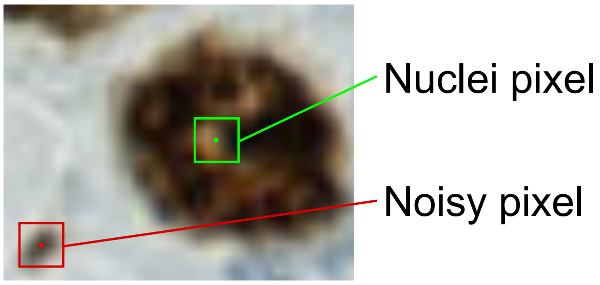


Figure 6.3: Two pixels are marked by a red and a green dot respectively. Using only the dotted pixel as a feature, both pixels would be classified as nuclei, despite the red dot being some sort of artefact. This could be avoided by using the 24 nearest neighbour pixels as features.

The cost of adding extra features comes at a price of computation time. While the computation time for SVMs is often considered to scale only with the number of data points [1], it actual also scales linearly with dimensionality [24]. Combining this with the result from chapter 5.1, the total scaling will then be $O(dN^p)$, where d is the dimensionality, N the data set size and $1 < p < 2$ the exponent of the data size scaling. However it should still be considered a point of interest for a possible follow-up project.

6.3 Kernel Width

In Fig. 5.1 the optimal value of γ was found to be $\gamma = 2^{-11}$. Knowing from chapter 2 that $\gamma = 1/2\sigma^2$, this is equivalent of

$$\sigma = \sqrt{\frac{1}{2\gamma}} \tag{6.1}$$

When inserting the optimal value of γ

$$\sigma = \sqrt{\frac{1}{2 \cdot 2^{-11}}} = 32 \tag{6.2}$$

Which is proportional to the width of the kernel. If the width is defined as the *Full Width Tenth of Maximum* (FWTM), the width can be calculated as [27]

$$2\sqrt{2\ln(10)}\sigma \tag{6.3}$$

Which when inserting $\sigma = 32$ gives a width of

$$2\sqrt{2\ln(10)} \cdot 32 \approx 137 \tag{6.4}$$

Meaning that the width of the kernel is a little more than half the width of the RGB space, which was the feature space used in the optimization of the hyper parameters. A graphical presentation of the RBF kernel width can be seen in Fig. 6.4, where the FWTM is shown on a 2-dimensional Gaussian distribution where $\sigma = 32$.

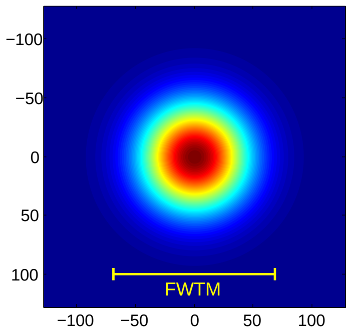


Figure 6.4: The FWTM of the RBF kernel, when using the optimized $\gamma = 2^{-11}$. The kernel is shown in a 2-dimensional space with a side length equivalent of the RGB space, i.e. 256.

This is a relatively wide kernel, resulting in a smooth classification boundary (depending on the value of C as well though). This makes sense when looking at Fig. 3.6b, where the classes are close to being linearly separable. It is also interesting that looking at the $> 97\%$ region in Fig. 5.1, the value of γ ranges from 2^{-8} to less than 2^{-12}. This gives a kernel width ranging from ≈ 50 to more than 200.

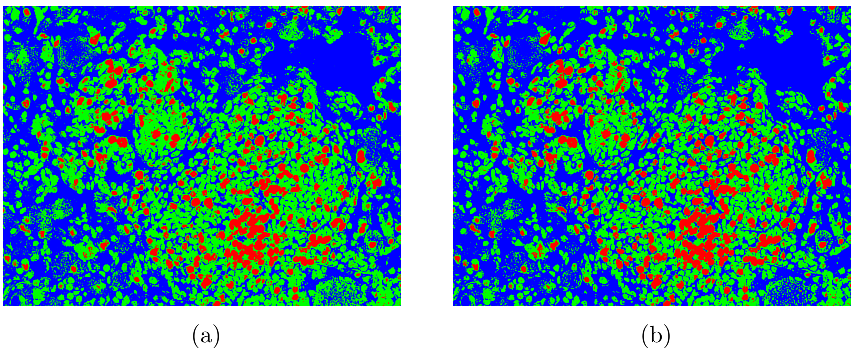


Figure 6.5: Differences in classification when $\gamma = 2^{-8}$ (a) and when $\gamma = 2^{-12}$ (b). The penalty term is the same in both, with $C = 2^8$.

In Fig. 6.5 an example can be seen of classification with two different values of γ, when the penalty term is constant at $C = 2^8$. In Fig. 6.5a a less smooth kernel with $\gamma = 2^{-8}$ is used, with an estimated accuracy of 99.4%. In Fig. 6.5b a smoother kernel is used, with $\gamma = 2^{-12}$, giving an estimated accuracy of 99.8%. The difference in the classifications (especially the ratio between green and blue) is larger than the difference in accuracy may suggest. Hence it might not be enough simply to look at the quantitative measure of success (accuracy), when deciding values of the hyper parameters.

6.4 Data Scaling

Currently there is a big difference in the value range for by different features. As mentioned the basic features (*red*, *green* and *blue* intensity), has a value between 0 and 255. On the other hand the *chromaticities* (see Eq. 3.1), will have values between 0 and 1. Besides making correct classification more difficult, this also means that if the data type is changed to *uint8* (for programming reasons), all values less than one will be reduced to zero. It is an issue that Visiopharm is aware of, but it is still an issue which should not be forgotten.

Another, and less critical, consideration in the scaling of data is whether or not all features should be scaled to full range. E.g. if the red intensity only has values from 26 to 217, should histogram equalization[1] be applied to the red intensity? This has not been investigated in this project, but might be a good potential topic for a follow-up project.

[1]Histogram equilization is a method which increases the contrast, by mapping data to its full value range.

CHAPTER 7

Conclusion

7.1 Improvements in Accuracy

This project has proved that, if looking solely at the accuracy, SVMs is an improvement over the *Bayes classifier* and *K-Means clustering* currently used in Visiomorph. In all six test images the SVM performed better than the current methods, with a significance level of $\alpha < 0.05$ (more than 95% chance of the result being correct). And with an average reduction in error of 52%, more than half the misclassified pixel can be avoided. This was achieved with fixed values of γ and C, and using only the basic features (RGB).

7.2 Computation Time

The fact that an improvement in accuracy can be achieved with a low number of features and fixed value hyper parameters, means that the training time can be kept at a reasonable level. Fig. 5.4 shows that a very stable accuracy can be achieved using ≈ 6000 data points for training, while still having a *training time* of few seconds (depending on the computer).

This still leaves an unanswered question regarding the *classification time*. However this probably makes more sense to test once LibSVM has been implemented in Visiomorph, as this would allow direct comparison to the existing methods. It cannot be tested in the same manner as the training time, as Visiopharm cannot provide a level of classification time which would be considered acceptable (the current methods are also time consuming).

7.3 Suggestions

Besides the direct comparison to Visiomorph, in terms of accuracy, a number of ways to improve the SVM was also investigated. These additional improvements might help increase accuracy or improve user friendliness (or both).

Automatic Optimization of Hyper Parameters

In chapter 4.1 a brute force method for optimizing the hyper parameters was described. Optimization of the hyper parameters prior to the classification might help increase the accuracy slightly. It does however *not* justify the huge increase in computation time, which it causes. If this feature is added, it should be as an optional one, turned off as default. As another option, optimization of the fixed parameters can be done by Visiopharm and included in new releases of Visiomorph.

Automatic Feature Selection

Automatic selection of features does give rise to small improvements in the classification, as it can be seen in Fig. 5.6. Additionally it can increase the user friendliness, as the user no longer has to select features manually. While automatic feature selection is not nearly as time consuming as optimization of the hyper parameters, it does increase the training time. Whether or not this increase is justified, should be investigated further before implementation. Additionally it might also be possible to make some more sophisticated algorithms for feature selection, which could improve the effect and/or decrease the time consumption.

Other Possibilities

Another thing that is definitely worth a closer look in the future work, is the use of neighbour pixels as features. Looking at the logic behind it, as well as the good results it has generated for others [14], this might be one of the most obvious topics for a follow-up project. It could also be interesting to take a closer look at the use of ensemble classifiers, in which multiple classifiers are used, and then a vote is made on which class a pixel belongs to. This could be either multiple SVMs or a mix of SVM, Bayesian and K-Means classifiers.

7.4 Summing Up

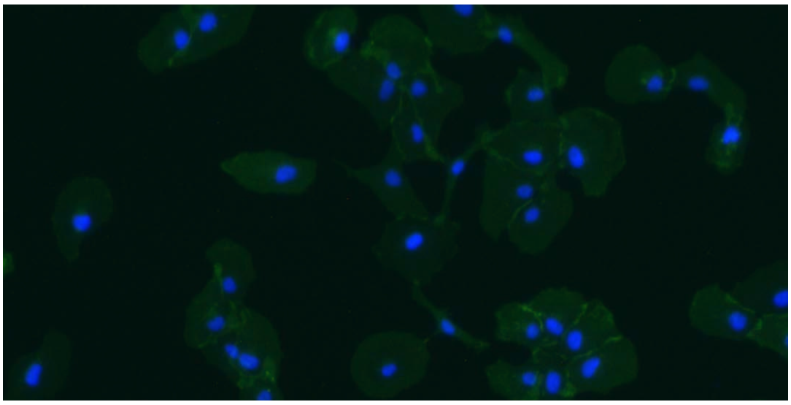
Overall the use of *support vector machines* offers a lot of new and exciting options, as well as an increase in accuracy. And if implemented with the LibSVM library, implementation is likely to be worth the cost, even considering that Visiopharm is not a large company. My personal recommendation to Visiopharm however, would be to perform a follow-up project, where the suggestions given are investigated further. This should ensure a more thorough implementation, taking the SVM classification a step further than just being "yet another classifier". In other words, if played correctly, the SVM is a great hand to hold. One which could possibly take Visiomorph to the next level.

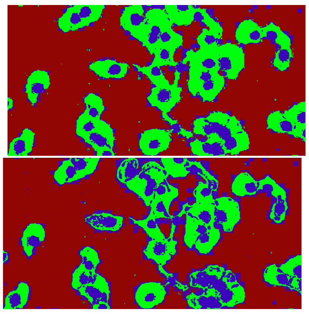
APPENDIX A

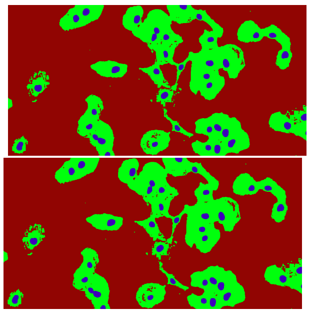
Full Result Set

A.1 Fluorescence Images

Image 1




(a) Bayesian


(b) K-means

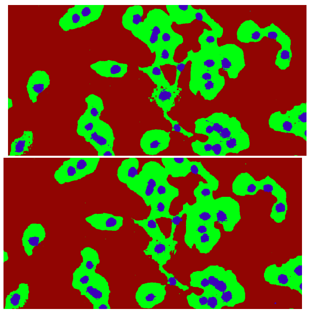
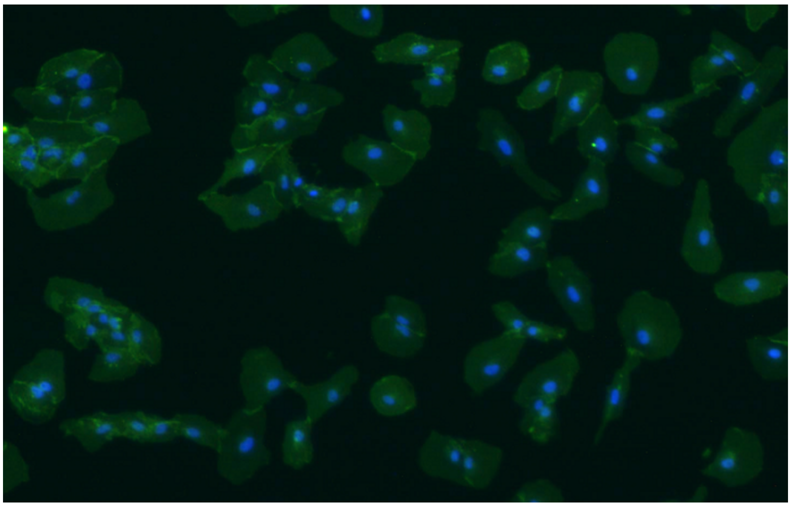
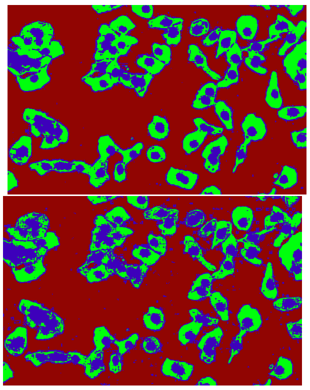
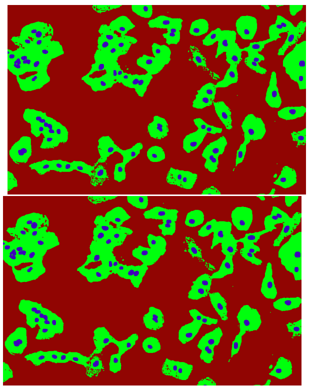

(c) Support vector machines

Image 2

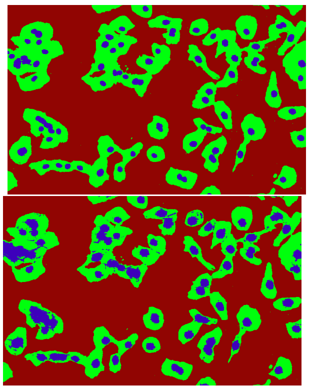




(a) Bayesian



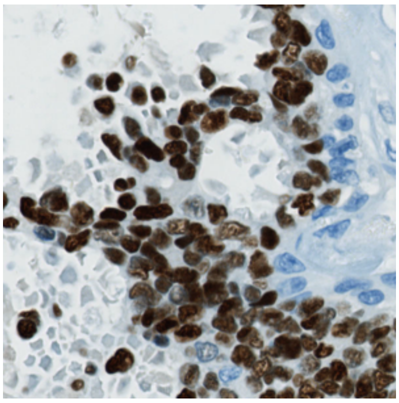
(b) K-means

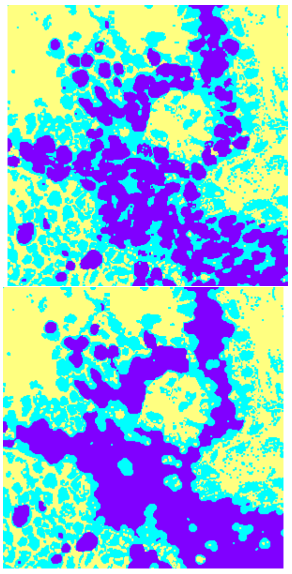


(c) Support vector machines

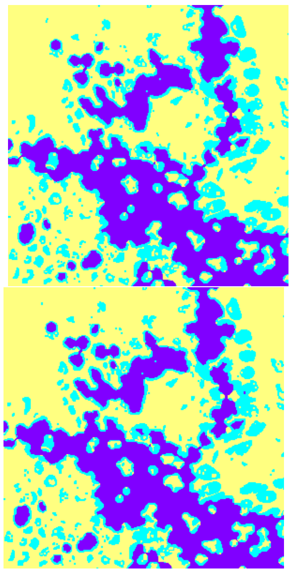
A.2 Immunostained Images

Image 1

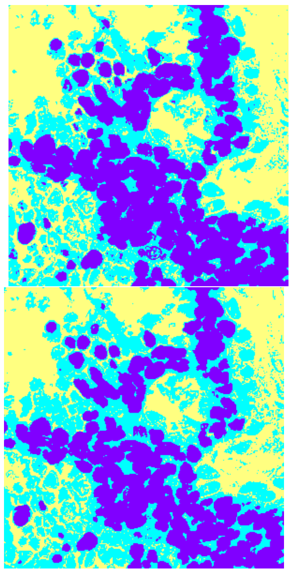




(a) Bayesian

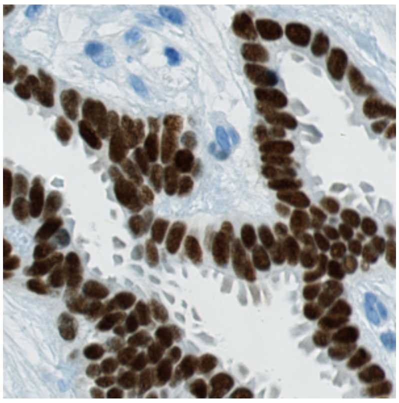


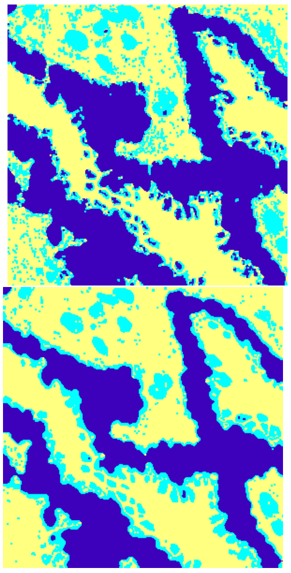
(b) K-means



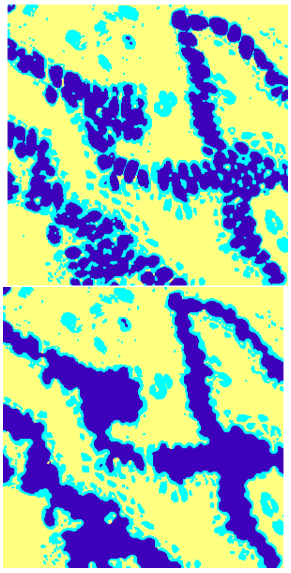
(c) Support vector machines

Image 2

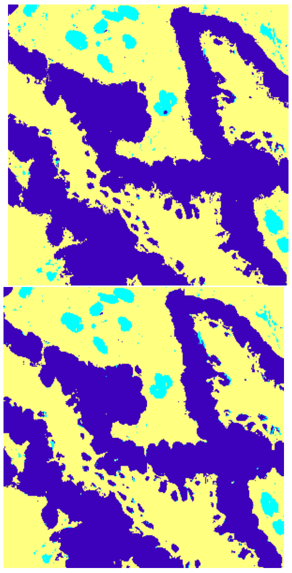




(a) Bayesian



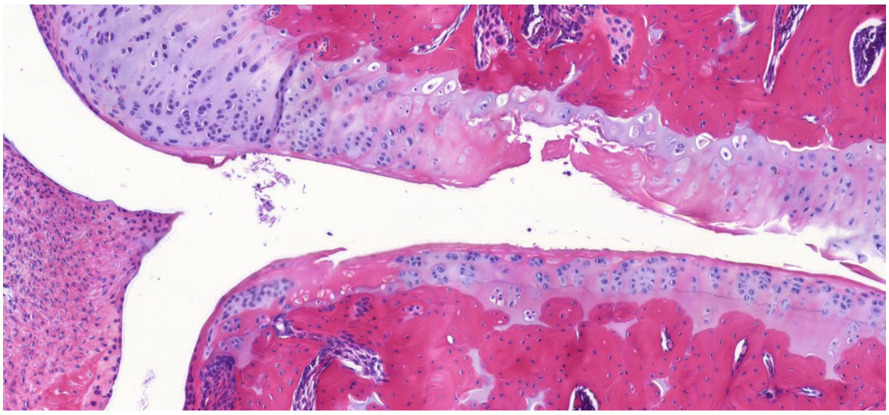
(b) K-means

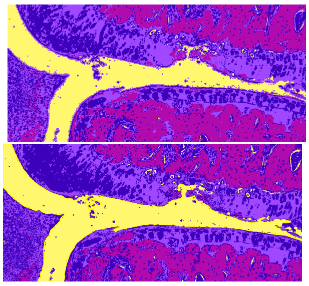


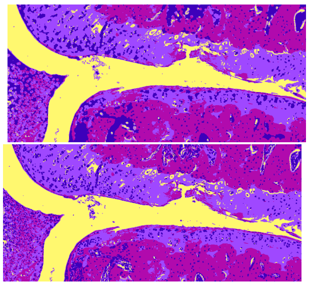
(c) Support vector machines

A.3 HE Stained Images

Image 1




(a) Bayesian


(b) K-means

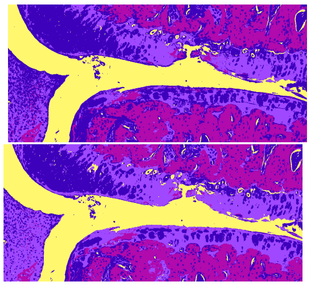
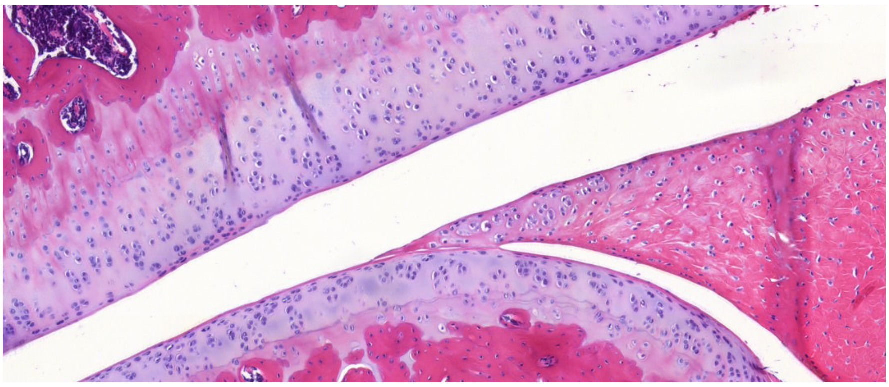
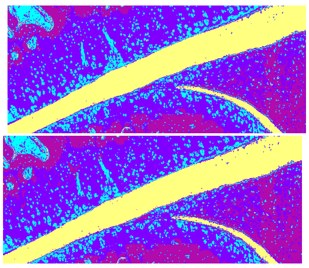
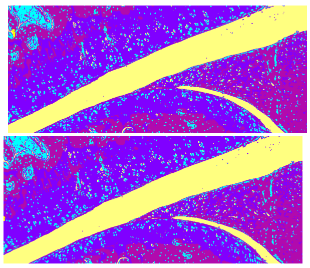

(c) Support vector machines

Image 2

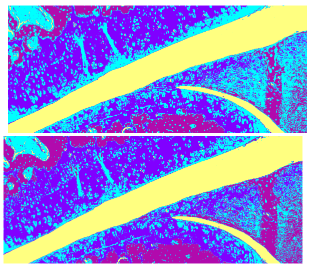




(a) Bayesian



(b) K-means



(c) Support vector machines

APPENDIX B

Copyright

List of Figures

List of Tables

Bibliography

[1] Cristopher M. Bishop. *Pattern Recognition and Machine Learning.* Number 978-0387-31073-2. Springer, 2006.

[2] H. Skip Brown. Hematoxylin & eosin (the routine stain). Technical report, Sigma-Aldrich, 2002.

[3] Chih-Chung Chang and Chih-Jen Lin. A practical guide to support vector classification. April 2010.

[4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[5] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine learning*, 46(1):131–159, 2002.

[6] Xinjia Chen. Coverage probability of wald interval for binomial parameters. Technical report, Louisiana State University, 2009.

[7] Nikolaos Giannakeas, Petros S Karvelis, Themis P Exarchos, Fanis G Kalatzis, and Dimitrios I Fotiadis. Segmentation of microarray images using pixel classification—comparison with clustering-based methods. *Computers in biology and medicine*, 43(6):705–716, 2013.

[8] Isabelle Guyon. A scaling law for the validation-set training-set size ratio. *AT&T Bell Laboratories*, 1997.

[9] J. Kapuscinski. Dapi - a dna-specific fluorescent-probe. *BIOTECHNIC and HISTOCHEMISTRY*, 70(5):220–233, 1995.

[10] SS Keerthi and CJ Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *NEURAL COMPUTATION*, 15(7):1667–1689, 2003.

[11] Rethabile Khutlang, Sriram Krishnan, Ronald Dendere, Andrew Whitelaw, Konstantinos Veropoulos, Genevieve Learmonth, and Tania S Douglas. Classification of mycobacterium tuberculosis in images of zn-stained sputum smears. *Information Technology in Biomedicine, IEEE Transactions on*, 14(4):949–957, 2010.

[12] Andreas Kårsnäs. *Image Analysis Methods and Tooks for Digital Histopathology Applications Relevant to Breast Cancer Diagnosis.* PhD thesis, Uppsala Universitet, 2014.

[13] Joseph R. Lakowicz. Principles of fluorescence spectroscopy. *Principles of Fluorescence Spectroscopy*, pages 1–954, 2006.

[14] Boris Lenseigne, Priscille Brodin, Hee Kyoung Jeon, Thierry Christophe, and Auguste Genovesio. Support vector machines for automatic detection of tuberculosis bacteria in confocal microscopy images. In *Biomedical Imaging: From Nano to Macro, 2007. ISBI 2007. 4th IEEE International Symposium on*, pages 85–88. IEEE, 2007.

[15] D. D. Lewis, C. Nedellec, and C. Rouveirol. Naive (bayes) at forty: the independence assumption in information retrieval. 1998.

[16] Olivier Lezoray and Hubert Cardot. Cooperation of color pixel classification schemes and color watershed: a study for microscopic images. *Image Processing, IEEE Transactions on*, 11(7):783–789, 2002.

[17] David MacKay. An example inference task: clustering. *Information Theory, Inference and Learning Algorithms*, pages 284–292, 2003.

[18] Ujjwal Maulik and Debasis Chakraborty. A self-trained ensemble with semisupervised svm: An application to pixel classification of remote sensing imagery. *Pattern Recognition*, 44(3):615–623, 2011.

[19] Afshin Rostamizadeh Mehryar Mohri and Ameet Talwalkar. *Foundations of Machine Learning.* Number 978-0-262-01825-8. The MIT Press, 2012.

[20] Susanne Holck Niels Marcusen, Flemming B. Sørensen and Torben Steiniche. *Patologi.* FADL's Forlag, 2010. Danish book.

[21] Liron Pantanowitz. Digital images and the future of digital pathology. *Journal of Pathology Informatics*, 1, 2010.

[22] John C Platt. Fast training of support vector machines using sequential minimal optimization. 1999.

[23] T. Scholzen and J. Gerdes. The ki-67 protein: From the known and the unknown. *JOURNAL OF CELLULAR PHYSIOLOGY*, 182(3):311–322, 2000.

[24] V Sreekanth, Andrea Vedaldi, Andrew Zisserman, and C Jawahar. Generalized rbf feature maps for efficient detection. 2010.

[25] Visiopharm. *Working With: VisiomorphDP, The Calculator, ArrayImager.* 2013.

[26] Xiang-Yang Wang, Ting Wang, and Juan Bu. Color image segmentation using pixel wise support vector machine classification. *Pattern Recognition*, 44(4):777–787, 2011.

[27] Eric Weisstein. Wolfram mathworld - gaussian function, 2014.

[28] Wikipedia. Fluorescence microscope, 2014. Provider of fluorescence figure.

[29] Hanns-Olof Wintzer, Irmgard Zipfel, Jürgen Schulte-Mönting, Ursula Hellerich, and Sabine von Kleist. Ki-67 immunostaining in human breast tumors and its relationship to prognosis. *Cancer*, 67(2):421–428, 1991.