2016

Machine Learning Techniques for Cervigram Image Analysis

Cheng Xin
Lehigh University

Machine Learning Techniques for Cervigram Image Analysis

by

Cheng Xin

A Thesis

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Computer Science

Lehigh University

May, 2016

This thesis is accepted and approved in partial fulfillment of the requirements for the Master of Science.

Date

Thesis Advisor

Chairperson of Department

Acknowledgement

First and foremost, I would like to thank my thesis advisor Prof. Xiaolei Huang. She generously provided me with every possible assistance for supporting my research. She taught me how to be a good and professional researcher.

I also thank Prof. Henry Baird, who retired last year. I was inspired by his excellent lectures to study more in the area of pattern recognition and machine learning. He was the first person to encourage me to complete this thesis and try to do more work that interests me.

I would like to thank my parents. Although they are in China, their trust in me and spiritual and financial support for my work, beyond the limit of time and space, are the most indispensable things in my life.

I would like to thank my lab mates, Tao Xu, Ting Xu, Sunhua Wan, and Ph.D. or master candidates in other labs, Ziyi Guo, Yi Luo, Wenbo Li, Qinghan Xue, Dawei Li, Chao Wang, Sicong Kuang. They had enlightening discussions with me on a lot of problems in theory and practice. They also gave me valuable suggestions when I met various problems in my research.

I would like to thank Prof. Donal Davis for teaching me topology, which inspired me to study mathematics seriously and opened the door to a new world for me.

I would like to thank Prof. Brian Davison for teaching me the course data mining. I got to know a lot of interesting problems and beautiful works in that course.

I would like to thank Prof. Mooi Choo Chuah for her kind help and suggestions during my last two years in the department of computer science at Lehigh.

Contents

List of Figures

List of Tables

Abstract

Machine learning is a popular technology widely used to solve a lot of problems in various areas in recent decades. In this work, we applied machine learning techniques to the problems of medical image analysis, especially cervigram image analysis. Combined with techniques developed in computer vision, we represent cervigram image data in the form of a combination of texture feature vector and color feature vector. We treat the task of detecting Cervical Intraepithelial Neoplasia (CIN) level as a classification problem in the view of machine learning and apply several popular machine learning classifiers to predict the categories. Furthermore, under receiver operating characteristic (ROC) curve as our performance measure, we do a comprehensive comparison among seven machine learning classification algorithms to see which ones might be suitable models for this kind of problems. From our experiments, we conjecture that the machine learning techniques can be a useful tool and ensemble-tree based models like Random Forest, Gradient Boosting Decision Tree and Adaboost outperform other algorithms for this task.

Chapter 1

Introduction

Machine learning is an idea of leading a system to automatically learn how to solve a specific problem better from some collection of given experience. This idea becomes one of the most popular tools in the area of AI in recent decades. When human beings expect the computer to solve more and more complicated problems, we find it is too hard to manually construct the effective programs for the computers. Even for some problems, we have few ideas of how to solve them by ourselves. The machine learning ideas is a very attractive alternative. It has spread rapidly throughout computer science and beyond. Besides AI, Machine learning is widely used in Web search, spam filters, recommender systems, ad placement, credit scoring, fraud detection, stock trading, drug design, medical image processing and many other applications. A recent report from the McKinsey Global Institute asserts that machine learning (a.k.a. data mining or predictive analytics) will be the driver of the next big wave of innovation [1].

In order to make a good application of machine learning to some specific problems we are interested in, it is important to study carefully about the theory of machine learning. Although a lot of machine learning algorithms has been designed as off-the-shelf tools, machine learning should never be regarded as a magic. There is no universal perfect machine learning for any tasks. Machine learning is more like a knowledge lever. We need carefully represent our knowledge about the specific problem for the computer and design a corresponding suitable machine learning model or algorithm.

Machine learning problem can be broadly classified into two categories, supervised learning and unsupervised learning, depending on whether the label information is provided in the dataset. The supervised learning means that some label information are included in the data set and the machine learning model is required to make prediction on the label values for some unseen data. Classification and regression are two most common tasks in the category of supervised learning problems. Unlike the supervised learning, the unsupervised learning is referred to a problems with data set without any label information available. The machine learning needs to find some meaningful information automatically from the given data description. Clustering analysis is a representation of the category of the unsupervised learning problems.

In our work, we care more about the supervised learning, especially those classification problems. The classification problem is that given a data set with label values in discrete space, the machine is asked to specify which of some k possible categories the input belongs to. More formally, given a data set $X = \left\{X^{(i)}\right\}_{i=1}^{N}$ where $X_{\mathrm{i}} \in \mathbb{R}^D$, which means each data sample X_i is in a D-dimensional feature space, and corresponding label values $y = \left\{y^{(i)}\right\}_{i=1}^{N}$, where $y^{(i)} \in \mathbb{Z}_k$, which means there are k choices of each label value $y^{(i)}$, a classification learner, or called a classifier, can be viewed as a function $f : \mathbb{R}^D \to \mathbb{Z}_{\mathrm{k}}$, i.e. it returns an output value in the category space $\mathbb{Z}_{\mathrm{k}}$ for each input sample in the data space $\mathbb{R}^D$.

Machine learning has been widely used to solve a lot of problems in various areas. It is also a useful tool in medical image analysis. In our work, we focus on analyzing the cervigram image with machine learning technologies. It shows us its power to be a great assistance for computer aided diagnosis.

Cervical cancer ranks as the second most common type of cancer in women aged 15 to 44 years worldwide [2]. Among death cases caused by cervical cancer, over 80% occurred in less developed regions. Therefore, there is a need for lower cost and more automated screening methods for early detection of cervical cancer, especially those applicable in low-resource regions. Screening procedures can help prevent cervical cancer by detecting cervical intraepithelial neoplasia (CIN), which is the potentially precancerous change and abnormal growth of squamous cells on the surface of the cervix. According to the WHO system [2], CIN is divided into three grades: CIN1 (mild), CIN2 (moderate), and CIN3 (severe). Lesions in CIN2/3+ require treatment, whereas mild dysplasia in CIN1 only needs conservative observation because it will typically be cleared by an immune response in a year. Thus, in clinical practice one important goal of screening is to differentiate CIN1 from CIN2/3 or cancer (denoted as CIN2/3+ [3]).

The most widely used cervical cancer screening methods today include the Pap test, HPV testing, and visual examination. Pap tests are effective, but suffer from low sensitivity in detecting CIN 2/3+ [4]. Moreover, Pap tests need a laboratory and trained personnel to evaluate the samples. The sensitivity of HPV tests in detecting CIN 2/3+ lesions varies greatly [4].

Colposcopy is a diagnostic procedure that often involves setting a biopsy. Digital Cervicography, a non-invasive visual examination method that takes a photograph of the cervix (called a cervigram) after the application of 5% acetic acid to the cervix epithelium, has great potential to be a primary or adjunctive screening tool in developing countries because of its low cost and accessibility in resource-poor regions. However, one concern with Cervicography is that the overall effectiveness of Cervicography has been questioned by reports of poor correlation between visual lesion recognition and high-grade disease as well as disagreement among experts when grading visual findings.

To address the concern and investigate the feasibility of using images as a screening method for cervical cancer, we conjecture that computer algorithms can be developed to improve the accuracy in grading lesions using visual (and image) information. This conjecture is inspired and encouraged by recent successes in computer-assisted Pap tests such as the ThinPrep Imaging System (TIS) [5], FocalPoint [6], and the work by Zhang et al. [7]; these computer-assisted Pap tests apply multi-feature Pap smear image classification using SVM and other machine learning algorithms, and they have been shown to be statistically more sensitive than manual methods with equivalent specificity.

From the perspective of machine learning, here we have a classification problem. Taking cervigram images as input, a machine learning model is required to predict the corresponding CIN level. The raw image data is in a quite high dimension. The information each pixel in an image can represent is limited, which is not suitable for applying traditional machine learning models directly. Fortunately, in the area of computer vision and image processing, there are various mature technologies and tricks of extracting generally meaningful information, or called features, from the image. As we have mentioned before, the machine learning is like a lever of knowledge. It should be designed to make fully use of existed experience and human beings' knowledge. And these features extracted from the image just represent our knowledge about the image and the task.

In our work, we applied several popular machine learning models on texture and color features extracted from a large set of cervigram images to do the classification task. And we also do a comprehensive comparison on seven machine learning algorithms to see which ones might be better choices for this kind of problems. Our dataset consists of 345 positive samples and 767 negative samples. So it is an imbalanced dataset. Each sample is in a 2538 dimensional feature space that is composed with PLBP, PLAB and PHOG.

We do our experiments on both the whole(imbalanced) dataset and balanced dataset after downsampling.

The construction of the paper is as follows: In Chapter 2, we study the theory of machine learning carefully, illustrate several important concepts and problems in machine leaning techniques, and describe seven popular machine learning classification models used in our work. In Chapter 3, we delineate the problem of cervigram image analysis, describe the database we construct and features we design for machine learning models. In Chapter 4, we illustrate the experiments and results of applying seven machine learning models described in Chapter 2 to the dataset delineated in Chapter 3. Finally, some concluding remarks and pointers to future directions of our work are provided in Chapter 7.

Chapter 2

Machine Learning

Although a lot of machine learning algorithms has been designed as off-the-shelf tools, machine learning should never be regarded as not a magic. There is no universal perfect machine learning for any task. Machine learning is more like a knowledge lever. We need carefully represent our knowledge about the specific problem for the computer and design a corresponding suitable machine learning model or algorithm.

2.1 What is machine learning?

Generally speaking, machine learning is an idea of letting a system to automatically learn how to solve a specific problem better from some collection of given experience. Compared with the traditional strategy that human beings design and construct programs manually for computers to solve some specific problems, machine learning is an attractive alternative. The key concept of the idea of machine learning is the learning ability or behavior of the machine learning system. So the first question is what do we mean by learning? There is a formal definition given in [8] that "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E." For different kinds of task, performance measure and experience, there might be various machine learning algorithm. We will introduce some important properties in machine learning models in general. And then we will give introduction to seven popular machine learning models used in our work later.

2.1.1 Experience

There are several kinds of format of the experience. However, since this experience should be processed by computers, they should be in some format of numerical values. Typically, the experience E is a dataset $X = \{x_i\}$. It is composed with a collection of observations x_i. Each x_i is a real-value vector in an D-dimentional space, i.e., $x_i \in \mathbb{R}^D$. We call it feature space. The data set can be composed with just raw data sample, like a picture that is represented as an $N \times N$ pixel matrix. Or it might also consist of data sample in some high-level feature space. And these high-level features embody human beings' knowledge about the data and the task.

2.1.2 Task

For a lot of tasks which are too complicated to solve with fixed programs designed and written by human beings, machine learning is an efficient alternative way to let computer to learn to deal with these tasks automatically. There is nearly no limit on the types of task, as long as it can be transferred by our human beings to a problem the machine can deal with. So from the view of the computer, a machine learning task can be generally described in the terms of how a machine learning system should produce outputs for the inputs. The input data sample is often described in a numerical way. We typically represent a sample as a vector $x \in \mathbb{R}^D$ where each entry x_i of the vector can be another feature vector $\{x_{i_1}, x_{i_2}, x_{i_3}, \ldots\}$. According to the types of inputs and outputs, machine learning tasks can be classified in different categories. One classifying strategy is composed with supervised learning and unsupervised learning, according to the situation whether there is any label information included in the data set. The supervised learning means there are label information included in the data set and the machine learning model is required to make prediction on the label values for some unseen data. Classification and regression are two most common tasks in the category of supervised learning problems. The unsupervised learning is referred to a problems with data set without any label information available. The machine learning needs to find some meaningful information automatically from the given data description. Clustering analysis is a representation of the category of the unsupervised learning problems. Of course there are tremendous other kinds of tasks and category strategies. Here we mainly focus on supervised learning, especially classification problems, which is also the machine learning task stuided most.

2.1.3 Performance Measure

A performance measure P is designed by human beings to evaluate the performance of a machine learning system for a specific task. So the performance measure P is task specific. For a task like classification, a straightforward choice of performance measure is the accuracy, the proportion of of samples for which the machine learning model predicts correctly. Or equivalently, we can also choose the error rate, which is just equal to 1 - accuracy. If we set model's correct prediction as 1 and incorrect prediction as 0, the expected accuracy score should be the expected probability of model's correct prediction.

Another widely used performance measure is the receiver operating characteristic (ROC). If the output of a classification model can be viewed as a probability with which the input should be classified as some category, the roc analysis may be applied to get more useful information from the outcome. For example, an optimal threshold other than 0.5 might be more reasonable in application.

2.2 Generalization

It should be noticed that the ultimate target of a machine learning algorithm is to give good performance on new, previously unseen inputs – not just those on which the model was trained [9]. And this key ability to perform well on previously unobserved inputs is called generalization.

Then one question might be asked is how could we affect the performance of the machine learning model on unseen inputs, given the observed training set only? The statistical learning theory provides some answers. There is a set of assumptions called the *i.i.d. assumptions*. They say all the samples in each dataset are *independent* from each other, and that the train set and test set are *identically distributed*, drawn from the same probability distribution as each other. We call this shared underlying distribution the *data generating distribution* [9]. Under this assumption, we could see that for the training set and test set sampled from the same data generating distribution, the expected training score of the machine learning model should be the same as the model's test score. This help us build a bridge between the training score and test score. In practice, when we apply a machine learning algorithm, we do not fix the model ahead of time, then check its performance on both training and test set. Typically, we first sample a training set, then use it to find a model with lower training score, then sample the test set. Starting from this point, the expected test score of a machine learning model is greater than or equal to its expected training score. This fact actually helps us to divide the performance of a machine learning model into two factors:

1. How could we improve the training score?

2. How could we make the gap between training score and test score as

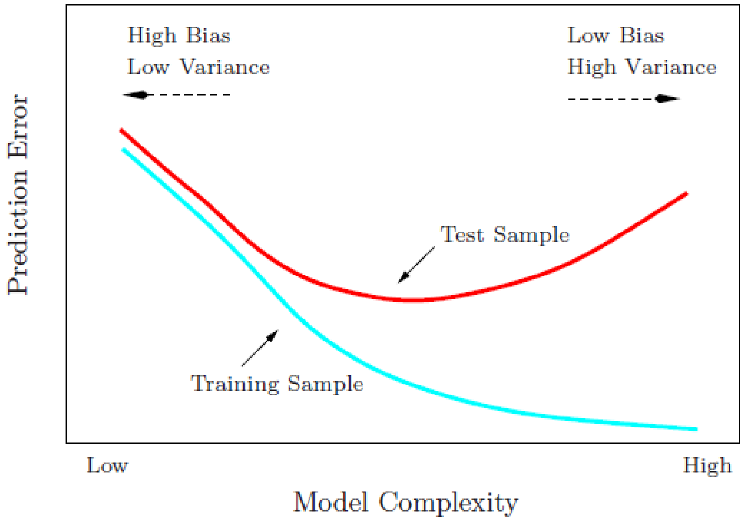


Figure 2.1: A general relation between training score and generalization score

small as possible?

A perfect machine learning model should hold both small training score and small gap between training score and test score. But under a lot situation in practice, it is hard to find a perfect way to reduce these two scores simultaneously. We often face a tradeoff between these two factors.

2.3 Underfitting and Overfitting

From the perspective of machine learner, there are two kinds of problems related to the two factors mentioned about, Underfitting and Overfitting.

Underfitting reveals the problem of our machine learning model that its capacity is not powerful enough to deal with this specific task. It often represents in the form that the model's training score are not good enough and the gap between training score and generalization score is often not too large in the scale of training score. Overfitting reveals the problem that the complixity of our machine learning model is beyond the complixity of the task. It usually represents in the form that the model's generalization score are much worse than its training score. This means our model trained on the training set captures meaningless information by mistake from the training set and this kind of information is harmful for getting a good generalization score on unseen dataset. This kinds of harmful information hid in training set has many faces. Noise is one example.

So these are two important potential problems we can read from the performance of the machine learning model on training set and test set. It can help us diagnose our model. But the reason behind the phenomenon is complicated. One way to understand it is from the statistical view.

For the performance score, mean square error (MSE), it is easy to get a useful representation formation through some calculation in statistics.

$$MSE(\hat{\theta}) = bias^2(\hat{\theta}) + var(\hat{\theta}) \tag{2.1}$$

$\hat{\theta}$ represents an estimator in statistics. In machine learning, it can be viewed as our model's parameters. What we hope is approaching to the true parameters of the underlying data generating distribution. The first term in the above equation represents the bias of the estimator, which is a statistics representing the difference between the expectation value of our estimator and a perfect estimator. The second term is called the variance of the estimator, which is a statistics representing the uncertainty or stability of the estimator. For a machine learning model, if we view our model as an estimator, then the bias is a learners tendency to consistently learn the same wrong thing, which is related to the training score of our model, and the variance is the tendency to learn random things irrespective of the real signal, which is related to the gap between the training score and generalization score.

2.4 No Free Lunch Theorem

However there is no universal trick that is able to reduce the bias and variance simultineously without more knowledge about the data. The reason is given by the *no free lunch theorem* [10]. This theorem states that "Any two optimization algorithms are equivalent when their performance is averaged

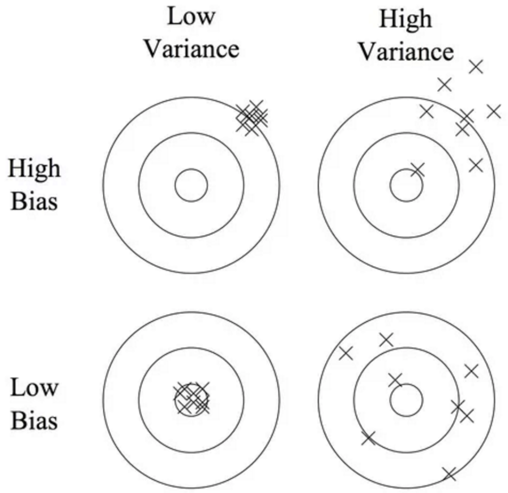


Figure 2.2: An illustration of bias and variance

across all possible problems". In other words, no machine learning algorithm is universally any better than any other, even the random guess. So we always need embody some knowledge or assumptions beyond the given data to do meanful generalization beyond it. The *i.i.d assumption* we mentioned about is one example. There is a quite good comment given in [11]:

"In retrospect, the need for knowledge in learning should not be surprising. Machine learning is not magic; it cant get something from nothing. What it does is get more from less. Programming, like all engineering, is a lot of work: we have to build everything from scratch. Learning is more like farming, which lets nature do most of the work. Farmers combine seeds with nutrients to grow crops. Learners combine knowledge with data to grow programs."

2.5 Regularization

The no free lunch theorem seems rather depressing. But fortunately, the problems we human beings care about are not uniformly distributed in the mathematic-possible space. We can still do something meanful under reasonalbe assumptions. In fact, even very general assumptions are helpful enough for the success of machine learning methods.

Back to the underfitting and overfitting problem we mentioned about. Underfitting is usually not allowable. Because it means there is no way for a model to get a good performance, since the problem is beyond its optimal capacity of this model. And overfitting problem is usually easy to deal with. We can add more params into our model to make it more complicated. So typically we design a powerful enough model and have it trained on the dataset. Then we need to find some ways to avoid overfitting. A common idea is called regularization.

A more complicated model means a larger model searching space, which is also called hypothesis space. It should be remembered that we do not fix a specific model for a machine learning system. We set a limit of the hypothesis space and let our machine learning algorithm to search an optimal model in the space. But there is no guarantee that an optimization algorithm used in our machine learning system is able to find the best solution. And usually it cannot. So one way we can do is to give some additional hints to the algorithm to help it get larger chance of finding a better solution. This is the idea of regularization. The regularization term can be viewed as a penalty on our unpreferred solution. This means that if both solutions are eligible before regularization, now one is preferred to another. The unpreferred solution may only be chosen if it fits the training set significantly better than the preferred solution.

2.6 Validation and Hyperparameter

One way to evaluate the generalization score of the machine learning model is to use a technique called cross validation. The idea behind the cross validation method is quite straightforward. Take a traditional 10-fold cross validation as an example. The original dataset is evenly and randomly divided into 10 folds. For each iteration of a overall 10 iterations of evaluation process, one fold is left aside. The machine learning is trained on the training set composed with the remaining 9 folds of data and then evaluated on the test set left aside ahead of time. After 10 iterations, we can get 10 generalization scores. Usually we use the average score as the final generalization score of our model on this dataset. The algorithm of cross validation is illustrated in 1

Most machine learning algorithms have settings that we can use to control the behavior of the learning algorithm. These settings are called hyperparameters [9]. The values of hyperparameters are not optimized by the learning algorithm itself (though we can design a nested learning procedure where one learning algorithm learns the best hyperparameters for another learning algorithm).

Sometimes a setting is chosen to be a hyperparameter that the learning algorithm does not learn because it is difficult to optimize. More frequently, we do not learn the hyperparameter because it is not appropriate to learn that hyperparameter on the training set. This applies to all hyperparameters that control model's capacity [9]. If learned on the training set, such hyperparameters would always choose the maximum possible model capacity, resulting in overfitting, like 2.2. We can always fit the training set better

Algorithm 1 Cross Validation Algorithm

```
Define:   xVal($X, A, L, k$)
Require: $X$, the given dataset;
Require: $A$, a machine learning algorithm, take training dataset as input
  and output a trained model $f(X')$ ;
Require: $L$, a loss function, used to evaluate the performance of a trained
  model $f$ on some test set $X'$;
Require: $k$, the given number of folds
  Divide $X$ into $k$ mutually exclusive subsets $X_i$, whose union is $X$.
  for $i$ from 1 to $k$ do
    $f_i \leftarrow A(X - X_i)$
    $e_i \leftarrow L(f_i, X_i)$
  end for
  return  $e$
```

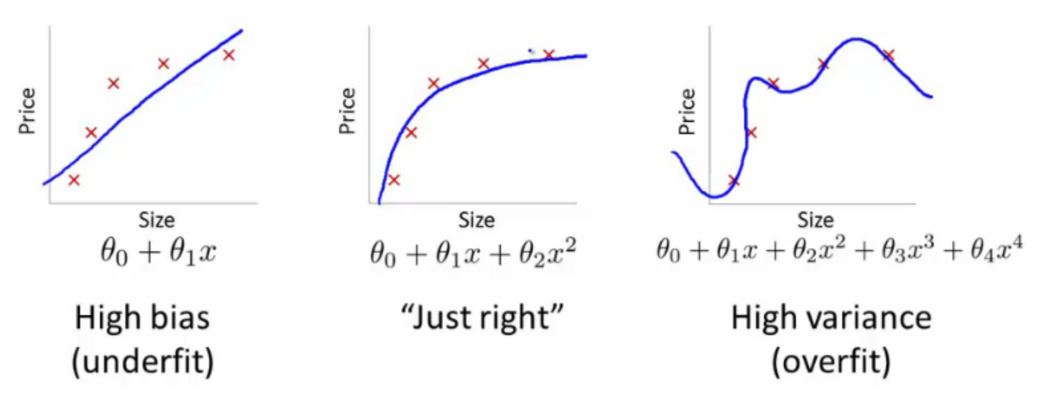


Figure 2.3: An illustration of overfitting with too high degree in polynomial model

with a higher degree polynomial and a weight decay setting of $\lambda = 0$.

To search a better hyperparameters setting, we can use a validation set to evaluate choices of hyperparameter values. The idea is similar to cross validation. We can do another cross validation in the inner loop with a validation set separated from the training set. The algorithm of double cross validation is illustrated in 2 This validation set plays a similar rule as the test set in the outer loop. The difference is the aim of outer loop of cross validation is to evaluate the generalization score of our model, while the inner cross validation in the inner loop is used to search a good setting of hyperparameters.

These two techniques are widely used in a lot machine leanring training and evaluation process. They are also used in our work.

Algorithm 2 double Cross Validation Algorithm

```
Define:  dxVal($X, A, L, k, d$)
Require: $X$, the given dataset;
Require: $A = \{A^j\}_{j=1}^{C}$, a set of machine learning algorithm with different hyperparameter setting. Each $A^j$ takes training dataset as input and output a trained model $f^j(X')$ ;
Require: $L$, a loss function, used to evaluate the performance of a trained model $f$ on some test set $X'$;
Require: $k$, the given number of folds in the outer loop
Require: $d$, the given number of folds in the inner loop
  Define: $A'(X')$:
  for $j$ from 1 to $C$ do
    $e_j = mean(kFoldXV(X', A_j, L, d))$
    $j^\star \leftarrow \arg\min_j e_j$
  end for
  return  xVal($X, A', L, k, d$)
```

2.7 Classifiers

The supervised learning means there are label information included in the data set and the machine learning model is required to make prediction on the label values for some unseen data. If the label values are in a discrete space (often a set of finit k possible choices), then it is called classification problem. Otherwise, if the label values changes in a continuous space, we call it a regression problem.

Here we mainly focus on the classification. The classification problem is that given a data set with label values in discrete space, the machine is asked to specify which of some k possible categories the input belongs to. For example, given a group of weather indices in the past several days and their corresponding weather records, a classification problem can be asking the machine to predict tomorrow's weather based on similiar weather indices.

Given a data set $X = \left\{X^{(i)}\right\}_{i=1}^{N}$ where $X_{\mathrm{i}} \in \mathbb{R}^D$, which means each data sample X_i is in a D-dimensional feature space, and corresponding label values $y = \left\{y^{(i)}\right\}_{i=1}^{N}$, where $y^{(i)} \in \mathbb{Z}_k$, which means there are k choices of each label value $y^{(i)}$, a classification learner, or called a classifier, can be viewed as a function $f : \mathbb{R}^D \rightarrow \mathbb{Z}_{\mathrm{k}}$. The classifier is trained on (X, y) and make prediction y' on some unseen data X'. The performance score of the classifier is evaluated by some loss function, like mean square error (MSE). There is a special case that if there are only two choices in the label value space y, i.e., $k = 2$ and $y^{(i)} \in \mathbb{Z}_2 = \{0, 1\}$, then this kind of task is called binary classification problem.

Here we give an introduction in detail to several popular machine learning classifiers. These classifiers are also applied in our work. Their performance on our dataset will be shown later.

Adaboost

Adaboost is a boosting tree model. The original idea is straightforward. It constructs a collection decision tree models iteratively on a collection of boosting generated samples. Every sample is initialized with an equal weight. A simple decision tree is built in the initial step. Then in each iteration, the tree performance α_m will be evaluated on the weight sum of misclassification.

$$\mathrm{err}_m = \sum_{i=1}^{N} w_i I\{y_i \neq G_m(x_i)\} \tag{2.2}$$

$$\alpha_m = \log((1 - \mathrm{err}_m)/\,\mathrm{err}_m) \tag{2.3}$$

where $w_i \in [0, 1]$ is the sample weight sum to one.

$$w_i \leftarrow w_i \cdot \exp\{\alpha_m \cdot I\{y_i \neq G_m(x_i)\}\tag{2.4}$$

According to the error score, each weight of misclassified sample will be updated respectively by Eq. 2.4. This process has a natural explanation like the learning process of our human beings. At first, we can learn simple parts fast and directly. Then we will pay more and more attention to those hard parts which we do not understand very well in the past.

In our experiments, to optimize hyper-parameters for AdaBoost, we search the depth (d) of each decision tree in 1, 2, 3, 4 and the number of weak classifiers from 10 to the whole feature size with an increment of 120/d.

GBDT

Gradient boosting decision tree is another kind of additive boosting model which in general can be expressed as Eq. 2.5

$$f(x) = \sum_{m=1}^{M} \beta_m b(x; \gamma_m) \tag{2.5}$$

where b_m are called expansion coefficients, like weight of tree in each iteration, and $b(x; \gamma_m)$ are usually simple basic functions(e.g.: decision tree) characterized by parameters γ_m. Then the training target becomes Eq. 2.6[12]

$$\min_{\{\beta_m, \gamma_m\}_1^N} \sum_{m=1}^{M} L(y_i, f(x_i)) \tag{2.6}$$

where $L(y_i, f(x_i))$ is some loss function. For most loss functions, Eq. 2.6 is a computationally intensive task. While it can be approximated by Forward stagewise modeling. The process is sequentially adding new basis functions to the expansion without adjusting the parameters and coefficients of those that have already been added. It is outlined in Algorithm 3

Algorithm 3 Forward Stagewise Additive Modeling

```
1. Initialize $f_0(x) = 0$.
2. For m = 1 to M:
        a) Compute
           $\arg\min\limits_{\beta,\gamma} \sum_{m=1}^{M} L(y_i, f_{m-1}(x_i) + \beta b(x_i;\gamma))$.
        b) Set $f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$
```

The adaboost can be transferred to a specific instance of that with exponential loss function.The main idea of GBDT is using the gradient of loss function as the addative step direction. Then build a tree approximating the gradient descent effect as the addative tree in each iteration. Unlike the Adaboost, it provides more freedom on choosing loss functions for the additive boosting model, without losing much training speed. The algorithm is outlined in Algorithm 4[12]

In our experiments, we optimize the hyper-parameters for GBDT by searching the number of trees among 10, 100, 200, 500, 1000, 2000 and the

Algorithm 4 Gradient Tree Boosting Algorithm

```
1. Initialize $f_0(x) = \mathrm{argmin}_\gamma \sum_{i=1}^{N} L(y_i, )$.

2. For m = 1 to M:

    a) For $i = 1, 2, \ldots, N$ compute:

       $r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}$

    b) Fit a regression tree to the targets $r_{im}$ giving terminal regions $R_{jm}$,

       $j = 1, 2, \ldots, J_m$.

    c) For $j = 1, 2, \ldots, J_m$ compute

       $\gamma_{jm} = \mathrm{argmin}_\gamma \sum L(y_i, f_{m-1}(x_i) + \gamma)$.

    d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$.
```

learning rate in 1, 0.1, 0.01, 0.001, 0.0001.

Random Forest

Random forest is a quite popular machine learning method in recent years. It has advantages like easy and fast training, being robust for overfitting problem, competitive performance on different kind of data sets for different matrices [13]. Random forest is a substantial modification of bagging that builds a large collection of de-correlated trees, and then averages them[12]. Each classifier in the ensemble is a decision tree trained on a bootstrap sample set of original data and when growing the tree, select a random subset of attributes as candidates for splitting at each node. The algorithm is outlined in Algorithm 5

In our experiments, we optimize hyper-parameters for RF by searching the number of trees in 10, 100, 200, 500, 225 1000, 2000 and searching the subset size of features for node splitting among sqrt, 100, 200, 500, 1000, 2000 where sqrt is the square root of the whole feature size.

Logistic regression is a kind of generalized linear model. For a binary classification problem, with labeled sample set $\{(x_i, y_i)\}_{i=1}^{N}$, it adds a sigmoid function (2.7) on the linear function $z(x) = w \cdot x + b$ to limit the response region in the range $(0, 1)$ (Fig. 2.4). This response value can be intuitively

Algorithm 5 Random Forest for Classification

```
1. For b = 1 to B:
    a) Draw a bootstrap sample $Z^\star$ of size $N$ from the training data.
    b) Grow a random-forest tree $T_m$ to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $n_{\text{min}}$ is reached.
        i. Select m variables at random from the $p$ variables.
        ii. Pick the best variable/split-point among the $m$.
        iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^M$.

Final classification:

$\widehat{C}_{\text{rf}}^{M}(x) = \text{majority vote}\{\hat{C}_m(x)\}_1^M$

where $\hat{C}_m$ be the class prediction of the $m$th random-forest tree.
```

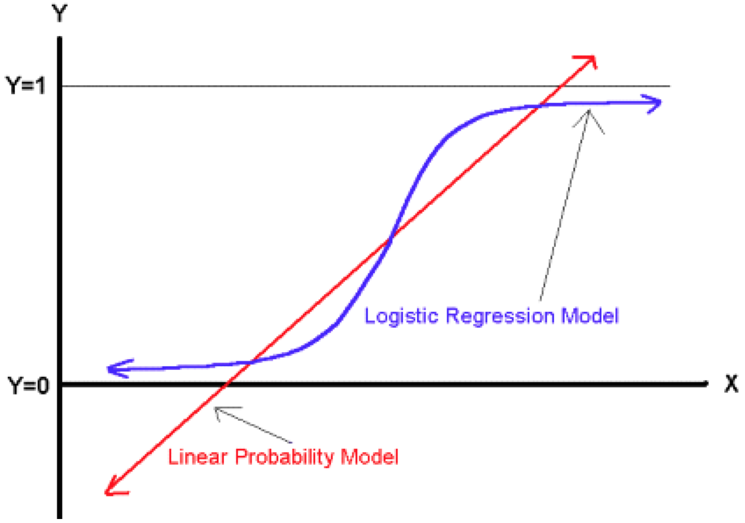


Figure 2.4: Sigmoid function in logistic regression

interpreted as the probability of the positive prediction. With mean square error as performance measure, the model can be optimized to minimize the loss function(2.8).

$$P_1(x_i) = \sigma(z(x_i)) = \frac{1}{1 + exp(-z(x_i))} \tag{2.7}$$

$$L(w) = -\frac{1}{N}[\sum_{i=1}^{N} y_i log P_1(x_i) + (1 - y_i) log(1 - P_1(x_i)] \tag{2.8}$$

In our experiments, we use the batch gradient descent algorithm with L2 regularization to train the model. The strength of regularization is searched from 10^{-5} to 10^5, with an increment of 1 for the exponent.

Multilayer perceptron (MLP) is a feedforward neural network. It can be viewed as a hierarchical nonlinear combination of logistic regression (Fig. 2.5).

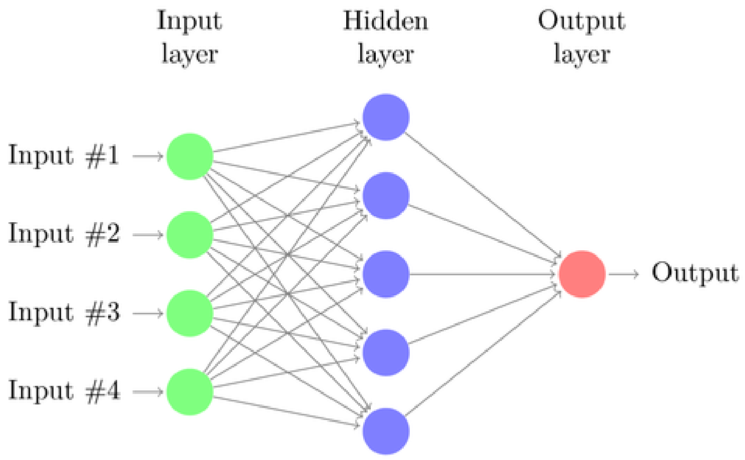


Figure 2.5: Architecture of the simplest three-layer MLP

MLP uses layerwise connected nodes to build the architecture of the model. Each node(except for the input nodes) can be viewed as a neuron with a nonlinear activation function. In our work, we use the simple sigmoid function(2.9) as the activation function,

$$\sigma(z(x)) = \frac{1}{1 + exp(-z(x))} \qquad z(x) = w * x + b \tag{2.9}$$

where the weight vector w and bias vector b in each layer pair are trained by the Back Propagation algorithm. We also introduce L2 regularization weight decay to prevent overfitting. We optimize parameters for MLP by searching the hidden layer size in {2, 3}, the hidden unit size in {0.0625*m, 0.125*m, 0.25*m} where m is the feature size 2538, and searching the weight decay strength among {0.0005, 0.0001, 0.00001, 0.0}.

Support vector machines (SVM) is one of the most influential approaches to supervised learning (Boser et al., 1992; Cortes and Vapnik, 1995). It is also one of the most widely used classifiers in medical image analysis [3, 7, 14, 15]. Similiar to Logistic Regression, it employs a linear funtion $w^T x+b$. But unlike Logistic Regression, SVM originally does not provide a probability outcome. It outpus 0 or 1 to indicate the final classification result directly.

The underlying geometrical idea is to find a hyperplane maximizing the margin to support vectors. The support vectors are those sample vectors which are closest to the hyperplane. So the model chooses optimal hyperplane completely decided by the support vectors. In other words, the choice hyperplane will not be influenced by those sample vectors far away from the hyperplane. Since the hard margin is usually not accessible, it allows soft margin which allows more sample vectors to becomes support vectors. When the original feature space is almost separable (without using kernal method), the target to optimize is

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \varepsilon_i \qquad s.t. \quad \forall i, \quad \varepsilon_i \geq 0, \quad y * (\langle w, x \rangle + b) \geq 1 - \varepsilon_i \tag{2.10}$$

It can be transferred equivalently to the form

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \max(0, 1 - y * (\langle w, x \rangle + b))) \tag{2.11}$$

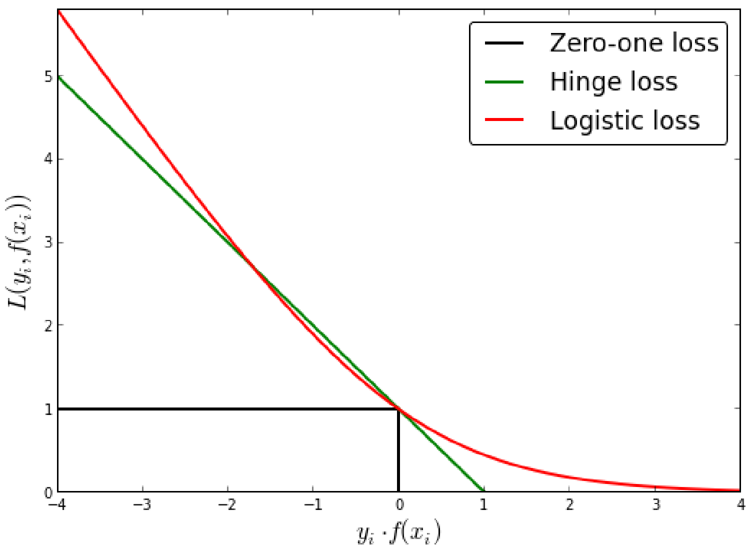


Figure 2.6: Illustration of logistic loss and hinge loss

The first term is a traditional l2 regularization. And the second term is called high loss. Compared with logistic regression, from the perspective of optimization process, we can find the largest difference between them is the choice of loss function. Fig. 2.6 shows the similarity and difference in an intuitive way.

One key innovation associated with support vector machines is the kernel trick. The kernel trick consists of observing that many machine learning algorithms can be written exclusively in terms of dot products between examples. For example, the linear function used in support vector machine can be transferred to the format

$$w \cdot x + b = \sum_{i=1}^{N} \alpha_i x \cdot x^{(i)} + b = \sum_{i=1}^{N} \alpha_i \langle x, x^{(i)} \rangle + b \tag{2.12}$$

where $x^{(i)}$ is a training sample and α is a vector of coefficients. The $\langle x, x^{(i)} \rangle$ represents the inner product of vectors x and $x^{(i)}$. The idea of kernel method is to apply a kernel function $k(x, x) = \langle \phi(x), \phi(x^{(i)}) \rangle$ on the inner product part. Then the former linear function becomes

$$z(\phi) = \sum_{i=1}^{N} \alpha_i \cdot \langle \phi(x), \phi(x^{(i)}) \rangle + b \tag{2.13}$$

We can see the new function z is nonlinear to x and linear to $\phi(x)$ and α. So under this process, we can transfer our original feature space to another feature space (often a higher dimensional space) and apply the linear function in a similar way as in the original feature space. With the inner product part being computed ahead of time, this method is also relatively computational efficient. So the kernel method assists the support vector machine to solve those nonlinearly separable problems effectively and efficiently.

There are several kinds of kernel functions. Those most commonly used in SVM are:

$$k(x,x) = \begin{cases} x \cdot x^{(i)} & Linear \\ (\gamma \cdot x \cdot x^{(i)} + b)^d & Polinomial \\ exp(-\gamma \cdot \left|x - x^{(i)}\right|^2) & Gaussian(RBF) \\ tanh(\gamma x \cdot x^{(i)} + C) & Sigmoid \end{cases} \tag{2.14}$$

In our experiments, we find liner kernel is better than others and optimize the hyper-parameter C. Let $C = 2^m$, we search m in the range [-8, 9] with a step increment of 1.

k-Nearest Neighbors (kNN) is one of the most common lazy classifiers, which classifies a new instance by a majority vote of its k nearest neighbors. In this paper, we use the Euclidean distance metric to find the k nearest neighbors. We search the optimal k value for our task in the range [1, 50] with a step increment of 1.

Chapter 3

Cervigram Image Analysis

Here we have a task that taking cervigram images as input, we need build a system which is able to predicting the corresponding CIN level. This task can be viewed as a supervised learning problem and since the value of CIN level is in discrete space, it is a classification problem.

The raw image data is in a quite high dimension. The information each pixel in an image can represent is limited, which is not suitable for applying traditional machine learning models directly. Fortunately, in the area of computer vision and image processing, there are various mature technologies and tricks that can be used to analyze an image. As we have mentioned before, the machine learning is not a magic. It should be designed to make fully use of existed experience and human beings' knowledge about the task.

We design a type of pyramid features. From each image, three complementary pyramid features are extracted, including Pyramid histogram in L*A*B* color space (PLAB), Pyramid Histogram of Oriented Gradients (PHOG), and Pyramid histogram of Local Binary Patterns (PLBP). With these sets of high-level features, we apply machine learning methods to this classification task. As we know, there is no universally best machine learning method for any tasks in general. We are interested in what kind of machine learning methods are more suitable for our problems in medical image analysis.

The seven classifiers introduced in section 2 are chosen to evaluate their performance on this task. We train binary classifiers to separate CIN1/Normal and CIN2/3+ images. All the classifiers are trained and tested on the same dataset, with a uniform parameter optimization strategy. They are then compared by ROC curves and other evaluation measures.

On the same dataset, our lower-cost image-based classifiers can perform comparably or better than human interpretation on other traditional screening results, such as Pap tests and HPV tests.

3.1 Cervigram Image Database

For our image-based CIN classification problem, here we first introduce the dataset we used, built from a large medical data archive collected by the National Cancer Institute (NCI) in the Guanacaste project [16]. The archive consists of data from 10,000 anonymized women, and the data is stored in the Multimedia Database Tool (MDT) developed by the National Library of Medicine [17]. In the archive, each patient typically had multiple visits at different ages. During each visit, multiple cervical screening tests including Cervicography were performed. The Cervicography test produced two cervigram images for a patient during her visit and the images were later sent to an expert for interpretation.

In our dataset, we collected 1112 patient visits, 345 positive (CIN2/3/cancer) and 767 negative (CIN1/Normal). For each patient, the ground truth diagnosis is based on the Worst Histology result of that patient visit: multiple expert histology interpretations were done on each biopsy; the most severe interpretation is labeled the Worst Histology for that visit in the database. Note that our dataset is imbalanced, i.e. it contains more negative cases than positive cases. Since many classification methods assume a balanced distribution of classes and require additional strategies to handle imbalanced

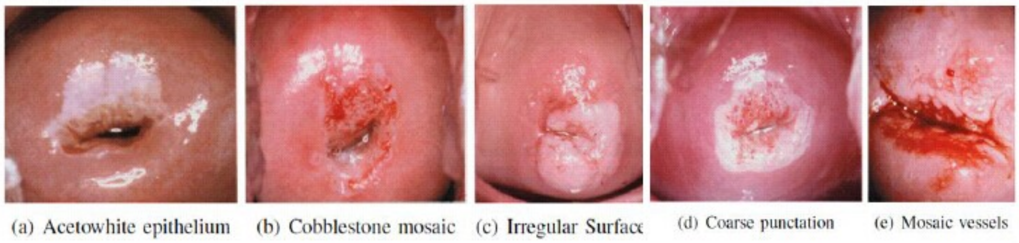


Figure 3.1: Illustration of visual observations in cervigrams

data, we apply undersampling to the negative visits and randomly choose 345 negative visits from each dataset. In our work, we use this balanced sub-dataset, including all 345 positive visits and the randomly selected 345 negative visits.

Interpretations based on cervigram images have been shown to be an effective way to detect CIN2/3+ [3]. Some of the most important visual observations in cervigrams include the acetowhite region, and features within that region, such as mosaicism, punctation, and atypical vessels; it is important to distinguish these possibly disease-related features from benign features such as polyps or cysts. Fig. 3.1 shows some example images of those observations [18]. To robustly identify these characteristics which are helpful for diagnosis, we propose a type of hand-crafted pyramid features.

3.2 Feature Extraction

With the technology of computer vision and image analysis, we get a way to represent our knowledge about the image data in the format of a high dimensional feature space. We extract multi-scale pyramid histogram features to encode the statistical appearance information in cervigrams, as shown in Fig. 3.2. First, we isolate the cervix region of interest (ROI) from the input image and resize it to 300*250 pixels. We use the method proposed in [3] to segment the ROI. Second, we transform the ROI image patch into different types of feature maps, including the local binary pattern (LBP) map, L*A*B color channels, and the image gradient maps. Third, a spatial pyramid of sub-regions is constructed for each feature map. Based on these constructed pyramids, pyramid LBP (PLBP), pyramid LAB (PLAB) and pyramid Histogram of Oriented Gradients (PHOG) features are extracted and concatenated to be a multi-feature descriptor.

3.2.1 Color Feature

Color plays an important role in cervical lesion classification, because one of the most important visual features on the cervix that have relevant diagnostic properties is the presence of Acetowhitened regions. Thus, the color feature

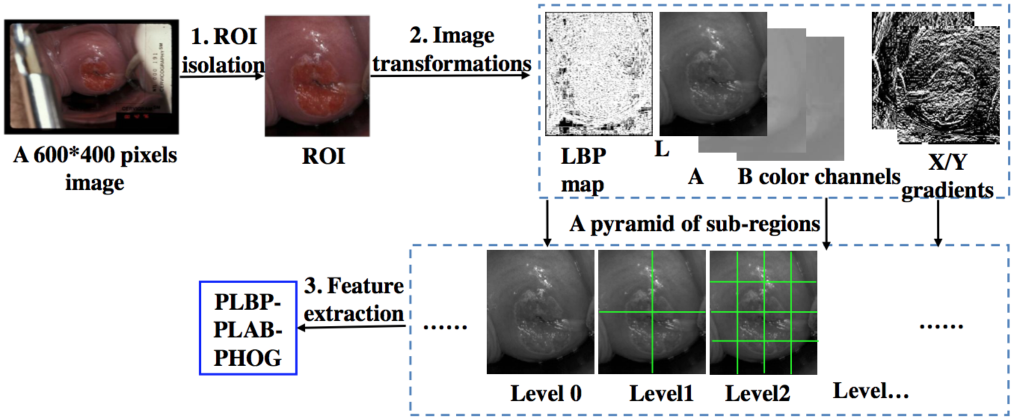


Figure 3.2: Image features extraction

is widely used in cervigram analysis [2, 7, 12]. We calculate the L*A*B color channels as our color feature maps. Then, to capture edge and shape information on a cervix, we calculate the gradient map, which is shown to be complementary to the color feature [2, 7].

3.2.2 Texture Feature

In addition to the color and gradient features, we introduce a local binary pattern (LBP) feature that extracts local texture characteristics for cervical lesion classification. Ojala et al. [19] first introduced LBP and showed its powerful ability for texture classification. In a local neighborhood of an input image, given a pixel (x_c, y_c) which is surrounded by 8 neighbors, we can calculate its LBP value by Eq. 3.1,

$$LBP(x_c, y_c) = \sum_{p=0}^{7} s(i_p - i_c) \cdot 2^p \tag{3.1}$$

Where i_c indicates the grayscale value of the center pixel (x_c, y_c); i_p corresponds to the grayscale value of the pth neighbor. $s(x)$ is a sign function where $s(x) = 1, if \quad x \geq 0; else, s(x) = 0$.

Later, several extensions of the original LBP operator were presented [20]. First, the LBP was extended to a circular neighborhood of different radii, denoted as $LBP_{P,R}$ which refers to P equally spaced pixels on a circle of radius R. Furthermore, the rotation invariant local binary pattern is defined in Eq. 3.2,

$$LBP_{P,R}^{ri} = \min_i \{ROR(LBP_{P,R}, i)\}_{i=1}^{P-1} \tag{3.2}$$

Where $ROR(LBP_{P,R}, i)$ performs a circular bitwise right shift on the P-bit $LBP_{P,R}$, for i number of times.

To obtain the LBP map, we compute the $LBP_{P,R}^{ri}$ value for each pixel in the input image. Because of the neighborhood constraints when capturing $LBP_{P,R}$ features, pixels on the boundary of the input image within the R range do not have any LBP values. We set those pixels values to be zeros or to be their closest neighbors LBP values.

In our work, we use $LBP_{8,1}^{ri}$. There is no need to use LBP with other radii because our pyramid histogram LBP feature (PLBP) can encode a multi-scale local binary pattern.

As Fig. 3.1 shows, we construct a spatial pyramid for each feature map. A pyramid is constructed by splitting the image into rectangular sub-regions, increasing the number of regions at each level, i.e., level 0 has 1 sub-region; level 1 has 4 sub-regions; level 2 has 16 sub-regions, and so forth. Histogram features are extracted within these pyramid sub-regions. The extracted pyramid histogram encodes the statistical distribution of feature values at different positions and scales in cervigrams.

For the PLBP feature, the total number of bins is 10 for the histogram of a subregion. A 4-level of pyramid is constructed resulting in a PLBP histogram feature that has 850 dimensions. For the PLAB feature, we extract 3 pyramid levels with a 16-bin histogram for each channel in L*A*B color space in each subregion. Thus, the PLAB color feature has 1,008 dimensions. In the gradient map, we calculate pyramid histogram of oriented gradients (PHOG). An 8-bin orientation histogram over 4 levels is used. Hence, the total vector size of our PHOG feature is 680. Finally, we construct a multi-feature descriptor by concatenating the three different types of features, PLBP-PLAB-PHOG. Thus, this handcrafted multi-feature descriptor has a vector size of 2,538.

Chapter 4

Experiments

In Section 3.1, we described the construction of two cervigram image datasets, D1 and D2, where each one contains 345 images from positive (CIN2/3+) patient visits and 767 images from negative (CIN1/normal) patient visits. Note that the datasets are imbalanced, i.e. they contain more negative cases than positive cases. Since many classification methods assume a balanced distribution of classes and require additional strategies to handle imbalanced data, we apply undersampling to the negative visits and randomly choose 345 negative visits from each dataset. The resulting two balanced datasets, D_1^{bal} and D_2^{bal}, use all 345 positive visits and the randomly selected 345 negative visits.

We conduct experiments to compare the seven classifiers described in Section 2.7, on the two balanced datasets D_1^{bal} and D_2^{bal}, and on the two larger imbalanced datasets, D1 and D2. The classifier implementations we use are from well known open source libraries. Our Random Forest, GBDT, and LR classifiers are implemented with scikit-learn [21]; the MLP classifier is provided by pylearn2 [22]; the SVM is offered by Libsvm [23]; the AdaBoost is provided by Appel et. al. [24]; and the kNN classifier is provided by the implementation in MATLAB.

We perform the same ten-round ten-fold cross validation using these seven classifiers. On each dataset, we randomly divide the samples (cervigrams) into ten folds. In the ten rounds, we rotationally use one fold for testing and nine folds for training. On the training set, we use a uniform strategy, Exhaustive Grid Search [23], to search for the optimal parameters of each classifier. Three cross validations are used in the parameter searching process. The exact parameters and search ranges for each classifier are discussed in the Section 3.

The results of the ten rounds are used to draw ROC curves. We compare different classifiers by analyzing their ROC curves, areas under ROC curves (AUC), and accuracy, sensitivity and specificity values at the point where the probability threshold is 0.5. We also compare the results of our image-based classifiers with several other screening tests results, obtained for the same visits that are used to construct our datasets.

Table 4.1: Overall AUC and accuracy (accu), sensitivity (sensi) and specificity (speci) at the default threshold on the balanced dataset D_1^{bal} and the imbalanced dataset D_1

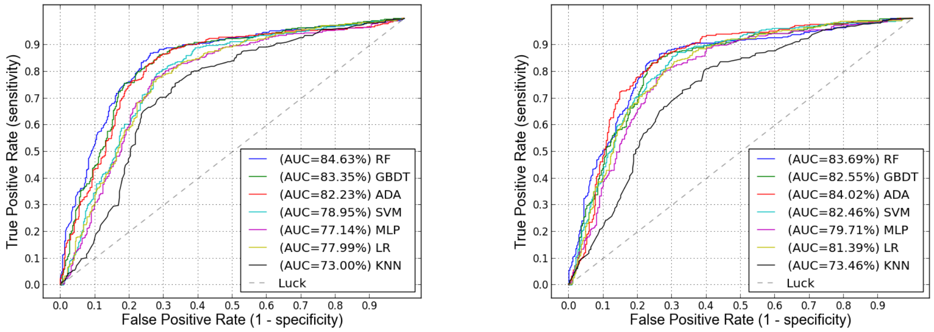
	D_1^{bal}				D_1			
Classifier	AUC(%)	accu(%)	sensi(%)	speci(%)	AUC(%)	accu(%)	sensi(%)	speci(%)
RF	84.82	80.00	84.06	75.94	84.83	78.24	67.54	83.05
GBDT	84.30	78.55	82.03	75.07	82.28	77.07	62.61	83.57
AdaBoost	82.23	76.81	77.68	75.94	82.53	76.44	57.97	84.75
SVM	78.95	74.78	76.52	73.04	79.82	74.37	46.67	86.83
LR	77.99	74.20	76.23	72.17	79.99	75.45	54.20	85.01
MLP	77.10	75.27	77.78	72.75	78.60	76.53	59.13	84.35
kNN	73.00	70.87	75.07	66.67	74.38	71.67	48.12	82.27

4.1 On Balanced Datasets

In our first set of experiments, we compare seven classifiers on the balanced dataset D_1^{bal} and D_2^{bal}. The comparison results are shown in Fig. 4.1 as ROC curves and in Table 4.1 with overall AUCs, and accuracy, sensitivity and specificity values at the default probability threshold 0.5. The ROC curves illustrate that the three ensemble-tree models RandomForest (RF), GBDT, and AdaBoost outperform other classifiers. AUCs in 4.1 also show that the ensemble-tree models have a better overall performance. At the 5% significance level, there is no difference between RandomForest, GBDT and AdaBoost. On D_1^{bal}, for instance, the $p-value$ is 0.0708 by paired t-test between RF (1st rank) and AdaBoost (3rd rank). However, these three ensemble-tree classifiers are significantly better than all other classifiers. On D^{bal}, the $p-value$ is 0.0062 and 1.7191 * 10^{-4}, by paired t-test between RF (1st rank) and SVM (4th rank), and between RF and kNN (lowest rank), respectively. We conjecture that the ensemble-tree models perform best because they are more robust to over-fitting than other models such as SVM and MLP when dealing with scalar data sets that are not too large.

4.2 On Imbalanced Datasets

We also conduct the same ten-round ten-fold experiments on the imbalanced datasets D1 and D2. The results are shown in Fig. 4.2 and Table 4.1. One clear difference between results on the imbalanced datasets and those on the

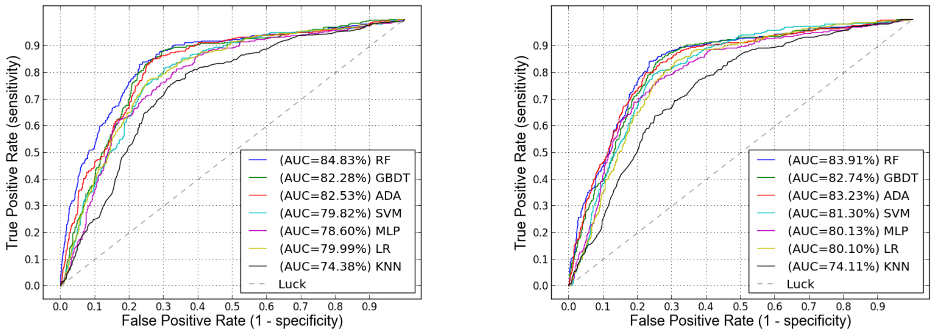


(a) ROC curves on D_1^{bal}

(b) ROC curves on D_2^{bal}

Figure 4.1: ROC curves on balanced datasets D_1^{bal} and D_2^{bal}.

balanced datasets is that, at the same default threshold, all seven classifiers give higher specificity values and lower sensitivity values on the imbalanced dataset (see Table 1, right column). This is expected since in the imbalanced datasets, there are more negative samples than positive samples, thus when penalizing equally errors on samples from any class and training to minimize the overall classification error, the classifiers trained on the imbalanced data become biased to the class with a majority of samples. Interestingly, since higher specificity is a desired property for a clinical test meant for screening, training classifiers on the imbalanced dataset (which more closely reflect the true underlying patient distribution) can be beneficial. Moreover, Fig. 4.2 shows that the overall ROC curves and AUCs on the imbalanced datasets are similar to that on the balanced datasets. Although more samples are



(a) ROC curves on D_1

(b) ROC curves on D_2

Figure 4.2: ROC curves on imbalanced datasets D1 and D2.

used to train classifiers on the imbalanced datasets, the overall performance by the classifiers did not seem to improve.

4.3 Cervigram Based RandomForest vs. Pap and HPV Tests

In this experiment, we first compute the average result of our image-based classifier RF to represent its visit-level performance on balanced and imbalanced datasets, respectively. We then compare the visit-level result of RF with Pap and HPV tests results, which are available for the same visits that are used to construct our datasets. As illustrated in Table 4.2, on both datasets the image-based RF classifier outperforms every single Pap test or

Table 4.2: Comparing visit-level sensitivity (sensi) and specificity (speci) of image-based RF classifier with that of Pap tests and HPV tests.

	Balanced dataset		Imbalanced dataset	
Method	sensi(%)	speci(%)	sensi(%)	speci(%)
Alfaro ThinPrep	20.69	81.82	20.69	85.27
Cytyc ThinPrep	49.55	88.46	49.55	89.77
Costa Rica Pap	39.42	88.12	39.42	89.31
Hopkins Pap	36.00	97.11	36.00	97.13
HPV16	33.82	94.19	33.82	92.49
HPV18	08.16	97.97	08.16	98.17
Cervigram based RF	51.00	90.00	49.00	90.00

HPV test at specificity around 90%.

Chapter 5

Conclusion and Future Work

In our work, we treat the problem of cervigram image analysis as a machine learning classification task and apply several popular machine learning algorithms to make predictions about the CIN level. From our experiments, it has been shown that machine learning is a useful tool to deal with this problem.

We also make a comprehensive comparison among several popular machine learning classifiers to figure out which one might be a suitable model for this kind of problem. We use ROC curve and the AUC score as the final performance measure. From the results we get, we find that ensemble-tree models—Random Forest, Gradient Boosting Decision Tree, and AdaBoost—

outperform other classifiers such as multi-layer perceptron, SVM, logistic regression and kNN, on this task. This finding is consistent with the conclusion in other works [25]. Another finding is that, training and testing on the larger imbalanced dataset (containing more negative samples) give similar overall performance (measured by AUC and accuracy) to that on the balanced dataset (with equal number of negative and positive samples). However, the results on the imbalanced dataset have higher specificity than sensitivity whereas the results on the balanced dataset have higher sensitivity.

We have also tried some simple ensemble strategies to combine a subset of the result of these seven classifiers, but it seems not to give a significant improvement on the performance. We think that some feature selection methods can be applied to improve the performance. Feature extraction techniques might also provide more useful information. Technologies for dealing with imbalanced data like oversampling can be applied to the whole data set. Based on large set of unlabeled data set in the original database, semi-supervised learning can also be considered a potential method to make improvements in the future work. We believe the machine learning methods can be applied to other medical image analysis problems. We hope to try our methods on other datasets to hunt for more general outcomes.

Bibliography

[1] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Byers, A.H.: Big data: The next frontier for innovation, competition, and productivity. (2011)

[2] WHO: Human papillomavirus and related diseases in the world. (2015)

[3] Kim, E., Huang, X.: A data driven approach to cervigram image analysis and classification. In: Color Medical Image analysis, Lecture Notes in Computational Vision and Biomechanics. Volume 6. (2013) 1–13

[4] Sankaranarayanan, R., Gaffikin, L., Jacob, M., et al.: A critical assessment of screening methods for cervical neoplasia. International Journal of Gynecology and Obstetrics **89** (2005) 4–12

[5] Biscotti, C.V., Dawson, A.E., et al.: Assisted primary screening using the automated thinprep imaging system. In: AJCP. Volume 123(2).

(2005) 281–287

[6] Wilbur, D.C., Black-Schaffer, W.S., Luff, R.D., Abraham, K.P., Kemper, C., Molina, J.T., Tench, W.D.: The becton dickinson focalpoint gs imaging system. American Journal of Clinical Pathology **132**(5) (2009) 767–775

[7] Zhang, J., Liu, Y.: Cervical cancer detection using svm based feature screening. In: MICCAI. Volume 3217. (2004) 873–880

[8] Mitchell, T.M., et al.: Machine learning. wcb (1997)

[9] Ian Goodfellow, Y.B., Courville, A.: Deep learning. Book in preparation for MIT Press (2016)

[10] Wolpert, D.H.: The lack of a priori distinctions between learning algorithms. Neural computation **8**(7) (1996) 1341–1390

[11] Domingos, P.: A few useful things to know about machine learning. Communications of the ACM **55**(10) (2012) 78–87

[12] Hastie, T., et al.: The elements of statistical learning. Volume 2. Springer (2009)

[13] Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In: ICML, ACM (2006) 161–168

[14] Morra, J.H., Tu, Z., Apostolova, L.G., et al.: Comparison of adaboost and support vector machines for detecting alzheimer's disease through automated hippocampal segmentation. In: Medical Imaging. Volume 29. (2010) 30–43

[15] Osareh, A., Mirmehdi, M., et al.: Comparative exudate classification using support vector machines and neural networks. In: MICCAI. Springer (2002) 413–420

[16] Herrero, R., Schiffman, M., Bratti, C., et al.: Design and methods of a population-based natural history study of cervical neoplasia in a rural province of costa rica: the guanacaste project. Rev Panam Salud Publica **1** (1997) 362–375

[17] Jeronimo, J., Long, L.R., Neve, L., Michael, B., Antani, S., Schiffman, M.: Digital tools for collecting data from cervigrams for research and training in colposcopy. Journal of Lower Genital Tract Disease **10**(1) (2006) 16–25

[18] Song, D., Kim, E., Huang, X., et al: Multi-modal entity coreference for cervical dysplasia diagnosis. In: Medical Imaging, IEEE (2014)

[19] Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern recognition **29**(1) (1996) 51–59

[20] Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. Pattern Analysis and Machine Intelligence, IEEE Transactions on **24**(7) (2002) 971–987

[21] Pedregosa, F., Varoquaux, G., Gramfort, A., et al.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12** (2011) 2825–2830

[22] Goodfellow, I.J., Warde-Farley, D., Lamblin, P., et al.: Pylearn2: a machine learning research library. arXiv:1308.4214 (2013)

[23] Chang, C., Lin, C.: LIBSVM: a library for support vector machines (2001)

[24] Appel, R., Fuchs, T., Dollr, P., Perona, P.: Quickly boosting decision trees pruning underachieving features early. In: ICML. (2013)

[25] Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? The Journal of Machine Learning Research **15**(1) (2014) 3133–3181

Biography

Cheng Xin was born on September 5th 1990, in Shanghai, China. He has received his Bachelor of Engineering, in Software Engineering, from Tongji University. Upon completion of his bachelor studies, he joined Lehigh University to pursue his master of science degree in Computer Science. His research interests include machine learning, deep learning, data mining, computer vision and applied topology.