CANCER DETECTION FOR LOW GRADE SQUAMOUS INTRAEPITHELIAL LESION

BY

KRITI CHAKDAR

Submitted to the graduate degree program in Electrical Engineering & Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science

Thesis Committee

____

Dr. Brian Potetz: Chairperson

Committee members

____

Dr. Arvin Agah

____

Dr. Luke Huan

The Thesis Committee for Kriti Chakdar certifies that this is the approved version of the following thesis:

Cancer Detection for low grade squamous intraepithelial lesion

____

Chairperson Dr. Brian Potetz

Date approved: 30th August,2012

Abstract

The National Cancer Institute estimates in 2012, about 577,190 Americans are expected to die of cancer, more than 1,500 people a day. Cancer is the second most common cause of death in the US, accounting for nearly 1 of every 4 deaths. Cancer diagnosis has a very important role in the early detection and treatment of cancer. Automating the cancer diagnosis process can play a very significant role in reducing the number of falsely identified or unidentified cases. The aim of this thesis is to demonstrate different machine learning approaches for cancer detection.

Dr. Tawfik, pathologist from University of Kansas medical Center (KUMC) is an inventor of a novel pathology tissue slicer. The data used in this study comes from this slicer, which successfully allows semi-automated cancer diagnosis and it has the potential to improve patient care.

In this study the slides are processed and visual features are computed and the dataset is made from scratch. After features extraction, different machine learning approaches are applied on the dataset which has shown its capability of extracting high-level representations from high-dimensional data. Support Vector Machine and Deep Belief Networks (DBN) are the concentration in this study. In the first section, Support vector machine is applied on the dataset. Next Deep Belief Network which is capable of extracting features in an unsupervised manner is implemented and with back-propagation the network is fine tuned. The results show that DBN can be effective when applied to cytological cancer diagnosis by increasing the accuracy in cancer detection. In the last section a subset of DBN features are selected and then appended with raw features and Support Vector Machine is trained and tested with that. It shows improvement over the first section results. In the end the study infers that Deep Belief Network can be successfully used over other leading classification methods for cancer detection.

Acknowledgement

I would like to thank all my advisors who have supported and directed me through this research. Dr. Brian Potetz was my faculty advisor and he has guided me at every stage of this research being patient every time. It would have not been possible to accomplish this thesis without his throughout help. I’m also thankful to Dr. Arvin Agah and Dr. Luke Huan who were on my thesis committee and professors of my graduate course works. All of them have helped me improving every day though my Research work and Graduate course work in University of Kansas.

Table of Contents

Chapter 1: Introduction 1
1.1 Motivation 1
1.2 Focus Area 4
1.3 Thesis Organization 5
Chapter 2: Background and Related Work 7
2.1 Statistical Methods 7
2.2 Machine Learning Methods 8
2.3 Deep Belief Network: 8
2.4 Energy-Based Models (EBM) 11
2.5 Restricted Boltzmann Machines (RBM) 14
2.6 Contrastive Divergence (CD-k) 18
Chapter 3: Processing and Data Analysis 19
3.1 Preprocessing of image slides 19
3.2 The Dataset 27
3.3 Outline of Processing Dataset 30
3.4 Classification Methods Used 31
Chapter 4: Applying SVM 33
4.1 Support Vector machine 33
4.2 Partitions 33
4.3 Results 34
4.5 Learning and Classification 38
4.6 Details result of SVM on one test set 38
4.7 Details result of SVM on second test set 39
Chapter 5: Applying Deep Belief Nets 41
5.1 Introduction and Motivation 41
5.2 Dataset Modified 41
5.3 Feature Extraction Using DBN: 42
5.4 Partitions 45
5.5 Normalization 45

5.6 Training the Deep Belief Network 46
5.7 Back propagation 50
5.8 Results of DBN with back propagation on set 1 51
5.9 Results of DBN with back propagation on set 2 53
Chapter 6: Combining SVM with DBN features 55
6.1 Motivation: 55
6.2 Selecting DBN features in the 55
6.3 Results with Appended 5 DBN features on set 1 56
6.4 Results with Appended 5 DBN features on set 2 58
6.3 Results with Appended 5 DBN features 60
Chapter 7: Conclusion and Discussions 63
7.1 Cell Block Preparation 63
7.2 Summary 63
7.3 Future Direction 63
7.4 Recent improvements in Deep Learning 64
References 66

Chapter 1: Introduction

1.1 Motivation

The precision in identifying cancerous structure from the benign structure has the potential to immediately improve health outcomes of one of our most pressing diseases. It will also contribute to one of artificial intelligence (AI) biggest remaining frontiers - the automated extraction of complex discriminative features.

The process of cancer diagnosis is semi-automatic and is prone to human error. It is also time consuming. The Papanicolaou (Pap) Smear-most common test to identify pre-cancerous or cancerous processes in cervix of human body generates large sized digital images. Pathologists have a challenging task of manually monitoring the digital images. It has been found that the test may fail because of “inadequate samples, insufficient time devoted to screening, or human fatigue” during examination of the images or biopsy report (Koss, 1989).

In the 1930s, in the United States the most common cause of cancer deaths in women was cervical cancer. But today it is not even in top ten causes of women's deaths in the United States. The reason in the significant decline in the incidence and mortality of cervical cancer is the Papanicolaou (Pap) test, developed by Gerge Nicholas Papanicolaou. According to the estimation over 3.5 million US women each year will have an abnormal Pap smear. Approximately two million US women are diagnosed annually as having “Atypical Cells of Undetermined Significance” (ASCUS) in their cervical cytologic specimens. Current techniques for obtaining Pap samples include the conventional smear and the more recently preferred liquid-based preparations. The conventional smears are often obtained using a combination of a plastic spatula, brush or a broom-like brush. They require immediate coating fixation with ethanol and polyethylene followed by submersion in 95% ethanol to prevent air-drying artifact that interferes with appropriate evaluation of samples.

Poor sensitivity of the conventional smears became apparent in the 1990's. Several studies have shown that mean sensitivity of the conventional smear ranged from 37 at worst to 73% at best. When sources of errors with the conventional smear were analyzed, it was discovered that sampling and/or preparation errors were responsible for about two thirds of false negative cases. In those situations cells were either not collected on the sampling devices or collected cells were not transferred to slides. Screening and/or interpretive errors were the cause for the remaining 1/3 of false negative cases. Abnormal cells were either missed by cytotechnologist/cytopathologist or were incorrectly classified because of poor preservation. Because of problems with conventional techniques liquid-based preparations were developed. The liquid based processing mitigated sampling errors and improved cell preservation. There was better randomized representation and even distribution of cells with minimization of obscuring material such as inflammation, blood and debris. Liquid based preparations were significantly more effective in detecting low grade and high grade squamous intraepithelial lesions (LG SIL and HGSIL) and were as good as conventional smear preparations in detecting endocervical lesions. Several liquid based technologies became available and are FDA approved. Currently available techniques include the ThinPrep Pap Test (HologicInc, Bedford, MA) and SurePath Pap Test (Becton, Dickenson and Co, Franklin Lakes, NJ). Each has its own advantages and disadvantages.

Talking about figures in a large hospital, a pathologist typically handles 100 grading cases per day, each consisting of about 2000 image frames. As the numbers say it, it’s a very tedious and time-consuming task. A computer system that performs automatic grading can assist the pathologists by providing second opinions, reducing their workload, and alerting them to cases that require closer attention, allowing them to focus on diagnosis and prognosis

Some more interesting facts in regarding Pap smear test are as follows:

- Over 500,000 women will develop cervical cancer worldwide with an annual death rate of close to 300,000/year.
- 55 million Pap tests are performed on a yearly basis in United Stated (source: NCI website)
- 3 million abnormal Pap tests are reported on a yearly basis in United States

Unfortunately, pap smear preparations produce thick slides with many layers of cells. This complicates diagnosis, and because there are many focal planes, digitization of slides is impractical.

Dr. Tawfik, pathologistfrom KUMC (University of Kansas Medical Center) is an inventor of a novel pathology tissue slicer. Briefly, the technology is either a manual or an electrical/ultrasonic device used to cut surgical tissue specimens into standardized, even slices ready for preparation of tissue blocks. The device is composed of a manual or an electric/ultrasonic cutting knife with multiple replaceable disposable blades, a specialized cutting board and a multi pin specimen holder device.

This device produced cell blocks from Pap smears, which allow samples to be condensed and sliced thinly. Thin slices simplify diagnosis and permit accurate digitization. Digital slide images make it possible to consult with expert pathologists remotely and can be better used in education, either in published material or online. Archival of digital images is substantially easier than glass slides. Finally, digital slide images permit computer-assisted diagnosis. Automated computer vision techniques are able to prescreen digital images to flag suspicious regions and to discard areas of images that contain clearly healthy tissue. Automated prescreening alleviates the fatigue faced by professional pathologists, who must often spend hours peering through microscopes. In this study the digital slides are read and different classifiers are used to gain one optimal performance on cancer dataset.

1.2 Focus Area

Within a pap smear preparation, architectural features are not available. Also, because cells in the CB are scattered spatially, the variance of morphological features is weaker than for histological preparations where the variance of cell size and shape can be measured within a small region of interest. For these preparations, cytological features are much more important. Thus, it is essential to develop effective textural features that can fully capture the properties of cells that correlate most strongly with cancer diagnosis. While modern textural features have proven highly useful, it remains that these approaches are limited to very basic texture features that are chosen by hand. A more effective strategy would be to apply machine learning techniques to search for highly discriminative visual features that are maximally predictive of cancer and pre-cancer grading. Until recently, such an approach has been difficult due to the vast size of the set of possible visual features. However, modern advances in computer vision and machine learning have produced new technologies that allow complex, non-linear image features to be learned from sample images using greedy, layer-by-layer training strategies.

A group of diagnosis studies have applied machine learning algorithms to learn (from data) how to distinguish the different classes from each other. Neural networks, K-nearest neighborhood, Logistic regression, Fuzzy systems, Linear discriminant analysis, Decision trees are among those. Keeping aside the machine learning methods which are implemented before, In this study Deep Learning Method is the main focus.

Deep Learning is a new area of Machine Learning research and it depends on an efficient, layer by layer procedure for learning. Using the deep learning method an auto encoder is being trained with a large dataset consisting of 576 features, for dimensionality reduction or feature extraction and then depending on the extracted features, the whole data set is classified into two groups- malignant and benign.

This study uses different classification methods and then their performances are compared to diagnosis the cells into malignant and healthy classes. The comparative study of various classifiers performance can possibly help in reducing errors in the examination of digital slide generated after the Pap smear test.

1.3 Thesis Organization

The thesis is organized into the following chapters:

- **Chapter 1: Introduction**

  In this section the limitations of cancer diagnosis procedure is discussed and how it can be improved with automated Cancer Diagnosis are discussed. It is also discussed how it can be achieved by using different classification methods.

- **Chapter 2: Background and Related Work**

  An overview is given of related studies for automating the cancer diagnosis process. Deep Belief Network, Restricted Boltzmann machine are explained in details.

- **Chapter 3: Processing and Data Analysis**

  A description of the dataset used in this study is given and how the data is analyzed to get better classification is also explained.

- **Chapter 4: Applying SVM**

  The results with Support Vector Machine classifier are discussed.

- **Chapter 5: Applying Deep Belief Network**

  The whole procedure of applying Deep Belief Network is discussed and the results are also tabled.

- **Chapter 6: Combining SVM with DBN features**

  In this section, how a subset of features, extracted from Deep Belief Network are selected and processed are discussed.

- **Chapter 7: Conclusion and Discussions**

  Pros and cons are discussed about the method followed in this study. And the possible ways improvements are discussed.

Chapter 2: Background and Related Work

Automated cancer diagnosis, both at the cellular and tissue-levels, is based on (i) extracting information from the histo-pathological images of stained biopsies and (ii) examining this information by using either statistical analysis or machine learning algorithms. There are three main computational processes in the automated cancer diagnosis: *preprocessing*, *feature extraction*, and *diagnosis*. The preprocessing used to enhance the image quality. It tries to reduce or fully eliminate the background noise from the image. This enhancement helps in identifying the focal areas lying on the image. The preprocessing helps and also decides the subsequent processes of feature extraction and diagnosis. The second process of feature extraction is aimed at the cellular or tissue-level.(Doyle, S., Agner, S., &Madabhushi, A. 2008).

2.1 Statistical Methods

The automated diagnosis uses the feature set to distinguish the malignant cell structures from other cells. In short, the process categorizes cells into healthy, benign and cancerous. It also assigns the grading to classify the level of malignancy of the tissue or cell. A section of research is conducted on statistical test on the features (ADD). This method finds existence of substantial difference in the value of at least one feature of the interest in the different classes. It calls for extra precaution in case of ‘Histo-pathological’ images. The precaution is needed because the data set comprises of various tissue images procured from one patient. These data sets are not independent and lead to wrong results.

2.2 Machine Learning Methods

The major contribution in the area of machine learning methods using neural networks were suggested by Esgiar et al. (2002) and other for K-nearest neighborhood. Researches by Demir et al. (2004), Gunduz et al. (2004), Zhou et al. (2002) and others were based on neural networks. The fuzzy systems were suggested by Blekas et al. (1998), logistic regression were studies extensively by Woolberg et al. (1995) and others. Decision trees were recommended by Wiltgen et al. (2003) and linear discriminant analysis was devised by Smolle (2000) and others. These studies utilize data from machine learning algorithms to distinguish categories or classes. The above studies utilized various techniques for evaluation. Smolle, Anderson et al. used no separate evaluation set. Schnorrenberg et al. (1996), Spyridonos et al. (2001) and others found leave-one-out approach effective. A considerable number of studies carried by Blekas et al. (1998), Wiltgen et al. (2003) and others found separate training and tests sets utilized separate training and test sets. Zhou et al. (2002) and others implemented K-fold cross validation.

The classification system that has been divided into two stages posed a problem of using a limited set of data in both the stages. In the first stage classifier is trained to learn the parameters of the system. In the second stage system is tested to evaluate to measure classifier's success rate. The use of same set of data in both the stages led to the problem of memorization prone to errors. To solve this crisis it has been recommended to use separate data sets for two stages. For this purpose the data set is divided into two groups.

2.3 Deep Belief Network:

The Deep Learning is a very new area of Machine Learning research that has been recently being applied to a range of problems in the sector of classification, collaborative filtering,semantic hashing, dimensionality reduction, and many more such areas Geoffrey

Hinton, inventor of the Deep Belief defined it as, “Deep Belief Networks are probabilistic generative models that are composed of multiple layers of stochastic latent variables. The top two layers have undirected, symmetric connections between them and form an associative memory. The lower layers receive top-down directed connections from the layer above.” The Deep belief nets consist of two crucial computational properties. In the first, it has a very efficient procedure that enables learning the top-down, generative weights which specifies how the variables on the one layer determine the probabilities of variables on the layer lying below. This procedure results in learning one layer of latent variables at a time. In the second, that happens after learning of the multiple layers, latent variables values in every layer can be understoodby a single, bottom-up pass which starts with an observed data vector in the bottom layer and uses the generative weights in the reverse direction.

The deep belief net is witnessing utilization in number of areas. Hinton, Osinddero&Teh (2006) used it for recognizing and generating images. The concept was also used for getting data from motion capture (Taylor, Hinton &Roweis, 2007).

The potential usefulness of Deep architectures that been has been shown in some domains like representing logic circuits, deep architectures can represent functions in a much more efficient manner than that of shallow ones. Invariance is another important factor for considering hierarchical structures. With the help of this algorithm, it makes possible to precisely train a deep network through learning each layer greedily in a manner that is unsupervised like a Restricted Boltzmann Machine (RBM).

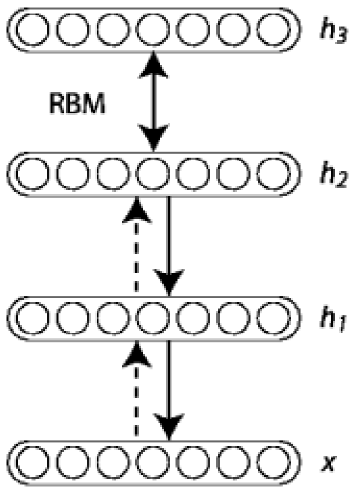
The mean-field hidden unit activations in one RBM are used as the features for the next RBM, and so forth. In a bid to train the RBMs with a Maximum Likelihood, a blocked Gibbs sampler is run for many iterations to get unbiased samples from the model distribution to come up with an effective gradient approximation. It has the capacity to train an RBM thoroughly by always beginning from the data, and then truncating the Gibbs chain.

Hinton performed and concluded that RBMs could be stacked and also trained in a greedy manner to form the Deep Belief Networks (DBN). DBNs are graphical models which learn to extract a deep hierarchical representation of the training data. (Theano Development Team, Deep Learning v0.1 documentation)They model the joint distribution between observed vector x and the ℓ hidden layers h^k as follows:

$$P(x, h^1, \ldots, h^\ell) = \left(\prod_{k=0}^{\ell-2} P(h^k | h^{k+1}) \right) P(h^{\ell-1}, h^\ell)$$

Equation1

where $x = h^0$, $P(h^{k-1}|h^k)$ is a conditional distribution for the visible units conditioned on the hidden units of the RBM at level k, and $P(h^{\ell-1}, h^\ell)$ is the visible-hidden joint distribution in the top-level RBM. This is illustrated in the figure below. (Theano Development Team, Deep Learning v0.1 documentation)



The principle of greedy layer-wise unsupervised training can be applied to DBNs with RBMs as the building blocks for each layer. The five-step process represented:

- Stage 1. The first layer is trained as an RBM which models the raw input $x = h^{(0)}$ as its visible layer.

- Stage 2. The first layer is used to get a representation of the input that will be utilized as data for the second layer. There are two common solutions. This representation is selected as being the mean activations $p(h^{(1)} = 1|h^{(0)})$ or samples of $p(h^{(1)}|h^{(0)})$.

- Stage 3. The second layer is trained as an RBM,it takes the transformed data (samples or mean activations) as training examples (for the visible layer of that RBM).

- Stage 4. Iterate (2 and 3) for the desired number of layers, each time propagating upward either samples or mean values.

- Stage 5.All parameters are fine-tuned for this deep architecture with respect to a proxy for the DBN log- likelihood, or with respect to a supervised training criterion.

This fine-tuning is done through supervised gradient descent. To be specific, use of logistic regression classifier is done to classify the input x that is based on the output of the last hidden layer $h^{(l)}$ of the DBN. The fine-tuning is then performed through supervised gradient descent of the negative log-likelihood cost function. Since, the supervised gradient is only non-null for the weights and hidden layer biases of each layer

2.4 Energy-Based Models (EBM)

(Theano Development Team, Deep Learning v0.1 documentation) The models based on energy associate with a scalar energy to every configuration of the variables of interest. Learning modifies this energy function in a way that its shape gives desirable properties. One such example, in which plausible or desirable configurations is expected to possess low energy.

Probabilistic models based on energy define a probability distribution through an energy function, as mentioned below:

$$p(x) = \frac{e^{-E(x)}}{Z}. \quad (1)$$

Equation1

The normalizing factor Z is called the **partition function** by analogy with physical systems.

$$Z = \sum_x e^{-E(x)}$$

A model based on energy could be studied by performing (stochastic) gradient descent on the empirical negative log-likelihood of the training data. And for the logistic regression it is advisable to first define the log-likelihood. And then the loss function as being the negative log-likelihood.

$$\mathcal{L}(\theta, \mathcal{D}) = \frac{1}{N} \sum_{x^{(i)} \in \mathcal{D}} \log\ p(x^{(i)})$$

$$\ell(\theta, \mathcal{D}) = -\mathcal{L}(\theta, \mathcal{D})$$

using the stochastic gradient $-\frac{\partial \log p(x^{(i)})}{\partial \theta}$, where θ are the parameters of the model.

EBMs with Hidden Units

(Theano Development Team, Deep Learning v0.1 documentation) In many of the cases, example x is not observed fully. It may be the case that it requires introduction of some non-observed variables to enhance the expressive power of the model. In such cases we try to consider an observed part (denoted by x here) with a **hidden** part h. This way we can make the equation:

$$P(x) = \sum_h P(x,h) = \sum_h \frac{e^{-E(x,h)}}{Z}.$$

Equation2

To map this formulation in such cases that is similar to Eq. (1), we then introduce the notation concluded from physics of **free energy** and define it, this way:

$$\mathcal{F}(x) = -\log \sum_h e^{-E(x,h)}$$

Equation 3

Thus, it enables the equation,

$$P(x) = \frac{e^{-\mathcal{F}(x)}}{Z} \text{ with } Z = \sum_x e^{-\mathcal{F}(x)}.$$

The data of negative log-likelihood gradient takes a curious form

$$-\frac{\partial \log p(x)}{\partial \theta} = \frac{\partial \mathcal{F}(x)}{\partial \theta} - \sum_{\tilde{x}} p(\tilde{x}) \frac{\partial \mathcal{F}(\tilde{x})}{\partial \theta}.$$

Equation 4

It is worth noticing that the above mentioned gradient has two terms. These two terms are **positive** and **negative phase**. Interestingly, positive and negative terms do not pertain to the signs of each terms mentioned in the equation. On the contrary, it reflects their effect on the probability density that is defined by the model. The role of first term is to increase the probability of training data. It happens by reduction of corresponding free energy. The role of the second term is to decreases the probability of samples that is generated using the model.

It's highly and too difficult to analytically determine this particular gradient. It's difficult because it requires the computation of $E_P[\frac{\partial \mathcal{F}(x)}{\partial \theta}]$.

(Theano Development Team, Deep Learning v0.1 documentation) The first course of action required make this tough computation easy is to make an estimationof the expectation by using a fixed number of model samples. The samples that are used for the estimation of the negative phase gradient are called **negative particles** that are denoted by $\mathcal{N}$. Then the gradient becomes:

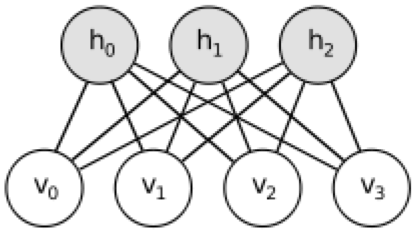
$$-\frac{\partial \log p(x)}{\partial \theta} \approx \frac{\partial \mathcal{F}(x)}{\partial \theta} - \frac{1}{|\mathcal{N}|} \sum_{\tilde{x} \in \mathcal{N}} \frac{\partial \mathcal{F}(\tilde{x})}{\partial \theta}.$$

Equation 5

here the need is to sample the like elements $\tilde{x}$ of $\mathcal{N}$ according to P. The above mentioned formula gives a practical and stochastic algorithm for EBM learning. One of the major weaknesses is the missing ingredient regarding extraction of these negative particles $\mathcal{N}$. The literature mentioning sampling methods suggests that Markov Chain Monte Carlo methods are nearly perfect for models like the Restricted Boltzmann Machines (RBM) that is a specific type of EBM.

2.5 Restricted Boltzmann Machines (RBM)

(Theano Development Team, Deep Learning v0.1 documentation) Boltzmann Machines (BMs) are considered to be a particular form of log-linear Markov Random Field (MRF), in which the energy function is linear in its free parameters. In a bid to make them powerful so that they can represent complicated distributions to switch from the limited parametric to a non-parametric setting, it is required that we should not observe some of the variables and term them as hidden. The more hidden variables or units result in increasing the capacity of the Boltzmann Machine (BM). Restricted Boltzmann Machines further restricts BMs those that have either visible-visible or hidden-hidden connections. A graphical representation of an RBM is depicted as follows:



The energy function $E(v,h)$ of an RBM is defined as:

$$E(v,h) = -b'v - c'h - h'Wv$$

Equation 6

Here W is the weights that is connecting hidden and visible units and b, c are the offsets of the visible and hidden layers respectively.

This translates directly to the following free energy formula:

$$\mathcal{F}(v) = -b'v - \sum_i \log \sum_{h_i} e^{h_i (c_i + W_i v)}.$$

Because of the specific structure of RBMs, visible and hidden units are conditionally independent given one-another. Using this property, we can write:

$$p(h|v) = \prod_i p(h_i|v)$$
$$p(v|h) = \prod_j p(v_j|h).$$

RBMs with binary units

(Theano Development Team, Deep Learning v0.1 documentation) In the commonly studied case of using binary units (where v_j and $h_i \in \{0,1\}$), we calculate from Eq. (6) and (2), a probabilistic version of the usual neuron activation function:

$$P(h_i = 1|v) = sigm(c_i + W_i v)$$

Equation 7

$$P(v_j = 1|h) = sigm(b_j + W'_j h)$$

Equation 8

The free energy of an RBM with binary units further simplifies to:

$$\mathcal{F}(v) = -b'v - \sum_i \log(1 + e^{(c_i + W_i v)}).$$

Equation 9

Update Equations with Binary Units

Combining Eqs. (5) with (9) we obtain the following log-likelihood gradients for an RBM with binary units:

$$-\frac{\partial \log p(v)}{\partial W_{ij}} = E_v[p(h_i|v) \cdot v_j] - v_j^{(i)} \cdot sigm(W_i \cdot v^{(i)} + c_i)$$
$$-\frac{\partial \log p(v)}{\partial c_i} = E_v[p(h_i|v)] - sigm(W_i \cdot v^{(i)})$$
$$-\frac{\partial \log p(v)}{\partial b_j} = E_v[p(v_j|h)] - v_j^{(i)}$$

(10)

Sampling in an RBM

(Theano Development Team, Deep Learning v0.1 documentation) Samples of $p(x)$ is calculated by running a Markov chain to convergence and using Gibbs sampling for the transition operator.

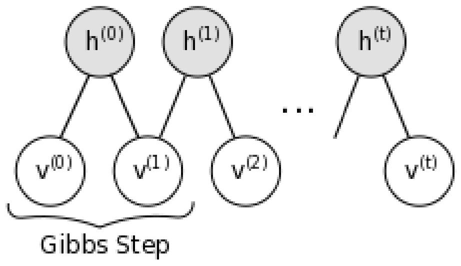
Gibbs sampling of the joint of N random variables $S = (S_1, ..., S_N)$ is done through a sequence of N sampling sub-steps of the form $S_i \sim p(S_i | S_{-i})$ where S_{-i} contains the $N-1$ other random variables in S excluding S_i.

For RBMs, S consists of the set of visible and hidden units. However, since they are conditionally independent, one can perform block Gibbs sampling. In this setting, visible units are sampled simultaneously given fixed values of the hidden units. Similarly, hidden units are sampled simultaneously given the visibles. A step in the Markov chain is thus taken as follows:

$$h^{(n+1)} \sim sigm(W'v^{(n)} + c)$$
$$v^{(n+1)} \sim sigm(Wh^{(n+1)} + b),$$

where $h^{(n)}$ refers to the set of all hidden units at the n-th step of the Markov chain. What it means is that, for example, $h_i^{(n+1)}$ is randomly chosen to be 1 (versus 0) with probability $sigm(W_i'v^{(n)} + c_i)$, and similarly, $v_j^{(n+1)}$ is randomly chosen to be 1 (versus 0) with probability $sigm(W_{.j}h^{(n+1)} + b_j)$.

This can be illustrated graphically:



As $t \to \infty$, samples $(v^{(t)}, h^{(t)})$ are guaranteed to be accurate samples of $p(v, h)$.

In theory, each parameter update in the learning process would require running one such chain to convergence. It is needless to say that doing so would be prohibitively expensive. As such, several algorithms have been devised for RBMs, in order to efficiently sample from $p(v, h)$ during the learning process.

2.6 Contrastive Divergence (CD-k)

(Theano Development Team, Deep Learning v0.1 documentation) Contrastive Divergence uses two tricks to speed up the sampling process:

- since we eventually want $p(v) \approx p_{train}(v)$ (the true, underlying distribution of the data), we initialize the Markov chain with a training example (i.e., from a distribution that is expected to be close to p, so that the chain will be already close to having converged to its final distribution p).
- CD does not wait for the chain to converge. Samples are obtained after only k-steps of Gibbs sampling. In pratice, $k = 1$ has been shown to work surprisingly well.

Chapter 3: Processing and Data Analysis

3.1 Preprocessing of image slides

The original digital image slides provided by KUMC are in .svs format and those are pretty huge having size over 40,000 × 40,000 pixels each.

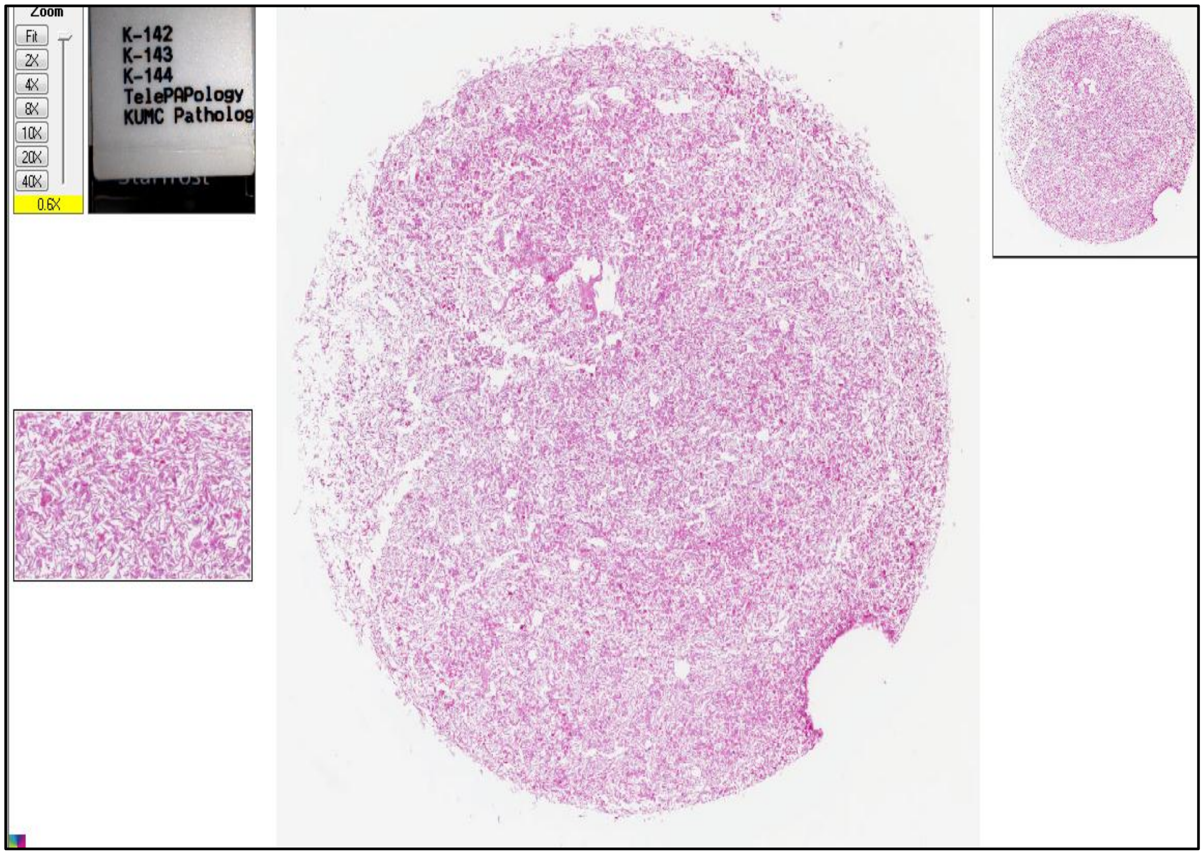


Figure 1: Full Size Digital Image

So, it was very important to preprocess the slides so that it can be used conveniently for the purpose of cancer detection. Below is the description of the method by which the large images were preprocessed.

The feature extraction phase has four main components. First, large images (often over 40,000 pixels wide) are broken into smaller tiles. Next, nuclei are located. Third, nuclei features are quantified. Finally, results from each tile are joined. Figure 4 shows the dependency chart for the feature extraction process.

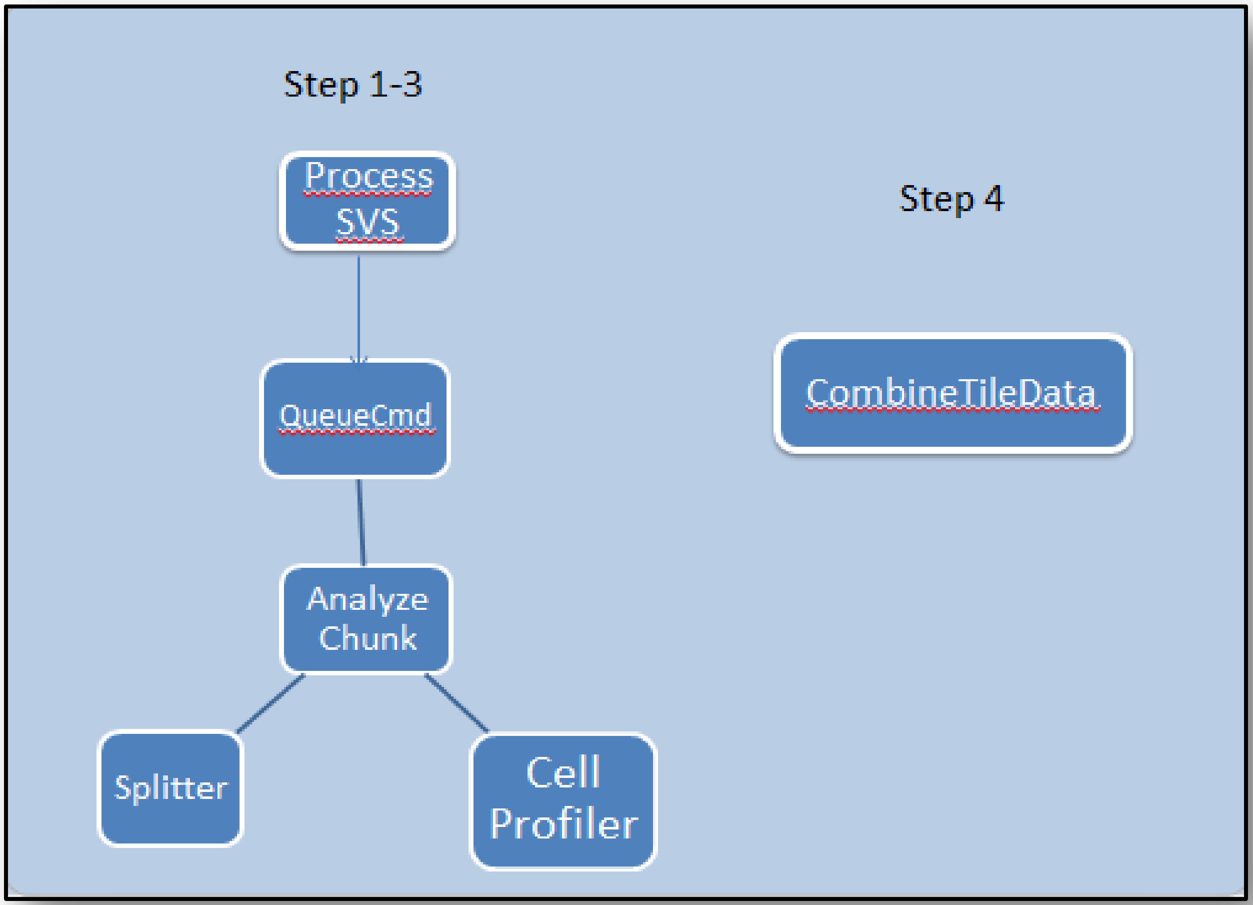


Figure 2: Feature extraction is performed by two main programs: ProcessSVS and CombineTileData. The figure shows the dependencies of each on other additional programs

ProcessSVS:

The program ProcessSVS executes steps 1 through 3. ProcessSVS begins by choosing the location of each tile. It then forks the analysis of each image tile into a separate process using QueueCmd. This allows tiles to be processed in parallel. From the given large .svs image, ProcessSVS chooses the locations of the boundaries of the chunks of the image that will be analyzed separately. Each chunk will be of exactly the same size (possibly slightly smaller than max_tile_width&max_tile_height). ProcessSVS uses the QueueCmd utility to call AnalyzeChunk for each chunk. The AnalyzeChunk is the program that actually saves out a piece of the image - ProcessSVS only controls the behavior of AnalyzeChunk. Ultimately, for each chunk two directories within <output_directory> will be created : ####_temp and ####_out, where #### is the ID number of the chunk (starts with 0001). ####_temp will contain the exported chunk images and the CellProfiler output. ####_out will contain the comma-delimited text file Features_####.csv,that contains the properties of every cell in the chunk.Because ProcessSVS has no way to tell when all the chunks are finished beinganalyzed, the program to combine the output of all the chunks into one file mustbe called separately.

AnalyzeChunk

AnalyzeChunk performs the bulk of the work: locating each cell and quantifying its features. AnalyzeChunk begins by applying a quadratic filter to the image tile designed to emphasize cell nuclei. This filter was trained using Fisher linear discriminants applied to hand-labeled images. The filtered images are sent to Cell Profiler (Carpenter,A., Jones, T., Lamprecht, M., Clarke, C., Kang, I., Friman, O.,Guertin, D., Chang, J., Lindquist, R., Moffat, J., Golland, P., & Sabatini. D. (2006) ,which locates the nuclei outlines using established techniques designed to distinguish two nuclei even if they overlap slightly. The segmentation of cell nuclei is the only responsibility of cell profiler in this software. AnalyzeChunk next collects quantitative features from each cell. AnalyzeChunk calls Splitter to find the specified chunk within the .svs image andexport it as a color .ppg image to temporary directory. The .ppg image is deletedafter it is loaded into matlab.If the desired chunk is the full size of the image, and the image is a tif,AnalyzeChunk operates in "tif mode". In this mode, AnalyzeChunk skips the step ofcalling Splitter, and instead reads in the image directly into matlab. Settingw<=0 is one way to be sure AnalyzeChunk that this happens. This is intended foruse with the small tif images used during training.AnalyzeChunk then downsamples the image, performs some image processing to emphasize the nuclei within the image (a quadratic filter learned via FisherLinear Discriminants), and exports this as a grayscale .tif image (also to thetemporary directory).Next, AnalyzeChunk calls CellProfiler on the tif image.

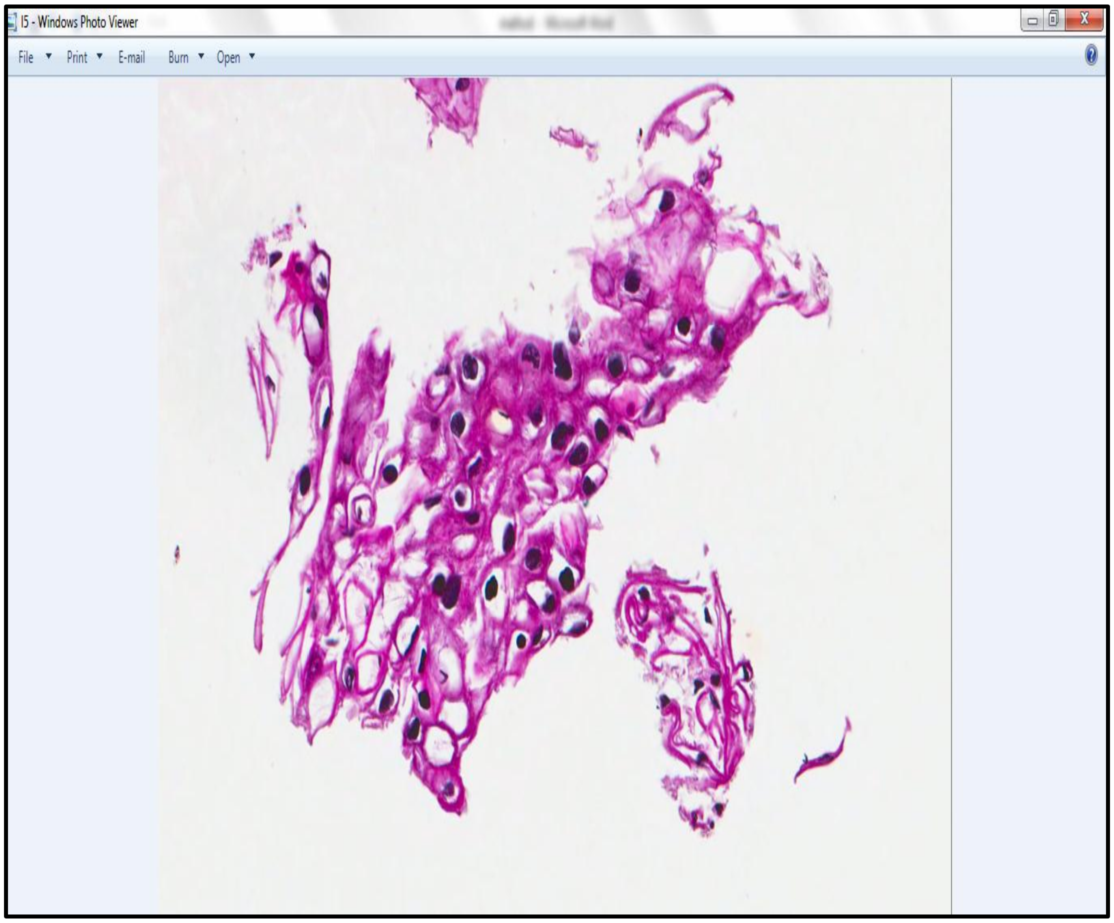


Figure 3: Full size images broken into small tiles

Inside the CellProfiler, pipelines are formed and nuclei in each image were located and then various nuclei features are quantified. Finally, results from each tile were joined. Quadratic filter was applied to the image tile designed to emphasize cell nuclei. This filter was trained using Fisher linear discriminants applied to hand-labeled images.

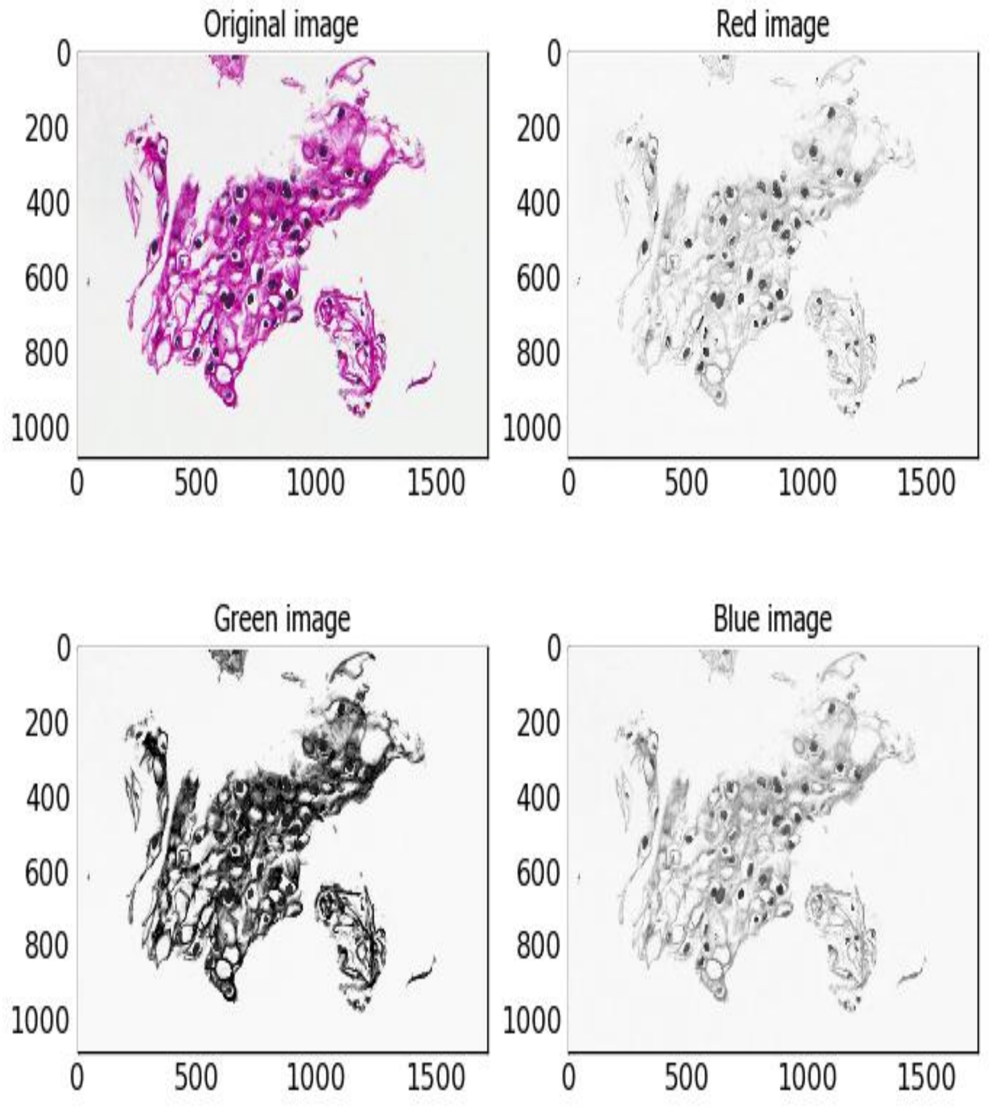


Figure 4: Analyzing Tiles

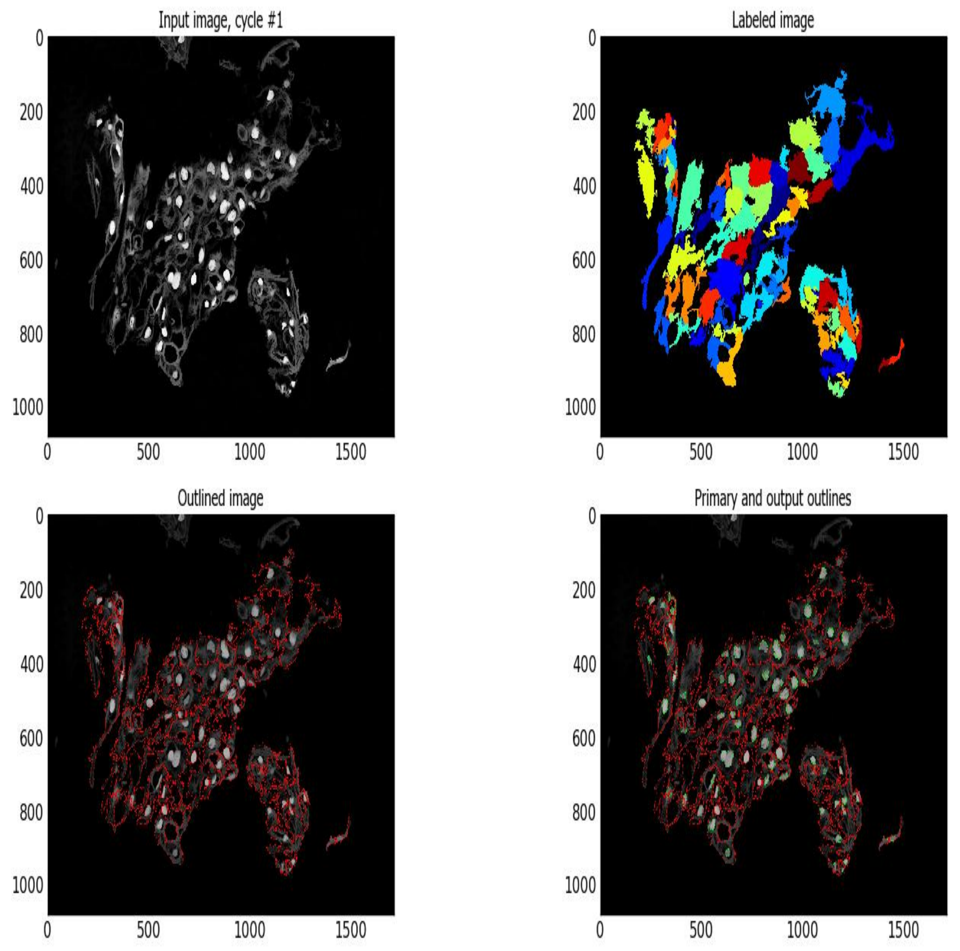


Figure 5: Extracting Nuclei Features

Finally, AnalyzeChunk reads in the output of CellProfiler. Using the celllocations and the morphological properties of the nuclei & cells, it creates alist of quantitative features for each cell. If AnalyzeChunk is not in tif mode, execution ends here. Otherwise, AnalyzeChunk continues by analyzing the spatial properties of the features (calls FinalSpatialProcessing). Next, it searches for an .xml file in the same directories and if one is found, it discards all cells that lie outside of the annotations.

The visual features extracted by AnalyzeChunk can be grouped into three broad categories. Morphological features describe physical properties of cells or the statistics of those properties over groups of cells. These features include the degree of variance of nuclei size and aspects of nuclei shape. Next, textural features describe low-level visual properties, which, in the case of histologic diagnosis applications, are designed to capture visual qualities including increases in nucleus size and the prominence of nucleoli. These features are measured by the histograms and co-occurance matrices of pixel colors. Such features capture the first and second order statistics of cellular appearance. Textural properties have an advantage in that they cannot be disturbed by errors that may occur during cell segmentation or other intermediate processing steps. Similar feature extraction strategies are described by (Doyle, S., Hwang, M,. Shah, K,2007). Finally, architectural features describe the spatial arrangement of cells within tissue. Here, we select the 20 nearest neighbors of each cell and measure the variance and deviation from the mean of key cell properties.

Splitter

It uses the OpenSlide library to open the large .svs image and export the requested chunk.

CombineTileData

It is a short matlab function to read in the output of ProcessSVS (when run on asingle .svs image), perform spatial analysis on it, and combine the results intoa single table of features for all the cells in the full-size image. If the image is larger than a single tile, then the 20 nearest neighbors of any cell may lie outside the tile boundaries. For this reason, the computation of nearest-neighbor features is performed by CombineTileData unless the image is smaller than a single tile.

3.2 The Dataset

The dataset was made of 10 full-size images from healthy patients (typically 40,000 × 40,000 pixels each), and 12 examples of LGSIL (Low Grade Squamous Intraepithelial Lesion) regions. The 21 LGSIL examples were hand-annotated regions within 11 images, outlined by pathologist Dr. OssamaTawfik.

The LGSIL sample in the dataset	The healthy Cell samples in the dataset
LGSIL_M12, LGSIL1_M12, LGSIL2_M12	**K-22**
LGSIL_K63	**K-20**
LGSIL_M4	**K-27**
LGSIL_M6	**K-31**
LGSIL_M13,LGSIL1_M13, LGSIL2_M13,LGSIL3_M13	**K-33-002**
LGSIL_M25, LGSIL1_M25	**K-36**
LGSIL_M26, LGSIL1_M26	**K-56**
LGSIL_M27	**K-144**
LGSIL_M36, LGSIL2_M36	**P-5**
LGSIL_M37, LGSIL1_M37, LGSIL2_M37	**P-20**
LGSIL_HGSIL1_M35	

Table 1: Database images

From each sample through the process described in 3.1, 54 features are extracted and aggregated into one file for each sample. The first two features are non-classification feature like cell location (Image Number, Object Number)

Morphological Features:

Morphological features describe physical properties of cells or the statistics of those properties over groups of cells. In this study the morphological features are extracted using Cell Profiler. In the dataset there are 5 morphological features: Nuclei Area, Nuclei Perimeter, Nuclei Area/Nuclei Perimeter, Nuclei extent, and Nuclei FormFactor.

Nucleoli Feature:

Given the nuclei major and minor axis and orientation 7 nucleoli features were computed. Contrast , correlation , energy , homogeneity were the basic ones and depending on circular window and patch number of gray pixels(between 0.1 and 0.9), number of black pixels(less than equal to 0.1) and nbhalf (less than equal to 0.5) were computed.

Textural Features:

Textural features describe low-level visual properties, which, in the case of histologic diagnosis applications, are designed to capture visual qualities including increases in nucleus size and the prominence of nucleoli. These features are measured by the histograms and co-occurrence matrices of pixel colors. Such features capture the first and second order statistics of cellular appearance. Textural properties have an advantage in that they cannot be disturbed by errors that may occur during cell segmentation or other intermediate processing steps. Similar feature extraction strategies are described by (Doyle, S., Hwang, M,. Shah, K,2007).Provided with the location of the cells in a sample, 12 textural features are computed in Matlab. Texture features are extracted separately from each "color band". With 25, 50 and 75 pixel widths at each radius contrast, correlation, energy and homogeneity are computed and this makes the 12 textural features.

Color Features:

During computing the color features three different windows with radii 5, 10 and 25 were taken and 6 features were computed for each window. The first three are mean patch and next three are variance patch.

Other than theses there are 5 features which are mean of selected morphological features and 5 features which are variance of morphological features.

Below is a table describing the representation and count of features of the dataset:

Feature Type	**Represented by**	**Count of Features**
Non Classification Features Like cell location	-1	2
Morphological Features from Cell Profiler	1	5
Texture Features(computed in Matlab)	3	12
Color Features	4	18
Nucleoli features	5	7
Mean of morphological features	1001	5
Variance of Morphological features	2001	5

Table 2: Features of the Dataset

So, as a whole in the dataset for each cell there were total 54features. Among which 2 are non-classification features like cell location, 5 morphological features from cell profiler, 12 texture features computed in matlab,18 color features, 7 nucleoli feature, 5 features which are mean of morphological feature, 5 feature which are variance of morphological features.

3.3 Outline of Processing Dataset

In general, any classification system consists of two stages:

(i) Training the classifier to learn the system parameters and

(ii) Testing the system to evaluate the success of the classifier.

If more data is used in training then it leads to better system designs, whereas if more data is used in testing that leads to more reliable evaluation of the system. Since the amount of available data was limited, and it is very important to test the network with extra data, the task to divide the whole dataset into training dataset and testing dataset was also very important. To train the network - different sets of training data with different ratios were used and then their performances were tested on the test set.

In statistics, machine learning, neural networks and other related areas one of the most discussed error is accurate estimation of generalization while working on a finite dataset. There are several kinds of estimation techniques proposed and examined in literature. Cross validation is a strong technique which is used to examine the reliability on the results. But there are different kinds of cross validation techniques which are being compared with each other to verify which is better for selecting a classifier. Leave one cross validation, k-fold cross validation are prime among those.

“Leave-one-outcross-validation has been shown to give an almost unbiased estimator of the generalization properties of statistical models, and therefore provides a sensible criterion for model selection and comparison.”(Gavin C. Cawley,2004)

It’s really important to evaluate the performance of a classifier accurately to predict the future assessment and more than that choosing a classifier for a given kind of dataset among several classifiers. Choosing a classifier or combining classifiers to get better performance out of it is one big task.

Performance was evaluated using a leave-out-two procedure: the network was trained on all-but-one healthy image and all-but-one LGSIL image, and then tested on the remaining healthy image and LGSIL image.

3.4 Classification Methods Used

In this study the whole dataset is trained and tested on two kinds of classifiers

1. Support Vector Machine and
2. Deep Belief Network along with back propagation.
3. SVM on raw features appended with high weight DBN features

In the first approach the extracted features are used for training Support Vector Machine with Gaussian Kernel Function and then the test dataset was used to see the performance. For this method the whole dataset was divided into several training and testing datasets to measure the performance of the system.

In the second approach, the dataset is modified a little bit (which will be explained in details in the next section) the whole dataset was divided into two parts Training and Testing. With the help of the training dataset, the learning process for the deep belief network was achieved. Using three layers and after extracting features two output nodes were assigned and according to the targets back propagation algorithm is used to fine tune the weights.

For the third approach high weight DBN features are appended with the raw features and then support vector machine with Gaussian Kernel Function is used to see the performance. All these methods are discussed in details in the next sections.

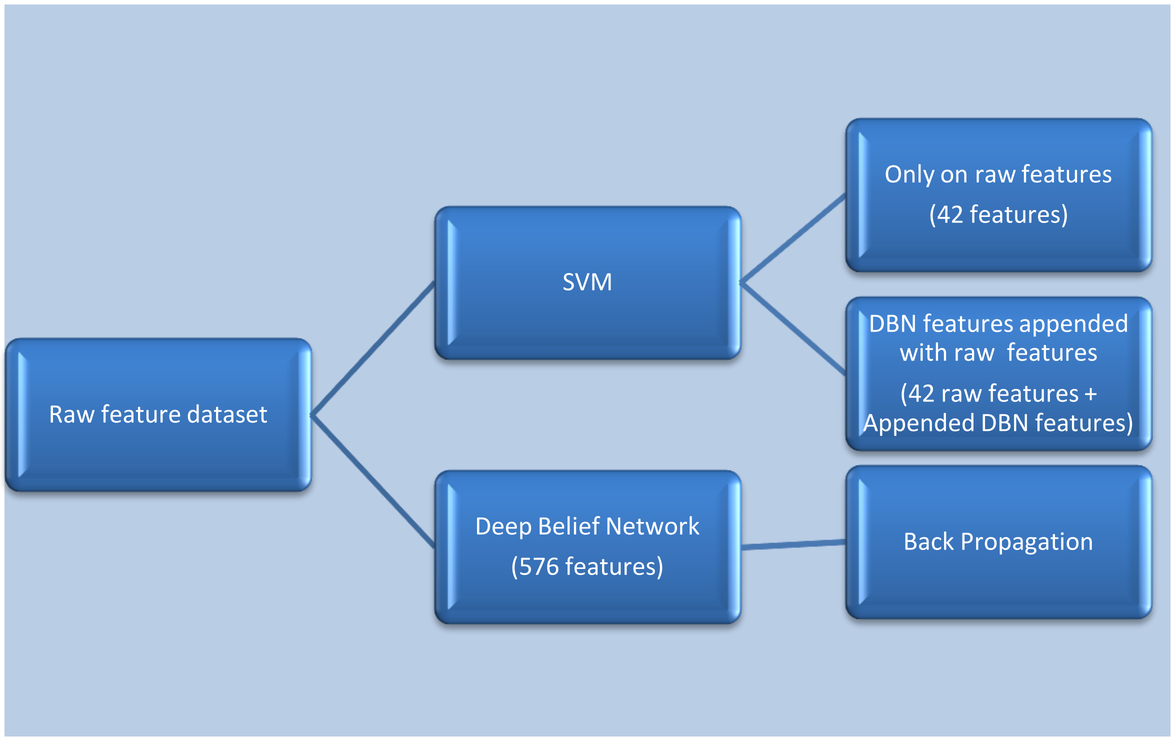

Raw feature dataset
SVM
Deep Belief Network (576 features)
Only on raw features (42 features)
DBN features appended with raw features (42 raw features + Appended DBN features)
Back Propagation

Figure 6: Flowchart of the Method

Chapter 4: Applying SVM

4.1 Support Vector machine

SVM is a learning technique developed by V. Vapnik and his team (AT&T Bell Labs., 1985) that can be seen as a new method for training polynomial, neural network, or Radial Basis Functions classifiers. The decision surfaces are found by solving a linearly constrained quadratic programming problem. This optimization problem is challenging because the quadratic form is completely dense and the memory requirements grow with the square of the number of data points. We present a decomposition algorithm that guarantees global optimality, and can be used to train SVM's over very large data sets.

4.2 Partitions

SVM light is used for this study. Performance was evaluated using a leave-out-two procedure: the software was trained on all-but-one healthy image and all-but-one LGSIL image, and then tested on the remaining healthy image and LGSIL image. This procedure was repeated 84 times (4 times for each LGSIL image), each time selecting a different pair of images for testing. All performance values are averaged over these 84 trials. The output of provides a numerical value for each cell: higher values indicate LGSIL, while lower values indicate healthy cells. With a bias of −0.92, the software correctly labeled 97.5% of healthy cells, and 92.4% of LGSIL cells. Other choices of bias produce different levels of trade-off between these two values. For example, a bias of −0.82 results in correct identification of 98.0% of healthy cells and 90.8% of LGSIL cells. Performance is shown under different bias levels in figure 7.

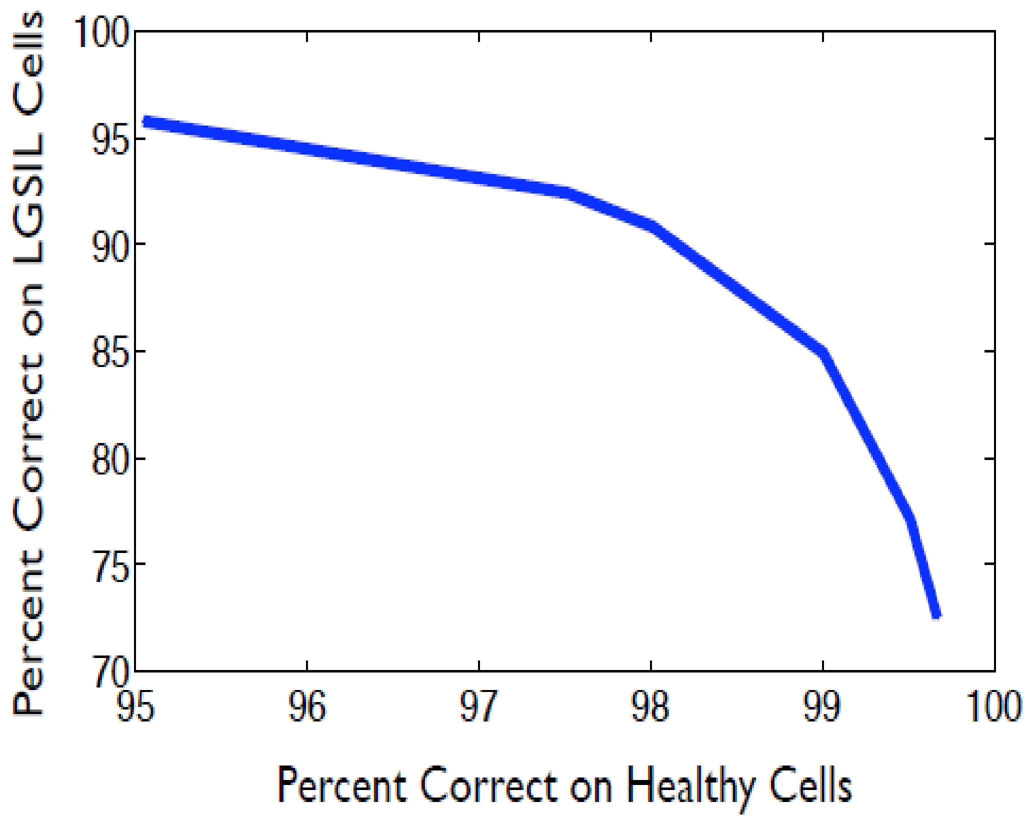


Figure 7: Performance of the algorithm, under different levels of bias. With a bias of -0.92, the software correctly labeled 97.5% of healthy cells, and 92.4% of LGSIL cells.

4.3 Results

As a result 438,194 cells within the healthy images and 2,350 cells within the LGSIL images are located. Typical results are shown in figures 8 and 9.

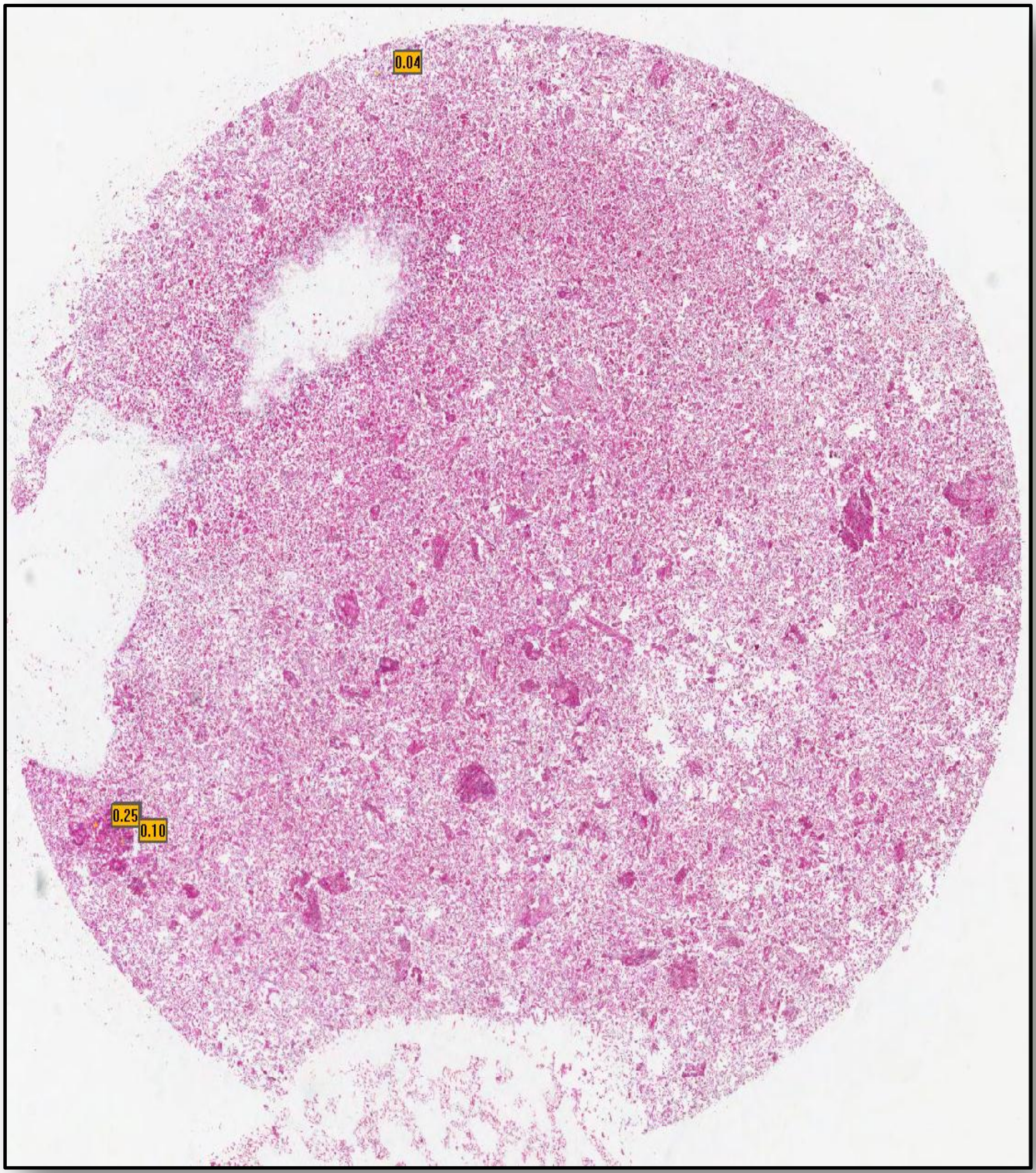


Figure 8: Typical results for samples from a healthy patient. Four mildly suspicious cells (SVM response >0) were discovered by the program. Only suspicious cells are shown

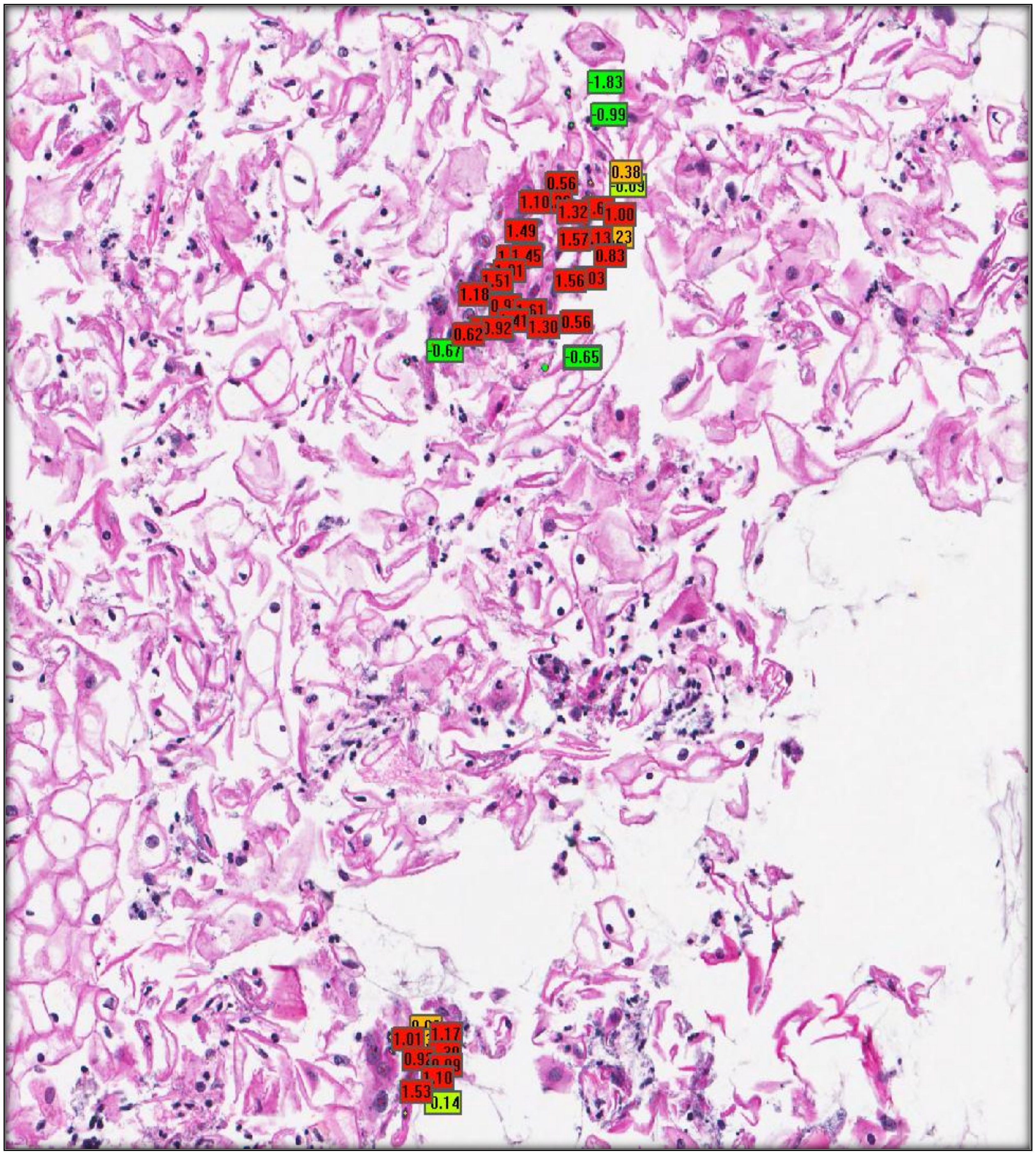


Figure 9: Results for an example LGSIL image. Two regions were graded LGSIL by OssamaTawfik. Several strongly suspicious cells were discovered by the software in both regions. Only cells from within the known-unhealthy regions are labeled.

LGSIL	Healthy	Identified as LGSIL (True Negative)	Actually LGSIL (True Negative+ False Positive)	Identified as Healthy (True Positive)	Actually Healthy (True Positive + False Negative)
LGSIL-HGSIL1_M35	K-144	12	15	23098	23098
LGSIL3_M13	K-144	138	158	23098	23098
LGSIL_M6	K-144	10	21		
LGSIL1_M12	K-20	155	355	75149	75216
LGSIL_K63	K-20	6	8	74988	75216
LGSIL1_M13	K-22	301	310	18050	18051
LGSIL_M12	K-22	130	181	18051	18051
LGSIL_M25	K-27	1	1	47889	47909
LGSIL_M13	K-27	53	55	47891	47909
LGSIL1_M26	K-31	110	128	40442	40442
LGSIL_M25	K-31	2	12	40442	40442
LGSIL1_M37	K-33-002	0	12	32229	32257
LGSIL_M26	K-33-002	126	153	3227	32257
LGSIL2_M12	K-36	18	42	68490	68490
LGSIL_M27	K-36	10	40	68489	68490
LGSIL2_M13	K-56	540	557	43329	44694
LGSIL_M36	K-56	36	42	43466	44694
LGSIL2_M36	P-20	17	18	57624	57644
LGSIL_M37	P-20	37	131	57625	57644
LGSIL2_M37	P-5	11	87	29162	29162
LGSIL_M4	P-5	13	24	29162	29162

Table 3: Results of applying SVM

4.5 Learning and Classification

Support vector machines (SVMS) are used for these classification. We use kernelized soft margin SVMs, with σ = 0.005, selected via cross-validation. SVMs are implemented using the SVMlight library (Thorsten Joachims,1999 2008) Training and classification are performed by LOO CancerEvaluation, which accepts the indices of the two images to be excluded, trains on the remaining images, and then tests performance on the two excluded images. Each component is described in more detail below.

4.6 Details result of SVM on one test set

The whole Dataset was divided into two parts training and testing. K22 and LGSIL-M12 were left for testing and rest of all was taken to use for training. In the test dataset there are 181 LGSIl and 18051 Healthy cells. The Deep Belief Network identified 137(137 + 0) as LGSIL among which 137were LGSIL.18095 (44+ 18051) were identified as Healthy, among which 18051 were "Healthy". So, overall 18188 cells were correctly classified from 18232 cells. So, overall performance was 99.756%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	137	0
Healthy(1)	44	18051

TP= 18051

True Positive (Predicted as positive (Healthy) and they are also positives (Healthy) (1, 1))

TN=137

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 44

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL)(1,-1))

FN=0

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**99.76%**
Specificity	TN/(TN+FP)	75.69%
Recall(Sensitivity)	TP/(TP+FN)	100.00%
Precision	TP/(TP+FP)	99.76%
F-measure	2xRecall x Precision/(Precision + Recall)	99.88%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	0.87

4.7 Details result of SVM on second test set

In the second datset P-20 and LGSIL-M37 were left for testing and rest of all was taken to use for training. In the test dataset there are 131 LGSIL and 54507 Healthy cells. The Deep Belief Network identified 37(37 + 0) as LGSIL among which 37 were LGSIL.54601 (94+ 54507) were identified as Healthy, among which 54507 were “Healthy”. So, overall 54544 cells were correctly classified from 54638 cells. So, overall performance was 99.83%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	37	0
Healthy(1)	94	54507

TP= 54507

True Positive (Predicted as positive (Healthy) and they are also positives (Healthy) (1, 1))

TN=37

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 94

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL)(1,-1))

FN=0

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**99.83%**
Specificity	TN/(TN+FP)	28.24%
Recall(Sensitivity)	TP/(TP+FN)	100.00%
Precision	TP/(TP+FP)	99.83%
F-measure	2xRecall x Precision/(Precision + Recall)	99.91%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	0.53

Chapter 5: Applying Deep Belief Nets

5.1 Introduction and Motivation

Deep Belief Networks are kind of new innovation in the field machine learning. It is capable of extracting distinctive features from a dataset and no label is needed for extracting features. So far Deep Belief Networks have not been applied to cytological Cancer Diagnosis. But it has shown promising results on similar methods. Deep belief nets have been used for generating and recognizing images (Hinton, Osindero&Teh 2006, Ranzato et. al. 2007, Bengioet.al., 2007), video sequences (Sutskever and Hinton, 2007), and motion-capture data (Taylor et. al. 2007).

In most of the other classification systems labeled data is used for training and the task of finding labeled data is itself pretty tough. Even for Cancer Diagnosis, if meaningful features can be extracted without labeled data then it can be of great significant in the course of cancer diagnosis.

5.2 Dataset Modified

The dataset which is used for Support Vector Machine analysis had 54 features. For the analysis with Deep Belief Networks 576 more features are added, totaling it to 630 features. In the successful applications of Deep Belief Networks, in the visible layer more of raw features are fed and from that input with layer by layer processing meaningful features are extracted.

For the same reason to make the input layer of the DBN more complete and similar to previously used datasets more features are added. But as the cancer cell samples are pretty huge and rich, adding the whole image pixel is not a very good idea. So to have a fewer dimension a different approach is taken. These features are mean color values within different regions centered at the nucleus center. Each feature is a section of an annulus that spans 45 degrees (so, 8 sections), and annuli range from 2 pixels to 50 in steps of 2 (24 annuli). Since there are 3 colors, that is 576 features.

5.3 Feature Extraction Using DBN:

.While modeling multi network there are basically two questions which come up.

(i) How many hidden layers and

(ii) How many neurons in each layer

According to Hinton, G. E. (2007) Generative models with only one hidden layer are much too simple for modeling the high-dimensional and richly structured sensory data that arrive at the cortex, but they have been pressed into service because, until recently, it was too difficult to perform inference in the more complicated, multilayer, nonlinear models that are clearly required.There have been many attempts to develop multilayer, nonlinear models (Lewicki, M. S., &Seinowski, T. J. 1997), (Hoyer, P. O., &Hyya, R. A. 2002), (Portilla, J., Strela, V., Wainwright, J., &Simoncelli, E. P. 2003). In Bayes nets (also called belief nets), which have been studied intensively in artificial intelligence and statistics, the hidden variables typically have discrete values. Exact inference is possible if every variable only has a few parents. This can occur in Bayes nets that are used to formalize expert knowledge in limited domains, but for more densely connected Bayes nets, exact inference is generally intractable.Extensive experiments by YoshuaBengio'sgroup suggest that several hidden layers is better than one. Results are fairly robust against changes in the size of a layer, but it is said the top layer should be big

Hinton et al. (2006) introduced a greedy, layer-by-layer unsupervised learning algorithm that consists of learning a stack of RBM's one layer at a time. After the stack of RBM's has been learned, the whole stack can be viewed as a single probabilistic model, called a "deep belief network" In this study, depending on the raw feature set which we planned to use, the Deep Belief Network was structured with four layers.

Next deciding the number of neurons in each layer is a very important part of deciding overall network architecture. Using too few neurons in the hidden layers will result in something called underfitting. Underfitting occurs when there are too few neurons in the hidden layers to adequately detect the signals in a complicated data set.

Using too many nodes in the hidden layers can result in several problems. First, too many nodes in the hidden layers may result in overfitting. Overfitting occurs when the network has so much information processing capacity that the limited amount of information contained in the training set is not enough to train all of the nodes in the hidden layers. A second problem can occur even when the training data is sufficient. An inordinately large number of nodes in the hidden layers can increase the time it takes to train the network. The amount of training time can increase to the point that it is impossible to adequately train the network. Obviously, some compromise must be reached between too many and too few nodes in the hidden layers.

In the Dataset used for Deep Belief Network structure the input data is highly enriched with image information along with some extracted feature set, totaling it to 576 features. For this reason, we expect more information processing in the first layer, so we used 1000 nodes as the first layer. Then after extracting that information and using those data as the input for the next layer, 400 nodes are used as next hidden node layer. And in the next layer 200 nodes are used.

The idea was to extract some meaningful features which can be distinct enough to classify them into two categories. After training the DBN with 3 layers, the last layer is the output layer which has 2 nodes, one signifying LGSIL cell and another Healthy.

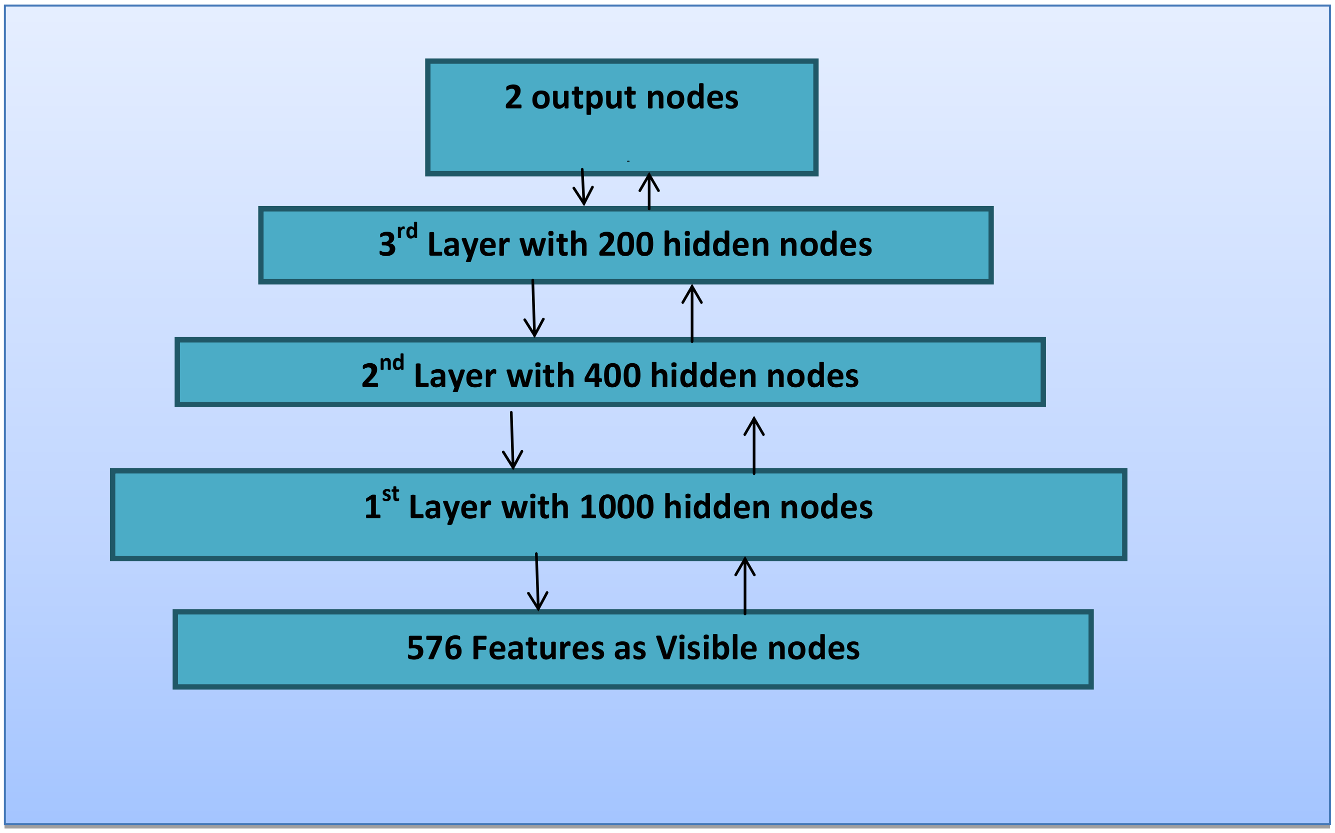

2 output nodes
3rd Layer with 200 hidden nodes
2nd Layer with 400 hidden nodes
1st Layer with 1000 hidden nodes
576 Features as Visible nodes

Figure 10: Structure of the Deep Belief Network used in this study

5.4 Partitions

The dataset which we have has more “Healthy Cell” samples than “LGSIL” cells. K-22 and LGSIL-M12 are left for testing and rest of all the cells are used for training. At this point the dataset consists of 409580 healthy cell instances and 2350 LGSIL cell instances. As deep belief network perform better on a balanced dataset, the LGSIL cells were repeated 200 times to make the DBN dataset balanced. So after the processing the dataset consists of 409,580 Healthy and 470,000 LGSIL instances, so total 879580 cells for training. And during testing K-22 and LGSIL-M12 consist of 18051 healthy and 181 LGSIL cell so overall 18232 cells.

In this study only one partition is used. The back propagation for 145 epochs took more than 1 week. So, it is tested on only one dataset.

5.5 Normalization

Normalization is a very important step for any algorithm to perform well. This normalization is required because DBNs treat node values as activation probabilities. As the feature values range a lot among different features the dataset was normalized to have feature values between 0 and 1. Simply normalization is done so that the values are uniformly distributed within distinct range.

If “A” is a matrix then the normalization techniques which is used in this study is as follows:

$$\mathbf{A} = \frac{(\mathbf{A} - \mathbf{min}(\mathbf{A}(:)))}{(\mathbf{max}(\mathbf{A}(:)) - \mathbf{min}(\mathbf{A}(:)))}$$

5.6 Training the Deep Belief Network

The dataset which is used in this study has 630 raw features extracted from the digital slides and it is hard to say which features have important information in predicting cells cancerous or healthy. So the first objective is to reduce the dimensionality of the dataset and take it down to fewer features which can be used easily for classifying purposes. So, removing the redundant features and making count of significant features are the main objective of training the Deep Belief Network.

While feeding the raw features to the visible nodes of the Deep Belief Networks, cell location features and mean and variance of the morphological features were not used.The objective is to extract meaningful features which plays important role to predict whether a cell is cancerous or healthy, in which cell locations are not supposed to play any role. So, only 576 features fed into the first layer or visible unitof the network..The Deep Belief Network as mention earlier three layers with 1000 nodes, 400 nodes and 200 nodes consecutively.

A deep belief net can be viewed as a composition of simple learning modules each of which is a restricted type of Boltzmann Machine that contains a layer of *visible units* that represent the data and a layer of hidden units that learn to represent features that capture higher-order correlations in the data.

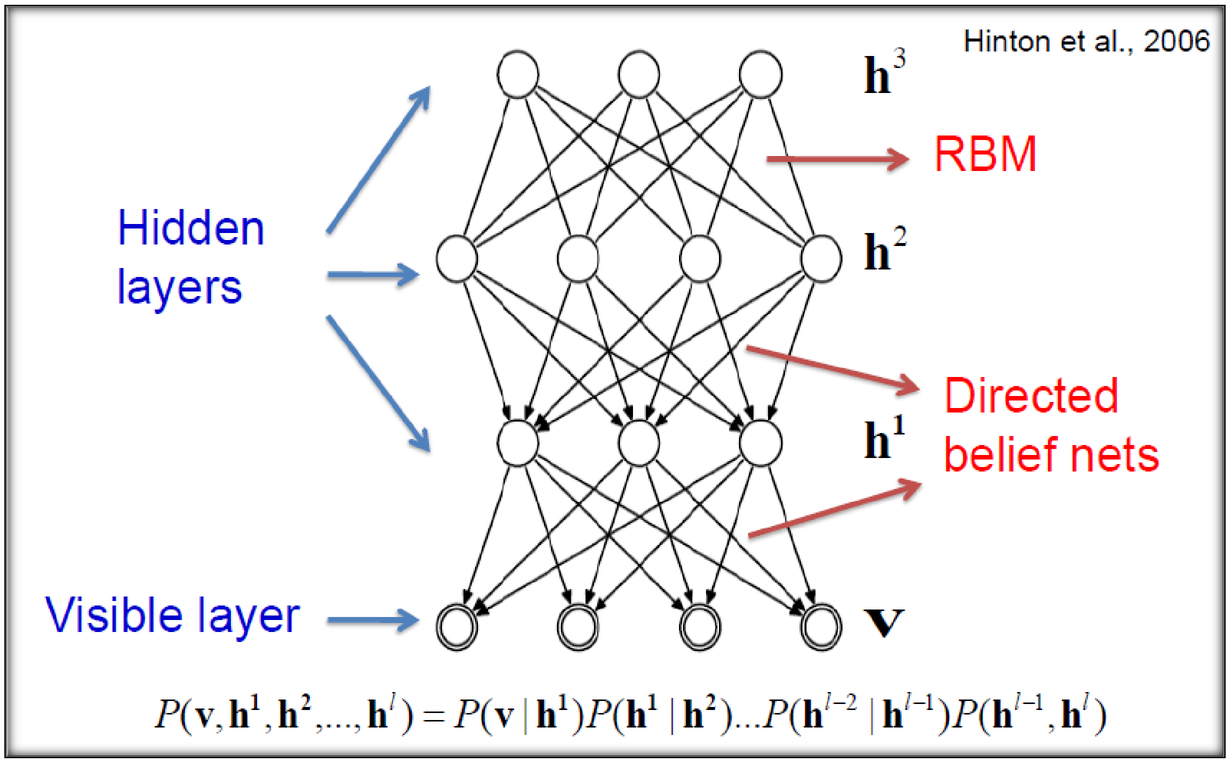

Hinton et al., 2006
$\mathbf{h}^3$
RBM
Hidden layers
$\mathbf{h}^2$
Directed belief nets
$\mathbf{h}^1$
Visible layer
$\mathbf{v}$
$P(\mathbf{v},\mathbf{h}^1,\mathbf{h}^2,...,\mathbf{h}^l) = P(\mathbf{v}\,|\,\mathbf{h}^1)P(\mathbf{h}^1\,|\,\mathbf{h}^2)...P(\mathbf{h}^{l-2}\,|\,\mathbf{h}^{l-1})P(\mathbf{h}^{l-1},\mathbf{h}^l)$

Figure 11: Structure of DBN

So, in the first step a Restricted Boltzmann Machine (RBM) is constructed with 576 raw features (v) in the visible unit and 1000 hidden nodes (h).After training the first layer and extracting features the second layer is to be trained.

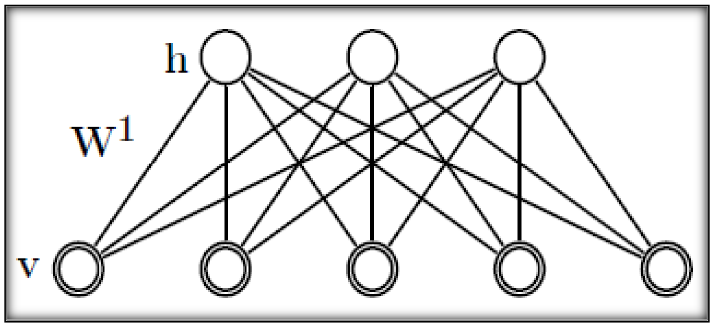


Figure 12: Bottom layer of DBN

So now another 400 features are stacked on top of the RBM to make another RBM. So this time the 1000 features (h1) which are extracted from the first layer of the Deep Belief Network is used as the visible nodes for the Restricted Boltzmann Machine and the second layer of the DBN which is 400 hidden nodes (h2) are used as the hidden nodes for the RBM. And in the same way like the first layer, this second layer is trained.

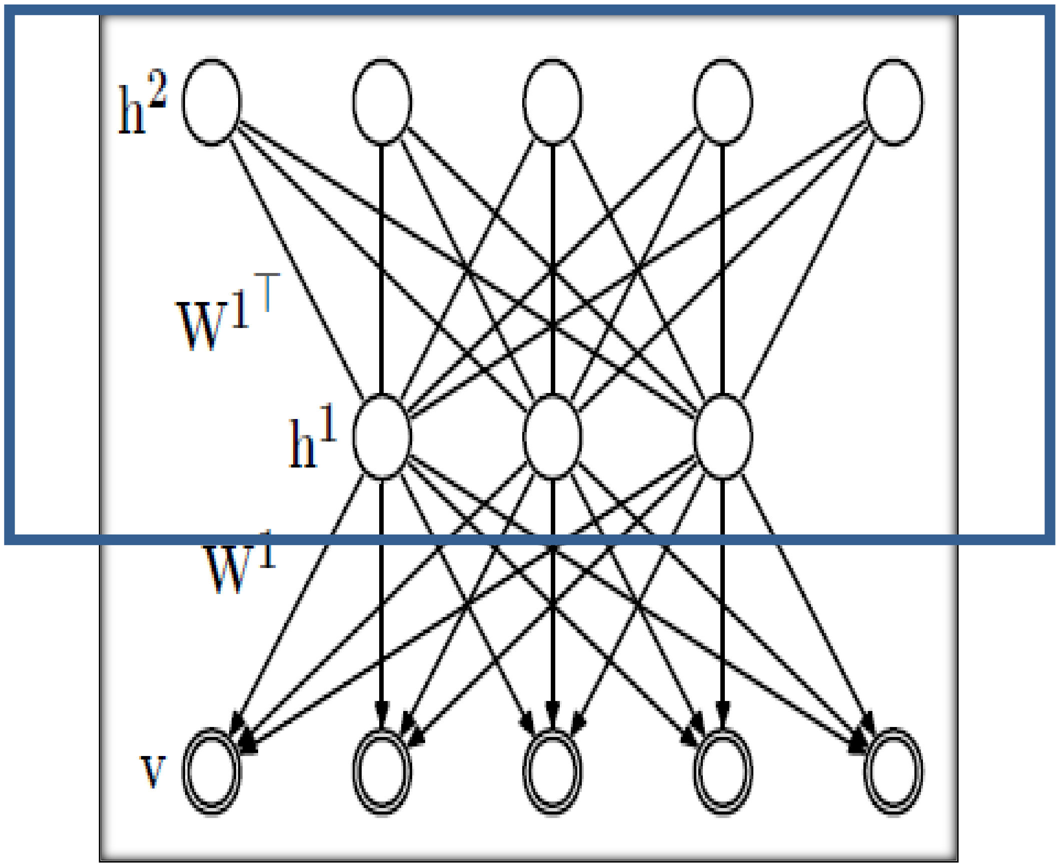


Figure 13: Second layer of DBN

Now another RBM is stacked on the existing network. So, 400 extracted features (h2) are used as the visible unit for the next layer RBM and this time 200 hidden nodes are used.

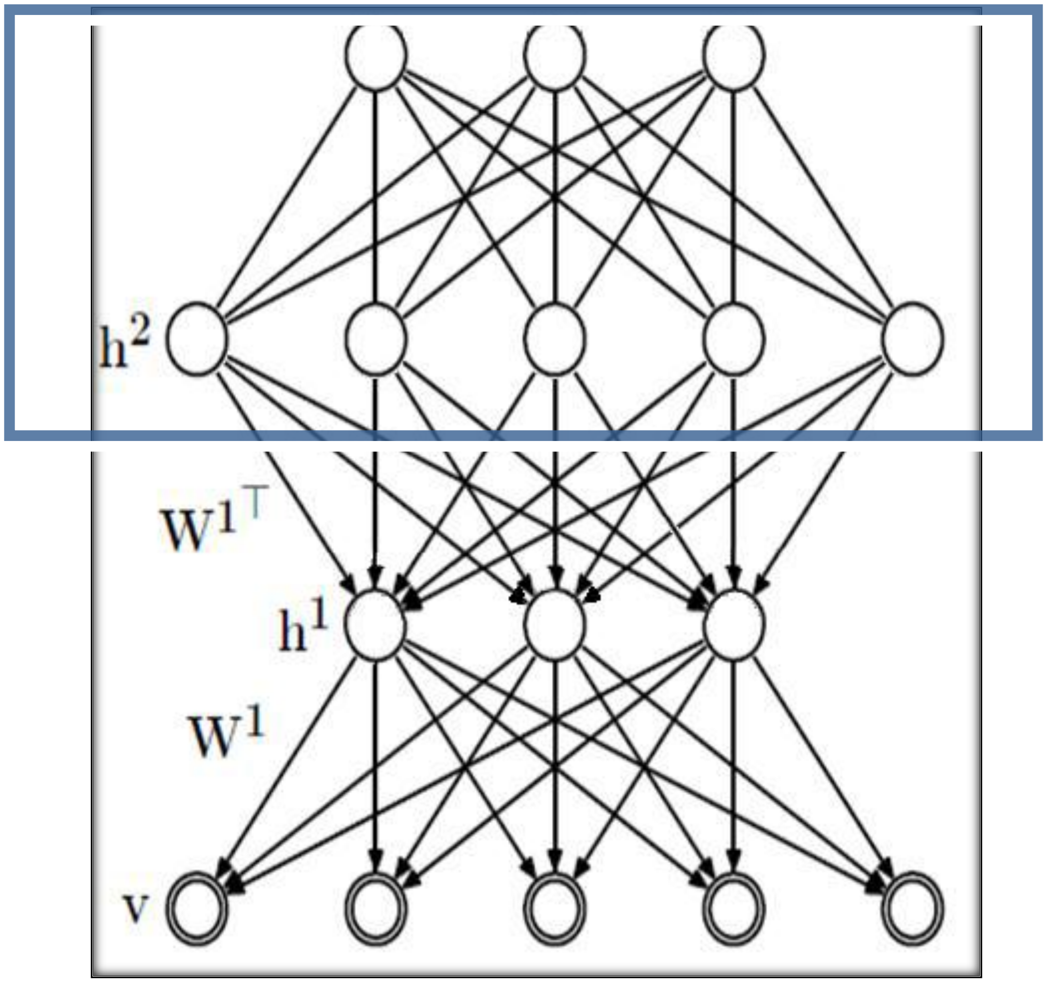


Figure 14: Top Layer of DBN

While running the code, the whole dataset is divided into batches and in each batch 100 cell instances are there. The whole dataset is randomized so that in each batch there is random number of healthy and LGSIL cells.

This layer by layer learning procedure is called Greedy Layer wise learning.

- It starts with the lowest level and stack upwards
- It trains each layer of auto-encoder on the intermediate code (features) from the layer below

- Top layer can have a different output (e.g., softmax non-linearity) to provide an output for classification

5.7 Back propagation

Back propagation has many problems like the gradient progressively gets more dilute, it gets stuck in the local minima. But when used with greedy pre-training it works better.
The back propagation starts once sensible features are detected which makes the initial gradients more sensible and back propagation only needs to perform a local search from a sensible starting point.

The Back-propagation algorithm is a supervised learning method for multi-layer feed-forward networks from the field of Artificial Neural Networks and more broadly Computational Intelligence. The name refers to the backward propagation of error during the training of the network.

After extracting 200 features from the 536 raw features, two output nodes are added one as LGSIL and Healthy cells. The initializing weights for the output node layer are again randomly assigned. As we have labeled data. Depending on the targets the error is back propagated to fine tune the weights.

During the training the error calculated is back propagated through the whole network. First total error at the top is calculated and then contribution to error at each step going forward is calculated.

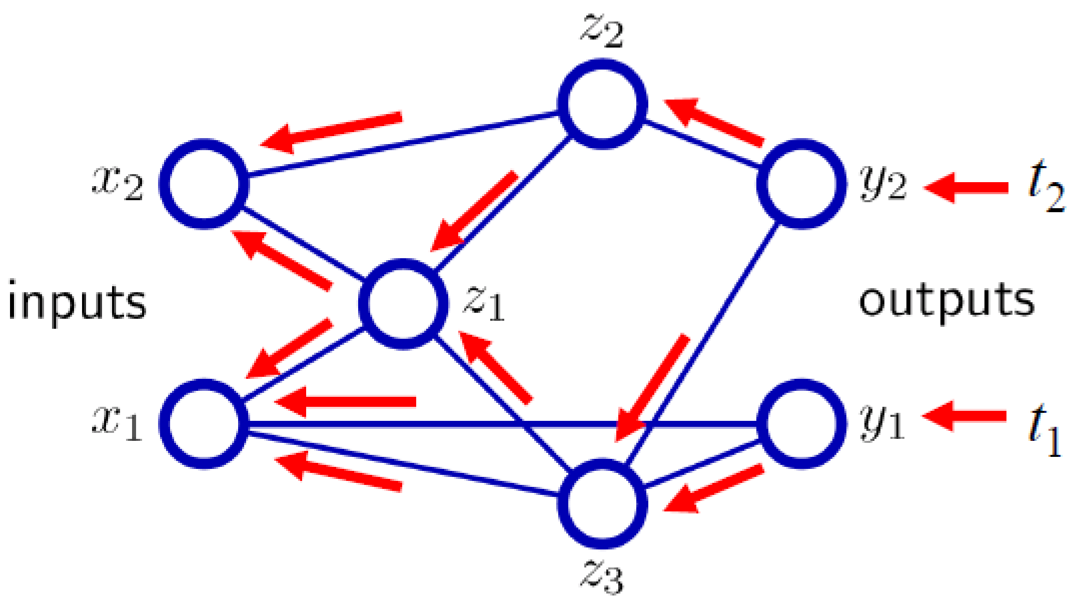


Figure 15: Back-propagation of error

In this study the propagation was done for 145 epochs. For the first 10 epoch's the output node layer weights are updating holding the other weights fixed and then in rest of the 135 epochs weights of all the layers are updated.

5.8 Results of DBN with back propagation on set 1

The whole Dataset was divided into two parts training and testing. K22 and LGSIL-M12 were left for testing and rest of all was taken to use for training. In the test dataset there are 181 LGSIl and 18051 Healthy cells. The Deep Belief Network identified 477 (159 + 318) as LGSIL among which 159 were LGSIL and 318 Healthy. 17755 (22+ 17733) were identified as Healthy, among which 17733 were actually "Healthy". So, overall 17892 cells were correctly classified from 18232 cells. So, overall performance was 98.14%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	159	318
Healthy(1)	22	17733

TP= 17733

True Positive (Predicted aspositive (Healthy) and they are also positives (Healthy) (1, 1))

TN=159

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 22

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL) (1,-1))

FN=318

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**98.14%**
Specificity	TN/(TN+FP)	87.85%
Recall(Sensitivity)	TP/(TP+FN)	98.24%
Precision	TP/(TP+FP)	99.88%
F-measure	2xRecall x Precision/(Precision + Recall)	99.05%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	0.53

While using only DBN features with back-propagation, it is able to classify more LGSIL cells than SVM only.

5.9 Results of DBN with back propagation on set 2

For the next datset, the whole Dataset was again divided into two parts training and testing. P-20 and LGSIL-M37 were left for testing and rest of all was taken to use for training. In the test dataset there are 131 LGSIl and 54507 Healthy cells. The Deep Belief Network identified 4170 (99 + 4071) as LGSIL among which 99 were LGSIL and 4071 were Healthy. 504468 (32+ 50436) were identified as Healthy, among which 50436 were actually "Healthy". So, overall 50535 cells were correctly classified from 54638cells. So, overall performance was 92.49%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	99	4071
Healthy(1)	32	50436

TP= 50436

True Positive (Predicted aspositive (Healthy) and they are also positives (Healthy) (1, 1))

TN=99

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 32

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL) (1,-1))

FN=4071

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**92.49%**
Specificity	TN/(TN+FP)	75.57%
Recall(Sensitivity)	TP/(TP+FN)	92.53%
Precision	TP/(TP+FP)	99.94%
F-measure	2xRecall x Precision/(Precision + Recall)	96.09%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	0.13

While using only DBN features with back-propagation, it is able to classify more LGSIL cells than SVM only.

Chapter 6: Combining SVM with DBN features

6.1 Motivation:

Training of Deep Belief Networks is unsupervised. So, the features extracted based on the first 3 layers (without back propagation) extracts some meaningful features which may be useful for Cancer Diagnosis or maybe totally irrelevant.

In this approach, from the extracted DBN feature nodes, nodes containing highest information were selected and appended with the raw features and used for SVM classification. The intention is to observe how the dataset will behave when unknown but important features are appended with some known features. The extracted features may play significant role or maybe they are not distinctive enough to predict a cell Healthy or LGSIL.But next question was: how many features to select from the 200 features. The number of raw features was 52, so it doesn't make sense to append high number of features with them. So, depending on the range of the weights in the connection a threshold was decided and the nodes whose connections had a value above that threshold were selected.

6.2 Selecting DBN features in the

After Deep Belief Network extract important information in 200 feature nodes, two output nodes are used to classify the sample as Healthy or LGSIL. With back propagation of error the whole network weights are fine-tuned after that. In the last layer high weight in the connection denotes features have important information in classifying cells. So depending on that, those few features were selected from 200 features which had high weight in the last layer (200 nodes and 2 output node layer) to classify cells LGSIL or Healthy. We took the absolute value of the weights. And if the absolute value was greater than 100 then 18 features are selected and when we gave the threshold as 120, 5 features are selected.

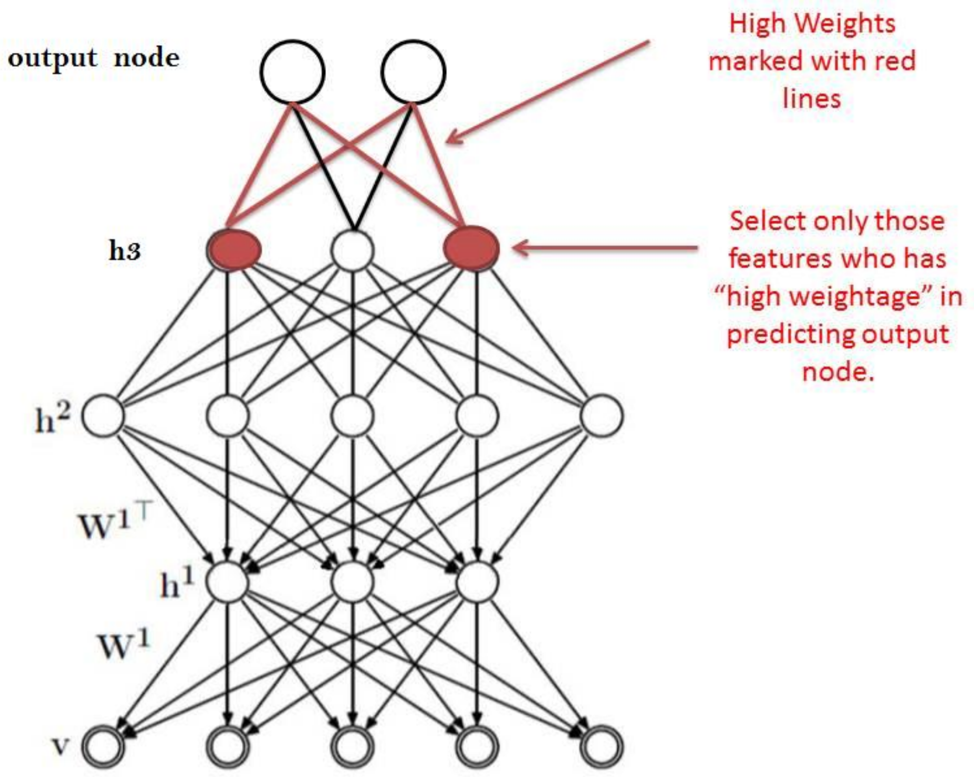

output node
h3
h2
W1⊤
h1
W1
v
High Weights marked with red lines
Select only those features who has "high weightage" in predicting output node.

Figure 16: Selection features from DBN

6.3 Results with Appended 5 DBN features on set 1

The whole Dataset was divided into two parts training and testing. K22 and LGSIL-M12 were left for testing and rest of all was taken to use for training. In the test dataset there are 181 LGSIL and 18051 Healthy cells. The Deep Belief Network identified 152(146 + 6) as LGSIL among which 137were LGSIL.18080 (35+ 18045) were identified as Healthy, among which 18045 were "Healthy". So, overall 18191 cells were correctly classified from 18232 cells. So, overall performance was 99.775%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	181	0
Healthy(1)	0	18051

TP= 18051

True Positive (Predicted as positive (Healthy) and they are also positives (Healthy) (1, 1))

TN=181

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 0

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL) (1,-1))

FN=0

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**100.00%**
Specificity	TN/(TN+FP)	100.00%
Recall(Sensitivity)	TP/(TP+FN)	100.00%
Precision	TP/(TP+FP)	100.00%
F-measure	2xRecall x Precision/(Precision + Recall)	100.00%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	1.00

The results show when 5 DBN features are appended with the raw features, the number of misclassification ofLGSIL decreases as well as it shows improvement in overall performanceof the classifier.

6.4 Results with Appended 5 DBN features on set 2

The whole Dataset was divided into two parts training and testing. K22 and LGSIL-M12 were left for testing and rest of all was taken to use for training. In the test dataset there are 181 LGSIL and 18051 Healthy cells. The Deep Belief Network identified 152(146 + 6) as LGSIL among which 137were LGSIL.18080 (35+ 18045) were identified as Healthy, among which 18045 were “Healthy”. So, overall 18191 cells were correctly classified from 18232 cells. So, overall performance was 99.775%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	181	0
Healthy(1)	0	18051

TP= 18051

True Positive (Predicted as positive (Healthy) and they are also positives (Healthy) (1, 1))

TN=181

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 0

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL) (1,-1))

FN=0

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**100.00%**
Specificity	TN/(TN+FP)	100.00%
Recall(Sensitivity)	TP/(TP+FN)	100.00%
Precision	TP/(TP+FP)	100.00%
F-measure	2xRecall x Precision/(Precision + Recall)	100.00%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	1.00

The results show when 5 DBN features are appended with the raw features, the number of misclassification of LGSIL decreases as well as it shows improvement in overall performance of the classifier.

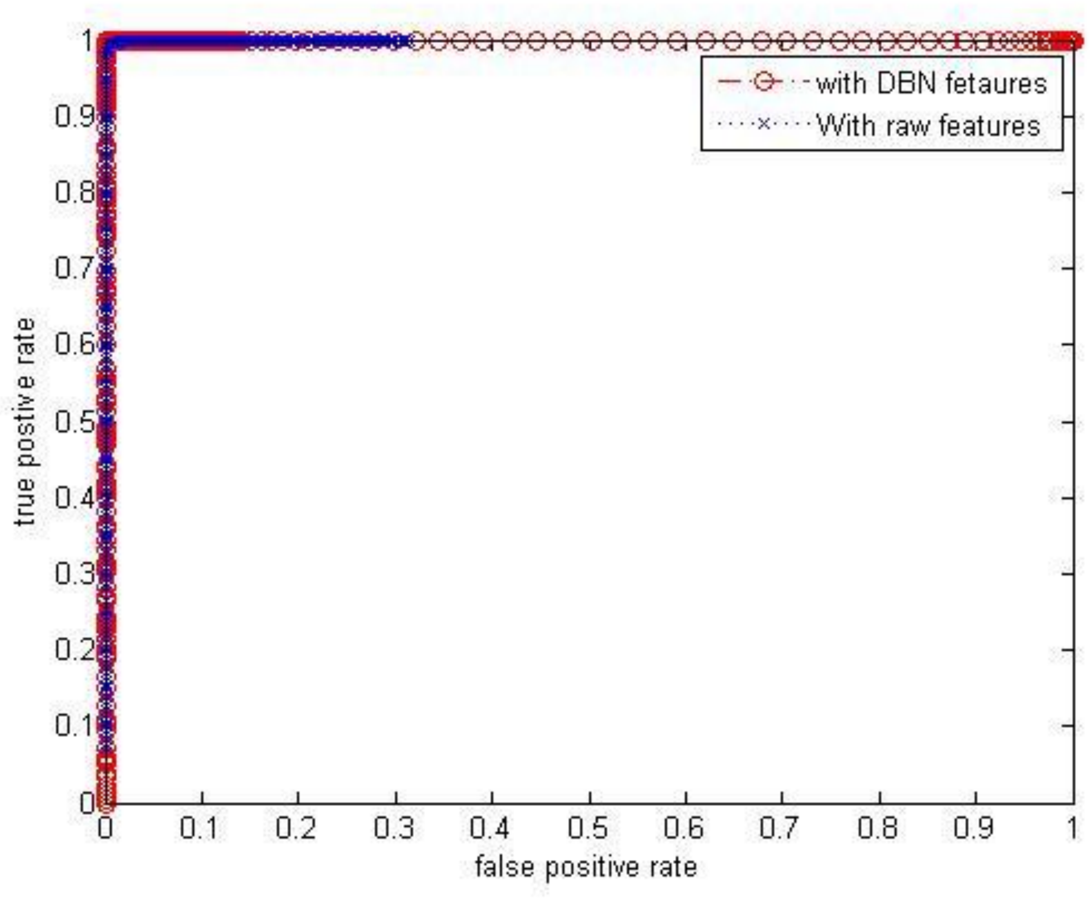


Figure 17: ROC curve for set1

6.3 Results with Appended 5 DBN features

The whole Dataset was divided into two parts training and testing. P-20 and LGSIL-M37 were left for testing and rest of all was taken to use for training. In the test dataset there are 131 LGSIL and 54507 Healthy cells. The Deep Belief Network identified 83 (80 + 3) as LGSIL among which 80 were LGSIL. 54555 (51+ 54504) were identified as Healthy, among which 54504 were "Healthy". So, overall 54584 cells were correctly classified from 54638 cells. So, overall performance was 99.90%

ACTUAL

PREDICTED

	LGSIL(-1)	Healthy(1)
LGSIL(-1)	80	3
Healthy(1)	51	54504

TP= 54504

True Positive (Predicted as positive (Healthy) and they are also positives (Healthy) (1, 1))

TN=80

True Negative (Predicted negative (LGSIL) and they are also negative (LGSIL) (-1,-1))

FP= 51

False Positive (Predicted as positive (Healthy) but actually are negative (LGSIL) (1,-1))

FN=3

False Negative (Predicted as negative (LGSIL) but actually are positive (Healthy) (-1,1))

Overall Accuracy	**(TP+TN)/(TP+TN+FP+FN)**	**99.90%**
Specificity	TN/(TN+FP)	61.07%
Recall(Sensitivity)	TP/(TP+FN)	99.99%
Precision	TP/(TP+FP)	99.91%
F-measure	2xRecall x Precision/(Precision + Recall)	99.95%
Correlation Coefficient	((TP x TN)-(FP x FN))/√((TP+FP)x(TP+FN)x(TN+FP)x(TN+FN))	0.77

The results show when 5 DBN features are appended with the raw features, the number of misclassification of LGSIL decreases as well as it shows improvement in overall performance of the classifier. In the ROC curve below it can be seen that with appended DBN features SVM gives better performance in this case also.

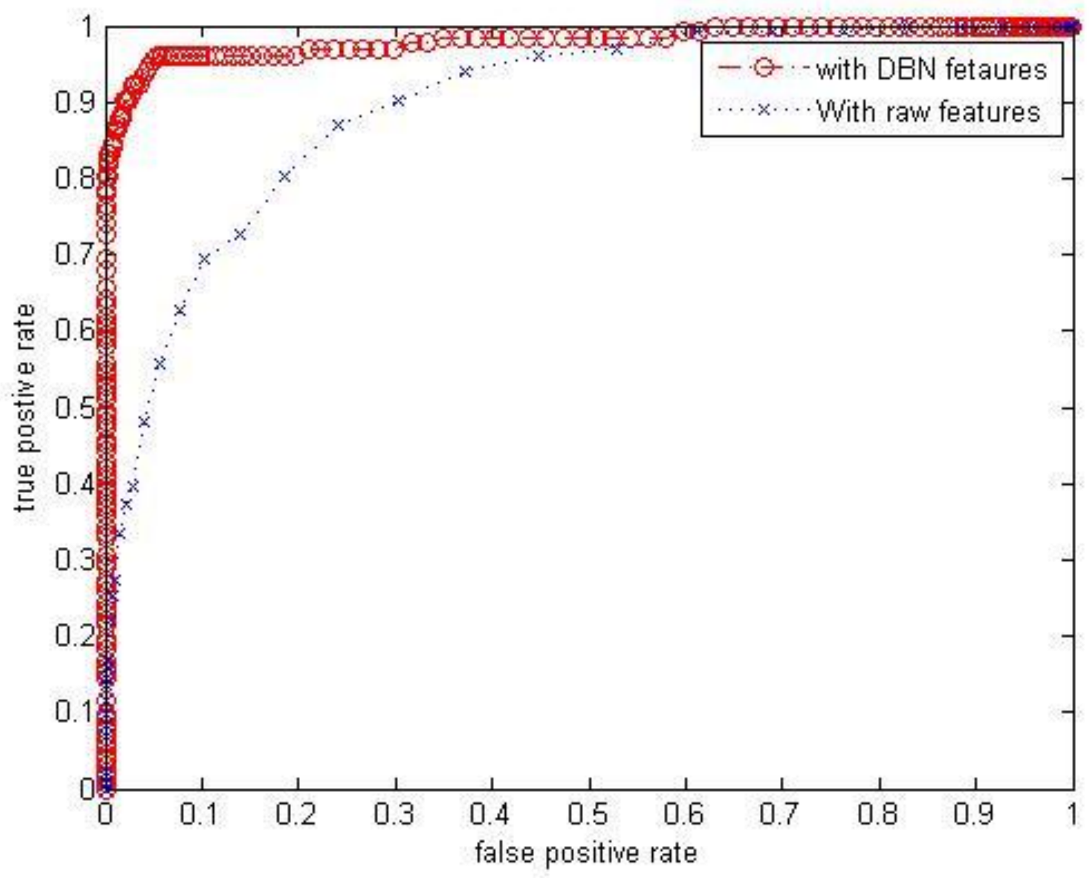


Figure 18: ROC curve for set2

Chapter 7: Conclusion and Discussions

The success of cancer treatment on a large extent depends on correct diagnosis. The automated cancer diagnosis has the capability to play a significant role in accurate diagnosis. In this study, we tried to analyze the data with some new prominent approaches and compare the results with some established classification methods.

7.1 Cell Block Preparation

This study utilizes the dataset, which is extracted from slides made from the cell block preparation made by pathologist Dr. Osama Twafik of University of Kansas Medical Centre (KUMC). It successfully allows semi-automated cancer diagnosis and it has the potential to improve patient care.

7.2 Summary

For cancer Diagnosis the raw dataset is high-dimensional and the structure is too complicated to be represented by a simple model. Another challenge is the lack of labeled data for this kind of analysis. To represent the complicated structure is itself a big task. Deep Belief Network in simple terms stacked Restricted Boltzmann Machines. And RBM provides a simple way to learn one layer of features without any supervision. And many layers of representation creates good generative model that can be fine-tuned.

7.3 Future Direction

In this study, the deep belief net approach is tried on only two sets because of the high computational time. If it can be tried on more than two training/testing split then it will be easy to conclude which DBN features are best. It will also help in selecting normalization method which will work best.

Selecting DBN features seems like an important aspect of the improvement and there may be better ways of doing this. For example, maybe the variance of the features should have been considered in addition to the weight. Maybe features can be found that were maximally discriminative using other means, like traditional feature selection techniques.

7.4 Recent improvements in Deep Learning

DBN techniques themselves are continually improving –so there are some methodologies which can be applied for analyzing the dataset of this study.

In the study, Swersky, 2010 provided a direct comparison of the Stochastic Maximum Likelihood algorithm and Contrastive Divergence for training Restricted Boltzmann Machines using the MNIST data set. They demonstrated that Stochastic Maximum Likelihood is superior when using the Restricted Boltzmann Machine as a classifier, and that the algorithm can be greatly improved using the technique of iterate averaging from the field of stochastic approximation. They further showed that training with optimal parameters for classification does not necessarily lead to optimal results when Restricted Boltzmann Machines are stacked to form a Deep Belief Network.

In the study by Plahl, 2012, they suggested the Sparse Encoding Symmetric Machine (SESM) as an alternative method for pre-training the deep belief network instead of Restricted Boltzmann Machines (RBM). The RBM objective function cannot be maximized directly. Therefore, it is not clear what function to monitor when deciding to stop the training, leading to a challenge in managing the computational costs. In this paper, they explore SESM to pre-train DBNs and apply this to speech recognition. Their results indicate that pre-trained DBNs using

SESM and RBMs achieve comparable performance and outperform randomly initialized DBNs with SESM providing a much easier stopping criterion relative to RBM. While the RBM relies on CD, SESM directly optimizes the objective function. Moreover, SESM provides easy criteria to stop the training, leading to less iteration. This technique maybe used with our dataset to see if it improves.

References

Koss, L. G. (1989). The papanicolaou test for cervical cancer detection. *Journal of the American Medical Association*, *261*(5), 737-743.

Taylor, G. W., Hinton, G. E., Roweis, S. (2007). Modelling human motion using binary latent

variables. In advances in neural information systems (Vol. 19) Cambridge, MA: MIT Press

In text-Demir, C., &Yener, Y. (2009).

Reference- Demir, C., &Yener, Y. (2009). Automated cancer diagnosis based on histopathological images: a systematic survey. *Technical report, rensselaer polytechnic institute*, *5*,

Stanikov, A., Tsamardinos, L., Dosbayev, Y., &Aliferris, C. F. (2005). Gems: A system for automated cancer diagnosis and biomarker discovery from microarray gene expression data. *International journal of medical informatics*, *74*, 491-503.

Beck, A. H., Sangoi, A. R., Leug, S., Marinelli, R. J., Vijver, M., West, R. B., Matt, R. &Neilsen, T. O. (2011, December 15). Systematic analysis of breast cancer morphology uncovers stromal features associated with survival. *Science transnational medicine*, *3*(108), 1-11.

Hinton, G. E. &Salakhutdinov, R. R. (2006, July 28). Reducing the dimensionality of data with neural networks. *Science transnational medicine*, *313*, 504-507.

Swersky, K., Chen, B., &Freitas, N. (n.d.). *A tutorial on stochastic approximation algorithms for training restricted boltzmann machines and deep belief nets*. Informally published manuscript, University of British Columbia, Department of Computer Science, University of British Columbia, Canada.

Hinton, G. E, Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural Computation, 18:1527-1554.

Salakhutdinov, R. R. and Hinton,G. E. (2007). Semantic Hashing.In Proceedings of the SIGIR Workshop on Information Retrieval and Applications of Graphical Models, Amsterdam.

Taylor, G. W., Hinton, G. E. &Roweis., S. (2007). Modeling human motion using binary latent variables.Advances in neural information processing systems 19, *MIT Press, Cambridge, MA*.
Joachims, T., (1999).Making large-scale support vector machine learning practical, p. 169–184.*MIT Press, Cambridge, MA, USA*.,

Bengio, Y., Lamblin, P., Popovici, P., &Larochelle, H., (2006). *Greedy layer-wise training of deep networks*.Informally published manuscript, Universite de Montreal, Universite de Montreal, Montreal, Canada.

Roux, N. L., &Bengio, Y. (2008). Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, *20*(6), 1631-1649.

Joachims, T. (1999) *Making large-scale support vector machine learning practical*, p 169–184.MIT Press, Cambridge, MA, USA.

Carpenter, A., Jones, T., Lamprecht, M., Clarke, C., Kang, I., Friman, O., Guertin, D., Chang, J., Lindquist, R., Moffat, J., Golland, P., & Sabatini. D. (2006) CellProfiler: Image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(10).

Doyle, S., Hwang, M,. Shah, K.., Madabhushi, A., Tomaszeweski, J,. & Feldman, M., Automated grading of prostate cancer using architectural and textural image features. In *International Symposium on Biomedical Imaging*, p. 1284–87, Washington DC, 2007.

Joachims, T. (2008, August 14). *Support vector machine*. Retrieved from http://svmlight.joachims.org/

Malpica, N., Solrzano, C., Vaquero, J., Santos, A., Vallcorba, I., Garca-Sagredo, J., &Pozo, F.,.(1997) Applying watershed algorithms to the segmentation of clustered nuclei. *Cytometry*, 28(4), 289–297.

W¨ahlby, C., Sintorn, F,.Erlandsson, G., Borgefors, &Bengtsson, E., (2004). Combining intensity, edge and shape information for 2D and 3D segmentation of cell nuclei in tissue sections.*Journal of microscopy*, 215(1), 67–76.

Denmir, C., &Yener, B. (2009). *Automated cancer diagnosis based on histopathological images: A systematic survey* . Informally published manuscript, RENSSELAER POLYTECHNIC INSTITUTE, DEPARTMENT OF COMPUTER SCIENCE,

Doyle, S., Agner, S., &Madabhushi, A. (2008, May).*Automated grading of breast cancer histopathology using spectral clustering with textural and architectural image features*. Paper presented at IEEE International Symposium on Biomedical Imaging 5th ieee international symposium on biomedical imaging: from nano to macro, Paris, France

Hinton, G. E. (2007). Learning multiple layers of representation.*Science direct*, *11*(10), 428-433.

Cawley, C. C., & Talbot, N. L. (2004).Fast exact leave-one-out cross-validation of sparse least-squares support vector machines.*Neural networks*, *17*(10), 1467-1475.

Joachims, T. (2008).*Support vector machine*. Informally published manuscript, Cornell University, Department of Computer Science, Retrieved from http://svmlight.joachims.org/

Lewicki, M. S., &Seinowski, T. J. (1997). Bayesian unsupervised learning of higher order structure. In M. Mozer (Ed.), *Advances in Neural Information Processing Systems* (Vol. 9, pp. 529-535). MIT Press.

Hoyer, P. O., &Hyya, R. A. (2002).*A multi-layer sparse coding network learns contour coding from natural images* . (pp. 1593-1605). Vision Res.

Karklin, Y., &Lewicki, M. S. (2003). Learning higher-order structures in natural images .*Network*, *14*, 483-499.

Schwartz, O., Senowski, T. J., & Dayan, P. (2006).Soft mixer assignment in a hierarchical generative model of natural scene statistics. *Neural comput*,*18*(11), 2680-2718.

Portilla, J., Strela, V., Wainwright, J., &Simoncelli, E. P. (2003). Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE transactions of image on processing*, *12*(11), 1138-1150.

Plahl, C., Sainath, T. N., Ramabhadran, B., &Nahamoo, D. (2012). Improved pre-training of deep belief networks using sparse enconding symmetric machnines. *Icassp*, *978*(1), 4165-4168.

Swersky, K., Chen, B., Marlin, B., &Freitas, N. (n.d.). *A tutorial on stochastic approximation algorithms for training restricted boltzmann machines and deep belief nets*.

Theano Development Team "*DeepLearning v0.1 documentation*" 2008-2010 LISA lab, Sep 04, 2012 <http://deeplearning.net/tutorial/contents.html#>