MATH SYSTEMS FOR DIAGNOSIS AND TREATMENT OF BREAST CANCER

1 Introduction

This year's Applied Math Master's Project investigated the detection, classification and growth of breast cancer tumors. The project was a collaborative effort by students **Connor Amorin**, **Gabriel P. Andrade**, **Sandra Castro-Pearson**, **Abdel Kader Geraldo**, **Brandon Iles**, **Dean Katsaros**, **Terry Mullen**, **Sam Nguyen**, **Oliver Spiro**, and **Melissa Sych** under the guidance of Professor Nathaniel Whitaker.

According to statistics from the CDC, breast cancer is the most common cancer diagnosis among women in the United States. Currently, radiology is an expert-based field where automated tools are at best relegated to the role of "second reader." Early detection is an enormously important part of breast cancer treatment, so our goal in this project is to create a machine learning pipeline for detection and diagnosis from mammogram images. We also model the growth and treatment of tumors using a system of ODEs. Due to a lack of human data, this last part of the pipeline uses data from experiments on lab mice, and is not restricted to breast cancer.

2 Image Dataset

The Curated Breast Imaging Subset of DDSM is an updated and standardized version of the Digital Database for Screening Mammography (DDSM). The database consists of 2,620 scanned film mammography studies. In addition to the mammography images, it contains the verified pathology information associated with each study. The images have been compressed and converted to DICOM format. The DDSM is one of few well-curated data sets available for research and Computer-Aided Diagnostics (CAD). Although useful, the set has problems concerning data size and noise, as the images are digitized from film. Our work follows a two-fold procedure. First, we implement a three-step image processing method for detecting abnormal masses in mammograms. Once we detect an abnormality, we seek to classify the mass as malignant or benign using classical machine learning techniques along with the histogram of oriented gradient feature extraction method and also using deep neural networks.

3 Detection

Tumors can be recognized as locally low density areas on mammograms, but they may vary in size, shape, image quality and more. Our image detection system consists of three distinct steps. First, we modify the local contrast of each pixel with a linear enhancement filter. Next, we subtract this enhanced image from the original and obtain an image with segmented masses. The third step binarizes the segmented image using a technique called adaptive local thresholding.

In order to enhance our original image, a transformation of pixel values is used which can be described in the following way:

Given a constant α, pixels in the original image $OI(i,j)$ are transformed as follows to give pixels in the enhanced image $E(i,j)$:

$$\text{If } OI(i,j) < \alpha,\ E(i,j) = a\log[1 + bOI(i,j)]$$

$$\text{If } OI(i,j) > \alpha,\ E(i,j) = \tfrac{1}{b}[\exp[\tfrac{OI(i,j)}{a}] - 1]$$

where α and a are chosen empirically, $b = \frac{(1-\exp(m/a))}{m}$, and m is the maximum grey level. The log function enhances the lower grey level (darker areas), and the inverse function enhances the higher grey levels (brighter areas).

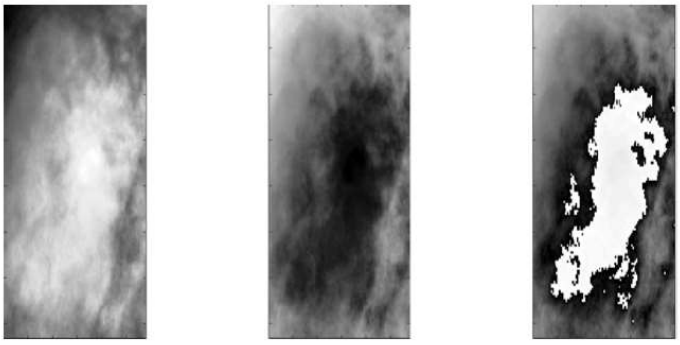

Figure 1: Original (left), enhanced (middle), and segmented (right) mammograms (GFP)

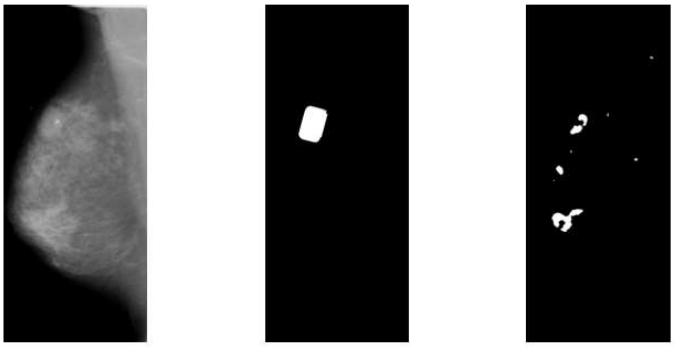

Figure 2: Original mammogram (left), the mask given by the data set (center), and the mask found by adaptive thresholding (right)

Segmentation of mass regions from normal tissue is achieved by subtracting the enhanced image *EI* from the original image *OI*. The segmented image, *SI* is the result.

Masses are generally more dense than the surrounding tissue, resulting in a bright spot on the segmented image. The image is binarized using adaptive local thresholding. For each pixel in the segmented image *SI*(*i,j*), the following criteria is used to classify the pixel as a potential mass pixel or normal pixel by the following rule:

$$\text{If } SI(i,j) \geq TH(i,j) \text{ and } SI_{dif} \geq M_{voisi}P$$

then *SI*(*i,j*) belongs to a suspicious area, else it belongs to normal tissue. *TH*(*i, j*) is the threshold value calculated by:

$$TH(i,j) = M_{voisi}P + \gamma SI_{dif}$$

$$\text{where } SI_{dif} = SI_{max}(i,j) - SI_{min}(i,j),$$

$M_{voisi}P$ is the average pixel intensity in a small area around the pixel *SI*(*i,j*), SI_{max} and SI_{min} are maximum and minimum intensity value in a large window drawn around the area, and γ is a thresholding coefficient that is chosen empirically.

4 Classification

Once an abnormality is detected, our aim is to classify it as either malignant or benign. To achieve this goal, we look to two classical machine learning techniques. Here, we use support vector machines and logistic regression with and without principle component analysis (PCA). These models were trained on HOG features as described in Ergin and Kilinc's *A New Feature Extraction Framework Based on Wavelets for Breast Cancer Diagnosis.*

A support vector machine (SVM) aims to find the hyperplane which maximizes the distance between the hyperplane and the nearest sample. Effectively, it separates the two classes, and creates the largest possible margin between the edge cases. Logistic regression is a special case of regression with a binary response variable. In logistic regression, the probability or odds of the response taking a particular value is modeled based on the combination of values taken by the predictors.

SVM achieved a 10-fold cross-validation accuracy of 63% for classifying mammogram images as having a benign or malignant tumor. Logistic regression performed poorly, obtaining 54% accuracy. Logistic regression with principle component analysis (PCA) performed the best, with an accuracy of 69%. These accuracies are not satisfactory, especially for medical purposes. We determined that our feature space was limiting our models, and for that reason we looked to neural networks.

To understand the neural network approach to the image classification task, we first need to pose the problem in a well defined manner.

Let $\mathcal{M} \subseteq [0,1]^{w \times h}$ be the space of mammogram images that are at most *w* pixels wide and *h* pixels high and let the labels {−1,1} represent a benign or malignant diagnosis respectively. Suppose there exists an optimal classification function $f^* : \mathcal{M} \to \{-1,1\}$. Then a neural network ultimately tries to fit an approximation function *f* to this true classifier function f^*. To this end, neural networks build *f* from the ground up by composing functions known as *layers* which in turn are formed from simple atomic functions known as *units*[1]. The units $g : \mathbb{R}^m \to \mathbb{R}$ are functions parametrized by a *weight vector* $\boldsymbol{w}$ and a scalar *bias b*. If *h* is a fixed nonlinear function[2] called an **activation**. Then

$$g(\boldsymbol{x}; \boldsymbol{w}, b) := h\left(\boldsymbol{w}^T \boldsymbol{x} + b\right)$$

A *layer* is then an aggregation of units into a column vector. If *d* is the dimension of this vector (called the *width* of the layer), then

$$f^{(i)} := \left(g_1^{(i)}, \ldots, g_d^{(i)}\right)^T$$

and the network is formed from the composition of these layers, with

$$f^{(i)}\left(f^{(i-1)}\right) = [g_1^{(i)}(f^{(i-1)}), \ldots, g_d^{(i)}(f^{(i-1)})]^T$$

Therefore if our network has *n* layers

$$f(\boldsymbol{x}) = f^{(n)}(\boldsymbol{x}) = f^{(n)}\left(f^{(n-1)}(\ldots(f^{(1)}(\boldsymbol{x})))\right)$$

where $x \in \mathcal{M}$. Once the data has been fed through all of these functions, we normalize the final layer and output {−1, 1} based on which class has higher probability. These networks are often represented graphically as in Figure 3. The above discussion gives the neural network prediction function, and it remains to describe the fitting process. We do this via a process known as "supervised learning" where initially we feed the network data $x \in \mathcal{M}$ that is labeled with its true class $y \in \{-1,1\}$. After the network has been given some of this labeled data we compute a *loss* function[3] to quantify how poor the network's predictions are on this data. We then use gradient descent to fit weights and biases to the data so that this loss is minimized.

[1]Also known as neurons.
[2]Usually a rectified linear or sigmoid function.
[3]This is similar to likelihood maximization used in linear regression, etc.

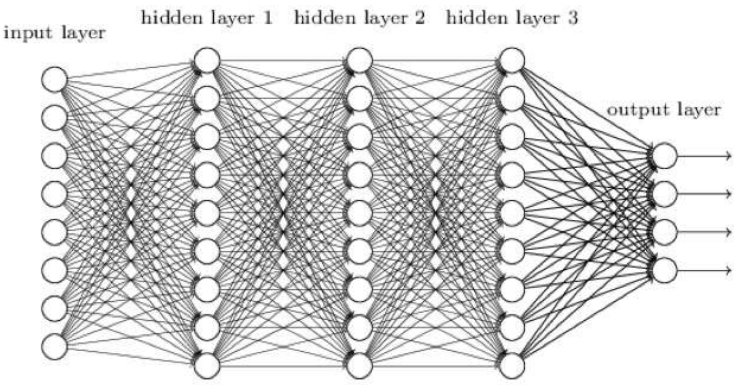

Figure 3: Fully connected network

The fully-connected network described above is not the architecture used for state-of-the-art image classification. Instead, we used a type of network known as a *convolutional neural network* (CNN). The functional form differs subtly in CNNs when compared to fully connected networks, but is considerably less intuitive. This is because the inner product in each atomic unit function is replaced by a (discrete) convolution, so

$$g_{mn}(\boldsymbol{x}; \boldsymbol{w}, b) = h\left((\boldsymbol{w} * \boldsymbol{x})_{mn} + b\right)$$

$$(\boldsymbol{w} * \boldsymbol{x})_{mn} = \sum_{k,l} w_{m+k,n+l} \cdot x_{kl}$$

This allows us to retain the 2D structure of each input image. The convolution maps the image to another 2D grid of units (the next layer) as depicted in Figure 4. Convolution is often successful in tasks involving images for numerous reasons such as lowering the number of parameters (which facilitates training), decreasing the representational burden on each unit, taking advantage of the local invariance properties found in most image classification tasks, etc.

Just using a neural network, even a CNN, was not enough for our task. Perhaps one of the biggest drawbacks of neural networks is the sheer volume of data they need to perform acceptably. For this reason researchers have focused much effort into leveraging parameters learned from other classifications tasks with massive amounts of data (e.g classifying images of cats and dogs). This use of pretrained networks is referred to as *transfer* learning. Essentially, we take a network with already-fit parameters, and then fit some subset of those to our own data.

We used transfer learning on every network we trained for this task, except for a simple baseline CNN that we trained from top to bottom as a point of comparison.

Another method we implemented to help cope with the deficiency of data is known as *data augmentation*. This is the naive, but effective, idea that we can take advantage of symmetries we may see in our data to gain a two pronged advantage: we effectively increase the sample size and we guard against over fitting. Essentially, data augmentation is the creation of data by taking data we have and transforming it in ways that might have appeared if we gather a new data sample. For example, in our domain of mass classification, reflecting a mammogram horizontally is reasonably something we may have seen "in the wild" and so we might use both the reflection and the original to train the network. We augmented with horizontal and vertical flips, as well as small-scale image zooms.

With all of this said, we have the machinery necessary to present our models and results. The first model we implemented and trained on was a simple three-layer convolutional network with two fully-connected layers at the end for classification. We trained this model only on our data as a point of comparison for all of the techniques and massive networks we implemented afterwards. After feeding this baseline the training data 180 times, it achieved ≈ 65% validation accuracy.

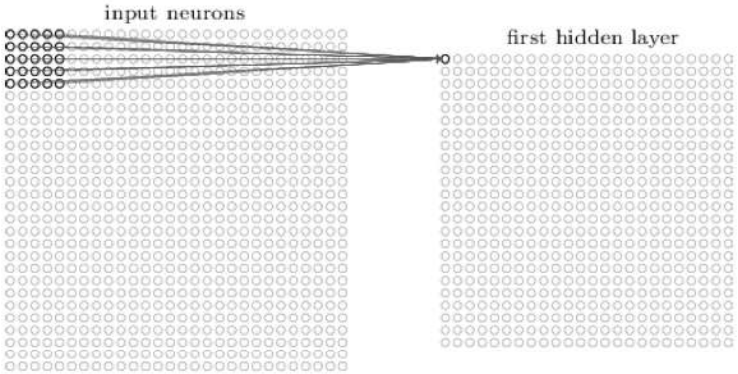

Figure 4: Two layers of a CNN

Next we used the VGG16 network. This architecture was developed by the Visual Geometry Group at Oxford and performed exceptionally well, given its simplicity, in the famous ImageNet 2014 competition, where it was trained on millions of images curated from the internet. As its name implies, the network consists of 16 layers. After 180 passes of the training data it achieved ≈ 72% validation accuracy.

Finally we used the third version of GoogLeNet. The first iteration of this network won multiple categories of the ImageNet 2014 competition (the same year as VGG16).

It became (in)famous for its clever "inception" modules which act like layers but are each more complicated than our baseline model. This performed with ≈ 78% accuracy after being fed the training data 180 times.

5 Modeling Tumor Growth and Treatment

For this portion of the project we replicated a 2014 paper written by López, Seoane and Sanjuán, called *A Validated Mathematical Model of Tumor Growth Including Tumor-Host Interaction, Cell-Mediated Immune Response and Chemotherapy*. There, the authors consider ODEs from the literature that model interactions among three cell populations: tumor, immune, and host cells. In particular, they non-dimensionalize these equations and focus on tumor-host and tumor-effector interactions since host and immune cell competition is negligible. The paper concludes by incorporating chemotherapy into the model and evaluates the effects on the ODEs using mouse data from Hiramoto and Ghanta (1974).

To understand this system, some intuition of the underlying physical processes is necessary; we begin with the tumor-host interactions. When tumor cells form, they interact with healthy host cell populations and compete for oxygen in the blood stream. Tumor cells multiply at a faster rate, overwhelming and eventually killing the host's cells. The body recognizes the problem and sends two types of cells to fight off the cancerous ones — natural killers and CD8+ T-lymphocytes — but for small time intervals the two populations can be expressed as a linear combination of one another, and so we refer to them communally as effector cells. Effector cells directly attack the tumor cells and reduce their population. This introduces our second point of comparison: tumor-effector interactions.

Due to discrepancies in units of measure, we non-dimensionalized the ODE. The equations representing the growth rates of the tumor ($\dot{x}$), host cells ($\dot{y}$), and effector cells ($\dot{z}$) then become

$$\dot{x} = x(1-x) - a_{12}yx - D(x,z)x$$
$$\dot{y} = r_2 y(1-y) - a_{21}xy$$
$$\dot{z} = 1 - d_3 z + g\frac{D^2(x,z)x^2}{h + D^2(x,z)x^2}z - a_{31}xz$$

where $D(x,z) = d\frac{f^\lambda z^\lambda}{sx^\lambda + f^\lambda z^\lambda}$.

It is also worth noting that, in the original ODEs, the host and tumor cells are assumed to follow a logistic growth.

Now that we had the ODEs in a form we could work with, we began by trying to understand the qualitative behavior of the system. We plotted nullclines and found the fixed points of this system of equations using given parameter values. We had three saddle points at (0.06, 0, 6.55), (0.1, 0.74, 3.02) and (0, 0, 8.93), and two stable fixed points at (0.65, 0, 0.31) and (0, 1, 8.39). The saddle points are physically uninteresting, but note that the two stable points correspond to total death of the host population and total death of the tumor cell population respectively. After plotting this using the same data as the authors, we could see that all of this matched their results.

With this information gained, our next goal was to evaluate it against experimental results and then incorporate chemotherapy into the model. We used data from Hiramoto and Ghanta (1974) that resulted from experiments where they injected mice with tumor cells and recorded the populations of host, effector, and tumor cells over a period of 36 days. Unfortunately the experimenters could not differentiate between host and effector cells when recording the data points, so these cell counts are combined and reported as "healthy" cells. Furthermore, the body needed 10 days before it noticed the tumor cells and began mounting a resistance, while the tumor cells needed about 15 days to begin growing consistently, so the populations were recorded on day 10 and then every 3 days beginning on the 18th day. After the 21st day, the mice began chemotherapy treatment. For simplicity we did not model the effect of the treatment on healthy host cells and instead focused on the tumor cells. The new growth rate of tumor cells is then

$$\dot{x} = x(1-x) - a_{12}yx - D(x,z)x - (1 - e^{-pu(t-\tau)_+})x$$

where τ is the point in time in which the treatment begins to take affect and $u(\theta) = u_0 e^{-\tilde{k}\theta}$ throttles the degree to which the treatment affects the tumor cell population after τ.

Using this data and least squares fitting we solved for the parameters *g*, *d*, and *s* above. Plotting our equations using the parameters we found resulted in Figure 5. We also plotted the new equation for the tumor cells after the chemotherapy had been applied; we see the results in Figure 6.

6 Conclusion

If the group had more time together, there are a few parts of our pipeline that could be improved. The detection and diagnosis stages would benefit from a newer, fully-digital dataset. These are just now becoming available, and will

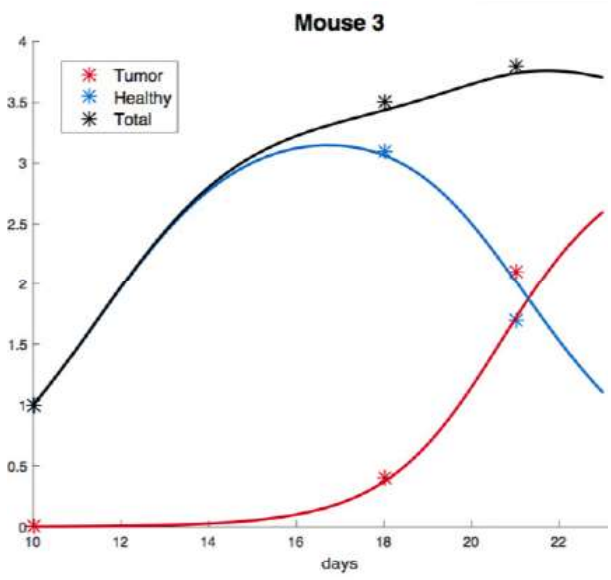

Figure 5: Growth of Cells in a Mouse

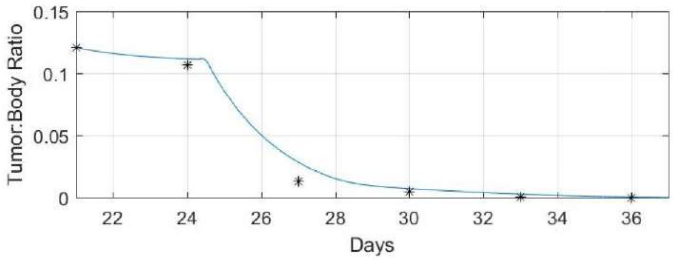

Figure. 6: Decay of Tumor Cells in Mouse after Chemotherapy

likely push the field forward. With larger high-resolution data in hand, we would be able to experiment with different architectures, perhaps eschewing transfer learning entirely. The treatment model would also benefit from newer data to compare with our results. If human data becomes available, this would be ideal. Furthermore, an important aspect of chemotherapy is its negative effect on healthy cells, so incorporating this into our models would be beneficial.

Unparalelled mathematician Maryam Mirzakhani, former Professor at Stanford University, succumbed to breast cancer in July 2017. Mirzakhani was the first woman to receive the Fields Medal in mathematics, earning the prestigious award for her contributions in the fields of Teichmüller dynamics, hyperbolic geometry, and ergodic theory.

Mirzakhani photo credit: Stanford University

STATISTICAL CONSULTING AND COLLABORATION SERVICES (SCCS)

Statistics is integral to the research work of nearly every scientific discipline. Historically, our department offered statistical consulting services to support other researchers. By 2003, administrative issues resulted in the end to these services. But in 2011, then-Head **Michael Lavine** revived statistical consulting in the Department as the new SCCS. Being a capable and curious statistician, although lacking external funding, he found many projects – both *pro bono* and fee-based – through our SCCS website and by word-of-mouth, and executed them either on his own or in collaboration with a small number of graduate students.

Graduate Student Consultants Lin Cong, Janelle Fredericks, and Jon Moyer lending (and building) their expertise.

By the summer of 2016, Professor **Krista Gile** joined the leadership of the SCCS in the interest of broadening the participation of students and increasing the volume of projects that could be supported. In the 2016-2017 academic year, Gile also offered a 1-credit graduate course on Statistical Consulting. Since then, Professors Gile and Lavine, along with over two dozen student consultants, have assisted with at least two dozen projects, whose clients include UMass student groups, UMass faculty and graduate students, as well as researchers and professionals outside the university. These diverse clients include researchers interested in reducing the epidemic of prescription opioid use, understanding mathematical patterns in music, and studying the relationship between exercise and academic performance.

In addition to supporting the work of these clients, the SCCS gives graduate (and a few undergraduate) students an opportunity to experience the wide variety of applications of statistics. Seeing real data, while working on a project that a client cares about and communicating with that client, offers our students great practical training for the data science careers of the future.