Identifying Melanoma using Computer Vision and an Artificial Neural Network: A Senior Thesis in Computer Science

Hannah Rivers

September 2011 – April 2012

Abstract

Proident craft beer whatever thundercats officia deserunt, chambray PBR. Do tumblr proident deserunt, biodiesel jean shorts lomo officia quis dolore hoodie. Craft beer mixtape incididunt laborum, duis sapiente est photo booth et irony. Portland american apparel fap tattooed art party. Freegan ea scenester cillum est, iphone skateboard ullamco keffiyeh. Bicycle rights culpa whatever yr. Yr quis dolore craft beer elit.

Contents

1 Thesis statement

With the proper image acquisition and preprocessing techniques and selection of classifiers, an artificial neural network can be created and trained to distinguish between malignant melanoma, dysplastic nevi, and common nevi.

2 Introduction to the problem

2.1 Background information on human classification of melanoma

1. What are the steps that go into classifying a melanoma? [11]

    (a) The most commonly accepted heuristic is the ABCDE rule. Meeting one or more of these criteria may indicate the presence of a melanoma.

    - i. Asymmetry; the shape of one half does not match the other
    - ii. Border; the edges are ragged, blurred, or irregular
    - iii. Color; the color is uneven and may include shades of black, brown, or tan
    - iv. Diameter; it's larger than a pencil eraser
    - v. Evolution; it changes in size, color spreads to surrounding skin, a new bump or nodule appears, there's any pain, itchiness, bleeding, oozing, or general irritation

    (b) Doctors recommend monthly self-examinations to look for skin lesions fitting the ABCDE rule.

    - i. Familiarize yourself with your moles so you can notice any changes. Use a hand-held mirror to inspect hard-to-see areas. Don't forget your scalp, groin, fingernails, soles, and the area between your toes.

    (c) If you find a suspicious mole, consult your dermatologist or physician. If they feel that it's possibly a melanoma, they will likely biopsy it and send it to a pathologist who will determine whether or not it's cancerous.

    - i. There are 3 types of biopsies:
        - A. Punch biopsy: A tool with a circular blade is used to remove a round piece of skin containing the suspicious mole.
        - B. Excisional biopsy: The entire mole is surgically removed.
        - C. Incisional biopsy: Only the most irregular part of a mole or growth is removed for laboratory analysis.

    (d) If you are diagnosed with melanoma, the next step is to determine the severity of the condition.

        i. Melanoma is staged between I and IV, where a stage I melanoma is small with a very successful treatment rate and a stage IV melanoma has spread to other organs and recovery is highly unlikely.

        ii. Stages are assigned by:

            A. Determining the thickness through examination under a microscope. In general, the thicker the tumor, the more serious the disease.

            B. Determining whether the melanoma has spread to nearby lymph nodes through a sentinel node biopsy. In this procedure, dye is injected into the area where the melanoma was removed and flows to lymph nodes. The first ones to collect dye are removed and tested for cancer cells. If cancer cells are not present in the closest lymph nodes, chances are that the melanoma has not spread.

2. How generally accurate are doctors?

    (a) The differentiation of early melanoma from other pigmented skin lesions is not trivial even for experienced dermatologists; in several cases, primary care physicians seem to underestimate melanoma in its early stage. [1]

    (b) However, a larger issue is the patient failing to notice and consult a doctor in time.

    (c) Usually, if the patient brings the lesion to the doctor's attention early, melanoma can be identified and treated.

2.2 Why it is an important problem

1. Skin cancer is the most common form of cancer in the United States, and melanoma is the most serious type of skin cancer. [2]

2. The incidence of both non-melanoma and melanoma skin cancers has been increasing over the past decades. Currently, between two and three million non-melanoma skin cancers and 132,000 melanoma skin cancers occur globally each year. One in every three cancers diagnosed is a skin cancer and, according to Skin Cancer Foundation Statistics, one in every five Americans will develop skin cancer in their lifetime. [3]

3. Melanoma can be cured if it is diagnosed and treated early. If melanoma is not removed in its early stages, cancer cells may grow downward from the skin surface and invade healthy tissue. If it spreads to other parts of the body it can be difficult to control. [2]

4. Dermatologists and doctors are expensive, and many cannot afford to consult with them at every suspicion. They are also not always accurate.

5. In general, the trend of modern medicine is creating machines to do tasks humans used to, with greater accuracy in less time and for less capital. This application is the first step down that road.

6. Disclaimer: The intent of this application is to serve as a guide to indicate the likelihood of melanoma. It is a primary recognition tool; it is not meant to be used as a diagnostic tool or as a replacement for a doctor.

2.3 Past approaches taken by machine learning

1. Overview of machine learning approach

   (a) First, the machine must be able to acquire pertinent data from images.

      i. The selection of what constitutes pertinent data as well as the method of acquisition varies.

   (b) Next, the data must be processed in such a way that analysis of results yields a classification of either melanoma or dysplastic nevus.

      i. The method of processing and analysis varies.

      ii. In machine learning, the processing is done through a series of adaptive algorithms that allow the computer to evolve its behaviors based on relationships and patterns in the data. This allows more intelligent decision making than a static algorithm.

2. Main design components [1]

   (a) Image acquisition

      i. What type of images are going to be input by users? These are the type of images you want to train on.

      ii. Where can you get this image set?

   (b) Feature selection/extraction

      i. Success of image recognition depends on correct selection of features for classification.

      ii. They must be measurable, with high sensitivity (high correlation and probability of positive response) and high specificity (high probability of a true negative response).

      iii. Different diagnostic methods (ABCDE rule, pattern analysis, Menzies method, 7-point checklist, texture analysis) are outlined under selection of classifiers.

      iv. Once you have selected these features, you need to figure out how to extract them from the image.

   (c) Image processing/analysis

i. Now that you have extracted the key features, you must determine the method through which to process them. Each of the diagnostic methods mentioned above include a set of features to extract, and rules through which to process them from data to information.

ii. It's a typical optimization problem, resolved with heuristic strategies, greedy or genetic algorithms, other computational intelligence methods, or strategies from statistical pattern recognition.

iii. In my case, I'm choosing to resolve it with an ANN.

(d) Classification methodology

i. You must now take the information generated above and decide for which thresholds it should be classified into different categories, as well as which categories to use.

3. Image acquisition techniques [1]

(a) High budget

i. Ideally, you may use epiluminence microscopy (ELM), which is the examination of skin lesions with a dermatoscope. This process consists of a magnifier, typically x10, a non-polarized light source, a transparent plate, and a liquid medium between the instrument and the skin, which allows detailed inspection of skin lesions unobstructed by reflections on the skin's surface. It is used to render the epidermis translucent, making the dermal features clearly visible.

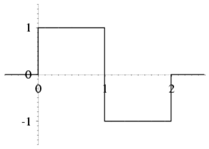
ii. Transmission electron microscopy (TEM) is another microscopy technique that reveals the structure of elastic networks in the dermis. A beam of electrons is transmitted through an ultra-thin specimen, which it interacts with as it passes through. From this interaction an image is formed; that image is magnified and focused onto a fluorescent screen on a layer of photographic film.

iii. Computed tomography (CT) scans use digital geometry processing to generate a three-dimensional image of the inside of an object from a large series of two-dimensional X-ray images taken around a single axis of rotation.

iv. Positron emission tomography (PET) scans employing fluorodeoxyglucose (FDG) are a nuclear medicine imaging technique that produces an image of functional processes in the body. Gamma ray pairs are emitted indirectly by a positron-emitting radionuclide (tracer) and detected by the system, then introduced into the body on a biologically active molecule. Computer analysis of tracer concentrations within the body are used to construct a three-dimensional image.

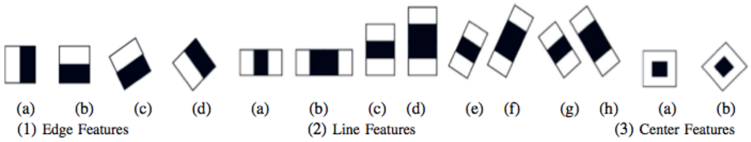
    - v. Multi-frequency electrical impedance tomography can create an image of a lesion by recording and analyzing the resistance with which electric waves of different frequencies move through it.
  - (b) Low budget
    - i. High-resolution, low-distortion cameras are the most readily available, however they don't account for color constancy, and it's difficult to get a large, centered, focused, and detailed image of the lesion.
    - ii. Instead, video cameras are commonly used that can be controlled and parameterized online in real time for a more three-dimensional and detailed image. Another benefit of viewing the lesion from different angles is that it's easier to account for color constancy.

4. Boosted Haar features [4]
  - (a) Haar-like features are digital image features used in the field of object recognition, named for their intuitive similarity to the Haar wavelet.
    - i. Wavelets are sets of non-linear bases used to project a function. The Haar wavelet is the simplest possible wavelet, which can be transformed into any other wave.



  - (b) A Haar-like feature categorizes an image into subsections based on the difference between summed pixel intensities in adjacent rectangular regions at a specific location in a detection window.
    - i. For instance, in the facial recognition problem, Haar wavelets are used to distinguish eyes, which are commonly shadowed, from cheeks, which are commonly the lightest part of the face.
  - (c) The key advantage of a Haar-like feature over most other features is its calculation speed. If a summed area table (also known as an integrated image) is used, a Haar-like feature of any size can be calculated in constant time.
    - i. A summed area table is a quick and efficient algorithm to sum values in a rectangular subset of a grid.
  - (d) As shown in the paper:
    - i. A 2D decomposition of the image with n^2 pixels yields n^2 wavelet coefficients corresponding to a distinct Haar wavelet.

ii. Looks at 4 edge features, 8 line, and 2 center-surround features.



(e) Different algorithms utilize Haar-like features, such as the Viola-Jones object detection framework.

i. The Viola-Jones object detection framework, proposed in 2001 for the facial recognition problem, was the first to provide competitive object detection rates in real time.

ii. It is implemented in OpenCV as cvHaarDetectObjects().

iii. In the detection phase, a target sized window is passed over the input image, and for each subsection of the image the pixel intensities are summed, thus calculating the Haar-like feature. The difference is then compared against a learned threshold that distinguishes objects from non-objects.

iv. Due to its simplicity, a Haar-like feature is only a weak learner or classifier; its detection quality is only slightly better than random guessing. Therefore either a large number of Haar-like features or a boosting algorithm to strengthen the best ones are necessary to describe an object with sufficient accuracy.

5. Boosting Algorithms

(a) Boosting is a machine learning meta-algorithm used with supervised learning to turn a set of weak learners into a single strong learner.

i. Meta-algorithms are iterative optimization algorithms that try to improve a candidate solution (a member of the set of all feasible solutions to the given problem), with regard to a given measure of quality.

ii. Weak learners are classifiers that perform slightly better than random guessing, while strong learners are strongly correlated with the true classification.

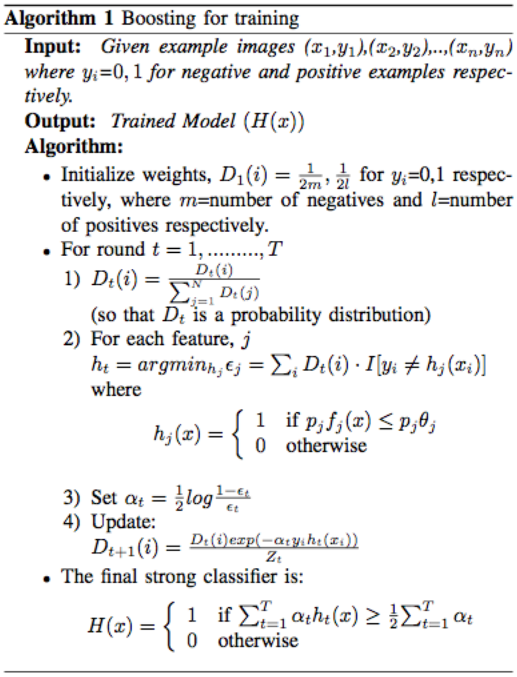
iii. In the paper [4], Haar-like features are used in conjunction with the AdaBoost boosting algorithm to increase their detection power.

A. AdaBoost stands for adaptive boosting.

B. The AdaBoost algorithm predisposes subsequent classifiers built in favor of instances misclassified by previous classifiers. In each round, it generates and calls a new weak classifier, updating the distribution of weights to indicate the importance of examples in the data set for the classification by increasing the weights of incorrectly classified examples and decreasing that of correctly classified ones so that the new classifier focuses on the former.

C. In the paper, each image category is trained separately, then the weights and weak classifiers are stored.

D. Next, the AdaBoost algorithm takes several weak classifiers given by Haar-like features and develops them into stronger models after a number of iterations, then the highly selective features that minimize the classification error are extracted.



Algorithm 1 Boosting for training

```
Input: Given example images $(x_1,y_1),(x_2,y_2),..,(x_n,y_n)$ where $y_i=0,1$ for negative and positive examples respectively.
Output: Trained Model $(H(x))$
Algorithm:
  • Initialize weights, $D_1(i) = \frac{1}{2m}, \frac{1}{2l}$ for $y_i$=0,1 respectively, where $m$=number of negatives and $l$=number of positives respectively.
  • For round $t = 1, ........., T$
    1) $D_t(i) = \frac{D_t(i)}{\sum_{j=1}^{N} D_t(j)}$
       (so that $D_t$ is a probability distribution)
    2) For each feature, $j$
       $h_t = argmin_{h_j} \epsilon_j = \sum_i D_t(i) \cdot I[y_i \neq h_j(x_i)]$
       where
       $h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) \leq p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$
    3) Set $\alpha_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t}$
    4) Update:
       $D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
  • The final strong classifier is:
       $H(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$
```

iv. The AdaBoost algorithm was the first successful boosting algorithm. Since then, LPBoost, TotalBoost, BrownBoost, MadaBoost, LogitBoost, and more have had success as well.

A. Some of these have built-in implementations in OpenCV.

6. Gabor Filter

(a) A Gabor filter is a linear filter used for edge detection in image processing.

i. The filter has a real and an imaginary component representing orthogonal directions.

(b) Like Haar wavelets, the Gabor filters are self-similar; all filters can be generated by dilation and rotation of one basic wavelet

(c) Gabor filters are similar to the human visual system in their representations of frequency and orientation.

i. In fact, simple cells in mammalian visual cortexes can be modeled by Gabor functions.

   (d) Their strength lies in texture representation and discrimination rather than object recognition.
      i. This is useful for the problem of skin lesion classification, as abnormal texture can be an indication of melanoma.
   (e) Open-source libraries for Gabor wavelet feature extractions exist, however all seem to be targeted at facial recognition.

3 Theory of neural computation

3.1 An introduction to neurocomputing [5]

1. Things we need to know to understand neurocomputing:
   (a) Enough neuroscience to understand why the models make certain approximations, and in which fields these approximations are more and less accurate.
   (b) Enough fundamental math and coding to understand the systems used.
   (c) Enough cognitive science to have some idea about what the brain is supposed to do, and how it does it. Our brains are not all-purpose computers, they are powerful at some tasks and weak at others. Their power comes from:
      i. enormous relative size
      ii. effective biological preprocessing
      iii. use of memory in place of computing power
      iv. efficiency with small number of operations
   (d) Finally, for all of this, we must assume that it's possible to make meaningful simplifications of some aspects of the nervous system. Some would argue that neurobiology is intrinsically too complicated to simplify, or that we don't know enough about its workings to make correct generalizations, which may be actually true.
2. What is neurocomputing?
   (a) It's most basically defined as brain-like computation; while there are many ways to organize a computing system, neurocomputing is an attempt to build computers that are designed like the brain, in hopes of emulating it.
      i. Brains have strengths and weaknesses, just like any other approach to computing.
      ii. They're good at pattern recognition, motor control, perception, flexible inference, intuition, and good guessing, however they are slow, imprecise, make erroneous generalizations, are prejudiced, and are usually incapable of explaining their actions. All of these properties, desirable or not, may be typical of neurocomputing.

(b) Biology is the practical science of what you can do with the resources available to you, not the theoretical ideal.

    i. Because of this, biological neural nets are a wealth of information in the practical engineering and economic sense as well as the computational sense. They provide an example of optimizing over available resources in a manner desirable to emulate.

3.2 Biological Neural Networks [5]

1. Biological neural networks are populations of interconnected neurons whose inputs or signalling targets define a recognizable circuit. They are the structure through which our brains process informtion.

   (a) Our brains have ~100 billion neurons.

   (b) Neurons are made up on the input end of dendrites, which branch out in a tree-like manner from the cell body, also known as the soma. Extending from the cell body is a long, thin projection known as the axon, the transmission line of the neuron. The axon can give rise to collateral branches, forming a vast, interconnected network. When axons reach their final destination, they branch again into the structure known as the terminal arborization. At the ends of the terminal arborization are synapses, comprising the output end of the neuron.

   (c) Dendrites receive inputs from other cells, the soma processes the inputs then transmits information along the axon to the synapses, whose outputs are received by other neurons via neurotransmitters diffused across the synaptic cleft.

      i. Each neuron is connected to thousands of other neurons which communicate through electrochemical signals.

      ii. Each neuron continuously receives signals from these other neurons and then sums up the inputs according to some process.

      iii. If the end result is greater than some threshold value, the neuron fires by generating a voltage, known as an action potentional, that transmits down the axon to the terminal arborization. This response is an all-or-nothing binary; there either is action potential or there is not.

      iv. After firing, a neuron has both an absolute and a relative refractory period during which it cannot fire again. For the former, there is no action threshold nor subsequent possibility of firing, but for the latter, the threshold is merely elevated. This relative refractory period, also known as synaptic resistance, is adaptable, which can cause modified or “learned” behavior of the neuron in which firing frequency will drop if a stimulus is maintained. These varying, modifiable resistances cause detailed interactions among the web of neurons that are the key to the nature of computation that neural networks perform.

   A. Hebb's Learning Rule: If the input of a neuron is repeatedly and persistently causing the neuron to fire, a metabolic change happens in the synapse of that particular input to reduce its resistance.

2. In the brain, functions are performed collectively and in parallel rather than there being a clear delineation of subtasks to which various neurons are assigned. This is a fundamental distinguishing property of artificial neural nets.

3. Biological ground rules:

   (a) No new neurons can be created.

   (b) Neurons must earn their existence or die. They are metabolically expensive, and thus biological pressure dictates that as few as possible must be used.

   (c) We have roughly 100 billion neurons at start.

3.3 Artificial Neural Networks [5]

1. It's a mathematical or computational model that either abstracts or is inspired by the structure and/or functional aspects of biological neural networks.

   (a) Neurons are represented as nodes, and synapses as weighted connections between nodes.

   (b) Nodes recieve input, apply some sort of summation function, then output according to the input's comparison against a threshold.

   (c) The nodes and connections form a network that learns by modifying the weights of the connections so that eventually a certain input yields a certain output; the network can perform some function.

2. It's a network of simple processing elements (artificial neurons/nodes) that exhibits complex global behavior determined by adaptive weighted connections between processing elements and element parameters. The modifying algorithm changes the structure of the network based on external or internal information during the learning phase to produce a desired signal flow.

3. They are usually used to model complex relationships between inputs and outputs or to find patterns in data.

4. There are some ways in which the biological model can be improved by technology after using it as a template, since computers are not constrained to biological pressures

   (a) For instance, we are capable of creating and maintaining as many neurons as we need (however most successful ANNs to date use on the order of hundreds or thousands of nodes rather than 100 billion).
   (b) Computers are also never innacurate or fatigued, and can perform operations very fast. This does not mean that an artificial neural network can't be wrong or slow, but that each individual step comprising the process won't be.

5. In modern software implementations of artificial neural networks, the approach inspired by biology has been largely abandoned for a more practical approach based on statistics and signal processing. Instead, components of neural networks are used as part of a larger system with both adaptive and non-adaptive elements.

3.4 Types of problems artificial neural nets are best suited for

1. "No Free Lunch" theorem [6]

   (a) The "No Free Lunch" theorem states that all optimization algorithms are equivalent when their performance is averaged across the entire possible problem set.
   (b) Since we know that not all optimization algorithms perform equally well on every problem, we can infer that each optimization algorithm performs better on some problems than on others.
   (c) There can be no globally optimal algorithm for solving optimization problems because on average, performance comes out to be the same.
   (d) Therefore, each algorithm has strengths and weaknesses, and individual problems have algorithms that will work better on them than others.

2. Practical applications for artificial neural networks have been slow in presenting themselves. In computers, hardware largely determines the software that runs efficiently on it. While a computer is always a programmable electronic device that can perform binary arithmetic and logic functions, much more goes into our modern devices. Computers able to adapt and learn have only been in existence for the last decade or so, since they must be able to handle behavioral change, changes in [figurative] internal wiring, and potential instability.

   (a) This problem can be thought of in the same way as the fact that we and chimpanzees differ. While 99% of our DNA is the same, the 1% difference is predominantly in structural genes which modify sizes and shapes of structures. While we contain no radical new hardware over chimps, rearrangements, inflations, contractions, and other modifications of the same physical structure has made our brains capable of considerably more computational power and flexibility.

3. It is important not to lose sight of the evolutionary perspective when considering neural networks.

   (a) Neural networks, like brains, cannot compute everything, but only arrive at reasonable solutions for a small but useful set of problems.

   (b) The practical applications of neural nets are the same problems selective pressure has caused animals to biologically adapt to solve: association, classification, categorization, generalization, rapid response, and quick inference from inadequate data.

   (c) A brain-like computing device is not a wonder computer capable of solving all problems. Its performance is highly problem and detail specific.

4. The utility of the artificial neural network models lies in the fact that they can be used to infer and use a function from a set of observations, particularly in applications where the complexity of the data or task makes the design of such functions by hand impractical.

5. The most successful applications of neural nets to date are:

   (a) function approximation, or regression analysis, including time series prediction and modeling

   (b) classification, including pattern and sequence recognition, novelty detection and sequential decision making

   (c) data processing, including filtering, clustering, blind signal separation and compression

3.5 Why an ANN is a good fit for this problem

1. Medical diagnosis falls under the theme of classification through pattern recognition.

4 Types of Neural Networks

4.1 Necessary concepts

1. To understand the difference between various types of neural nets, you first must be familiar with several concepts.

2. Supervised v. unsupervised learning

   (a) For all neural nets, a training set of data is necessary to teach the network the necessary connections for desired flow.

i. Usually, we know the appropriate output for the given data set, and can use that information to calculate then minimize the discrepancy between the actual and ideal output. Using that knowledge to appropriately modify and train the network is called supervised learning.

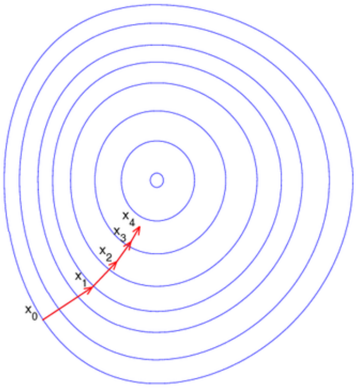
ii. Sometimes, we don't know the appropriate output, and instead need the system to take the input data and somehow organize it in an appropriate way. Methods to suit these needs are much more difficult to construct and use, however can be very rewarding. These methods utilize unsupervised learning.

3. Gradient descent

   (a) It's a first-order optimization algorithm.

   (b) To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (approximate or exact) of the function at the current point.

      i. Notice that gradient descent does not necessarily converge to the absolute minimum. The minimum it converges on is based on the starting values.



   (c) Gradient descent is based on the observation that if the multivariable function $F(\mathbf{x})$ is defined and differentiable in a neighborhood of a point a, then F(x) decreases fastest if one moves from a in the direction of the negative gradient of F at a, $-\nabla F(\mathbf{a})$.

   (d) It follows that, if $\mathbf{b} = \mathbf{a} - \gamma \nabla F(\mathbf{a})$ for step size $\gamma > 0$ a small enough number, then $F(\mathbf{a}) \geq F(\mathbf{b})$. With this observation in mind, one starts with a guess $\mathbf{x}_0$ for a local minimum of F, and considers the sequence $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots$ such that $\mathbf{x}_{n+1} = \mathbf{x}_n - \gamma_n \nabla F(\mathbf{x}_n),\ n \geq 0$ (note that the value of the step size γ is allowed to change at every iteration).

   (e) We have $F(\mathbf{x}_0) \geq F(\mathbf{x}_1) \geq F(\mathbf{x}_2) \geq \cdots$, so the sequence $(\mathbf{x}_n)$ converges to a local minimum.

4. Bayesian probability [12]

   (a) In general, probability is viewed in the *classical* or *frequentist* interpretation, where it is defined in terms of the frequencies of random, repeatable events. However, in *Bayesian* probability, probability provides a quantification of uncertainty, or a measure of a state of knowledge. It's an extension of logic that enables reasoning with uncertain statements. To evaluate the probability of a hypothesis, the person specifies some prior probability which is updated in light of new relevant data.

      i. For instance, we speak of the probability of events such as the earth being destroyed or a person dying. These are not repeatable events from which we can determine a probability through frequencies. However, they still have a likelihood of occurrence that we can educatedly guess at through consideration of relevant evidence, such as the number of asteroids that have come close or the person's consumption of carcinogens. When we get new evidence, we can reassess our hypotheses. Bayesian probability allows us to quantify these expressions of uncertainty.

      ii. In 1946, Cox showed that if we numerically quantify degrees of belief, then a simple set of axioms encoding common sense properties of such beliefs leads uniquely to a set of rules for manipulating degrees of belief that are equivalent to the sum and product rules of classical probability. (paraphrase) In 2003, Jaynes used this to prove that probability theory could be regarded as an extension of Boolean logic to situations involving uncertainty. Over the years, many mathematicians have proposed various sets of axioms that all have behaved precisely according to the rules of probability.

   (b) Bayes' theorem:

      i. $p(w|D) = \frac{p(D|w)p(w)}{p(D)}$

      ii. D is the set of observed data, w is the event whose uncertainty we wish to numerically express.

      iii. It expresses the uncertainty of the event w depending on the set of observed data D as the product of the uncertainty of the evidence as a probability distribution over w times the prior probability distribution of w, scaled to make sure it integrates to one and is therefore a valid probability density.

      iv. i.e. The posterior probability is proportional to the likelihood times the prior probability distribution, where all of these quantities are viewed as functions of w.

   (c) An advantage Bayesian probability holds over classical probability is that the inclusion of prior knowledge arises naturally. For instance, Bayesian probability won't assign a fair coin a 100% probability of landing on tails if it does so three times of three. However, the likelihood it assigns is highly dependent on the prior distribution used. Bayesian methods based on poor choices of prior distributions can yield poor results with high confidence.

  (d) Bayesian principles are largely used in pattern recognition and other machine learning methods, as we can incorporate evidence as we progress.

4.2 The Perceptron [5]

1. The perceptron was the first influential model of a simple artificial neural network, proposed by psychologist Frank Rosenblatt in 1958.

2. It is the simplest kind of feedforward neural network: it functions as a linear classifier.

   (a) Feed-forward means that there are no cycles or loops, the connections only move forward.

3. Although its learning capabilities are extremely limited, it proved to be a valuable tool in prediction and explanation of human cognition. Interestingly, its limitations that make it less effective as a computing device are very similar to those of humans, and as such it was widely used among psychologists and cognitive scientists to help form many theories about the workings of the brain

4. It has an input sensory layer called a retina, which is partially connected to an association layer. It is important theoretically that these layers are not fully connected; this means that each unit in the association layer is computing a different function of the image on the retina. The association layer is then reciprocally connected to a response layer by way of unidirectionally modifiable weighted "synapses" (in the direction of the response layer).

5. The goal of the perceptron is the activation of a single appropriate unit in the response layer for a certain set of inputs. The activation of any appropriate unit is known as a candidate solution, or a member of the set of possible solutions for a given problem.

6. The basic computing element of the perceptron is the *threshold logic unit*, or TLU, comprised of n input weights with strengths $w[i]$. It sums the product of the input weights and their respective strengths and nonlinearly transforms them into binary values indicating activation if the scaled summed input is over a certain threshold.

   (a) The activation function is binary; either a correct unit is activated or it is not. There is no graded activation.

7. The perceptron functions as a pattern classifier. It is solely a linear discriminant model and works only when two patterns are linearly separable (a hyperplane can be drawn between them). The perceptron convergence theorem states that if there exists an exact solution, the perceptron learning algorithm is guaranteed to learn the classification it in a finite number of steps. If however, there is not an exact solution and the training data is not linearly separable, the algorithm will run indefinitely without converging.

   (a) If necessary, include proof of the convergence of the perceptron learning algorithm (p. 220 in Anderson Intro to NN).

   (b) The perceptron learning algorithm uses knowledge of past results to modify the weights of the connections, thereby improving the performance of the network.

   (c) However, many interesting classifications are not linearly separable and require a more complex decision surface to separate the classes.

8. Along with the ADALINE (an early gradient descent algorithm using a non-binary error function), the perceptron framed the network learning problem in a way that was fundamentally accepted and unconsciously integrated into scientific thought, shaping the evolution of computational science and future network models for years.

4.3 Multi-Layer Perceptron (MLP)

1. Multilayer perceptrons are comprised of multiple stages of processing, each of which individually resembles the perceptron model. The difference lies in the intermediate units' use of continuous sigmoidal nonlinearities instead of step functions. This type of processing means that the neural network function is differentiable with respect to the network parameters, a fact which plays a central role in network training.

   (a) If the hidden units used linear activation functions, the network could be simplified to an input, association, and output layer, i.e. a perceptron (because the composition of successive linear transformations is itself a linear transformation).

   (b) The intermediate units are referred to as "hidden" because their function is obscured from sight.

2. A trademark of multilayer perceptrons is that they do not have a connection to biological neural networks like other types of ANNs.

3. They are defined as a feed-forward artificial neural network model consisting of multiple fully connected layers of nodes in a directed graph that maps sets of input data onto a set of appropriate output.

   (a) There's an input layer, an output layer, and at least one hidden layer.

        i. Think of it as a set of nested functions.

   (b) Fully connected means each node in one layer connects with a weight w$_{ij}$ to each node in the subsequent layer.

4. Each node of an MLP has a non-linear (sigmoidal) activation function meant to model the frequency of action potentials of biological neurons instead of the simplified binary function.

   (a) The two most common activation functions are the hyperbolic tangent function $\Phi(y_i) = \tanh(v_i)$ which ranges from $(-1, 1)$, and the logistic function $\Phi(y_i) = (1+e^{-v_i})^{-1}$ which ranges from $(0, 1)$, where y_i is the output of the i^{th} node and v_i is the weighted sum of the input synapses.

5. MLPs use backpropogation, a supervised learning technique, to train the network.

   (a) Backpropogation is a generalization of the least mean squares algorithm used by the linear perceptron. It's an efficient technique for evaluating the gradient of an error function for a feed-forward neural net, which is the first step in most training algorithms.

   (b) It's a local "message passing" scheme in which information is sent both forwards and backwards through the network.

   (c) It depends on the activation function being differentiable, a key feature of MLPs.

   (d) Learning in the network occurs by modifying the connection weights by a factor calculated through gradient descent after a piece of data is processed, so that the change is based on the amount of error generated compared to the expected result.

   (e) The first phase of training is propagation.

        i. The training data is propagated forward through the network, generating output activations by applying the selected sigmoidal function to the weighted sum of the inputs.

            A. The input vector x_n is applied to the network and propagated forward using $a_j = \sum_i w_{ji} z_i$ and $z_j = h(a_j)$ where z is the activation, w is the weight, h is the nonlinear activation function, and a is the weighted sum of the inputs.

            B. The error d_k is then calculated for all the output units using $d_k = y_k - t_k$ where y is the output (a linear combination of the input variables), and t is a value specified by the error function.

        ii. Then backward propagation of the output activations through the ANN occurs using the target output vector as a starting point.

      A. The algorithm backpropogates using $d_j = h'(a_j)\sum_k w_{kj}d_k$ to obtain the error for each hidden unit in the network. The value of d for a particular hidden unit can be obtained by propagating the previous values of d backwards from units higher up in the network. Because we already know the error values for the output units, we can evaluate them for all of the hidden units in a feed-forward network by recursively applying the equation above.

      B. The equation $\frac{dE_n}{dw_{ij}} = d_j z_i$ is used to evaluate the required derivatives where E_n is the summed error function, w_{ij} is the weight, z_i is the activation of unit i and d_j is the error.

   (f) The second phase of training is weight update using the error values d_k calculated in phase one.

      i. For each weighted synapse you must:

         A. first multiply d_k and the input activation to get the gradient of weight,

         B. the change the weight in the opposite direction of the gradient by subtracting a ratio of it from the weight.

   (g) Phases one and two are then repeated until the desired accuracy is achieved.

   (h) There are practical limitations to the performance of MLPs such as requiring inputs to be scaled and normalized, and slow convergence that is not guaranteed. Because gradient descent may converge to any local minimum on the error surface, the MLP may not reach the absolute minimum.

6. Despite these limitations, multilayer perceptrons using a backpropogation algorithm are the standard algorithm for any supervised-learning pattern recognition process .

7. They are useful in research in terms of their ability to solve problems stochastically, which often allows one to get approximate solutions for extremely complex problems.

4.4 Deep Belief Networks (DBN) [8]

1. Deep learning is a field of machine learning based on algorithms for learning multiple levels of representation of data in order to model more complex relationships.

2. Features are arranged in a hierarchy known as a deep architecture, in which higher-level features and concepts are defined in terms of lower ones.

3. Deep learning algorithms have proven to be skilled at feature extraction in high-dimensional, structured data, and can implicitly learn the data's distribution function.

   (a) Most of these algorithms are unsupervised, and thus are very useful at finding patterns that humans might not arrive at due to prejudices or predilections.

4. Deep belief networks are probabilistic generative models whose structures are composed of multiple layers of stochastic latent (deterministic and present, although not visible) variables typically having binary values.

   (a) The top two layers have mutual undirected symmetric connections and form an associative memory.

   (b) The lower layers get top-down, directed connections from the layer above.

   (c) The states of the units in the lowest layer represent a data vector, as in MLPs.

   (d) Typically, DBNs use a logistic function of the weighted input received from above (or below) to determine the probability that a binary latent variable has a value of 1 during top-down generation (or bottom-up inference), but other types of variables can be used as long as the variables are in the exponential family (so that log probability is linear in the parameters).

5. The two most significant properties of deep belief networks are:

   (a) An efficient layer by layer procedure for learning top-down, generative weights which determine how variables in a layer depend on the state of the layer above, and

   (b) after learning, the values of latent variables in every layer can be inferred by a single bottom-up pass starting with the observed data vector and moving backwards using the generative weights.

      i. This means that you can add a final layer of variables (composing the desired output) and backpropogate the error derivatives, as in the MLP.

         A. Backpropogation will work better if feature detectors in the hidden layers are initialized by learning a deep belief network that models the structure of the input data.

6. They're learned one layer at a time, using the values of the latent variables in one layer as the training data for the next layer.

   (a) It's a greedy algorithm, optimizing at each step for that step alone with no foresight.

7. Very importantly, they can be combined with other learning procedures to fine-tune the weights. This makes them very useful tools for data representation.

8. You can view a DBN as a composition of single learning modules, each of which is a type of restricted Boltzmann machine with a layer of visible units (representing data) and hidden units (representing features that capture higher-order correlations), whose layers are connected by a matrix of symmetrically weighted connections. There are no connections within a layer.

9. They can be represented by the function $p(v) = \sum_h p(h|W)p(v|h, W)$ where W is matrix of weights, v is a vector of activities for the visible units, and h is a sample vector of hidden units.

10. Deep belief networks are best at generating and recognizing images, video sequences, and motion-capture data.

5 Approaches considered/selected approaches

5.1 Questions of intent

1. How am I going to approach this problem/what is my focus?

   (a) I could look at different algorithmic approaches and compare their performances to learn more about algorithms.

   (b) Or I could focus on having a finished, working product.

      i. I could write it from the ground up for the programming experience, maybe even end with a library.

      ii. Or I could use software and libraries already available, piece them together, and try to make the most advanced product I can.

   (c) Final Decision: Full focus on functionality and marketability.

      i. Since this is a relevant problem with real applications, a working product is more important than an understanding of methods.

      ii. Also I want a product to showcase my abilities.

2. What do I want the network to output?

   (a) Pure binary classification

      i. for this problem, black and white answers aren't likely to be very accurate on the whole

      ii. it's irresponsible for a medically-focused amateur application to give definite answers

   (b) Likelihood

      i. better, but it's a classification problem

   (c) Prognosis

      i. too difficult, adds too much to the problem

   - (d) Best idea is binary classification with a likelihood statistic.

3. Different algorithmic approaches
   - (a) Neural nets work incredibly well on classification problems.
   - (b) If lacking length, later I can include a section on different machine learning algorithms and their strengths and weaknesses.

4. Preprocessing techniques
   - (a) (this section will be expanded once I begin actually programming the image recognition portion)
   - (b) Normalizing techniques
     - i. need to make sure image is centered and scaled (while still maintaining information about diameter), with lighting/color accounted for
   - (c) What data do I want to give the neural net?
     - i. Do I want to put all the computing of features during the preprocessing, or do I want to give the neural net a "virtual retina" to determine features itself?
   - (d) What type of input do I want to give the neural net?
     - i. binary vector?

5. Selection of classifiers [1]
   - (a) see practical limitations
   - (b) ABCD rule
     - i. asymmetry decided by bisecting lesion with 2 optimal axes, overlapping two halves along axis, then dividing non-overlapping area difference by total area
     - ii. border: create regions of interest (ROI) by dividing lesion into pie pieces (image segmenting), see whether there is a sharp cutoff or a gradual fade at periphery based on thresholding, region growing, and color transformation, or by using a classical edge detector algorithm
     - iii. color: look at number of colors present (light/dark brown, black, red, white, slate blue), the color texture, RGB, saturation, hue, Y-Luminance, UV chrominance components, spherical coordinates LAB average, min/max/avg/sd of channel values color intensity
     - iv. d is for differential structures: number of structural components present (pigment network, 3+ dots, 2+ globules, >10% structureless areas, 3+ streaks)

  - v. if wavelet analysis is added, accuracy improves by 60%
- (c) Pattern analysis
  - i. identify specific patterns
  - ii. global: reticular, globular, cobblestone, homogenous, starburst, parallel, multi-component, nonspecific
  - iii. local: pigment, network, dots/globules/moles, streaks, blue-whitish veil, regression structures, hypo-pigmentation, blotches, vascular structures
- (d) Menzies method
  - i. add negative and positive features
  - ii. negative features: symmetry, single color
  - iii. positive features: blue-whitish veil, brown dots, pseudopods, radical steaming, scar-like depigmentation, peripheral black dots/globules, 5-6 colors, blue/grey dots, broadened network
- (e) 7-point checklist
  - i. 7 criteria assess chromatic characteristics and shape/texture weighted differently, gives score, if 3+, then malignant
  - ii. 1– atypical pigment network
  - iii. 2– blue-white veil
  - iv. 3– atypical vascular pattern
  - v. 4– irregular streaks
  - vi. 5– irregular dots/globules
  - vii. 6– irregular blotches
  - viii. 7– regression structures
- (f) Texture analysis
  - i. attempts to quantify fine, rough, irregular
  - ii. identify, measure, and utilize differences
  - iii. statistical and structural
  - iv. lots of dissimilarity equations

6. Software considered
   - (a) OpenCV
     - i. Open Source Computer Vision Library
     - ii. Library of programming functions mainly aimed at real time computer vision and image processing, developed by Intel and now supported by Willow Garage.
     - iii. Includes a statistical machine learning library.
     - iv. Has a Python wrapper.
     - v. Supported by Mac OSX

vi. Has official releases; is well-established.

(b) PyCVF

i. Python Computer Vision Framework

ii. Takes in library, dependencies, toolkits, then uniformizes concepts to make applications that use and extend the framework.

iii. Can use OpenCV and Orange.

iv. Has its own concepts: database, models, nodes (processing units), machine learning models, datatypes, structures, and experiments.

v. Has its own datatypes: images (3d numpy array), video, audio, vectors, and vector sets.

(c) PyVision

i. Object-oriented Computer Vision Toolkit that allows rapid prototyping and analysis of computer vision algorithms.

ii. Provides simple framework to unify scipy/numpy, OpenCV, and other computer vision and machine learning software packages.

iii. Has a set of analysis tools.

iv. Mainly used for facial recognition, but open to additions.

v. Someone's Ph.D. project.

(d) PyCV

i. INACTIVE– relies on old versions of scipy, numpy, and OpenCV.

ii. Package of C++ and Python modules implementing various algorithms that are useful in computer vision.

iii. Augments the capabilities of OpenCV.

iv. Python interface to OpenCV.

v. Fast training and selection of Haar-like features, and variants of AdaBoost boosting algorithm.

vi. Once again, focuses on facial recognition.

vii. Once again, someone's Ph.D. project.

(e) Pynopticon

i. Toolbox that allows you to create and train your own object recognition classifiers.

ii. Create a dataset of your favorite image categories, choose some feature extraction methods, post-processing, and a classifier to train, it does the rest.

iii. Integrates with Orange for a GUI.

iv. Couldn't get Orange to recognize it in the past.

(f) Orange

i. Component-based data mining and machine learning software suite.

ii. Visual programming front-end for explorative data analysis and visualization, and Python bindings and libraries for scripting.

iii. Includes comprehensive set of components for data preprocessing, feature scoring and filtering, modeling, model evaluation, and exploration techniques.

iv. Has a Python interface for developing new algorithms and procedures.

v. Large toolbox.

vi. Flexible, powerful, well-established.

(g) Theano

i. J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley and Y. Bengio. “Theano: A CPU and GPU Math Expression Compiler”. Proceedings of the Python for Scientific Computing Conference (SciPy) 2010. June 30 - July 3, Austin, TX (BibTEX)

ii. Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

iii. Combines aspects of a computer algebra system (CAS) with aspects of an optimizing compiler, which is particularly useful for tasks in which complicated mathematical expressions are evaluated repeatedly and evaluation speed is critical.

iv. It has optimized symbolic features such as automatic differentiation, graph merging, simplification, using memory aliasing to avoid redundant calculation, and more.

v. Theano was written at the LISA lab to support rapid development of efficient machine learning algorithms.

vi. It’s named after the Greek mathematician, who may have been Pythagoras’ wife.

vii. It’s released under a BSD license

5.2 Practical Limitations

1. Image acquisition techniques

(a) Methods of dealing with lighting

i. Lighting is variable, so colors distort.

ii. this is more of an issue with the app, not the training group, but it’s important to keep in mind

iii. have the user take pictures in multiple different lightings (florescent, sunlight)

iv. have a built-in algorithm that turns down pictures that are too poorly lit

v. color bar on the side, self-reported absolute color, adjust image appropriately

vi. user questionnaire on variety of topics

(b) Camera quality

i. Commercially available cameras work poorly for miles less than 0.5 cm in diameter.

ii. the majority of pictures submitted will be from sub-par point-and-shoot or camera phones

iii. can I get a training group of images of that quality?

iv. there won't be enough detail for proper edge detection

v. possibly include user questionnaire about the ABCDE of the lesion, weight the importance of their quantifiable answers against what the camera is viewing across different features

(c) Formatting

i. what types of images should I accept?

ii. I don't know enough about filetypes at this time to say conclusively

(d) User error

i. probably going to have to trust the user, maybe include pictures in the questionnaire, make it as fool-proof as possible

ii. the best way to account for this is to include user error in the training set

iii. can't go through and complete a questionnaire for each lesion in the training set

2. Selection of classifiers

(a) Due to limited computing power and a single programmer, some more complex and accurate schemes have to be passed over, and previously established schemes may have to be used.

3. Software

(a) Due to time constraints and a single programmer, I must use software already written rather than customized.

(b) Due to limited funding, no professional software can be used, only freeware.

(c) Some very promising libraries are now inactive due to conflicting retrograde dependencies with host programs.

(d) Must have wrapper in language I am familiar with (preferably Python).

5.3 Selected Approaches

1. This section will be written after programming.

6 Final product

6.1 Structure of ANN

6.2 Marked-up code

6.3 Results with benchmarks

7 Future (or current) plans/applications

7.1 Phone app

1. Development process

7.2 Web app

1. Development process

References

[1] Overview of Advanced CV Systems for Skin Lesions Characterization

[2] http://www.nlm.nih.gov/medlineplus/skincancer.html

[3] http://www.who.int/uv/faq/skincancer/en/index1.html

[4] Retrieval and Ranking of Biomedical Images using Boosted Haar Features

[5] "An Introduction to Neural Networks," Anderson

[6] http://www.no-free-lunch.org/

[7] http://www.deeplearning.net/

[8] Geoffrey E. Hinton (2009), Scholarpedia, 4(5):5947. doi:10.4249/scholarpedia.5947

[9] http://www.ai-junkie.com/ann/evolved/nnt1.html

[10] http://ulcar.uml.edu/~iag/CS/Intro-to-ANN.html

[11] http://www.mayoclinic.com/health/melanoma/DS00439/DSECTION=tests-and-diagnosis

[12] Pattern Recognition and Machine Learning –Bishop