On Deep Learning and its Applications in Medical Related Image Recognition

Edward Huber
Rochester Institute of Technology
emh3580@rit.edu

Mark Nuneviller
Rochester Institute of Technology
mn1846@rit.edu

Abstract
Image recognition software is being researched and use more and more each year, and its applications seem to be unbounded. It can be used in anything and everything from self driving cars to medical devices. Currently, many doctors are working with Data Scientists to design artificial intelligence that can aid in identifying a plethora of medical ailments in the human body. One such ailment is cancer and more particularly skin cancer. As there are seven very common cancerous lesions found in patients with or developing skin cancer, it seems to be the perfect medical condition to give focus to in regard to image recognition. We set out to create a machine learning algorithm that would accurately predict the class of a cancerous lesion as one of seven pre-labelled types of cancer based on photo data provided to the program using a convolutional neural network in Python through the Keras package operating in conjunction with TensorFlow.

1. Introduction
One in five Americans will develop skin cancer by the age of 70, a statistic that makes skin cancer the most common form of cancer found around the world. More people are diagnosed with some form of skin cancer each year than the aggregate of all other monitored cancers. Worldwide, more than 5400 people die of non-melanoma related skin cancer per month and 6850 from melanoma per month. The average five year survival rate for all melanomas in the United States is ninety-two percent, but when detected early, patient survival rate jump to over ninety-nine percent. However, when skin cancer is detected late, the cancer is known to spread to the lymph notes, and survival rates fall to sixty-five percent; if undetected when the cancer has passed to the lymph nodes it tends to spread to other vital organs and a survival rate of only twenty-five percent. There is no doubt that the cost of treating cancer is large, and the United State spends approximately $8.1 Billion per year in healthcare costs that are related to skin cancer. {4]

2. Background
The aforementioned being said, it is clear that a major part of preventing and/or curing skin cancer is early detection. For that reason, we have decided to test our abilities at creating a machine learning model that can accurately analyze images of skin lesions and predict whether or not the image shown is of a potentially cancerous lesion. While our project focused only the machine learning aspect of the problem at hand, a finalized version could certainly include some sort of web or phone based applications from which the user takes a photo of the potential lesion which is subsequently uploaded to the cloud server where a pre-trained model is deployed to identify the lesion within seconds, alerting the user whether they should seek medical attention.
However, and it should go without being said, a visit to a Dermatologist and applicable doctors is generally superior and preferred to an image recognition application, as these doctors have years of experience seeing and monitoring the lesions and the side affects that go along with them. An application can classify the lesion as one of any given number of subgroups and should never be considered a means of taking the place of a trained medical doctor. While they are highly skilled, adaptable, and adept at problem solving, they are not scaleable. A doctor is only able to see a finite and limited number of patients per day, while an application is infinitely scaleable and therefore can be made available at a low cost, perhaps for a nominal fee of a few dollars, maybe even free if monetized appropriately. Cost of access to healthcare is something that holds back many patients in need from seeing a doctor when they should. Relating back to early detection and its affects on survival rates, waiting will cause survival rates to fall. An application such as one running pre-trained classification models is a way for those that are hesitant to see a doctor because of the cost a way to accurately and dependably decide if it is the step that they need to take. It can be used as a low cost pre-screening tool that acts as a catalyst in getting potential patients into a medical office at the right time. The overall effects of such an application if approved and able to be available to the public include lowering overall healthcare costs, providing diagnostic or pre-screening access to more people, and improving the health outcomes for those without health insurance.

Deep learning in healthcare already goes far beyond skin cancer recognition. Some other applications for deep learning algorithms include:

- Genomics, the practice of inferring an organism's phenotype from its genotype. In a sense, genomics asks, how are physical traits mapped to the genome? After sequencing a genome, the task of predicting their outcomes based on the genome is most appropriately solved through deep learning.
- Insurance fraud, which in the U.S. costs the healthcare system $68 billion per year. [5] This is money that could otherwise go towards providing healthcare to people in need. There are millions if not billions of healthcare transactions per year, a number that is far too large to be analyzed by the human eye. Deep learning can be applied on a massive scale and is trainable to detect this type of fraud
- All medical imaging devices can be improved as deep learning based imaged recognition models are improved

The average cost to bring a drug to market is $2.5 billion and pharmaceutical companies are employing deep learning to shorten the cycle to discover and lower the cost.

3. Related Works
To date there are many studies in which deep learning has been used in healthcare methods and has had a profound effect. In one study, researchers used a similar concept to recognize polyps during a colonoscopy. Colorectal cancer (CRC) is the second leading cause of cancer deaths in the U.S. CRC arises from precancerous polyps that dwell in the colon for an average of ten years. At some point, these polyps can become cancerous, which is why regular colonoscopy and the removal of polyps is generally highly suggested. Just as is the case with skin cancer, early detection is key to prevent and treat CRC. Over seventy percent of CRC cases are considered to be preventable is early detection measures are taken. Unfortunately and even with patients who do have regular checkups, an estimated twenty percent of precancerous polyps are missed by the Colonoscopists during examinations. Aiding these professionals with image recognition artificial intelligence can undoubtedly increase detection rates and therefore reduce the patient's risk of either getting the cancer or having the cancer spread further. Researchers in this study used Keras and TensorFlow Convolution Neural Network models to analyze 8400 colonoscopic images. [9]

All of these models consisted of the same basic building blocks: convolutional layers using rectified linear (ReLu) activation functions, fully connected layers, maximum pooling layers, non-linear activation functions, and dropout layers. Models used the Adam optimizer and the last layer used Softmax output units. Data augmentation, a process by which a slight amount of noise is introduced to the images (by rotating, flipping, zooming, or shearing the image), and dropout layers were used to improve model generalization. In addition to scratch build models, the researchers also employed pretrained and prebuilt models such as ResNet50 and VGG16. The best end result was at ninety-six percent accuracy [9] out of sample images - as good or better than human prediction power. This research is a great example of how a colonoscopies might use artificial intelligence to enhance their own abilities.

In a second study, researchers used deep learning to analyze images of eyes and attempt to predict Diabetic Retinopathy (DR). DR is one of the leading causes of blindness in the world. Approximately 425 million people worldwide have diabetes, and DR is the most common and worst microvascular complication in patients with diabetes. It can advance without symptoms up until the patient completely loses eyesight. Nearly all type 1 diabetics and sixty percent of type 2 diabetics will develop retinopathy during the first 20 years from the onset of their diabetes. Like many chronic diseases, early detection of DR is a key to treatment. However, in many if not most instances, DR is left to progress until the damage is irreversible. To screen for the disease, retina specialists assess color fundus photographs (CFP) which are high resolution images of the internals of the eye in order to diagnose.

Researchers believe that they key to expanding screening services could be in artificial intelligence. In the study, convolutional neural networks were used to classify CFP images as having DR or not having DR. They implemented transfer learning, taking a pretrained model, in this case the popular inceptionV3 model, and retrained it on the CFP data. They used the Adam optimizer and ran their model for 50 epochs, then evaluated the classifier by Area-under-the-Curve (AUC) analysis. Their best model had an AUC value of 0.79 using five fold cross validation. Their data was curated from an unrelated clinical trial, so the researchers thought that their model was not generalizable as the clinical trial population was specifically selected for another condition and did not represent a randomized or scientifically selected sample for their project. However, this project shows promise and the researchers still hope to build on its successes using a different data source. [1]

4. Methodologies and Tools

4.1 Deep Learning

In the 1950's scientists and mathematicians became awed by the idea of Artificial Intelligence. By the 1980's the study and implementation of Machine Learning had become the focus, and in recent years the concept of Deep Learning, a specific implementation of machine learning into artificial intelligence has become the concept driving major breakthroughs in everything from the auto industry to advances in healthcare. Deep learning is commonly defined as a subfield of machine learning concerned with algorithms inspired by the structure and function of brain, otherwise known as artificial neural networks. One can simplify this idea in just a few words: learning by example.

Deep learning was conceptualized in the 1980's with the rise of machine learning, but it was not until the 2000's that it was able to be implemented in large scales. The main reason is that in order to be most accurate and appropriate for use in the desired applications, deep learning algorithms need large, even huge (think terabytes) of accurate and well-labeled data. Secondly, deep learning for large scale applications (such as self driving cars) requires a massive amount of computing power the likes of which was not readily available until the post 2000 era. Without systems such as high performing GPUs, deep learning algorithms are inefficient, as they may take weeks to train on low performance system. On modern systems, training times are usually a few hours or less.

Deep learning is categorized as a supervised style of machine learning. This means that the computer is given what are called training sets and test sets. The computer runs the code over the training set in order to learn a function, each observation being an input and output of the function. For a very basic example, if a computer is provided two pictures, one of a dog and one of a horse while given the correct answer for each picture, one can then show the computer a different picture of a horse and ask the computer to decide whether it is a dog or a horse. In reality, to achieve maximum accuracy, the computer would need to be provided with thousands of labelled pictures of each class to learn exactly what a dog and a horse would look like, respectively. Deep learning is not restricted to vision, although that is what this project focuses on. It can be implemented for speech recognition, general audio recognition, and natural language processing.

4.2 TensorFlow

TensorFlow is an open source machine learning library originally created for use with Python by Google Brain programmers. It computes using tensors, a tensor being a vector or a matrix of n-dimensions that can conceivably represent any type of data. In a tensor, all of the values have an identical data type and known shape, which consequently is the dimensionality of the matrix or array that is the tensor.

It operates by creating data flow graphs, or structures that show how data moves through a series of processing nodes. As it is based on graph computations, developers are able to visualize the construction of their neural network using Tensorboard, a very useful tool for debugging their model. Every node represents a mathematical operation while each connection between nodes is a tensor. [11] While TensorFlow is written for use with Python, all the mathematical operations that the program performs are done as C++ binaries, with Python acting as the traffic director between the nodes and piecing them together. [10]

4.3 Convolutional Neural Networks

The objective of this project is to use a convolutional neural network (CNN) to accurately predict the type of skin cancer in each image provided to the machine, using the 'Skin Cancer MNIST: HAM10000' data available from Kaggle. To understand how and why this objective might work, one needs to understand the basics of a CNN's, which currently are a very common deep learning technique applied to image visual image analysis. CNN's can be categorized as regularized multilayer perceptrons.

Regularization is the mathematical technique of adding data or information to prevent over fitting of a model. Regularization helps to do this because classification models can only infer a function of a variable based solely on actual observations without taking into account new observations that will occur over time. For every CNN model there is a loss function, which describes the numerical cost of predicting an output based on the observations provided based on a loss metric L(X,Y) where X is the design matrix and Y is the vector of observations or target outputs. One method of decreasing the loss is to add variables, and generally speaking, any number of variables may be used. Many times adding variable will increase the accuracy of the model, and in doing this the model may become overfit. This means that while it performs well on observed data, the model will perform poorly when predicting and generalizing, as it tends to recall the data that it was shown repeatedly and learned the background noise from that specific data. Regularization involves penalizing the loss function in order to account for changes in data that may or may not be seen.

At its most basic level, a perceptron is a single layer neural network, a linear classifier. It consists of input values, weights (bias), net sum, and the activation function. A perceptron works by taking all input values and multiply them by their weights according to the data, and these weights are summed to give the weighted (net) sum, which used in the activation function resulting in an output. The weights of each input can be thought of as something similar to coefficients in a regression model.

Understanding tensors, regularization and the basic definition of a perceptron is essential to understand how a CNN works. Data is input to the model as a tensor with shape:

(number of images) x (image width) x (image height) x (image depth)

Once each image has passed through the convolutional layer, the image is abstracted a feature map returning shape:

(number of images) x (feature map width) x (feature map height) x (feature map channels)

The above mentioned features should always be definitive of a convolutional layer in a neural network. Each layer convolves the input before passing it on to the next layer. After the prescribed layers have been processed, the program will perform classification on a validation set of images, outputting the predicted classification in relation to the actual classification and an accuracy percentage.

For this project, a mixture of ReLu functions were used in the initial layers and the Softmax function was used for the final layer prediction. A ReLu is a Rectified Linear Unit most commonly used in activation of deep neural nets. ReLu is preferred as an activation function for a few reasons:

- It offers fast, and effective training of deep neural nets on large datasets.
- It is one-sided (does not handle negative values), although this can also be a problem.
- It uses efficient gradient propagation which generally eliminates vanishing gradient issues
- It handles computation efficiently with addition or multiplication.
- It scales well.

While these are benefits of the ReLu function, its two main problems are that it is not differentiable at zero and also that it does not handle negative values. This being said, ReLu is still the most commonly used activation function in deep neural networks. [6] It is common practice to follow it with a Softmax function in the final layer. A Softmax function is a normalized exponential function that is a generalization of the logistic function that turns the k-dimensional vector of arbitrary real values to a k-dimensional vector of real values in the range of zero to one. This is because Softmax represents a probability distribution over k different outcomes. For this reason it is highly suggested to be used in discriminative models like cross-entropy which was used in the code for this project. It tends to operate best on the outer layers of a neural network, again generally the last layer, as its probability distribution representation does not limit the classification model to just two possibilities. It calculates the probability of predicting each and every class in a multi-categorical dataset. [2]

4.4 Python and Keras

In order to use a CNN on the HAM 10000 data, we used the programming language python combined with its Keras package. Anyone involved in coding, machine learning, AI, et center knows that python is by far the most common programming language used to build and train machine learning algorithms, particularly neural networks such as CNN's. Python is the language of choice for a few reasons:

- Python involves simple, clear, and accessible syntax.
- If Keras was not decided on preemptively, Python frameworks and libraries are numerous for machine learning and neural network programming
- Because of its clear syntax, Python is quite suitable for collaboration.
- There is large community support for Python and its ancillary packages, thus it is always being updated and experiencing breakthroughs daily.

Recall that an explanation of TensorFlow was previously provided. That is because it is necessary to understand TensorFlow in order to understand what Keras packages do and how to implement them. Whenever Keras is used, TensorFlow operates on its front end, allowing simply coded yet complex neural networks to be created through Keras. Keras is an open source deep learning library that enables fast experimentation by providing a somewhat standard neural network model implemented through a basic programming model. Keras allows the programmer to use high level abstraction such as models, layers, and hyper-parameters instead of requiring the programmer to know how to and spend the time converting the data into tensors and matrices. As the Keras documentation states, “It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.” [3]

4.5 Local and Cloud Model Building

During the model building portion of this project, we used two separate approaches. First was to build CNN models from the ground up on our local machines. Secondly, we used image recognition software on cloud services.

Our locally built models were done in Jupyter Notebook using Python 3 and the Keras package on a Macbook Air and an ASUS ROG laptop. The images were resized to 120 by 90, vectorized, and then split in a training, validation, and testing sets at a 70-10-20 ratio, respectively.

Our chosen data set is highly imbalanced, so we applied ImageGenerator to our training set. This is a preprocessing method that slightly alters the original image by flipping, rotating, shifting, zooming, and shearing. These alterations inject some noise into the data, and the end result is actually a more generalizable model. ImageGenerator is only used on the training data, not testing or validation sets.

We used scratch-built, multi-layer CNN models and experimented with the Adam, Nadam, and SGD optimizers. Learning rates between 0.01 and 0.001 were tried. As this is a classification problem, we used categorical cross-entropy as the appropriate loss function.

Neural networks are highly complex and flexible models. They are capable of tremendous predictive power, but their extreme flexibility can lead to overfitting. That is why they are fit in steps called epochs and validated with a separate data set throughout the training period. Another way to increase generalizability in a CNN is through the use of dropout. Dropout is a feature that forces sections of the deep network to shutdown, or 'drop out', forcing the algorithm to find new connections. Finding these new pathways leads to a final model that is less promote overfitting.

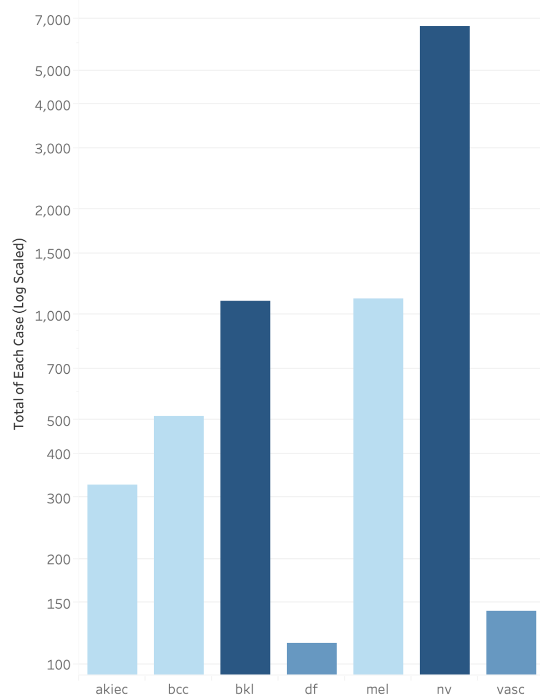
Models were allowed to train for fifty epochs and then inspected for over-fitting via tuning plots showing validation accuracy versus number of epochs.The out-of-sample test set was used to perform model evaluation on the basis of accuracy and inspection of the confusion matrix. Baseline accuracy was defined as results from predicting the most common class (for example the baseline accuracy for a 6-sided die is 0.17).

Our cloud models were built on the Google Cloud using Google's image classification service called AutoML Vision. Google does not share the details of the inner workings of AutoML, but we believe that it is safe to assume that they are using tensor flow and a very deep pretrained CNN model. We allotted 50 node hours for training; in this case a node is an 8 core CPU with 30 GB of RAM and an NVIDIA Tesla V100 GPU. It should be noted that this GPU is especially designed for AI applications and costs \$3200. AutoML runs 8 node clusters to train models, so 50 node hours in completed in more like 6 hours in real time. Google's AutoML uses a very similar train, validation, test split. Models are evaluated for accuracy on the test set as well as area-under-the-curve (AUC), sensitivity, and specificity.

5. Modeling

5.1 The Data

For this study, we decided to use the HAM10000 dataset, otherwise known as the "Human Against Machine with 10000 training images" dataset. The data was originally released by Phillip Tschandl, Cliff Rosendahl, and Harald Kittler in 2018 as they sought out to train "neural networks for automated diagnosis of pigmented skin lesions". [8] As described by the authors, the data was collected by aggregating "dermatoscopic images from different populations acquired and stored by different modalities". [8] According to them the observations include an overall representation of the "important diagnostic categories in the realm of pigmented lesions". [8] To date, this data can be found on many websites as they have provided it for use to the Data Science community, and we obtained it through https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000. The set consists of 10015 images distributed as follows:

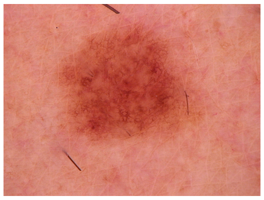


Count of Dx Type for each Dx. Color shows distinct count of Dx Type.

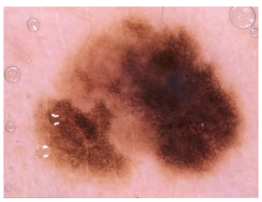
The exact breakdown (with examples shown):

- 6705 cases of melanocytic nevi, benign neoplasms of melanocytes which are generally symmetric in color and structure.

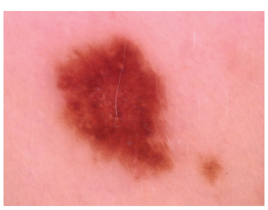
- 113 of melanoma, a malignant neoplasm derived from melonocytes, with only pigmented version being included in the study, according to the authors.



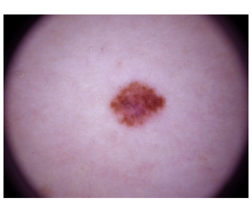
- 1099 of pigmented benign keratosis, a generic class of the disease that includes multiple subgroups and generally is grouped as seborrheic keratosis. It may cause confusion to machine learning algorithms in that it can appear to be similar to melanoma.



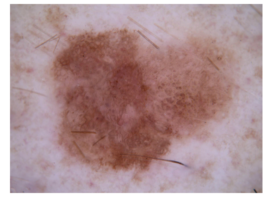
- 514 basal cell carcinoma, which according to skincancer.org is the "most common form of skin cancer and the most frequently occurring form of all cancers" (The Skin Cancer Foundation, 2020). These lesions look like red or pink patches and growths, bumps, scars, and open sores, widely varying from person to person.



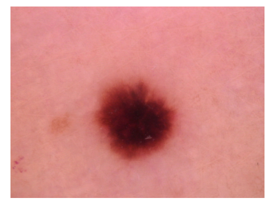
- 327 actinic keratosis, a rough and scaly lesion usually developing from overexposure to the sun, most commonly found on the face, back of hands, and other commonly exposed body regions.



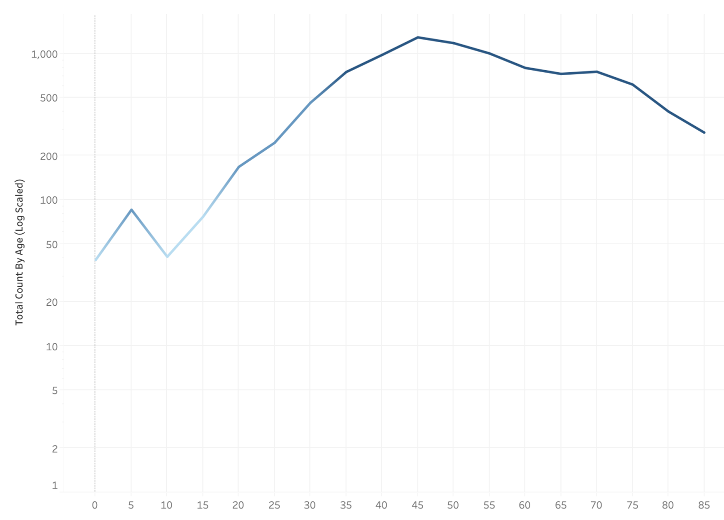
- 142 vascular lesions, including cherry angiomas, angiokeratomas, and pyogenic granulomas. These are generally abnormalities in blood vessels. While not technically dangerous they are included as their possible link to skin cancer is researched.



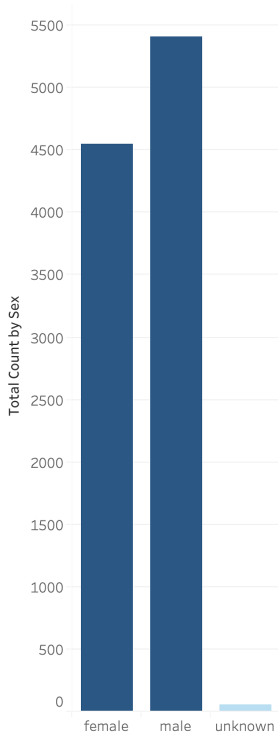
- 115 dermatofibroma, benign nodules that are commonly found on the lower leg. These are also known as cutaneous fibrous histiocytoma.



Although many factors available for analysis in the data were not used for modeling in this study, it is important to look at some key statistics of the data to understand what the background of the analysis is. Looking at the graph below, one sees that age appears to be a factor in who develops one of the seven types of cancers included in the study.
We see that the total cases by age peaks at age 45 for the observations collected and tapers off more slowly as age increases over 45 than it grew as age approached 45.



The trend of count of Dx Type for Age. Color shows distinct count of Dx Type.



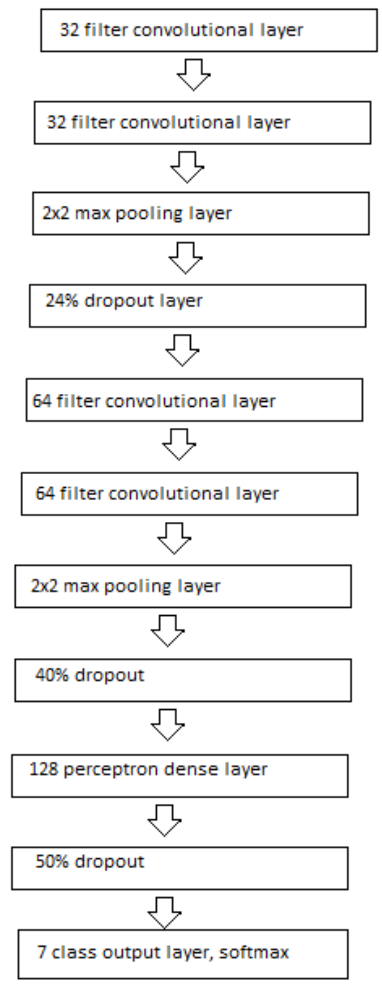
Count of Dx Type for each Sex. Color shows distinct count of Dx Type.

Here we see that gender may or may not be related to who develops one of the included types of skin cancer, as we see just under 1000 more men women were included in the study. It is important to note that there were a few unknown labels for the sex variable. A final variable of interest that is available in the data is where on the body the lesion occurred. Looking at the next table provides an interesting glimpse at the distribution of occurrence on body region for each sex. Most appear to be generally an even split with a few notable exceptions. Acral (extremity) occurrences were only found in female, as well as 70.83% of genital occurrences. On the reverse, men tend to show significantly more occurrences on the back and scalp.

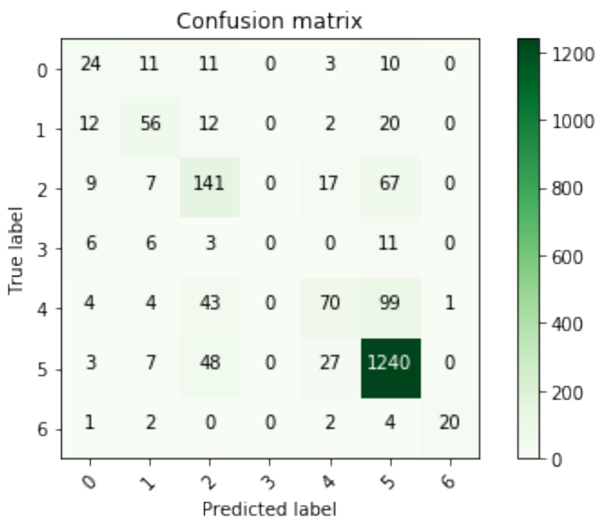
	female	male	unknown
abdomen	42.37%	57.44%	0.20%
acral	100.00%		
back	38.46%	61.45%	0.09%
chest	34.89%	65.11%	
ear	53.57%	46.43%	
face	46.71%	53.29%	
foot	54.23%	44.83%	0.94%
genital	70.83%	29.17%	
hand	63.33%	36.67%	
lower extremity	55.46%	44.54%	
neck	43.45%	56.55%	
scalp	24.22%	75.78%	
trunk	45.44%	54.34%	0.21%
unknown	40.60%	39.32%	20.09%
upper extremity	44.36%	55.64%	

5.2 Modeling Results

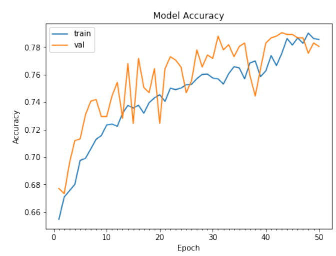
Many different CNN models were compared in this project, but the best locally built model was a simple multi layer CNN, its structure shown below:

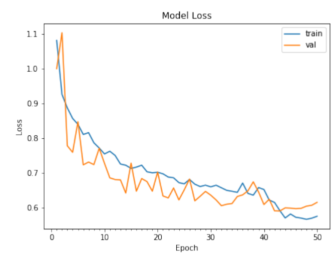


This model was fit using the Adam optimizer and had an accuracy rate of seventy-seven percent with a baseline accuracy of sixty-nine percent. Inspecting the confusion matrix we can see that the model had a bias towards predicting melanocytic nevi (5) and dermatofibroma (2), and was therefore most sensitive to these types of cancer. The model was the least accurate at predicting melanoma (4).



When we look at tuning charts, we can see that fifty epochs were a sufficient number of steps to run the model. By fifty epochs, the loss rate and the accuracy have leveled off. When comparing the training set and validation set lines, we can see that overfitting is not a problem. In an overfit model the validation accuracy would be far below the training set accuracy. In modeling this data set we did not find that increasing the complexity of the model resulted in increased accuracy.





True Label \ Predicted Label	akiec	nv	bkl	mel	df	vasc	bcc
akiec	55%	6%	12%	6%	3%	-	18%
nv	-	98%	1%	1%	0%	0%	0%
bkl	1%	14%	81%	4%	-	-	1%
mel	-	23%	5%	72%	-	-	-
df	-	36%	-	-	64%	-	-
vasc	-	7%	-	-	-	86%	7%
bcc	4%	6%	-	-	-	-	90%

True Label \ Predicted Label	akiec	nv	bkl	mel	df	vasc	bcc
akiec	18	2	4	2	1	-	6
nv	-	657	5	4	2	1	1
bkl	1	15	89	4	-	-	1
mel	-	26	5	80	-	-	-
df	-	4	-	-	7	-	-
vasc	-	1	-	-	-	12	1
bcc	2	3	-	-	-	-	46

We tried many ways to add layers to our model, but in every case the more complex model would revert to baseline accuracy of sixty-nine percent. When evaluating on the test set, we would find that these models would predict class 5 (malnocytic nevi) for nearly every record. Besides adding layers, we also experimented with different optimizers and learning rates. In each case the algorithm would get stuck at the local optima of predicting class (5), and not find the global optima.

The AutoML required some data wrangling to format the images in a way that it can digest. After uploading the files, the model was trained for fifty node hours. The cloud based model was ninety-one percent accurate. This model still has a bias towards class (5), malnocytic nevi, and many of its mistakes are the misclassification of other lesions as class (5). This model struggled most with identifying class (1), actinic keratoses. The model resides on Google's cloud, and is ready to be deployed to the web at any time. AutoML did outperform our locally built model, and was simple to implement. However, two AutoML models were ran for the cost of $175 each, for a total of $350 (we had free trial credits to cover the expense). While it is clear that AutoML is a very good machine learning platform, it is also expensive.

6. Conclusions

Skin cancer is the most common type of cancer in the world. While it is usually not life threatening, lesions that are left untreated can spread to other parts of the body and increase risks. In order to automate and increase access to skin cancer screening, we developed machine learning models to classify skin lesions. While we focused on the machine learning portion of the software, it could later be adapted to an internet or phone based application.

Using Keras on Python, we trained a CNN model that was seventy-seven percent accurate in distinguishing between seven different classes of cancerous lesions. We also developed a cloud based model that was ninety-one percent accurate. We used the

HAM10000 dataset—a collection of 10,000 labeled images belonging to seven different classes of cancer.

This was a very interesting project, and the final model, particularly our cloud based model, could be used to in assisting physicians in diagnoses. CNN based deep learning certainly has a role in skin cancer diagnoses, and this project proves that. However, due to the nature of our data set, our model can not be used as a screening tool for the population at large. Our data set only included cancerous lesions, no benign lesions. What the patient would want is a cancer vs non-cancer classifier to use a prescreening tool. If the app diagnosed the lesion as cancerous, then the patient would head to a doctors office for an examination and perhaps a biopsy. If there was a data set of cancerous vs non-cancerous lesions existed, then the exact same methodologies that we employed would probably work in that scenario too.

We found that CNN based models are very good at distinguishing small details in images, as the average human would certainly have done more poorly in the classification of the HAM10000 data set than our models. Other studies have found CNN to be useful diagnosing other ailments such as colon cancer and eye degeneration from diabetes. Image classification will only get more advanced and more accurate in the future and will hopefully provide low cost screening tools to billions of people worldwide.

References

1. Arcadu, Filippo, et al. Deep learning algortihm predicts diabetic retinopathy progression in individual patients. Npj Digital Medicine. Article Number 92. 2019. https://www.nature.com/articles/s41746-019-0172-3

2. Mahmood, Hamza. (Nov. 26,2018). "The Softmax Function, Simplified". Retrieved on April 17, 2020 from https://towardsdatascience.com/softmax-function-simplified-714068bf8156

3. No Author. (2020). "Keras: The Python Deep Learning Library". Retrived on April 20, 2020 from https://keras.io

4. No Author. (April 16, 2020). "Skin Cancer Facts and Statistics: What you need to know". Retrieved on April 25 from https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/

5. No Author. (2020). "Statistics". Retrieved on April 19, 2020 from https://www.bcbsm.com/health-care-fraud/fraud-statistics.html

6. Sharma, Sagar. (September 6, 2017). "Activation Functions in Neural Netowrks". Retrieved on April 19, 20202 from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

7. The Skin Cancer Foundation. (2020). "Basal Cell Carcinoma Overview". Retrieved on April 24 from https://www.skincancer.org/skin-cancer-information/basal-cell-carcinoma/

8. Tschandl, P., Rosendahl, C., Kittler, H. "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions". Retrieved on April 27 from https://www.ncbi.nlm.nih.gov/pubmed/30106392

9. Urban, Gregor, et al. (Oct. 2018). "Deep learning Localizes and Identifies Polyps in Real Time With 96% Accuracy in Screening Colonoscopy". October 2018, vol 155 Issue 4. Pages 1069-1078. Retrieved on April 20, 2020 from https://www.gastrojournal.org/article/S0016-5085(18)34659-6/fulltext

10. Uniqtech. (March 5, 2020). "Understand the history and evolution of Tensorflow by revisiting Tensorflow 1.0 Part 1". Retrieved on April 21 from https://medium.com/data-science-bootcamp/understand-the-history-and-evolution-of-tensorflow-by-revisiting-tensorflow-1-0-part-1-247cff27a9c2

11. Yegalulp, Serdar. (June 2020). "What is TensorFlow? The machine learning library explained". Retrieved on April 20, 2020 from https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html

Appendix: Skin Lesion Diagnoses Tool Cloud Architecture