HIGH SCHOOL EDITION
Journal of Student Research
Volume 10 Issue 3 (2021)

Optimization of a Convolutional Neural Network for the Automated Diagnosis of Melanoma

Kemka Ihemelandu[1] and Chukwuemeka Ihemelandu[#]

[1]McDonogh School, Owings Mills, MD, USA
[#]Advisor

ABSTRACT

The incidence of melanoma has been increasing rapidly over the past two decades, making melanoma a current public health crisis. Unfortunately, even as screening efforts continue to expand in an effort to ameliorate the death rate from melanoma, there is a need to improve diagnostic accuracy to decrease misdiagnosis. Artificial intelligence (AI) a new frontier in patient care has the ability to improve the accuracy of melanoma diagnosis. Convolutional neural network (CNN) a form of deep neural network, most commonly applied to analyze visual imagery, has been shown to outperform the human brain in pattern recognition. However, there are noted limitations with the accuracy of the CNN models. Our aim in this study was the optimization of convolutional neural network algorithms for the automated diagnosis of melanoma. We hypothesized that Optimal selection of the momentum and batch hyperparameter increases model accuracy. Our most successful model developed during this study, showed that optimal selection of momentum of 0.25, batch size of 2, led to a superior performance and a faster model training time, with an accuracy of ~ 83% after nine hours of training. We did notice a lack of diversity in the dataset used, with a noted class imbalance favoring lighter vs. darker skin tone. Training set image transformations did not result in a superior model performance in our study.

Introduction

Skin cancer is estimated by the American Cancer Society to be the most predominant of all cancers, with melanoma accounting for ~ 1% of all skin cancers but resulting in a greater proportion of skin cancer related deaths[1]. Melanoma is defined by the National Cancer Institute as a "form of cancer that begins in melanocytes (cells that make the pigment melanin), and may arise in a mole"[2]. Approximately, 106,110 new cases of melanoma are anticipated to be diagnosed in the USA for the year 2021, with ~ 7,180 associated deaths[2]. There has been a noted rapid increase in the incidence of melanoma over the past two decades thought to be fueled by ultra violet radiation and global warming, making melanoma a current public health crisis. Unfortunately, even as screening efforts continue to expand in an effort to ameliorate the death rate from melanoma, there is a need to improve diagnostic accuracy to decrease misdiagnosis. Presently histopathologic analysis of surgical specimen following excision remains the gold standard for diagnosis of melanoma, which unfortunately has documented limitations including; its invasiveness, and discordance rates of up to 15% among pathologists[3]. As such improving diagnostic accuracy to decrease misdiagnosis is urgently needed to help decrease morbidity and increase overall survival[4]. Artificial intelligence (AI) a new frontier in patient care has the ability to improve the accuracy of melanoma diagnosis. Machine Learning (ML) is a form of AI in which the computer is not specifically programmed to perform a specific task but rather learns repeatedly to make predictions or decisions. Deep Learning (DL) a form of ML uses artificial neural networks modeled after the human brain to learn from large amounts of data. Convolutional neural network (CNN) is a form of deep neural network, most commonly applied to analyze visual imagery[5]. Convolutional neural networks have been shown to outperform the human brain in pattern recognition. AI will be extremely useful in speed and efficiency of melanoma diagnosis especially in communities with limited access to healthcare. However, there are noted limitations with the accuracy of the models used

to test. In this study our aim is the optimization of convolutional neural network algorithms for automated diagnosis of melanoma, we hypothesize that optimal selection of the momentum and batch hyperparameter increases model accuracy.

Methods and Material

Supervised machine learning which involves learning a task that maps an input to an outcome on the basis of a previous sample input-outcome result[6] will be used for this study. Our goal is to develop a model, which is able to very efficiently and accurately differentiate between dermatoscopic images of melanoma and non-melanoma (malignant and benign) skin lesions. We constructed the neural networks for this study using the torch.nn package. This project was carried out using a Dell Inspiron 3880 desktop computer with an intel core i5-10400 processor. We used Python for our modelling. Other computing platforms used include: Anacondia, sublime text, Scikit-Learn, TensorFlow, pyTorch. "Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands (https://docs.anaconda.com/anaconda/navigator/)". Navigator allows you to find the packages you want, install them in an environment, run the packages, and update them. We then accessed TensorBoard: TensorFlow's visualization toolkit (https://www.tensorflow.org/tensorboard) using anaconda navigator. TensorBoard provides you with the visualization and tooling needed for machine learning modelling. Within anaconda, a version of the base (root) python environment was installed which accessed the terminal. Sublime Text was then installed (https://www.sublimetext.com/). Sublime Text is a shareware cross-platform source code editor with a Python application programming interface, which we used to write the code. GitHub was used to host our code to help keep track of progress and changes to the code. PyTorch (https://pytorch.org/) an open source machine learning library based on the Torch library, and used for applications such as computer vision and natural language processing was imported into the beginning of the code. Scikit-learn a software machine learning library for the Python programming language (https://scikit-learn.org/stable/) was also downloaded.

Data Collection

The dataset titled DermMel which consists of melanoma and benign dermatoscopic images: "https://www.kaggle.com/ghulammujtaba000/melanoma-and-benign-images-only" was retrieved from Kaggle.com. This dataset DermMel has three directories: test with a total of 1780 images, train with a total 5341 images and a validation directory with a total of 1781 images. The images were resized to 224 x 224 shape as our model requires an image shape of this size. This was accomplished using the Keras Image Data Generator. Transformations were carried out using Pillow (PIL), a Python image library. PIL has an image module which can open, display, rotate, flip and crop images in Python. Data Augmentation helps to increase the volume of a training set. Models trained with data augmentation are thought to generalize better.

Machine learning model

CNN was chosen as the most appropriate model for our project, as CNN in addition to not requiring image pre-processing techniques, obviates the need to manually extract features[7] [8]. Using torch.nn package we constructed neural networks. An nn.Module contains layers, and a method forward(input) that returns the output (https://pytorch.org/tutorials/beginner/blitz/neural_networks_tutorial.html). The protocol for training a neural network involves: defining the neural network, recapitulating over a dataset of inputs, analyzing input through the network, calculating the loss (cost) function (to what degree does the output differ from the input), processing the observed gradients back into the network's variables, updating the variables of the network, using a simple update rule: weight = weight - learning_rate

* gradient. Gradient descent utilizes the cost function to optimize the model during training. The cost function takes the aggregate of the variances of the distance between the model prediction and real values for training set data points: $J(\theta) = \frac{1}{2D}\sum_{i=1}^{D} (y_i - h(x_i))^2$. The cost function may be used to determine the accuracy of the model and how to change the parameters of the model. The number of training data points is represented by D indexed by i, the training input features by vector x_i, output features by y_i, and predictive model represented by h(). Gradient descent can be used to find the best performing model as measured by the cost function. Using equation: $\theta_j := \theta_j - \alpha \frac{\delta J(\theta)}{\delta \theta_j}$, we can iteratively minimize the cost function, to help determine the parameters which optimize model performance. Θ represents the parameters and j is an index over the parameters. There can be multiple parameters and j will be replaced in correspondence of the wanted parameter. The α represents the learning rate, and $\frac{\delta J(\theta)}{\delta \theta_j}$ represents the slope of the cost function with respect to parameter j. Supervised machine learning is used to minimize the cost function, as a minimized cost function reflects a model which is making predictions considered accurate in relation to the training set. Following finding of the minimal cost function, it can be used to determine performance and identify the best trained model for the training set. We began by randomly initializing our model parameters, following this we applied iteration of gradient descent. The gradient descent was used to find the gradient of the cost function which we used to determine how to change the model parameters to reach the minimal cost function. We set a learning rate to determine the size of each step the gradient descent takes towards the vertex in intervals until it reached the minimum cost function. A value for the learning rate was chosen keeping in mind that if the learning rate is too high, the gradient descent may take too far of a step and miss the minimal cost function, and if the learning rate is too low, the size of the step will be too small, and the model will take too long to reach the minimum cost function. A good learning rate takes decent sized steps in an efficient amount of time. Gradient descent calculates the slope of the cost function at the current parameter values and uses the slope to iteratively change the parameter values. If the parameter is at the position where the line is decreasing from left to right, the point has a positive slope. If the parameter is at a position where the line is increasing from left to right, the point has a negative point. If a model has many input features, those features may have a diverse range of values this means that gradient descent may operate slowly due to the variation in range. Normalization can be used to solve this problem. Using two columns of data and taking the first number of the parameter and dividing it with every number in the column helps the range become the same for each parameter, and helps the gradient descent to take easier steps to reach the minimum value.

Finding the optimal hyperparameters

We explored different hyperparameter settings to improve our model. These included: momentum, batch size and transformation. Optimizing a CNN involves a number of steps including: manipulating the learning rate, choosing an optimizer and a loss function, identifying the batch size and number of epochs.

Adding Momentum

Momentum is designed to speed up the optimization process for Stochastic Gradient Descent (SGD) which thereby reduces instability in the model and overfitting. Momentum values of 0.1, 0.25, 0.5, 0.75, and 0.9 were tested while setting the batch size to 5 and the learning rate to 0.001. Momentum calculates speed in a certain direction, and it determines how fast the model is being trained, also accounting for stochastic gradient descent. The speed and efficiency in which the model is trained can also be determined by the learning rate[9].

Batch Size

Batch size is known to affect model performance, with larger batch sizes tending to reduce performance. We chose a smaller batch size range 2, 4, 6 and 8, from which to determine how many pictures the model should train with at each iteration of stochastic gradient descent. The general idea is that the lower the batch size, the faster the model would train, however there would be fewer images in each iteration to train from. We hypothesized that batch size would be critical in influencing model performance.

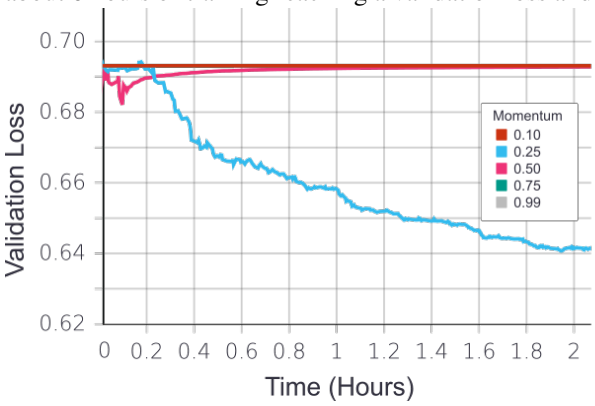
The model was trained twelve times while comparing two main parameters: the momentum and the batch size. While training the model, comparisons were done following performance of different transformations. The learning rate was kept constant at 0.001. The momentum and the batch size effect on the gradient descent was observed. Also, effect of transformations on the training model was noted.

Model Performance Visualization

The tensorboard was used to visualize how well the model was training, displaying two different graphs of loss and accuracy. Since the pictures were split into two categories, training and validation, there was a single loss and accuracy graph for each category (Figure 1, 2). The x-axis displayed the time, in hours, it took for the model to train with its set parameters and the y-axis displayed the cost function. The loss graph is calculating the minimal value of the cost function while the accuracy is calculating how often the model guesses if the picture has melanoma correctly. With the tensorboard graphs, the ideal visualization from the training should be that the loss graph is gradually decreasing while the accuracy is increasing.

Results

Final training involved training the full dataset, using the optimal hyperparameters and transfer learning methods derived from the subset of images. The optimal hyperparameters found were a learning rate of 0.001, with momentum of 0.25, batch size of 2 and fine tuning on the entire architecture. The final dataset was split into three subsets: Training, validation and testing. As additional data increases a model's ability to converge and reduces overfitting, it was hoped that significant improvements in model performance would be observed. The CNN model was trained with 50 epochs. Our study showed that optimal selection of momentum increased accuracy by 9.7%. For the first set of experiments, the learning rate was kept as the control at 0.001 and a random batch size of 5 was chosen. Five different values for the momentum hyperparameter were tested, starting from 0.10 to 0.99, slightly increasing each time to see what momentum would provide the best results within the validation cross entropy loss function and the validation accuracy. Out of the five trials, the best momentum was proven to be 0.25 providing a loss and accuracy of 0.64 and 65% respectively after 2 hours of training (Fig. 1). The superior performance at a momentum of 0.25 was consistent after about 6 hours of training reaching a validation loss and accuracy of 0.63 and 69% respectively.



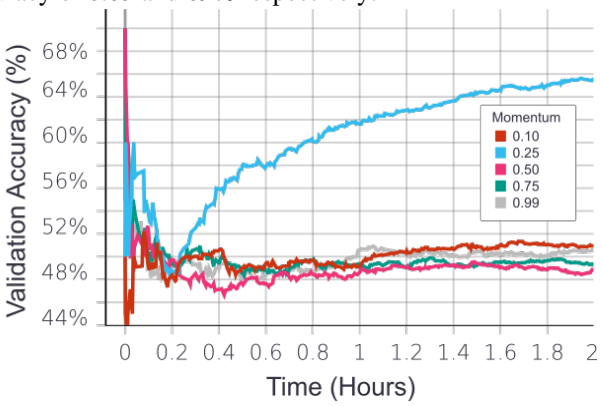


Figure 1. Model cross-entropy loss and accuracy across the range of momentum parameter values tested. The model shows that the momentum of 0.25 (the blue line) shows a better performance than the other momentums reaching a cross-entropy loss of 0.64 and an accuracy of 64%.

Our study also showed that a lower batch size leads to superior performance and a faster model training time. For the second set of experiments, the learning rate continued to stay as the control at 0.001. The momentum was set as 0.25, based on this value having the best performance in the first set of experiments. Four different batch sizes, ranging from 2 to 8 were tested at an incremental value of 2. Out of the four trials that were done, the best batch size was proven to be 2 with a loss function of 0.63 and a validation accuracy of 66% (Fig. 2). The performance of a batch size of 2 was continuously superior to the rest, running for 9 hours and having an overall loss function of 0.45 and an accuracy of 83%.

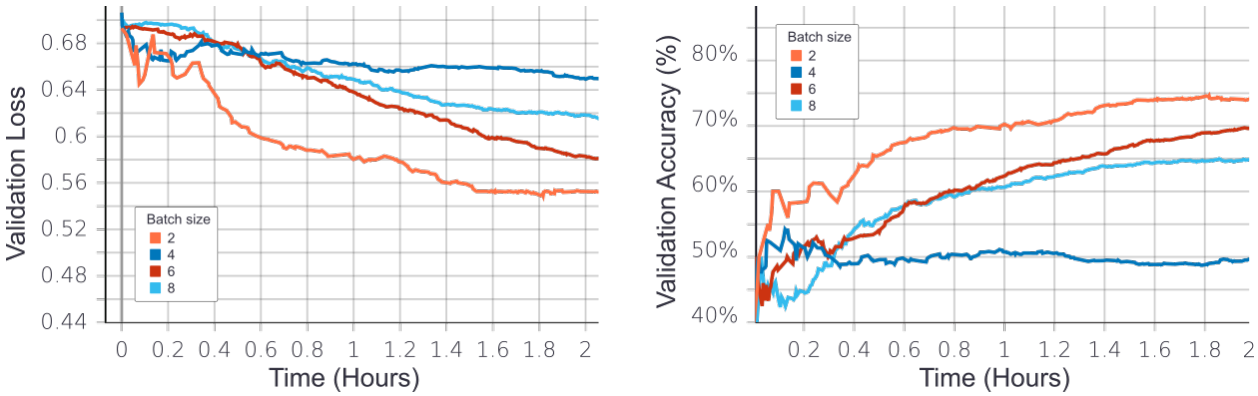


Figure 2. Model of validation loss and accuracy tested in iterations of batch sizes from two to eight.
Training the model shows that the batch size of two had the best performance out of the four batch sizes tested, reaching a cross-entropy loss of 0.55 and an accuracy of 75%.

Interestingly, our study documented that training set image transformations did not result in superior model performance. We ran two experiments with horizontal and vertical transformations, we set the parameters as the best outcome from the previous experiments: a LR of 0.001, momentum of 0.25 and a batch size of 2. We then set the control as the best trial without the transformation and used the different validation loss functions and validation accuracy percentages to compare and contrast if the transformation truly helped the model's performance. With the horizontal transformation, the validation loss function was 0.68 and the validation accuracy was 51%. With the vertical transformation, the validation loss was 0.66 and the validation was 50%. Compared to the experiment without the transformation, the validation loss function decreased while the validation accuracy dropped to an average of 27%.

Discussion

This study shows that machine learning has the ability to efficiently and accurately detect melanoma. In this study our aim was the optimization of a convolutional neural network algorithm for automated diagnosis of melanoma, we hypothesized that optimal selection of the momentum and batch hyperparameter increases model accuracy. Our most successful model developed during this study, showed that optimal selection of momentum of 0.25, batch size of 2, led to a superior performance and a faster model training time, with an accuracy of ~ 83% after nine hours of training. Training set image transformations did not result in a superior model performance in our study.

Being able to achieve an accuracy level of this magnitude, AI definitely lends its self as a diagnostic tool. Our model exemplifies the potential role of AI in dermatology and healthcare in general. According to a study by Urbancek et al, from a total of 936 histologicaly confirmed melanomas, 150 (16%) were diagnosed incorrectly [10]. Machine learning models such as demonstrated in our current study, can help lower the percentage of misdiagnosis and can even be developed to detect melanoma more rapidly than the original method of manual diagnosis. Machine learning can advance detection by allowing patients to do it in their own homes. With the help of the DermLite HUD, a dermatoscope camera which can be attached to a mobile device. If this model was available to the public and more developed and trained, patients could then detect melanoma at their house, and then take it to the doctors. Machine learning is the beginning of building a better and more developed medical world.

Conclusion

Optimal selection of the momentum and batch hyperparameter increases model accuracy. Our most successful model developed during this study, showed that optimal selection of momentum of 0.25, batch size of 2, led to a superior performance and a faster model training time, with an accuracy of ~ 83% after nine hours of training. We did notice a lack of diversity in the dataset used, with a noted class imbalance favoring lighter vs. darker skin tone. Training set image transformations did not result in a superior model performance in our study

Limitations

Include a small dataset with noted class imbalance of light vs. dark skin.

Acknowledgments

I would first like to thank my family for believing in me and being my biggest supporters. Your endless encouragement made this possible. I would like to acknowledge Dr. Parsa Akbar for guiding me through the research process. A special thanks to Mr. Thompson at McDonogh School for his continuous support and mentorship.

References

[1] https://www.cancer.org/cancer/melanoma-skin-cancer/about/key-statistics.html

[2] https://www.cancer.gov/publications/dictionaries/cancer-terms/def/melanoma

[3] Elmore, J. G., Barnhill, R. L., Elder, D. E., Longton, G. M., Pepe, M. S., Reisch, L. M., Carney, P. A., Titus, L. J., Nelson, H. D., Onega, T., Tosteson, A., Weinstock, M. A., Knezevich, S. R., & Piepkorn, M. W. (2017). Pathologists' diagnosis of invasive melanoma and melanocytic proliferations: observer accuracy and reproducibility study. BMJ (Clinical research ed.), 357, j2813. https://doi.org/10.1136/bmj.j2813.

[4] Merlino, G., Herlyn, M., Fisher, D. E., Bastian, B. C., Flaherty, K. T., Davies, M. A., Wargo, J. A., Curiel-Lewandrowski, C., Weber, M. J., Leachman, S. A., Soengas, M. S., McMahon, M., Harbour, J. W., Swetter, S. M., Aplin, A. E., Atkins, M. B., Bosenberg, M. W., Dummer, R., Gershenwald, J. E., Halpern, A. C., … Ronai, Z. A. (2016). The state of melanoma: challenges and opportunities. Pigment cell & melanoma research, 29(4), 404–416. https://doi.org/10.1111/pcmr.12475

[5] Valueva, M.V.; Nagornov, N.N.; Lyakhov, P.A., Valuev G.V., Chervyakov N.I. (2020). Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. Mathematics and Computers in Simulation. Elsevier BV, 177: 232–243.

[6] Stuart J. Russell, Peter Norvig (2010) Artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall ISBN 9780136042594

[7] Doaa A. Shoieb, Sherin M. Youssef, and Walid M. Aly, (2016) "Computer-Aided Model for Skin Diagnosis Using Deep Learning," Journal of Image and Graphics, Vol. 4, No. 2, pp. 122-129. doi: 10.18178/joig.4.2.122-129

[8] Kieffer, B., Babaie, M., Kalra, S. and Tizhoosh, H.R. (2017). Convolutional Neural Networks for Histopathology Image Classification: Training vs. Using Pre-Trained Networks. 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA).

[9] Chee, Jerry, and Ping Li. 2020. “Understanding and Detecting Convergence for Stochastic Gradient Descent with Momentum.” arXiv [cs.LG]. arXiv. http://arxiv.org/abs/2008.12224.

[10] Urbancek S, Fedorcova P, Tomkova J, Sutka R (2015) Misdiagnosis of Melanoma: A 7 Year Single-Center Analysis. Pigmentary Disorders 2: 208. doi:10.4172/2376-0427.1000208