DeepDerm: Detection of Cancerous Skin Lesions Through Deep Learning

Santosh Murugan
smurugan@stanford.edu

Anna Verwillow
anna19@stanford.edu

1 Introduction

With over 17 million new cases and almost 10 million deaths per year, cancer is one of the deadliest diseases in the world. [1] Although funding for cancer-related initiatives has steadily increased in recent years, many oncological problems - whether in prevention, diagnosis, or therapeutics - remain largely intractable. In the past, lack of access to genetic and high-quality imaging data (crucial for genetic-based, visually-presenting diseases such as various types of cancer) was a significant rate-limiting step, but as genomic sequencing and medical imaging technology has improved, researchers are faced with a new problem: harnessing these massive quantities of multi-dimensional data to improve clinical decision-making.

Machine learning (ML) and Artificial Intelligence (AI) techniques hold great promise for clinical data analysis of this nature. Machine learning models have historically been used for image classification, sequence prediction, and natural language processing in applications from self-driving cars to climate change. Within healthcare, machine learning has found applications in Electronic Health Records analysis and cancer imaging. These techniques often demonstrate tremendous success, even outperforming human experts.

In this study, we focus on the automated classification of skin cancers. Skin cancer is the most common form of cancer, affecting approximately 20% of all Americans by age 70. More importantly, however, many forms are highly preventable if caught early. Through early detection, the 5 year survival rate of the melanoma, the most deadly form, can be up to 99%; however, delayed diagnosis causes the survival rate to drop dramatically to 23%. [2] This detection is generally performed by pathologists and dermatologists, but is not perfect. We implement several deep learning (DL) models that automatically classify cancerous (basal cell carcinomas and melanomas) vs. non-cancerous (Bowen's, benign keratosis-like lesions, dermatofibromas, melanocytic nevi, and vascular) skin lesions with high accuracy, and can be used to guide clinical decision-making. As the technology improves to match or surpass an individual practitioner, employing automated lesion classification will allow medical professionals to spend more time with patients.

[Project code supplied upon request to smurugan@stanford.edu]

2 Related Work

The International Skin Imaging Collaboration (ISIC) proposed skin cancer detection challenges in 2016 [3], 2017 [4], and 2018 [5], resulting in over 1000 registrations and 300 submissions to the disease classification task. 2016 had only 900 images for binary classification into "benign" or "malignant"; 2017 segmented into three labels: "melanoma" (374 training, 30 validation, 117 test), "seborrheic keratosis" (254, 42, and 90), and "benign nevi" (1372, 78, 393); 2018: 10,015 dermoscopic images with 7 ground truth classification labels. The increasing power of increased data lead to the highest scoring model in 2018 with an AUC of 0.885, though the architectures of the submitted algorithms were not published. This is the largest, standardized data set of images to date.

Additional studies have built deep learning models for skin cancer detection and classification. [6,7,8,9] Most, however, are under-powered with few images to train on.

The study with the highest AUC to date by Esteva et al scored in the mid 90s, 94%-96% depending on the sub-problem. [6] They trained a CNN over 125,000 images, the largest data set to date. The opted for two binary classification problems rather than multiclass, meaning they got to pool their data into fewer problems: keratinocyte carcinomas versus benign seborrheic keratoses (identification of the most common cancers) and malignant melanomas versus benign nevi (identification of the deadliest skin cancer). This level of classification outperforms the average of 21 board-certified dermatologists.

3 Dataset and Features

3.1 Features

We are using the Skin Cancer MNIST dataset, available from Kaggle and published in Nature [10]. The dataset contains 10,015 images of skin lesions, covering seven of the major subcategories (labels). The dataset was curated in 2018, and there is one lesion per image. There are 10 columns in the dataset, corresponding to the disease class and sub-class, age and sex of patient, localization of the lesion, and the corresponding image name.

3.2 Data Preprocessing

We implemented a data ingestion pipeline which integrates Google Drive (where our data repository resides) with Google CoLaboratory (where our machine learning models are implemented). This pipeline makes use of Pandas dataframes to keep track of non-image features along with the file paths for the actual images.

Following the data loading, we removed certain columns with troublesome data formats (e.g. mostly NULL values), performed data augmentation techniques to increase the diversity of images in our dataset using Keras's built-in Image Data Generator module, and resized the input images to fit the required default image sizes for our models (299x299 for InceptionV3, 244x244 for VGG19, and ResNet50, and 75x100 for our *de novo* model). In the process of doing so, we ended up needing to remove three classes of skin lesions present in the original dataset due to limited compute resources so as to avoid memory overallocation errors. We also did not use one-hot encodings and instead opted for training with Keras's sparse categorical crossentropy metric.

Lastly, we utilized scikit-learn to perform a 70:10:20 training-validation-test split of our data to evaluate performance. While we considered implementing K-Fold validation, given the size of our dataset, we felt a more straightforward split would be yield relatively high-fidelity results.

4 Methods

All five of the following model architectures were written in Keras with a TensorFlow (version 1.15.0) backend.

4.1 Baseline Architecture

Following the data pre-processing, we set out creating our baseline model. The basic model architecture is a Convolutional Neural Network with a Conv2D layer (32 neurons, 3x3 kernels with "same" padding and ReLU activation), followed by Max Pooling, Flatten, and Dense layers. Because this is multi-class classification, the final dense layer was equipped with softmax activation. We used an Adam optimizer (learning rate: 1e-3, betas: 0.9-0.999, no decay), and categorical cross-entropy loss. We then fit the model (10 epochs, with a batch size of 32), and evaluated it with respect to our metrics.

4.2 Applied Architectures

While the baseline architecture performed better than chance, we decided to leverage existing image-classification architectures in the hopes of improving performance. Implementing these models in Keras further required resizing the input images, as well as modifying each architecture's output (Dense) layer to suit our task.

4.2.1 ResNet-50

ResNet-50 is a fifty layer convolutional neural network which has been deployed for various image classification tasks. [3] By employing a framework called residual learning, architectures of this kind can often stack more layers than most others, often leading to significant improvements in performance. During training, the layer-by-layer weights for ResNet-50 can either be randomly initialized or initialized with pre-trained values from large visual datasets such as ImageNet. In cases where the number of neurons in the output layer is not 1000 (by default), however, Keras does not allow the use of pre-trained ImageNet weights. Thus, because we required 4 output neurons (one for each class), we chose to randomly initialize the layer weights, and attempted to mitigate the lack of pre-training with a bigger train-test split of the dataset.

4.2.2 VGG19

VGG19, developed by Simonyan et al. at the University of Oxford, represents a model architecture with 19 weight layers. Its primary advantage is derived from the stacking of convolutional layers with smaller (3x3) kernels than previous architectures. The convolutional layers then feed into dense layers, including a final dense layer with softmax activation for multiclass classification. [4] With regard to the Keras implementation, we encountered the same issue as in ResNet-50 wherein we ended up training randomly initialized layer weights as opposed to the pre-trained ImageNet weights. For hyperparameter tuning, we encountered very slow model training when using a batch size of 32, and extremely high RAM usage when using a batch size of 512. While this may be acceptable in a research setting, this particular hyperparameter setting would likely pose an obstacle when scaling to even larger datasets or in clinical settings. Thus, we chose to set the batch size to 256, and report the results here.

4.2.3 InceptionV3

The Inception architectures, developed by Szegedy et al. from Google, are comprised of layers traditionally found in many neural networks (max- and average-pooling, convolutional, dense with softmax activation), as well as "Inception" modules. Whereas in other models it might be difficult to determine the appropriate kernel dimensions for a particular convolutional layer, Inception modules effectively try multiple kernels of distinct dimensions and feed this concatenated input to the next layer. Additional implementation nuances present in Version 3 allow for a reduction in the total number of model parameters, thus reducing computational cost associated with training and test, which is particularly important for our application. [5] We used randomly initialized weights, and four output neurons.

4.3 Metrics

We evaluated the network with regard to accuracy and "area under the receiver-operating curve" (AUROC). AUROC can often be helpful in cases where the dataset has high class imbalance. Unfortunately, scikit-learn does not come pre-equipped with an AUROC metric for multiclass

classification. We wrote an implementation which performs one-vs-all AUROC for each of the classes, and report the AUROC averaged across all classes. Further, the included confusion matrices provide more granular, class-by-class metrics.

5 Results and Discussion

5.1 By Model

We assembled confusion matrices for each of the four models. Columns represent the actual class; rows represent the predicted class. Each cell entry represents the number of classifications. The diagonal represents those that were classified correctly and numbers off the diagonal signal misclassification.

Each of the models learned to classify into only a subset of the labels, resulting in a number of off-target predictions. This phenomenon, and classification as a whole, would improve with further model training.

As mentioned earlier, in the project as a whole we focused on four of the seven data labels due to limited computational ability. For VGG19, we further removed two labels because the larger model continued with errors for our kernel size, but still trained a model on two labels for the sake of comparison.

Initial	0	1	2	3
0	42	31	0	0
1	25	67	0	8
2	139	54	16	6
3	13	3	4	3

Figure 1: Confusion Matrix for Initial Model

ResNet	0	1	2	3
0	27	0	46	0
1	34	0	66	0
2	62	0	153	0
3	10	0	13	0

Figure 2: Confusion Matrix for ResNet

VGG19	0	1	2	3
0	0	58		
1	0	111		
2				
3				

Figure 3: Confusion Matrix for VGG19
Note: For the sake of training, because this became highly computationally expensive, we modified it to two classes

InceptionV3	0	1	2	3
0	27	0	73	0
1	34	0	100	0
2	62	0	215	0
3	10	0	23	0

Figure 4: Confusion Matrix for InceptionV3

5.2 Across the Models

We experimented with a number of different models of increasing complexity to address the high bias we were finding in our error results. We evaluated the models by classification accuracy on training and test sets as well as the AUC found by the method described above. We found some benefit as the training and test errors did both improve with complexity.

Table 1: Comparison between Models Run

Algorithm	Training Accuracy	Test Accuracy	AUC
Initial	n/a	0.311	0.597
ResNet	.610	.438	0.541
VGG19	0.592	0.657	0.5
InceptionV3	0.640	0.523	0.443

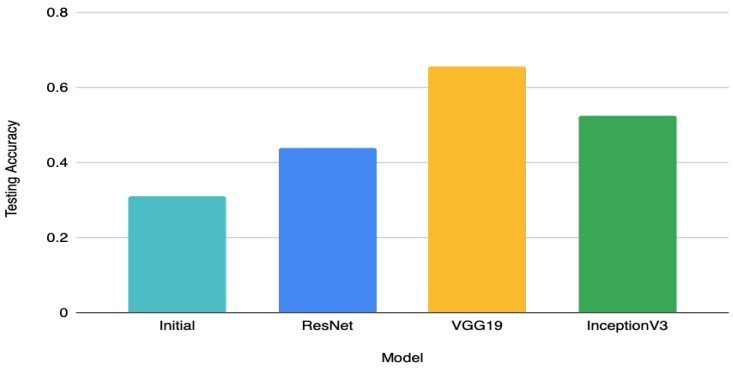

Figure 5: Test Error Results

6 Conclusions

6.1 Discussion

The training and testing errors are lower than previously published results by at least 20%, potentially demonstrating both a high bias and high variance problem. That being said, there were a number of differences between previously published work and the models trained within. For example, many of the models implemented in the literature turned the problem into a series of binary classification tasks, whereas our approach focused exclusively on a single multi-class classification task. Moreover, the sizes of the training sets in the literature often varied greatly from ours, with the best published model using almost 130,000 images. In contrast, our training strategy (approximately 1% of this size) was inherently more restricted.

With regard to test error, our best performing model was VGG19. This is surprising - of the models tried here (out of the 18 models available in Keras's Applications module), the VGG model is in the bottom 3 with regard to Top-1 and Top-5 accuracies on the ImageNet validation dataset. While it may be the case that ImageNet and our dataset simply favor different model architectures, it certainly indicates that more resources could be dedicated towards training and hyperparameter tuning to confirm these results.

Lastly, it is important to remember that accuracy and AUC are not the only factors one would considering when choosing a model to deploy. For example, in some clinical settings, speed of training is extremely important. A model such as VGG19 with approximately 140 million parameters takes orders of magnitude longer to train than our initial model, which is over 300x smaller. In such a case, one might be inclined to use a more lightweight model, as long as the difference in accuracy/AUC is non-significant. Ultimately, such a decision would need to be made in conjunction by doctors and ML experts.

6.2 Future Directions

First and foremost, with substantial additional computational power and time, we would hope to perform a more expansive hyperparameter search, and decrease what appears to be high bias and high variance problems.

To address the high bias, we could train each of these models for a larger number of epochs. To address the variance, we could add features from the metadata on each lesion (e.g. localization), which are collected from each patient. Another approach might be to source additional images, although the we would need to ensure that these images are distributed appropriately between the training and test sets.

Armed with these better tuned models, we would then hope to expand beyond this dataset towards similar image classfication tasks in the same domain. As an example, we could consider applying InceptionV3 or VGG19 towards identifying breast cancer nodules from mammograms. There is certainly no shortage of opportunities in this space!

7 Contributions

Santosh and Anna participated equally, including both the coding portion and project write-up.

References

[1] Worldwide cancer statistics. Cancer Research UK. https://www.cancerresearchuk.org/health-professional/cancer-statistics/worldwide-cancer

[2] Cancer Facts and Figures 2018. American Cancer Society. https://www.cancer.org/content/dam/cancer-org/research/cancer-facts-andstatistics/annual-cancer-facts-and-figures/2018/cancer-facts-and-figures-2018.pdf.

[3] Gutman D, et al. "Skin Lesion Analysis toward Melanoma Detection: A Challenge at the International Symposium on Biomedical Imaging (ISBI) 2016, hosted by the International Skin Imaging Collaboration (ISIC)". eprint arXiv:1605.01397. 2016.

[4] Codella, Noel, et al. "Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)." Submitted on 13 Oct 2017 (v1), last revised 8 Jan 2018 (this version, v3). eprint arXiv:1710.05006 [cs.CV]

[5] Codella, Noel, et al. "Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)." Submitted on 9 Feb 2019 (v1), last revised 29 Mar 2019 (this version, v2). eprint arXiv:1902.03368 [cs.CV]

[6] Esteva, Andre, et al. "Dermatologist-level classification of skin cancer with deep neural networks." Nature, Volume 542, pages 115–118(2017)
[7] Codella NCF, Nguyen B, Pankanti S, Gutman D, Helba B, Halpern A, Smith JR. "Deep learning ensembles for melanoma recognition in dermoscopy images" In: IBM Journal of Research and Development, vol. 61, no. 4/5, 2017.

[8] Diaz, I.G. "Incorporating the Knowledge of Dermatologists to Convolutional Neural Networks for the Diagnosis of Skin Lesions. 2017 International Symposium on Biomedical Imaging (ISBI) Challenge on Skin Lesion Analysis Towards Melanoma Detection." Available: https://arxiv.org/abs/1703.01976

[9] Masood, Ammara, et al. "Self-supervised learning model for skin cancer diagnosis." 2015 7th International IEEE/EMBS Conference on Neural Engineering (NER), 22-24 April 2015.

[10] Tschandl, Philipp, et al. "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions." Scientific Data, Volume 5, Article number: 180161 (2018).