

Automated Diagnosis of Pigmented Skin Lesions

Group 37 - Lucas Lin, Raul Pegan, Noopur Khachane, Louise Xu
University of California, San Diego
<https://github.com/raulpegan/ECE228-Group37>

June 2019

1 Abstract

In recent years, advances in computer vision have allowed many object classification tasks to be automated. We are especially interested in the application of object classification networks for automated diagnosis of skin lesions, since skin cancer is the most common type of cancer. There currently exist many publications that detail different methods for skin lesion classification, but it is difficult to fairly compare their effectiveness since they use different datasets, classes, evaluation metrics and so on. In this paper, we hope to provide a fair system to evaluate different CNN architectures and learning methods by using a common dataset.

2 Introduction

Cancer is the second leading cause of death in the United States, with skin cancer as one of the most common. Skin cancer itself will not cause death, but if left undiagnosed, can spread to other organs eventually leading to death. [5] However, early detection of dangerous skin lesions, such as melanoma, can prevent spreading and allow for more treatment options. A skin lesion is an abnormal lump, bump, ulcer, sore or colored area on the skin. The current detection methods of skin lesions include biopsy, imaging, and blood test. [1] These methods require expensive equipment and for patients to visit a dermatologist at a clinic. In underdeveloped regions, traveling to a clinic may prove burdensome. The cost, stigma, and accessibility of these test methods may limit patients' willingness to seek help. This has led to the research of many deep learning methods to classify the most common types of skin lesions from photographic images. The convenience that comes from being able to take a picture and know immediately whether a skin lesion is concerning would greatly increase the

early detection rate. A complication that arises with the evaluation of new skin lesion classification models is that the use of different databases and metrics makes them difficult to compare [3]. Therefore, we propose a comparison of several of the best performing models on the HAM1000 dataset. The input to our algorithm will be RGB images of skin lesions. We will then use various neural networks to output a predicted diagnosis. We then used a variety of network architectures for the task, including VGG [12], ResNet [6], and Inception [14].

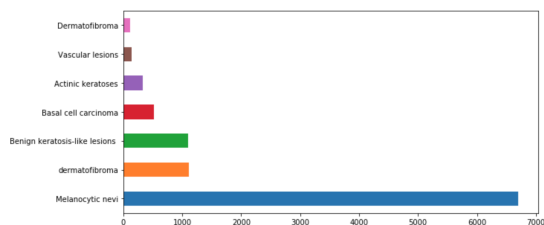
3 Related Work

We found the survey "Skin Cancer Classification Using Convolutional Neural Networks: Systematic Review" which provided an overview of various existing methods using CNNs for automated skin lesion diagnosis [3]. The methods reviewed were divided into two categories: CNN as a feature extractor and end-to-end learning. The end-to-end learning category was then subdivided into learning from scratch and transfer learning. Due to the limitations in our computing resources, we chose to focus on the transfer learning and feature extractor methods. Among these methods, we chose to experiment with VGG, ResNet, and InceptionNet for transfer learning, and a combination of InceptionNet with kNN, K-means, and SVM for CNN as a feature extractor.

As the authors of the paper noted, it is not easy to compare the methods from the original papers since they were trained on datasets of varying sizes and evaluated with different metrics. In order to fairly evaluate the performance of these different methods, we used the same HAM10000 dataset with each of our CNNs.

4 Dataset and Features

The HAM10000 dataset is a subset of the ISIC Skin Lesion dataset, containing 10015 images of seven different classes of skin lesions.[10] The seven classes are actinic keratoses (akiec), basal cell carcinoma (bcc), benign keratosis-like lesions (bkl), dermatofibroma (df), melanoma (mel), melanocytic nevi (nv), and vascular lesions (vasc). In addition to the images of the lesions themselves, the dataset also contains metadata about each patient such as their age or the location of each skin lesion. A disadvantage of this dataset is the sample imbalance between classes; the highest occurring class, nv, represents 66.9% of the data while the lowest occurring class, df, only represents 1.1% of the data. This causes bias when training the data. This bias is particularly insidious since it will make the overfitted model make false negatives occur more often, a problem that has a higher negative impact in the use case of cancer detection.



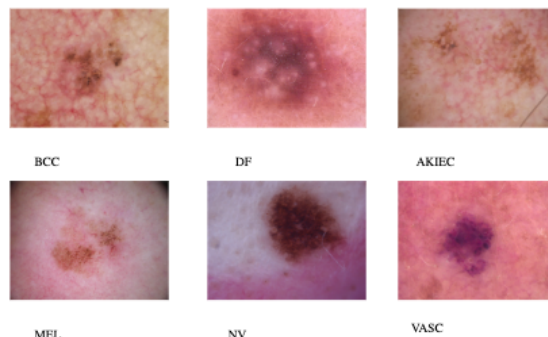
	lesion_id	image_id	dx	dx_type	age	sex	localization
0	HAM_0003559	ISIC_0025810	bkl	consensus	75.0	male	face
1	HAM_0004240	ISIC_0032517	mel	histo	50.0	male	back
2	HAM_0002610	ISIC_0026876	vasc	consensus	0.0	female	abdomen
3	HAM_0003229	ISIC_0031140	bcc	histo	60.0	male	chest
4	HAM_0005963	ISIC_0028129	nv	follow_up	45.0	male	foot

To mitigate the effects of imbalance, we initially applied a combination of oversampling the under-represented data and undersampling the over-represented data so all classes would have roughly the same amount of samples. We also found alternative success in using sample weights to increase the importance of undersampled data. Additionally, image augmentation was introduced to avoid overfitting the data. As a result, preprocessing each image included rescaling RGB values, random shearing, random horizontal flipping, and random zoom.

To divide the dataset, we used an 80/20 train-test split and further divided the training set into an 80/20 training-validation split. This amounts to 6408 training images, 1602 validation images, and 2003 test images. The models were not cross validated because the dataset is rather large and

would take a significant amount of time to train the model k times. The data splitting was achieved by making the training, validation, and test directories and class subdirectories. Then the metadata was shuffled and divided into the data split for randomization. Next the images were moved to their respective folders and fed into the networks using the "flow_from_directory" method of the DataGenerator class in keras.[4]

The features used for transfer learning were the images themselves, and features used for feature extraction were the output of the CNN. Sample images are shown here:



5 Methods

5.1 Transfer Learning

We used transfer learning for three convolutional neural network (CNN) models: Inception, VGG, and ResNet using the Keras library[4]. A CNN is a deep learning algorithm that takes in an image and applies different filters to extract spatial and temporal patterns. Deeper layers of the network are used to represent higher levels of feature abstraction and consequently allow the network to differentiate the images into classes. The weights of the CNN are updated through back-propagating the loss of training images.

5.1.1 Inception V3

Inception is a CNN introduced by Google in 2015 for the ImageNet Large Scale Visual Recognition Competition (ILSVRC). This model is advantageous because of its computational efficiency and low number of parameters. The third version of GoogLeNet is the model used by Esteva et al. in their 2017 skin lesion classification paper. This iteration of inception introduces factorization techniques in addition to auxiliary classifiers and grid

size reduction. Factorization is a parameter reduction method through the replacement of single large convolutions by multiple smaller convolutions. A result of parameter reduction, this 42 layer network has a decreased risk of over-fitting the data.[14]

5.1.2 VGG19

The VGG network was introduced in 2014 as a competitor for the ImageNet Large Scale Visual Recognition Competition (ILSVRC) by the Visual Geometry Group at Oxford University [12]. This network's approach to CNN is through the usage of small (3x3) convolutions, allowing the network to instead focus on creating a deeper structure rather than complex filters. This network had very good results while also generalizing well to other datasets, becoming a common network architecture for state-of-the-art convolutional neural network projects.

5.1.3 ResNet50

The ResNet CNN architecture was introduced by Microsoft Research in 2015 for the ILSVRC [6]. ResNet introduced the concept of skip connections, which allowed for much deeper networks by learning residual representation functions. These residual connections allow for information of lower levels of the network to impact the weights of the higher levels, allowing for small features in the image to have a noticeable impact in the deeper layers of the network.

5.1.4 InceptionResnet

While not a part of the original list of common pretrained networks used in the article, we decided to try some of the other networks that had a high accuracy on ImageNet. InceptionResnet was a network created in order to combine the benefit of residual connections, or skip connections that jump over certain layers, with the common network Inception to create a network which outperforms both Inception and Resnet on Imagenet.[13]

5.1.5 DenseNet201

This was another network which was not presented in the original article, but has comparable accuracy to the other networks on ImageNet. It contains much more connections than a traditional CNN because every layer is connected to all of its previous layers (utilizing similar concepts to ResNet) which

eliminates problems such as the vanishing gradient, and also benefits from re-using past features. [7]

5.2 CNN Feature Extraction

In addition to transfer learning, we explored the use of the first few layers of a CNN to extract the general features of the skin lesions. These early features are thought to represent general shapes and patterns. The output of the feature extractor is then used to train a machine learning model which will then be used to classify the images. The CNN we used was ResNet pretrained on imagenet because this model gave the best initial results for transfer learning. The machine learning models were chosen based on those successful in the literature and implemented with sklearn. [3][11]

5.2.1 K-Means

K-Means is an unsupervised machine learning model that creates a predetermined number clusters of the training data based on feature similarities between samples. Our implementation created 7 clusters for each skin lesion class. Then each cluster was assigned to a single class by the softmax of each cluster. If multiple subjects had the same softmax, then the lowest second softmax is used as a tiebreaker with the other cluster assigned to the second softmax class.[9]

5.2.2 K Nearest Neighbors

Our K Nearest Neighbors (KNN) model takes a sample and determines the five closest training samples in feature space said sample. The sample is then predicted to be most common class occurrence of the nearest neighbors. The features are taken from the output of the CNN.[8]

5.2.3 Support Vector Machine

A support vector machine classifier (SVC) creates optimal hyper-planes in the feature space to separate the different classes using a kernel function. For our studies, we used the radial basis function kernel which is a Gaussian based kernel because it matches the complexity of the data while maintaining a relatively efficient training time. The SVC is trained to achieve two goals. The first is to maximize the distance between the closest samples of each class, and the second is to minimize the number of misclassified samples.[2]

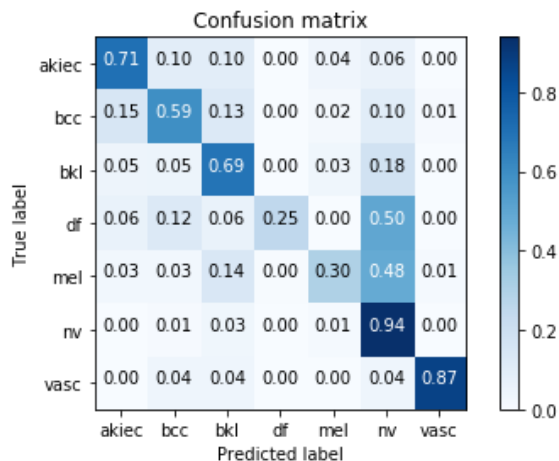
6 Experiments

We implemented a variable learning rate during model training to boost performance and decrease training time. This is a method that monitors the validation loss of the model and reduces the learning rate by one half when there has been no change in loss after two epochs. The variable learning rate allows for faster learning at the beginning of training while maintaining the fine tuning towards the end of training. When deciding on the initial learning rate for the models, we looked at the plot of training and validation loss at each epoch. The learning rate must be high enough to quickly decrease the loss, but low enough to not cause an elbow in the loss.

For performance evaluation, we looked at the overall accuracy of validation predictions and distributions in the confusion matrices. A good model will have high values along the diagonal and low random values for the off-diagonal.

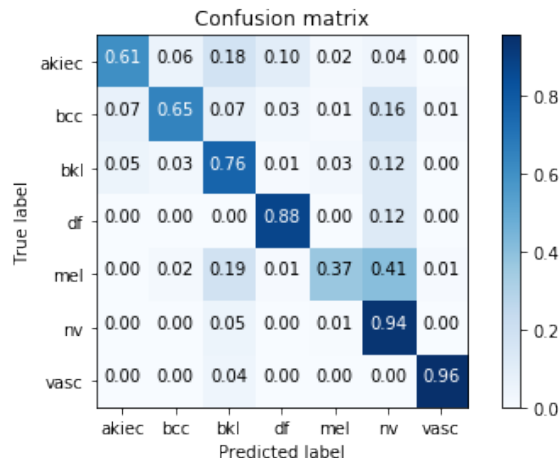
For batch size, we picked a large enough batch size to allow for fast training but not large enough to lose the granularity and precision. Additionally, we needed to ensure that the batch size would be small enough to fit in our available GPU resources, which was a 1080Ti. We settled on a batch size of 64.

The confusion matrix for ResNet after 10 epochs yielded decent results, as follows:

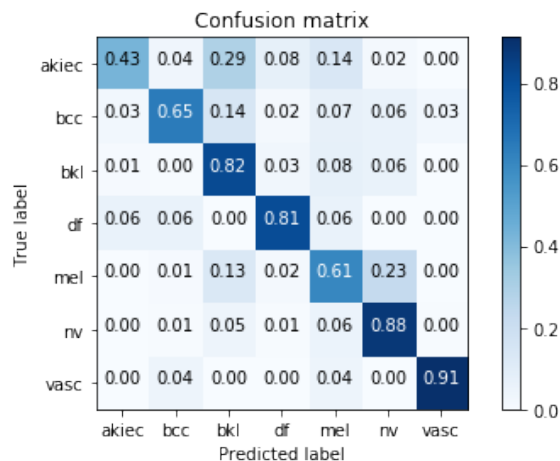


This network yielded a large amount of misclassified labels in the melanoma class, which is particularly hurtful. This model seems to be biased toward the nv class.

The confusion matrix for DenseNet after 10 epochs yielded good results, as follows:

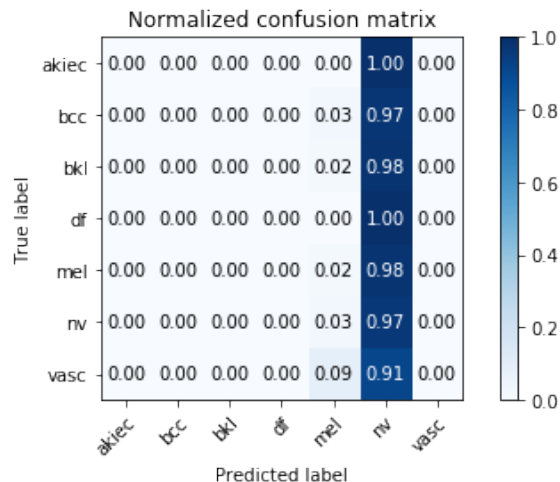


Most of the labels were correctly classified in the majority of cases. For the InceptionResNet model, the results were as follows after 10 epochs:



This model was more accurate for melanoma (mel) detection, making it a better choice when avoiding false negatives.

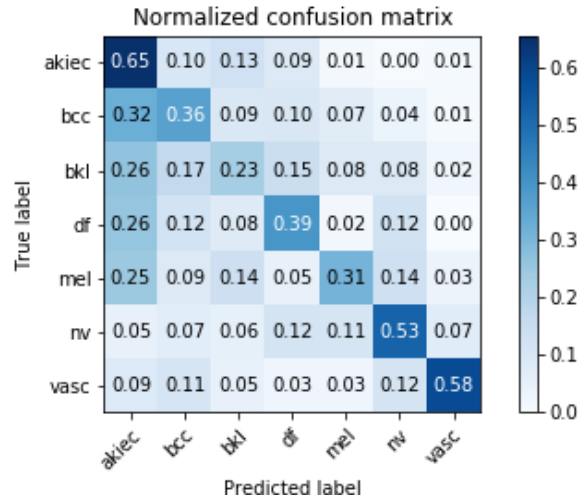
For VGG, the results were not good at all:



This leads us to believe that residual connections

have a very strong and useful impact for the network architecture of this problem, as all the previously discussed effective models use residual connections.

For feature extraction, we did not observe any particularly good results. Here is our attempt at utilizing knn:



Though the results seem to be pointing towards a diagonal in the confusion matrix, they are not nearly as effective as using a full CNN.

7 Conclusion

In this project we tackled a variety of different challenges within the world of CNNs. A unbalanced dataset, coupled with hard to detect features, as well as limited computing resources led us to pursue optimal forms of obtaining results. We found DenseNet and ResNet to have the best performance, and we believe this is due to the residual connections allowing for smaller details to impact the output of deeper layers. If we had more time we would want to explore working with a even larger dataset (perhaps using more data augmentation), and running more experiments on residual-like architectures as they seemed to outperform our other options.

8 Contributions

Lucas Lin contributed to preprocessing, inception, and transfer learning. Louise Xu contributed to preprocessing and resnet. Noopur Khachane contributed to inceptionresnet and densnet. Raul Pegan contributed to VGG and transfer learning.

The group dynamic consisted of frequent group meetings coupled with remote work.

References

- [1] Tests for melanoma skin cancer, May 2016.
- [2] Asa Ben-Hur, David Horn, Hava T. Siegelmann, and Vladimir Vapnik. Support vector clustering. *J. Mach. Learn. Res.*, 2:125–137, March 2002.
- [3] Titus Josef Brinker, Achim Hekler, Jochen Sven Utikal, Niels Grabe, Dirk Schadendorf, Joachim Klode, Carola Berking, Theresa Steeb, Alexander H Enk, Christof Von Kalle, and et al. Skin cancer classification using convolutional neural networks: Systematic review. *Journal of Medical Internet Research*, 20(10), 2018.
- [4] François Chollet et al. Keras. <https://keras.io>, 2015.
- [5] American Society for Dermatologic Surgery. Skin lesions.
- [6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
- [7] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.
- [8] J. M. Keller, M. R. Gray, and J. A. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(4):580–585, July 1985.
- [9] Stuart P. Lloyd. "least squares quantization in pcm.". *Information Theory, IEEE Transactions on 28.2*, pages 129–137, 1982.
- [10] Kevin Mader. Skin cancer mnist: Ham10000. Sep 2018.
- [11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [12] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv e-prints*, page arXiv:1409.1556, Sep 2014.

- [13] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. 2016.
- [14] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.