Skin Lesion Analysis Toward Melanoma Detection*

Mohit Agarwal[1], Nandita Damaraju [2] and Sahbi Chaieb[3]

***Abstract*— Melanoma is very aggressive form of skin cancer that starts out with lesions on the skin. Early detection is key to longer survival for people with an incidence of melanoma. Current techniques use manual, invasive procedures that are slow and inaccurate. Previous approaches have tried to automate this task with satisfactory results. There is a scope for harnessing the power of deep learning techniques to get a better performance. The approach involves three tasks, Segmentation, Feature Extraction and Classification. Out results indicate that the convolutional networks performed very well for all the three tasks.**

I. INTRODUCTION

Melanoma is a cancer of the neural crestderived cells that provide pigmentation to skin and other tissues. Over the past 4 decades, the incidence of melanoma has increased more rapidly than that of any other malignancy in the United States. Survival of patients with malignant melanoma is directly related to early detection. The current techniques to identify a melanoma is a manual scan of the skin, but the only way to accurately diagnose melanoma is with a biopsy. There is a scope for developing more accurate, non-invasive, automated techniques to identify melanomas in order to reduce medical costs and facilitate early detection.

II. RELATED WORK

Previous approaches are oriented toward pattern recognition. These previous approaches can be classified based on whether they target global features or local features. The global approaches follow the outline of lesion segmentation, feature extraction, feature selection, and lesion classification. The local approaches on the other hand divide the image into patches and extract features from these patches. All these approaches use Statistical Learning, Naive Bayes, SVMs etc. Quite a few approaches have also used neural networks. One approach a vanilla implementation of deep nets to achieve a segmentation accuracy of 90%. Another approach uses self generating neural networks to perform segmentation. There is however a dearth of deep learning efforts into this task.
CNNs have proved to be very effective at tasks such as segmentation, localization and feature extraction and should definitely be explored for these tasks. A more recent approach used CNNs pre-trained on the ILSVRC 2012 data for feature extraction followed by SVMs for classification using the features extracted from the pre-trained CNN. This approach however, did not customize the CNN to the task at hand.
In this project, we targeted three different tasks:

- **Segmentation**: Lesion segmentation from dermoscopic images
- **Feature Extraction**: Detection and localization of visual dermoscopic features and patterns
- **Classification**: Predict the state of the lesion disease: benign or malignant.

Above mentioned tasks are part of ISBI 2016 challenge, we participated in the challenge and we will mention the results obtained in the following paragraphs.

III. SEGMENTATION

A skin lesion, in a dermoscopy image, is a single bounded region that is most often distinguishable from the normal surrounding skin by virtue of different color or texture; this area is considered to be the region of interest for further processing.

A. Approach and techniques

Prior approaches to this problem have used clustering, localized and distributed region identification, edge detection, fuzzy logic, supervised learning, graph theory and probabilistic modeling. A combination of these methods has also been used to improve performance. CNNs however have proved to be effective at segmentation with Fully Convolutional Networks and more recently the SegNet that uses an encoder-decoder approach. We decided to adapt SegNet to our task at hand to classify pixels into lesion and non lesion.

1) Data Set: The data for this part was obtained from the ISBI Phase I challenge. The lesion and the ground truth images then have to be preprocessed to adapt to the SegNet implementation

2) Implementation:

- This task involves obtaining the network trained on the CamVid road scenes dataset and modifying it to suit our task
- Then the network would be trained on our dataset and network modified to accomodate binary classification of pixels into lesion and non-lesion instead of the 11 classed that SegNet comprises.

*This work was not supported by any organization

[1]Mohit Agarwal is a graduate student at the Department of Electrical Engineering,Georgia Tech, Atlanta

[2]Nandita Damaraju is a graduate student at the Department of Computational Science and Engineering, Georgia Tech, Atlanta

[3]Sahbi Chaieb is a graduate student at the Department of Computer Science, Georgia Tech, Atlanta

3) Experiments:

- Tweaking hyperparameters to achieve best possible performance
- Choosing a suitable metric to test our performance. We chose Jaccard similarity in order to test the output of the network.
- Visualizing the output to understand what regions are being correctly labeled and what regions are being misclassified.
- Analyzing the hidden layers to understand what results in misclassification
- Modifying the network appropriately based on the previous step
- Adding suitable preprocessing/postprocessing for images to aid in segmentation

B. Architecture

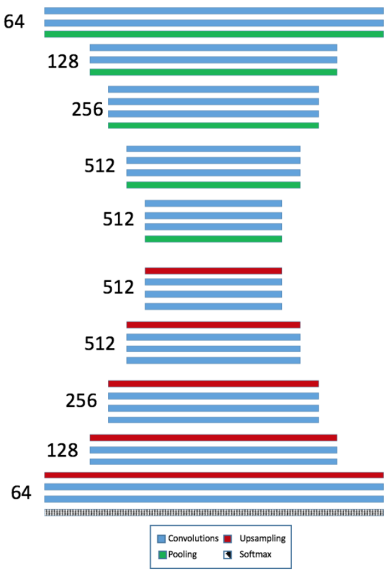

Fig. 1. Encoder Decoder architecture of SegNet

SegNet is presented as a deep fully convolutional neural network architecture for pixel-wise segmentation. It has an encoder-decoder architecture followed by a classification layer.
SegNet is the state of the art for the segmentation task, in terms of memory versus accuracy trade-off. It was designed for scene segmentation, but we decided to test on a more standard problem.

1) Input/Output: SegNet is a pixel-wise segmentation algorithm, it takes an image as an input and outputs the class of each pixel. In our problem, we will classify the pixels into two classes: skin and lesion.

2) Encoder: The architecture of the encoder network is similar to the architecture of the VGG16 network [vi], we find the same 13 convolutional layers. This also allows to speed up the training process by using pre-trained networks.

3) Decoder: The decoder part is used to obtain features for accurate pixel-wise classification, each encoder layer has a corresponding decoder layer. The final output is fed to a soft-max classifier to get class probabilities.

4) Pretraining: Since the architecture of the encoder is similar to VGG16, we can initialize the weights of the encoder using the VGG network trained on large scale data (ImageNet).

5) Hyperparameters: The main hyperparameters used in this network are the dimensions of the convolutional/pooling layers, and the gradient descent parameters: learning rate, number of iteration, momentum.

C. Design Choices

SegNet's novelty lies in the implementation of the decoder in the "encoderdecoder" architecture. The decoder stores the indices of the max pooling to memory and then later uses these indices to perform the upsampling. The alternative would have been training the upsampling layer with the rest of the layers, but this results in a higher number of parameters and a slower execution. Memorizing max-pooling indices results in a sharper boundary delineation.

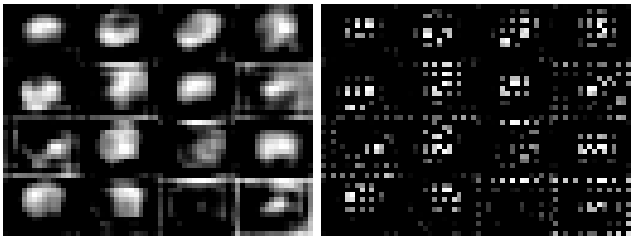

Fig. 2. Upsampling in segnet by storing max pooling indices. Left is an image of the layer output before upsampling and right represent the image after upsampling. Only the maximum values at corresponding indices are carried forward.

D. Best and worst performances

We analyzed images that had the best and worst Jaccard Indices. The middle image represents the output of the segmentation. The black pixels represent True positives and true negatives, whereas the pink pixels represent false negatives and the blue pixels represent false positives. The worst performing images had either all their misclassified pixels as False Negatives (the pink pixels in Fig 3b represent misclassified pixels or in this case false negatives for the ground truth given in image 3c) or False Positives (the blue pixels in Fig 4b represent misclassified pixels or in this case false positives for the ground truth given in image 4c) or had huge chunks of either. The better performing networks on the other hand had a mix of False Negatives and False Positive pixels mostly concentrated around the periphery of the lesion (Fig 8). We believe that appropriate pre/post

processing applied to these images will help circumvent this problem in addition to tweaking the network to identify more correct boundaries.

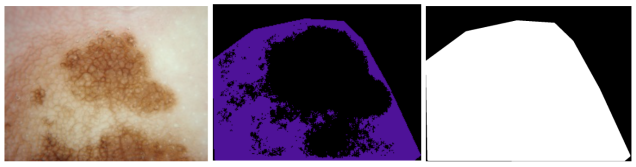

Fig. 3. Misclassified pixels: False Negatives

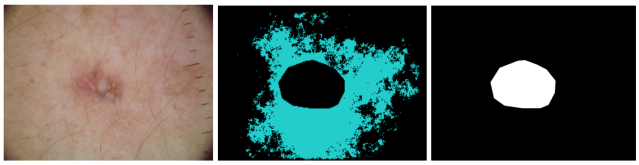

Fig. 4. Misclassified pixels: False Positives

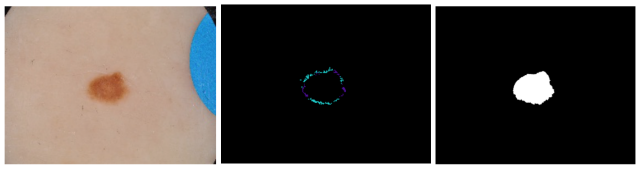

Fig. 5. Misclassified pixels: Boundary misclassification

E. Post processing

When the images were analyzed for the false negative and false positive pixels, most of the were around the boundary of the lesion or they were small islands in the background. There was to believe that the removal of these islands and smoothing of edges would result in a better pixel classification. The opening module of the OpenCV library in python was used for island removal and a Gaussian Blur for edge smoothing.

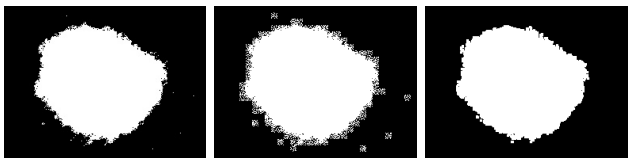

Fig. 6. Left: Original image, Middle: Island Detection, Right: Edge Smoothing and Island removal

F. Results and Analysis

1) Training Curve: As the testing mostly lower than the training accuracy, we are not overfitting our data. However to determine an accurate stopping point we need to do a more thorough analysis. Currently we stop at 400 epochs. The presence of occasional severe dips in training accuracy suggests tweaking the hyperparameters. We started out with a batch size of 2, where the dips observed were deeper. The figure below corresponds to a batch size of 7, which was the maximum our system configuration would permit. We have reason to believe that a higher batch size would result in a smoother training curve, but lacked the resources to test our hypothesis. We achieved a testing accuracy of 0.942 with the aforementioned approach.

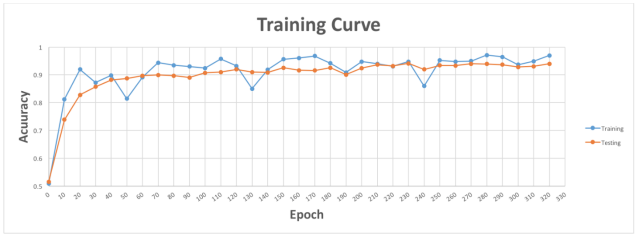

Fig. 7. Training curve

2) Hidden Layer Visualization: We visualized the output of hidden layers. We noticed that included layers are successfully able to recognize boundaries, and higher layers can see more complex features.

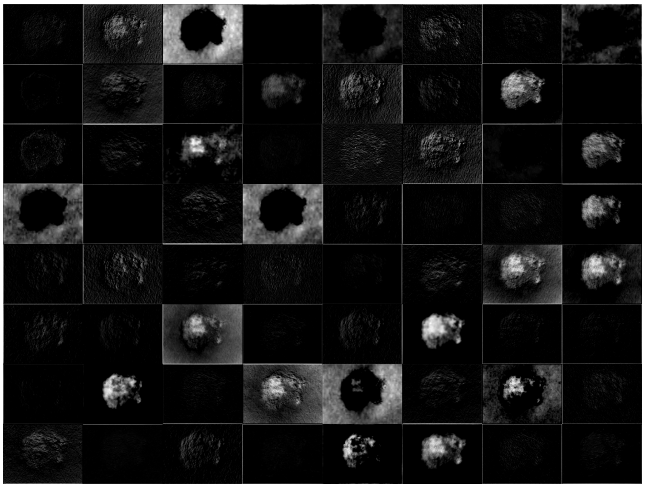

Fig. 8. Vizualizing the initial convolutional layer to understand the segmentation process

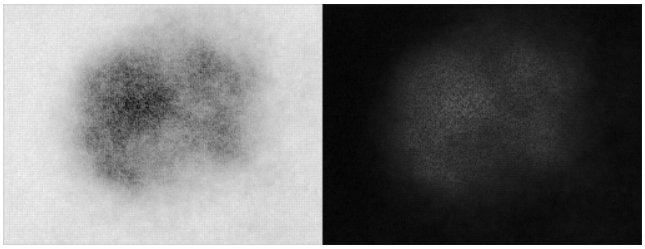

Fig. 9. Final Layer vizualization-Right: probabilities of pixels being lesion, Left: probabilities of pixels being non-lesion

IV. Feature Extraction

In this task, we had to do some localization and classification of dermoscopic features. The expected output is the probability of presence a feature for each superpixel in the image. The superpixel tiles are supplied. More specifically, we will predict the presence and absence of the "globules" and "streaks" dermoscopic features. The presence or absence of this features helps in improving classification of skin lesion images into cancerous and noncancerous ones.

A. Approach and techniques

The usual techniques for localization and classification in Deep Learning are the techniques related to the Object detection problem. The most used implementation use RCNN, however this seemed oversized to the task at hand, considering the size of the features. Instead, we opted for an approach that uses pixel-level classification. And we decided to use the same algorithm used for segmentation: SegNet.
The data used contains images of lesions and their superpixel mask, and the ground truth is a JSON file that contains the probabilities of globule and streak for each superpixel.

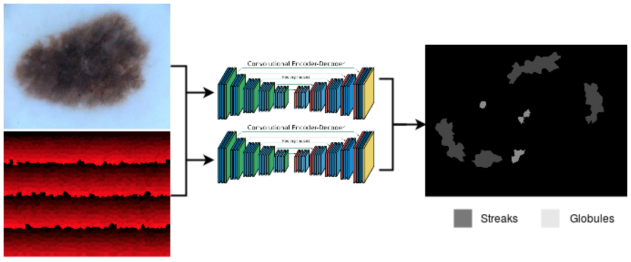

Fig. 10. Architecture of network used for feature extraction

B. Architecture

We choose the standard implementation of SegNet to do the pixel level classification and we aggregate the results over all the pixels in a superpixel to infer the probabilities.

C. Design Choices

One possibility was to train one network to classify each pixel: globule, strike, nothing or both. We can then get the probabilities of each pixel being one of these classes, however its not obvious to infer the probability of a superpixel being a globule or a streak from this classification. In addition to that, the low frequency of the globule class made learning the other classes harder after using weights for calibration.To solve this problem we decided to train two separate networks: one network for detecting globules and a second to detect streaks. We get the probabilities of a globule or a streak for each superpixel by dividing the number of pixels labeled as globule/streak by the total number of pixels in the superpixel.

D. Results and Analysis

After the training, we achieved the following Training scores:
Globules Network: 0.988
Streaks Network: 0.891
The metric used for comparison in the challenge is the average precision. We achieved an average precision of 0.147 on the final (private) dataset, with a global accuracy of 0.903
We ranked 2nd in this task.

V. Classification

This part deals with classifying given image (skin disorder snap) as either malignant or non-malignant. Today, unfortunately, the widely used method for this task is manual inspection by field experts (which presents a quite low accuracy [v]). However, efforts have been made by researchers to use traditional ML/PR techniques.

A. Approach and techniques

The starting point in classification is to use a pretrained model and fine tune it with given training set. For this purpose, we choose VGG-16 as a pretrained model trained on (ILSVRC-2012 DataSet). The intuition behind using this model, is that transfer learning is proved to perform better for classification purposes (and when significant training dataset is not available) due to similarities in low-level features.

B. Architecture

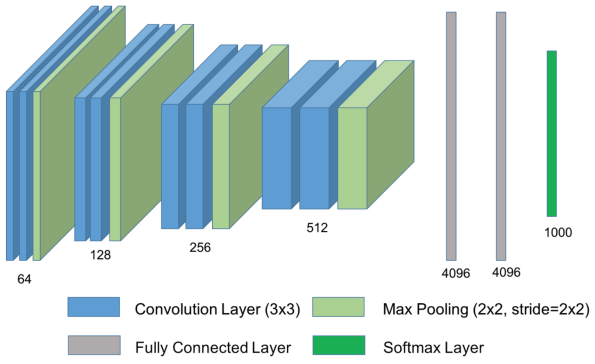

Fig. 11. Architecture of network used for classification

We are using VGG-16 and VGG-19 as our base networks to design our classification system. The rationale behind using these models are: These models are state-of-the-art in computer vision, as they stand out among top3 winners of ILSVRC-2012 Challenge Availability of pretrained architecture of VGG-16 and VGG-19 on keras as compared to other architectures (LeNet, GoogLeNet, AlexNet etc.) Transfer learning has proved to be highly efficient in visual recognition, as it helps in learning the low level features of images (through a larger dataset) including edges, patterns etc.

C. Design Choices

Rationale behind authors choices for design architecture of VGG16/19

- Lesser size of convolutional and pooling layer, with large depth. Ex. Stack of three 3x3 convolutional layers with unit stride as compared to large convolutional filters (7x7 stride=4)
- Three 3x3 conv. filters with unit stride has equivalent receptive field of 7x7, but advantages including better discriminative ability of decision function (multiple rectification units), decreased number of parameters (as three 3x3 has $24C^2$, but 7x7 has $49C^2$, less chances of overfitting).

D. Data Augmentation

Training dataset given for ISBI 2016 Challenge was biased, as around 85% of the provided images belongs to non-cancerous class and rest 15% in cancerous class. Due to this inherent bias present in dataset, our model learns to predict non-cancerous always (mostly) to give high accuracy, which wont work well in test data. For this we randomly selected few cancerous images and performed data augmentation (by flipping, rotating etc.) to achieve the ratio of cancerous and noncancerous dataset as 50:50. It also helps in providing different representations of available dataset, to make our deep learning model more robust.

E. Hyperparameters

Final hyperparameters we chose were, learning rate 1e-3, decay 1e-6, momentum = 0.1 and nesterov as false, based on experimental tweaking (trial and error methods). With very high learning rate, optimization algorithm might diverge. With very slow learning rate, optimization could take a lot of time. Hence, Learning rate decay is used to slowly decrease the learning rates over time. With high learning rate decay value, learning rate decreases very fast, hence takes much higher time in convergence and could possibly stuck at local optima. Momentum affects the path to be taken to reach optimum. It adds the fraction of previous weight update to the current one. If the gradient remains in similar direction, then momentum increases the step size, otherwise momentum smoothes the fluctuations. It also results in reduction of training time (as fluctuations are reduced).

1) Overfitting and dropout: The model was overfitting initially, which was tackled by adding more dropout layers and increasing the dropout content of the dropout layers.

2) Activation function choice: We wanted to choose the best activation function which can better discriminate our dataset. We experimented with TanH and ReLUs, and ReLUs turned out to be better in terms of both training and testing accuracy.

F. Trainable Layers

We set the last 3 layers to be trainable to fine-tune the architecture which includes all three fully connected layers. Including convolutional/pooling layers in trainable layers, does not improve the accuracy significantly but results in increasing the training time.

G. Results and Analysis

1) Training Curve: We achieved a training accuracy of 0.97 and a testing accuracy of 0.82 with the above implementation.

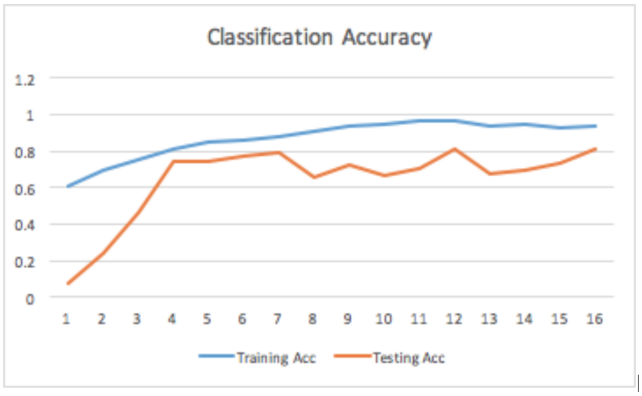

Fig. 12. Training Curve

2) Hidden Layer Visualization: The following visualization of the 64 different channels in the first convolution layer gives us a better picture of how classification is happening

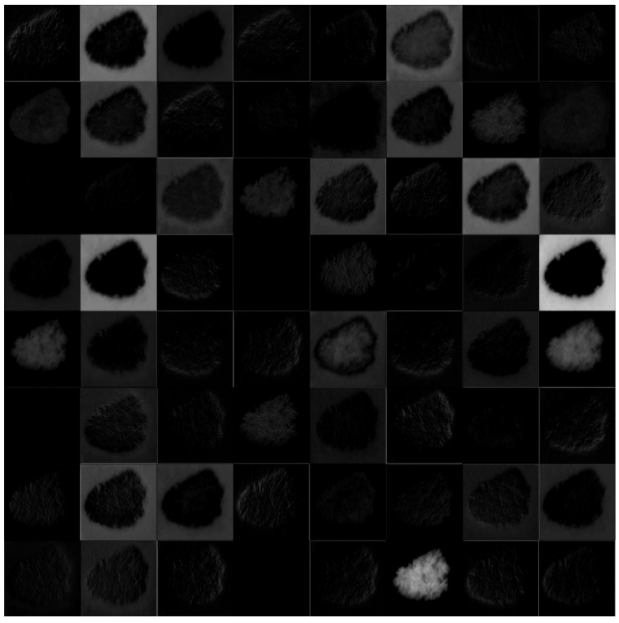

Fig. 13. Hidden Layer visualization

VI. Discussion

The above document summarizes a detailed attempt at using stateoftheart techniques for melanoma detection. As hypothesized earlier, Convolutional Networks performed better than other vanilla approaches at handling this task. We achieved an accuracy of 0.94 for Segmentation, 0.9 for feature extraction and 0.82 for classification. There are many ways this could be extended to further improve performance. The most obvious way would be to combine all the three tasks to improve performance. This would require a very large network and hence system with a superior configuration to train on all the images, which was out of our reach and hence was not explored. We also had very little wiggle room when it came down to our choice of hyperparameters owing to the system limitations. We believe that this was a great-first-pass at using convolutional networks for this task and this has a lot of scope for improvement.

APPENDIX

Our code can be found at https://github.com/NanditaDamaraju/DL8803

ACKNOWLEDGMENT

We would like to thank Dr. Zsolt Kira who helped us out with architecutral choices and improvements. His inputs were invaluable and we are very grateful. We would also like to thank the organizers of the ISBI challenge for providing a well-curated dataset and for helping us out with any data related issues.

References

[1] Chudnovsky, Yakov, Paul A. Khavari, and Amy E. Adams. "Melanoma genetics and the development of rational therapeutics." The Journal of clinical investigation 115.4 (2005): 813-824.

[2] Rigel, Darrell S., and John A. Carucci. "Malignant melanoma: prevention, early detection, and treatment in the 21st century." CA: a cancer journal for clinicians 50.4 (2000): 215-236.

[3] Barata, Catarina, et al. "Two systems for the detection of melanomas in dermoscopy images using texture and color features." Systems Journal, IEEE 8.3 (2014): 965-979.

[4] Codella, Noel, et al. "Deep Learning, Sparse Coding, and SVM for Melanoma Recognition in Dermoscopy Images." Machine Learning in Medical Imaging. Springer International Publishing, 2015. 118-126.

[5] Melli R, Grana C, Cucchiara R. Comparison of color clustering algorithms for segmentation of dermatological images. InMedical Imaging 2006 Mar 2 (pp. 61443S-61443S). International Society for Optics and Photonics.

[6] Esteva, Andre, Brett Kuprel, and Sebastian Thrun. "Deep Networks for Early Stage Skin Disease and Skin Cancer Classification."

[7] Abbas Q, Celebi ME, Fondn Garca I, Rashid M. Lesion border detection in dermoscopy images using dynamic programming. Skin Research and Technology. 2011 Feb 1;17(1):91-100.

[8] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014