Comparative Study of Various Machine Learning Methods for Skin Cancer Classification

Tiffany Nakai and Leah Restad

Introduction

Skin cancer is the most common type of cancer, with 3.5 million new cases diagnosed annually. If detected early, basal and squamous cell carcinomas are highly treatable. Overall, the five-year survival rate in the United States is 93%, but early detection is crucial to achieve this rate [12].

Previous works involving machine learning algorithms for skin lesion classification have:

- Compared machine learning algorithms to convolutional neural networks with substantive preprocessing of reordering, resizing, and augmenting of images [10]
- Analyzed dataset size, ad hoc image selection, and racial bias present in other research into the accuracy of machine learning algorithms for skin lesion classification [8]
- Demonstrated that deep convolutional neural networks with sufficient training data can classify images of skin lesions just as accurately, if not more so, than dermatologists [5]
- Evaluated the accuracy of other individual machine learning algorithms when applied to skin lesion classification [1, 2, 3, 5, 6, 8, 11]

Current research demonstrates that with sufficient data, machine learning models can fairly accurately classify images of skin lesions. According to Esteva et al., machine learning algorithms for skin lesion classification can be performed via mobile devices, which are easier to access than dermatologists, and can therefore decrease the barrier to early detection [5].

Though there are existing works both evaluating and comparing the accuracy machine learning algorithms when applied to skin lesion classification, most research has focused on standardized or heavily pre-processed data. Our work expands upon existing research by providing a comparative analysis of machine learning algorithms for skin lesion classification on raw data, which would more closely resemble the images input to skin lesion classifiers on mobile devices. Our main contributions are:

- An additional benchmarking of the state-of-the-art skin lesion classifiers to identify the most performant models across metrics for training speed and accuracy.
- An interpretability study to identify the models that can best stratify feature embeddings of images prior to the final classification layer.
- An accessible implementation to run the nine state-of-the-art skin models on a custom dataset and conduct comparative analyses for future work.

The findings from this research may be applied to evaluate currently deployed skin lesion classifiers and potentially optimize current deployments to optimize for accuracy and training time to improve ease of early detection of skin cancer carcinomas and encourage early treatment.

Data

We are using Skin Cancer MNIST:HAM10000 dataset from Kaggle [15]. Link here: https://www.kaggle.com/datasets/kmader/skin-cancer-mnist-ham10000. This is a dataset that consists of 10,000 images of skin cancer samples. This dataset consists of skin cancer samples for 7 classes of skin cancer: Actinic keratoses and intraepithelial carcinoma / Bowen's disease (akiec), basal cell carcinoma (bcc), benign keratosis-like lesions (bkl), dermatofibroma (df), melanoma (mel), melanocytic nevi (nv), and vascular lesions (vasc) [14]. The images have jpeg format with resolution 600 x 450 pixels.

Figure 1 is an image of the number of images per class in this dataset. There is a class imbalance, with significantly more melanocytic nevi (nv) represented. Imbalances in datasets are pretty common within medical datasets, so it is not surprising that it is present in this one [10].

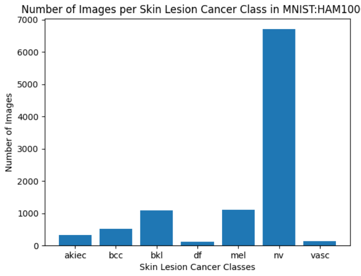

Figure 1: Bar plot of number of images per skin cancer class

We chose this dataset because it has skin cancer image samples associated with their dermatologist-labeled class. This made it useful for our project because we wanted to compare methods of classifying images for skin cancer, so an image dataset was crucial in enabling us to do this comparative study. Outside of manually cropping images to the skin lesion, no other adjustments have been made to the images, so this would accurately reflect the images input to a skin lesion classifier on a mobile device for public use.

For our project, we split this dataset into a 60-20-20 train-validation-test set. We split the dataset by this 60-20-20 proportion for each class, so the train, validation, and test set all have the same proportion of each skin cancer class.

Methods

We compared many supervised learning image classification methods. We only used transfer learning methods using pre-built and ImageNet pre-trained architectures from PyTorch [18]. The machine learning algorithms included AlexNet [7], DenseNet121 [13], EfficientNetB3 [6], EfficientNetB4 [6], EfficientNetB5 [6], InceptionV3 [2], MobileNetV2 [3], VGG16 [9], and VGG19 [1], each of which had existing research indicating potential utility for skin lesion classification. The machine learning algorithms, along with the number of parameters and layers used, are summarized in Figure 2.

Model	Number of Parameters	Number of Layers
AlexNet	61,100,840	8
DenseNet121	7,978,856	72
EfficientNetB3	12,233,232	384
EfficientNetB4	19,341,616	474
EfficientNetB5	30,389,784	576
InceptionV3	27,161,264	48
MobileNetV2	3,504,872	53
VGG16	138,357,544	16
VGG19	143,667,240	19

Figure 2: Table of models with parameter number and number of layers.

All of the code was written in Python. We used PyTorch for training and inference with trained models. Tensorflow was used to determine the number of layers in the EfficientNet models. We built off of existing PyTorch boilerplate code for custom Datasets, Dataloaders, and transfer learning to conduct our comparative analysis [14,15]. Our code implementation can be found at this GitHub repo link: https://github.com/Cotna7676/comp_med_22

We validated our results using best validation accuracy f-score, test accuracy f-score, precision, recall, weighted precision, and weighted recall, based off of template code [15-temp]. We analyzed our results by comparing the validation results across machine learning algorithms.

We created visuals for the results of each machine learning algorithm with the following libraries: sci-kit learn for t-SNE [19] and confusion matrices [20] and umap for UMAP [21].

Results

We found that DenseNet121 and MobileNetV2 performed the best in terms of accuracy among all the nine models we tested, though AlexNet had the fastest training time with a roughly comparable accuracy. We reached this conclusion by comparing validation and test accuracies, t-SNE plots, UMAP plots, and confusion matrices.

We ran each of the 9 models for 100 epochs. We ran all of these models locally using cpu (as we had AWS issues), hence why the training time for some models are very large. Some models took over 60 hours to train. We used SGD optimizer with learning rate = 0.001 and momentum = 0.9. We used CrossEntropyLoss for our loss function. As we wanted to compare model architectures, we kept these parameters the same across all of the models.

Figure 3 is a summary table of the models, with training time, accuracy, and precision. The weighted precision and weighted recall account for the class imbalance. MobileNetV2 and DenseNet121 had the best f-scores, along with the best weighted precision and recall. AlexNet had the shortest training time, though it had a roughly median accuracy amongst the nine models we analyzed.

Model	Training Time	Best Validation Accuracy (F-score)	Test Accuracy (F-score)	Precision	Recall	Weighted Precision	Weighted Recall
AlexNet	**444m 40s**	76.9%	75.9%	58.2	50.9	74.5	76.5
DenseNet121	3789m 33s	79.4%	77.6%	64.7	53.2	77.2	78.7
EfficientNetB3	2226m 35s	77.5%	76.1%	60.9	44.7	73.6	76.1
EfficientNetB4	2824m 43s	71.8%	71.7%	41.8	27.6	67.6	73.3
EfficientNetB5	3239m 30s	75.8%	74.6%	**66.5**	41.1	72.2	75.1
InceptionV3	3641m 22s	74.3%	72.9%	62.1	36.3	70.5	73.9
MobileNetV2	2457m 1s	**80.1%**	**78.2%**	64.4	**56.0**	**78.0**	**79.4**
VGG16	3154m 15s	71.2%	72.4%	52.1	37.7	69.8	73.5
VGG19	3421m 52s	71.5%	69.5%	46.4	33.7	67.2	71.0

Figure 3: Table of models with training time, validation and test accuracies, precision and recall

To visualize the results, we removed the last classification layer and ran t-SNE on features. MobileNetV2 and DenseNet121 appear to best stratify the classes within the final layer of the model, demonstrated by the better class grouping in the t-SNE plots for the models.

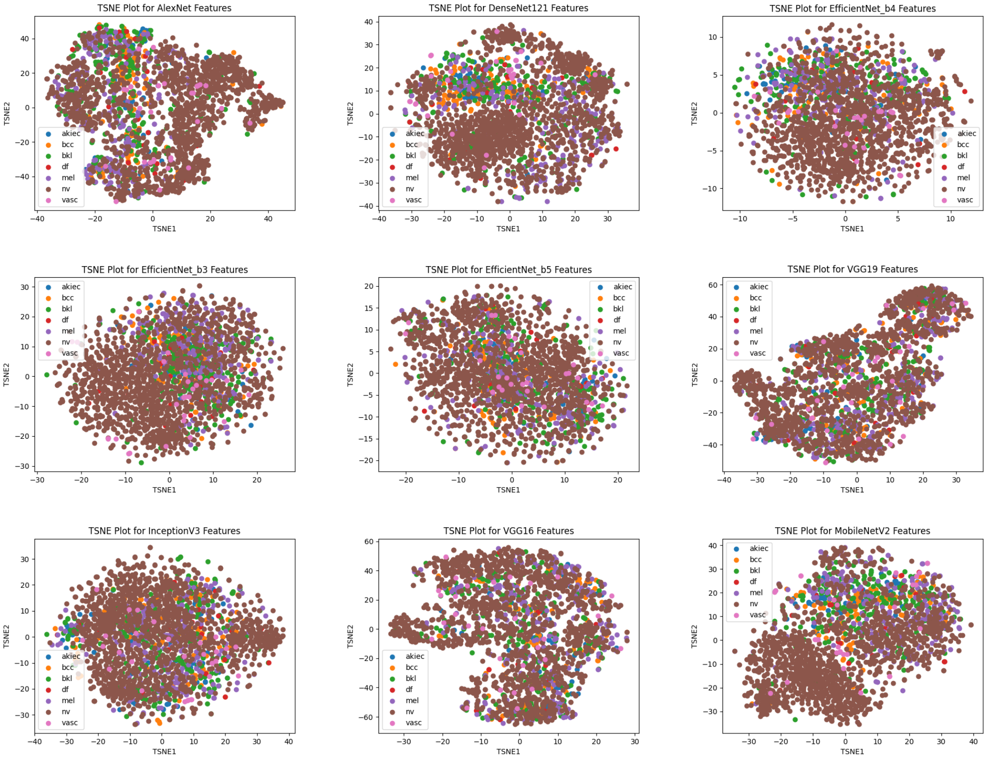

Figure 4: t-SNE visualization of feature extraction by model

As t-SNE did not show very clear results, we also used UMAP to visualize the results. Furthermore, UMAP may give a better presentation of the global structure, which can provide a deeper insight into the internal representation of each class within the models we analyze [11]. As before, MobileNetV2 and DenseNet121 continued to show relatively clear grouping, but AlexNet and EfficientNetB3 seemed to show better grouping with UMAP as well.

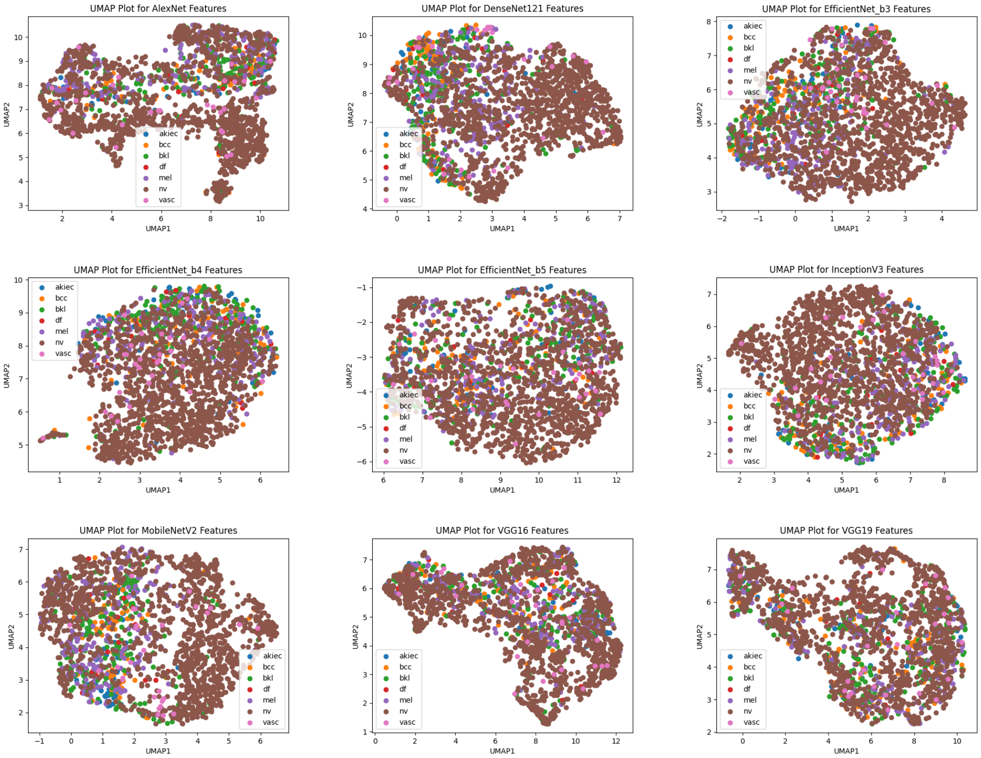

Figure 5: UMAP visualization of feature extraction by model

Finally, we generated confusion matrices to investigate the relative performance of our models for each class. These matrices are also known as error matrices and they illustrate how well the model classifies, so they illustrate the amount of true/false positive/negative values for each class [4]. The rows represent true instances of a class and the columns represent the predicted instances of a class. Yellow spots indicate high correspondence of a true and a predicted class (more true positives), while dark blue/purple indicate low correspondence.

In Figure 6, all of the models predict the nv class well, but none of the other 6 classes appear to. This can be attributed to the disproportionately large amount of nv in the dataset. The models that seemed to classify the best based on these matrices (determined by examining the yellowness of cells on the diagonal axis of the plot) are DenseNet121, MobileNetV2, EfficientNetB3, and AlexNet.

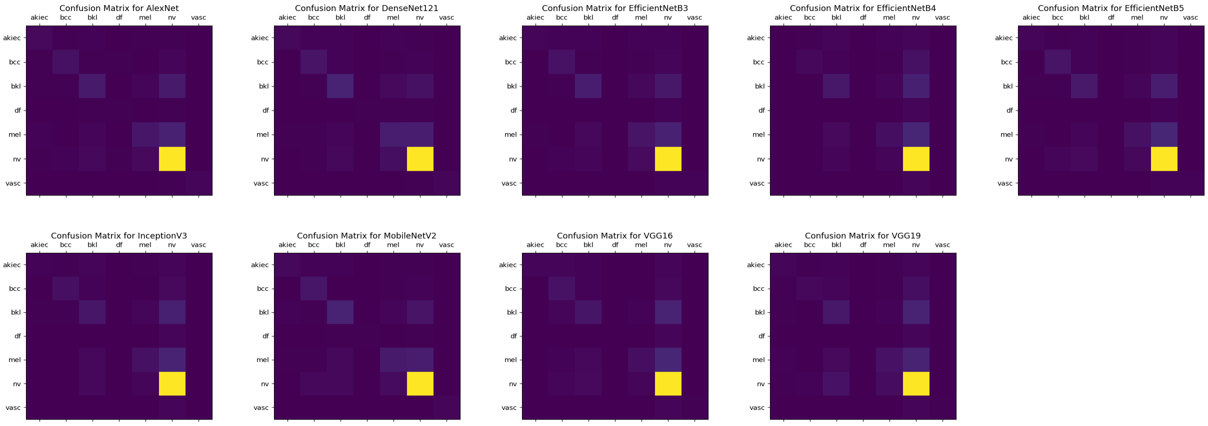

Figure 6: Confusion matrices by model

Thus, from these comparisons and visualizations, we found DenseNet121 and MobileNetV2 to be the best performing models among these nine.

Conclusion

MobileNetV2 and DenseNet performed best among the nine models trained, with the best validation accuracy f-scores, best test accuracy f-scores, best weighted precision, and best weighted recall. Additionally, AlexNet had the fastest training time while maintaining near-average accuracies relative the nine models we compared. However, the imbalance in the dataset meant that all models were heavily skewed towards the skin cancer class with the highest number of images, the melanocytic nevi (nv).

The results indicate that future work should be done to try and make the models less affected by this class imbalance, so that they can be compared more evenly across skin lesion classes. Additionally, some of the reference research was able to perform well by incorporating metadata provided (age, sex, location of sample on body, etc) with the HAM10000 dataset, so future research should be done to incorporate this metadata when training and comparing the models, to determine if this changes which models perform best.

References

[1] Abuared, N., Panthakkan, A., Al-Saad, M., Amin, S. & Mansoor, W. Skin Cancer Classification Model Based on VGG19 and Transfer Learning. https://pure.strath.ac.uk/ws/portalfiles/portal/134360852/Aburaed_etal_ICSPIS_2021_Skin_cancer_classification_model_based_on_VGG19.pdf
[2] AnuRadha, T., Bhavya, S., Narasimha, R., Ramya, M., & Sujana, S. (2018). Classification of skin cancer images using TensorFlow and inception v3. https://www.researchgate.net/publication/325117430_Classification_of_skin_cancer_images_using_TensorFlow_and_inception_v3
[3] Bhosale, S. (2022). Comparison of Deep Learning and Machine Learning Models and Frameworks for Skin Lesion Classification. https://arxiv.org/pdf/2207.12715.pdf
[4] Confusion Matrix. (2022, August 31). In *Wikipedia*. https://en.wikipedia.org/wiki/Confusion_matrix
[5] Esteva, A., Kuprel, B., Novoa, R., Ko, J., Sweater, S., Blau, H., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. https://www.nature.com/articles/nature21056
[6] Ha, Q., Liu, B., & Liu, F. (2020). Identifying Melanoma Images using EfficientNet Ensemble: Winning Solution to the SIIM-ISIC Melanoma Classification. https://arxiv.org/pdf/2010.05351.pdf
[7] Hosny, K., Kassem, M., & Food, M. (2019). Classification of skin lesions using transfer learning and augmentation with Alex-net. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0217293
[8] Kassem, M., Hosny, K., Damasevicius, R., & Eltoukhy, M. (2021). Machine Learning an Deep Learning Methods for Skin Lesion Classification and Diagnosis: A Systematic Review. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8391467/
[9] Manasa, K., & Murthy, V. (2021). Skin Cancer Detection Using VGG-16. https://ejmcm.com/article_7276.html
[10] Shetty, B., Fernandes, R., Rodrigues, A., Chengoden, R., Bhattacharya, S., & Lakshmanna, K. (2022). Skin lesion classification of dermoscopic images using machine learning and convolutional neural network. https://www.nature.com/articles/s41598-022-22644-9
[11] Tran, M. (2022). Comparing UMAP vs t-SNE in Single-cell RNA-Seq Data Visualization, Simply Explained. https://blog.bioturing.com/2022/01/14/umap-vs-t-sne-single-cell-rna-seq-data-visualization
[12] Wu, W. (2022). [Lecture notes 02518 Computational Medicine]. Computational Biology Department, School of Computer Science, Carnegie Mellon University. https://canvas.cmu.edu/courses/29899
[13] Zare, R., & Pourkazemi, A. DenseNet approach to segmentation and classification of dermatoscopic skin lesions images. https://arxiv.org/pdf/2110.04632.pdf

References - Tools

[14] https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/DBW86T
[15] https://www.kaggle.com/datasets/kmader/skin-cancer-mnist-ham10000
[16] https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html#load-data
[17] https://pytorch.org/tutorials/beginner/basics/data_tutorial.html
[18] https://pytorch.org/vision/stable/models.html
[19] https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html
[20] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
[21] https://umap.scikit-tda.org/index.html