A Multi-Model Approach to Skin Lesion Analysis with Machine Learning

Aarya Kulshrestha, Bowen Yi, Christopher Erndteman, Maaz Hussain, Shankhin Chirmade
Department of Computer Science and Engineering, University of Michigan
{akulshre, bowenyi, chrisern, maazh, shankhin}@umich.edu

1. Introduction

Skin cancer affects millions of people worldwide. Early detection of skin cancer is one of the largest factors in determining survival rate. According to the American Cancer Society, the five-year survival rate when melanoma is detected while it is localized and has not spread is over 99% [18]. Patients need easy access to skin cancer screening tools. Therefore, it became clear that skin cancer detection is the perfect application for computer vision technology using convolutional neural networks.

To address the importance of early and accurate detection of cancerous skin deformations, our team researched available classification networks to take inspiration from. We noticed that many papers used fine-tuning of popular convolutional neural network (CNN) architectures such as AlexNet [12] and ResNet [9]. Two research journals tackled a similar challenge to our own. Firstly, a 2021 study [2] employed a single large CNN that classified the same 7 skin lesions classified by our model. Secondly, Alkarakatly, T. et al. [1] provided a classification of 3 different lesion types using a CNN. Consequently, we decided to create a CNN that classified lesion-specific features more effectively by utilizing a 3 model structure, leading to improved model performance. More importantly, our model provides better interpretability, giving the patient a cancerous prediction accompanied by a lesion classification.

Our first model provides a probability estimation that the provided deformity is cancerous. It states whether the provided skin deformation is: very unlikely cancerous, unlikely cancerous, likely cancerous, or very likely cancerous. Once the primary model has predicted the cancerous nature of the image, the image is then further processed by one of the two secondary models. Provided that the image has been classified as cancerous, the image is passed to a model that classifies the type of cancer. The two classifications our data provides are melanoma (mel) and basal cell carcinoma (bcc). If the primary model estimates that the image is non-cancerous, the image is passed to a model that classifies non-cancerous deformities. The non-cancerous classification model provides five classifications, actinic keratoses and intraepithelial carcinoma/Bowen's disease (akiec), Benign lesions of the keratosis (bkl), dermatofibroma (df), melanocytic nevi (nv), and vascular lesions (vasc) [22].

The idea of cancer detection became clear to us after the father of a member of our project team was diagnosed with bladder cancer in January. While bladder cancer is not easily detectable early through CV techniques, skin cancer is. Therefore it became clear to our team the importance of utilizing our computer vision experience gained in EECS 442: Computer Vision to research this life-saving application.

Our application's importance is evident outside the doctor's office, providing detection before an appointment. The recommended treatment time for melanoma before serious progression is 4-6 weeks [10]. However, the average wait time for a dermatologist appointment (after a recommendation from a primary physician) is 28.8 days [11], and the average wait time for a primary care appointment is about 26 days [16]. A method of at-home detection could be beneficial for catching early-onset melanoma before an appointment.

Our work aims to develop a mostly accurate program to identify early skin deformities (both cancerous and not) with limited computational power. Our model achieves this by specializing in a computational approach that differentiates between benign and malignant mole types on a patient's skin; Giving us freedom to perform a nuanced analysis of the input images. This focus allows us to develop a scalable, accurate, and user-friendly tool for early skin cancer detection. The increasing availability of digital dermatology and telemedicine, [25] has laid the foundation for our project to significantly enhance modern remote diagnosis capabilities, with heightened importance for regions with limited access to dermatologists.

Our project code is publicly available on GitHub.[1]

2. Background

Throughout the creation of our three distinct models, we utilized differing medical journals, datasets, and publications as the basis of our development. The publication Skin Diseases Classification Using Deep Learning Methods proved useful for guiding our medical knowledge while developing

[1] **https://github.com/bowenyi-umich/Skin-Cancer-Detection-442-Final-Project-.git**

our models [22]. This work provided important statistical information on each classification. Additionally, the neural network design detailed in this journal provided us with the idea of using dropout layers at the end of each block of our network.

During our research, we experimented with multiple dataset options. Our initial dataset source was the International Skin Imaging Collaboration (ISIC) [6]. However, upon further consideration and the goal of providing classification for skin lesion type in accompaniment to cancerous probability, we decided to utilize the Harvard HAM10000 dataset instead [20].

To grasp the nuances and values of this work in its entirety, there are a few technical expectations of the reader. Firstly, basic medical knowledge of skin cancer terminology is expected. We refer to melanoma and basal cell carcinoma (bcc) as malignant/cancerous, and the other 5 skin lesions as benign/non-cancerous. Moreover, a basic understanding of convolutional neural networks, including corresponding hyperparameters and various types of layers, is required to analyze and evaluate the methodology portion of this report. However, in the absence of technical knowledge, many will still find value in reading the results of this report as it pertains to the difficulties and successes of utilizing machine learning techniques for skin cancer classification.

3. Methodology

3.1. Dataloader and Preprocessing

Throughout the development process of our three distinct models, our team faced computation challenges and restrictions. We looked into three options for computation: (1) Google Colab, (2) Kaggle, and (3) Locally running Jupyter Notebooks. The dataset we chose to use was not available on Kaggle, and Kaggle did not support the required upload size for our dataset. Additionally, when running Jupyter Notebooks locally, we did not have access to the GPU memory needed for adequate training. Google Colab was the clear choice for development due to its integration with Google Drive and access to GPU memory. It is notable to mention Google Colab's limitations on GPU resources and its effects on model training. Due to GPU resource maximums, we were unable to train our models for extended periods (sessions beyond 3 hours).

Data integration is a key aspect of training our three models. Our models utilize supervised learning requiring large amounts of pre-labeled data. We decided to use the HAM10000 dataset for training our models [21]. The HAM10000 dataset contains much less data than the alternative ISIC dataset (2023), however, the HAM10000 had more accurate labeling of data classifications [6]. We found that the ISIC dataset contained thousands of images labeled as "unknown" for the classification, and while having more images in the aggregate, had fewer images for specific classes such as bcc or akiec. The dataset makeup can be found in Figure 1.

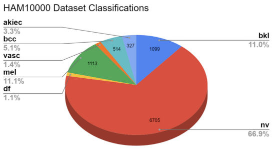


Figure 1. Dataset Class Makeup

Loading 10,000 images for training with corresponding metadata proved to be an initial challenge. Our first iteration attempted to load the images directly into a Python dictionary with the corresponding class, however, this was incredibly slow. The estimated time to load all 10,000 images was 17.3 days. Upon realizing that this method would provide unreliable results for data loading, we turned to the Python library PyTorch [15]. We used compression to reduce the data size of the images, as the images took too long to load into Google Drive. For data loading, we used PyTorch's built-in dataloader class. This class requires a custom subclass of the PyTorch Dataset class. Our derived class, ImageData, contained the paths to three CSV files containing the partitioned metadata for our dataset. The three partitioned CSV files contained metadata for the entire set, metadata for just the cancerous images, and metadata for only the non-cancerous, the CSVs were partitioned into 80% training, 10% validation, and 10% test. Upon iterating through the data loader object, our class returns three vectors containing: (a) image name, (b) classification, and (c) image as a tensor. Appendix section A contains two raw, non-preprocessed example outputs from the data loader matched with the classification and image name for training, validation, and test data. We applied random rotation and Gaussian noise to the images to augment our dataset and increase the model's ability to generalize. Two samples of preprocessed images are presented in the appendix section A.

3.2. Overall structure

The goal of our program is to use various custom CNN models to help the user learn more about their inputted skin abnormality. The first model in our program classifies the user's image as either malignant or benign. Using that data to decide whether to run model 2 — the model deciding whether or not the disease is melanoma or bcc (the two types of cancer available in the dataset) — or model 3 — the model classifying what type of non-cancerous abnormality the lesion could be. While the bulk of the project was data formatting and processing, and the creation of these 3 models, we added a section to the main program driver to call a medical LLM that we found during our research. This

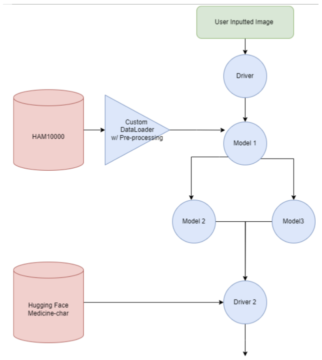

User Inputted Image
Driver
HAM10000
Custom DataLoader w/ Pre-processing
Model 1
Model 2
Model3
Hugging Face Medicine-char
Driver 2

Figure 2. Program Structure

provides further information about the disease predicted by our model. See fig 2 of our program structure, where green represents user input, red corresponds to outside sources, and blue represents our custom code (note the sections in blue did use outside libraries such as PyTorch and numpy).

3.3. Model 1

All of our models were implemented using PyTorch [15]. The model 1 architecture was driven by the optimization of the limited computation power available to us. Each block of our model consists of a 2D convolutional layer of decreasing height, decreasing width, and increasing dimensionality. We used ReLu activation and dropout layering to prevent overfitting. We use these throughout the model to maintain consistency. Max-pooling was utilized throughout the model to maintain computational efficiency. Without max-pooling our model took about an hour to complete 3 epochs. Using max-pooling helped decrease training time while increasing feature extraction and preventing overfitting. We originally had fewer convolutions and a smaller input size into the fully connected layer, however, this gave us prediction accuracy not much higher than random. We also found that more layers or a bigger filter size would cause the model to lose complexity, resulting in limited learning. We used BCE logit loss for its built-in sigmoid function and continued to use Adam from PyTorch as our optimizer (this is also true for model 2). See a visual diagram of our model 1 architecture in fig 3.

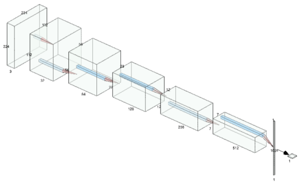


Figure 3. Model 1 Architecture

We resized all the images to be 224x224 because we experienced issues with inconsistent image sizing in the original dataset. We chose these dimensions by looking at the design of AlexNet [13]. We tuned various hyperparameters to optimize our model. We first looked at the learning rate of our model. We experimented with balancing optimization and speed. We found that a learning rate of .0001 was far too slow, but a learning rate of .1 would hurt the performance of our model by around 10% (most likely due to being unable to find local minima during gradient descent), so we decided to use 0.001 as our learning rate. We also found that our training accuracy did not substantially increase with epochs above 4.

One problem during this part of the project was an imbalance in our dataset between labels. Only about 20% of the dataset was malignant images. We first considered data augmentation, specifically artificially increasing the number of malignant images. However, since we were already struggling with long training times we decided to use downsampling. We also wrote code to apply heavier weights to the minority class during classification, yet, we did not see much difference. Only about a 0.01 difference in validation accuracy so we attempted another option. We downsampled the non-cancerous dataset while maintaining relative proportions, giving equal proportions of benign and malignant data (this also helped our issues with Google Colab). This yielded worse results due to the model over-fitting certain features in the cancerous label. We also wanted to compare our results with an established model. We chose to compare our results with a CNN called LeNet [14], an older model that isn't specifically meant for medical uses, but is easy to access and compare to our model. The accuracy of this model and its comparison with the performance of LeNet can be seen in Fig 4 of the result section.

3.4. Model 2

The second model is designed to detect melanoma. Melanoma is one of the most common types of skin cancer, and one of two malignant skin diseases in our dataset (the other being basal cell carcinoma). We again used supervised binary classification for this model. We used similar hyperparameter tuning techniques as model 1 and experimented with preprocessing as mentioned above. Another strategy we tried is using a residual block. We again used convolutions of increasing dimensionality and decreasing size, and we used max-pooling, again mostly for the effects of preventing overfitting. However, at first, we saw slightly worse results than we had for model 1. Our theory is that this is because there was substantially less data for this model, as this model was only being trained on the melanoma and basal cell carcinoma datasets. The use of a residual block in our architecture highlights a major difference between models 1 and 2. With a residual block, we can help prevent vanishing gradients and overfitting through regularization.

We used a residual block of dimensions 64 to 128 and a stride of 2. Our custom residual block also has the advantage of having arguments for the input and output dimensions, which we implemented to grant better flexibility when tuning the model. Again, we can see the effects of using and not using residual blocks in the results section below with Fig 6 and Fig 7 respectively. In addition, visualizations of the model architecture with and without residual connections are presented in the appendix Fig 10 and Fig 11, created using netro [17].

3.5. Model 3

Our third model was implemented to do a multi-class classification of images classified as benign by our first model. As mentioned before, we have five benign classifications, actinic keratoses and intraepithelial carcinoma (akiec), Benign lesions of the keratosis (bkl), dermatofibroma (df), melanocytic nevi (nv), and vascular lesions (vasc) (Udristoiu et al., 2020). The main challenge of this model was implementing multi-class classification instead of binary classification. We still used PyTorch and CNN's but our architecture had to change for multiple labels. We also added further code for the driver program to extract the best predictions. We used cross-entropy loss instead of BCE loss in this section as we were dealing with a multi-class classification. We also continued to use the Adam optimizer. We again ran this model with and without residual blocks. We used 2 batch normalization layers with our ReLU activation to increase the speed and consistency of our model, using 2d max pooling and drop-out layers to decrease the complexity of our model. This was because we faced more Colab limitations than in model 2 due to there being more images to train on and more classes. We decided to use softmax as well to assign probabilities to each of the 5 classes. Lastly, our fully connected layer was also different because of the change in input and output size. See the appendix fig 12 for the full architecture.

3.6. Goals

One of our main goals to differentiate our research from other established models was to architect a pipeline of multiple models that gives the user a better understanding of their current condition. We first used the DataLoader to process and partition our data. We train and evaluate the 3 different models. Then, given the input from the user we classify their skin abnormality with model 1 as malignant or not. Given the output of model 1 we either classify on model 2 or 3. Most of the literature mentioned in this paper use CNNs to classify lesions, but grouped the cancerous and non-cancerous lesions into one model. In this same vein, we had some extra time and wanted to add features outside the scope of 442. In order to help a potential user better understand their condition, our program takes the output of these models and uses an LLM to provide context.

3.7. LLM

While our models have decent accuracy in classifying the model, real-world diagnosis involves seeking medical advice from a qualified professional. We wanted to do something a bit beyond the scope of the class/project and try to experiment with somewhat replicating this real-world interaction in our program. A 2023 paper [4] highlights the potential of Large Language Models (LLMs) as a supplementary tool for retrieving medical information. Inspired by the work, we wanted to add something extra ontop of our 3 custom models. Our pipeline leverages LLMs to simulate a preliminary information-gathering process similar to a doctor's initial assessment. However, it's crucial to emphasize, as noted in Clusmann's work [4], that LLMs are not a replacement for professional medical advice.

After identification of a skin disease by our previous models, our pipeline leverages a large language model (LLM) called AdaptLLM/medicine-chat [3] for retrieval of relevant medical information about the identified disease. Medicine-chat is specifically trained on a massive dataset of PubMed Abstracts and FreeLaw Opinions [8], providing it with a deep understanding of both the medical and legal aspects of various conditions. For consideration, we also evaluate the performance of Google Gemini [19], another LLM that is pre-trained on a much broader dataset, but isn't focused on medical information, so we were curious to compare there results. In order to evaluate these results we used the following procedure :

1. Disease Identification: Our previous models (model 1 & 2 & 3) analyze the input image to identify the type of skin disease.
2. Prompt Construction: Leveraging the patient's gender, age, and affected body part information, the pipeline constructs a prompt combining disease details and patient specifics.
3. LLM Response: This prompt is then fed to both AdaptLLM/medicine-chat and Google Gemini for response generation.
4. Response Evaluation: We then evaluate the responses from both models to determine which provides the most relevant and informative medical advice.

4. Results

The goal of our project outlined in the proposal was to get a testing accuracy of around 80%. Thus, we used the accuracy of the testing data as the final metric for each model. We also wanted to make the program a helpful interface for users to better understand their skin abnormalities. In this section, we will evaluate if we met those goals.

Model Number	Testing Accuracy(%)
1	89.07
2	87.27
3	84.13

Table 1. Testing Accuracy of all 3 Models

4.1. Model 1 Results

We will start with model 1. Model 1 was by far the most computationally expensive (as it used the largest dataset), and this limited us in the number of epochs we wanted to show. After initial testing, we found that the training and validation accuracy did not increase past 4 epochs. This also took a lot of time to run on Colab and, at some points, we ran into Colab GPU resource depletion; Therefore we only used 3 epochs for model 1. We also wanted to test our dataset against an established CNN so we downloaded LeNet and ran the same training on our dataset. Looking at the two graphs LeNet performed only slightly better. It is important to note that LeNet is more than 25 years old, and not partially designed for this exact usage as our custom CNNs are, but LeNet is very highly regarded, so we are relatively content with the results for training accuracy. We also ran our testing data and LeNet had a final test accuracy of 89.83% compared to our accuracy of 89.07%. The training accuracy for LeNet and our model are detailed in Fig 4 and Fig 5.

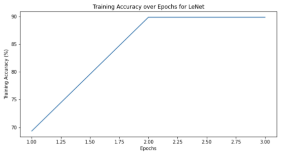


Figure 4. Training Accuracy of LeNet

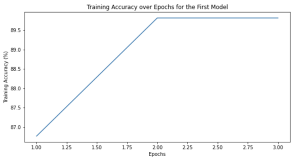


Figure 5. First Model Training Accuracy

4.2. Model 2 Results

As mentioned in the methodology section, one of the first things we wanted to do was measure the use of preprocessing. We chose to do this comparison on model 2 because it was the quickest model that used the least amount of GPU resources, so we could use 10 epochs. Overall we found that the preprocessing we used did not have much effect on the dataset. As mentioned in our methodology, we included resizing, rotation, and normalization. The comparison of the preprocessed vs non-preprocessed is shown below in Fig 6 and Fig 8. Our model 2 had a final test accuracy of 87.27% without preprocessing and a final test accuracy of 85.45%. We think this may be because the preprocessing techniques we chose were not very helpful for this test set, but this is something we want to look into in the future.

For model 2 we wanted to experiment with residuals, and we found that without a residual block, our model had barely improved from epoch to epoch. This resulted in a very flat training accuracy and a very poor validation accuracy. Theoretically, residuals help with continuous learning between convolution layers so it makes sense that they helped our model so much. Overall we found that with residual training we had an accuracy of 87.27% and without the residual block in our architecture we had 63.03% (note that because of this reason, we tested our preprocessed vs unprocessed models with the code including residuals). Something we noticed in both graphs is that after some number of epochs (around 7) our model training accuracy decreased. We did not notice this in model 1 because we had to use fewer epochs to save time. However, we believe this is likely due to over-fitting of the model. Since model 2 only trained on different types of cancerous skin lesions (melanoma and basal cell carcinoma), it had the least amount of data, so we are very happy with the results.

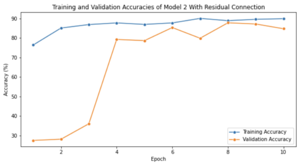


Figure 6. Training and Validation Accuracy for Model 2, with Residual Connections

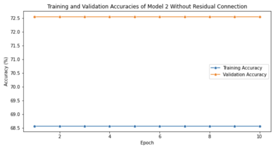


Figure 7. Training and Validation Accuracy for Model 2, without Residual Connection

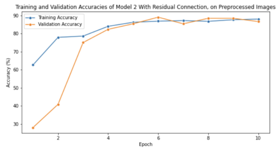


Figure 8. Model 2 with Residual Connections, Trained on Preprocessed Images

4.3. Model 3 Results

Model 3 was more difficult because we were not using binary classification. Our initial model 3, before hyperparameter tuning, without residuals, and using preprocessing had results with test accuracy in the 40% range, so model 3 had the best improvement from it's original state. We used what we had learned in model 2 and applied it to model 3. To get the most accurate classification we decided to not use preprocessing, but we did use residuals. In terms of computational complexity model 3 was still very slow on Colab (though slightly less than model 1 due to a slightly smaller dataset size), again this forced us to use 3 epochs. The final test accuracy was 84.13%. Overall, we are happy with the results from model 3, especially since this was our more difficult model to implement. In addition to the graphs of each model's accuracy over time, we also have examples in the appendix highlighting how our code would work for a user inputting an image of a skin abnormality, we have examples in Appendix A of a situation using both model 2 and model 3.

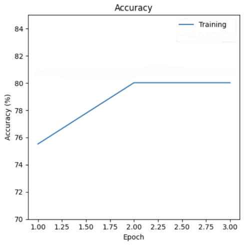


Figure 9. Accuracy of Model 3, with Residual Connection

4.4. LLM Results

Our findings reveal that while Gemini can provide high-level advice, domain-specific medicine-chat provides more helpful and focused medical advice. Sample prompts and model responses for both medicine-chat and Gemini are shown in the table 2 and 3.

5. Conclusion

In real-world applications, our unique three-model approach would be significantly beneficial than using a single model for all classifications. By modularizing the tasks, our architecture not only reduces the risk of overfitting, but also improves the accuracy of the results by focusing on particular features of malignant and benign skin abnormalities. This method will allow people to determine the type of lesion more accurately before starting further tests and procedures. Precision of this level is extremely beneficial because it can significantly reduce the amount of time and money spent in diagnosis, allowing for more patient clarity and faster diagnose in the medical process. Our model's straightforward classification will enable communication between patients and health care professionals, leading to a swifter decision allowing for early stage cancer detection.

As a team, we managed to accomplish our initial goals and create three original model CNN network. However, given extra time, computation and GPU resources, we have several ideas and implementations that we would have loved to implement to expand and refine our models.

Firstly, we would have explored applying and integrating Vision Transformers (ViT) [7] models such as vit-base-patch16-224-in21k [5, 24], to enhance our existing 3-model system's predictions. ViT models are known for showing superb performance in image classification and integrating such models would greatly improve the final output of our model by capturing more complex skin patterns. Secondly, we would have explored expanding our pre-processing techniques beyond rotation and Gaussian blurring (e.g. Image flipping and color normalization). We would have incorporated libraries such as scikit-image [23] to aid us in more complex preprocessing if needed.

Thirdly, we would like to investigate the impact of skin color on the results produced by our model. As of now, almost every image in the HAM 10000 dataset is set on a light skinned patient. We would also explore integrating demographic information and medical history of the patient to create a more personalized result. Finally, although not directly related to the models that we created, we would have liked to do more tuning on the Medicine Chat models to craft a better response for the specific patient.

No project is without its own set of challenges, and ours was no exception. One key challenge that we faced during the testing of our models was that our training dataset was biased towards non-cancerous moles. Therefore, our initial models over-fitted heavily, drastically reducing the testing accuracy. We combatted this by writing code to down-sample the dataset to allow for less bias in the training set.

Another major hurdle that we faced was limited access to powerful GPU resources. Google Colab's runtime limitation of 3 hours was a major obstacle because our model took roughly 4 hours to train due to the large dataset. Due to this, we had to come up with careful time management strategies and write efficient code to reduce the training times.

References

[1] T. Alkarakatly, S. Eidhah, M. Al-Sarawani, A. Al-Sobhi, and M. Bilal. Skin lesions identification using deep convolutional neural network. In *2019 International Conference on Advances in the Emerging Computing Technologies (AECT)*, 2020. 1

[2] C. Calderón, K. Sanchez, S. Castillo, and H. Arguello. Bilsk: A bilinear convolutional neural network approach for skin lesion classification. *Computer Methods and Programs in Biomedicine Update*, 1:100036, 2021. 1

[3] Daixuan Cheng, Shaohan Huang, and Furu Wei. Adapting large language models via reading comprehension. In *The Twelfth International Conference on Learning Representations*, 2024. 4

[4] Jan Clusmann, Fiona R Kolbinger, Hannah Sophie Muti, Zunamys I Carrero, Jan-Niklas Eckardt, Narmin Ghaffari Laleh, Chiara Maria Lavinia Löffler, Sophie-Caroline Schwarzkopf, Michaela Unger, Gregory P Veldhuizen, et al. The future landscape of large language models in medicine. *Communications medicine*, 3(1):141, 2023. 4

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[6] Nick DiSanto. Isic melanoma dataset. IEEE Dataport, 2023. 2

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[8] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027, 2021. 4

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[10] Lindsey S. Huff, Cassandra A. Chang, Jeffrey F. Thomas, Margaret Cook-Shimanek, Paula Blomquist, Nellie Konnikov, and Robert P. Dellavalle. Defining an acceptable period of time from melanoma biopsy to excision. *Dermatology Reports*, 4(1):e2, 2012. 1

[11] Farzana Huq, Mio Nakamura, Katherine Black, Hannah Chubb, and Yolanda Helfrich. Association of dermatology wait times with insurance coverage in michigan. *The American journal of managed care*, 26(10):432–437, 2020. 1

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 1

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 3

[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

[15] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 2, 3

[16] Ed Phippen. New survey: Physician appointment wait times getting longer, January 5 2023. 1

[17] Lutz Roeder. netro. 4

[18] American Cancer Society. Survival rates for melanoma skin cancer. https://www.cancer.org/cancer/types/melanoma-skin-cancer/detection-diagnosis-staging/survival-rates-for-melanoma-skin-cancer-by-stage.html, 2024. 1

[19] Gemini Team. Gemini: A family of highly capable multimodal models, 2024. 4

[20] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci Data*, 5:180161, 2018. 2

[21] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018. 2

[22] L. Udristoiu, A. E. Stanca, A. E. Ghenea, C. M. Vasile, M. Popescu, Ș. C. Udristoiu, A. V. IACOB, S. Castravete, L. G. Gruionu, and G. Griuionu. Skin diseases classification using deep learning methods. *Current Health Sciences Journal*, 46(2):136–140, 2020. 1, 2

[23] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014. 6

[24] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020. 6

[25] Joyce Xia, Alice J Tan, Bianca Biglione, Bethany Cucka, Lauren Ko, Emily D Nguyen, Charbel C Khoury, Malcolm K Robinson, Sagar U Nigwekar, and Daniela Kroshinsky. Nephrogenic calciphylaxis arising after bariatric surgery: a case series. *American Journal of Nephrology*, 55(2):196–201, 2024. 1

A. Appendix

Project Code

https://github.com/bowenyi-umich/Skin-Cancer-Detection.git

Model Architecture Visualization

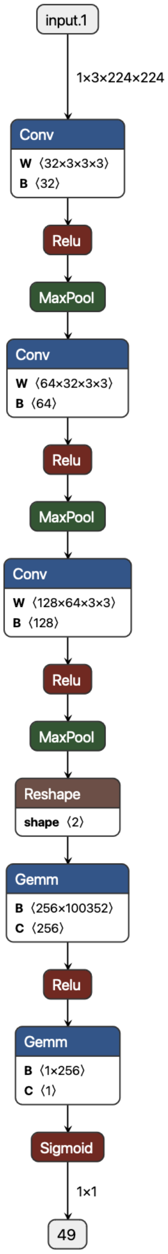


Figure 10. Model 2 without Residual Connection

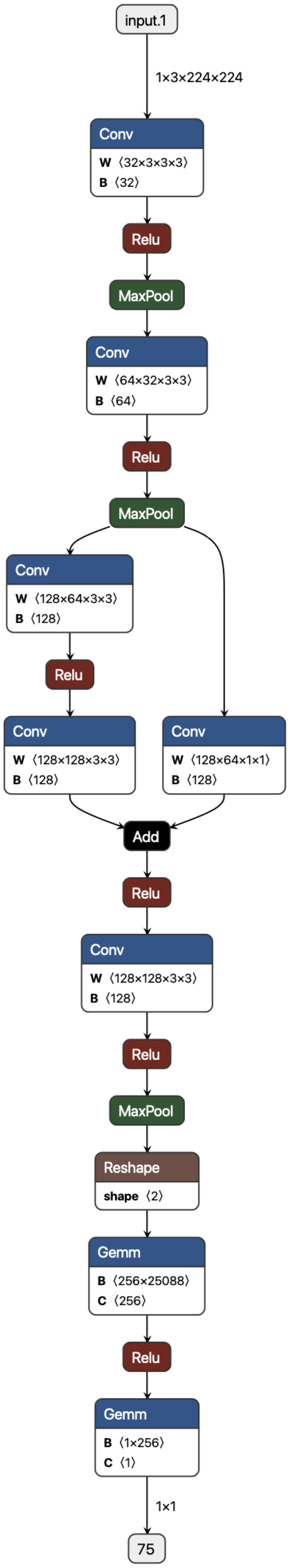


Figure 11. Model 2 with Residual Connection

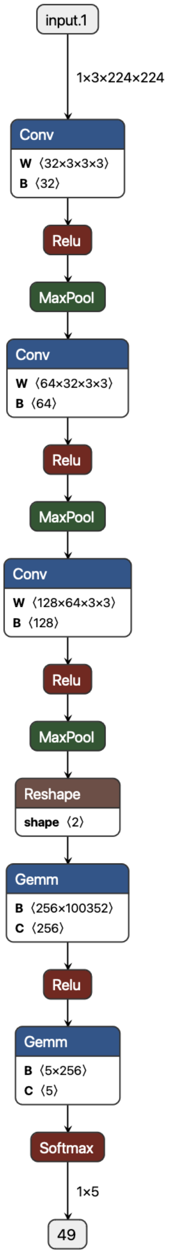


Figure 12. Model 3 without Residual Connection

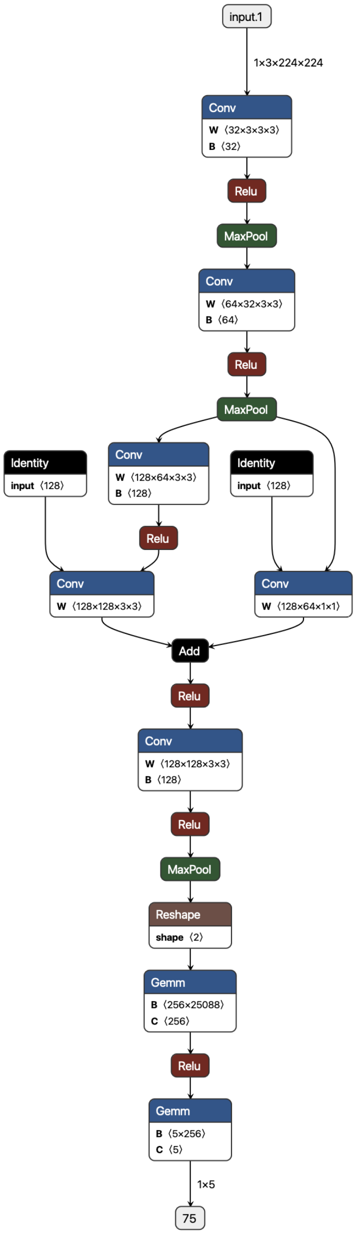


Figure 13. Model 3 with Residual Connection

Prompt	Model Response
'I am a patient with melanoma disease on my behind ear area of my body. I am 18 years old and my sex is male. Apart from immediately consulting a doctor, what should my next steps be?'	As a patient with melanoma disease, it is important to take immediate action to ensure that the cancer is diagnosed and treated as soon as possible. Here are some steps you can take: 1. Consult a doctor: As you mentioned, the first step is to consult a doctor. You can start by making an appointment with your primary care physician or a dermatologist. They will examine your skin and perform a biopsy to confirm the diagnosis. 2. Get a second opinion: If you are not satisfied with the diagnosis or treatment plan, you can seek a second opinion from another doctor. You can ask your primary care physician or dermatologist for a referral to a specialist.

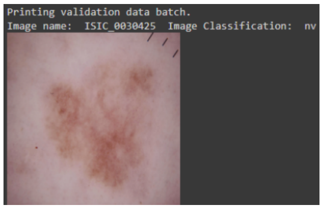
Table 2. Sample Prompts and Model Responses for Medicine-chat

Prompt	Model Response
'I am a patient with melanoma disease on my behind ear area of my body. I am 18 years old and my sex is male. Apart from immediately consulting a doctor, what should my next steps be?'	**Additional Steps for Melanoma Management:** **1. Gather Medical Information:** * Obtain a copy of your medical records, including pathology reports and treatment plans. * Research reputable sources for information on melanoma and its treatment options. * Consult with a dermatologist, oncologist, or other healthcare professionals specializing in melanoma. **2. Seek Emotional Support:** * Connect with support groups or online communities for individuals with melanoma. * Talk to family, friends, or a therapist about your emotions and concerns. * Practice stress-reducing techniques such as meditation, yoga, or spending time in nature.

Table 3. Sample Prompts and Model Responses for Google Gemini

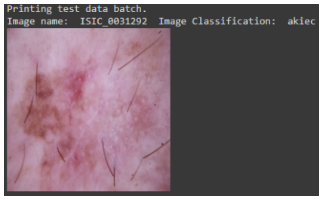
Sample Prompts and Model Response

Non-preprocessed Images



```
Printing validation data batch.
Image name:  ISIC_0030425  Image Classification:  nv
```

Figure 14. Non-preprocessed Image 1



```
Printing test data batch.
Image name:  ISIC_0031292  Image Classification:  akiec
```

Figure 15. Non-preprocessed Image 2

Preprocessed Images Samples

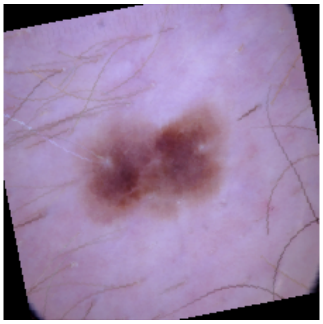


Figure 16. First Sample of Preprocessed Image

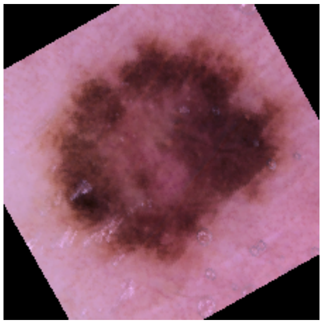


Figure 17. Second Sample of Preprocessed Image

Model Pipeline Example output

(given an image of skin cancer)

```
the given skin abnormality is:
very likely malignant
probably of melanoma: .7707
```

Your medical advice is (LLM powered):

(given an image of a benign mole)

```
the given skin abnormality is: most likley output: nv
with probably: .6607
```

Your medical advice is (LLM powered):