

TRANSFER LEARNING CONVOLUTIONAL NEURAL NETWORKS FOR SKIN CANCER CLASSIFICATION

Mansurova M. , Nauryzbayev A. ¹, Vegh L. 

Abstract Skin cancer is a widespread disease that requires accurate classification methods. Deep learning models can support the classification of different skin cancer classes. This study evaluates several Convolutional Neural Network (CNN) models for this task. Transfer learning is applied to adapt pretrained CNN architectures to skin lesion classification. The HAM10000 dataset is used for model training and evaluation. Several CNN architectures are compared using multiple performance measures. The evaluation considers macro accuracy, network depth, parameter count, and epoch duration. These measures are used to assess classification performance and computational requirements. The results show that most evaluated models achieve satisfactory classification accuracy. No clear evidence of severe overfitting is observed for most models. Among the evaluated architectures, ResNet18 provides the most suitable results. It achieves a favorable balance between classification performance and computational requirements. Therefore, ResNet18 is identified as the most suitable model for this dataset. The findings support transfer learning for automated skin cancer classification.

Keywords: Cutaneous Malignancy, Deep Learning, Skin Cancer, Information Systems, Modeling and Simulation, Applied Mathematics, Transfer Learning, Data Augmentation, Convolutional Neural Networks, Medical Image Classification, Medical Informatics.

AMS Mathematics Subject Classification: 68T05, 68T01, 92B20.

DOI: 10.32523/2306-6172-2026-14-3-104-114

1 Introduction

Globally one of the most widespread and dangerous types of cancer diseases are skin cancers. In particular, the melanoma type skin cancer incidence rate is projected to increase to 96 000 in 2040 from 53 000 in 2020 [1]. Skin cancer is a globally widespread disease, so its accurate and early detection is significant to reduce rates of mortality. There are two major groups of skin cancer types: Melanoma skin cancer (MSC) and Non-melanoma skin cancers (NMSCs). However, there are also many rare skin cancer types. Furthermore, MSC can be divided into sub types such as superficial spreading melanoma (SSM), lentigo maligna melanoma (LMM), nodular melanoma (NM) and acral lentiginous melanoma (ALM). There are a considerable number of different types of skin cancer [2]. However, in this paper only classification of only 7 types of skin cancers from the HAM10000 dataset was considered.

Screening for skin cancer is time consuming and the global incidence rate of skin cancer grows [3]. As a result, workload on dermatologist doctors globally increases. Therefore, automation of skin cancer type classification is a relevant issue. To develop AI models that can classify types of cancer many datasets were developed. Among these datasets HAM10000 is one of the biggest skin cancer datasets. This dataset has 7 classes and 10015 images of skin lesions [4]. This dataset allows to train deep neural networks for skin cancer classification.

¹Corresponding Author.

Among many different architectures of deep neural networks the CNNs and ViTs are commonly applied for image classification. Many open source models achieved state of the art accuracy on the ImageNet dataset. Therefore, these models are adaptable for the HAM10000 dataset using transfer learning. The transfer learning method allows the use of many different models. However, unlike CNNs the ViT models require bigger datasets in terms of image size. It means, to successfully train ViT models without overfitting it is required to have large initial dataset or do pretraining on large datasets. In contrast, the CNN models have strong inductive bias. This allows the use of CNN models for small and medium size dataset [5]. Therefore, in this paper only open source CNN models were selected to be adapted for the HAM10000 dataset using transfer learning. Consequently, in this work all training experiments were done using the following models: AlexNet [6], GoogleNet [7], ConvNeXt-tiny [8], InceptionV3 [9], VGG19 [10], DenseNet121 [11], DenseNet201 [11], ResNet18 [12], ResNet34 [12], ResNet50 [12], and ResNet152 [12]. In the next sections the literature review, data preparation, data processing, data augmentation, architecture overview of models, training pipeline description, performance metrics description, experimental environment and results are presented.

2 Literature review

Some researchers tried to apply classical machine learning models such as decision tree (DT), random forest (RF), support vector machine (SVM), k-nearest neighbor (KNN), logistic regression (LR), gaussian naïve bayes (NB), linear discriminant analysis (LDA). The decision tree (DT) and random forest (RF) had micro accuracy 68% and 87% respectively. However, the micro accuracies of all other mentioned classical machine learning models were below 60%. Authors also trained a custom CNN model. This model achieved micro accuracy of 94%. They downsampled a dataset from 10015 to 700 images. It means, they took only 100 images from each class. They divided the dataset into 80% for training and 20% for testing. Before training models they applied geometric data augmentation doubling images in each class. As a result, the total size of each class in the training set after augmentation was 200 images. Also, when training their custom CNN model they used the following hyper parameters: batch size of 32, Adam optimizer, categorical crossentropy loss and 150 number epochs [13]. Overall, this work presents a successful application of classical machine learning models and of the custom CNN model on the HAM10000 dataset. However, the authors only used 700 images from the entire dataset of 10015 images. As a result, they did not use more than 9000 images that offer a significant amount of unique features. The resulting models without learning these features might generalize poorly, because they learned from a small dataset.

In other works authors used CNN models that are pretrained on ImageNet along with their custom models on HAM10000 dataset for comparison [14]. This application of transfer learning allows objective comparison and establishes a baseline performance. For example, after developing the custom model SkinNet-14, authors compared its performance to the CNN models. They used the following CNN models: VGG16, VGG19, ResNet50, ResNet50V2, ResNet152 and MobileNet. Their results demonstrate micro accuracy of these CNN models vary from 43.42% to 81.21%. The best CNN model was VGG16 with 81.21% micro accuracy and 82.1% F1 score. The SkinNet-14 uses CCT, a ViT-convolution hybrid model and it achieves 97.85% micro accuracy and 97.92% F1 score. All models in this study were trained on 32 by 32 images for 200 epochs. Before training images were preprocessed and the dataset was augmented. The data preprocessing step included application of morphological opening, NLMD, CLAHE, Gaussian Blur, and resize of initial images to 224x224. The dataset was augmented using geometric, photometric and elastic deformations. The dataset size in-

creased significantly from 10015 to 50075. To train models authors used the following hyper parameters: Adam optimizer, batch size of 128, ReLu activation function and categorical cross-entropy loss [3]. In general, authors in this paper demonstrated a custom hybrid architecture that has elements of CNN and ViT architectures. The proposed model SkinNet-14 presents high accuracy and F1 score. However, the image size of 32 by 32 loses important information and features such as small veins, skin textures and other unique small details.

In another work, authors used CNN models along with their custom model to achieve best performance for each model using ant colony algorithm for hyper parameter optimization. They used the following models: XceptionNet, DenseNet201, DenseNet169, GoogleNet, DenseNet121, MobileNetV2 and Custom CNN. Their best performing model among selected ones was XceptionNet. This model achieved 96.5% micro accuracy and 97% F1 score. They used 224 by 224 images with minimal data preprocessing. They used simple geometric data augmentations such as flipping, rotating, and scaling of images. After data augmentation they increased the total number of images to 38 569 [15]. In general, the authors selected the XceptionNet as their best model. This model achieved high micro accuracy and F1 score. However, authors of another paper analyzed the HAM10000 dataset and found that some images in the dataset were duplicated [16]. This means the same skin cancer image was photographed from different angles and magnifications. This paper did not mention that this dataset was split into training and testing subsets using lesion IDs in their work.

Application of transfer learning in the field of AI in medicine is a widely accepted method of training models to classify medical images. For the HAM10000 dataset transfer learning is also commonly used by researchers. Using transfer learning authors of discussed papers were able to finetune open source models using different methods.

3 Data preparation

3.1 Overview of the dataset

The dataset used in this study is HAM10000. This dataset contains 10015 of dermatoscopic images of skin pigmented lesions. There are 7 classes of skin lesions namely: actinic keratoses and intraepithelial carcinoma / Bowen’s disease (akiec), basal cell carcinoma (bcc), benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, bkl), dermatofibroma (df), melanoma (mel), melanocytic nevi (nv) and vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, vasc) [4]. The table 1 below shows the number of images in each class.

Table 1: Number of images in each class of the dataset

Class Name	Number of Images
akiec	327
bcc	514
bkl	1099
df	115
mel	1112
nv	6705
vasc	142

Table 1 shows that the dataset is imbalanced. It means some classes are overrepresented, while at the same time other classes are underrepresented. For example, the nv class has 6705

images and the df class has 115 images. As a result, nv class has 58 times more images than df class.

3.2 Data preprocessing and augmentation.

Before the data preprocessing stage dataset was divided into train, validation and test subsets. As a result, the dataset was divided into 70% train, 10% validation and 20% test. The splitting of the dataset was done using lesion IDs. This method eliminated data leakage between resulting training, validation, and test datasets. The resulting datasets were shown on table 2.

Table 2: The train, validation and test datasets sizes

Class Name	Number of Images in Train	Number of Images in Validation	Number of Images in Test
akiec	230	31	66
bcc	366	52	96
bkl	774	109	216
df	76	13	26
mel	778	117	218
nv	4679	685	1341
vasc	99	17	26

After dividing the dataset into train, validation and test sets, the data preprocessing and data augmentation was applied. During the data preprocessing stage initial images were cropped, consequently changing image sizes from 600x450 to 450x450. Then all images were resized into 224x224. In some of the reviewed papers authors reported that removing artifacts, applying Gaussian blur, CLAHE, NLMD, or Photometric data augmentation increased overall accuracy of deep neural networks. However, in this study we empirically found that CLAHE, Photometric data augmentation and removal of artifacts did not increase accuracy of CNN models. Additionally, Gaussian blur significantly decreased accuracy of all models. We assume that performing Gaussian blur operations would distort or misrepresent key features of the skin lesions. Consequently, in this paper we only used image cropping and resizing during the data preprocessing stage. Furthermore, in order to keep important features, the data augmentation stage we only used geometrical data augmentations and random sharpness adjustments of sharpness. The geometric data augmentations included random horizontal and vertical flips, random rotations and translations. The data preprocessing was applied to train, validation and test sets, while data augmentation was applied only to the train set. In the data loader of the training set each class was upsampled depending on their initial size. The nv class that had 4679 images was not upsampled, while akiec, bcc, bkl, df, mel and vasc had the multiplication factor of 8, 5, 2, 26, 2, and 20. After data augmentation all classes except nv got upsampled up to approximately 2000 images. As a result, the total number of images in the train dataset increased from 7002 to 15409. The resulting train, validation and test dataset sizes after data augmentation are shown in table 3.

All classes except nv have nearly 2000 images after data preprocessing and data augmentation stages. The images were increased to 2000 images to conserve computational resources and to not over augment classes that had a small number of images initially. The table 3 shows that df and vasc classes that initially 76 and 99 images after data augmentation have approximately 2000 images. It is possible to increase the number of images higher than 2000, but it would lead to overfitting on minority classes namely df and vasc during training. Overall, we used moderate data augmentation and did not increase the dataset significantly. However, to

Table 3: The size of train, validation and train datasets after data augmentation

Class name	Number of Images in Train	Number of Images in Validation	Number of Images in Test
akiec	1840	31	66
bcc	1830	52	96
bkl	1548	109	216
df	1976	13	26
mel	1556	117	218
nv	4679	685	1341
vasc	1980	17	26

compensate for that and prevent overfitting we used a weighted categorical crossentropy loss function with label smoothing.

4 Methodology

4.1 Model Architecture overview

Some authors of reviewed papers used pre-trained models on ImageNet dataset to finetune and compare them with their custom models. In this work a similar approach was applied. These CNN models that were pre-trained on ImageNet were used for skin cancer type classification. The following open source models namely AlexNet, GoogleNet, ConvNeXt-tiny, InceptionV3, VGG19, DenseNet121, DenseNet201, ResNet18, ResNet34, ResNet50, and ResNet152 were used via transfer learning to represent different depth and parameter size. The depth and parameter size of used models are shown below in table 4.

Table 4: Original parameter size and depth of models

Model Name	Parameter Size (millions)	Depth
AlexNet	61.0	8
GoogleNet	6.6	22
InceptionV3	27.2	42
VGG19	143.0	19
ResNet18	11.7	18
ResNet34	21.8	34
ResNet50	25.6	50
ResNet152	60.5	152
ConvNeXt-tiny	28.6	27
DenseNet121	8.0	121
DenseNet201	20.0	201

These models are pretrained on the Imagenet dataset, and then they were fine-tuned. The fine-tuning was done without freezing weights. Before performing fine-tuning classifier heads of CNN architecture models were modified, because originally they had 1000 classes for Imagenet dataset. In fine-tuning 3 different classifier head architectures that fit the number of classes in the HAM10000 dataset were used. The structural information of classification heads are shown in table 5.

After modifying the classifier head of models the depth and number of weights changed from original numbers. However, the overall architecture of CNN models remained unchanged, while only classification heads were modified. After fine-tuning these models were tested and

Table 5: Custom classification head structures

Classifier Type	Number of Layers	Number of Output Classes	Number of Neurons in Each Layer
Small	1	7	(7)
Medium	3	7	(256, 128, 7)
Big	5	7	(256, 128, 128, 128, 7)

compared using evaluation metrics.

4.2 Architecture evaluation metrics

The given architectures were compared on following metrics: accuracy, precision, recall, F1 score, specificity and area under curve (AUC). All metrics except AUC have both the version of overall value and equally weighted across all classes. In this study we mainly focus on an equally weighted version of these metrics. The equally weighted metrics show more objectively if models learned all classes to the sufficient degree without underfitting or overfitting. For example, it is possible to achieve accuracy above 90%, while some classes will have accuracy below 50%. Therefore, we also additionally measured the accuracy of each class separately. Overall, the goal is to find the model for CNN architectures with the least possible number of parameters and depth that achieves sufficient accuracy for all 7 classes of HAM10000 dataset.

4.3 Training Pipeline Description

The training loop runs for 50 epochs. Each epoch all performance metrics are evaluated both for training set and validation set. If validation loss is not improving for 5 epochs the learning rate decreases. The learning rate initialized at 0.0001. Also, if validation loss does not improve for 10 epochs the early stopping of training occurs. To measure loss a weighted categorical crossentropy loss function with label smoothing of 0.3 was used. The AdamW optimizer was used along with weight decay of 0.001. The selected CNN models are fine tuned on the training dataset. During training, performance metrics are saved every training epoch. After training, the model weights are saved to disk for future use. All collected epoch metrics are also saved to a table on disk. After training all model data are tested on a test dataset. Test results are also saved to a table. After all data is collected, it is aggregated into a single database and its performance is analyzed. All models were subject to the same hyperparameters, such as batch size, number of epochs, learning rate, loss functions, and optimizer. All hyperparameters in the studies were chosen to ensure that all models could run without errors during training. For example, the batch size was limited to 128, as some models were too large, overflowing the GPU’s video memory and causing errors. While some models were small and could have been run with batch sizes up to 512, this would have made the comparison biased. In particular, the fact that some models could have been run with batch sizes of 512 and trained more quickly demonstrates the experimental relevance and scientific importance of finding models with a low number of weights and shallow depth.

5 Experimental environment

To train deep learning models, we used resources from Paperspace’s cloud computing platform. The cloud server that was used for the experiments had an RTX A6000 GPU with 48 GB of video memory, 45 GB of RAM, and an 8-core CPU. The latest version of the Linux operating system was pre-installed on a cloud server. We created a virtual environment on the server computer and Python version 3.12, Pytorch 2.11, Jupyter Notebook, and other

Python libraries were installed in it. The entire training pipeline runs from the main Jupyter notebook, while all functions and configurations are saved in corresponding Python files.

6 Results

In this section of the paper the results of experiments are presented. As presented earlier in table 5 each model backbone has 3 structurally different heads. Therefore, each model backbone has 3 different results after it has been fine tuned and tested using three different heads. After conducting experiments with these models we obtained results for all of them. However, to present results compactly we show only one result for each model. This was achieved by keeping only the best performing classifier head. The main performance metric that we used is macro accuracy as it objectively represents how well model learned patterns of each class. The type of the head in the following tables are specified in the classifier type column. Table 6 illustrates general performance metrics such as micro accuracy, macro recall, macro specificity, macro F1 score, macro precision and AUC.

Table 6: Performance of models on general metrics

Model Name	Classifier Type	Micro Accuracy	Macro Recall	Macro Specificity	Macro F1 Score	Macro Precision	AUC
AlexNet	Big	74.4	69.9	95.2	61.5	56.2	89.5
GoogleNet	Small	71.3	58.6	94.5	52.4	50.3	83.9
InceptionV3	Small	66.1	54.8	94.0	46.1	41.8	81.0
VGG19	Medium	80.3	73.7	95.7	68.1	64.0	89.2
ResNet18	Small	82.1	72.6	96.1	68.8	66.0	92.6
ResNet34	Small	82.1	73.0	96.0	71.0	70.4	91.2
ResNet50	Big	81.3	71.5	95.7	69.5	68.5	87.5
ResNet152	Small	79.7	70.8	95.9	67.3	64.9	93.3
ConvNeXt-tiny	Small	82.6	76.6	96.1	73.9	72.1	92.5
DenseNet121	Small	82.4	77.7	96.2	71.3	66.9	93.1
DenseNet201	Small	83.2	73.0	96.0	71.9	71.2	90.1

The results in table 6 show that the majority of models achieved micro accuracy above or close to 80%. Similarly, the majority of models have above 90% of macro specificity and AUC. In the table 7 below test accuracies of all seven classes of models are demonstrated. Also, in table 7 the macro accuracy is presented in the last column.

The results in table 7 show that majority models achieved sufficient accuracy across all classes in order to provide macro accuracy of 70% or above. However, the results in table 7 indicate that macro accuracy of GoogleNet and InceptionV3 are 58.6% and 54.8% respectively. It means these two models have small macro accuracy. The macro accuracy metric objectively confirms that models can differentiate between classes and demonstrates that there is no overfitting or underfitting.

For a more detailed analysis of the experimental data, four graphs were constructed. The first graph (Fig. 1) shows the dependence of macro accuracy on the neural network depth. All models except GoogleNet and InceptionV3 with depth below 50 layers achieve macro accuracy of 70% or above.

The second graph (Fig. 2) shows the dependence of macro accuracy on the neural network parameter count. All models except GoogleNet and InceptionV3 with parameter count below

Table 7: Test accuracy of models across classes

Model Name	Classifier Type	Class Accuracy							Macro Accuracy
		akiec	bcc	bkl	df	mel	nv	vasc	
AlexNet	Big	71.2	71.0	60.0	61.5	58.0	80.0	88.4	70.0
GoogleNet	Small	36.4	55.2	43.1	50.0	73.4	78.8	73.1	58.6
InceptionV3	Small	65.2	45.8	44.0	23.1	59.6	72.9	73.1	54.8
VGG19	Medium	69.7	76.0	65.7	73.1	55.0	87.6	88.5	73.7
ResNet18	Small	57.6	78.1	70.8	65.4	58.3	89.4	88.5	72.6
ResNet34	Small	59.1	80.2	72.2	61.5	56.4	89.3	92.3	73.0
ResNet50	Big	66.7	59.3	67.1	69.2	60.1	89.4	88.5	71.5
ResNet152	Small	54.5	78.1	75.0	69.2	60.1	85.3	73.1	70.8
ConvNeXt-tiny	Small	62.1	75.0	70.0	88.4	64.2	89.0	88.4	76.6
DenseNet121	Small	62.1	83.3	68.5	88.5	60.1	88.9	92.3	77.7
DenseNet201	Small	65.2	71.9	70.0	73.1	54.1	91.9	84.6	72.9

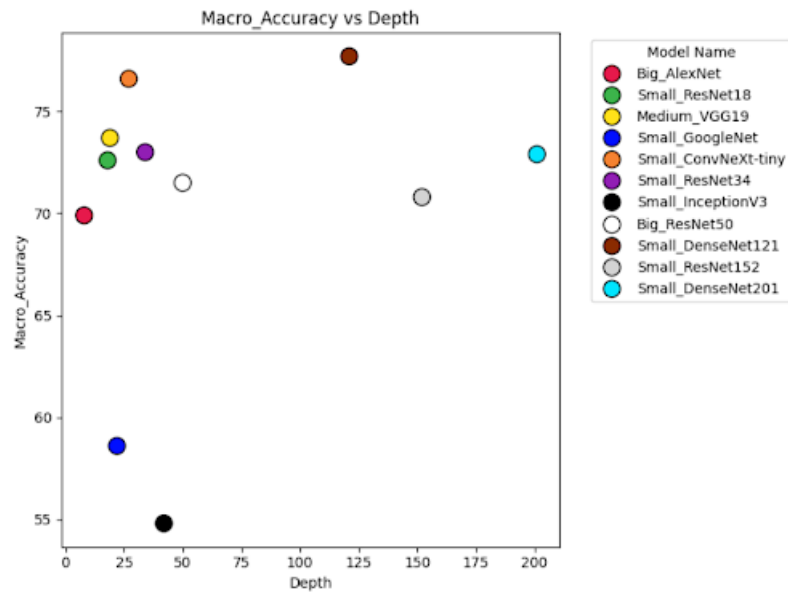


Figure 1: Dependence of macro accuracy from depth

40 million achieve macro accuracy of 70% or above. According to table 7, figure 1 and figure 2, it was decided to remove these 2 models from further consideration.

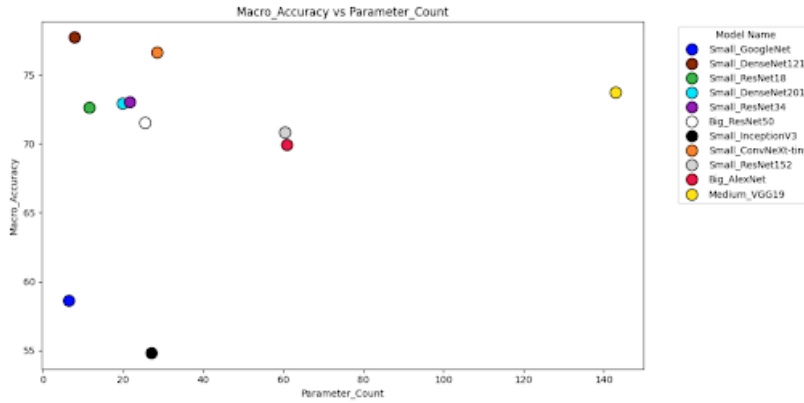


Figure 2: Dependence of macro accuracy from parameter count

In order to do advanced analysis of the remaining 9 models, the dependence of the duration of epochs on the depth and number of parameters of the neural networks was additionally considered.

The third graph (Fig. 3) shows the dependence of epoch duration on the neural network depth. This dependence shows that 3 models namely AlexNet, ResNet18 and ResNet34 have the smallest epoch duration. Therefore, only these 3 models were kept for further consideration.

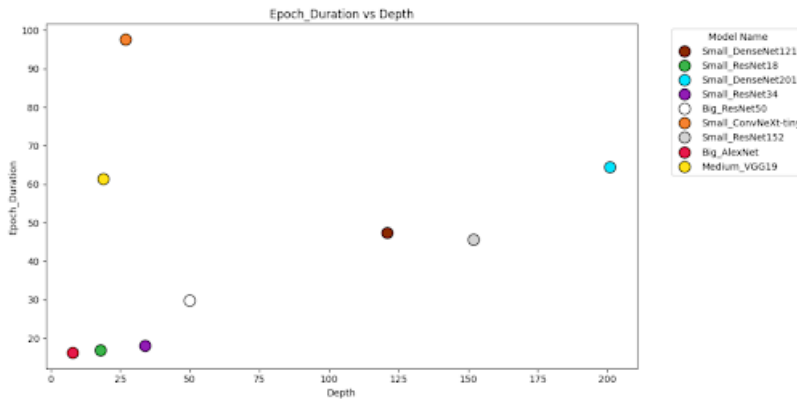


Figure 3: Dependence of epoch duration from depth

For further clarification, the dependence of the duration of epochs on the number of parameters (Fig. 4) was constructed. This graph also shows that 3 models namely AlexNet, ResNet18 and ResNet34 have the smallest epoch duration. However, AlexNet has above 60 million parameters. ResNet18 and ResNet34 have a number of parameters below 22 million. Therefore, for further consideration only 2 models were kept namely ResNet18 and ResNet34.

The ResNet18 and ResNet34 have 11 and 21 million parameters respectively. From all models that were used in this paper these two models are among the smallest models in terms of parameter count and depth. Also, these models have a small average epoch duration. Therefore, these two models are considered to be the most efficient models, because their training time and computational cost are the smallest compared to other considered models.

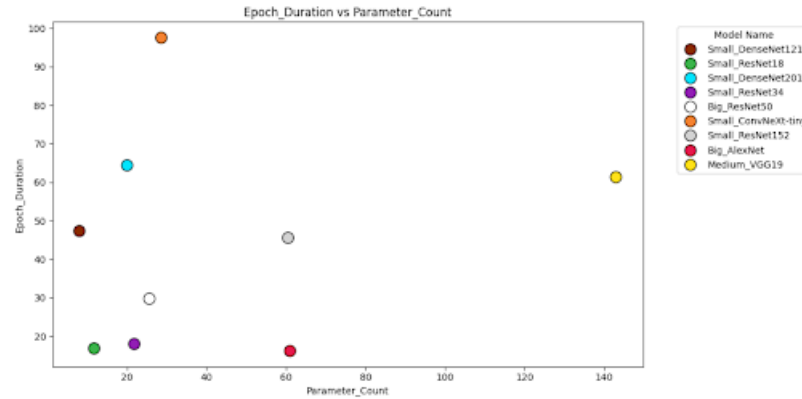


Figure 4: Dependence of epoch duration from parameter count

Furthermore, the ResNet18 model has almost 2 times less number of parameters and depth compared to ResNet34. Consequently, the ResNet18 is the best model among all considered models.

7 Conclusion

Several CNN architectures are evaluated for skin cancer classification. Transfer learning is applied to pretrained CNN models using the HAM10000 dataset. The results show satisfactory classification performance across the investigated classes. Most models achieve acceptable accuracy without clear signs of severe overfitting. Pretrained features therefore support classification across different skin lesion classes. Model selection is based on more than classification accuracy. Several CNN models achieve similar general accuracy values. However, their computational characteristics differ across the evaluated models. Therefore, macro accuracy is also considered during model evaluation. Network depth, parameter count, and average epoch duration are also assessed. These metrics provide information about accuracy and computational requirements. Among the evaluated models, ResNet18 provides the most suitable results. It achieves a suitable balance between accuracy and model complexity. Its moderate depth also supports efficient model training. Its parameter count remains suitable for the classification task. Therefore, ResNet18 is selected for the HAM10000 dataset. The results support transfer learning for automated skin cancer classification. Further validation should use additional datasets and independent clinical data.

References

- [1] Wang, M., Gao, X., Zhang, L., *Recent global patterns in skin cancer incidence, mortality, and prevalence*, Chin Med J (Engl), Vol. 138, Issue 2 (2025), pages 185-192. <https://doi.org/10.1097/CM9.0000000000003416>
- [2] Roky, A.H., Islam, M.M., Ahasan, A.M.F., Mostaq, M.S., Mahmud, M.Z., Amin, M.N., Mahmud, M.A., *Overview of skin cancer types and prevalence rates across continents*, Cancer Pathog Ther, Vol. 3, Issue 2 (2024), pages 89-100. <https://doi.org/10.1016/j.cpt.2024.08.002>
- [3] Al Mahmud, A., Azam, S., Khan, I.U. et al., *SkinNet-14: a deep learning framework for accurate skin cancer classification using low-resolution dermoscopy images with optimized training time*, Neural Comput & Applic, Vol. 36 (2024), pages 18935-18959. <https://doi.org/10.1007/s00521-024-10225-y>
- [4] Tschandl, Philipp, *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*, Harvard Dataverse (2018). <https://doi.org/10.7910/DVN/DBW86T>

- [5] Dosovitskiy, A. et al., *An image is worth 16x16 words: Transformers for image recognition at scale*, OpenReview. <https://openreview.net/forum?id=YicbFdNTTy>
- [6] Krizhevsky, A., Sutskever, I., Hinton, G.E., *ImageNet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems, Vol. 25 (2012), pages 1097–1105.
- [7] Szegedy, C. et al., *Going deeper with convolutions*, 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA (2015), pages 1-9. <https://doi.org/10.1109/CVPR.2015.7298594>
- [8] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., Xie, S., *A ConvNet for the 2020s*, 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA (2022), pages 11966-11976. <https://doi.org/10.1109/CVPR52688.2022.01167>
- [9] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., *Rethinking the Inception Architecture for Computer Vision*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA (2016), pages 2818-2826. <https://doi.org/10.1109/CVPR.2016.308>
- [10] Simonyan, K., Zisserman, A., *Very Deep Convolutional Networks for Large-Scale Image Recognition*, Computer Science (2014). <https://doi.org/10.48550/arXiv.1409.1556>
- [11] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., *Densely Connected Convolutional Networks*, 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA (2017), pages 2261-2269. <https://doi.org/10.1109/CVPR.2017.243>
- [12] He, K., Zhang, X., Ren, S., Sun, J., *Deep Residual Learning for Image Recognition*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA (2016), pages 770-778. <https://doi.org/10.1109/CVPR.2016.90>
- [13] Shetty, B., Fernandes, R., Rodrigues, A.P., Chengoden, R., Bhattacharya, S., Lakshmana, K., *Skin lesion classification of dermoscopic images using machine learning and convolutional neural network*, Sci Rep, Vol. 12, Issue 1 (2022), page 18134. <https://doi.org/10.1038/s41598-022-22644-9>
- [14] Adebisi, A., Abdalnabi, N., Hoffman, E.S., et al., *Accurate Skin Lesion Classification Using Multi-modal Learning on the HAM10000 Dataset*, medRxiv (2024). <https://doi.org/10.1101/2024.05.30.24308213>
- [15] Sarhan, A.M., Ali, H.A., Yasser, S. et al., *Achieving high-accuracy skin cancer classification with deep learning optimized by ant colony algorithm*, Journal of Electrical Systems and Inf Technol, Vol. 12 (2025), article 49. <https://doi.org/10.1186/s43067-025-00243-8>
- [16] Abhishek, K., Jain, A., Hamarneh, G., *Investigating the Quality of DermaMNIST and Fitzpatrick17k Dermatological Image Datasets*, Sci Data, Vol. 12 (2025), article 196. <https://doi.org/10.1038/s41597-025-04382-5>

Madina Mansurova,
Al-Farabi Kazakh National University (KazNU),
Almaty, Kazakhstan,

Anuar Nauryzbayev*,
Al-Farabi Kazakh National University (KazNU),
Almaty, Kazakhstan,
nauryzbayev_anuar@live.kaznu.kz

Ladislav Vegh,
Faculty of Economics and Informatics,
J. Selye University, Slovakia.

Received 12.02.2026, Accepted 03.06.2026, Available online 30.09.2026.