Skin Cancer Detection Using Neural Networks

Oksana Abramova
University of Padua, Department of Mathematics, oksana.abramova@studenti.unipd.it
Chueva Ekaterina
University of Padua, Department of Physics and Astronomy, ekaterina.chueva@studenti.unipd.it

Abstract

This project explores the application of computer vision techniques for skin cancer detection with the task of image classification. The methodology involved data preprocessing, model training and optimisation. We initially constructed a convolutional neural network model (CNN) as a baseline, followed by integrating transfer learning and simple attention mechanism to improve the results. Models of ResNet and DenseNet families were used for transfer learning as well as for exploring the effect of several modifications to the architecture.

The results highlight the effectiveness of the chosen architectures in automated skin cancer detection showcasing the potential of these techniques to assist in early diagnosis. The best model in our case was found to be pre-trained ResNet-50 with modifications, which achieved 91.7% *accuracy.*

1. Introduction

Skin cancer was found to be the fourth most frequent type of newly diagnosed cancer worldwide in a study from 2020 [1]. Among all skin cancers, melanoma is considered to be the deadliest type, for which the best treatment is early diagnosis [2]. Early diagnosis reduces the treatment costs and raises the patient survival rate.

In order to correctly detect and classify skin lesions, a non-invasive technique called dermoscropy was developed. Dermoscopic images collecting consists of covering skin lesion with mineral oil, alcohol or water, and taking a picture through a dermatoscope [2]. Later, the images are examined by a specifically trained dermatologist, paying attention to colour and surface structure of the skin. This examination requires a lot of experience and training, since besides malignant skin lesions (e.g. melanoma, basal cell carcinoma), there are many types of benign skin lesions (e.g. seborrheic keratosis), which sometimes can look alike.

Since skin cancer occurrences are growing worldwide, an attention of research community was drawn to developing automated ways of skin lesion classification and especially skin cancer detection. These automated approaches can help dermatologists in early diagnosis and improve clinical outcomes.

2. Related Work

The first automated approaches for skin cancer detection usually used the following pipeline: pre-processing, lesion segmentation, feature extraction (with optional feature selection), classification [3]. The "ABCD" rule (asymmetry, border, colour, diameter) had been a popular guide for classification of skin lesions [4] and the features were extracted according to it. After, hand-extracted features were fed to different classifiers, such as logistic regression, KNN and others [5]. However, these approaches did not show satisfactory performance while remaining a very complicated task.

Recent approaches are using deep neural networks, especially convolutional neural networks (CNNs)[5]. Moreover, pre-trained CNNs are found to be an effective solution detection that achieve the best results in skin cancer detection task.

There are also methods that combine hand-extracted features with a later use of deep neural networks.

3. Dataset

For this project, we chose a publicly available Melanoma Skin Cancer Dataset of 10,000 images [6]. This dataset is a compilation of dermoscopic images from different directories of The International Skin Imaging Collaboration (ISIC).

The dataset is divided into two subsets: train set (9600 images) and test set (1000 images). We dedicated 20 % of the train images for a validation set in order to monitor our models via learning curves.

Each image belongs to one of two classes: benign (non-cancerous) or malignant (cancerous). Examples of the images can be seen on fig. 1. We explored the two classes'

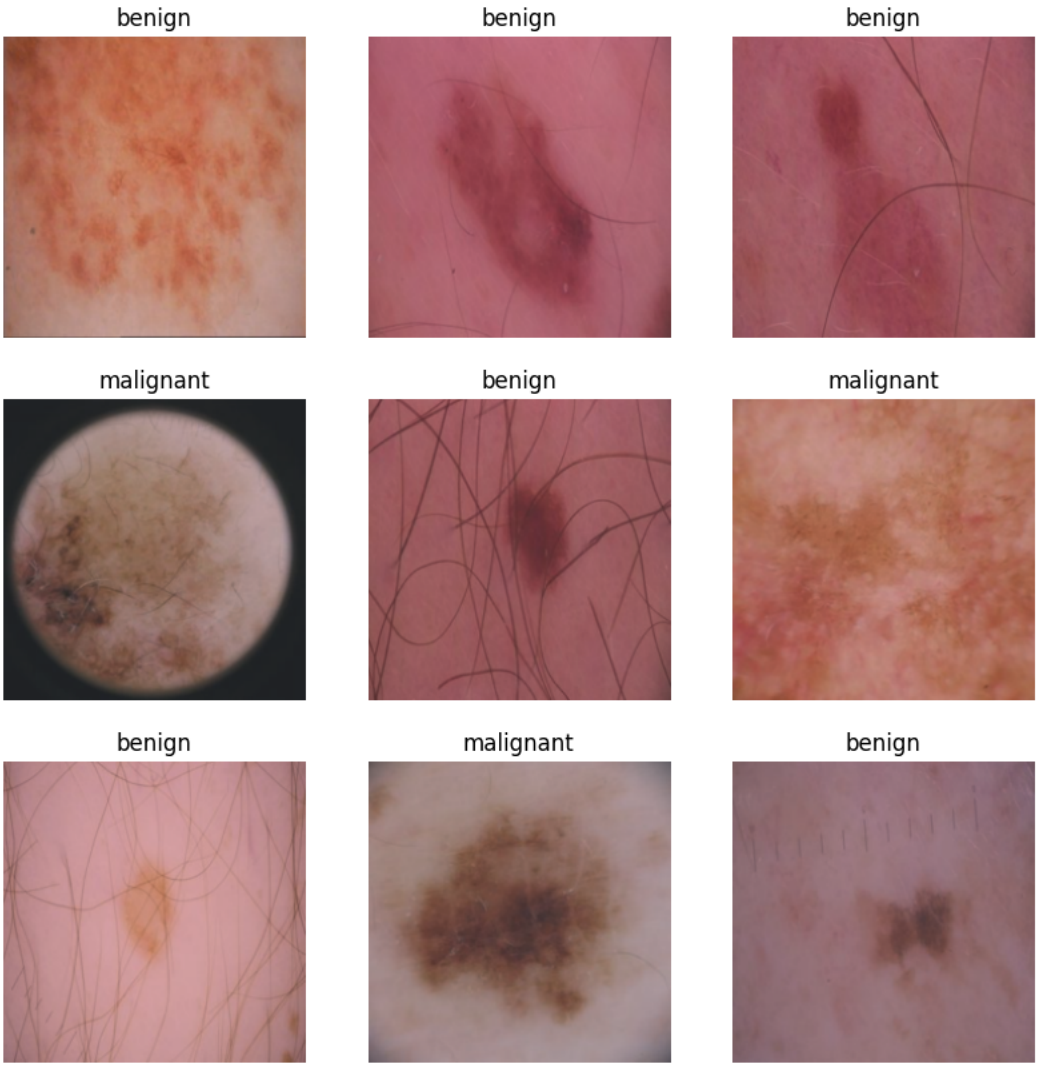

Figure 1. Examples of images from the chosen dataset.

distribution and found that the dataset is balanced (fig. 2).

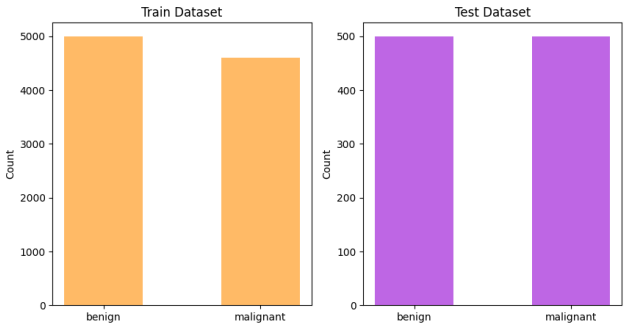

Figure 2. Class distribution in train and test sets.

3.1. Pre-processing

Before applying any of the algorithms discussed below, we have pre-processed the images in the following way, using the available functions in PyTorch [7]:

1) for the train, validation, and test sets:

- resizing images to 224×224 pixels;
- normalising values per channel:

$$\text{output[channel]} = \frac{(\text{input[channel]} - \text{mean[channel]})}{\text{std[channel]}}$$

the values for the mean and standard deviation were found specifically for the considered dataset;

2) only for the train and validation sets:

- random horizontal flip with a probability equal to 0.5;
- random crop of the image with keeping 0.6 of image width and image height;

We applied these additional transformations to the train and validation sets for additional regularisation, because the models can easily fall into overfitting.

4. Method

In order to classify dermoscopic images as either malignant or benign skin lesions, we adopted a deep feature extraction approach, thus we used neural networks to solve the task. The general pipeline is presented on a fig. 3.

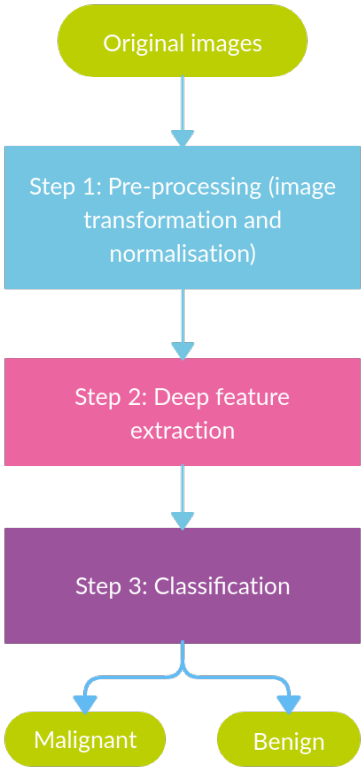

Figure 3. Pipeline.

Here, we discuss step 2 of the above pipeline in detail. We skip step 1, since it has been already discussed in the dataset section.

4.1. Deep feature extraction

In this step we used different neural network architectures to extract deep features, that will be later used for skin cancer classification. As reported in [5], CNNs became prevalent in this task, however, due to large number of parameters and limited datasets, CNNs tend to overfit. To solve this problem, a transfer learning approach was used.

Here we firstly explored a simple CNN architecture to use it as a baseline, then we also examined several pre-trained models. All pre-trained models had weights trained on ImageNet-1k dataset (1k classes dataset) [8]. All pre-trained models were adapted to the problem in question, meaning that the last layer was modified in order to have output equal to two (number of classes in the dataset). If we did not add any extra layers to the architecture (see experiments section), we have frozen the weights in all the layers except the last one.

For all the models we used cross-entropy loss, Adam optimiser with a learning rate 0.001 and a weight decay 10^{-4}. The batch size was set to 500 for all models, except the baseline and the model with attention (where it was set to 250 due to memory limits), in order to ensure better generalisation. Training was performed for 10 epochs. For each epoch we tracked train and validation loss, ensuring that models do not overfit. All the code was written using PyTorch package and the pre-trained models were imported from there. To run the models, we used Google Colab's T4 GPUs.

Below we briefly describe each architecture and evaluation metrics and will come back to further details in the experiments and results section.

4.1.1 Architectures

Baseline CNN. To start, we built a simple convolutional neural network. Our model consists of three convolutional layers ($W_{\text{out1}} = 32, W_{\text{out2}} = 64, W_{\text{out3}} = 128$), with a 3×3 kernel and padding equal to 1 for both dimensions. With each layer, the number of filters increases in order to capture high-level features from the input. Following each convolutional layer, a max pooling is applied with a 2×2 kernel and a stride of 2 to reduce the spatial dimensions. After flattening, we then proceed with a fully connected layer followed by a a ReLU activation function (here the output dimension equals to 512, we also use dropout with $p = 0.5$ for regularisation in this layer). After this, we again have a fully connected layer with an output equal to number of classes (2 in our case).

ResNet. The first type of pre-trained models that we used is Residual neural networks (ResNets) [9]. This type of networks use what is known as "skip connections", allowing them to learn residual mappings. The training becomes faster and the vanishing gradient problem is prevented. In this work we examined ResNet-18 (18 layers, which include convolutional layers, batch normalisation, and ReLU activation function) and ResNet-50 (50 layers, deeper network than ResNet-18).

DenseNet. The second type of pre-trained models we examined is Densely Connected Convolutional Networks (DenseNet) [10]. Compared to tradional CNNs, these type of CNNs have direct connections of each to every other

layer (thus for L layers there are $L(L+1)/2$ connections), which is also an approach for tackling vanishing gradient problem. In this work we used DenseNet-121, which has the following layers: one 7×7 convolutional layer, 58 3×3 convolutional layers, 61 1×1 convolutional layers, 4 average pooling layers, 1 fully connected layer.

Attention mechanism. In order to try and improve our results, a simple attention mechanism was introduced. This architecture consists of a CNN model with an attention module afterwards. CNN part is built of four convolutional layers, each of them followed by batch normalisation and a max pooling layer. After the final convolutional layer, the output is passed through an attention module, that consists of:

- Convolutional layer with kernel of 1×1, that reduces the depth of feature map from 256 to 128 channels to highlight the important features while preserving the spatial dimensions;
- Batch normalisation and ReLU activation to introduce non-linearity;
- Second convolutional layer restores the depth of the feature map to 256 channels;
- Sigmoid activation function provides attention weights in range between 0 and 1;

The attention weights are then applied to the feature maps through element-wise multiplication. This scales the feature maps according to their learnt importance - emphasising certain spatial regions. After applying attention, the flattened feature map goes through a fully connected layer that expands the features to 1024 dimensions and is followed by ReLU activation function (for regularisation purposes the dropout with $p = 0.7$ is used). Then, we have another fully connected layer with 512 output dimensions followed by ReLU activation. And final fully connected layer that produces the output corresponding to the number of classes.

4.1.2 Evaluation metrics

We used several metrics in order to evaluate our results. Here we use abbreviations: true positive (TP), true negative (TN), false negative (FN), false positive (FP), positive (P), negative (N).

Sensitivity. $\text{sensitivity} = \dfrac{\text{TP}}{\text{TP} + \text{FN}} =$ probability of "malignant" given that the patient has cancer

Specificity. $\text{specificity} = \dfrac{\text{TN}}{\text{TN} + \text{FP}} =$ probability of "benign" given that the patient is healthy

F1-score. $\text{f1-score} = \dfrac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$

Accuracy. $\text{accuracy} = \dfrac{\text{TP} + \text{TN}}{\text{P} + \text{N}}$

5. Experiments

In this section, we describe variants of several architectures in more detail. All the information for baseline CNN and CNN with attention was provided above.

ResNet. First experiments on the pre-trained model were carried out using ResNet-18 without any changes, so with only one last learnable layer. Later, in order to fine-tune the model for our specific task, few modifications were applied. In particular, a new fully connected layer ($W_{\text{in}} = 128$) with batch normalisation and ReLU activation function was added. Number of neurons was experimentally chosen. It was found that fewer neurons resulted in underfitting, while a much larger number (≈ 500) led to overfitting. Batch normalisation had a positive impact on the model. This second model is referred to as "ResNet-18 + layer" in the results.

Second experiment was carried out with ResNet-50 architecture. As before, the first approach is to have only one last learnable layer. The second variation is exactly the same, except the number of neurons in the new fully connected layer is equal to 64. Again, this parameter was experimentally found to prohibit overfitting. The second architecture, similar to the previous, is referred to as "ResNet-50 + layer" in the results.

DenseNet. Finally, the last set of experiments were carried out with DenseNet architecture. Similar to before, the first variant consists only of the last learnable layer. The second variant was adopted from [5], as on different from our benchmark datasets this architecture was reported to be the best. This model consists of a DenseNet followed by a multilayer perceptron (MLP), thus we refer to this model as "DenseNet + MLP". MLP has four layers with $50, 200, 200, 100$ neurons. After each layer there is batch normalisation followed by ReLU.

6. Results

The results can be seen in table 1, the best scores in each column are highlighted. We can observe that the three best scores belong to ResNet-18 architecture with an additional layer. However, we are hesitant to report this model as the best to use for our skin cancer detection task. As we can see, this model has the worst sensitivity among all. This evaluation metric is especially crucial for our task, since it is responsible for correct detection of cancerous skin lesions. If we look at the second model with the same accuracy, ResNet-50 with an additional layer, we can come to

conclusion that it can be reported as the best found one in this work, as it has a good trade-off between all the metrics.

In general, we found that for this specific dataset we do not require deeper networks to achieve a high accuracy, and smaller models achieve satisfactory performance. We can see that by comparing DenseNets to two types of ResNets. Thus, additional layers are more important than the number of layers with frozen pre-trained weights.

The DenseNet model with added MLP was not found to be the best for our task. Moreover, it was the only model to clearly overfit in the settings described above, it can be seen in fig. 4. However, it is probable that further parameter optimisation can improve its performance.

The CNN with attention was found to have satisfactory, but not outstanding results. It is possible that further experiments with architecture could improve its accuracy.

Overall, using pre-trained models makes the results better. However, they are not far from a simple baseline CNN. Thus, it is possible that these results are limited due to the need of further pre-processing of the dataset, e.g. hair removal or zooming into specific area of interest.

References

[1] H. Sung, J. Ferlay, R. L. Siegel, M. Laversanne, I. Soerjomataram, A. Jemal, and F. Bray, "Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *CA: a cancer journal for clinicians*, vol. 71, no. 3, pp. 209–249, 2021.

[2] G. Argenziano and H. P. Soyer, "Dermoscopy of pigmented skin lesions – a valuable tool for early," *The Lancet Oncology*, vol. 2, no. 7, pp. 443–449, 2001.

[3] A. Adekanmi and V. Serestina, "Deep learning techniques for skin lesion analysis and melanoma cancer detection: a survey of state-of-the-art," *Artificial Intelligence Review*, 2021.

[4] F. Nachbar, W. Stolz, T. Merkle, A. B. Cognetta, T. Vogt, M. Landthaler, P. Bilek, O. Braun-Falco, and G. Plewig, "The abcd rule of dermatoscopy: high prospective value in the diagnosis of doubtful melanocytic skin lesions," *Journal of the American Academy of Dermatology*, vol. 30, no. 4, pp. 551–559, 1994.

[5] H. K. Gajera, D. R. Nayak, and M. A. Zaveri, "A comprehensive analysis of dermoscopy images for melanoma detection via deep cnn features," *Biomedical Signal Processing and Control*, vol. 79, p. 104186, 2023.

[6] M. H. Javid, "Melanoma skin cancer dataset of 10000 images," 2022.

[7] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates, Inc., 2019.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[10] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

Model	Sensitivity	Specificity	F-1 Score	Accuracy
Baseline CNN	**0.93**	0.84	0.88	88.5
ResNet-18	0.91	0.89	0.9	89.8
ResNet-18 + layer	0.85	**0.99**	**0.92**	**91.7**
ResNet-50	0.91	0.87	0.89	89.1
ResNet-50 + layer	0.9	0.93	**0.92**	**91.7**
DenseNet-121	0.88	0.92	0.9	90.2
DenseNet-121 + MLP	0.89	0.94	0.91	91.3
CNN+attention	0.88	0.94	0.91	91.1

Table 1. Results

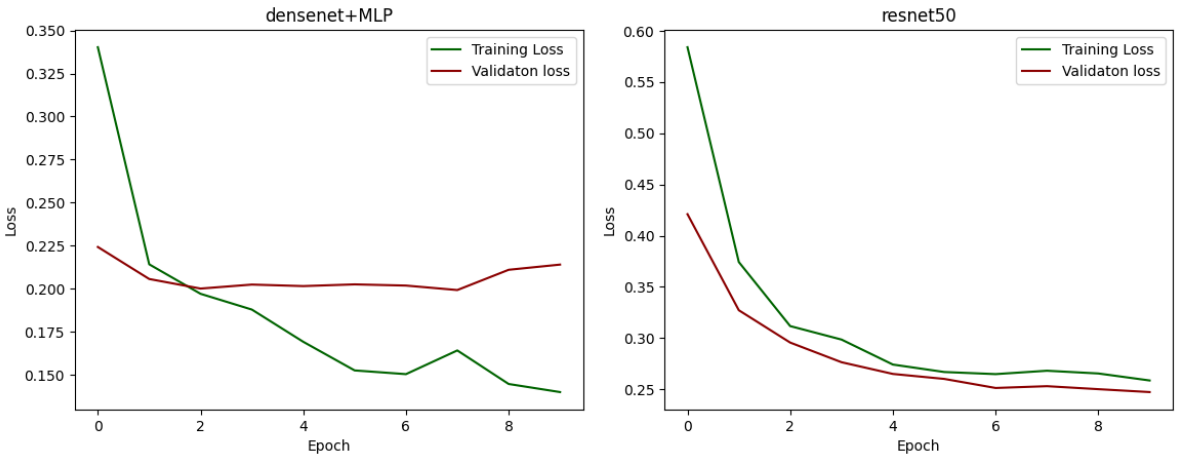

Figure 4. DenseNet + MLP vs. ResNet-50 Learning curve.