Classification of Skin Lesions

Paulo Roberto de Moura Júnior
IMA205 - Machine Learning for Image and Object Recognition, Télécom Paris

May 5th, 2024

1 Introduction

A skin lesion is defined as a superficial growth or patch of the skin that is visually different and/or has a different texture than its surrounding area. Skin lesions, such as moles or birthmarks, can degenerate and become cancer, with melanoma being the deadliest skin cancer. Its incidence has increased during the last decades, especially in the areas mostly populated by white people. The most effective treatment is an early detection followed by surgical excision. This is why several approaches for skin cancer detection have been proposed in the last years (non-invasive computer-aided diagnosis (CAD).
The goal of this work is to classify dermoscopic images of skin lesions from ISIC dataset [1] among eight different diagnostic classes:

1. Melanoma
2. Melanocytic nevus
3. Basal cell carcinoma
4. Actinic keratosis
5. Benign keratosis
6. Dermatofibroma
7. Vascular lesion
8. Squamous cell carcinoma

The dataset is composed of 25331 dermoscopic images of skin lesions with, when available, their relative segmentation and metadata (age, sex and anatomical position). Data have been randomly split into a training-validation set (75%) and a testing set (25%). The goal of the project is to estimate the correct class of each dermoscopic image in the test set.

2 Metodology and Results

To achieve the goal of the challenge, two methods were proposed. The first consists in computing a set features such as Asymmetry, Border irregularity, Color and the Dimension of the lesion (usually called the ABCD rule) for all images in dataset using their respective segmentation mask, which defines the lesion region. After the feature extraction, traditional machine learning models such as K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) were trained to classify the lesions. As the the features extraction from an image requires a segmentation mask, which are not available for all images in the dataset, two segmentation methods were implemented to solve this problem, a classical Otsu thresholding and a Convolutional Neural Network (CNN) based on U-Net architecture [2]. The second approach consists in indirectly extracting deep features through CNNs, trained only with the lesion images without the need of segmentation masks. Here two architectures were considered, LeNet-5 [3] and Inception-ResNet-V2 [4].

2.1 Segmentation

The first difficult encountered when dealing with the given dataset was the missing masks of some skin lesion images, which are required to perform ABCD features extraction. To deal with this, a corpus containing all images that had a ground truth segmentation mask was gathered, which was finally composed by the total of 2593 images (10,24% of the whole dataset). The chosen metrics to measure the performance of the segmentation methods are the mean and the standard deviation of Dice Coefficient [5], computed over the whole corpus.
At first, a segmentation method based on Otsu thresholding was considered, which is composed by the following steps:

1. Convert images from RGB to Grayscale.
2. Identify black marks on the borders of the images in the form of a binary mask.
3. Compute segmentation masks through Otsu thresholding only with the points outside of the black marks.
4. Compute Dice Coefficient.

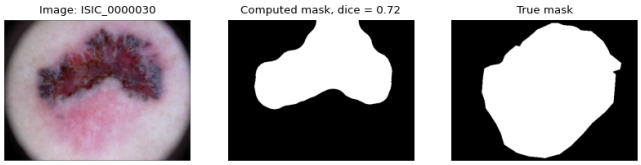

Figure 1: segmentation result for a lesion with multiple texture patterns by Otsu-based method

The results for Dice Coefficient with Otsu segmentation method are given in table 1. The implemented method has been shown to be simple and fast to compute, however its mean dice is too low and its standard deviation is too high, i.e., not suitable for features extraction. It has been noticed through visual inspection of some examples that this method can indeed perform well in some cases, but when the lesion has a complex pattern, such as multiple regions with different texture (figure 1), Otsu thresholding was not able to segment the whole lesion area, which can impact significantly the ABCD features extraction. This behaviour is characteristic of Otsu since it performs a global threshold on the image.

Table 1: results for Otsu segmentation

Dice Coefficient	
Mean	Standard Deviation
0.624	0.310

In order to possibly overcome the difficult of complex texture in the lesions and improve Dice coefficient metrics, a second method was implemented, based on the U-Net CNN architecture. This model was chosen because of its potential to be fast trained and its performance obtained in medical segmentation tasks, such as the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks [2].
The model was trained with 224x224 resized images and the results for Dice Coefficient with U-Net CNN segmentation are given in table 2. It's noticeable that the segmentation with U-Net outperforms Otsu since mean and standard deviation of Dice Coefficient have improved. Because of that, all missing segmentation masks were generate using U-Net based method. These results can be explained by the ability of CNNs to learn complex patterns in the image by minimizing a loss function, which is not possible by Otsu. As further improvement, one could consider using data augmentation in the training pipeline of the CNN [2], allowing the model to generalize better.

Table 2: results for U-Net segmentation

Dice Coefficient	
Mean	Standard Deviation
0.874	0.131

2.2 Feature extraction and metadata pre-processing

In order to perform image classification through traditional machine learning models, known features used as biomarkers for skin lesion classification were extracted from the images, known as ABCD features. In addition to that, as metadata were available for all images in the dataset, these data were included in the features list and pre-processed to deal with missing values. In total, 39 features were computed for each image.
The metadata features composed by age, sex and position of the lesion had an important amount of missing data. To deal with that, sex and position were converted to binary features in one-hot-encoding form, so that missing values would received zero (false) for all positions in the encoding. Missing values in age were replaced by the median age in order to not influence the classification.
The ABCD features chosen to be extracted were based on previous work on lesion segmentation, such as area and perimeter asymmetry [6] and Local Binary Patterns (LBP) descriptor for texture [7]. The complete list of features can be checked on the Python script implemented.

2.3 Classification with KNN and SVM

To perform classification of the skin lesions, the original train set was randomly splitted into a training (80%) and a validation set (20%). The main difficult encountered in the dataset is the high imbalance of classes, which could bias the classifiers to focus more in reducing misclassification for frequent classes.
As a first simplest approach, the KNN classifier was applied with 10-fold cross-validation to determine the best number of neighbors hyperparameter in order to minimize a weighted accuracy score described on equation 1, which penalizes more misclassifications in the less frequent classes.

$$\text{WA} = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i I(y_i = f_i) \qquad (1)$$

In equation 1, y_i are ground truths, f_i are the predicted results, and w_i are the weights of the $i-th$ test image. If we make the hypothesis that we have K groups - or classes - called $G = G_1, \ldots, G_K$ and we associate the same weight w_t to all images of the same group t, we

obtain that the weights are equal to $w_t = \frac{N}{k|G_t|}$. The weights in training-validation set were considered to be equal to testing set.
The KNN classifier results are shown in table 3 . KNN was found to bias the results towards the most frequent class, which is an expected characteristic of the classifier. Thus, its weighted accuracy was smaller on validation set, but at same time the classifier was found to be overfitting on training data and cross-validation wasn't able to solve this problem.

Table 3: classification accuracies for KNN

	Accuracy	Weighted Accuracy
Train	1.00	1.00
Validation	0.61	0.38
Test	—	0.31

In order to solve the problem of class imbalancement and better control overfitting, the SVM classifier with with Radial Basis Function (RBF) kernel was implemented, and its hyperparameters C and γ were tuned through 5-fold cross-validation, using weighted accuracy (equation 1) as scoring. This method was chosen because the SVM classifier implementation of Scikit-learn [8] allows the use of a "class weight" parameter which compensates class imbalancement within training set. In addition to that, SVM with RBF kernel has been verified to be a global approximator [9], thereby enhancing the classifier's capability to discriminate between classes within the dataset more effectively compared to the linear version.
The SVM classifier results are shown in table 4. As improvements, the class imbalancement bias has dicreased and the overfitting problem has also improved, since train and validation accuracies are closer and validation and test weighted accuracies are also closer.

Table 4: classification accuracies for SVM with RBF

	Accuracy	Weighted Accuracy
Train	0.63	0.69
Validation	0.57	0.51
Test	—	0.50

Finally, the oversampling of minority classes, implemented through Adaptive Synthetic Sampling (ADASYN) [10] algorithm was applied to the dataset, instead of using SVM "class weight" parameter, in order to compare the results with SVM inherented method.
The SVM classifier results with ADASYN are shown in table 5. As the dataset size increased, it became slower to perform cross-validation techniques, which made the task of reducing overfitting and optimizing the hyperparameters of the model harder. As a consequence, the model has been shown to be overfitting on training data and performed worse than SVM with "class weight" parameter. So, it's not clear if the oversampling would be a better approach than parsing the class weights to the classifier.

Table 5: classification accuracies for SVM with RBF and ADASYN

	Accuracy	Weighted Accuracy
Train	0.99	0.99
Validation	0.64	0.42
Test	—	0.45

2.4 Classification with CNNs

For the purpose of extracting complex features from the images that could capture better their structure than ABCD features and consequently improve results, classification methods based on Convolution Neural Networks were implemented, allowing the extraction of deep features. In these methods the segmentation masks generated for the lesion images were not used, as the CNN model can adapt itself to ignore the background and focus on the lesion area.
To perform the classification, the original train set was randomly splitted into a training (80%) and a validation set (20%), as in previous methods. The CNN models were implemented through PyTorch package in Python and to overcome class imbalancement issue, the class weights were passed to a parameter "weight" of the loss object in PyTorch implementation, which compensates the class imbalance in chosen loss function.
As a first approach, due to its fast training and ease to implement, LeNet-5 CNN model was chosen. For training the model, the input images were resized to 32x32 and then center-cropped to 28x28 format, which is the optimized input format for the model [3]. The chosen loss function was the Negative Log-Likelihood [11] with 'weight' parameter and the chosen optimizer was Adam [12] with learning rate equal to 0.0001. As the validation would be performed during training, a scheduler was implemented to refine the update of model parameters when the validation loss is close to the minimum (plateau). The model was then trained in 100 epochs with batch size of 64.
The LeNet implementation results are shown in table 6. The obtained results are very similar to SVM results which may be because LeNet-5 model is too simple to capture the complexity of the data. But even with 28x28 images the model was able to reach same results as SVM, which shows that the CNN approach has potential to reach better results using bigger images and more complex layers. Furthermore, the optimization

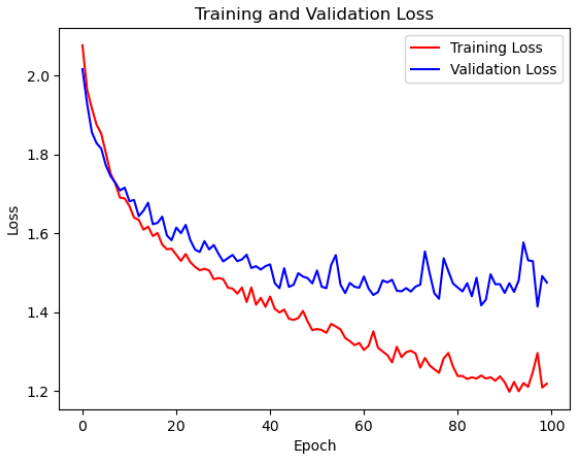

Figure 2: training and validation losses over epochs in LeNet-5

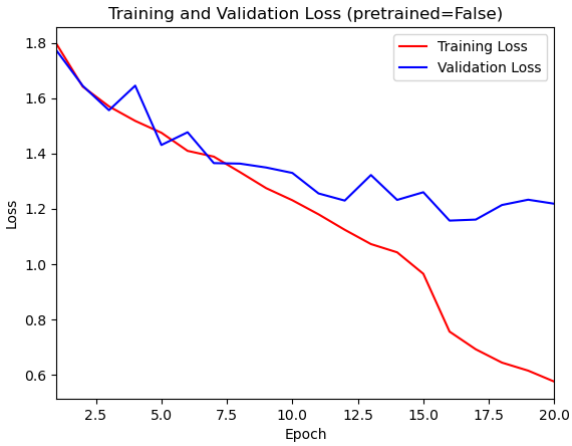

Figure 3: training and validation losses over epochs in Inception-ResNet-v2 model without pre-trained weights

of the model has reached a plateau, so training it more would make the model to overfit training data, as seen in figure 2.

Table 6: classification accuracies using LeNet-5 model

	Accuracy	Weighted Accuracy
Train	0.60	0.64
Validation	0.53	0.48
Test	—	0.46

A second and more complex architecture was proposed in order to input bigger images and extract more complex features from them, potentially increasing the classification accuracy. The chosen model was Inception-ResNet-v2, due to its superior performance in skin lesion classification, shown in previous work [4], and ease model manipulation using HuggingFace's Pytorch Image Models implementation [13]. For training the model, the input images were resized to 224x224, the chosen loss function was the Categorical Cross-Entropy [14] with 'weights' parameter, the chosen optimizer was Adam with learning rate equal to 0.0001 with on-plateau scheduler and the batch size was 32. The model was then trained in two configurations, at first initializing it with pre-trained weights (obtained in an ImageNet dataset training [13]) and 20 epochs, and at second loading the model with random weights and 25 epochs. The train pipeline was implemented in a way that the model parameters are saved when the minimum validation loss is found, before the validation loss starts increasing, as in overfitting scenarios.
The two Inception-ResNet-v2 implementations (with and without pre-trained weights) results are shown in tables 7 and 8. The plot in figures 3 and 4 show that after a certain number of iterations both models start to overfit. The highest train, validation and test accuracies were obtained using pre-trained weights, but looking at the plot of figure 4 it's clear that in this case the model overfits after a few epochs, which explains the differences in train, validation and test accuracies. The differences between weighted accuracy and accuracy metrics in the models show that the less frequent classes still have bad accuracies compared to the more frequent classes.

Table 7: classification accuracies using Inception-ResNet-v2 model without pre-trained weights

	Accuracy	Weighted Accuracy
Train	0.74	0.84
Validation	0.63	0.60
Test	—	0.59

Table 8: classification accuracies using Inception-ResNet-v2 model with pre-trained weights

	Accuracy	Weighted Accuracy
Train	1.00	1.00
Validation	0.81	0.70
Test	—	0.72

3 Conclusion

In conclusion, this work underscores the significant potential of machine learning methodologies in advancing the classification of dermoscopic images of skin lesions. With the Inception-ResNet-v2 model achieving a top weighted accuracy score of 0.72, the results demonstrate the effectiveness of deep learning architectures in accurately categorizing diverse types of skin lesions. The CNN models have demonstrated superiority for classifying skin lesions when compared to some

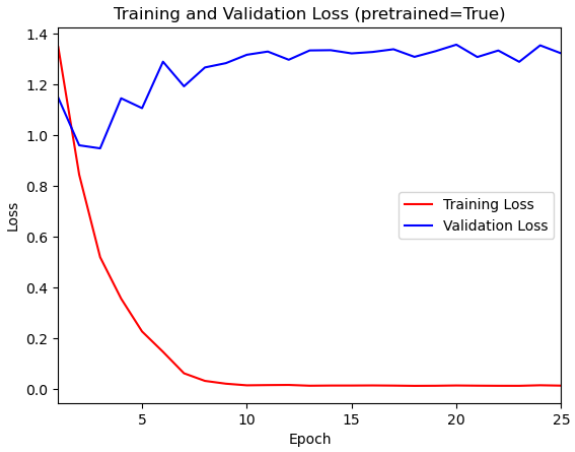

Figure 4: training and validation losses over epochs in Inception-ResNet-v2 model with pre-trained weights

classical machine learning methods, such as SVM and KNN. However, it is imperative to address the challenge of overfitting observed in almost all model implementations, specially in Inception-ResNet-v2, which can still be better optimized as shown here. To mitigate overfitting and enhance generalization, future works could explore a range of strategies. These include employing extensive data augmentation techniques to diversify the training data and integrating dropout regularization in CNN layers to prevent over-reliance on specific features. In addition to algorithmic enhancements, the integration of clinical expertise and domain knowledge into the model development process could refine classification results, ensuring clinical relevance and practical applicability.

References

[1] I. S. I. Collaboration, "ISIC Dataset," https://www.isic-archive.com/.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, pDF available. [Online]. Available: https://ieeexplore.ieee.org/document/726791

[4] S. Das and K. Sharma, "Inception-resnet-v2 based skin lesion classification for early detection and treatment," in *2023 International Conference on Computational Intelligence for Information, Security and Communication Applications (CIISCA)*. Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2023, pp. 143–146. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CIISCA59740.2023.00037

[5] W. contributors, "Sørensen–dice coefficient," https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient, 2024, accessed: May 2, 2024.

[6] H. Ganster, P. Pinz, R. Rohrer, E. Wildling, M. Binder, and H. Kittler, "Automated melanoma recognition," *IEEE Transactions on Medical Imaging*, vol. 20, no. 3, pp. 233–239, 2001.

[7] M. Zortea, T. R. Schopf, K. Thon, M. Geilhufe, K. Hindberg, H. Kirchesch, K. Møllersen, J. Schulz, S. O. Skrøvseth, and F. Godtliebsen, "Performance of a dermoscopy-based computer vision system for the diagnosis of pigmented skin lesions compared with visual evaluation by experienced dermatologists," *Artificial Intelligence in Medicine*, vol. 60, no. 1, pp. 13–26, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0933365713001589

[8] S. contributors, "Scikit-learn: Support vector machine (svc)," https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, 2024, accessed: May 2, 2024.

[9] J. Wang, Q. Chen, and Y. Chen, "Rbf kernel based support vector machine with universal approximation and its application," in *Advances in Neural Networks – ISNN 2004*, F.-L. Yin, J. Wang, and C. Guo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 512–517.

[10] I. contributors, "Imbalanced-learn: Adasyn," https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.ADASYN.html, 2024, accessed: May 2, 2024.

[11] PyTorch Contributors, "PyTorch Documentation: torch.nn.nllloss," https://pytorch.org/docs/stable/generated/torch.nn.NLLLoss.html, 2024, accessed: May 2, 2024.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014, accessed: May 2, 2024. [Online]. Available: https://arxiv.org/abs/1412.6980

[13] Hugging Face, "Timm Documentation: Inception-resnet-v2," https://huggingface.co/docs/timm/models/inception_resnet_v2.html, 2024, accessed: May 2, 2024.

[14] P. Contributors, “PyTorch Documentation: torch.nn.CrossEntropyLoss,” https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html, 2024, accessed: May 2, 2024.