Multi-class Skin Lesion Classification using a Deep Ensemble of ImageNet-pretrained Dual Path Network (DPN) models

Shlomo Kashani*
Head of Deep Learning
DeepOncology AI
shlomo@deeponcology.ai

Abstract

*This short paper describes our contribution to the 2018 ISIC Skin Image Analysis Challenge [***ISICSkin10:online***]. We participated in Task 3: Disease and Lesion Classification. The officially stated goal of this multi-class image classification challenge is to classify dermoscopic images into seven different disease categories.*

*In this work, we developed an Ensemble learning pipeline in PyTorch using state of the art ImageNet-pretrained [***Cadenepr71:online***] convolutional neural networks (CNNs) as base classifiers. Much like classical bagging, base learners were trained on diverse stratified data-set folds generated by bootstrapping the provided training set. Furthermore, additional variation was introduced by using different CNN architectures. In particular, we opted for an exhaustive optimization based search tactic, generating few hundred predictions by permuting (1) N different random seeds, (2) K different data folds and, (3) M different CNN models, mostly consisting of Dual Path Network (DPN92, DPN107, DPN131) architectures [***DBLP:journals/corr/ChenLXJYF17***].*

*Subsequently, out of these N*K*M predictions, we included only the top ten performing models. Lastly, using a CNN Ensemble, we amalgamated these ten models into a single prediction. This approach which selects the skin lesion type predicted by the most confident base learner, yielded a local cross-validation accuracy score of 96% using our own K-fold cross validation set, while yielding a score of 91.4% on the official non-labeled validation set provided by the organizers. We unfortunately joined the competition late, and we look forward to improving on these results.*

I. Introduction

Melanoma is a lethal form of malignant skin cancer, frequently misdiagnosed as a benign skin lesion or even left completely undiagnosed. In the United States alone, melanoma accounts for an estimated 6,750 deaths per annum [**jemal:2016**]. With a 5-year survival rate of 98%, early diagnosis and treatment is now more likely and possibly the most suitable means for melanoma related death reduction.

Dermoscopy images [1] are widely used in the detection and diagnosis of skin lesions. Dermatologists, relying on personal experience, are involved in a laborious task of manually searching dermoccopy images for lesions [**grob:2005**]. Therefore, there is a very real need for automated analysis tools, providing assistance to clinicians screening for skin metastases. In this work we address some of these fundamental issues by realizing a deep learning pipeline to automate the analysis of skin lesions.

* **These authors contributed equally to this work.**

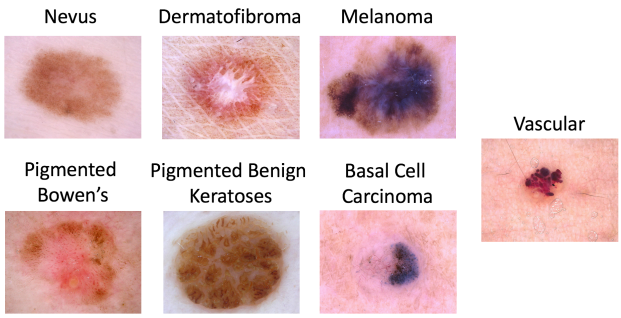


Figure 1: **Skin lesion categories.** An exemplary visualization of the data provided by the organizers.

II. Methods

As suggested in [**1805089740:online**] *Do Better ImageNet Models Transfer Better?* we optimized and ensembled numerous ImageNet-pretrained CNNs to resolve our classification task. All the experiments were conducted on a Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz, 256GB Mem, trained on 8 GeForce GTX 1080 (Pascal) GPU's with CUDA v. 8.0 (driver 375.20) and cuDNN (v. 5005).

At the beginning, we mostly used Dual Path Networks (DPN) which were first proposed in [**DBLP:journals/corr/ChenLXJYF17**] as a hybrid network design that incorporates the basic idea of DenseNet [**DensNet**] with that of ResNeXt [**Xie2016**]. However, later on we expanded our CNN inventory to include networks ranging in ImageNet top-1 accuracy from 77.152% to 81.304%. We used the publicly available network weights and models from the PyTorch pre-trained model repository [**Cadenepr71:online**].

Table 1: **Original ImageNet-pretrained** CNNs used in transfer learning

CNN Model	Classes	Image Size	Top-1 accuracy
ResNet101	1000	224	78.956
ResNet152	1000	224	78.428
DPN98	1000	224	79.224
DPN107	1000	224	79.746
DPN131	1000	224	79.432
SeNet154	1000	224	81.304
SeResneXT101	1000	224	80.236
DenseNet201	1000	224	77.152
DenseNet161	1000	224	77.560
InceptionV4	1000	299	80.062

The respective ImageNet top-1 accuracy, penultimate layer feature dimension, parameter count and input image dimensions for each CNN architecture are available at (https://github.com/Cadene/pretrained-models.pytorch) and depicted in Table 1).

By exploiting CNN's and transfer learning [**chu:2016**], we implement a two-phase pipeline that can accurately classify skin lesions in dermoscopy images.

III. Data Collection

The dataset provided by the competition organizers, is part of the [**Tschandl2018_HAM10000**] dermoscopic image dataset. For Task 3, lesion classification, the training dataset consists of approximately 10015 images falling into seven distinct skin lesion categories. The respective frequency of each category is reported in table 2.

Table 2: Skin lesion frequencies

Category	Frequency
1	**6705**
0	1113
4	1099
2	514
3	327
6	142
5	**115**

The provided training set is imbalanced, with the majority class **(NV)** being more than **sixty** times the size of the minority skin lesion type **(VASC)**. Accordingly, in an attempt not to learn a possibly unfair hypothesis, we tried more than a few approaches for generating balanced training samples from the provided training set.

We investigated down sampling, bootstrapping and the penalization of misclassified classes depending on the size of classes, in so doing forcing the CNN to emphasize poorly represented skin lesion types. We finally opted for a class balancing approach coupled with repeated K-fold sampling which achieved the highest validation accuracy. For classification purposes, we encoded the labels as follows:

```
MEL': 0, 'NV': 1, 'BCC': 2, 'AKIEC': 3, 'BKL': 4, 'DF': 5, 'VASC': 6
```

I. Data Augmentation

Due to the relatively small size of our training set, and the complexity of the visual tasks we are trying to solve, our approach relies heavily on data augmentation. During run-time, we utilized an extensive data augmentation protocol by implementing random rotations, horizontal/vertical flips, brightness, contrast, color-balance and sharpness transformations. The augmentations are exemplified as follows:

```
self.transforms = []
if rotate:
 self.transforms.append(RandomRotate())
if flip:
 self.transforms.append(RandomFlip())
if brightness != 0:
 self.transforms.append(PILBrightness(brightness))
if contrast != 0:
 self.transforms.append(PILContrast(contrast))
if colorbalance != 0:
 self.transforms.append(PILColorBalance(colorbalance))
if sharpness != 0:
 self.transforms.append(PILSharpness(sharpness))
```

IV. Deep Learning Pipeline

We fine tuned *all* layers of our CNN by training for between 10 to 120 epochs with ADAM (batches of 32-64 images), and using a categorical cross entropy loss.

I. From 1000 classes to 7 classes

In order to fine tune our CNN's, the (original) output layer with 1000 classes was removed and the CNN's was adjusted so that the (new) classification layer comprised seven softmax neurons emitting posterior probabilities of class membership for each lesion type. This idiom is exemplified on the **Dpn107 CNN** as follows:

```
import torch
class Dpn107Finetune(nn.Module):

def __init__(self, num_classes: int, net_kwards):
  super().__init__()
  self.net = pretrainedmodels.dpn107(**net_kwards)
  self.net.__name__= str (self.net)
  self.net.classifier = torch.nn.Conv2d(2688, num_classes,kernel_size=1)
  print(self.net)
```

II. CNN model generation

Model selection was implemented by saving the network achieving the highest validation accuracy (during and) at the end of training. During training, whenever the validation accuracy surpassed a threshold of 85%, a checkpoint for the currently running CNN was saved noting the respective accuracy, loss and epoch. Lastly, generalization performance was evaluated by testing the saved CNN on an independent test set that we kept aside.

V. Results

The results of our best **single** performance assessment, for each architecture are depicted in Table 3 (classification report).

I. Single Model Scores

An exemplary confusion matrix for the ResNet152 CNN is presented in Figure 2.

The performance assessment, on test data, including precision, recall, and their harmonic mean (F measure) across classes, weighted by their support is presented in Table 4.

The competition organizers allowed infinite validation set prediction submissions with an immediate feedback on the respective resulting accuracy. For instance, for our CNN checkpoint model entitled, *0.853_dpn107_finetune_87.8648_0.3777.csv*, the submission resulted in a score of 0.853, which is one of our highest ranking single CNN models. The leader-board score for the DPN107 CNN is presented in Figure 3.

Model	Validation Acc / Loss	Validation Acc (SUBMISSION)
ResNet101	92.3% / 0.28	84.9%
ResNet152	90.2% / 0.347	**87.7**%
DPN98	85.4% / 0.54	82.4%
DPN107	87.8% / 0.377	85.3%
DPN131	86% / 0.45	81.1%
SeNet154	88.7% / 0.35	83.38%
SeResneXT101	87.5% / 0.59	84.8%
DenseNet201	low%	low%
DenseNet161	low%	low%
InceptionV4	low%	low%
InceptionV3	low%	low%
InceptionResNetV2	low%	low%
Xception	low%	low%

Table 3: **Best single model performance metrics across CNNs.** Validation and test accuracies are reported for each of the base CNN learners. **Note: the scores represent accuracy on the validation set provided by the competition organizers; no labels were provided for this set and hence it was impossible to create a respective confusion matrix.**

Precision	Recall	*F* Measure
0.9537433884033649	0.976997421635158	0.9643136788394481

Table 4: **ResNet152 model Performance on the ISIC2018 data set with support: [106 626 44 34 82 9 10].**

II. Ensemble Scores

We developed an ensemble system using the CNNs in Table 3 as base classifiers. All ensemble members were trained as described in section I. Idiomatically, the source code for generating the ensemble is as follows:

```
l = []
for i,f in enumerate(filelist):
  temp =  pd.read_csv(f)
  l.append(temp)

arr = np.stack(l,axis=-1)
avg_results = pd.DataFrame(arr[:,:-1,:].mean(axis=2))
avg_results['image'] = l[0]['image']
avg_results.columns = l[0].columns
```

We did not have time to investigate more sophisticated ensembling methods such as meta-learners. The leader-board score for the Ensemble CNN is presented in Figure 4.

Compared to the base learner's results our ensemble system was capable of boosting accuracies and we were able to improve performance of each base classifier, achieving an overall accuracy of 91.5% on the validation leader-board.

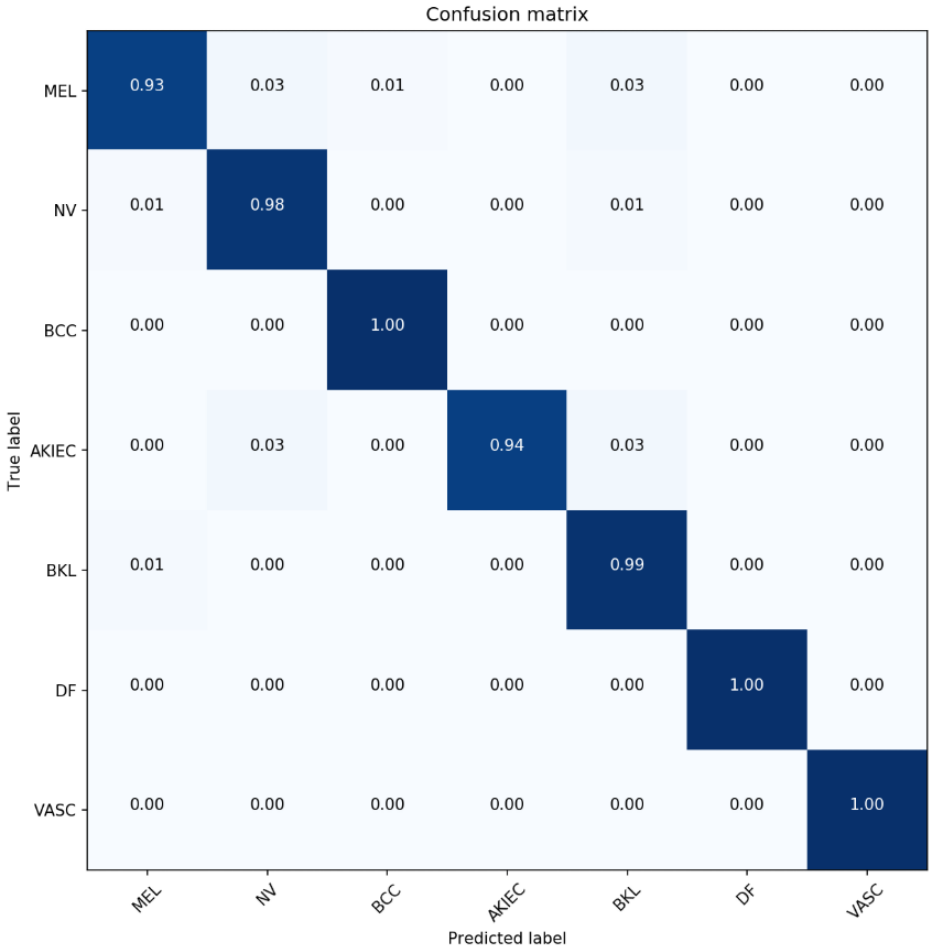
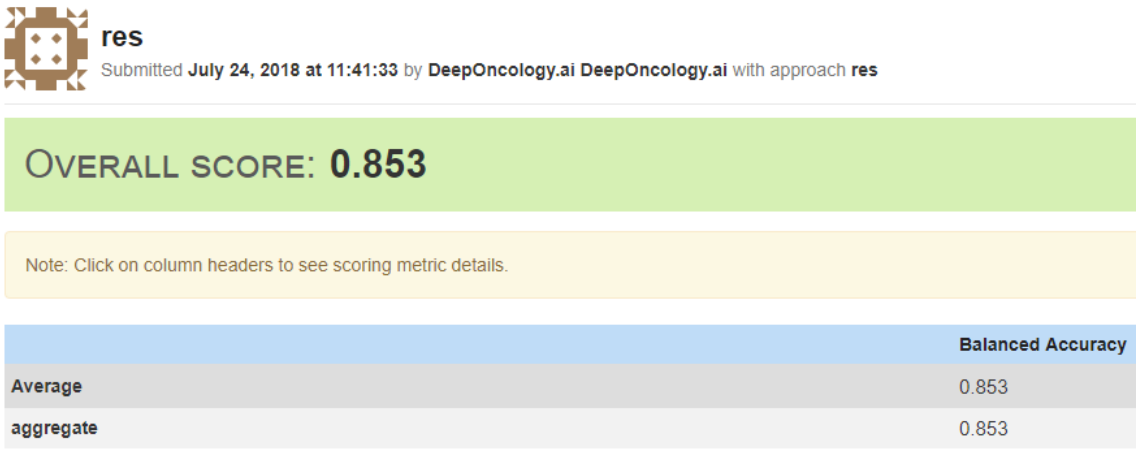


Figure 2: **Resnet152's confusion matrix.** Visualisation of error rates, on our independent test set, across skin lesion types.



res

Submitted **July 24, 2018 at 11:41:33** by **DeepOncology.ai DeepOncology.ai** with approach **res**

OVERALL SCORE: **0.853**

Note: Click on column headers to see scoring metric details.

	Balanced Accuracy
Average	0.853
aggregate	0.853

Figure 3: **Actual Leader-board score**

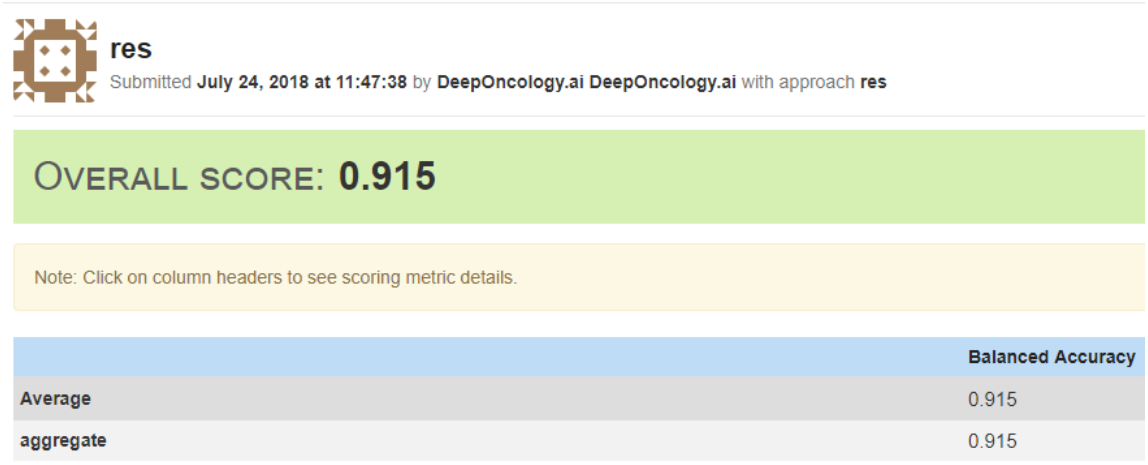


Figure 4: **Actual Leader-board score**