Background
Competition Name: SIIM-ISIC Melanoma Classification
Team Name: aloe
Private Leaderboard Score: 0.9485
Private Leaderboard Place: 2nd

Name: Ian Pan
Location: Cleveland, OH, USA
Email: ianpan358@gmail.com

I am currently a preliminary medicine intern at University Hospitals Cleveland Medical Center. I received my MD earlier this year in May 2020 from Brown University. I will be starting radiology residency training next summer at Brigham & Women's Hospital in Boston, MA. My research interests lie at the intersection of machine learning, specifically deep learning and computer vision, and medicine, specifically radiology and medical imaging. I train neural networks on various medical imaging tasks (e.g., disease detection, organ segmentation) across many different imaging modalities (e.g., radiography, computed tomography, magnetic resonance imaging).

I am also an avid Kaggler. I enjoy participating in Kaggle competitions to further my data science skills. Prior to this challenge, I earned gold medals in 4 other competitions (1st, 5th, 8th, and 12th), each time as part of a duo. One of those competitions was last year's SIIM challenge focused on pneumothorax segmentation. I entered this competition because it was another opportunity for me to participate in a competition at the intersection of machine learning and medicine.

I spent roughly 10-20 hours per week on this competition, primarily over the last 3-4 weeks. As I just started my job as a junior doctor, my time was limited. Once I had my experimental pipeline up and running, it was mostly a matter of running experiments before work, analyzing the results after work, and then spending some time deciding on what experiments to run next.

Summary
My solution was centered on the EfficientNet family of convolutional neural networks, specifically EfficientNet-B6 and B7. I used the RandAugment strategy for data augmentation. I also used cosine annealing with warm restarts as the learning rate scheduler. I leveraged the ISIC 2019 training data, which provided extra melanoma examples to learn from and prevented overfitting to the 2020 data. All models were trained in PyTorch 1.6, using native automatic mixed precision, on 4 NVIDIA Quadro RTX6000 GPUs. I also pseudolabeled the test set to further increase the number of training examples. My final solution was an ensemble of 15 models (5-fold EfficientNet-B6; 5-fold EfficientNet-B7; 5-fold EfficientNet-B6, pseudolabeled). Each model took 2-4 hours to train. My source code is available on GitHub: https://github.com/i-pan/kaggle-melanoma.

Training Methods

All experiments were conducted through 5-fold cross-validation for the most robust results. I used the triple-stratified K-fold splits provided by Chris Deotte* to prevent data leakage.
*https://www.kaggle.com/c/siim-isic-melanoma-classification/discussion/165526

First, I trained a model on the ISIC 2019 training data due to its more granular labels. Specifically, I trained a 3-class model to classify images into: 1) melanoma, 2) nevus, 3) other. The intuition behind this is that melanoma versus benign nevus is probably the most difficult discrimination task, as they can appear similar in many cases.

Second, because the ISIC 2020 data had a large chunk of data labeled as “unknown” for diagnosis, I used the above model to predict a nevus probability for every unknown image. To determine the threshold at which I would label the image as “nevus” as opposed to “other”, I also ran the model on all of the images in the 2020 data that had a known diagnosis of nevus. I used the 5th percentile of the predictions on known nevi as the threshold, above which the unknown images would be labeled as nevi.

Then, I combined the ISIC 2019 and ISIC 2020 training data together. Each example in the data now belonged to one of 3 classes: 1) melanoma, 2) nevus, 3) other. Because the ISIC 2019 data had a higher prevalence of melanomas, I upsampled the melanomas (by a factory of 7) in the ISIC 2020 data so that approximately 50% of the melanomas would be from 2019 and the other half from 2020.

Models were trained only using the images, without metadata. I did train models with both image data and metadata, but those models performed worse on the private leaderboard.

I trained models using the AdamW optimizer and cosine annealing with warm restarts. I trained for 6-9 epochs with 3 snapshots and used the snapshot with the highest validation AUC. RandAugment was used for data augmentation. Augmentations included: contrast, brightness, and color manipulations; zoom in/out; rotation; shearing; translations; and cutout. EfficientNet-B6 was trained on 512x512 images with a batch size of 64. EfficientNet-B7 was trained on 640x640 images with a batch size of 32 using gradient accumulation of 2. Generalized mean pooling and multisample dropout were applied at the end of the network.

I then obtained predicted probabilities for each class on the ISIC 2020 test data. I added the pseudolabeled ISIC 2020 test data back into the training set and repeated the training process. I trained using the predicted probabilities, without assigning the test data to a specific class.

Interesting Findings

Having a robust local cross-validation was important given that the public leaderboard only had 77-78 melanoma examples. It was clear that competitors were severely overfitting the public leaderboard. Though I focused heavily on my CV score, I also used the public LB to pick among models with similar CV scores.

RandAugment was also an important element of my solution. Given the small number of ISIC 2020 melanoma examples, having a strong augmentation strategy was crucial.

Finally, pseudolabeling was also an important part of my final solution. It allowed me to further increase the number of melanoma examples.

Simple Features and Methods

My best single model was a 5-fold EfficientNet-B6, trained on ISIC 2019, ISIC 2020, and pseudolabeled data, achieving a private leaderboard score of 0.9495. It turns out I did not select this single model, and it was actually better than my ensemble of 3 5-fold models as described above.

Model Execution Time

EfficientNet-B6 trained on 512x512 images takes about 2 hours to train.
EfficientNet-B7 trained on 640x640 images takes about 4 hours to train.
The simplified model described above would take 2 hours to train each fold, 10 hours total.
Using 10-crop test time augmentation, it takes less than 1 second to perform inference on 1 image.

References

EfficientNet
https://arxiv.org/abs/1905.11946
https://github.com/rwightman/pytorch-image-models

RandAugment
https://arxiv.org/abs/1909.13719
https://github.com/ildoonet/pytorch-randaugment