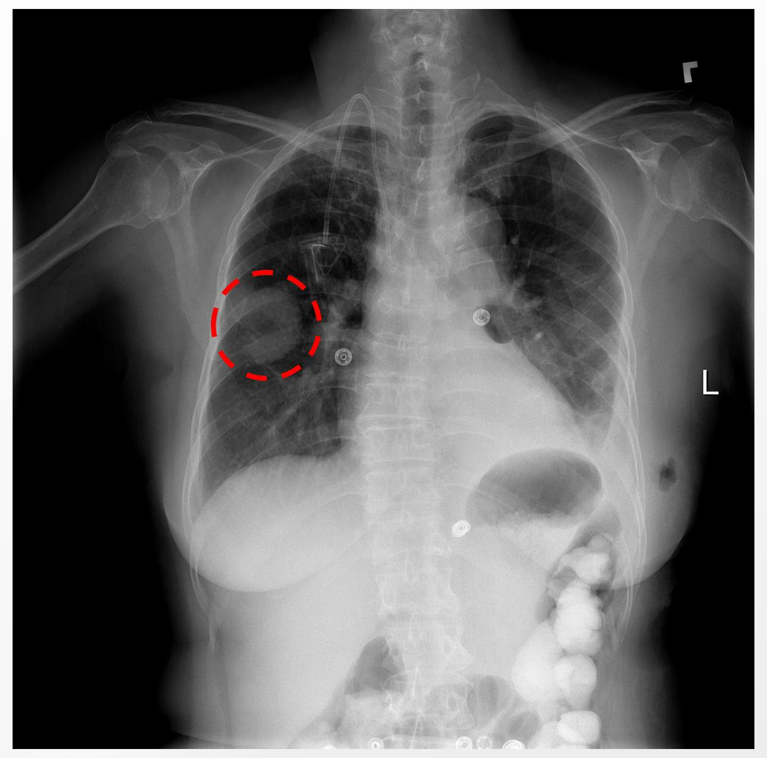
Supercomputer-scale training of large AI Radiology models

Valeriu Codreanu[1], Damian Podăreanu[1], Lucas A. Wilson[2], Srinivas Varadharajan[2], Vikram Saletore[3]

[1]SURFsara, [2]Dell EMC, [3]Intel

Introduction



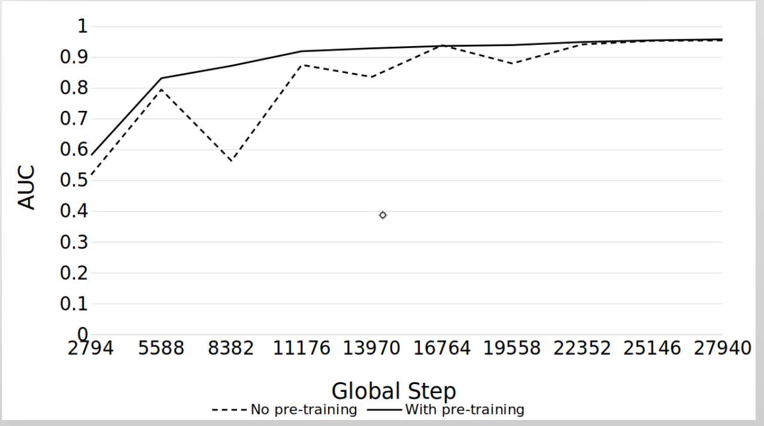
In recent years, the healthcare industry has been moving towards fully digitized workflows. This facilitates the adoption of artificial intelligence algorithms, particularly in cases where medical doctors do not reach a consensus. The final goal of these algorithms is to support decision making and help with standardization. Therefore obtaining high quality models that can be quickly trained, becomes critical for this industry. In this poster we present our findings for training three different high accuracy artificial neural network models for identifying pneumonia, emphysema, and a host of other lung afflictions. This was done on a dataset released by NIH Clinical Center, containing over 100,000 chest x-ray images from more than 30,000 patients and 14 different pathologies.

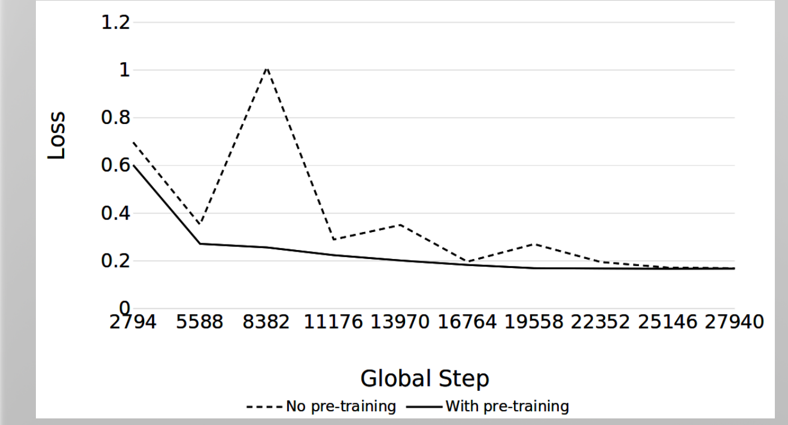
Emphysema is estimated to affect more than 3 million people in the U.S., and more than 65 million people worldwide. Severe emphysema is life threatening, and early detection is important to try to halt progression. Pneumonia affects more than 1 million people each year in the U.S., and more than 450 million each year worldwide. Every year, 1.4 million people die from pneumonia worldwide.

The original ChestXray-14 images are of size 1024x1024, but general practice is to downsample the input images to 224x224 to fit the input layer of traditional ImageNet-pretrained models. Our approach is to utilize as much of the available information as possible, so we experiment with larger than common practice inputs. We compare this approach with models featuring a large number of trainable parameters as well as state-of-the-art architectures used for generic image recognition tasks.

Methodology

Stanford University researchers originally developed the CheXNet model by fine-tuning a 121-layer DenseNet topology originally trained on the ImageNet dataset. The data for fine tuning the model came from the U.S. National Institute of Health (NIH) ChestXray-14 dataset.





Use of pretrained models should always be a first consideration when developing new deep learning models, as the need to isolate common features is eliminated.

In the two plots from the left-hand side we present the accuracy and loss during training of the mammography model with and without pretraining. This (solid line) provides a much smoother loss profile and faster time to high accuracy vs. the model that was not pretrained. Use of existing models should always be a first consideration when developing new deep learning models, as the need to isolate common features is eliminated.

In order to obtain the desired accuracy, when picking an existing checkpoint, we do not pick the last one. We start with the learning rate at which the model was training when the checkpoint was saved. We also perform gradual warmup of the learning rate, proportional to the global batch size.

Our code is developed in Tensorflow and we pack the data in the TF Records format so that it can be efficiently consumed asynchronously with the computation. Horovod was then added to parallelize the training.

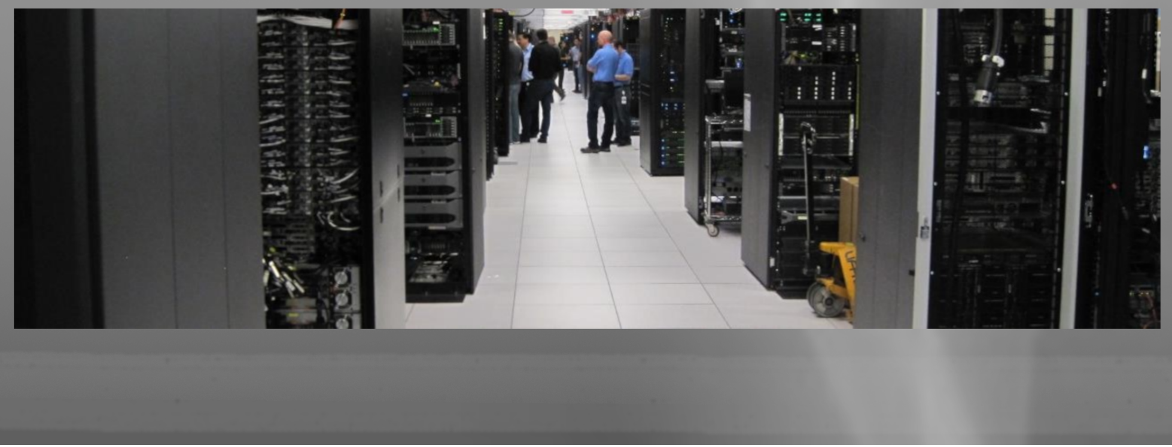
Horovod uses the Ring-AllReduce approach to distributed deep learning, which take a single-program, multiple-data (SPMD) model approach to the parallelization process, using MPI for communications.

Each MPI process has a unique copy of the network being trained. Each process looks at a slice of the training data, and exchanges gradient information using the MPI_AllGather operation. Loss information is aggregated using MPI_AllReduce.

We have also compared the efficiency of our Tensorflow implementation with a Keras-Tensorflow one. For our architecture and dataset, we've noticed the Tensorflow-only code is approximately 4 times faster when doing a 128 node run, distributed with Horovod.

Global batch size	Framework	Number of nodes	Time per Epoch
4096	Keras	128	85 s
4096	Tensorflow	128	18 s

All our experiments have been performed on Dell EMC's Zenith cluster. An image from the Dell EMC HPC and AI innovation lab can be found below.



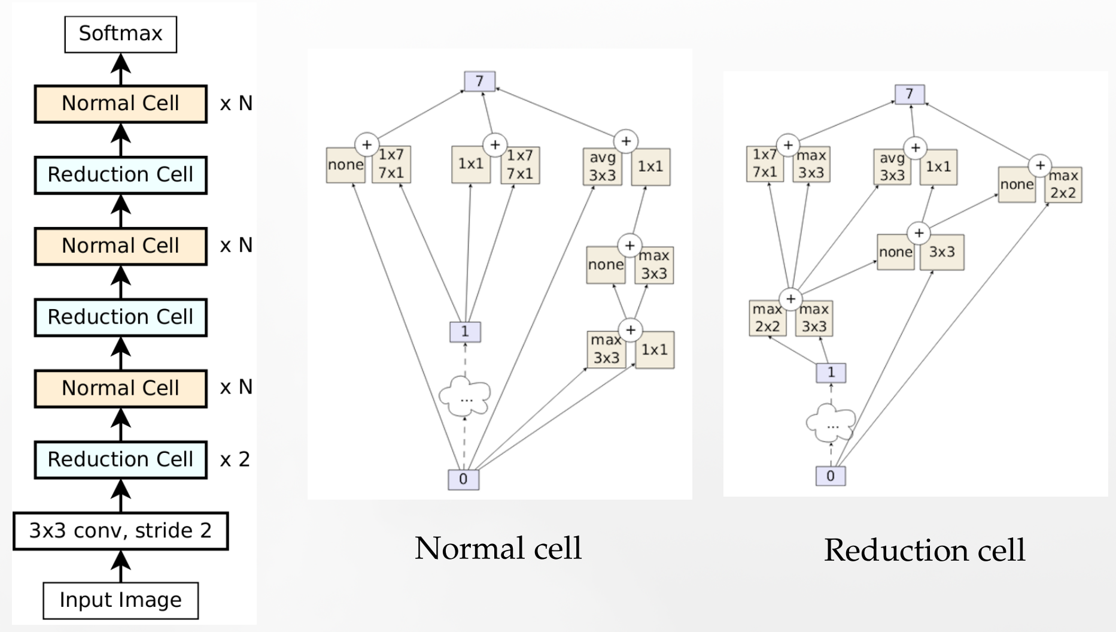
Exploring network architectures on the ChestXray-14 dataset

Layer	Output shape
Input	3x224x224
Conv 1, 7x7, stride 2	64x112x112
Max Pool, 3x3	64x56x56
ResBlock 2a	256x56x56
ResBlock 2b	256x56x56
ResBlock 2c	256x56x56
ResBlock 3a	512x28x28
ResBlock 3b	512x28x28
ResBlock 3c	512x28x28
ResBlock 3d	512x28x28
ResBlock 4a	1024x14x14
ResBlock 4b	1024x14x14
ResBlock 4c	1024x14x14
ResBlock 4d	1024x14x14
ResBlock 4e	1024x14x14
ResBlock 4f	1024x14x14
ResBlock 5a	2048x7x7
ResBlock 5b	2048x7x7
ResBlock 5c	2048x7x7
Average pool	2048x1x1
Flatten	2048
Output	14

ResNet-50 architecture

Layer	Output shape
Input	3x896x896
Conv 1, 7x7, stride 2	64x448x448
Max Pool, 3x3	64x224x224
ResBlock 2a	256x112x112
ResBlock 2b	256x112x112
ResBlock 2c	256x112x112
ResBlock 3a	512x56x56
ResBlock 3b	512x56x56
ResBlock 3c	512x56x56
ResBlock 4a	1024x28x28
ResBlock 4b	1024x28x28
ResBlock 4c	1024x28x28
ResBlock 4d	1024x28x28
ResBlock 5a	2048x14x14
ResBlock 5b	2048x14x14
ResBlock 5c	2048x14x14
ResBlock 5d	2048x14x14
ResBlock 5e	2048x14x14
ResBlock 5f	2048x14x14
ResBlock 6a	4096x7x7
ResBlock 6b	4096x7x7
ResBlock 6c	4096x7x7
Average Pool	4096x1
Flatten	4096
Output	14

ResNet-59 architecture
(large input ResNet-50)



Outline of the overall model

Normal cell

Reduction cell

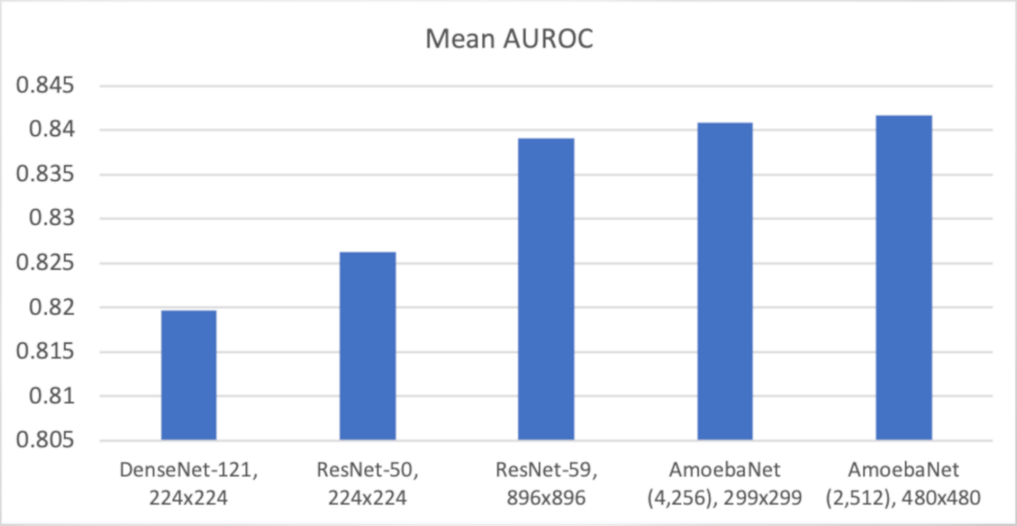
AmoebaNet-D architecture

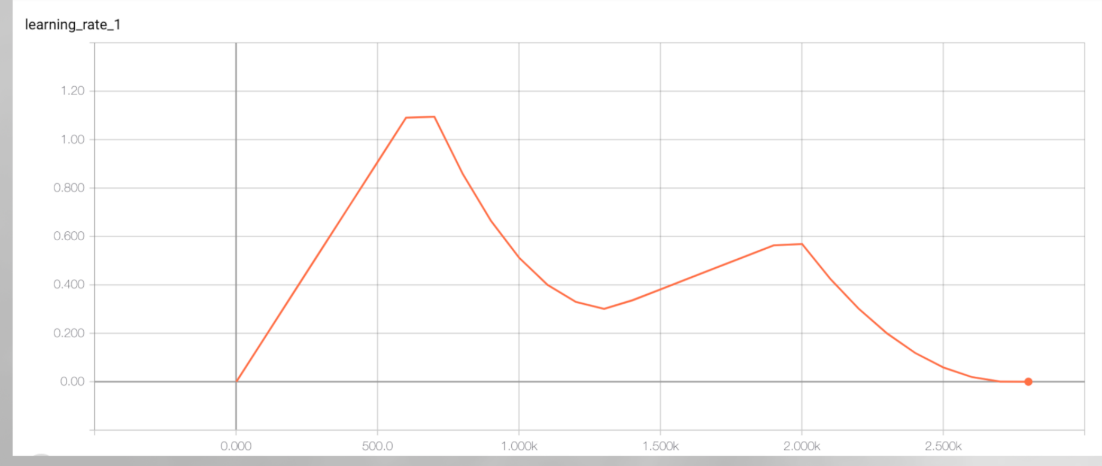
In our current work we have focused on three different Convolutional Neural Network designs. The first one is a standard ResNet-50. In our previous research this proved to be superior to other Neural Network designs such as DenseNet for classification tasks.

ResNet-59 is a natural evolution of ResNet-50 when increasing the input size to 1024x1024 pixels. This architecture features an additional of a stride-2 convolution in the second residual block an increased number of layers in order to deal with the extra from the input layer. This leads to approximately four times more trainable parameters than in the standard architecture. For example, when using a batch size of 64, the memory footprint of the model is 43 GB. This model was trained only for 60 epochs on our teacher training set, ImageNet-1k, reaching a top-1 accuracy of 78% and a top-5 accuracy of 94%, with a global batch size of 10240, using 256 nodes.

The AmoebaNet-D architecture is automatically discovered and we have trained two flavors of it. One with 299x299 input, the other with 480x480 input. In the both cases, the input layers have less neurons than ResNet-59. In the first case, the total number of trainable parameters is also smaller than for ResNet-59. The table below contains a finetuning comparison between the three architectures in terms of throughput, memory consumption and approximate training time when using 256 nodes. Also below we have plotted the mean Area Under the Curve for all these architectures, including the baseline offered by DenseNet-121.

	ResNet-50	ResNet-59	AmoebaNet (4,256), 299x299	AmoebaNet (2,512), 480x480
Training throughput / node [img/s/node]	90	22.5	13.3	3.5
Memory consumption / node/ batch of 64 images [GB]	11	43	61	125
Approximate training time on 256 Zenith nodes [minutes]	8.8	36	53	198
Top-1 accuracy on ImageNet-1K	76.2%	78.1%	79.9%	80.9%
Mean AUROC on ChestXray-14	0.826	0.838	0.840	0.842

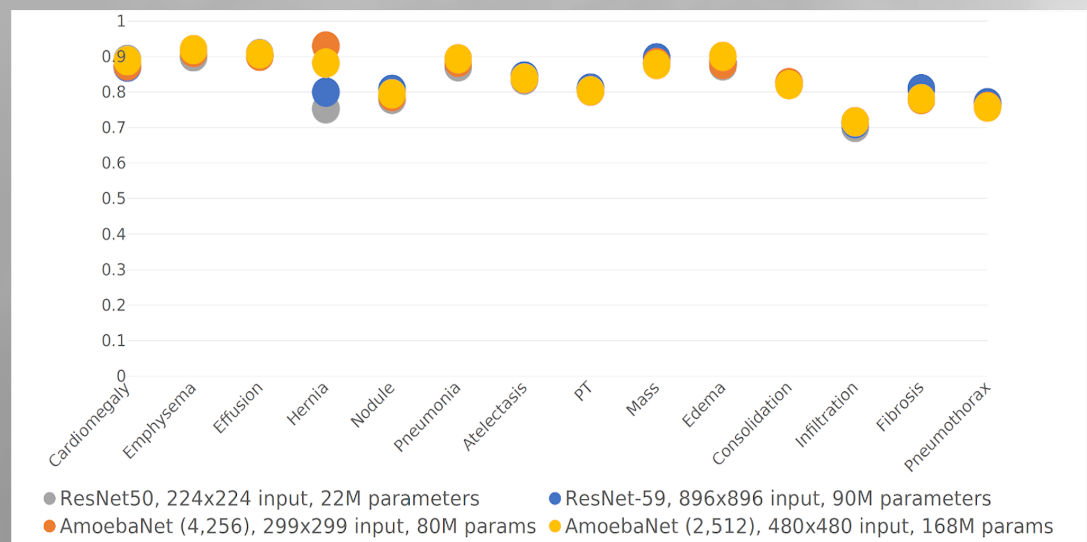




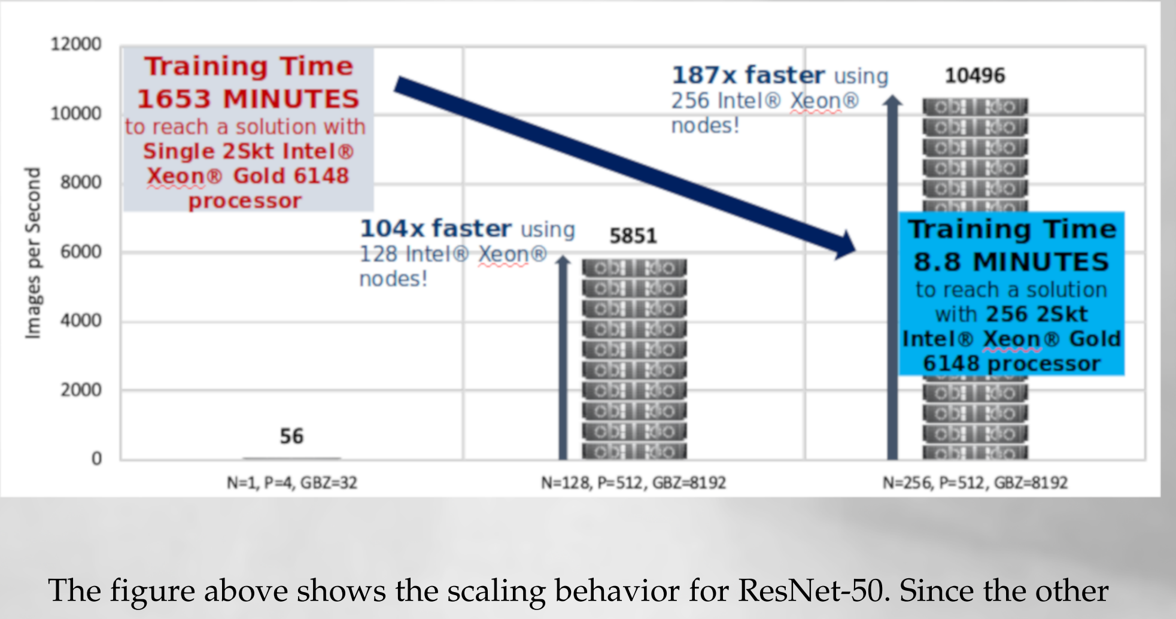
In all these experiments we used a variant of the cyclic learning rate, plotted in the figure on the left-hand side.

The learning rate is increased and decreased in several cycles. Each learning rate inflection point is half of the previous value. The number of cycles determine final accuracy.

This technique ensures that overfitting is reduced, while diversity is obtained from the learning rate variation. This could be expressed also as inside model transfer learning.



Plot showing category accuracy using all the tested architectures



The figure above shows the scaling behavior for ResNet-50. Since the other architectures have more computational demands, the scaling performance is better in their case. Even in the case of ResNet-50, we can observe that by using 256 nodes, the total time required to obtain a trained model is reduced by 187 times.

Conclusions

Scale-out, large-batch training is an effective way to speed up neural network training on unaccelerated Intel® Xeon® platforms. We have shown in this work that we can efficiently leverage supercomputing infrastructures to train models going from moderate scale (e.g. DenseNet), all the way to large, highly accurate models (e.g. large AmoebaNets).

After experimenting with the original DenseNet-121 model and making sure we reach a reasonable baseline, we decided to also evaluate the performance of the very popular ResNet-50 topology. This allowed us improve the mean AUROC from 0.819 to 0.826, a result achieved in similar training time. In order to better extract the details from the ChestXray-14 images that are of higher resolution (1024x1024), we have designed an upscaled version of ResNet-50, called ResNet-59. This improved accuracy on ImageNet-1K from around 76% to 78%, but more importantly improved the mean AUROC on ChestXray-14 from 0.826 to 0.838.

Our next milestone was switching from the residual network architecture to some of the more modern topologies. We chose to experiment with AmoebaNets, since they currently hold the state-of-the art accuracy in various vision tasks. When training a large AmoebaNet model (168 million parameters), we managed to obtain a mean AUROC of 0.842, significantly better compared to the DenseNet-121 baseline, outperforming it in all 14 different pathologies.

For all used network topologies, we first perform a full training run using the ImageNet dataset, followed by transfer learning on the target ChestXray-14 dataset. All training runs are performed using large batches and 128-256 Intel® Xeon®-based compute nodes, showing that the methodology for performing large-batch training at scale applies to both training from scratch and to transfer learning.

Transfer learning and fine tuning approaches are an intelligent way of improving the time-to-solution of models trained on moderate-scale datasets, as reaching the accuracy targets typically significantly fewer training iterations. Fine tuning models already trained on ImageNet (or even other datasets) should be the default approach for those beginning a deep learning project for image classification.

Future work

Although we have significantly improved over the base DenseNet model, we still see that the predictive performance of our best performing models reaches a plateau.

We believe that one option to further improve on the validation accuracy of such a system is to perform additional data augmentation in order to alleviate the inherent class imbalance.

We plan to tackle this problem by using class-conditional generative adversarial networks that have the potential to balance the class distribution in the dataset. With this augmented dataset we plan to re-train the large-input models and expect to reach even higher accuracy levels.

Furthermore, we want to apply the methodology presented in this work to other medical datasets, with a close eye on comparing the feature transferability of ImageNet-pretrained networks against ChestXray-14 pretrained networks on the same target medical datasets. This future work will aim to offer practitioners from the medical diagnostic fields the best practices for applying deep learning techniques to medical imaging data.

References and further reading

CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning

Training an AI Radiologist with Distributed Deep Learning

Diagnosing Lung Disease Using Deep Learning

Efficient Neural Network training on DellEMC Intel Xeon(R) Based Supercomputers

Scale out for large minibatch SGD: Residual network training on ImageNet-1K with improved accuracy and reduced time to train

Fast and Accurate Training of an AI Radiologist