Arti Patel (ap8qk)

Adonis Lu (ayl3yq)

Mariah Hurt (mes3wv)

Cedric Harper (cbh2ta)

Sudeepti Surapaneni (ss9ud)

**SYS 6016 : Final Report**

**A Review and Comparison of CNN Architectures and Classifiers for Pneumonia Detection In X-Ray Images**

**Introduction**

In the midst of the coronavirus pandemic, it is evident that one of the fastest-growing AI applications is medicine. We will need to build robust prognostic models and help health professionals diagnose diseases from image data including x-ray images and biopsies and to improve the clinical decision-making process. Having said that, we wanted to further explore our study of CNNs especially within the domain of medicine and diagnosis. With this project, our goal is to experiment, evaluate and compare the accuracies of different models (varying architectures, pre-trained vs. non-pre-trained models) in addition to implementing different regularization techniques and fine tuning approaches.

**Literature Review**

In a review of studies designed to investigate the use of AI or computer vision in medical diagnostics, it was found that only 6 percent used an external validation set which is necessary for deploying a diagnostic tool in the field, while most studies were focused on implementing algorithms to investigate the feasibility of a diagnostic tool [1]. Our study fits into the category of studies investigating feasibility, while the dataset has been cleaned of low-quality scans and divided into test, train, and validate sets, all the scans come from the same medical institution which makes it impossible to claim that any algorithm resulting from this study would be ready for testing in other medical institutions [1][2]. Machine learning applications are being studied for more than just diagnostics in medical imaging, with studies on image restoration, segmentation, and de-noising being of great interest as well [3]. Due to the nature of our dataset and our objective, we will focus on predictive diagnostic applications.

In medical imaging convolutional neural networks are a common machine learning technique[3]. Transfer learning approaches that employ pre-trained architectures are also common due to the sometimes limited medical datasets[3]. AlexNet and GoogleNet were found to be some of the most useful in classifying medical images [13]. GoogleNet attempts to improve upon AlexNet[??]. Another study, specifically focused on chest x-rays, explored using ResNet50 with different depths and explored incorporating non-image data (such as patient age) along with the images [5]. In their study they found that incorporating the non-image data and using a depth of 38 yielded optimal results [5]. Their study was a multi-class classification problem, focused not just on pneumonia, with a very low sample size for some pathologies so it is very possible that different architectures will better serve our question. They also explored training a network from scratch on chest x-ray data rather than using fine tuning or bottlenecking on a pre-trained network, which is something we will explore as well [5].

Previous studies have investigated using pre-trained CNNs such as VGG, AlexNet, and DenseNet as feature extractors and then using different classifiers such as SVMs, Random Forest, and KNN[6]. A model called CheXNet is a CNN that has been trained on over 100,000 frontal chest X-Rays that has been found to perform nearly as well as trained Radiologists [7]. Another study, with a similar goal of classifying chest X-Rays using a CNN found the pre-trained Xception architecture to be the best choice[8]. Most recently, a study of 1427 X-Rays including COVID-19 patients was conducted similarly using transfer learning from a pre-trained architecture[9]. The two best performing architectures they found were VGG19 and MobileNet v2[9].

A previous study on pneumonia classification first reduced the size of the input images from 1024 x1024 to 224 x 224 in order to limit the amount of computation power required, we may consider doing this as well if needed [6]. Additionally we will look to the literature for guidance on hyperparameter selection. In a study of COVID-19 X-Ray classification, ADAM was used as an optimizer and a batch size of 64 was used [9].

**Data**

Our dataset is from Kaggle, uploaded two years ago with a set of 5,863 chest X-ray images. The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal). We do not include the validation set in our models since it only contains 16 images. The train test split provided to us is about a 90/10 split. The data is labelled in two classes; normal X-Rays and pneumonia positive X-Rays. Chest X-ray images (anterior-posterior) were selected from retrospective cohorts of pediatric patients of one to five years old from Guangzhou Women and Children’s Medical Center, Guangzhou. All chest X-ray imaging was performed as part of patients’ routine clinical care. For the analysis of chest x-ray images, all chest radiographs were initially screened for quality control by removing all low quality or unreadable scans. The diagnoses for the images were then graded by two expert physicians before being cleared for training the AI system. In order to account for any grading errors, the evaluation set was also checked by a third expert.

**Methods/Analysis**

**VGG19-**

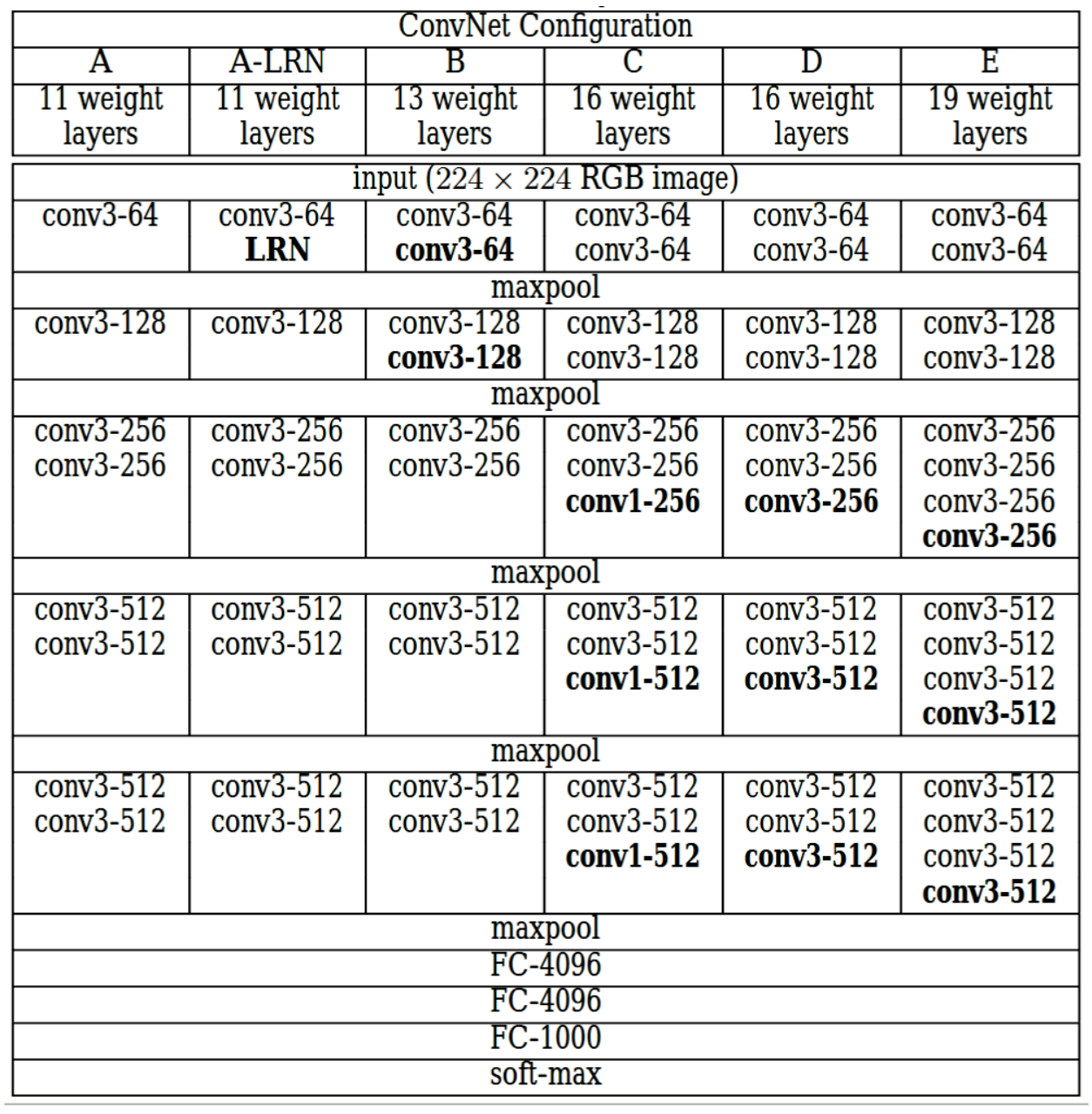
The VGG19 architecture is a variant of the VGG model consisting of 19 layers. It is pretrained on the ImageNet database consisting of 14,197,122 images. VGG is a successor of AlexNet created by Visual Geometry Group at Oxford. It carries some of the same ideas as AlexNet and adds deep convolutional neural layers to improve accuracy

.

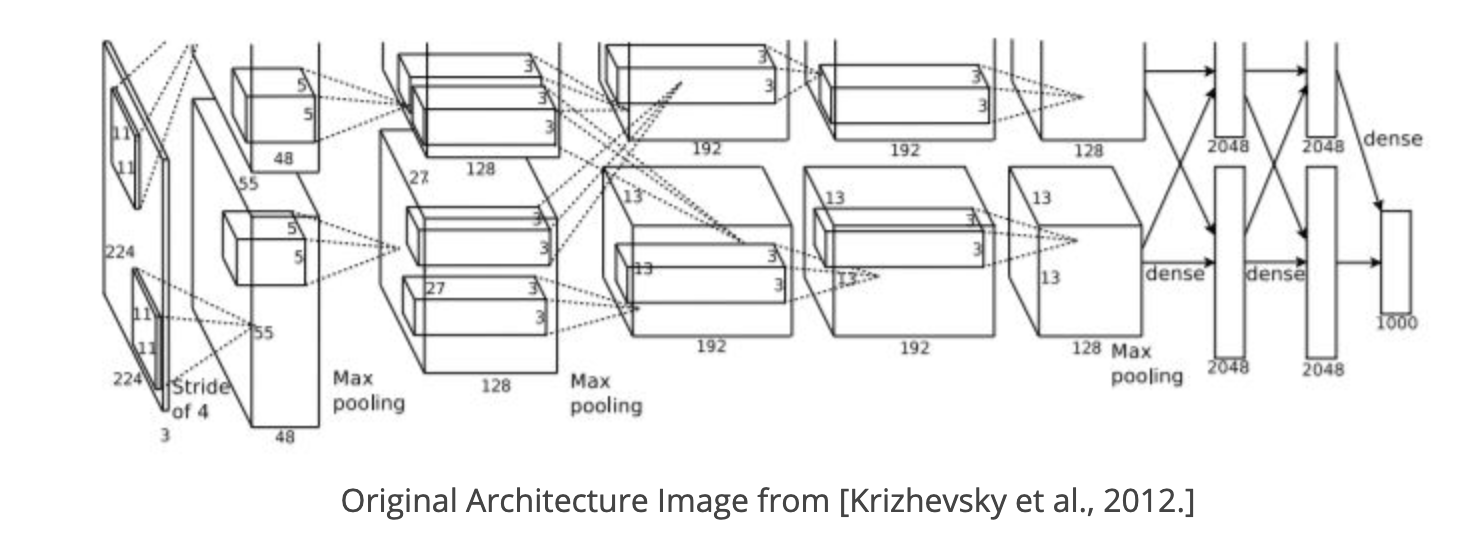
The architecture itself is as follows; a fixed input of 224x224x3 is fed into two convolutional layers with 3x3 kernels and stride 1. This then goes through a max pool layer. This process repeats with increasing number layers and increasing number of nodes, each time followed by max pooling. Max pooling is performed with 2x2 windows and stride 2. This is followed by a rectified linear unit and three fully connected layers. The first two fully connected layers are of size 4096 and the last s of size 1000. Finally, a softmax function is used to classify the output. A summary of the architecture can be found below.

The VGG19 architecture was recently found to be one of the two best performing accuracies compared to other convolutional networks in a study of automatic detection of COVID-19 in chest X-rays [9]. However due to the number of samples in each class, an accuracy metric is not the best to use when comparing models. For this reason, MobileNet was considered the best model due to its highest specificity rate. This limitation of an imbalanced training set can be overcome as an extension of the study.

We use the VGG19 architecture to build three models; one by bottlenecking and another by fine-tuning. We find that the fine-tuned model performs marginally better than the bottenecked model. This could be because our dataset is fairly large with 5,863 images, so the fully connected layers at the end are specific enough to classify the dataset, whereas a smaller dataset would need the initial layers to learn the dataset.

[14]

**AlexNet-**

AlexNet is a convolutional neural network architecture developed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton [10]. A 2019 article from the Journal of Stomatal, Oral, and Maxillofacial Surgery reviewing the use of CNNs in medical image analysis found AlexNet to be one of the top performing architectures[4].It is composed of 5 convolutional layers followed by three fully connected layers, it utilizes relu as the activation function throughout and softmax as the final activation function. You can see by looking at the images of the network, that max pooling is used after three of the convolutional layers. The network does utilize dropout on the fully connected layers which the original AlexNet paper says is done to prevent overfitting [10]. While a pre-trained AlexNet implementation is not available in Keras, it is available via PyTorch. AlexNet with weights from training on ImageNet data can be loaded as a model in PyTorch. Options to finetune are available. We explored how the network performed on the test set data without fine-tuning and with 10 epochs of fine-tuning on our set of training data. 

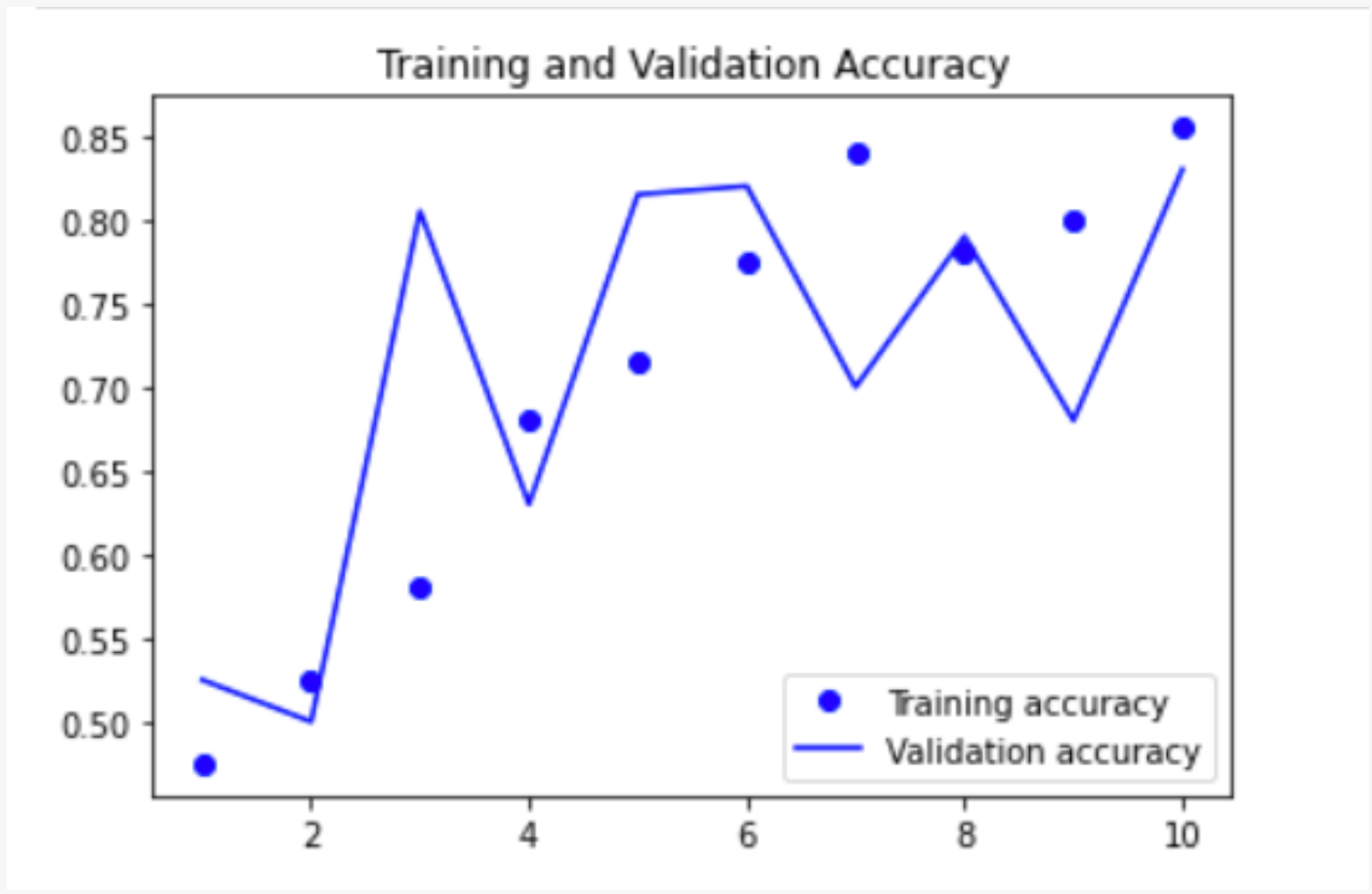
**InceptionV3 -**

Inception v3 is a widely-used and prominent CNN that is the third iteration which has undergone refinements created by Google after InceptionV1 and InceptionV2. The model architecture is based on the original paper: ["Rethinking the Inception Architecture for Computer Vision"](https://arxiv.org/abs/1512.00567) by Szegedy, et. al. It is considered to be “the culmination of many ideas developed by multiple researchers over the years” [13]. Researchers at Google created this model in hopes of creating a deep learning network that will avoid representational bottlenecks and to improve efficiency of computations by using factorization methods[13]. Inception V3 factorizes 5×5 convolution to two 3×3 convolution operations and replaces 7×7 to a series of 3×3 convolutions[13]. Given these enhancements, InceptionV3 is proven to have a lower computation cost than VGGNet. Inception V3 consists of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, dropouts, and fully connected layers. All convolutional layers are followed by batch normalization and ReLU activation. We decided to use InceptionV3 because we found it to be a very effective and viable option for transfer learning. Given the fact that InceptionV3 has well-fit parameters and bottleneck layers with highly optimized representations of the input data, we decided to retrain our final layer with our dataset. We adopted this transfer learning approach because we 1) are working with a small training set and 2) wanted to leverage the pre-trained weights and the pre-built architecture associated with InceptionV3.

**Results**

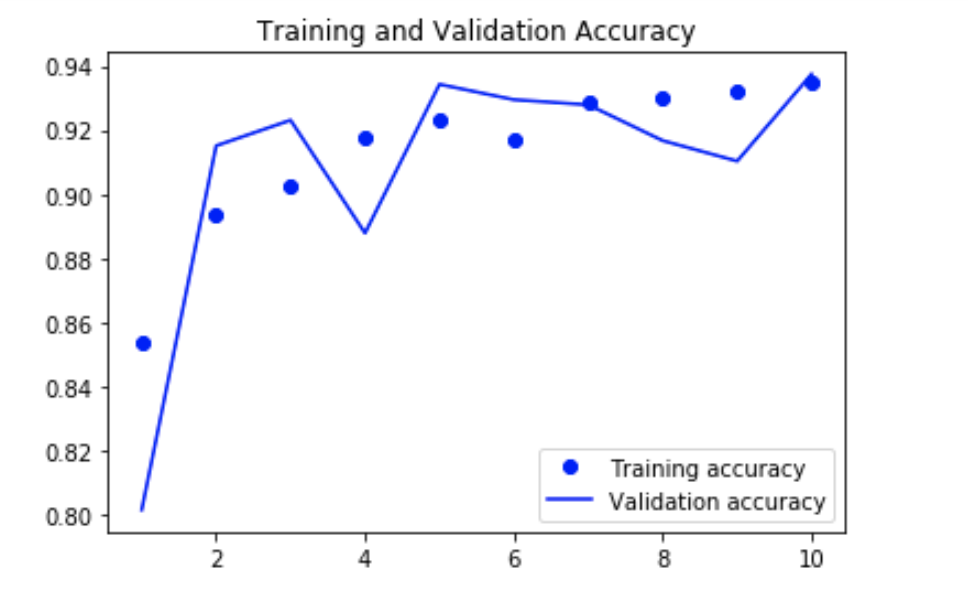
**VGG19 -**

The VGG19 architecture was used to produce three models for our dataset.The first two are the bottleneck andfine-tune versions. For the third model, we unlocked all of the layers in the VGG19 architecture and let them train on our dataset, essentially using the same architecture to build our model. The bottleneck model was built by passing our training and test sets into the VGG19 and then saved both the training and validation features into arrays. We then train a small model on top of the VGG19 which consists of one dense layer, one dropout layer of 0.5, and finally a sigmoid activation to produce our binary classification. The fine-tuned model is built by freezing the initial convolutional layers in VGG19, and then training two fully connected layers specific to our dataset. The final model that is built by unlocking all the VGG19 ultimately fit the data best. Both the training and validation accuracy achieved 85% accuracy by the tenth epoch. This model also does not seem to overfit the way the bottleneck and fine-tuned models do.



**AlexNet-**

We found that the pre-trained AlexNet network without any fine-tuning classified X-Ray images at a rate very close to random chance for a binary classification problem, with a 46% accuracy rate. We then turned to fine-tuning. We allowed the Alex-Net model with the pre-trained weights to train for 10 epoches on the training data. This greatly improved the accuracy to 93.7%.



**InceptionV3 -**

Similar to the two other models, we allowed the InceptionV3 model with the pre-trained weights to train for 10 epochs on the training data. By adopting a transfer learning approach, we retrained our final layer. We built a new model from the InceptionV3 by instantiating a new model with the original image input and the bottleneck layer from the InceptionV3 as output. Given that this was a binary classification problem, we inserted a sigmoid activation in the last (output) layer. We adopted various approaches and tested different parameters (RMS vs Adam, different augmentation parameters). For our final model, we decided to adopt these parameters when augmenting the data as it was proven to work for our dataset and achieve 89 percent accuracy in a paper from the Journal of Big Data[7].

Rotation range = 3,

width shift range = 0.05,

height shift range = 0.05,

shear range = 0.05,

zoom range = 0.05,

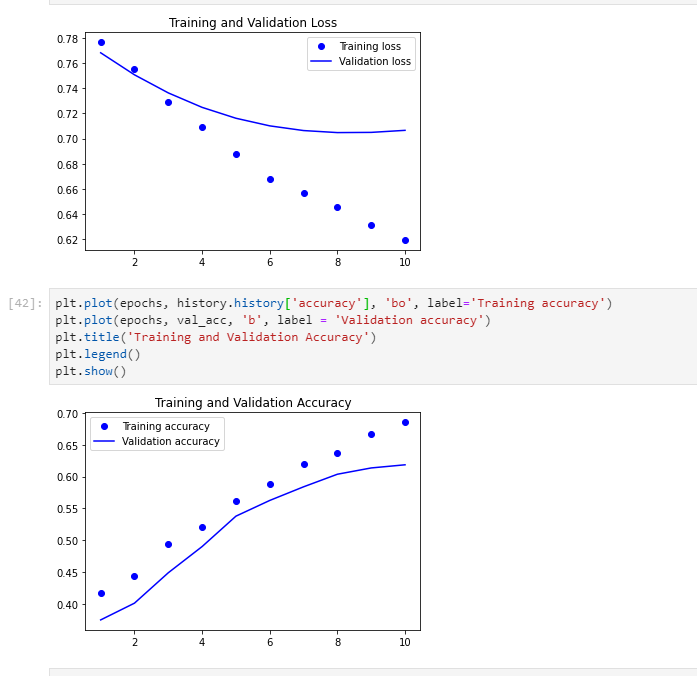
fill mode = ’constant’,

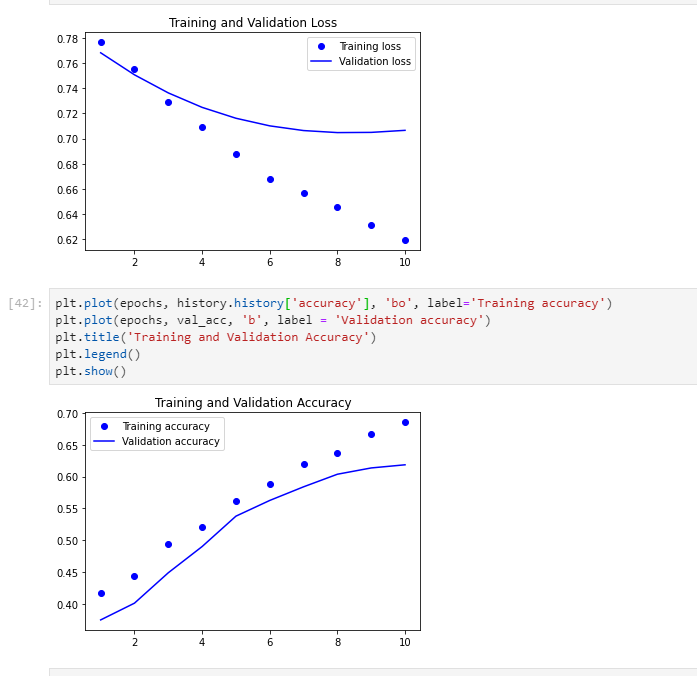
cval = 0.,

horizontal flip = True

vertical flip = True

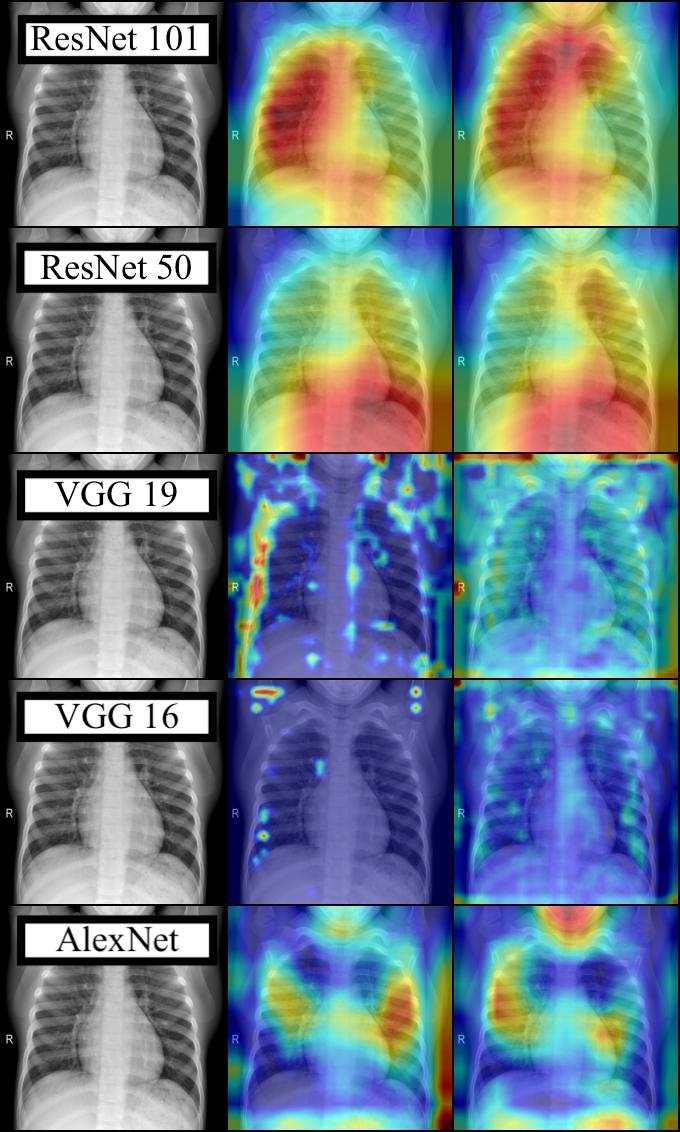
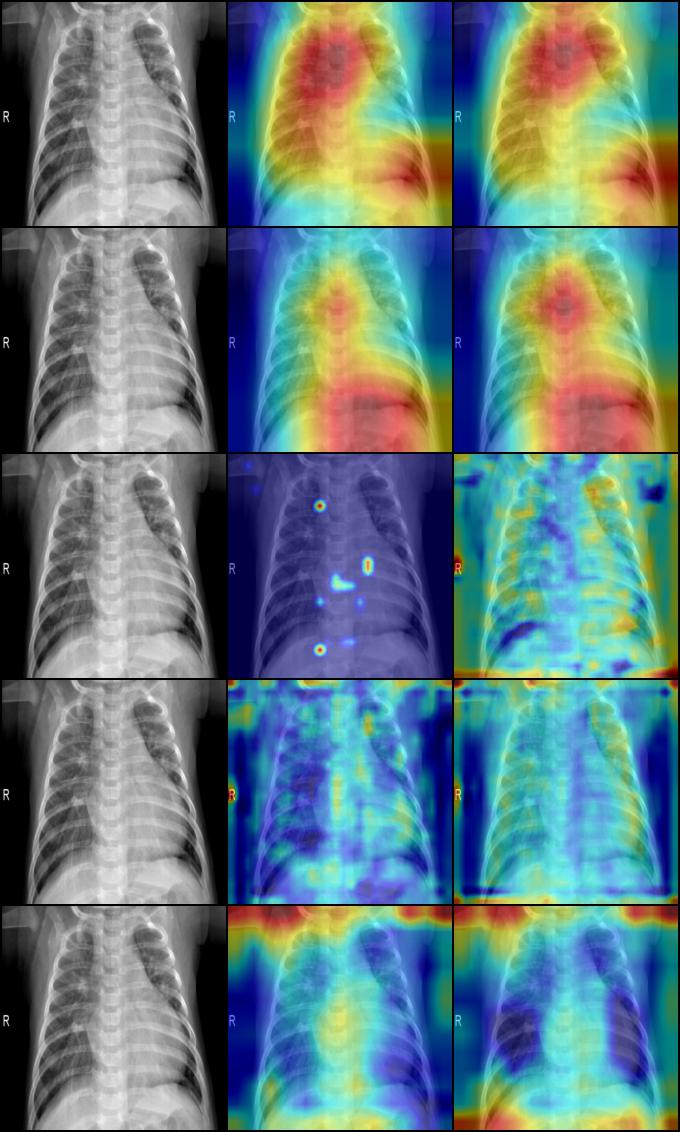
The earlier iterations of our model were overfitting such that our training accuracy ranged 88 to 93 percent but our validation accuracy was ranging from 65 to 76 percent. For our final model (as pictured below), we lowered the learning rate to 1e-6 and tried to prevent overfitting to a certain extent. However, the accuracy seems to plateau. Given the time and computational limitations, we were not able to increase the number of epochs. It would be interesting to see whether or not the accuracy and loss would continue to follow the same pattern if we increased the number of epochs. We would have explored further methods and techniques associated with finetuning, optimization with respect to image data in the form of x-rays or sought ways in which we can expand our dataset.

****

****

**Grad-CAMS -**

We wanted to use these Grad-CAMS which stand for Gradient-Weighted Class Activation Mappings to qualitatively assess the performance of our multiple models by comparing the results of their application to select images within the dataset. The Grad-CAM results were obtained by using the Pytorch module, GradCam. After loading our ResNet 101, ResNet 50, VGG19, VGG16, and AlexNet models, we applied Grad-CAM functions. As pictured below, Grad-CAMs produce localization maps that highlight specific areas of the x-rays that the model deems most important while making the classification decision. We implemented the methods for a residual network architecture by extracting the activation values from an intermediate convolution layer and using them to generate a heat map identifying the important areas of a given x-ray. It specifically leverages the gradient information about classes as it goes to the final layer of a convolutional neural network in order to produce a rough localization map of the important regions in the image. The most important regions rely on the calculations of the most significant neurons for a specific class[11].

****

Furthermore, we feel that these Grad-CAMS inject interpretability and transparency to our models. We compared various models using Grad-CAMS because we felt that visualizing activation areas will potentially allow domain experts (health professionals) to corroborate the model results based on common practices and guidelines. Viewing the images through the lens of the Grad-CAMS will allow pulmonologists or radiologists to confirm if the models are classifying the images based on real anatomical or biological features that make domain sense.

**Conclusion**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Training Loss** | **Training Accuracy** | **Validation Loss** | **Validation Accuracy** |
| **Custom** | **0.0725** | **.987** | **2.9422** | **.7919** |
| **VGG19** | **0.0413** | **.9882** | **3.6979** | **.7661** |
| **AlexNet** | **0.1660** | **0.9350** | **0.1778** | **0.9375** |
| **InceptionV3** | **0.6196** | **0.6852** | **0.7066** | **.6184** |

In conclusion, AlexNet performed better than InceptionV3 and VGG19 over the course of 10 epochs. The AlexNet and InceptionV3 models seemed to overfit the least, while the custom and VGG19 model overfit the most. There may be some correlation with the results of our heatmaps with the results of our model. The heatmaps our GradCams produced indicated that AlexNet tends to perform well with highlighting opaque areas while the highlights in the VGG models tend to be more splotchy and harder to interpret.

In the future, we would explore further methods associated with finetuning, optimization and data augmentation with respect to image data in the form of x-rays. We also would dive deeper into understanding how the processes and guidelines domain experts adhere to when making a classification decision. We feel that this would help us engineer the models especially guide the feature mapping process. Currently, our model only takes into account chest X-rays, but a combination of features including symptoms, comorbidities, and age of the patients can also be useful when making a decision. This project could be extended to classifying different types of pneumonia; bacterial vs viral (Multiclass Classification) and furthermore, different types of lung infections (influenza , tuberculosis, bronchitis) or respiratory diseases. Due to the higher cost of false negatives in medical diagnoses, a different metric that weighs importance to different types of misclassification may be useful. Loss functions that places a higher cost onto false negatives and metric such as specificity can be used to extend this project.

**Citations**

[1] “KoreaMed Synapse.” https://synapse.koreamed.org/search.php?where=aview&id=10.3348/kjr.2019.0025&code=0068KJR&vmode=FULL (accessed Apr. 26, 2020).

[2]Z. Li, M. Dong, S. Wen, X. Hu, P. Zhou, and Z. Zeng, “CLU-CNNs: Object detection for medical images,” *Neurocomputing*, vol. 350, pp. 53–59, Jul. 2019, doi: [10.1016/j.neucom.2019.04.028](https://doi.org/10.1016/j.neucom.2019.04.028).

[3] A. S. Lundervold and A. Lundervold, “An overview of deep learning in medical imaging focusing on MRI,” *Zeitschrift für Medizinische Physik*, vol. 29, no. 2, pp. 102–127, May 2019, doi: 10.1016/j.zemedi.2018.11.002.

[4] A. Fourcade and R. H. Khonsari, “Deep learning in medical image analysis: A third eye for doctors,” *Journal of Stomatology, Oral and Maxillofacial Surgery*, vol. 120, no. 4, pp. 279–288, Sep. 2019, doi: 10.1016/j.jormas.2019.06.002.

[5]I. M. Baltruschat, H. Nickisch, M. Grass, T. Knopp, and A. Saalbach, “Comparison of Deep Learning Approaches for Multi-Label Chest X-Ray Classification,” *Scientific Reports*, vol. 9, no. 1, pp. 1–10, Apr. 2019, doi: [10.1038/s41598-019-42294-8](https://doi.org/10.1038/s41598-019-42294-8).

[6]“Pneumonia Detection Using CNN based Feature Extraction - IEEE Conference Publication.”<https://ieeexplore.ieee.org/abstract/document/8869364/> (accessed Apr. 28, 2020).

[7]S. S. Yadav and S. M. Jadhav, “Deep convolutional neural network based medical image classification for disease diagnosis,” *J Big Data*, vol. 6, no. 1, p. 113, Dec. 2019, doi: [10.1186/s40537-019-0276-2](https://doi.org/10.1186/s40537-019-0276-2).

[8]S. Mondal, K. Agarwal, and M. Rashid, “Deep Learning Approach for Automatic Classification of X-Ray Images using Convolutional Neural Network,” in *2019 Fifth International Conference on Image Information Processing (ICIIP)*, Nov. 2019, pp. 326–331, doi: [10.1109/ICIIP47207.2019.8985687](https://doi.org/10.1109/ICIIP47207.2019.8985687).

[9]I. D. Apostolopoulos and T. A. Mpesiana, “Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks,” *Phys Eng Sci Med*, pp. 1–6, Apr. 2020, doi: [10.1007/s13246-020-00865-4](https://doi.org/10.1007/s13246-020-00865-4).

[10]A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[11]Selvaraju, Ramprasaath R. et al. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization.” International Journal of Computer Vision (2019): n. pag. Crossref. Web.

[12]A. K. Jaiswal, P. Tiwari, S. Kumar, D. Gupta, A. Khanna, and J. J. P. C. Rodrigues, “Identifying pneumonia in chest X-rays: A deep learning approach,” Measurement, vol. 145, pp. 511–518, Oct. 2019, doi: 10.1016/j.measurement.2019.05.076.

[13]C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” *arXiv:1512.00567 [cs]*, Dec. 2015.

[14] Aakash Kaushik. Understanding the VGG19 Architecture. <https://iq.opengenus.org/vgg19-architecture/>