Pneumonia Detection in Chest X-Rays using Deep Transfer Learning

ECE457B COURSE PROJECT, Winter 2020

Atpouthan Paskaran, 20627995, a2paskar@edu.uwaterloo.ca

Xiule Fan, 20602179, x54fan@edu.uwaterloo.ca

Andrew Robertson, ad3rober@edu.uwaterloo.ca

(Cluster 2)

Friday, April 3rd, 2020

Abstract

Advancement in technology has increased the image data available in the health care sector. Normally these images are reviewed by health care professionals. However, the increasing amount of data has inevitably brought higher workload to them and also increased the possibility of human error [1]. The objective of this project is to leverage deep learning and other machine learning algorithms to correctly classify data from medical images. In particular, the theme of the project will encompass classification of chest X-ray images into the classifications of normal and pneumonia. With the power of machining learning, a relatively simple algorithm can be trained to identify these different characteristics with great precision within a couple of hours. This algorithm could be used to help doctors make diagnosis about pneumonia more quickly. Doctors can then reallocate the time saved from diagnosis to other tasks such as determining treatment or ordering further tests for the patient.

This classification problem is tackled by different techniques such as convolutional neural networks and transfer learning. A convolutional neural network can often complete an image classification task with high precision. In transfer learning, a model which has been trained on other datasets is used for a new problem. Using the pre-existing model can lead to more accurate results or lower computing time. After implementing transfer learning with a variety of pre-trained neural networks including VGG16, InceptionResNetV2, InceptionV3, ResNet50 and Xception, it was found that a customized VGG16 pre-trained network produced the highest validation accuracy of 86.54%- making it a reliable option for a classifier for this problem.

Table of Contents

1. Introduction

Advancement in image acquisition devices and computing in recent years has brought the health care sector into a big data era in which more and more medical image data has been collected in hospitals. Traditionally, medical images are reviewed and interpreted by health care professionals. However, the increasing amount of data has inevitably brought higher workload to doctors and radiologists [1]. When they are working under heavy workload and high pressure, human error is more likely to occur, which will be problematic for their patients. Luckily, advancement in technology also provides tools like machine learning to help solve this problem.

In this project, application of various machine learning techniques is explored to classify pneumonia X-ray images. The group chose pneumonia specifically due to some of the epidemics and pandemics that occurred in the past twenty years. For example, pneumonia is one common symptom of disease caused by COVID-19, SARS-CoV and MERS-CoV [2]. Compared to normal people's X-ray images, pneumonia patients' X-ray images show distinct white spots in the lung area [3]. This unique feature is ideal for the application of machine learning.

With the use of machine learning, a model can be trained to identify this feature. Doctors can use such a model to diagnose if a patient has pneumonia or distinguish the specific type of pneumonia. Using such an algorithm can help doctors screen their patients more quickly so that they can focus on suggesting treatment or further testing.

In a regular hospital setting, this algorithm may not be extremely helpful since the number of patients with pneumonia is probably relatively low. However, as mentioned above, pneumonia is one of the common symptoms in recent epidemics and pandemics. When one of these outbreaks occurred, hospitals received a significantly higher number of patients. Being able to identify if a patient has pneumonia or not is key to making a fast diagnosis, which is important to give proper treatment to the patient and more importantly, control the outbreak. This type of automated detection of the most serious cases could be especially useful when hospitals are under-staffed due to a large influx of patients.

To accomplish this task of developing an algorithm to classify pneumonia in chest X-ray images, a publicly available dataset is utilized. The classifier is built by implementing a convolutional neural network for feature extraction with a fully-connected classifier. Additionally, transfer learning with pre-trained networks is also explored to verify its performance for this task. After the classifier is trained successfully, test data is sent into the network to test how accurate the model is.

2. Motivation

Although there exists many other algorithms for the task of image classification, convolutional neural networks (CNN) have shown outstanding performance in recent years. Compared to other methods, one of the main advantages of CNNs is that they can detect features from an image automatically [4]. When using other methods, users typically need to preprocess the image data to extract features before applying the methods. A CNN, however, can learn how to extract these features based on the image's pixels [5]. Additionally, implementation of a CNN generally requires less memory compared to a comparable standard neural network, like a fully connected network. Oftentimes, it takes longer time to train the comparable standard neural network and the performance is not as good as the one generated by a CNN.

One limitation of standard neural networks and CNNs is that they rely on a large dataset in order to achieve satisfactory results. If the dataset for a particular problem does not contain enough information, these models will fail the classification task. To remedy this problem, transfer learning is used. When using transfer learning, a pre-existing model is adapted to solve a new problem. The pre-existing model has been trained on datasets which might not be related to the new problem. However, by leveraging knowledge from the previous datasets, it is possible to achieve more accurate results faster by using less training data [6].

3. Background

This section provides background information on topics that are relevant to the proposed problem and solution.

3.1 Pneumonia

Pneumonia is a type of infection that affects one or both lungs. Mild pneumonia may bring discomfort to patients, while severe cases can become life-threatening. Pneumonia can be caused by a number of different pathogens but it is most commonly the result of a bacterial and viral infection [7]. A common diagnosis procedure for pneumonia is through the examination of chest X-ray images. As shown in Figure 1, the chest X-ray images for a normal person, a person with bacterial pneumonia and a person with viral pneumonia have distinct differences. Patients with bacterial pneumonia typically exhibit a focal lobar consolidation which is indicated by the white arrows in the figure. On the other hand, chest X-ray images of patients with viral pneumonia often shows a more diffuse pattern [8]. Based on these differences, doctors can give diagnosis and offer further treatment. These differences are also the basis of building an artificial neural network for classification.

Figure 1: Pneumonia in Chest X-ray Images [8]

3.2 Machine Learning

Machine learning is a subset of artificial intelligence that focuses on the study of algorithms and mathematical models to allow computer systems to perform specific tasks without explicit instructions. Instead, with machine learning, machines can rely on pattern recognition to make important decisions. Machine learning has a wide range of applications in many industries such as banking, linguistics, marketing, robotics and many more [9]. Machine learning algorithms require large data sets in which processing techniques can be used to extract important features which are used to train the algorithm to make accurate predictions or perform specific tasks. Machine learning algorithms are able to learn from their respective datasets through learning paradigms such as supervised, unsupervised and reinforcement learning. In supervised learning paradigms, the dataset that the algorithm is meant to learn from, are structured into input features and corresponding labels. By having access to the appropriate label of the given data, machine learning algorithms are able to learn the corresponding relationship that maps the input space to the output space.

3.3 Artificial Neural Networks (ANNs)

ANNs are connectionist machine learning models used to solve problems in pattern classification or function interpolation. ANNs contain layers of computational elements called neurons that are all interconnected by weights. Figure 2 depicts an ANN with a single hidden layer along with its neurons and weights (depicted as arrows).

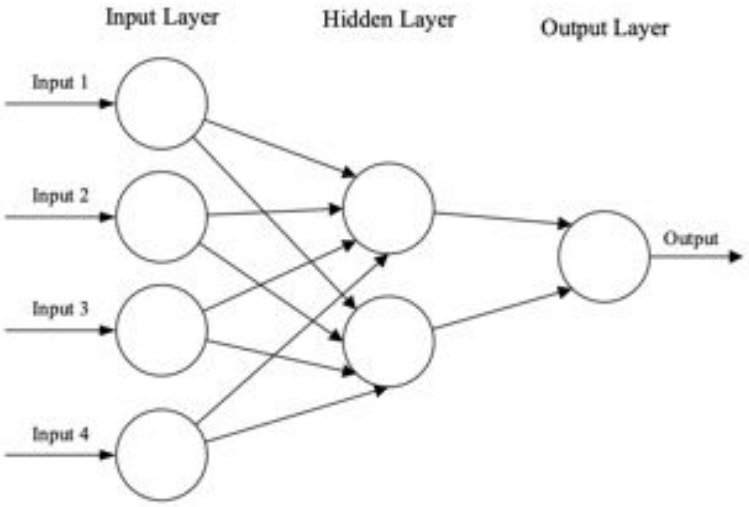


Figure 2: Representation of an ANN [10]

ANNs are able to learn from the labeled dataset (referred to as the training process) by back propagating the gradient of the error between the actual label and the predicted label from the network and use this information to make appropriate updates to the weights of the network.

3.4 Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNNs) are specialized ANNs that are primarily used in the field of pattern recognition for images. The specialized layers of CNNs which include convolutional and pooling layers allow these networks to learn high level feature representations within the image data, which can then be used in conjunction with traditional dense/ fully connected layers to solve pattern recognition problems [10]. In contrast, the labelled datasets for traditional ANNs contain input features which are hand-engineered from raw data, whereas to reiterate, CNNs are able to extract trainable features from the raw data. Figure 3 depicts a simple CNN with a convolutional layer, followed by a pooling layer and some two fully connected layers leading to the output classification layer.

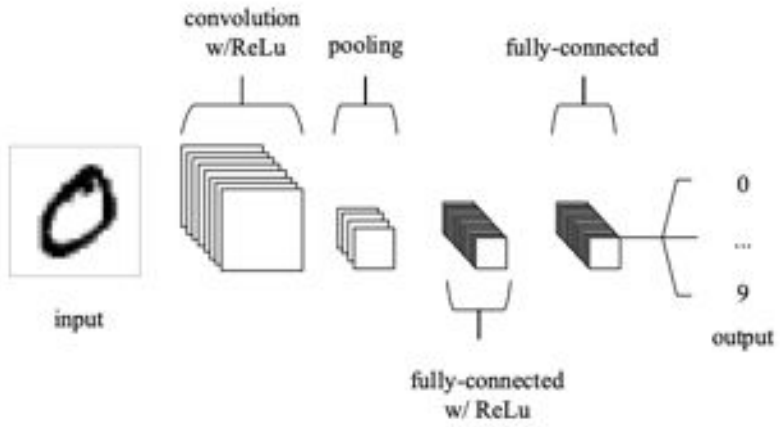


Figure 3: Representation of a CNN [10].

3.4.1 Convolutional Layer

This specialized layer of a CNN applies the convolution operation using filters (of weights) in order to convert the input image (expressed as 3D array) into a feature map which contains the most important details of the original image (decided by the network during training). The basic methodology of the convolution operation is demonstrated in Figure 4.

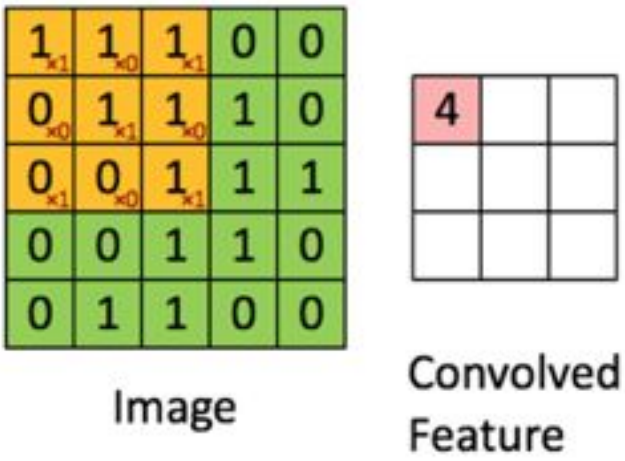


Figure 4: Convolution Operation of a 3x3 Filter

The advantage of utilizing a convolutional layer as opposed to a regular fully connected layer is that the number of parameters utilized in a convolutional layer are significantly reduced. For instance, if the input to a CNN is an image of size 64x64x3 (RGB) and a 6x6(x3) filter is used in the convolutional layer, then the number of parameters for this layer becomes 6x6x3=108 weights. In contrast, if a fully connected layer was used in a standard ANN, the first layer would contain 64x64x3=12,288 weights [10]. This reduction in parameters of the network decreases

the complexity of the CNN (as well as computational complexity) to combat against overfitting and reducing training time [10].

3.4.2 Pooling layer

This specialized layer of a CNN simply reduces the dimensionality of the feature maps generated by a convolutional layer, by either extracting the max value feature of a given region in the feature map or the average value feature of a given region in the feature map, denoted as max pooling layer and average pooling layer respectively. Although the pooling layer leads to a loss of data from the feature map, researchers have shown that this loss of data produces a regularization effect, ultimately making the model extract more general features, while reducing the total number of parameters of the network (since pooling layers are fixed functions with no trainable parameters) [10]. Figure 5 illustrates the two main types of pooling layers.

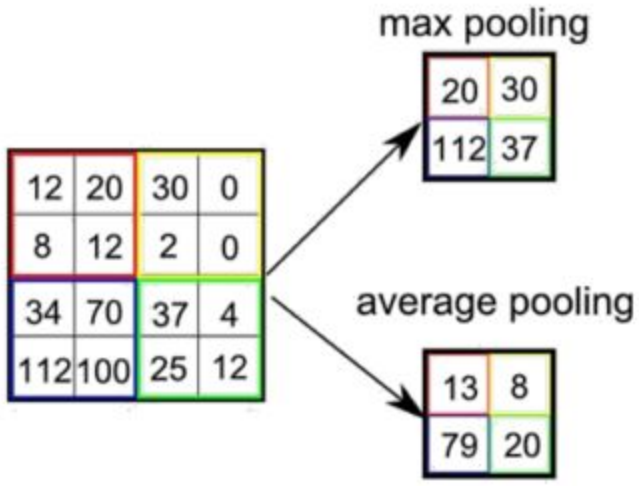


Figure 5: Result of the two types of pooling layers, using 2x2 filter

3.5 Transfer Learning

Transfer learning is the ability to transfer knowledge between tasks. It is primarily used as a tool to solve the problem of insufficient training data for a certain task [11]. Specifically, the type of transfer learning used in this project is called transductive transfer learning. It is a learning process in which knowledge and insights of a pre-trained model for a certain task, learned using a large-scale dataset, are transferred to a new model to aid its learning of a new, similar task with limited training data. These insights from the pre-trained model are represented as the optimized weights that were learned during the training process, which can be directly inserted into a new network of the same architecture (layers, neurons per layer, etc.), or portions of the pre-trained network could be simply inserted as a part of a new network [12]. Then, by training the newer layers of the concatenated network (freezing the weights of the pre-trained network) using the insufficient training data could still yield good performance. This is because the

feature extractions learned previously by the various layers in the pre-trained network have already been trained to identify common elements like shapes and edges. This can prove to be useful for the new task, allowing training of the new task to be more effective [12]. In addition to performance benefits, by implementing transfer learning, training can be made significantly quicker since training a new model from scratch could take many weeks and multiple GPUs, but by only needing to train the outer layers of the new concatenated network results in a decreased amount of required computation. The concept of transfer learning is illustrated in Figure 6.

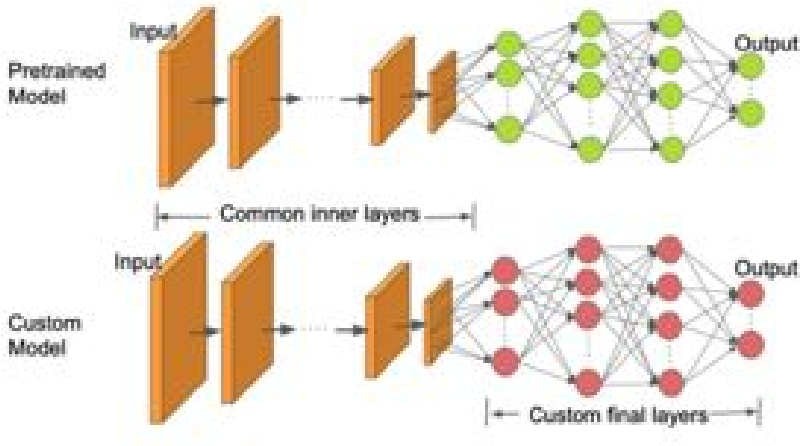


Figure 6: Pre-trained Model Architecture Leveraged for Transfer Learning

4. Solution

The section presents the implementation of the solutions to the proposed problem.

4.1 Dataset

The training and validation data used in this project is collected from a publicly available dataset of chest x-ray images. This dataset contains 5863 images of labelled x-ray images of normal healthy individuals and those with viral or bacterial pneumonia [13]. The original dataset is divided into training and validation sets. The dataset was prepared by resizing each image in the training and validation data to a size of 200x200 pixels and reading the RGB components into an array.

The available dataset contained around twice as many of normal x-ray images as it did images of patients with pneumonia. To avoid encountering a vanishing gradients problem during training where the model becomes biased to only predicting that the x-ray is normal, the number of training data points from each class was equalized. This was done using a RandomUnderSampler that randomly selects an equal number of datapoints from each class from the original training and validation datasets. After processing the original dataset, 2682 samples are used for training and 468 samples are used for validation. This is rather limited

compared to the extensive training datasets typically used for image classification tasks which contain hundreds of thousand or even millions of images. This motivates the need to use a transfer learning approach for solving the classification problem.

4.2 Basic CNN Model

To determine a baseline for the performance of the transfer learning approach, a basic CNN model is first trained on the collected dataset. The model takes as input the 200x200 RGB components of the image. The construction of the model is shown below in Figure 7.

```
input_shape = (200, 200, 3)
model = Sequential()
model.add(Conv2D(16, kernel_size=(3, 3), activation='relu', input_shape=input_shape))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(1, activation='sigmoid'))
```

Figure 7. Construction of Basic CNN Model

The sigmoid activation function is used because this is a binary classification task. The output is rounded to either 0 or 1, with 0 representing a normal x-ray image, and a 1 representing an x-ray image of a patient with pneumonia. The Adam optimizer is used with a decreased learning rate of 1e-4. This is done to prevent bouncing around an optimum during training, or allowing the model to quickly reach a local maximum resulting in overfitting. Since this is a binary classification task, the binary cross entropy loss function is used. The network is trained for 9 epochs, using a batch size of 32.

4.3 Transfer Learning with Pre-Trained CNN Models

After determining the baseline performance of a simple CNN, pre-trained CNN models are leveraged with transfer learning for the classification task. Without knowing which existing network will extract features that are relevant for this classification task, a variety of networks are explored. These include VGG16, InceptionResNetV2, InceptionV3, ResNet50 and Xception. Each of these nets has been pre-trained on the ImageNet dataset and is available through the

Keras Applications API. Each model and the model weights are loaded without their top layers so that a new fully-connected classifier may be appended to each model.

4.3.1 Dense Output Layer

Transfer learning is first carried out by flattening the output of the topless model and appending only an output layer with a single unit. This output layer is an attempt at a simple mapping of features extracted from the pre-trained model to a class prediction. The weights of the pre-trained model are frozen and only the weights connecting the flattened layers to the output layer are trained. The construction of this simple transfer learning system is shown in Figure 8.

```
base_model = pretrainedmodel # Topless
# Add top layer
x = base_model.output
x = Flatten()(x)
predictions = Dense(1, activation='sigmoid')(x)
model = Model(inputs=base_model.input, outputs=predictions)
# Train top layer
for layer in base_model.layers:
    layer.trainable = False
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```

Figure 8. Construction of Simple Transfer Learning System

4.3.2 Adding Additional Dense Layers

The transfer learning system is complexified in the hope of achieving better classification performance by changing the fully-connected classifier from just an output layer to a classifier that includes two dense layers with 512 units and ReLU activation functions and one dense output layer with a single output unit and sigmoid activation. Dropout layers with a 0.3 dropout rate are also added after the dense layers to help prevent overfitting in the fully-connected classifier. Again, the weights of the pre-trained model are frozen and only the weights of the fully-connected classifier are trained. The construction of this fully-connected classifier is shown in Figure 9.

```
base_model = pretrainedmodel # Topless
# Flatten output layer of base model
x = base_model.output
x = Flatten()(x)
base_model = Model(inputs=base_model.input, outputs=x)
for layer in base_model.layers:
    layer.trainable = False
print("Model before adding dense layers")
base_model.summary()
# Add dense layers
input_shape = base_model.output_shape[1]
model = Sequential()
model.add(base_model)
model.add(Dense(512, activation='relu', input_dim=input_shape))
model.add(Dropout(0.3))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(1, activation='sigmoid'))
print("Model after adding dense layers")
model.compile(loss='binary_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```

Figure 9. Construction of Transfer Learning System with Additional Dense Layers

4.3.3 Fine-Tuning

The transfer learning system is further optimized by unfreezing certain layers of the best performing model that was seen thus far. In general, the earlier layers of CNNs extract common features like edges and shapes and the later layers extract task specific features. This strategy of unfreezing layers attempts to discard pre-trained task specific features in order to learn the features that are relevant to this pneumonia detection task. It is shown in Section 5 that VGG16 is most adept at feature extraction for this pneumonia classification task. An attempt is made to further optimize the classifier performance by fine-tuning the last two convolutional blocks of the VGG16 network. If the transfer learning method using VGG16 with two additional dense layers is the most performant model, the process of fine tuning that network is shown below in Figure 10.

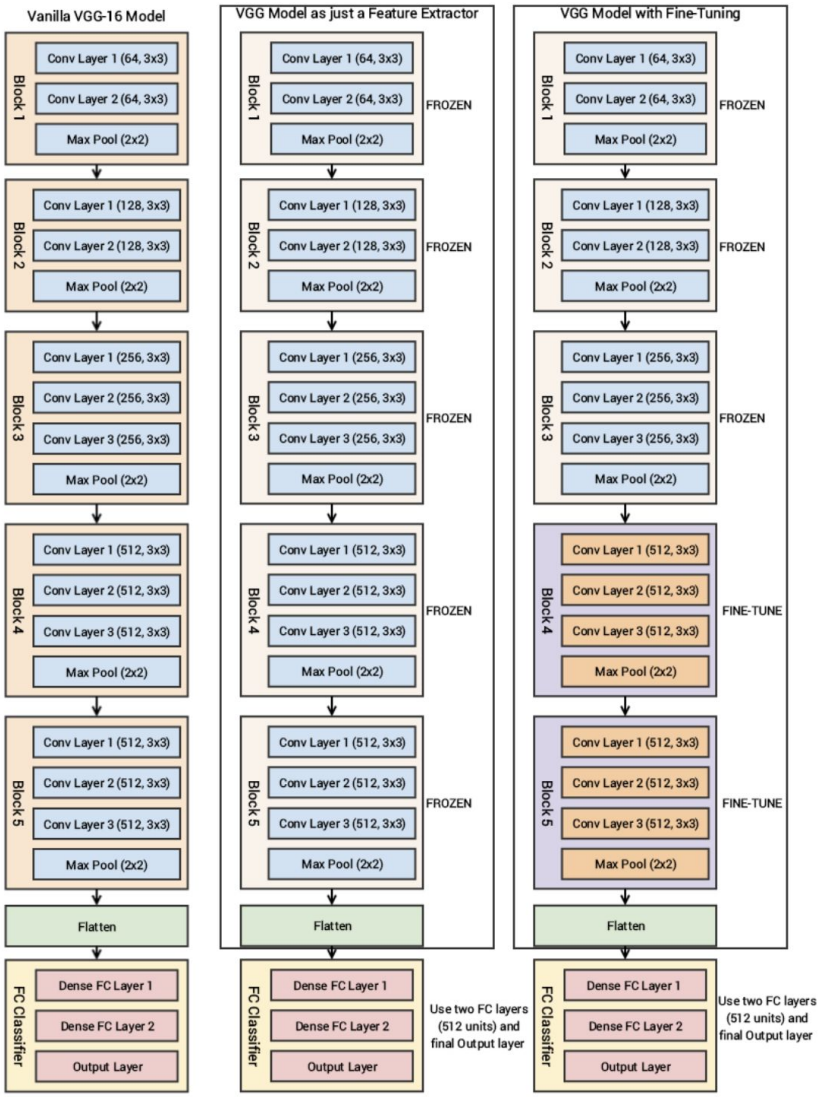


Figure 10. Transfer Learning Strategies on VGG16 Model

5. Results and Analysis

This section presents the training and validation results for each permutation of the classifier and provides some analysis on why the proposed solutions achieve these results.

5.2 Basic CNN Model

The training and validation accuracy and loss results for the basic CNN model are shown below in Figure 11.

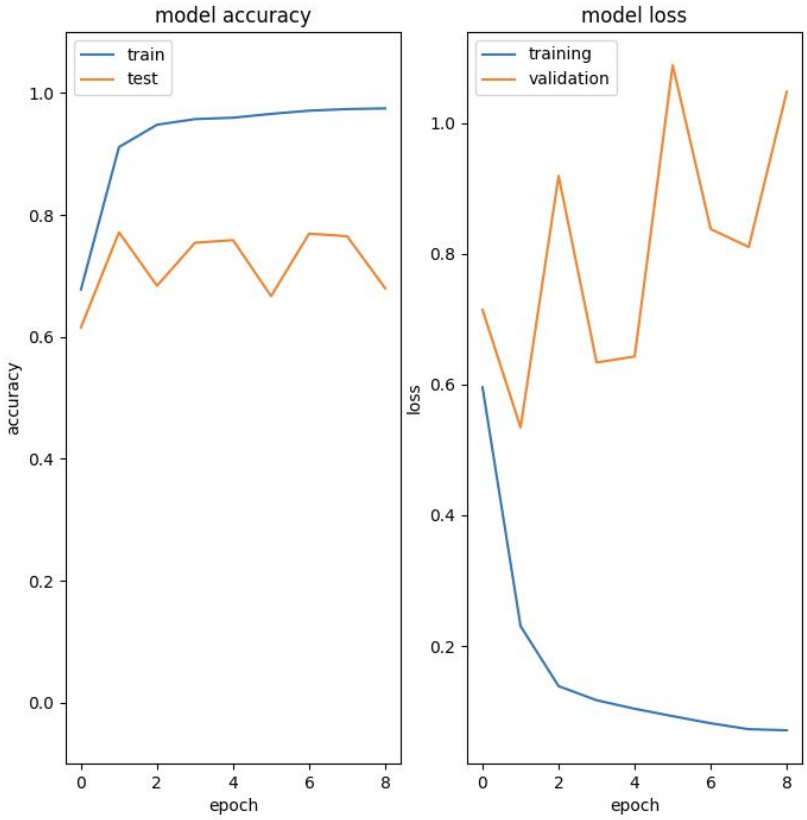


Figure 11. Training and Validation Results of Basic CNN Model

The accuracy curve shows training accuracy increasing across epochs, as expected. But the validation accuracy does not seem to improve past the second epoch. This can be explained by the loss curve, which shows that although training loss is decreasing, validation loss is increasing. This indicates overfitting to the training dataset by the basic CNN model. This would be expected given the limited size of the training set. The accuracy results are specifically highlighted in Figure 12 to show the training epoch which gives the highest validation accuracy of 77.14%.

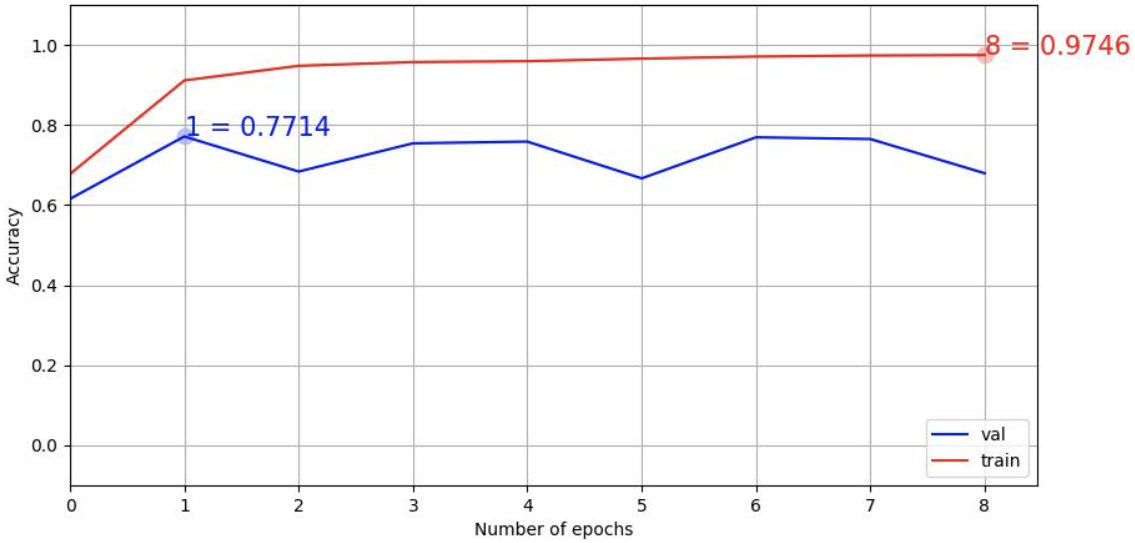


Figure 12. Accuracy Results of Basic CNN Model

5.3 Transfer Learning with Pre-Trained CNN Models

This section presents the results and analysis for various transfer learning strategies.

5.3.1 Dense Output Layer

The peak validation accuracy results across all training epochs for each base model used in the transfer learning method with only a dense output layer appended are shown in Table 1.

Table 1. Peak Validation Accuracies with only a Dense Output Layer

Base Model	Peak Validation Accuracy
VGG16	85.68%
InceptionResNetV2	63.68%
InceptionV3	50%
ResNet50	78.63%
Xception	82.48%

Both VGG16, ResNet50 and Xception all outperform the basic CNN model. This shows that using these networks for feature extraction yields positive results for the pneumonia classification task. InceptionResNetV2 and InceptionV3 did not produce features that were able to be used effectively for the task of pneumonia classification with only a single dense output layer. VGG16 clearly outperforms the other base models used for feature extraction with a peak validation accuracy of 85.68%. Figure 13 presents a closer look at the accuracy and loss performance of the simple VGG transfer learning method on the training and validation sets.

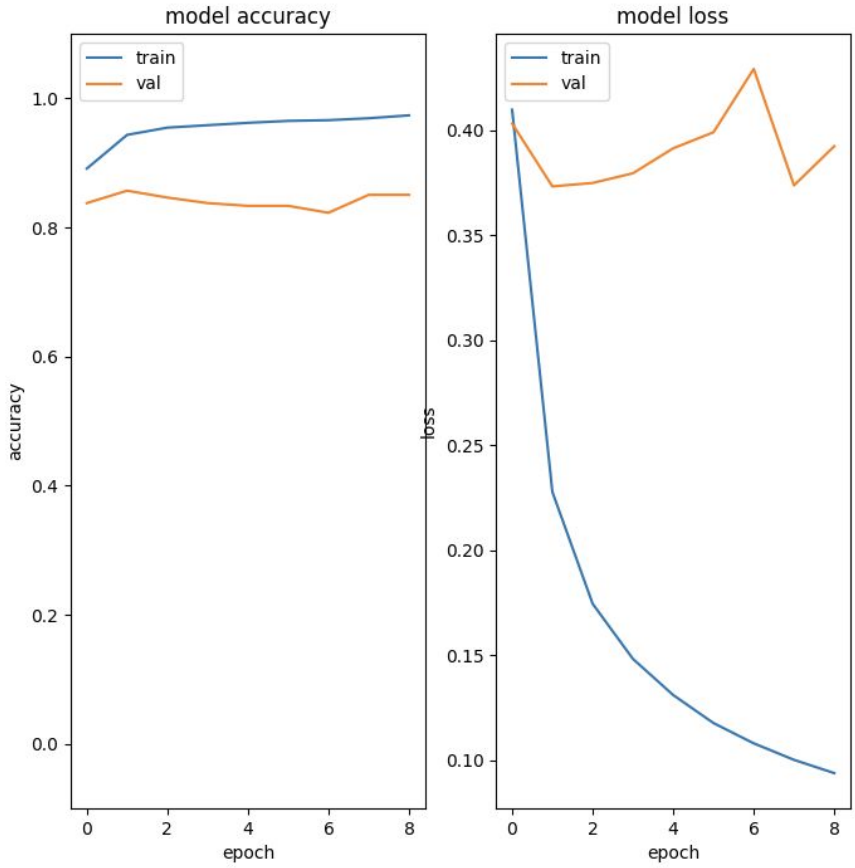


Figure 13. Training and Validation Results of Simple Transfer Learning Method using VGG16

Figure 13 shows that validation accuracy and loss remain relatively constant across training epochs, meaning that the classifier is not becoming overfit to the training data across epochs. Although an accuracy results of 85.68% on the classification task is commendable, it may be possible to improve the models performance by adding additional dense layers to the fully-connected classifier that is appended to each model.

5.3.2 Adding Additional Dense Layers

The peak validation accuracy results across all training epochs for each base model used in the transfer learning system with two additional dense layers of 512 units are shown in Table 2.

Table 2. Peak Validation Accuracies after Adding Two Additional Dense Layers

Base Model	Peak Validation Accuracy
VGG16	86.32%
InceptionResNetV2	53.21%
InceptionV3	50%
ResNet50	78.85%
Xception	77.14%

Adding additional dense layers to the base models does not seem to achieve great results. It offers a marginal improvement in the performance for the models using VGG16 and ResNet50 as the base model. However it results in a degradation of performance when InceptionResNetV2, InceptionV3 and Xception are used for feature extraction. Figure 14 shows the training and validation accuracy and loss for VGG16 with additional dense layers.

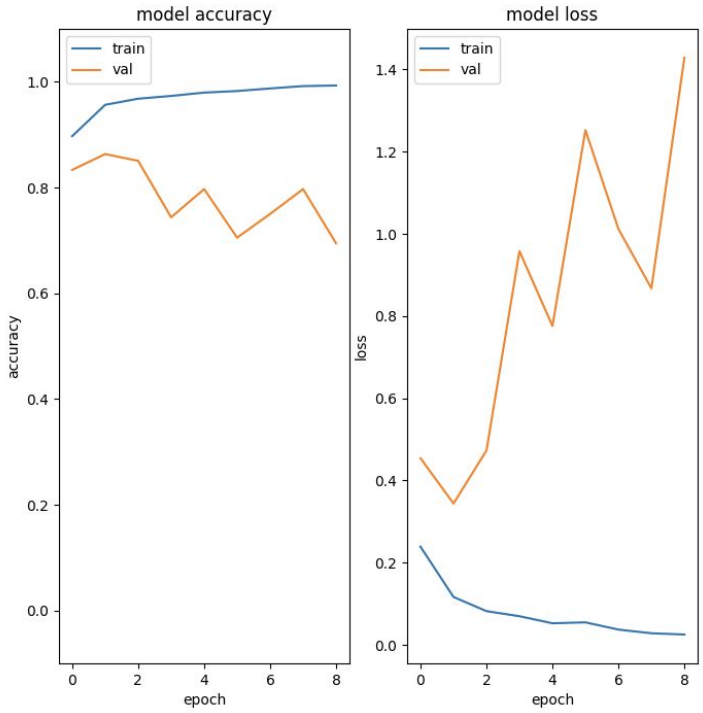


Figure 14. Training and Validation Results of Transfer Learning Method using VGG16 with Additional Dense Layers

Figure 14 shows that validation loss is increasing across epochs after the second epoch. This means that the fully-connected classifier is becoming overfit to the training data. Since the additional dense layers only offer a minor performance improvement and causes the model to suffer from overfitting, the model that uses VGG16 as a feature extractor and only has a single dense output layer is used during fine-tuning.

5.3.3 Fine-Tuning

The accuracy results for fine-tuning of the model using VGG16 for feature extraction with trained weights on the last two convolutional blocks and a dense output layer are shown in Figure 15.

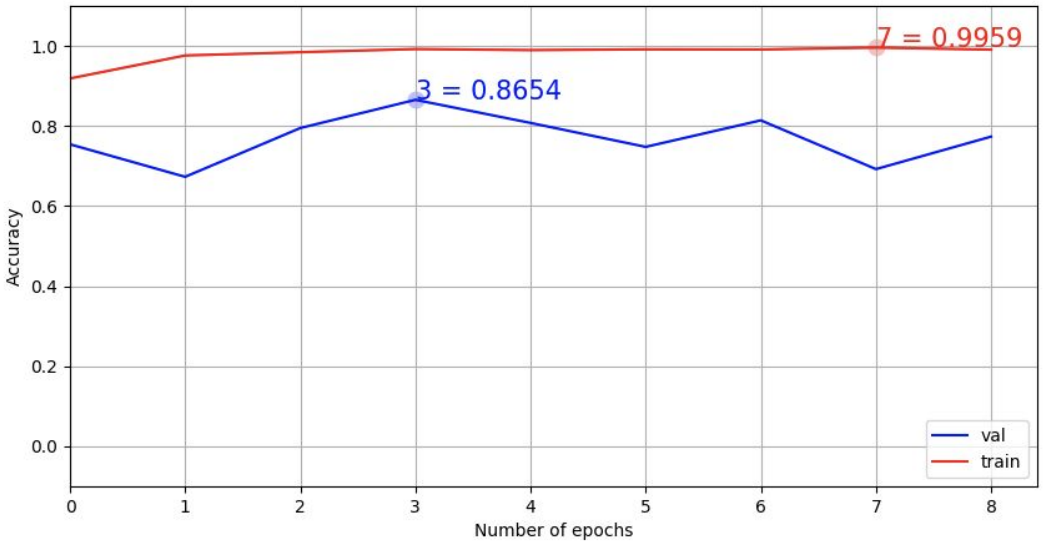


Figure 15. Accuracy Results after Fine-Tuning

The fine-tuned network using VGG16 as a base model with a dense output layer yields a peak validation accuracy of 86.54%. Fine-tuning offers a modest increase to the peak performance of the frozen model. These results are comparable to the most performant publicly available kernel for the same dataset which uses a similar transfer learning approach with Xception and achieves a validation accuracy of 87% [8].

The confusion matrix for the classifier's predictions on the validation dataset is shown in Figure 16.

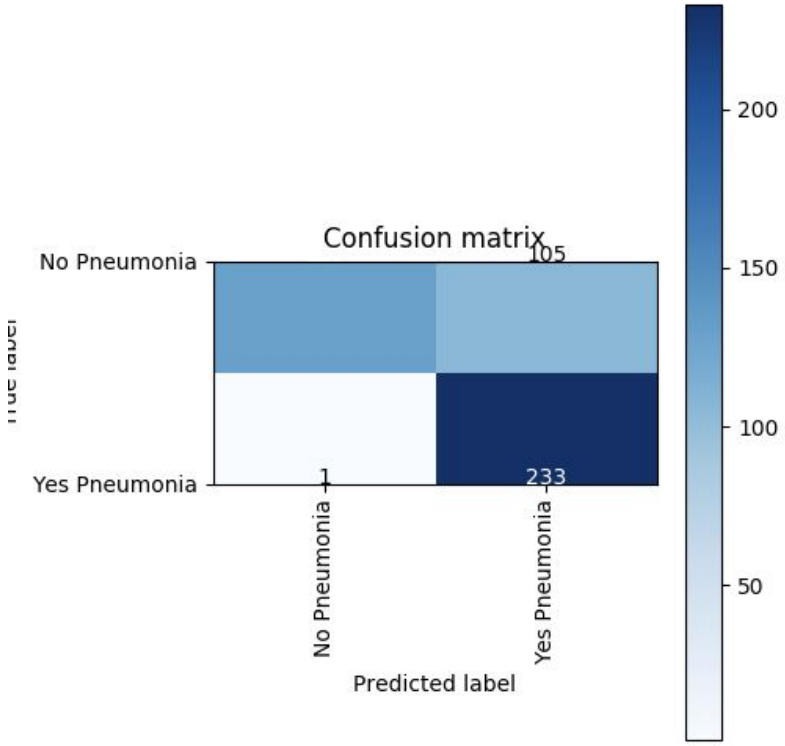


Figure 15. Confusion Matrix of Fine-Tuned Model Predictions on Validation Set

The confusion matrix shows that the model is more biased towards predicting that a provided X-ray image contains a patient with pneumonia. This is desirable, as it would be detrimental to have many incorrect predictions on patients that actually have pneumonia. It also does not do much harm to falsely predict that a healthy individual has pneumonia.

6. Conclusion

Convolutional neural networks are exceptionally well suited for image classification tasks. Classifying pneumonia in chest X-ray images using CNNs would be beneficial to health professionals for rapid diagnosis in situations when resources are stretched thin. Using VGG16 as a feature extractor with a dense output layer provides and fine-tuning the last two convolutional blocks of the network yields a peak accuracy of 86.54%. Furthermore, the model is modestly biased towards predicting false positive results. This would hopefully lead to very few missed diagnoses of pneumonia when exercising the classifier on test images. Ultimately, the classifier provides an effective solution for detecting pneumonia in chest X-ray images.

References

[1] M. Razzak, S. Naz, and A. Zaib, “Deep learning for medical image processing: overview, challenges and future,” 2017, *arXiv*:1704.06825.

[2] B. L. Tesini, “Coronaviruses and acute respiratory syndromes (COVID-19, MERS, and SARS),” *Merck Manual*, 2020. [Online]. Available: https://www.merckmanuals.com/en-ca/professional/infectious-diseases/respiratory-viruses/coronaviruses-and-acute-respiratory-syndromes-covid-19,-mers,-and-sars. [Accessed Apr. 2, 2020]

[3] “Pneumonia,” *RadiologyInfo.org*, Jan. 23, 2019. [Online]. Available: https://www.radiologyinfo.org/en/info.cfm?pg=pneumonia. [Accessed Apr. 2, 2020].

[4] A. Dertat, “Applied deep learning - part 4: convolutional neural networks,” *Towards Data Science*, Nov. 8, 2017. [Online]. Available: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2. [Accessed Apr. 3, 2020].

[5] S. Bhatt, “Understanding the basics of CNN with image classification,” *becominghuman.ai*, Oct. 4, 2019. [Online]. Available: https://becominghuman.ai/understanding-the-basics-of-cnn-with-image-classification-7f3a9ddea8f9. [Accessed Apr. 3, 2020].

[6] D. Sarkar, “A comprehensive hands-on guide to transfer learning with real-world applications in deep learning,” *Towards Data Science*, Nov. 14, 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a. [Accessed Apr. 3, 2020].

[7] Mayo Clinic, "Pneumonia," Mar. 13, 2018. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/pneumonia/symptoms-causes/syc-20354204. [Accessed Apr. 2, 2020].

[8] K. S. Daniel, M. Goldbaum, W. Cai and et al., "Identifying medical diagnoses and treatable diseases by image-based deep learning," *Cell,* vol. 172, no. 5, pp. 1122-1131, 2018.

[9] I. Valchanov, “Machine learning, an overview,” *Oracle AI and Data Science Blog*, Jan. 25, 2018. [Online]. Available: https://blogs.oracle.com/datascience/machine-learning%3a-an-overview. [Accessed Apr. 3, 2020].

[10] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," 2015, *arXiv*:1511.08458.

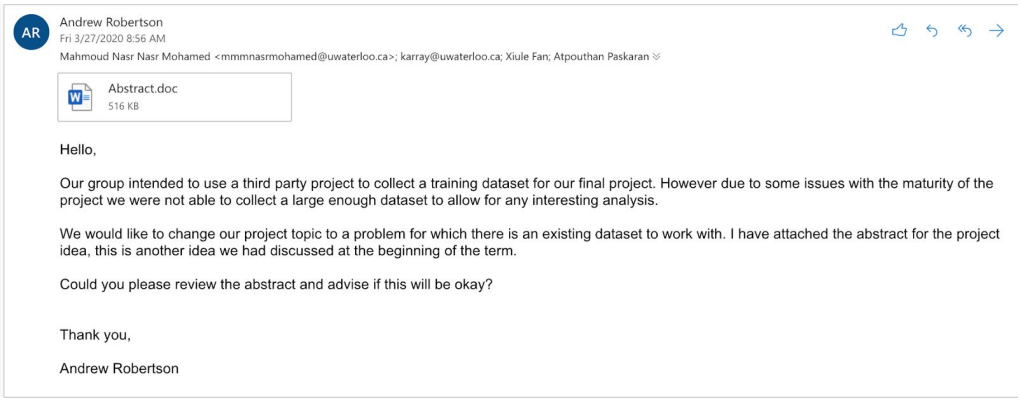
[11] C. Tan, F. Sun, T. Kong and et al., “A survey on deep transfer learning,” 2018, *arXiv*:1808.01974.

[12] S. Nayak, “Image classification using transfer learning in PyTorch,” *Learn OpenCV*, May 20, 2019. [Online]. Available: https://www.learnopencv.com/image-classification-using-transfer-learning-in-pytorch/. [Accessed Apr. 3, 2020].

[13] P. Mooney, “Chest X-ray images (pneumonia),” Kaggle, 2018. [Online]. Available: https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia. [Accessed Apr. 3, 2020].

Appendix

Our group encountered many issues with the idea we originally proposed. Because of this we switched project topics ~2 weeks before the Apr 3 due date. The email conversation describing the circumstances is included below as per the request of the course instructor.



AR

Andrew Robertson
Fri 3/27/2020 8:56 AM
Mahmoud Nasr Nasr Mohamed <mmmnasrmohamed@uwaterloo.ca>; karray@uwaterloo.ca; Xiule Fan; Atpouthan Paskaran

Abstract.doc
516 KB

Hello,

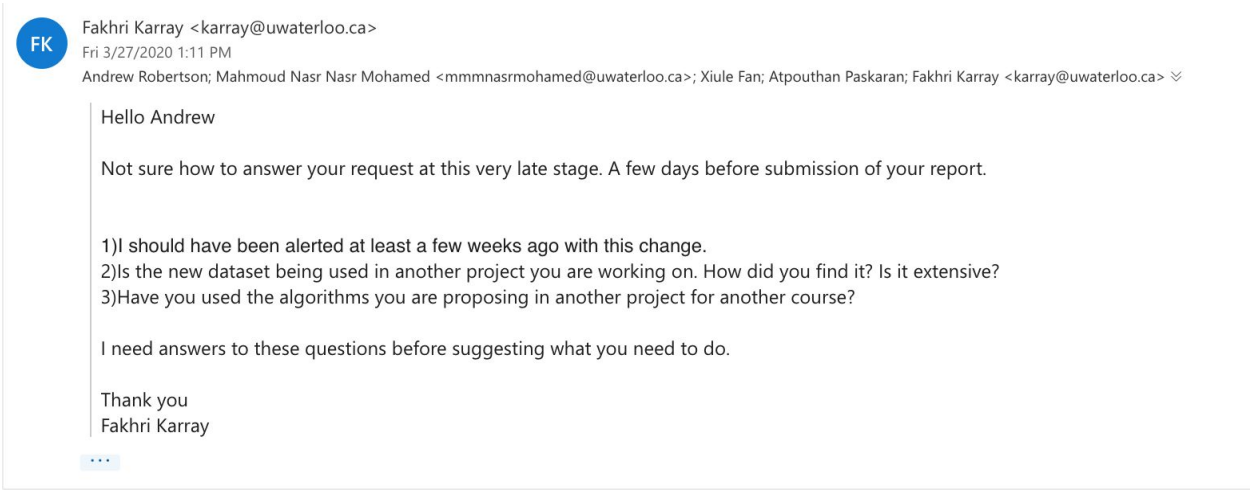
Our group intended to use a third party project to collect a training dataset for our final project. However due to some issues with the maturity of the project we were not able to collect a large enough dataset to allow for any interesting analysis.

We would like to change our project topic to a problem for which there is an existing dataset to work with. I have attached the abstract for the project idea, this is another idea we had discussed at the beginning of the term.

Could you please review the abstract and advise if this will be okay?

Thank you,

Andrew Robertson



FK

Fakhri Karray <karray@uwaterloo.ca>
Fri 3/27/2020 1:11 PM
Andrew Robertson; Mahmoud Nasr Nasr Mohamed <mmmnasrmohamed@uwaterloo.ca>; Xiule Fan; Atpouthan Paskaran; Fakhri Karray <karray@uwaterloo.ca>

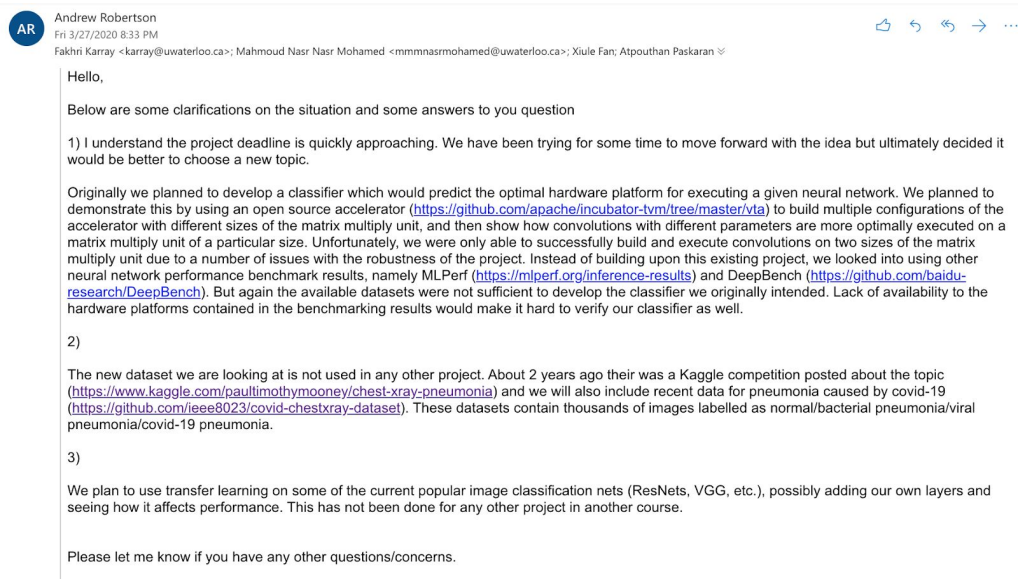
Hello Andrew

Not sure how to answer your request at this very late stage. A few days before submission of your report.

1)I should have been alerted at least a few weeks ago with this change.
2)Is the new dataset being used in another project you are working on. How did you find it? Is it extensive?
3)Have you used the algorithms you are proposing in another project for another course?

I need answers to these questions before suggesting what you need to do.

Thank you
Fakhri Karray



AR

Andrew Robertson
Fri 3/27/2020 8:33 PM
Fakhri Karray <karray@uwaterloo.ca>; Mahmoud Nasr Nasr Mohamed <mmmnasrmohamed@uwaterloo.ca>; Xiule Fan; Atpouthan Paskaran

Hello,

Below are some clarifications on the situation and some answers to you question

1) I understand the project deadline is quickly approaching. We have been trying for some time to move forward with the idea but ultimately decided it would be better to choose a new topic.

Originally we planned to develop a classifier which would predict the optimal hardware platform for executing a given neural network. We planned to demonstrate this by using an open source accelerator (https://github.com/apache/incubator-tvm/tree/master/vta) to build multiple configurations of the accelerator with different sizes of the matrix multiply unit, and then show how convolutions with different parameters are more optimally executed on a matrix multiply unit of a particular size. Unfortunately, we were only able to successfully build and execute convolutions on two sizes of the matrix multiply unit due to a number of issues with the robustness of the project. Instead of building upon this existing project, we looked into using other neural network performance benchmark results, namely MLPerf (https://mlperf.org/inference-results) and DeepBench (https://github.com/baidu-research/DeepBench). But again the available datasets were not sufficient to develop the classifier we originally intended. Lack of availability to the hardware platforms contained in the benchmarking results would make it hard to verify our classifier as well.

2)

The new dataset we are looking at is not used in any other project. About 2 years ago their was a Kaggle competition posted about the topic (https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia) and we will also include recent data for pneumonia caused by covid-19 (https://github.com/ieee8023/covid-chestxray-dataset). These datasets contain thousands of images labelled as normal/bacterial pneumonia/viral pneumonia/covid-19 pneumonia.

3)

We plan to use transfer learning on some of the current popular image classification nets (ResNets, VGG, etc.), possibly adding our own layers and seeing how it affects performance. This has not been done for any other project in another course.

Please let me know if you have any other questions/concerns.

FK

Fakhri Karray <karray@uwaterloo.ca>
Sat 3/28/2020 12:53 PM
Andrew Robertson; Mahmoud Nasr Nasr Mohamed <mmmnasrmohamed@uwaterloo.ca>; Xiule Fan; Atpouthan Paskaran

OK please proceed with the new project. I also need you to highlight this email exchange in your report as an appendix so I am reminded of your circumstances.

Good luck and make sure to make it as comprehensive as possible (with qualitative results and reliable benchmark comparisons) and that your results are reproducible. Some of my TAs will be asked to reproduce some of the results that are obtained in the various projects.

Thanks
Fakhri