An Analysis of Transfer Learning using the SimCLR Contrastive Learning Framework

by
Tanner Skluzacek

Submitted in fulfillment
of the requirements for the
Mathematics Senior Capstone Project

at the
University of Minnesota-Twin Cities
May 2022

Capstone Advisor:
Jeff Calder, Professor, Mathematics

Contents

Chapter 1

Introduction

Contrastive learning is a machine learning technique that has shown great performance in unsupervised and semi-supervised learning. In 2020 a group of researchers released a paper outlining a new framework for contrastive learning that combines years of work on the topic to create a simple but effective framework (Chen et al. 2020). This framework, called SimCLR, has great performance on ImageNet classification compared to previous approaches. Additionally, the SimCLR framework has been shown to be effectively applied to transfer learning problems where the pretrained model is frozen and used to learn classification on a different dataset. This report gives an overview of the SimCLR paper and looks at the effectiveness of using the SimCLR network to apply transfer learning to classifying chest radiographs related to COVID-19 detection.

Chapter 2

Background on SimCLR and Contrastive Learning

2.1 What is Contrastive Learning

The two main approaches in machine learning are supervised learning and unsupervised learning. In supervised learning, models learn over time by using accuracy metrics from the known labels of the dataset. In unsupervised learning, however, models discover hidden relationships in the data without the need for labels and do some sort of clustering and segmentation. Additionally, the term semi-supervised learning refers to training in which a small subset of the dataset is labeled to be used in training. Labeling data can be a very costly task, making semi-supervised and unsupervised learning very relevant for many applications.

Contrastive learning is an approach where a model aims to maximize agreement between similar data and maximize disagreement between dissimilar data. A model using this approach, for example, could learn to distinguish between similar and dissimilar images for the purpose of image classification. This approach often uses few or no labels at all and the networks are relatively simple.

2.1.1 Early Contrastive Learning

The standard contrastive learning approach is to learn classification tasks by contrasting positive pairs against negative pairs. This methodology was introduced in a 2006 paper (Hadsell, Chopra, and LeCun 2006). In this paper, they research dimensionality reduction in which high dimensional inputs are mapped to low dimensional spaces where locality of inputs is preserved. This means that "similar" points in the input are near each other in the mapped lower dimension space. Their contrastive learning approach was to use an energy based model that uses neighborhood relationships to create the mapping function. For a family of functions G, parameterized by W, the goal is to find a value of W that maps the inputs into a lower dimensional space such that the euclidean distance in this space, $D_W(\overrightarrow{X_1}, \overrightarrow{X_2}) = ||G_W(\overrightarrow{X_1}) - G_W(\overrightarrow{X_2})||_2$ approximates the semantic similarity of the inputs from the set of neighborhood relationships.

The paper goes on to define the methodology and contrastive loss function. Let $\mathcal{I}$ be the set of high dimensional training vectors $\overrightarrow{X_i}$ that define the input. For each $\overrightarrow{X_i} \in \mathcal{I}$,

let $S_{\overrightarrow{X_i}}$ define the set of training vectors that are similar to $\overrightarrow{X_i}$. Next, the loss function is introduced such that its minimization can produce such a function that maps similar vectors in the input space to nearby points in the lower dimensional output space and dissimilar inputs to distant points. Let $\overrightarrow{X_1}, \overrightarrow{X_2} \in \mathcal{I}$ be a pair of input vectors shown to the system. Let Y be a binary label assigned to this pair such that $Y = 0$ if the points are similar and $Y = 1$ if the points are dissimilar. For shorthand $D_W(\overrightarrow{X_1}, \overrightarrow{X_2})$ is written as D_W. Then, the general form of the loss function is:

$$\mathcal{L}(W) = \sum_{i=1}^{P} L(W, (Y, \overrightarrow{X_1}, \overrightarrow{X_2})^i) \tag{2.1}$$

$$L(W, (Y, \overrightarrow{X_1}, \overrightarrow{X_2})^i) = (1 - Y)L_S(D_W^i) + YL_D(D_W^i) \tag{2.2}$$

where $(Y, \overrightarrow{X_1}, \overrightarrow{X_2})^i$ is the i-th labeled sample pair, L_S is the partial loss function for a pair of similar points, L_D is the partial loss function for a pair of dissimilar points, and P is the number of training pairs. L_S and L_D are defined such that minimizing L with respect to W would result in low values of D_W for similar pairs and high values of D_W for dissimilar pairs. The training algorithm as follows:

Step 1: For each input sample $\overrightarrow{X_i}$, do:

- Using prior knowledge create each $S_{\overrightarrow{X_i}}$
- Pair the sample $\overrightarrow{X_i}$ with all other training sample such that $Y_{ij} = 0$ if $\overrightarrow{X_j} \in S_{\overrightarrow{X_i}}$, and $Y_{ij} = 1$ otherwise.

Step 2: Repeat until convergence. For each pair $(\overrightarrow{X_i}, \overrightarrow{X_j})$ in the training set, do:

- If $Y_{ij} = 0$ then update W to decrease D_W. If $Y_{ij} = 1$ the update W to increase D_W.

This algorithm and the loss function define the basis of contrastive learning. In this basic approach, outside knowledge is used to create sets of similar inputs for each example. Then, the model iterates through all pairs and maximizes the agreement between similar pairs while maximizing the disagreement between dissimilar pairs. The SimCLR framework and other contrastive learning methods extend on this general approach.

2.2 The SimCLR Framework

Similar to previous approaches, the SimCLR framework learns representations by trying to maximize agreement between similar examples and maximize disagreement between dissimilar examples (Chen et al. 2020). The framework combines pieces from previous work to create a simple yet effective approach. The four major components of the SimCLR framework are the data augmentation, base encoder, projection head, and contrastive loss function.

2.2.1 Data Augmentation

In the previously mentioned contrastive learning paper (Section 2.1.1), outside knowledge was used to create sets of similar examples for each input. In the SimCLR framework, however, similar pairs are created by taking a single data example and passing it through two different image augmentation functions. This results in two correlated views of the same example that we call a positive pair. By doing this, there is no need for outside information about similarity of images as the positive pairs are created in the framework itself. Figure 2.1 provides examples of the data augmentation operators studied in the SimCLR paper.

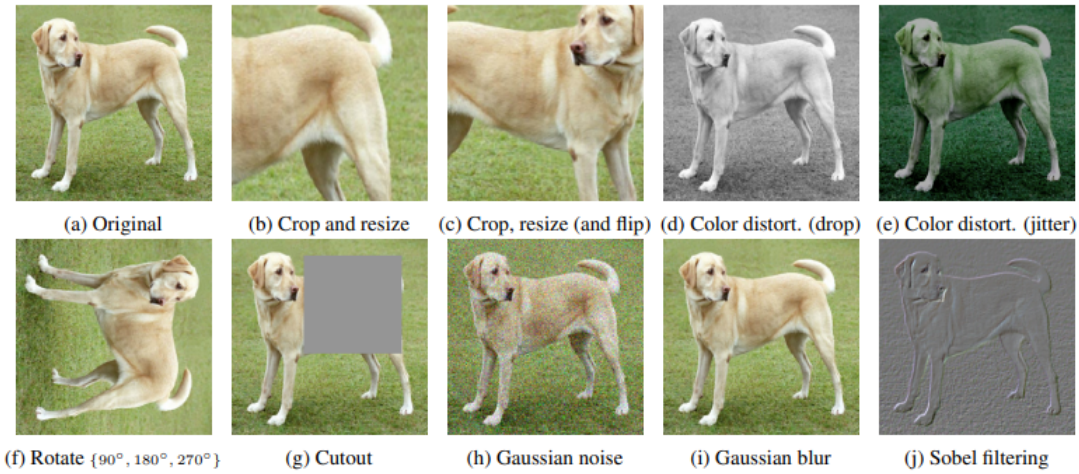

Figure 2.1: Data augmentation operators (Chen et al. 2020)

Numerous augmentations are applied to the example image sequentially to create the positive pairs. The top performing augmentations will be discussed in the results section.

2.2.2 Base Encoder

The base encoder $f(\cdot)$ is a neural network who's goal is to extract representation vectors from the examples created in the data augmentation step. There is no specific analysis done on this base encoder and the standard ResNet is used in the SimCLR paper. Therefore, given an augmented input x_i, we do $h_i = f(x_i) = ResNet(x_i)$ where $h_i \in \mathbb{R}^d$ is the output after the final layer of the ResNet for example x_i.

2.2.3 Projection Head

The projection head is a small neural network $g(\cdot)$ that maps the outputs from the base encoder to a space where the contrastive loss is actually applied. This is done with a MLP with one hidden layer. Let z_i be the output for this step from data example x_i and σ be a ReLU non-linearity. We obtain $z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$.

2.2.4 Contrastive Loss Function

The contrastive loss function is defined to be able to predict the task of finding the other positive pair when given an input. Given N examples, we use data augmentation to create N positive pairs which results in $2N$ total data points. Given a positive pair, we treat the other $2(N-1)$ examples as negatives to both of the inputs in the given positive pair. Let $sim(u,v) = \frac{u^T v}{||u||||v||}$ denote the dot product between the ℓ_2 normalized u and v. Then, the contrastive loss function for a positive pair of examples (i,j) is defined as:

$$\ell_2 = -\log \frac{\exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(sim(z_i, z_k)/\tau)} \tag{2.3}$$

where τ denotes a temperature parameter and $\mathbb{1}_{[k \neq i]} \in \{0,1\}$ is an indicator function that evaluates to 1 iff $k \neq i$.

2.2.5 SimCLR Learning Algorithm

From the paper, they give a high-level overview of the main SimCLR learning algorithm as follows. Given a batch of data $\{x_k\}_{k=1}^N$ do the following:

Algorithm 1: Main SimCLR Learning Algorithm

Data: $\{x_k\}_{k=1}^N$
for $k \in \{1, ..., N\}$ **do**
  Select two data augmentation functions $t1(\cdot), t2(\cdot)$
  $\hat{x}_{2k-1} = t1(x_k)$
  $h_{2k-1} = f(\hat{x}_{2k-1})$
  $z_{2k-1} = g(h_{2k-1})$
  $\hat{x}_{2k} = t2(x_k)$
  $h_{2k} = f(\hat{x}_{2k})$
  $z_{2k} = g(h_{2k})$
end
for $i \in \{1, ..., 2N\}$ *and* $j \in \{1, ..., 2N\}$ **do**
  $s_{i,j} = \frac{z_i^T z_j}{||z_i||||z_j||}$
end
def $\ell(i,j)$**:**
  return $-\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
Update networks f and g to minimize $\mathcal{L}$

This algorithm is run for multiple batches of data with batch size N. At the end, the encoder network $f(\cdot)$ is returned and $g(\cdot)$ is thrown away.

2.3 Paper Findings and Results

2.3.1 Best Data Augmentation

Since the data used to train the contrastive learning network is positive pairs, the data augmentation is a crucial step in learning image features. In the paper, they describe how their testing found that single transformations of images for positive pairs are not enough to learn good representations. Instead, a pair of transformations proved to be the best strategy for augmenting the data. The composition that resulted in the best performance was random cropping and random color distortion applied sequentially.

2.3.2 Base Encoder Size

In the paper, they find that increasing both width and depth of the base encoder improves performance for unsupervised contrastive learning.

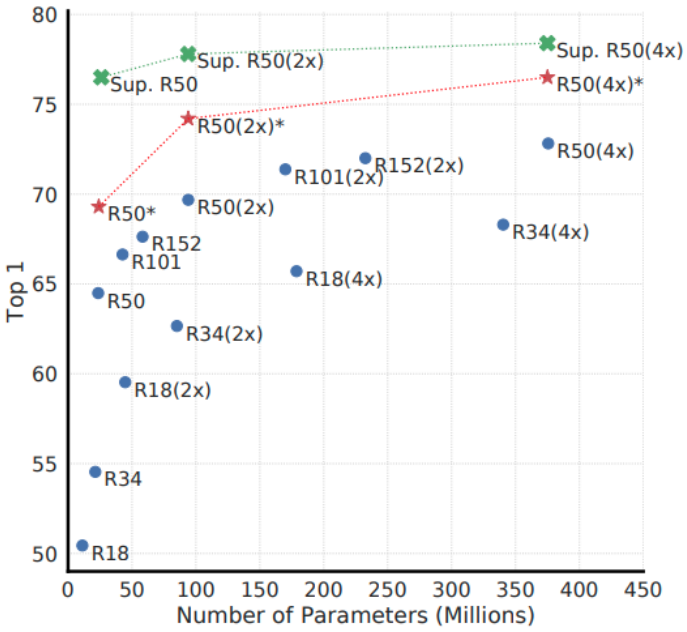

Figure 2.2: Performance for base encoder sizes (Chen et al. 2020)

Figure 2.2 highlights these findings. Models in blue dots and red dots are from the paper and were trained for 100 and 1000 epochs respectively. Models in green crosses are supervised ResNets trained for 90 epochs from He et al. 2016. The performance metric used was the top 1 accuracy on ImageNet with linear evaluation. They conclude that unsupervised learning benefits more from bigger models compared to supervised learning.

2.3.3 Loss Function and Batch Size

In the paper, they find that the NT-Xent loss or Normalized Temperature-scaled Cross Entropy Loss (Equation 2.3) was the best for model performance. Additionally, they find that both larger batch sizes and more training epochs aid in contrastive learning.

With larger batch sizes, there are more negative examples for each positive pair which helps with convergence. Training longer additionally provides more negative examples and combinations.

2.3.4 Performance Results

SimCLR performed very well compared to other models in linear evaluation. The regular ResNet-50 had a top 1 accuracy of 69.3 and a top 5 accuracy of 89.0. The ResNet-50 with a hidden layer width multiplier of 4x had a top 1 accuracy of 76.5 and a top 5 accuracy of 93.2 on ImageNet. Both of these models were trained for 1000 epochs and outperformed previous approaches.

SimCLR also performed well in semi-supervised learning. Using a random sample of 1% of ImageNet labels to fine tune the model yielded a top 5 accuracy of 85.8 and using a random sample of 10% of ImageNet labels yielded a top 5 accuracy of 92.6.

The ability of the network to do transfer learning was also reported in this paper. Transfer learning performance was tested on 12 image datasets ranging from images of things like cars, aircrafts and flowers to name a few. Two types of transfer learning were tested. First, transfer learning was tested using a linear classifier with the frozen pretrained network. Additionally, transfer learning with fine-tuning was tested in which the entire network is fine-tuned using the weights of the pretrained network as initialization. Their transfer learning results for all datasets using a SimCLR ResNet-50 models pretrained on ImageNet performed similarly to baseline supervised approaches.

2.4 Importance of Paper

This paper and its findings are relevant because it builds on a significant amount of previous research in contrastive learning to create a simple but effective framework. Additionally, the framework shows a lot of potential for other transfer learning applications. This is could be very useful for machine learning problems where there isn't a lot of labeled data or when the process of labeling data is costly.

Chapter 3

Transfer Learning Using the SimCLR Model

3.1 What is Transfer Learning

Transfer learning is the process of using a previously learned model on a new problem. This can greatly speed up training time as we can solve a new problem by simply using the weights from a pre-trained model to grab features and then attach a few layers to train for our specific task. This could provide us with the complexity of a larger model without having to train all of the weights. Transfer learning is very effective for machine learning problems where there is not a large amount of training data or where labeling and creating training data is a very expensive task.

3.2 Applying Transfer Learning to Chest Radiographs

For this experimentation, the effectiveness of transfer learning using the weights from the SimCLR network trained on ImageNet for classification of chest radiograph images was explored.

3.2.1 Kaggle COVID-19 Classification Competition

The data for this experimentation was sourced from a Kaggle competition hosted by the Society for Imaging Informatics in Medicine that took place in May of 2021(link). The goal of this competition was to create models that could identify and localize COVID-19 abnormalities on chest radiographs making it both an object detection and classification task. This task is relevant because COVID-19 has had a significant impact on the entire world and machine learning can be used to help detect COVID-19 in patients. Using this approach could lead to faster detection times which could allow patients to get earlier treatment before experiencing the more serious complications. There are currently guidelines for radiologists to assist in differentiating COVID-19 from other types of infection, however, machine learning models may be able to apply some assistance.

Label	Positive/Negative Distribution
Negative for Pneumonia	27%/73%
Typical Appearance	47%/53%
Indeterminate Appearance	17%/83%
Atypical Appearance	8%/92%

Table 3.1: Binary distribution for each label

3.2.2 About the Data

The included training dataset consists of 6,334 chest scans that were labeled by a panel of experienced radiologists. Each chest scan has the following binary attributes: "Negative for Pneumonia", "Typical Appearance", "Indeterminate Appearance", and "Atypical Appearance". Table 3.1 describes the class distribution for each of these binary variables in the dataset. Figure 3.1 provides example images for each of the four labels.

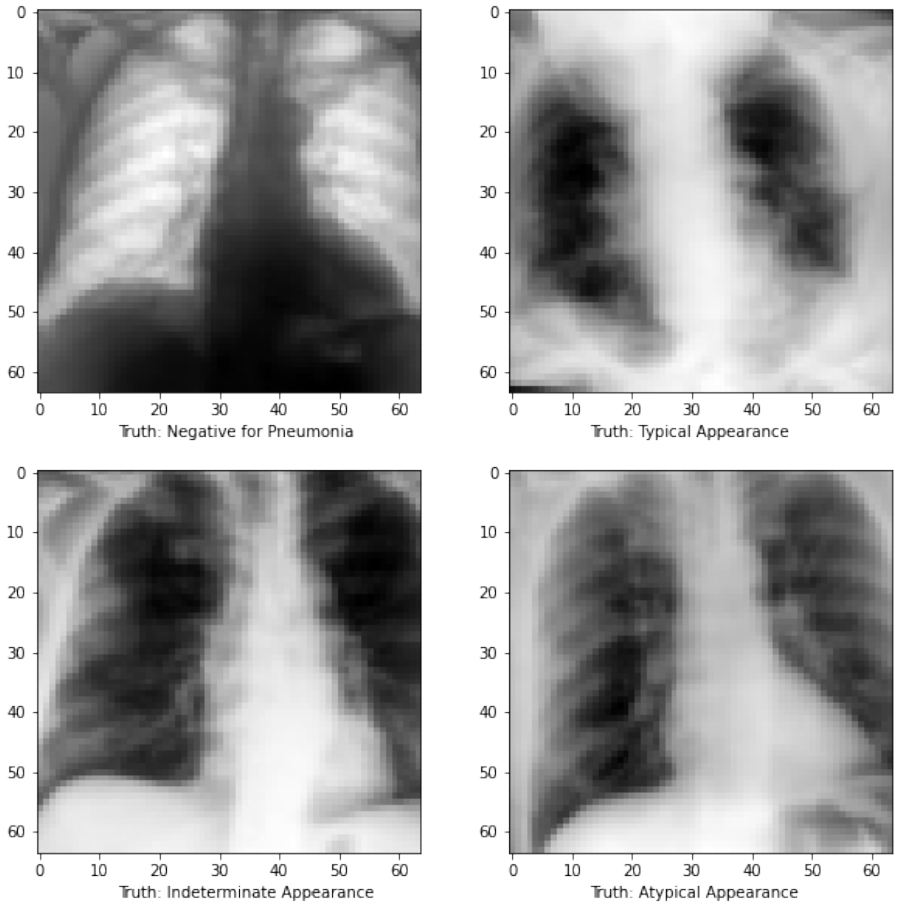

Figure 3.1: Chest radiograph example for each class

3.2.3 Data Pre-processing

First, the DICOM chest scans were processed into single-channel grayscale 64×64 images. Next, the chest images needed to be converted to three channel RGB images so that they can match the input of the pretrained SimCLR network. This was done by using the function "image.grayscale_to_rgb()" from the TensorFlow library. Next the images are reshaped to 224×224 images. This is so that the images can be passed in the network as it was trained on 224×224 images from ImageNet.

3.2.4 Experimentation Goal

The goal of this experimentation was to extend the research by applying transfer learning using the SimCLR network to a new dataset. Initial testing revealed no significant results when attempting to do multi-class classification on the above labels. Instead, we experiment with the ability to do binary classification for the presence of one label. The label chosen was "Typical Appearance" as it has the best class balance.

3.2.5 Model Structure

The input of the model has the shape $(224, 224, 3)$. This input is based on the input that was used to train the pretrained SimCLR model on ImageNet. Next, the model contains the pretrained network. For my analysis, I use a SimCLR ResNet-50 model that was fine-tuned on 100% of labels(link). This model has around 34 million parameters and was trained on ImageNet using the SimCLR framework and then fine-tuned with all labels. The model is loaded from the provided Google Cloud location using Keras. We set the weights to frozen so that they do not change during the transfer learning training. After this, a dropout layer is added. Next, there is a hidden layer with a ReLU activation function followed by another dropout layer. Finally, there is a single node output layer with sigmoid activation. The model uses a gradient descent optimizer and binary cross-entropy for the loss.

The experimentation in this project differed from the transfer learning in the original paper in a few key ways. First, this experimentation involved attaching a non-linear model to the pretrained SimCLR base encoder. In the paper, they simply add a linear layer for their transfer learning. This adjustment was made because additional attempts to use only a linear layer for transfer learning on chest radiographs resulted in very low training accuracy even after many epochs. Additionally, the transfer learning from the paper used a custom LARS optimizer. In this experimentation, however, the standard gradient descent optimizer was used.

3.3 Results and Analysis

3.3.1 Method

The code for this experimentation extends off of the interactive python notebooks that were referenced by the paper for transfer learning. First, the chest radiographs are downloaded from a server and loaded into a TensorFlow dataset object. Then, the images are converted to shape $(224, 224, 3)$ so that they can be input into the model. There

are 6,334 images in the dataset and we use 5,400 for training and 934 for testing which is about an 85/15 train-test split. The model is then trained on the training dataset for a set number of epochs and the loss and accuracy are reported. Finally, the model is evaluated on the testing dataset and the accuracy is reported. Google Colab Notebook code link: $https://colab.research.google.com/drive/1hWTDcPFlRNQAnu9x-_7Ds7VFR79tqVmz?usp=sharing$.

3.3.2 Numerical Results

This experimentation included tuning parameters like number of training epochs, hidden layer size, and the addition of dropout. To grade performance, the accuracy of the models on the validation set was compared. The best performance was with 25 epochs, 500 hidden nodes, and 20% dropout both before and after the hidden layer. The accuracy on the testing dataset for this configuration was around 73%. Figures 3.1 and 3.2 display model accuracy and model loss over the training epochs for the top performing configuration.

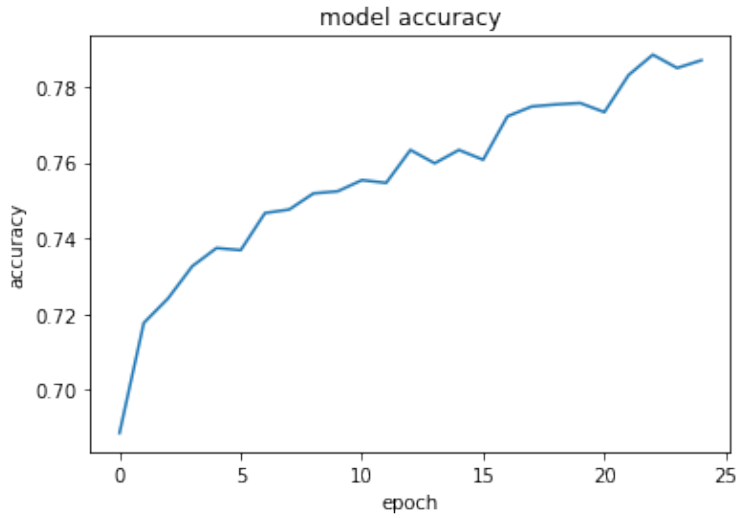

Figure 3.2: Model accuracy by epoch during training

Hidden Nodes	Dropout (Y/N)	Training Epochs	Training Accuracy	Testing Accuracy
500	N	30	98.8%	71.0%
500	Y	25	79.0%	**73.6%**
500	Y	20	78.3%	70.9%
100	N	30	97.3%	71.1%
100	Y	25	79.4%	72.5%
100	Y	20	77.6%	70.1%

Table 3.2: Experimentation results

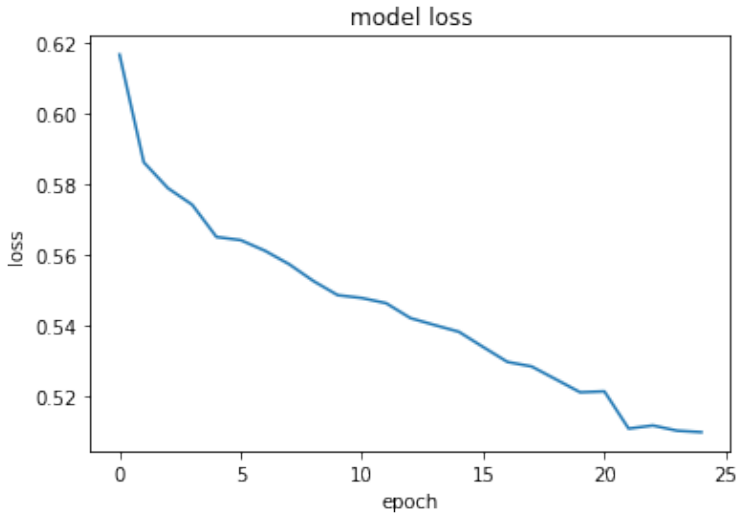

Figure 3.3: Model loss by epoch during training

Table 3.2 provides the results for the other configurations tested in this experimentation. As seen from the data, not all configurations were tested to save on runtime usage.

3.3.3 Analysis and Observations

Based on Table 3.2 we see that the best performance in my experimentation was a 73.6% accuracy on the testing dataset. This isn't the highest accuracy, however, both the training and testing datasets had about a 50-50 class distribution so the model was able to learn some representation of the data.

After initial runs, the focus was to reduce the overfitting of the model. By doing around 30 epochs in training the model was able to have close to 99% accuracy on the training data, however, the testing accuracy remained around 70%. To combat this overfitting problem, three approaches were tested: reducing model complexity by decreasing the number of hidden nodes from 500 to 100, adding dropout layers, and decreasing training epochs. Adding dropout layers and decreasing the training epochs to 25 seemed to improve performance.

The conclusion drawn from the relatively low accuracy compared to the transfer learning done in the paper is that the lung data may just be too fundamentally different for

transfer learning to be effective. The pretrained SimCLR network was trained on images from ImageNet which is comprised of images of things like animals and everyday objects. The chest images, however, were fundamentally very different and were single channel grayscale images. It is for this reason that the pretrained network may not have been able to effectively extract key features from the chest images to learn good relations.

3.4 Improvements and Future Work

One extension to this experimentation would be to include the custom LARS optimizer that they use in the original paper. Additionally, this project could be extended by allowing some training of the pretrained SimCLR network instead of freezing all of the weights during the transfer learning training.

Chapter 4

Conclusion

In this report, an overview of contrastive learning, transfer learning, and the SimCLR framework was given. The SimCLR framework outlines components including data augmentation, ResNet-50 base encoder, projection head, and contrastive loss function that lead to a simple but effective approach to learning image representations. Additionally, experimentation was conducted to explore the effectiveness of using the pretrained SimCLR network to do transfer learning on a chest radiograph classification problem. In this experimentation, it was found that attaching a nonlinear network to the pretrained base encoder resulted in a 73.6% testing accuracy on one of the binary labels indicating if the chests radiographs had "Typical Appearance".

Bibliography

Chen, Ting et al. (July 2020). "A Simple Framework for Contrastive Learning of Visual Representations". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20j.html.

Hadsell, R., S. Chopra, and Y. LeCun (2006). "Dimensionality Reduction by Learning an Invariant Mapping". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2, pp. 1735–1742. DOI: 10.1109/CVPR.2006.100.

He, Kaiming et al. (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.