



**TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS**

THESIS NO: 075MSICE006

**Disentangled representation learning for semi-supervised
classification of chest x-ray images**

by

Dipkamal Bhusal

A THESIS

**SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND
COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN
INFORMATION AND COMMUNICATION ENGINEERING**

**DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL**

August, 2021

Disentangled representation learning for semi-supervised classification
of chest x-ray images

by
Dipkamal Bhusal
075MSICE006

Thesis Supervisor
Dr. Sanjeeb Prasad Panday

A thesis submitted in partial fulfillment of the requirements for the
degree of Masters of Science in Information and Communication
Engineering

Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk Campus
Tribhuvan University
Lalitpur, Nepal

August, 2021

COPYRIGHT©

The author has agreed that the library, Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk Campus, may make this thesis freely available for inspection. Moreover the author has agreed that the permission for extensive copying of this thesis work for scholarly purpose may be granted by the professor(s), who supervised the thesis work recorded herein or, in their absence, by the Head of the Department, wherein this thesis was done. It is understood that the recognition will be given to the author of this thesis and to the Department of Electronics and Computer Engineering, Pulchowk Campus in any use of the material of this thesis. Copying of publication or other use of this thesis for financial gain without approval of the Department of Electronics and Computer Engineering, Institute of Engineering, Pulchowk Campus and author's written permission is prohibited.

Request for permission to copy or to make any use of the material in this thesis in whole or part should be addressed to:

Head

Department of Electronics and Computer Engineering

Institute of Engineering, Pulchowk Campus

Pulchowk, Lalitpur, Nepal

DECLARATION

I declare that the work hereby submitted for Master of Science in Information and Communication Engineering (MSICE) at IOE, Pulchowk Campus entitled **“Disentangled representation learning for semi-supervised classification of chest x-ray images”** is my own work and has not been previously submitted by me at any university for any academic award.

I authorize IOE, Pulchowk Campus to lend this thesis to other institution or individuals for the purpose of scholarly research.

Dipkamal Bhusal

075MSICE006

August, 2021

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled “**Disentangled representation learning for semi-supervised classification of chest x-ray images**”, submitted by **Dipkamal Bhusal** in partial fulfillment of the requirement for the award of the degree of “**Master of Science in Information and Communication Engineering**”.

.....
Supervisor: Dr. Sanjeeb Prasad Panday
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

.....
External Examiner: Om Bikram Thapa
CTO, Vianet Communications Pvt. Ltd.

.....
Committee Chairperson: Dr. Basanta Joshi
Department of Electronics and Computer Engineering,
Institute of Engineering, Tribhuvan University

Date: August, 2021

DEPARTMENTAL ACCEPTANCE

The thesis entitled “**Disentangled representation learning for semi-supervised classification of chest x-ray images**”, submitted by **Dipkamal Bhusal** in partial fulfillment of the requirement for the award of the degree of “**Master of Science in Information and Communication Engineering**” has been accepted as a bonafide record of work independently carried out by him in the department.

.....

Prof. Dr. Ram Krishna Maharjan

Head of the Department,

Department of Electronics and Computer Engineering,

Pulchowk Campus,

Institute of Engineering,

Tribhuvan University,

Nepal.

ACKNOWLEDGEMENT

I would like to pay my sincere gratitude to my supervisor Dr. Sanjeeb Prasad Panday for providing valuable suggestions and feedback during the entire journey of thesis. His valuable guidance during proposal preparation, and implementation has helped me in carrying out this work efficiently. I would also like to pay my sincere thanks to our coordinator Dr. Basanta Joshi for his cordial cooperation, assistance and guidance during the journey. I am also thankful to Prof. Dr. Shashidhar Ram Joshi, Prof. Dr. Ram Krishna Maharjan, Prof. Dr. Subarna Shakya, Dr. Surendra Shrestha, Dr. Nanda Bikram Adhikari, and all other faculty members of the Department of Electronics and Computer Engineering, Pulchowk Campus for their suggestions, feedback and motivation in the process of thesis completion.

ABSTRACT

Acquisition of medical dataset is a difficult and expensive process as it requires expertise of doctors, radiologists or other medical professionals. Chest X-rays are very common medical imaging technique, hence there are large repository of such images, but same cannot be said of other medical images. However, success of deep learning architecture primarily depends on the availability of large labeled datasets. Hence, semi-supervised learning that uses a small labeled dataset to develop a model for a large unlabeled dataset can be effective particularly in medical imaging. A drawback of common semi-supervised learning models like wrapper methods (eg. self-training) or randomization methods is that latent encoding learnt by a deep neural models is ignored and the label prediction is dependent on the partially labeled dataset obtained from a supervised learning method in the input-output space. This approach ignores the representation learning of the model, which is the goal of any machine learning algorithm, to obtain a true data distribution of the input data. The stochastic state of the hidden state can contribute significantly to the prediction since the distribution of latent variable plays a major role in approximating the data distribution. This research primarily focussed on extracting the stochastic feature of the hidden state of the input images for classification using limited labeled dataset. A total of 50000 chest x-ray images were used for first creating a baseline model of basic convolutional neural net supervised classification that obtained an overall accuracy of **80.07%**. With only 5000 labeled images, the classification accuracy reduced to **60.97%**. Then, using a variational auto-encoder based semi-supervised model considering 5000 labeled and 45000 unlabeled images, the accuracy up to **72.9%** was obtained which suggested that the use of unlabeled images could contribute to the better generalization of a machine learning model. The thesis also compared the generative ability of auto-encoder and variational auto-encoder before selecting the VAE as our representation model.

Keywords: Semi-supervised classification, Representation Learning, Deep learning, Variational Auto-Encoder, Convolutional Neural Network, Latent embedding

TABLE OF CONTENTS

COPYRIGHT	iii
DECLARATION	iv
RECOMMENDATION	v
DEPARTMENTAL ACCEPTANCE	vi
ACKNOWLEDGEMENT	vii
ABSTRACT	viii
TABLE OF CONTENTS	ix
List of Figures	xii
LIST OF TABLES	xiv
LIST OF ABBREVIATIONS	xv
1 INTRODUCTION	1
1.1 Background	1
1.2 Problem Statement	1
1.3 Supervised and semi-supervised learning	2
1.4 Representation Learning	2
1.5 Challenges	3
1.6 Objectives	3
1.7 Contribution of this Thesis	4
1.8 Outline of the Thesis	4
2 LITERATURE REVIEW	5
2.1 Representation learning	5
2.2 Semi-supervised learning	7
2.3 Semi-supervised learning in disease diagnosis	10

2.4	Research Gap	10
3	THEORETICAL BACKGROUND	12
3.1	Convolutional Neural Network	12
3.2	Gradient Descent Optimization	14
3.3	Dropout	17
3.4	Latent Space and Generalization Gap	17
3.5	Auto-Encoder	18
3.6	Variational Auto Encoder	19
3.7	Model Performance Metrics	23
3.7.1	Precision and Recall	24
3.7.2	AUC Score	24
4	METHODOLOGY	26
4.1	Tools and Resources Used	26
4.2	Dataset	26
4.3	Pre-processing	27
4.4	Proposed System	27
5	EXPERIMENT AND RESULTS	30
5.1	Baseline model	30
5.2	Auto-encoder model	34
5.3	Denoising auto-encoder model	36
5.4	Variational auto-encoder	37
5.4.1	Model I	38
5.4.2	Model II	40
5.5	Generative Model Comparison	43
5.6	Semi-supervised learning	43

6	CONCLUSION	45
6.1	Conclusion	45
6.2	Limitation	46
6.3	Future Scope	46
	REFERENCES	51

List of Figures

Figure 2.1	A block diagram of wrapper method	8
Figure 2.2	A block diagram of self ensembling mechanism	9
Figure 3.1	A block diagram of a convolutional neural network	12
Figure 3.2	Auto-encoder block diagram	18
Figure 3.3	Block diagram of training process in VAE	20
Figure 3.4	Variational Auto-encoder	22
Figure 4.1	A sample of chest x-ray from our dataset	27
Figure 4.2	Variational Auto-Encoder	28
Figure 4.3	VAE SSL model	28
Figure 5.1	Baseline CNN model	31
Figure 5.2	Loss vs Epoch of Baseline CNN model for 20000 training images	31
Figure 5.3	Loss vs Epoch of Baseline CNN model for 50000 training images	33
Figure 5.4	Auto-encoder model	34
Figure 5.5	Loss vs Epoch of Auto-encoder model of 20000 training images	35
Figure 5.6	Reconstruction by Auto-encoder model	36
Figure 5.7	Attempting to create new images from randomly sampling from latent vector	36
Figure 5.8	Input images and reconstructed image from denoising AE . .	37
Figure 5.9	Block diagram of VAE Model 1	38
Figure 5.10	Reconstruction of input by VAE Model 1	39

Figure 5.11 Loss vs Epoch of VAE Model 1	39
Figure 5.12 Block diagram of VAE Model 2	40
Figure 5.13 Reconstruction ability of VAE Model 2	40
Figure 5.14 Loss vs Epoch of VAE Model 2	41
Figure 5.15 Sampling randomly from latent vector of VAE Model 2 . . .	41
Figure 5.16 Normal distribution of first fifty elements of latent vector . .	42

LIST OF TABLES

Table 5.1	Table depicting model performance of baseline model for 20000 training images	32
Table 5.2	Table depicting performance of baseline model	33
Table 5.3	Table depicting model performance on varying size of training dataset	33
Table 5.4	Table depicting model performance of an auto-encoder model for 20000 training images	35
Table 5.5	Table depicting model performance of an auto-encoder model for 50000 images	36
Table 5.6	Table depicting model performance of a denoising auto-encoder model	37
Table 5.7	Table depicting model performance of VAE1	39
Table 5.8	Table depicting model performance of VAE2	42
Table 5.9	Table depicting overall comparison of generative model performance	43
Table 5.10	Table depicting VAE model performance on varying size of labeled dataset	44
Table 5.11	Comparison of model vs size of labeled dataset vs accuracy . .	44

LIST OF ABBREVIATIONS

AUC	: Area Under Curve
ANN	: Artificial Neural Network
AE	: Auto-Encoder
CNN	: Convolutional Neural Network
FN	: False Negative
FP	: False Positive
SSL	: Semi-supervised learning
SGD	: Stochastic Gradient Descent
TN	: True Negative
TP	: True Positive
VAE	: Variational Auto Encoder

CHAPTER 1

INTRODUCTION

1.1 Background

Semi-supervised learning (SSL) approach uses a small amount of labeled data with a large amount of unlabeled data during training for making predictions on unseen data. The goal of any semi-supervised algorithm is to improve the generalization ability of a machine learning model using the readily available unlabeled datasets. Most of such approaches ignore the latent encoding learnt by the neural networks and the prediction of the model is entirely dependent on the partially labeled dataset. However, recent advancement in representation learning has shown that the stochastic state of the hidden state of any neural network contributes significantly to the prediction since the distribution of latent variable plays a major role in approximating the data distribution. Such stochastic embedding of the latent variables can improve the SSL ensemble prediction by providing an improved data distribution based on separated latent space.

1.2 Problem Statement

Deep learning has achieved tremendous achievements in accuracy in various image analysis tasks thanks to the availability of large repository of labeled images. Since labeling images is a costly process, this becomes a hurdle to further research in medical imaging disease diagnosis. Semi-supervised learning thus can provide a promising solution by utilizing the massive unlabeled data to improve the learning.

The analytical learning theory for machine learning proposed in [1] rationalizes that the latent or hidden space of a deep learning model contributes more to the generalization ability. A model insensitive to the changes in latent space has greater discriminative power. Accepting this fact, the goal of this thesis work is to devise a SSL approach that is based on disentangled representation learning of the stochastic

latent space i.e. develop an semi-supervised model that comprises of a generative model based representation learning followed by a network for semi-supervised classification and evaluate the model for thoracic diseases classification.

1.3 Supervised and semi-supervised learning

Machine learning approaches are broadly divided into supervised and unsupervised learning. While in former, there is the availability of labeled dataset (x,y) , and the goal is to build a classifier or regressor that can rightly predict the output for unseen input x , in unsupervised learning, there are no annotated labels for inputs and the task of the classifier is to find the underlying data distribution and data structure for the specific tasks assigned. In semi-supervised learning, both the idea of supervised and unsupervised are combined to use a small set of the labeled datasets and an unlabeled dataset to perform improved learning. Semi-supervised methods are very important when there is a scarcity of labeled datasets like disease diagnosis. If there is abundant unlabelled data and certain assumptions about the distribution of the data can be made, the unlabelled data can assist in the construction of a better classifier. More than often, the whole labeling process of datasets is costly and flawed hence a reliably performing semi-supervised solutions will have a huge impact. However, we must be aware that unlabeled dataset may not always improve the classifier or worse, degrade it. Hence, careful consideration must be taken.

1.4 Representation Learning

A discriminative model, for example a simple neural network classifier, is able to obtain important features from the data. Such feature extractions are used by the model for classification. This kind of model can be represented from probabilistic perspective as estimating the $p(y|x)$, where y is the output class and x is the input data point. For example, the probability of an image belonging to the class cat or a dog. However, with generative model which is a form of representation learning, the goal is to understand the underlying distribution of the data for underst explains

how the data was generated, mimicking the hidden distribution that provides the ability to sample from the distribution for generating new data. Such models can be defined as estimating the probability $p(x)$, where x is the data point. It estimates the probability of observing the data point x in the distribution. A representation model of the data hence captures the most important attribute of the data distribution that differ from each other, hence contributing to understanding the real distribution of data.

1.5 Challenges

There is a large inter-subject variations in medical datasets as each individuals medical images like x-rays pertaining to the same disease can vary vastly. Hence, training with such inter-subject variations can fail to develop a model that generalizes well to a different set of test population affecting the overall model performance. Hence, the generalization based on population can fail to apply to a diverse group of people. Representation learning can aid to this issue by incorporating the variation and learning the hidden representation of the data however, the large inter-subject variations can still fail to improve the overall generalization ability and the model could be biased to a certain group of training set.

1.6 Objectives

The objective of this research is to investigate various representation learning method to understand the generative capacity of each method for latent embedding extraction that can used for self-supervised and semi-supervised classification of chest x-ray images. Main objectives are given below.

- To develop a semi-supervised approach for medical imaging classification using disentangled representation of the latent space
- To compare the generalization ability of the proposed SSL model with baseline models

1.7 Contribution of this Thesis

The main contribution of this thesis to the medical AI can be seen in the implementation of a generative model like variational auto-encoder in making semi-supervised classification with limited labeled datasets. A more detailed list of the various contributions is provided below,

- Use of deep neural network to analyse the disease diagnosis problems
- Investigation of generative model for representation learning and self-supervised classification of chest x-ray images
- Semi-supervised learning using limited labeled chest x-ray and large size of unlabeled x-rays

1.8 Outline of the Thesis

The remaining part of the document is organized as follows,

Chapter 2 describes the state of the art of the representation learning, and semi-supervised learning focusing on medical disease diagnosis.

Chapter 3 describes the theoretical background of this thesis work.

Chapter 4 describes the methodology used in the research focusing on dataset, tools and resources, preprocessing and system architecture.

Chapter 5 describes the implementation and analysis of the research.

Chapter 6 contains the summary and future scope of the thesis.

CHAPTER 2

LITERATURE REVIEW

2.1 Representation learning

The main objective of a machine learning task is to obtain the true data representation for performing a task, hence the question is to understand what kind of representation is better than the other. If a certain data representation makes the learning task easier, we consider that data represents a good one. For example: in a supervised classification task using a linear classifier at the last layer, the hidden layers of the network have to contribute to forming the linear classifier. When we train with this criterion of supervised learning, each representation in every hidden layer has to take on features that make the classification more accurate and linear. However, supervised learning does not speak about learned intermediate features. Unsupervised and semi-supervised however try to explicitly impose some conditions on the hidden features to design them in some manner to improve the learning.

Representation learning is particularly useful when we have a large size of unlabeled training data and relatively little labeled training data. Training a supervised model on a small dataset can effectively result in overfitting. Semi-supervised learning can prevent this overfitting by learning from the both labeled and unlabeled data. We can use the unlabeled data in some manner for an appropriate data representation and use such representation for any task. So, how can one find a better representation of the data?

A common hypothesis accepted widely is to find the disentangled representation that corresponds to the underlying causes of the observed dataset. According to this hypothesis, we first find a representation for $p(x)$ where x is the input dataset which can be helpful in computing $p(y|x)$. To find a separable representation learning, we need to acquire the underlying hidden representation of what we observe i.e. if a representation h represents the hidden attribute of the input x ,

then we first obtain the $p(h)$ which will be helpful in predicting y from h .

Assume that y is an output case and h represents all the hidden factors. A true representation model can then be represented as,

$$p(h, x) = p(x|h)p(h) \quad (2.1)$$

So, a better generalization of a deep learning model can be obtained by learning a generative model that recovers the factors h and $p(x|h)$. Study of Berkhahn et al [2] showed that generative model can provide extra information to the classifier for improving the classification accuracy. Another study performed by Bengio et al [3] supported the study of Berkhahn et al. They argued that entanglement of different data representation can hide the crucial explanatory factors of a data. Hence, a generic and powerful representation learning methods that could disentangle such data variations can improve the chances of success of machine learning algorithms.

Generative Adversarial Networks [4] have been widely applied to recognize highly salient features in a feed-forward classifier and are actively used in exploring representation methods with the main aim of finding the salient hidden feature of the input that are crucial in predicting output. In [5], the authors demonstrated that use of GAN in learning the hierarchy of representation from images were helpful in improving the overall performance of unsupervised learning. Addressing the shortcomings of generator-discriminator model of GANs, in [6] authors added an encoder and modifying the discriminator to improve the generative capability of the GAN model. Their generation-based models achieved the state of the art in unsupervised representation learning on ImageNet. In [7], encoder-decoder element of GAN was used to develop a disentangled representation of pose discrepancy in images for accurate face recognition. Works done in [8] addresses the problem of separating the latent variables of style and content in various language models. Zhang and team proposed a convolutional and convolutional auto-encoder framework for learning latent representation of long text sequences. They argued that the use of representation learning in decoding and reconstructing long text paragraph using auto-encoder had improved the accuracy and recommended the use of such methods in natural language processing [9].

However, there are few limitations of use of Generative Adversarial Network in representation learning. Such models have high chances of non-convergence especially when model parameters oscillate and destabilize. GAN produces sharp images as generative models but the generator can collapse producing only limited variation of images. Thus they do not yet represent the true data distribution. There are also high chances of overfitting of data due to imbalance between generator and discriminator model.

There has been an increment in application of Variational Auto-Encoder in various data domain since they have been regarded as one of the better generative models that is able to generate a wide variety of data, learning from the input. VAE [10] helps to realize the posterior inference of latent variables that can be used as representation learning in deep neural networks. Works carried out in [11] showed the use of encoder-decoder approach of VAE and auto-encoder to extract the latent representation of speech signal for capturing high level semantic content from the speech. In [12], VAE was used to disentangle the latent space of text semantics and use the attributes to generate high quality text. The use of variational auto-encoder has been most widely explored in image and video [13] [14].

2.2 Semi-supervised learning

The common assumption of semi-supervised learning is that if two samples are close in the input space then both of them might have same labels. This assumption known as smoothness assumption is one of the widely used principle in developing semi-supervised learning approach. Based on this approach, one of the most widely used semi-supervised learning approach is self-training wrapper method, where labels from a model are propagated and reused for re-training.

In self-training method, the discriminatory classifier is first trained with the labeled examples. The classifier is also asked to classify the unlabeled data. The most probable labels of untrained data are used as labeled data in next cycle and added to the training set. The classifier is re-trained and the process repeated. This method has been widely used in natural language processing tasks [15] [16] [17]. Rosenberg applied this method in object detection system proving that semi-supervised

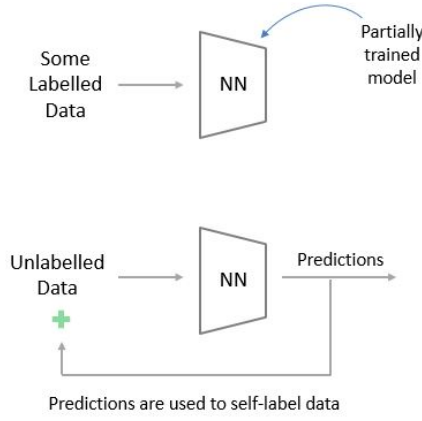


Figure 2.1: A block diagram of wrapper method

learning could perform as better as the supervised approaches [18]. However, these methods are prone to error due to poor predictions. Transductive SVMs [19] implemented Support Vector Machines(SVM) but with some modification to ensure few unlabeled observations near the margin. The authors tried to minimize the misclassification for unlabeled observations. But this approach seemed to be difficult to extend to a larger set of unlabeled images. Some graph based methods proposed by aimed to propagate the label between the similar nodes from labeled to unlabeled by finding some minimum criteria. However, such graph based methods were too dependent on the overall structure of the data-points and require complicated analysis.

One of the new approaches in SSL is to introduce randomization and augmentation in input space to improve the generalization ability of the deep learning model. The result was that the model was less sensitive to the latent space. Recent solutions like self-ensembling method that uses consistency-based regularization are one of the simplest and most efficient methods for training a deep neural network with only a small labeled training data [20]. The main idea behind self-ensembling is to form a consensus of prediction of unknown labels using the network in training outputs on different epochs i.e. to use an ensemble of the previous outputs of a neural network as an unsupervised target. The method uses different regularization and input augmentation for making an ensemble prediction. The authors of the proposed methodology were able to obtain a new record for two standard SSL benchmarks.

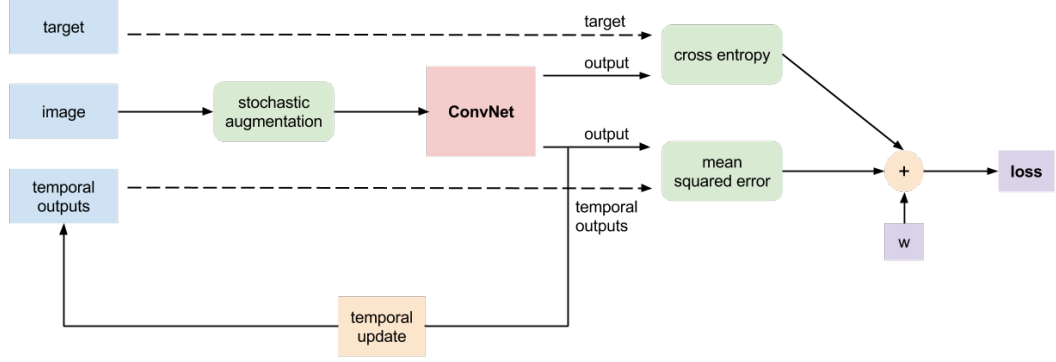


Figure 2.2: A block diagram of self ensembling mechanism

An ensemble of multiple neural networks generally provides better and stable performance than a single one as it has been observed in classical machine learning with bagging [21]. In neural networks, such an ensemble is exploited using dropout or other regularization methods. Self-training is one of the oldest type of SSL method where new labels are predicted from partially labeled data. It was proposed in 1995 for linguistic analysis [22]. Whitney and Sarkar developed a graph based approach in 2012 [23] while label propagation method was proposed in 2002 by Zhu and Ghahramani that inferred the labels of unlabeled training data comparing the labeled data using some kind of distance metric [24].

Neural network-based approaches generally train a feed-forward classifier with labeled data but introducing some form of penalty from unsupervised data embedding [25]. Generative models have also been used extensively for semi-supervised learning purposes. These models assume the problem of learning as a imputation task aiming to learn the underlying data distribution. One of the oldest approach assumes a model $p(x|y) = p(y) * p(x|y)$ taking in Gaussian approximation of the model. Using large amount of unlabeled data, the components of the original data distribution is identified. This algorithm was used for text classification by Nigam et al [26] and for face recognition by Baluja [27]. Other models using generative approaches such as Gaussian or Markov models based were not that accurate due to need of large states to store.

2.3 Semi-supervised learning in disease diagnosis

Recently, there has been an increment in implementation of semi-supervised approaches to medical imaging. A method, called FocalMix, was proposed in 2020 that developed novel semi-supervised learning for 3D medical image detection [28]. The authors proposed a generalization of the focal loss that to use soft target training labels with skewed distributions by proposing a target prediction strategy using anchor-level ensembles of augmented image patches by rotation and flipping. In [7], binary classification of x-ray images was performed using an SSL methodology using GAN approach. In [29], the authors used an unsupervised disentangled representation for myocardial segmentation. In 2020, researchers at RIT proposed a global latent mixing method where a neural network was trained using a mixing of of labeled and unlabeled data, at both the input and latent space for better regularization [30]. In [31], transfer learning, multi-task learning and semi-supervised learning were unified into one framework to extract extra performance for endoscopic images. A research work based on dimensionality reduction method was proposed in 2011 that transformed high-dimensional input of 3D medical images to a low-dimensional representation, and extended it to the semi-supervised learning setting [32]. The proposed algorithm for semi-supervised learning was evaluated with benchmarks and the results were comparable to the state-of-the-art.

2.4 Research Gap

It is tempting to apply advances in semi-supervised learning to medical imaging analysis. However, most works at the present are still concerned with disease diagnosis tasks and objective of learning classification approaches are not explored to the full potential. As mentioned in section 3.1, some implementations have been performed to show the added benefit of a SSL approach in medical imaging. But most research is focused on making an ensemble prediction with labeled and unlabeled dataset or modifying the loss function or applying different data augmentation for stability. Independent works in representation learning has been

however explored widely especially in image recognition tasks. But exploring the stochastic latent space for medical image classification is a very new research topic in this field. In [33], a disentangled representation learning of input images was obtained using a variational auto-encoder which was subsequently used in another VAE based semi-supervised model. In this work, we use a VAE based representation learning to extract salient features of medical images from the stochastic latent space and use such disentangled features in a semi-supervised network.

CHAPTER 3

THEORETICAL BACKGROUND

3.1 Convolutional Neural Network

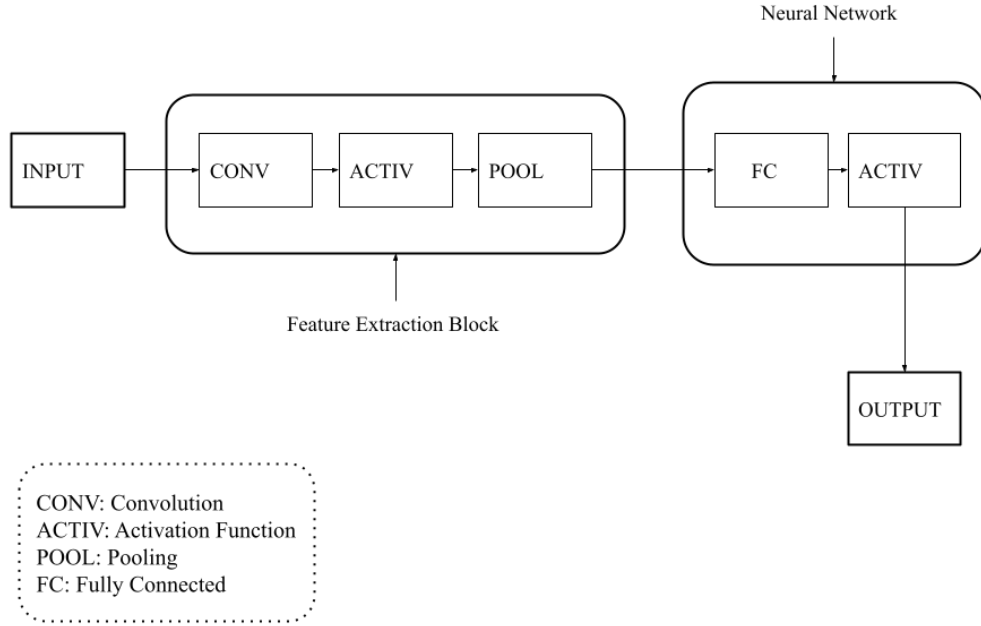


Figure 3.1: A block diagram of a cnn

For processing images, CNNs are highly optimized and are efficient in learning and extracting characteristics and abstractions. They help train the neural network for the optimum minimization of error, producing highly optimized weights. CNN was first proposed by Fukushima in 1988 for visual pattern recognition. He proposed a hierarchical network named neocognitron consisting of many layers of neurons with a variable connection between each layer with the ability to recognize patterns by learning. The lower stage layers of the neocognitron network extracted input images' local features, while the higher stage layers gradually integrated into identifying more specific and global features of the images. However, because of limits in computation resources, this network did not receive enough exposure.

Only, in the 1990s, when LeCun et al. applied a gradient descent based algorithm to CNN and obtained successful results for the handwritten digits recognition task, researchers started working on this architecture.

Basically, a CNN is a network of convolution and pooling layers applied one after another for a number of depths to extract essential features from the input data. Convolution is the sum of element-wise product between a tensor and a kernel. There are two important operations before convolving an input tensor and a kernel, padding and stride. We use padding in an input image so that the kernel takes into account the corner pixels in an image or boundaries data in other inputs. Stride is the step we take over an input tensor in a convolutional product. The larger the stride, the more shrinking the size of input. Let (nH, nW, nC) be the dimension of the input image and (f, f, nK) be the kernel dimension where f is generally an odd dimension. Let s and p represents the size of stride and padding then, then the dimension of the output after convolution operation is

$$output = [(nH + 2p - f)/s + 1, (nW + 2p - f)/s + 1, nK] \quad (3.1)$$

The filters in the convolutional layer have weights and biases to learn that are updated during the back-propagation phase. After the convolutional product, we apply an activation function. The choice of activation depends on the requirement of the problem but most commonly ReLU activation function is used in the hidden layers these days. It is computationally efficient compared to sigmoid or tanh functions. A ReLU activation can be represented mathematically as,

$$f(z) = \max(0, z) \quad (3.2)$$

After applying convolution and activation, we down-sample the feature size with a pooling operation. Pooling slides a filter with no learnable parameters over a tensor and computes either the max or average of the values over the window of the filter. This operation only affects nH and nW . The dimension of an image after pooling with a fxf filter is given by,

$$output = [(nH + 2p - f)/s + 1, (nW + 2p - f)/s + 1, nK] \quad (3.3)$$

where, p and s are padding and stride values for the pooling operation. We apply convolution and pooling a number of times depending on the requirement of the feature extraction and then connect the output to a fully connected network which is essentially a feed-forward neural network with weights and biases. These are updated during the back propagation phase.

3.2 Gradient Descent Optimization

Gradient descent is used to optimize the neural network loss function $J(\theta)$ which is parameterized by a neural model's parameters $\theta \in R^d$. To minimize the objective function, the parameters are updated in the opposite direction of the gradient of the loss function $\nabla_{\theta}J(\theta)$ w.r.t. to the parameters. The learning rate η determines the size of the steps taken to reach the minimum value. Batch gradient descent performs the optimization by computing the gradient of the loss function with respect to the parameters θ for the whole training dataset:

$$\theta = \theta - \eta \cdot \nabla_{\theta}J(\theta) \quad (3.4)$$

Batch gradients only calculate the gradients for the complete training dataset for a single optimization of parameters, hence, this method performs slowly and consumes a lot of memory.

Stochastic gradient descent (SGD) however updates the neural network parameters for each training example $x^{(i)}$ and label $y^{(i)}$ given by the equation:

$$\theta = \theta - \eta \cdot \nabla_{\theta}J(\theta; x^{(i)}; y^{(i)}) \quad (3.5)$$

Since SGD performs parameters update with every training example, the objective function heavily fluctuates. Another optimization method, called Mini-batch gradient descent, uses the both approaches by performing the parameter update only for a batch of n training examples, providing us a stable convergence of the

objective function:

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; x^{(i:i+n)}; y^{(i:i+n)}) \quad (3.6)$$

Still, traditional mini-batch gradient descent has several limitations. Firstly, the selection of the best learning rate is a difficult decision. Too small learning rate means slow convergence and too large learning rate can prohibit the best convergence to minimum thus the loss function can either fluctuate or diverge away. Use of learning rate scheduler can solve this problem but this needs to be pre-defined hence they do not take into account the changes to the objective function occurred between the epochs of training. They are unadaptive solution. Also, in traditional mini-batch gradient descent, the same learning rate is applied to all the parameters of the neural model. However, a sparse data with multiple parameters can benefit more if individual learning rate is applied to each of the parameter. If the loss function is highly non-convex, the neural network can get trapped in sub optimal local minima with the mini-batch gradient descent.

Momentum approach can help accelerate SGD in the relevant direction by damping the oscillations of the objective function. To achieve this, it adds a fraction γ of the update vector of the past time step to the current update vector:

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_{\theta} J(\theta) \\ \theta &= \theta - v_t \end{aligned} \quad (3.7)$$

Adagrad is another optimization algorithm for gradient-based optimization that adapts the learning rate to the parameters. It performs smaller updates for parameters associated with frequently occurring features, and larger updates for parameters associated with infrequent features. In the earlier method, for all parameters of the model, same update was performed i.e. every parameter θ_i used the same learning rate η . Adagrad uses a different learning rate for every parameter θ_i at every time step t . Let g_t denotes the gradient at time step t . Then, the partial derivative of the loss function w.r.t. to the parameter θ_i at time step t is $g_{t,i}$:

$$g_{t,i} = \nabla_{\theta} J(\theta_{t,i}) \quad (3.8)$$

Then the parameters update for every parameter θ_i is:

$$\theta_{t+1,i} = \theta_{t,i} - \eta \cdot g_{t,i} \quad (3.9)$$

The general learning rate is modified by Adagrad at each time step for every parameter based on the past gradients that have been computed for:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot g_{t,i} \quad (3.10)$$

where, $G_t \in R^{d \times d}$ is a diagonal matrix where each diagonal element $G_{t,ii}$ is the sum of the squares of the gradients w.r.t. θ_i . RMSprop resolves the radically diminishing learning rates of Adagrad optimization.

$$\begin{aligned} E[g^2]_t &= 0.9E[g^2]_{t-1} + 0.1g_t^2 \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \end{aligned} \quad (3.11)$$

Adaptive Moment Estimation (Adam) is one of the modern methods that also computes adaptive learning rates for each parameter combining the feature of momentum and RMSprop by storing both exponentially decaying average of past squared gradients and exponentially decaying average of past gradients. The decaying averages of past and past squared gradients m_t and v_t can be computed respectively as follows:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \end{aligned} \quad (3.12)$$

where, m_t and v_t are the mean and variance of the gradients respectively. Then, to update the parameters,

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \quad (3.13)$$

For fast convergence and training a deep neural network, one of the adaptive learning rate optimizer is preferred. Adam is used most commonly.

3.3 Dropout

Dropout is a technique used as a regularization to prevent overfitting. In a dropout, randomly selected neurons are ignored during training. Contribution to activate in a neurons removed temporarily in the forward pass and no update in weight is performed in the backward pass as well. Dropout is not used after training when making a prediction. Normally, a dropout of 0.2 to 0.5 is used which means 20%-50% of neurons is cut off.

3.4 Latent Space and Generalization Gap

Analytical learning theory for machine learning [1] provides insights on the mathematical bounds before any actual learning is done. This insight helps to design better algorithms in machine learning.

Let us assume a dataset, $S_M = S^1, S^2, S^3, \dots, S^m$ is used to develop a classifier via a learning algorithm Λ such that the obtained model is $y_\Lambda(S_M)$. The goal is then to minimize the expected error $E_\mu[Ly_\Lambda(S_M)] = E_z[Ly_\Lambda(S_M)(z)]$ with respect to a true unknown measure μ , where Ly_Λ combines loss function with the model y_Λ . For example: in a supervised classification problem, $y_\Lambda(z) = L(y_\Lambda(x), y)$ considering x, y is input and its label. The expected error mentioned earlier can be approximated by the expression $(1/m) * \sum_{i:m} L(y_\Lambda(s_m)(Z^i))$ where Z^i is the dataset.

Hence, the generalization gap can be approximated by the following expectation expression:

$$\Delta = E_\mu[Ly_\Lambda(S_M)] - E_{S_m}[Ly_\Lambda(S_M)] \quad (3.14)$$

The upper bound on the generalization error is provided by theorem 1 in [5]

$$\Delta_g \leq V[f].D^* \quad (3.15)$$

where $V[f]$ measures the variation between the evaluation of function f in each perturbation of variables. D known as star-discrepancy measures discrepancy

between the latent projections of an available data set D and true data distribution.

Let us consider that z_y is the latent variable related to a data distribution y and z_o is the latent variable unrelated to y . Then, according to the analytical theory [1], the variation evaluation $V[f]$ will be least when the latent space representations z are actually close to the representation of the true data distribution, which is indeed given by the representation learning. Hence, the argument that the latent space is able to provide a better representation of data and hence is able to reduce the generalization error when evaluating the test dataset holds true.

3.5 Auto-Encoder

Auto-encoder consists of an encoder and a decoder. The encoder 'f' maps the original space X to a latent space Z while the decoder 'g' maps the latent space back to original input space. Hence, the characteristic of a good auto-encoder can be represented with the following loss expression:

$$E_{x \in p(x)} [|X - g \circ f(x)|^2] \approx 0 \quad (3.16)$$

If we are given two mappings f and g with parameters θ_f and θ_g , then training the auto-encoder means minimizing an empirical estimate of the loss expressed in equation [4.3].

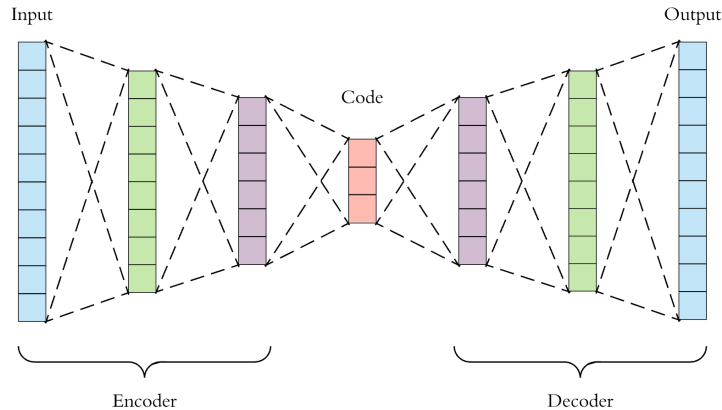


Figure 3.2: Auto-encoder block diagram

A linear auto-encoder expressed in terms of linear expression of encoder and decoder is essentially a PCA. But, if we express encoder and decoder in terms of neural

networks, better results are obtained. Hence, encoder and decoder is represented by multi-layer perceptrons.

After introducing a density model over the latent space, Z , we can sample the mean and standard deviation of the Gaussian distribution of the latent variables. Mapping such samples to the input space X with a decoder gives the representation ability of the auto-encoder. Auto-encoder produces extremely unsatisfactory results in this regard. They convert the input data into an encoding vector where each dimension of data that represents some learned attribute outputs a single value for each dimension. The decoder network takes such discrete single value for recreating the original input which it does quite effectively. However, auto-encoders can only recreate the original input samples. They are unable to learn the data generating distribution itself. So, if we give random samples from the latent distribution, the model fails to recreate the input image. Hence, auto-encoders are suitable for replicating images not for generative models.

3.6 Variational Auto Encoder

A variational auto-encoder (VAE) is a generative model that provides a probabilistic approach for describing an observation in hidden space. Thus, using VAE, we obtain an encoder that describes a probability distribution for each latent feature, instead of an encoder which outputs just a single value to describe each latent state attribute. Because of this, the latent spaces of VAE are continuous, allowing random sampling. This helps in generating new unseen data. Unlike the autoencoder, VAE understands the distribution of data in the latent space, z .

Variational Auto Encoder is used to disentangle and obtain the representation learning of the data distribution from the latent space. It consists of an encoder and decoder, both of which are, in neural network perspective, neural nets or multilayer perceptron. The encoder, in true sense, is a probabilistic encoder $q_\phi(z|x)$, that when given some model parameters ϕ and input to the encoder datapoint x , models the probability of obtaining the latent vector, z . Input data x is assumed to be generated by a distribution with compared to the hidden representation. The network has weights and biases ϕ . The representation z is lower dimensional

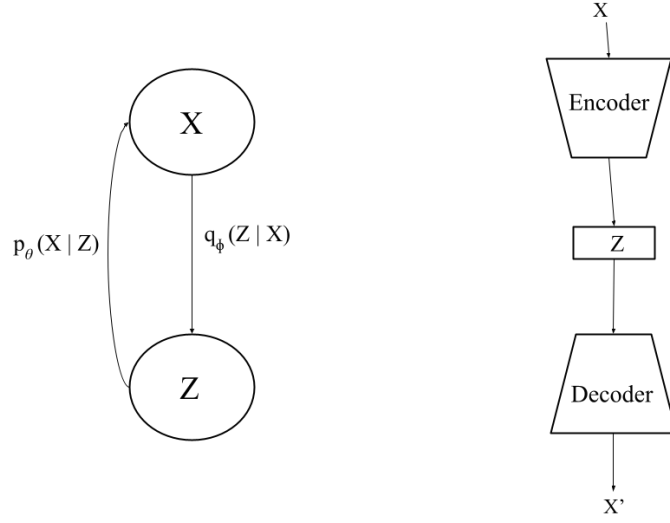


Figure 3.3: Block diagram of training process in VAE

compared to the input data. Its dimension is a hyperparameter. The encoder then has to learn an efficient representation of the data so that its encoding can measure the true nature of data and produce the same x using the z . Assume that the encoder is $q_\phi(z|x)$ and is a gaussian probability density. Hence, the encoder outputs parameter of the gaussian density which are mean and variance. We can sample from this distribution and obtain representation of z . The decoder is also a neural network or in true sense, a probabilistic decoder, $p_\theta(x|z)$, which when given some model parameters θ and the input of the hidden representation z , then it gives the probability distribution of obtaining the data point x given the latent vector z . This network also has weights and biases. The decoder is represented as $p_\theta(x|z)$.

If we assume our input image in a 28×28 image of an x ray, the encoder encodes the 784 dimensional data into a latent representation space z which will be less than 784. If the images only have black or white color then each pixel can be represented by 0 or 1 and hence, its probability distribution can be presented by a Bernoulli. The decoder receives input z and outputs 784 Bernoulli parameters each for 784 pixels and decodes the real valued numbers.

Thus, the goal of VAE is to learn the distribution of the latent vector z , $q_\phi(z|x)$. Using this data distribution, the latent vector z can be sampled that can fed into

the decoder network $p_\theta(x|z)$ to create a new data point x .

Let us assume that the input data point is represented by x while z is the latent vector obtained from the encoding process. Consider $p(z, x)$ as the joint probability distribution between the input data point and the latent vector. Then, the problem of encoding can be represented mathematically by the expression:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} \quad (3.17)$$

$p(x)$ is given by,

$$p(x) = \int_z p(x|z)p(z)dz \quad (3.18)$$

Expression [3.18] is intractable since its integral is not available in closed form as there are multiple integrals over all the latent vector z . This problem is solved by variational inference method that poses the inference problem in terms of an optimization problem. To do this, the problem $p(z|x)$ is modeled using $q(z|x)$ that has a simple and well-known distribution like Gaussian. Then, we can use KL-divergence to find the approximation difference between two distribution using the expression,

$$p(x) = \int_z p(x|z)p(z)dz \quad (3.19)$$

$$\begin{aligned} KL(q_\phi(z|x)||p_\theta(z|x)) &= \sum_z q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z|x)} \\ &= E_{z \in q_\phi(z|x)} \log \frac{q_\phi(z|x)}{p_\theta(z|x)} \\ &= E_{z \in q_\phi(z|x)} [\log(q_\phi(z|x)) - \log(p_\theta(z|x))] \end{aligned} \quad (3.20)$$

Using equations [3.17] and [3.20],

$$\begin{aligned} KL(q_\phi(z|x)||p_\theta(z|x)) &= E_{z'} [\log(q_\phi(z|x)) - \log \frac{(p_\theta(x|z))p_\theta(z)}{p_\theta(x)}] \\ &= E_{z'} [\log(q_\phi(z|x)) - \log(p_\theta(x|z)) - \log p_\theta(z) + p_\theta(x)] \end{aligned} \quad (3.21)$$

Rearranging the expression, we obtain,

$$\begin{aligned}
KL(q_\phi(z|x)||p_\theta(z|x)) - \log p_\theta(x) &= E_{z'}[\log(q_\phi(z|x)) - \log(p_\theta(x|z)) - \log p_\theta(z)] \\
&= E_{z'}[\log(p_\theta(x|z))] - E_{z'}[\log(q_\phi(z|x)) - \log p_\theta(z)] \\
&= E_{z'}[\log(p_\theta(x|z))] - KL[(q_\phi(z|x)||p_\theta(z)]
\end{aligned} \tag{3.22}$$

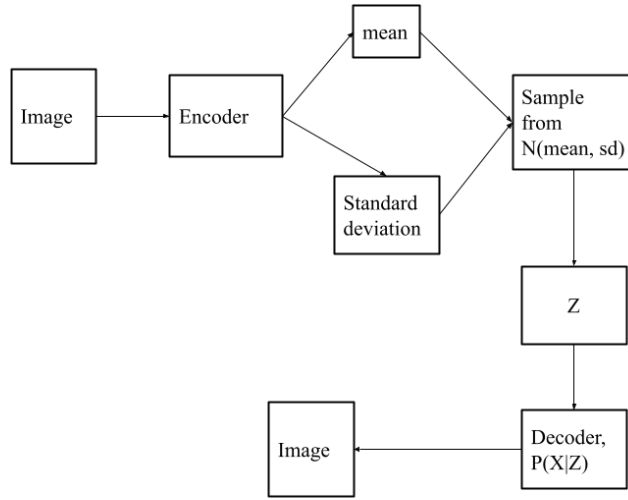


Figure 3.4: Variational Auto-encoder

The first term in the equation [3.22] represents the likelihood of reconstruction and the second term is the KL divergence term that ensures that the distribution learned by the encoder is similar to the prior distribution of the latent vector. This is the objective function of variational auto-encoder. There will certainly be loss in encoding and subsequent decoding since all the information cannot be transmitted. This information loss is measured with reconstruction log-likelihood.

The loss function of the variational auto-encoder can be obtained as,

$$\begin{aligned}
Loss, l_i(\theta, \phi) &= -ObjectiveFunction \\
&= -E_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] + KL(q_\phi(z|x)||p(z))
\end{aligned} \tag{3.23}$$

Our target is hence to obtain optimal parameters θ and ϕ such that

$$\theta^*, \phi^* = \operatorname{argmin}(\theta, \phi) L(\theta, \phi) \quad (3.24)$$

where θ is the parameter of the decoder network and ϕ is the parameter of the encoder network.

In equation [4.9], the first term is the reconstruction loss, also called log-likelihood, that encourages the decoder to learn the right representation of the data for accurate reconstruction. The other term is a regularizer called Kullback-Leiber divergence that measures the divergence between encoder distribution $q_\phi(z|x)$ and $p(z)$, measuring the loss of information when q is used to represent p . We consider standard normal distribution for p . The training of VAE is performed using gradient descent for loss optimization with respect to the encoder-decoder parameters.

The variational auto-encoder is thus trained using a set of input images that helps to learn the mean and standard deviation of the latent space. This Gaussian density of the latent variables provides us the data generating distribution. For reproducing an input image or similar one, we sample from one of the centroid within the latent space and pass through the decoder network to obtain the image in the input space.

3.7 Model Performance Metrics

To determine the quality and correctness of a classification model, following basic evaluation metrics are computed

1. TP: The actual label and model classification are both positive.
2. FP: The actual label is negative but model classification is positive.
3. TN: The actual label and model classification are both negative.
4. FN: The actual label is positive but model classification is negative.

For a classification problem, accuracy which is computed as the ratio of the correctly classified examples to the total number of examples provided is a common evaluation metric. However, with this approach, in medical AI there is a shortcoming of getting the same metric value regardless of different cases. Let's assume that there is a model that predicts that any patient does not have emphysema regardless of patient's measurements. If just the accuracy is computed, a higher accuracy is obtained if there are patients with fewer cases of emphysema in the test set. But a higher accuracy does not necessarily mean a better model. Hence, it is customary to compute other advanced measures to check how well the model predicts positives for patients with disease and non-disease for cases that actually are diseased and non-diseased.

3.7.1 Precision and Recall

Recall is defined as the probability that the model predicts that the patient has diseases given that the patient has disease. It is also known as True Positive Rate or Sensitivity,

$$Recall = TP/(TP + FN) \quad (3.25)$$

Precision also called Positive Predicted Value, provides the probability that a patient has the disease given that the model prediction is positive.

$$Precision = TP/(TP + FP) \quad (3.26)$$

3.7.2 AUC Score

A classification model predicts the True Positives and Negatives based on the assumption of a threshold that dictates what output label is considered positive and negative. This value of the threshold could have been any random number hence this is an arbitrary choice and should not affect the decision provided by the model. A method that helps see how the threshold plays out the decision of the model is produced by a roc curve. The area under the ROC curve also called AUC or C-statistic which is a measure of goodness of fit. This summarizes the model

output across all thresholds, and provides a good sense of the discriminative power of a given model.

CHAPTER 4

METHODOLOGY

4.1 Tools and Resources Used

For the formulated thesis work, the following tools and resources are used:

1. **Python:** Python is a high-level programming language commonly used in machine learning research projects.
2. **Numpy:** Numpy is a library widely used for scientific computing in Python by providing a multidimensional array object, and functions and methods to process them.
3. **Pandas:** Pandas is a Python library for doing fast and quick data manipulation and analysis.
4. **Matplotlib:** Matplotlib is the most common plotting library for the Python language.
5. **Keras:** Keras is an open-source library that provides a Python interface for neural networks.
6. **Jupyter Notebook:** A Jupyter notebook allows us to write and execute Python code locally in web browsers. It makes very easy to code, execute and change in bits and pieces and hence it is widely used in scientific computing.
7. **Anaconda:** Anaconda is a framework with Python distribution for scientific computing.

4.2 Dataset

For the formulated problem, the ChestX-ray8 [34] dataset and Guangzhou [35] are used. Different experiments are carried to evaluate the representation learning performance of deep generative models and are compared with a baseline model.



Figure 4.1: A sample of chest x-ray [34]

4.3 Pre-processing

The input data consists of chest x-ray images which have very high resolutions. To prepare the dataset images before building a deep learning model, images are first prepared using the ImageDataGenerator class from the Keras framework. The ImageDataGenerator class provides support for basic data augmentation and preprocessing. Using the generator, the mean and standard deviation of the input data distribution is normalized for standardizing the input. The input after each epoch is shuffled to ensure ‘bad’ epochs that are not representative of the overall dataset in training is not used. Image size is also set to 64 by 64 pixels since neural networks receive input of the same size. This size is a reasonable dimension for the deep layer convolutional network considering the depth of the network and the computational overhead.

4.4 Proposed System

Our training data can be represented as $D_l \cup D_u$ where D_l is a set of labeled dataset and D_u is a set of unlabeled dataset. Our model comprises of two different systems. The first one is a variational autoencoder which is used to extract the disentangled representation of the input data, while the second one is a semi-supervised learning block.

After the VAE has been fully trained, the ”encoder” can be directly used to help with semi-supervised learning. Firstly, using all the labeled and unlabeled data

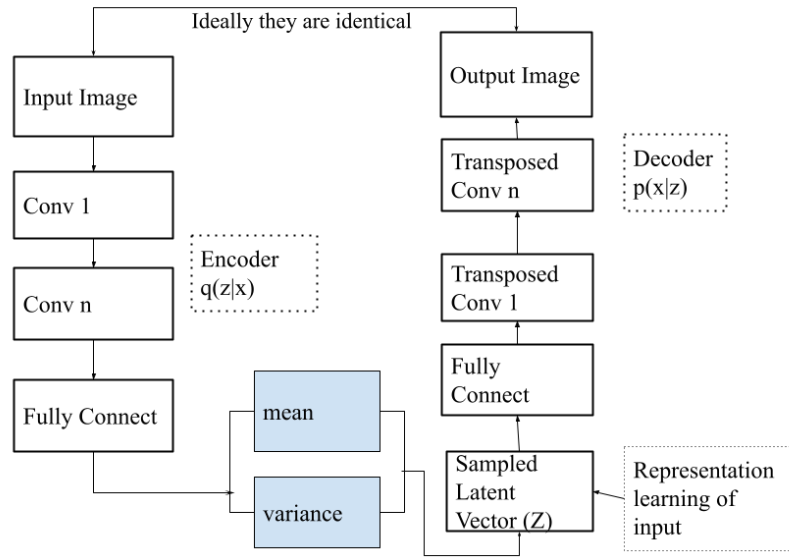


Figure 4.2: Variational Auto-Encoder

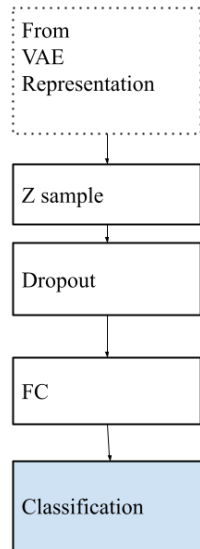


Figure 4.3: VAE SSL model

points, the VAE is trained which transforms the observed data (X) into the latent space defined by the Z variables. Using the pair of (Z, Y) , a standard supervised learning can be solved.

According to the analytical theory on machine learning, the latent space defined

by z should capture some useful information about our data such that it's easily separable in our supervised learning problem. So, the main idea in this work as presented in the diagrams 4.2 and 4.3 is that the disentangled representation learning of the input image using a VAE is fed into the semi-supervised model to make a classification of chest x-ray diseases.

CHAPTER 5

EXPERIMENT AND RESULTS

The aim of this research work was to evaluate the generalization ability of a representation learning model in disease classification using a limited labeled dataset. Experiment performed with chest x-ray images collected from standard source of ChestX-ray8 [34] dataset and Guangzhou dataset [35]. A total of 50000 images were collected with pneumonia condition or normal for training with an additional 5000 images for test set.

The first block of the proposed model, generating compressed representation of the input images, was first completed. The performance of different generative models was evaluated before selecting variational auto-encoder for subsequent stage. All the generative models were trained using self-supervised learning and compared for classification performance as well. In self-supervised method, the feature of the x-ray images are extracted in unsupervised manner, however the classification is performed using supervised method; hence, the name self-supervised.

5.1 Baseline model

To compare the generative models used for representation learning of x-rays, first a base model of a deep convolutional neural network is devised to compute the performance of the classifier. This is a simple supervised learning method implemented only to compare our the self-supervised learning ability of other models.

Initially, I experimented with 20000 images with a CNN model of 3 convolutional layers. I trained the model with a learning rate of 0.001, dropout value of 0.2, and the batch size of 32 images. Figure 5.2 shows the plot of loss vs epoch of the model; which reveals that the training data size is still under-represented relative to the validation dataset. This condition might occur if the training data is few compared to validation. Under-representation condition is identified when there

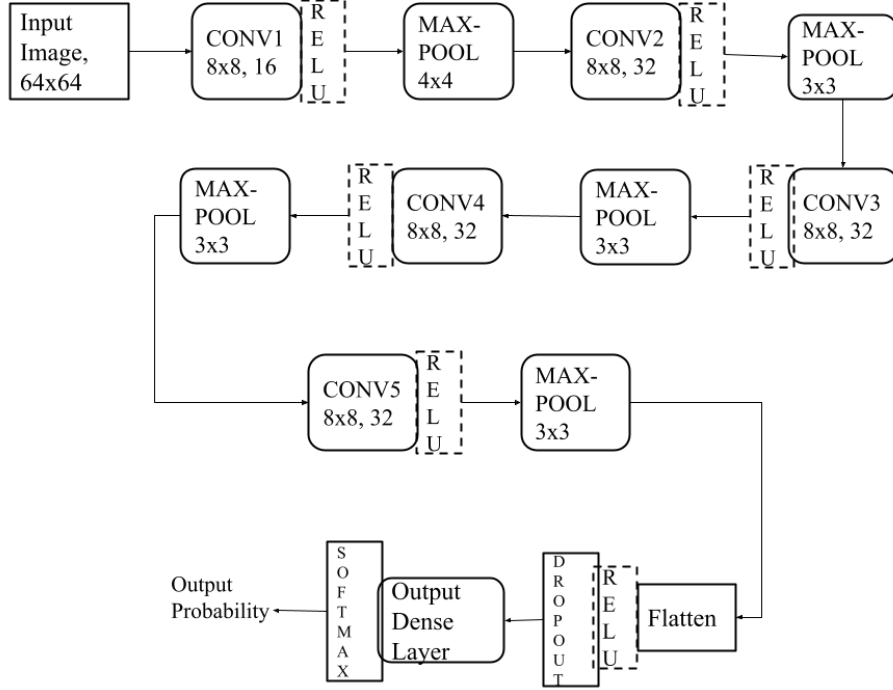


Figure 5.1: Baseline CNN model

is a large gap between training and validation loss even though both the loss are decreasing as the training has continued. Hence, I collected more training images for next experiment.

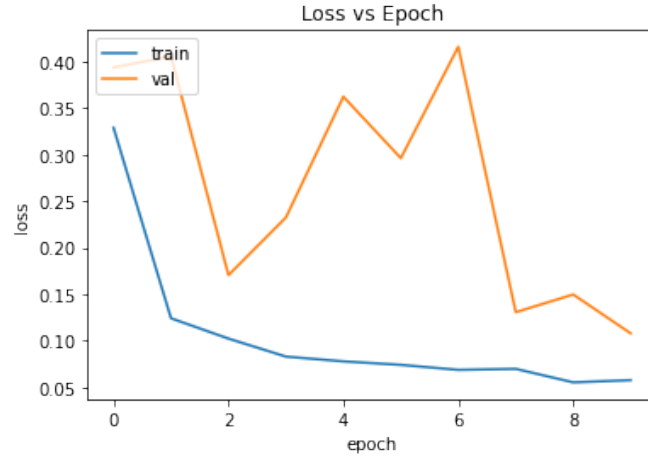


Figure 5.2: Loss vs Epoch of Baseline CNN model for 20000 training images

Evaluation metrics obtained by training the model and testing it against the test dataset for 20000 training images is shown in Table 5.1:

Table 5.1: Table depicting model performance of baseline model for 20000 training images

Metrics	Performance
Training accuracy	96%
Validation accuracy	86%
Testing accuracy	80%
Precision	0.76
Recall	0.98
F1 score	0.86
AUC score	0.74

For 50000 training images, a new CNN model consisting of 5 convolutional layers, and pooling layers; with RELU activation function was developed. The number of layers used in a deep learning model is determined by experiments or taking reference from similar standard projects. With an increment in training images, we generally increase the depth of the project. Similar image classification projects of similar number of training images are referenced while deciding the depth of the model architecture. It is important to notice that just an increase in number of layers won't improve the classification results of a CNN model as the architecture suffers from vanishing gradient problem. Also, computing gradients and back-propagating errors for very deep neural net is computationally expensive, so I settled on an accepted depth of CNN.

To train the CNN model, I used a learning rate of 0.001 which is the default learning rate value of Adam optimizer in Keras. A dropout value of 0.2 was applied in the final layer so that the model does not over fit the input data. The batch size, the number of training samples used by the optimizer before updating the model parameters, of 32 images was used. Figure 5.3 shows the plot of loss vs epoch.

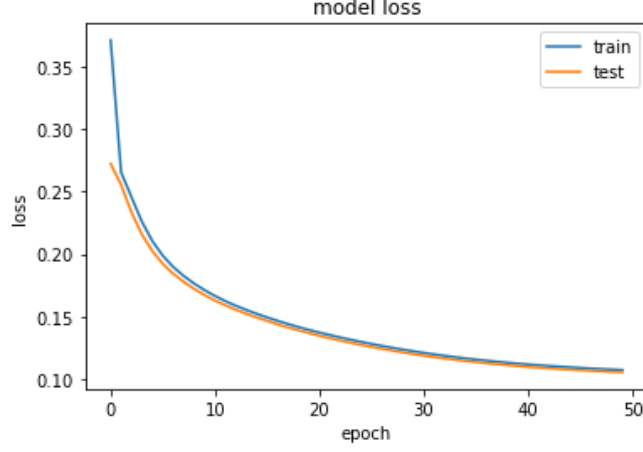


Figure 5.3: Loss vs Epoch of Baseline CNN model 50000 training images

Evaluation metrics obtained by training the model and testing it against the test dataset is shown in Table 5.2:

Table 5.2: Table depicting performance of baseline model

Metrics	Performance
Training accuracy	96.21%
Testing accuracy	80.73%
Precision	0.79
Recall	0.91
F1 score	0.80
AUC score	0.74

To compare how the size of overall labeled dataset affects the prediction accuracy on the test dataset, I again ran the baseline model on different dataset samples. The same architecture was used only by varying the training images number.

Table 5.3: Table depicting model performance on varying size of training dataset

Sample size	Accuracy
1000	43.35%
2000	48.44%
5000	60.97%
10000	67.31%
25000	76.75%
50000	80.73%

We can clearly observe that around 76% accuracy is achieved with 25000 samples, and we need to increase the datasize by two times to improve the accuracy from 76% to 80% with this baseline CNN network. If we have only around 5000 labeled

samples but an abundant of 45000 unlabeled samples then is there any method that helps us learn from the unlabeled dataset in an unsupervised manner that contribute to improving the accuracy with a limited labeled dataset? This is the goal of semi-supervised learning and subsequent models.

5.2 Auto-encoder model

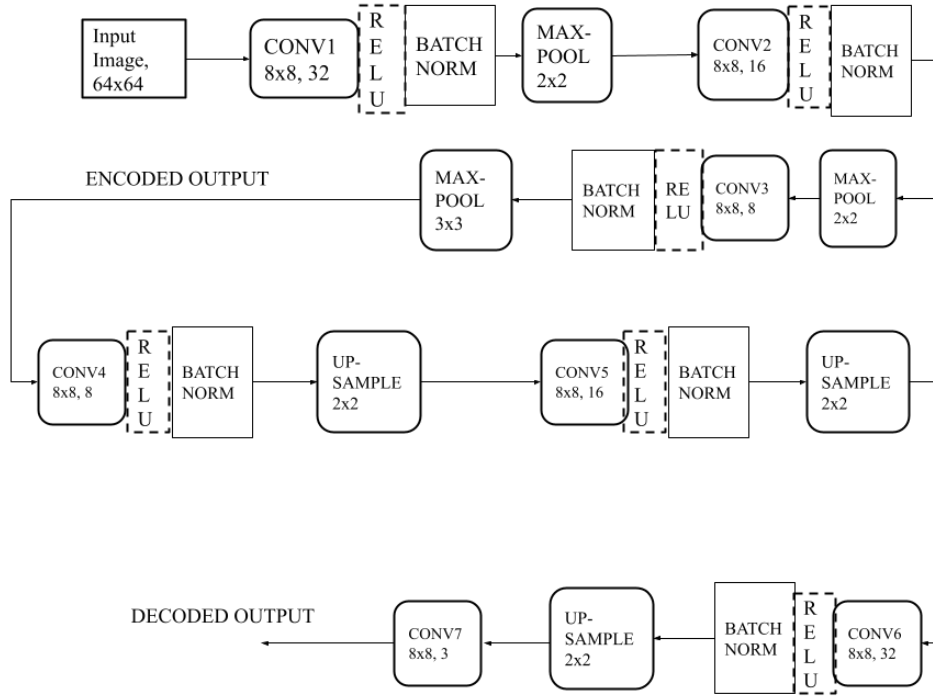


Figure 5.4: Auto-encoder model

Auto-encoders take an unsupervised approach for representation learning by designing an encoder-decoder architecture of a neural network. The neural net forces the original input to output a compressed representation due to the bottleneck in the network. Then, a decoder reconstructs the compressed input utilizing the data distribution and correlation learnt by the encoder network. For most auto-encoder, the loss function consists of a reconstruction loss with an added regularizer to prevent overfitting of the data. The architecture of an autoencoder is displayed in Figure 5.4 which consists of convolution and max-pool layers in the encoder side whereas in decoder side, the attempt to recreate the encoded output is handled by subsequent convolution and upsample layers.

For 20000 initial images, three layers of encoder-decoder architecture was used with standard weight and learning rate initialization. Figure 5.5 shows the loss vs epoch plot and it clearly shows that the model has fairly fit well with training and validation loss has decreased to a point of stability with small gap between them. However, there is a very fast convergence of the loss hence, for the subsequent stage, more training images were collected.

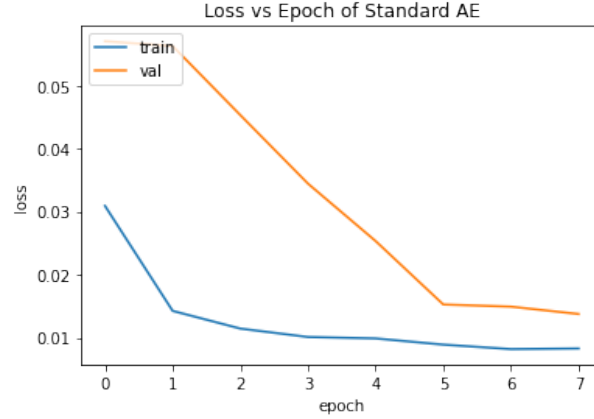


Figure 5.5: Loss vs Epoch of Auto-encoder model of 20000 training images

Evaluation metrics obtained by testing the model against the test dataset for 20000 training images is shown in Table 5.4:

Table 5.4: Table depicting model performance of an auto-encoder model for 20000 training images

Metrics	Performance
Training accuracy	95%
Validation accuracy	75%
Testing accuracy	87%
Precision	0.80
Recall	0.95
F1 score	0.88
AUC score	0.72

A drawback of standard auto-encoder is that when the input images are corrupted by even small amount of noise, the the reconstructed images start to distort. Figure 5.6 reveals the reconstruction ability of the AE model. There is no sharpness in the reconstructed images due to the reconstruction loss that averages out the differences between individual pixel values. Also, there is a no definite distribution to sample latent vectors, hence, it is not possible to generate random images. Figure 5.7

shows that we cannot create new images from the standard auto-encoder model. Hence, the auto-encoder model is not the perfect candidate for generative modeling.

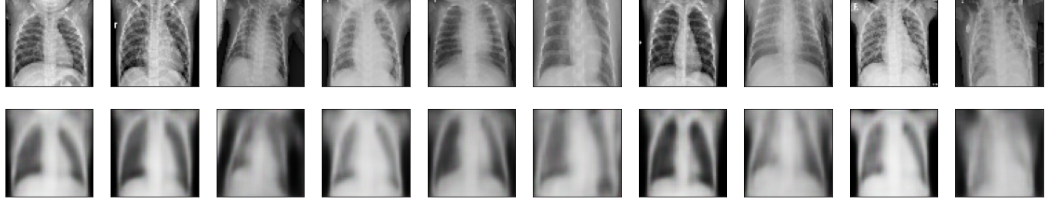


Figure 5.6: Reconstruction by Auto-encoder model

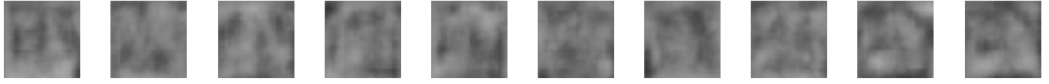


Figure 5.7: Attempting to create new images from randomly sampling from latent vector

After the reconstruction, the model was again trained in self-supervised manner, to check how the feature extracted from the latent space would be useful in the classification of diseases labeled in the x-ray image.

Table 5.5: Table depicting model performance of an auto-encoder model for 50000 images

Metrics	Performance
Training accuracy	91%
Testing accuracy	82%
Precision	0.76
Recall	0.89
F1 score	0.82
AUC score	0.72

To train this self-supervised model, I used a learning rate of 0.001 which is the default learning rate value of Adam optimizer in Keras. The batch size of 32 images was used and the model was trained for 20 epochs. The images were shuffled after every epoch. Evaluation metrics obtained by testing the model against the test dataset is shown in Table 5.5.

5.3 Denoising auto-encoder model

Denoising auto-encoders are tasked with not just reconstructing the input by training a neural network but also be sensitive enough to the important information

of the input image and discard the noises. Denoising auto-encoder is able to produce a generalized encoder-decoder model even when the input data is slightly corrupted. With this approach, our model isn't memorizing the training data since the input and output are different. However, the problem of generating new input data from the normal distribution persists.

The ability of denoising auto-encoder ability in reconstructing the original input even when the input images are corrupted is shown by Figure 5.8.

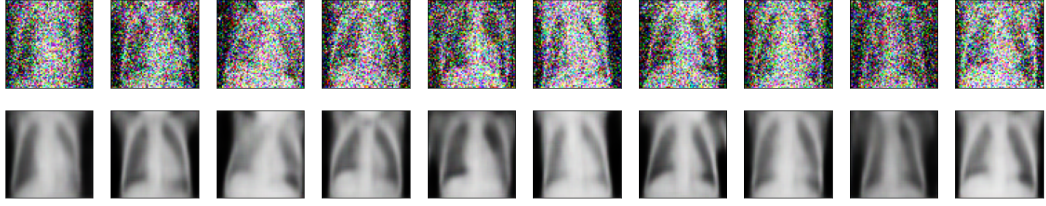


Figure 5.8: Input images and reconstructed image from denoising AE

After the representation learning ability of denoising AE model, I again use the feature extracted in evaluating the self-supervised classification ability of the model and obtain the following results.

Table 5.6: Table depicting model performance of a denoising auto-encoder model

Metrics	Performance
Training accuracy	92%
Testing accuracy	70%
Precision	0.66
Recall	0.88
F1 score	0.75
AUC score	0.63

5.4 Variational auto-encoder

Variational Auto-encoders (VAE) are designed to address the most drawbacks of auto-encoder models. VAE is trained to create new images from randomly sampling the latent vector of the encoder model which are assumed to bear a standard normal distribution. That means, while a standard auto-encoder maps the input data to a point in the latent space, variational auto-encoder encodes the input data to a standard normal distribution function which is represented in terms

of mean and co-variance. To make sure that the encoder maps the input data to a standard normal distribution, the loss function is modified to incorporate the KL divergence loss that ensures that the encoding of the input data are indeed similar to a normal distribution. Hence, to generate new images, we can just sample from the standard normal distribution and the decoder will create the new images. The decoder is however identical to a standard auto-encoder.

5.4.1 Model I

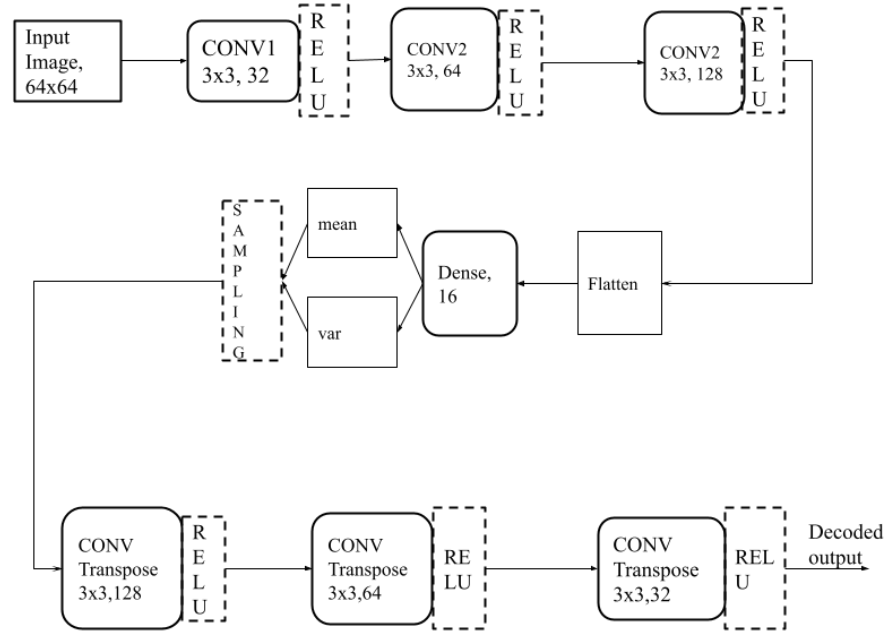


Figure 5.9: Block diagram of VAE Model 1

Firstly, I created a VAE model using only three convolution layers with relu as activation function and no batch normalization. The block diagram of the model is shown in Figure 5.9 and is self-explanatory. Figure 5.10 shows the reconstruction ability of the VAE.

When the feature extracted by this model was evaluated in self-supervised model, the evaluation metrics obtained were poorer than the earlier AE models. Use of only 2 latent dimension i.e. the bottleneck might have been too small and might

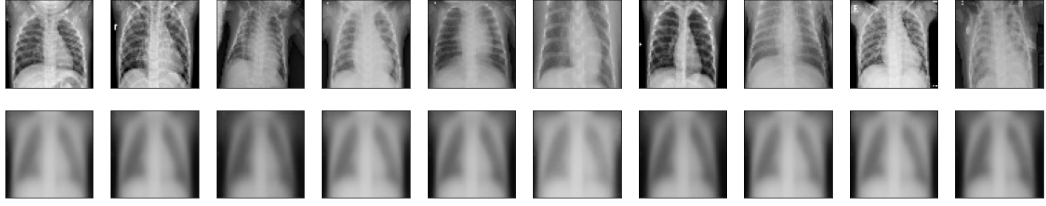


Figure 5.10: Reconstruction of input by VAE Model 1

have compressed the x-ray image too much that important information was lost during encoding process. Or, as revealed by the plot in Figure 5.11, we have clearly underfitted the training dataset, which can be recognized by an almost flat training loss. Model has learnt nothing from the training cycle due to its limited complexity.

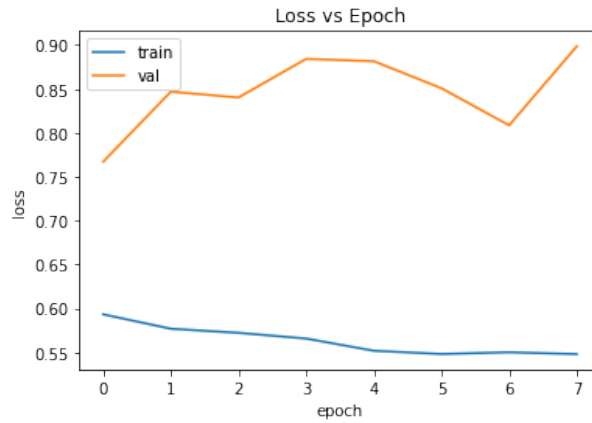


Figure 5.11: Loss vs Epoch of VAE Model 1

AUC score from the evaluation Table 5.7 also shows that the model is a random classifier and is not better than any random predictor that predicts presence or absence of disease out of whim.

Table 5.7: Table depicting model performance of VAE1

Metrics	Performance
Training accuracy	71%
Testing accuracy	53%
Precision	0.525
Recall	0.4
F1 score	0.45
AUC score	0.5

5.4.2 Model II

Considering the results of VAE Model I, I increased the encoder-decoder convolution and convolution transpose layers to 5, decided to use leaky relu instead of Relu, added batch normalization to the input images and also increased the latent dimension, bottleneck for compression, to 3 from 2. Figure 5.13 shows the reconstruction ability of the VAE model 2. There is blurriness in image because of the RMSE loss that averages out the difference between image pixels.

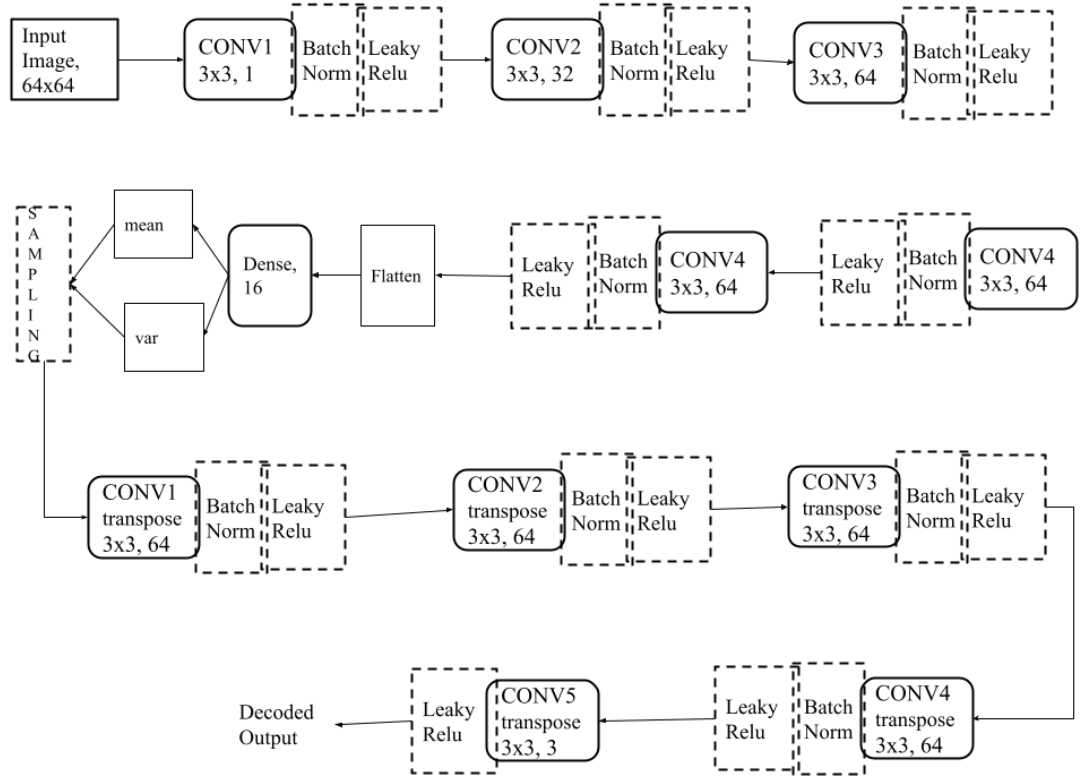


Figure 5.12: Block diagram of VAE Model 2

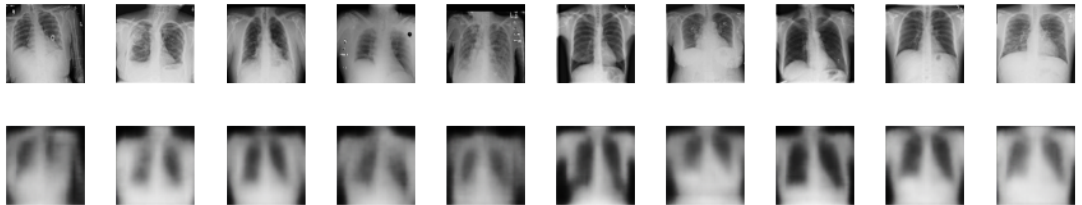


Figure 5.13: Reconstruction ability of VAE Model 2

Sampling randomly from the latent vector gives us the creation of images in figure Figure 5.15. VAE thus bears the ability to produce new input from vectors sampled

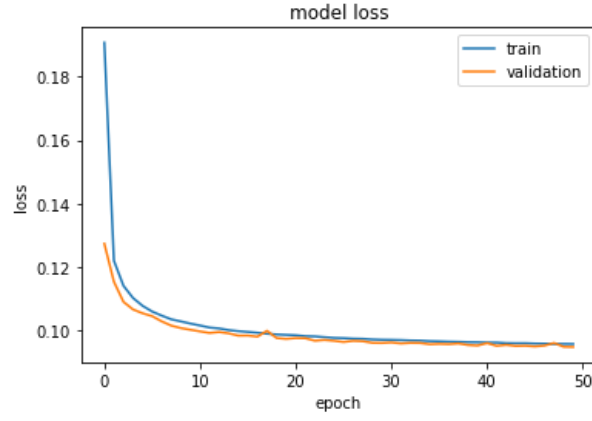


Figure 5.14: Loss vs Epoch of VAE Model 2

from the a normal gaussian distribution.

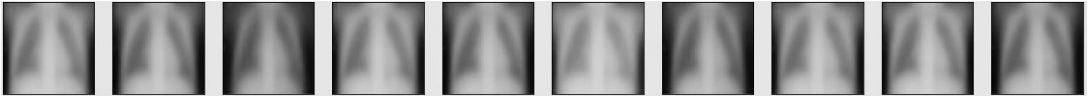


Figure 5.15: Sampling randomly from latent vector of VAE Model 2

To check if the latent vectors are indeed distributed normally, I plot the first 50 elements of the encoded latent vector Z in Figure 5.16 to check if the addition of KL divergence to the VAE is justified.

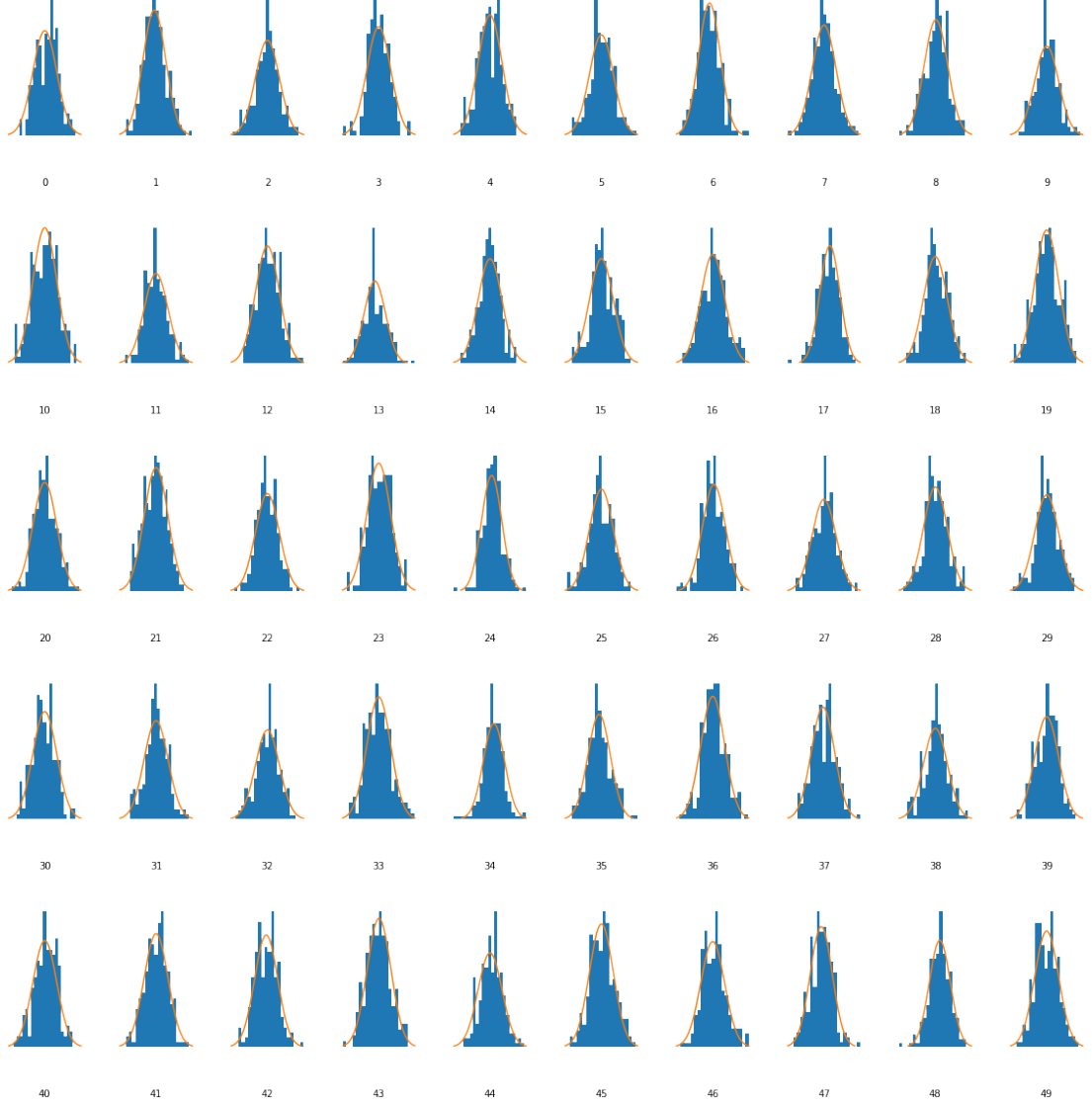


Figure 5.16: Normal distribution of first fifty elements of the encoded latent vector

The feature extracted by this VAE model was evaluated in self-supervised manner and the evaluation metrics was obtained as shown in Table 5.8.

Table 5.8: Table depicting model performance of VAE2

Metrics	Performance
Training accuracy	80%
Testing accuracy	73%
Precision	0.78
Recall	0.85
F1 score	0.82
AUC score	0.81

5.5 Generative Model Comparison

Table 5.9: Table depicting overall comparison of generative model performance

	Accuracy	Precision	Recall	F1	AUC
Baseline model	80.7%	0.79	0.91	0.8	0.74
Auto-encoder model	82%	0.76	0.89	0.82	0.72
Denoising auto-encoder	70%	0.66	0.88	0.75	0.63
Variational Auto-encoder I	53%	0.525	0.4	0.45	0.5
Variational Auto-encoder II	73%	0.78	0.85	0.82	0.81

5.6 Semi-supervised learning

As described in section 4.4, a variational auto-encoder is a generative model that produces the latent representation of the input data in an isotopic normal distribution form. The goal of VAE is then to recreate the original input using a decoder that translates the latent representation into the reconstructed input. Assuming we have $N+M$ number of dataset where N are labeled images and M are unlabeled images, the property of VAE in semi-supervised learning uses the following approach.:

1. Perform the training of a VAE model using $N+M$ datapoints.
2. Transformation of the $N+M$ input data (X) into the latent space defined by the latent vector Z
3. Implementation of supervised learning using only the hidden representation of labeled data, (Z,Y)

Theoretically, the latent representation vector, Z , should capture important information of the input images that are separable in our classification model.

We selected the VAE as our generative model since Table 5.9 shows that the discriminative ability of the model is better compared to rest of the representation learning models. Although the test accuracy was low compared to an auto-encoder model, its precision was larger compared to the AE model.

Table 5.10: Table depicting VAE model performance on varying size of labeled dataset

Labeled size	Accuracy
1000	37.2%
2000	62.1%
5000	72.9%

Table 5.11: Comparison of model vs size of labeled dataset vs accuracy

Model	N=1000	N=2000	N=5000
Baseline (CNN)	43.35%	48.44%	60.97%
M1 Model [33]	42%	63%	71%
M2 Model [33]	62%	80%	90%
Proposed model	37.2%	62.1%	72.9%

The proposed model performed better than the baseline model when the model was trained with 2000 or more labeled images and a large number of unlabeled images. When compared against the existing state of the art generative semi-supervised modeling M1 and M2, the proposed model performed almost the same or better than the M1 model. However, M2 model still outperformed the proposed model. This could be either because of the dataset (M1 and M2 were trained and tested against the standard MNIST dataset) or the complicated architecture of M2 that combines the property of VAE, AE and baseline CNN in an end-to-end manner for feature extraction.

CHAPTER 6

CONCLUSION

6.1 Conclusion

Disentangled representation learning for semi-supervised classification of chest x-rays is presented in this dissertation. Baseline CNN model for disease diagnosis, comparison of generative modeling capacity of different models and use of VAE for semi-supervised classification has been implemented and experimented successfully with positive results. Semi-supervised classification is still however a difficult field of research. In what manner do unlabeled dataset can contribute to disease diagnosis is a fairly new topic of research.

For the experimentation with the chest x-rays, 50000 chest x-rays with presence or absence of Pneumonia were collected from standard medical dataset sources. First a baseline model using convolutional neural network was designed for classification of x-rays based on the different sizes of labeled training dataset. For a dataset size of 1000, 2000, 5000, 10000, 25000 and 50000 labeled chest x-ray images, classification accuracy of 43%, 48%, 61%, 67%, 77% and 81% were obtained. To improve our classification accuracy from 76% with 25000 samples to 80%, the data size needed to be increased by two times. If there were 5000 labeled samples but an abundant of 45000 unlabeled samples then how could we utilize that unlabeled samples to improve the accuracy? This was the goal of semi-supervised learning and this dissertation.

Firstly, before selecting the variational auto-encoder as the generative model for semi-supervised classification, basic representation models like auto-encoder and denoising auto-encoder were both experimented. While the auto-encoder model showed better test accuracy of 82% during self-supervised classification compared to VAE accuracy of 73%, it failed to recreate the original samples over a random range of Gaussian parameters. AUC score (0.81) and precision (0.78) of VAE was comparatively better. Also, VAE model was able to create new images while

sampling randomly from the latent vector, hence, VAE was selected as our model for its better generative and discriminative ability.

For the final experiment, after the transformation of the VAE model using $N+M$ data-points where N denotes number of labeled images and M denotes number of unlabeled images, supervised learning using only the hidden representation of labeled data was implemented. The VAE model was able to outperform the baseline CNN model with limited labeled dataset. While the CNN model's accuracy was 43%, 48% and 61% with 1000, 2000 and 5000 labeled images, the VAE model had better accuracy with more than 2000 labeled images and subsequent unlabeled images. The model's accuracy with 1000, 2000 and 5000 labeled images and with 49000, 48000 and 45000 unlabeled images respectively were 37%, 62% and 73%.

6.2 Limitation

It must be noticed that, for any deep learning diagnosis algorithm to be practically applied, its performance criteria must be globally accepted. Any model with accuracy below 95% or with unaccepted precision or recall (less than 0.8) or low AUC score (less than 0.8-0.9) cannot be taken into practical consideration as such models could produce disastrous results. The aim of this thesis, thus, is not to build a diagnosis system but to inspect how unlabeled dataset can benefit the generalization of a classifier. Since there is always an abundance of unlabeled images, a semi-supervised model can pave the way for rapid application of such images in building an medical diagnosis model, the performance of which, however, must be closely monitored before implementing them in real life.

6.3 Future Scope

Generative modeling is a very active research topic particularly its implications on semi-supervised learning. Abundance of unlabeled dataset and the ability of representation learning to mimic the underlying data distribution makes it the perfect fit for exploring semi-supervised learning. For further works, this research can be extended to:

- Propose a multi-label disease diagnosis using limited labeled dataset of medical images with multiple pathology conditions
- Compare the ability of generative adversarial network in SSL
- Implement the existing model to evaluate disease diagnosis in other medical images other than x-rays

REFERENCES

- [1] Kenji Kawaguchi, Yoshua Bengio, Vikas Verma, and Leslie Pack Kaelbling. Generalization in machine learning via analytical learning theory. *Neural and Evolutionary Computing (cs.NE)*, 2018.
- [2] Shang Da. A generative model for semi-supervised learning. *Creative Components*.382., 2019.
- [3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- [4] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Conference on Neural Information Processing Systems*, 2014.
- [5] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*, 2015.
- [6] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *Conference on Neural Information Processing Systems*, 2019.
- [7] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1415–1424, 2017.
- [8] Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. Disentangled representation learning for non-parallel text style transfer. *Empirical Methods in Natural Language Processing*, 2018.
- [9] Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. Deconvolutional paragraph representation learning. *Advances*

- in Neural Information Processing Systems (pp. 4170-4180). Neural information processing systems foundation.*, 2017.
- [10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
 - [11] Jan Chorowski, Ron J Weiss, Samy Bengio, and Aäron van den Oord. Unsupervised speech representation learning using wavenet autoencoders. *IEEE/ACM transactions on audio, speech, and language processing*, 27(12):2041–2053, 2019.
 - [12] Vikash Balasubramanian, Ivan Kobyzev, Hareesh Bahuleyan, Ilya Shapiro, and Olga Vechtomova. Polarized-vae: Proximity based disentangled representation learning for text generation. *arXiv preprint arXiv:2004.10809*, 2020.
 - [13] Tristan Bepler, Ellen D Zhong, Kotaro Kelley, Edward Brignole, and Bonnie Berger. Explicitly disentangling image content from translation and rotation with spatial-vae. *Conference on Neural Information Processing Systems*, 2019.
 - [14] Chung-Ching Lin, Ying Hung, Rogerio Feris, and Linglin He. Video instance segmentation tracking with a modified vae architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13147–13157, 2020.
 - [15] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.
 - [16] Ellen Riloff and Janyce Wiebe. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 105–112, 2003.
 - [17] Beatriz Maeireizo, Diane Litman, and Rebecca Hwa. Co-training for predicting emotions with spoken dialogue data. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 202–205, 2004.
 - [18] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. 2005.

- [19] Thorsten Joachims et al. Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pages 200–209, 1999.
- [20] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *International Conference on Learning Representations*, 2016.
- [21] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.
- [22] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.
- [23] Yi Luan, Mari Ostendorf, and Hannaneh Hajishirzi. Scientific information extraction with semi-supervised neural tagging. *arXiv preprint arXiv:1708.06075*, 2017.
- [24] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. 2002.
- [25] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. Stacked what-where auto-encoders. *Neural and Evolutionary Computing*, 2015.
- [26] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2):103–134, 2000.
- [27] Henry A Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38, 1998.
- [28] Dong Wang, Yuan Zhang, Kexin Zhang, and Liwei Wang. Focalmix: Semi-supervised learning for 3d medical image detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3951–3960, 2020.
- [29] Agisilaos Chartsias, Thomas Joyce, Giorgos Papanastasiou, Scott Semple, Michelle Williams, David Newby, Rohan Dharmakumar, and Sotirios A

- Tsaftaris. Factorised spatial representation learning: Application in semi-supervised myocardial segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 490–498. Springer, 2018.
- [30] Prashnna Kumar Gyawali, Sandesh Ghimire, Pradeep Bajracharya, Zhiyuan Li, and Linwei Wang. Semi-supervised medical image classification with global latent mixing. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 604–613. Springer, 2020.
- [31] Hong Shang, Zhongqian Sun, Wei Yang, Xinghui Fu, Han Zheng, Jia Chang, and Junzhou Huang. Leveraging other datasets for medical imaging classification: evaluation of transfer, multi-task and semi-supervised learning. In *International conference on medical image computing and computer-assisted intervention*, pages 431–439. Springer, 2019.
- [32] Nematollah K Batmanghelich, Ben Taskar, and Christos Davatzikos. Generative-discriminative basis learning for medical imaging. *IEEE transactions on medical imaging*, 31(1):51–69, 2011.
- [33] Diederik P Kingma, Danilo J Rezende, Shakir Mohamed, and Max Welling. Semi-supervised learning with deep generative models. *Conference on Neural Information Processing Systems*, 2014.
- [34] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2097–2106, 2017.
- [35] Kang; Goldbaum Michael Kermany, Daniel; Zhang. Labeled optical coherence tomography (oct) and chest x-ray images for classification. In *Mendeley Data*, 2018.