Creating an Image Noise Robust Model for TB Detection in Chest X-Rays

Adrian McIntosh
Electrical and Computer Engineering
University of California, Los Angeles
Los Angeles, United States
adrianmcintosh@myucla.edu

***Abstract*—This paper describes the development of a CNN to classify Tuberculosis in Chest X-rays to be more robust to noise through a variety of methods including Denoising Techniques, an adjusted auto-encoder, as well as a combined approach.**

I. Introduction and Background

A. Motivation

A common tradeoff in medical imaging is between the quality and noise of the image and the dose or time taken to make the reading. In X-ray imaging, using doses of radiation leads to lower image quality. Finding a way to classify X-Ray images having certain diseases or problems such as tuberculosis is something that has been researched and experimented with, but by making use of Signal and Image Processing techniques, and by adapting the design of the Neural Network used for Classification, we can make it more robust to this noise, allowing the X-Rays to be taken more safely for the patient.

A more standard way of managing this low-dose noise is to first run a denoising algorithm on the noisy images to produce an estimation of a denoised version of the image and then feed that into the Neural Network. However, the theory is that this method removes information from the image, as much of the noise being removed still contains useful information about the signal. For example, Poisson Noise, commonly found in X-ray images is a function of the signal (or signal dependent) which implies that it could be useful to avoid removing the noise, but rather use it as a feature to inform a representational model of the image, that is not an estimation of the original image itself.

B. Problem Statement

With this in mind, we can summarise the problem statement as follows:

How can we develop a Convolutional Neural Network to be more robust to image noise by using the noise as a feature rather than removing it?

II. Methodology

A. Data set & Starter Code

The data set used in this project comes from [1]. And consists of about 4000 Chest X-ray images, a quarter of which depict X-rays with Tuberculosis, and the rest are Normal. The intention of this dataset is for them to be used for creating Tuberculosis Detection models.

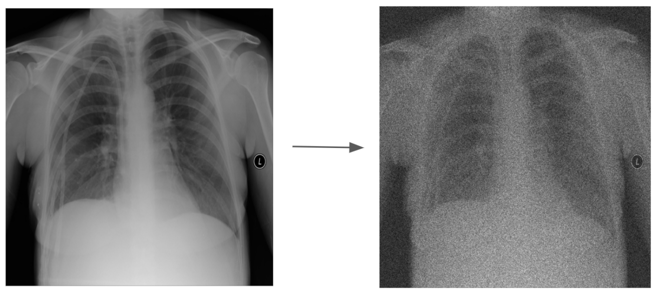


Fig. 1. Clean to Noisy Image Generation

For this project, an existing designed model will be used and modified to work on a noisy version of this data.

The existing model that was chosen was [2] which was used as a starting point for implementing this project.

B. Creating the Noise

Before any tests or experiments could be done on the data, the noisy dataset had to be generated. This was done using the Poisson-Gaussian Mixture Model from [3] and [4] which is a representation of the noise often found in low-dose X-ray scans. The model is described mathematically in Equation 1 and the function used to implement this model was adapted from [4].

$$\alpha \sim \mathcal{P}(ax), \quad \beta \sim \mathcal{N}(0, b^2), \quad \eta_p = \frac{1}{a}\alpha, \quad \eta_g = \beta, \quad y = \eta_p + \eta_g = \frac{1}{a}\alpha + \beta,$$
.

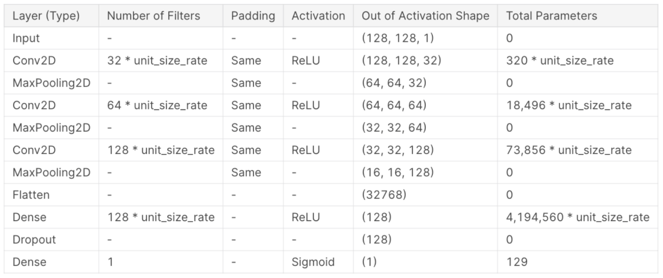
Equation 1: Poisson-Gaussian Mixture Model

This was first attempted with the parameters set out in [3], but the noise created did not work as well as expected, and they were manually adjusted until [chosen values] were chosen.

An example of the original (clean) images and the noisy generated counterpart is shown in Figure 1.

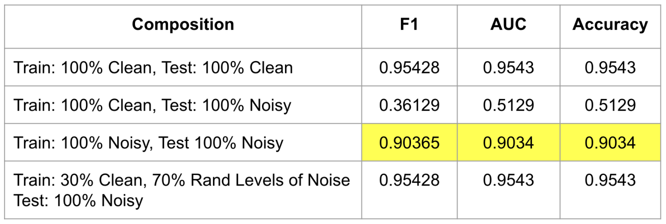
Three sets of data were used in the project. The first was the original "clean" dataset without any noise added. Second, a Noisy dataset was created with the realistic parameters described, and a third Mixed set made up of 30% clean images and 70% made up of a random distribution of noisy images generated with parameters ranging from [x to y]

After this, a train-test split was created with a ratio of 80:20.



Layer (Type)	Number of Filters	Padding	Activation	Out of Activation Shape	Total Parameters
Input	-	-	-	(128, 128, 1)	0
Conv2D	32 * unit_size_rate	Same	ReLU	(128, 128, 32)	320 * unit_size_rate
MaxPooling2D	-	Same	-	(64, 64, 32)	0
Conv2D	64 * unit_size_rate	Same	ReLU	(64, 64, 64)	18,496 * unit_size_rate
MaxPooling2D	-	Same	-	(32, 32, 64)	0
Conv2D	128 * unit_size_rate	Same	ReLU	(32, 32, 128)	73,856 * unit_size_rate
MaxPooling2D	-	Same	-	(16, 16, 128)	0
Flatten	-	-	-	(32768)	0
Dense	128 * unit_size_rate	-	ReLU	(128)	4,194,560 * unit_size_rate
Dropout	-	-	-	(128)	0
Dense	1	-	Sigmoid	(1)	129

TABLE I
ARCHITECTURE OF CNN



Composition	F1	AUC	Accuracy
Train: 100% Clean, Test: 100% Clean	0.95428	0.9543	0.9543
Train: 100% Clean, Test: 100% Noisy	0.36129	0.5129	0.5129
Train: 100% Noisy, Test 100% Noisy	0.90365	0.9034	0.9034
Train: 30% Clean, 70% Rand Levels of Noise Test: 100% Noisy	0.95428	0.9543	0.9543

TABLE II
TRAIN TEST BASELINE EXPERIMENTS

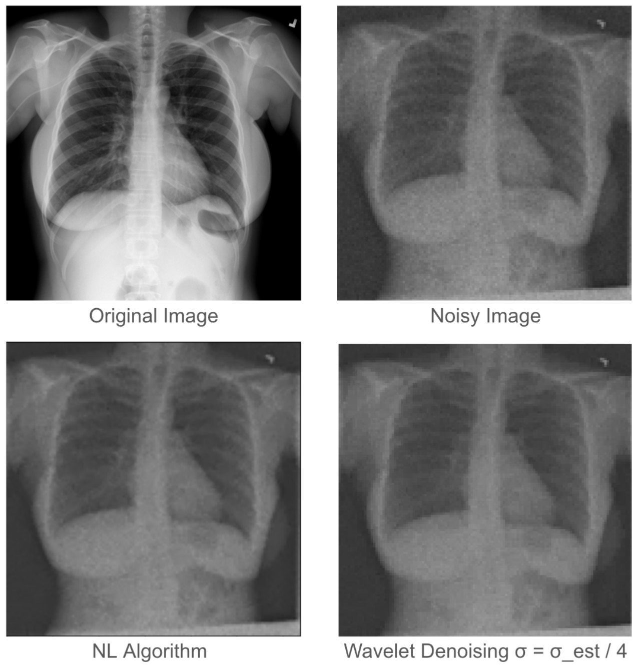


Fig. 2. Denoising Methods

C. *CNN Architecture*

The starting architecture used was the one from [2] and consisted of the architecture shown in TABLE I.

The Preprocessing of the images was also incorporated into the model used for this paper. This included the following:

1) **Data Type conversion**: the images were converted to NumPy arrays to save memory
2) **Data Augmentation:** Due to the imbalanced proportions of TB and Normal on the dataset, a more even distribution was created by adding scaled and rotated versions of the TB images to the dataset.

D. *Initial Testing: Training and Test Sets*

A few different experiments were run to get a baseline performance measure for the CNN Architecture.

First, the clean data set was tested, receiving an accuracy score of 96%. The rest of these results are shown in TABLE II.

From this, it can be seen that the Noisy Training and Testing situation gives us an accuracy of about 90%. It is this case that is most relevant to our situation described in the introduction, so this will be used as the baseline for testing the other changes to the model.

E. *Noise Removal Techniques*

A logical and common way to make an image classifier more robust to image noise is to try to remove as much of the noise as possible. To compare the different methods, this technique was tested on the data. Two denoising algorithms were tested: Non-Local Means and Wavelet Denoising. Examples of these are shown in Figure 2.

1) Non-Local Means: This algorithm, described in [5] essentially works by separating the image into pixel centers and their surrounding "patches" and comparing them. The output image is computed as a weighted average of the pixel values with higher weights being assigned to more similar patches, in theory reducing noise.

2) Wavelet Denoising: This algorithm works by taking the wavelet transform of an image, applying a high- and low-pass filter to the image in the wavelet domain, and then computing the inverse wavelet transform to get the denoised image [6]. The model was set up to run each of these algorithms and feed the denoised images into the CNN. This was done once for each NLM and Wavelet Denoising.

F. *Adjusting the CNN Model*

The CNN model itself was then adjusted in a few different ways, to find the effects on the noise-robustness of the model.

G. *Auto-encoder*

An autoencoder is a type of neural network typically used for compression or image denoising, but the design of an autoencoder can be used to inform a larger CNN by encoding a lower-dimensional representation of the input images and feeding that into the model.

Essentially an Autoencoder is a shallow CNN that is trained to recreate an input image as the output of the network as shown in Figure 3. This way, the autoencoder can extract the features of the set of images that are most essential to represent the information in the image in the encoded (hidden layer). The idea is that this lower-dimensional representation,

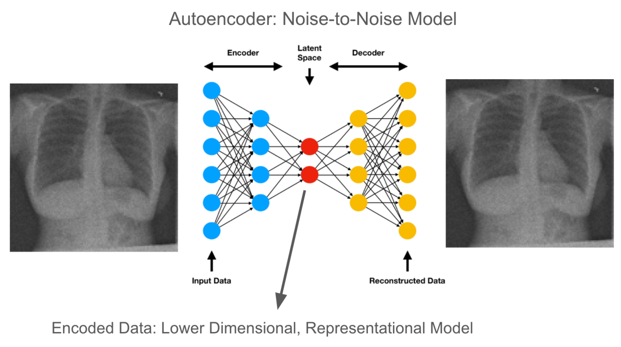

Autoencoder: Noise-to-Noise Model
Encoder
Latent Space
Decoder
Input Data
Reconstructed Data
Encoded Data: Lower Dimensional, Representational Model

Fig. 3. General Autoencoder

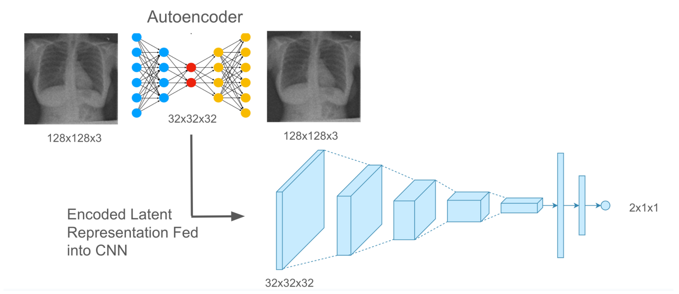


Fig. 4. Modified CNN with Autoencoder

including the noise can be used to best represent the image without losing information through denoising.

In the project, a noise-to-noise autoencoder was trained on the Noisy version of the dataset. Once this was completed, the hidden layer (of lower-dimensional representation) was then fed as input to the CNN (just changing the dimensions of the input) and it was trained and tested on this. This is shown in Figure 4.

H. Fast Fourier Transform

Based on the success of the wavelet denoising algorithm and the autoencoder method, it is clear that using representational models has promise. For this reason, the CNN was adjusted with the use of an alternative representation - the Fourier Transform. An example of the images used is shown in Figure 5.

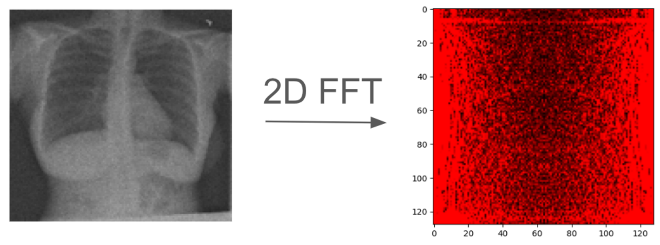


Fig. 5. Example of 2-dimensional FFT

$$\textbf{L1:}\quad \text{Modified loss function} = \text{Loss function} + \lambda \sum_{i=1}^{n} |W_i|$$

$$\textbf{L2:}\quad \text{Modified loss function} = \text{Loss function} + \lambda \sum_{i=1}^{n} W_i^2$$

Fig. 6. Regularization Equations

Composition	**PSNR (dB)**	**SSIM**	**F1**	**AUC**	**Accuracy**
Nearest Local Means	**13.35**	0.4840	0.92	0.92	0.92
Wavelet Denoising σ = σ_est / 4	**13.35**	**0.4920**	**0.944**	**0.944**	**0.944**

TABLE III
RESULTS FROM DENOISING METHODS

I. Regularization

Based on the results from the initial testing of the train-test configurations, the train noise test noise model was overfitting. Adding Regularisation to the Loss function of the model is a standard technique for combatting overfitting. Both L1 and L2 Regularisation were experimented with and the best parameters for lambda (as in Figure 6) were used.

III. Results and Discussion

The results of the experiments are discussed in the following section.

A. Training and Test Sets

As seen in TABLE II, the baseline where the CNN is trained on clean data but tested on Noisy data performs very poorly, however once trained on a noisy or mixed training set, is much more successful.

We also see that the Mixed training set model performed the best on the Noisy Data. This is likely because being mixed, it was the least likely to overfit the data.

B. Denoising Techniques

[ref: table of denoising results] shows the results of the different denoising algorithms on the image classifier.

Additionally, in TABLE III we compare how the accuracy of the classifier and the noise measurement relate to each other. Even though the SSIM of the Wavelet Denoiser is less than 1% greater than that of the NLM Denoiser, the increase in accuracy is significant (more than 2%).

C. Adjusting the CNN

The following section describes the results of how the adjustments to the CNN architecture that were made affect the noise-robustness of the model.

The results of the different adjustments are shown in TABLE IV. From this, we can see that the Auto-encoder was the most effective adjustment made.

Method	F1	AUC	Accuracy
Training on Noisy Data	0.90365	0.9034	0.9034
Denoising: NLM	0.92	0.92	0.92
Denoising: Wavelet	**0.944**	**0.944**	**0.944**
Autoencoder Adapted Model	**0.937**	**0.937**	**0.937**
Fourier Transform Model	0.925	0.925	0.925
Aggressive Regularization (L2 norm)	0.92	0.92	0.92

TABLE IV
COMPARISON OF CNN ADJUSTMENT RESULTS

D. Discussion

Based on the results, the two methods that involved the use of representational models (the Wavelet Denoising and Auto-encoder) were the most effective at contributing to the noise-robust model. This suggests there is potential for further developing representations of the noisy images that provide as much relevant information as possible.

Additionally, seeing that all of the methods used were able to improve the model, we can see that a combined approach to the model could be beneficial, making use of different aspects of this project to inform a more effective solution.

IV. CONCLUSION

The following section describes a few final points regarding the project.

A. Limitations

Although largely successful, this project has a few limitations and areas that could be improved:

1) **High Baseline Performance**
The lowest accuracy the project was working to improve upon was already 90%. This has made it more difficult to compare the effectiveness of the different methods as they have such a limited range to improve within.
2) **Confidence in Accuracy Ratings**
The method for computing the accuracy score in this project has space to improve. Currently, it only works on the train-test split that is the same groupings for every method, while a more accurate model in general might be trained using K-Fold Cross Validation for example.
3) **Time Constraints**
As a class project, this paper was completed within the constrained time frame of a Quarter at UCLA, so time constraints meant that the project could only reach a limited stage for this submission, but the next steps are outlined in the next section.

B. Future Questions

The following are a few questions that remained unanswered and would be useful for future work relating to the project:

1) **How well would a combined approach work on the data?**
Having investigated such a range of noise-robustness methods, it is evident that they are not only effective but, being different, they could be combined to try to get as accurate a result as possible. This should be done and tested as a future step. A combined approach could be done in a few ways (and in this future work, should be experimented in a few ways), such as using Ensemble Methods such as a Voting Classifier, where one runs the test data on a few models, and the option for classification with the most ”votes” wins.
One could also use a chained approach such as first using the Wavelet Denoising, then applying the Autoencoder, and then running the CNN on the encoded data, but with L2 Regularisation. A few of these different chains could be experimented with.
2) **What Representational Model will best suit Noise-Robustness?**
The two best methods included Wavelet Denoising and the Autoencoder, which both make use of a representational model to transform the data before feeding it into a network in such a way as to minimize the effect of the noise. Seeing that both of these make use of latent representations, an investigation specifically focused on finding the best latent representation of the images without just removing the noise would be particularly interesting.

Based on the results, we can see that an Image Noise Robust Convolutional Neural Network is not only possible but can be quite accurate with only access to Noisy Data. Additionally, this was done using latent representations of the data, where it is transformed into another domain, possibly edited in some way, and the result is fed into the Neural Network, which provides an opportunity for further investigation.

REFERENCES

[1] S. Jaeger, S. Candemir, S. Antani, Y.-X. J. Wáng, P.-X. Lu, and G. Thoma, “Two public chest X-ray datasets for computer-aided screening of pulmonary diseases,‘’ *Quantitative imaging in medicine and surgery*, vol. 4, no. 6, pp. 475, 2014.

[2] Pisit Janthawee, “Tuberculosis Detection CNNs with LIME,‘’ [Online]. Available: https://www.kaggle.com/code/pisitjanthawee13/tuberculosis-detection-cnns-with-lime.

[3] S. Lee, M. S. Lee, and M. G. Kang, “Poisson-Gaussian Noise Analysis and Estimation for Low-Dose X-ray Images in the NSCT Domain,‘’ *Sensors*, vol. 18, no. 4, pp. 1019, 2018, doi: 10.3390/s18041019.

[4] N. Bähler, M. E. Helou, É. Objois, K. Okumuş and S. Süsstrunk, “PoGaIN: Poisson-Gaussian Image Noise Modeling From Paired Samples,‘’ *IEEE Signal Processing Letters*, vol. 29, pp. 2602-2606, 2022, doi: 10.1109/LSP.2022.3227522.

[5] A. Buades, B. Coll, and J.-M. Morel, “Non-Local Means Denoising,‘’ *Image Processing On Line*, vol. 1, pp. 208-212, 2011, doi: 10.5201/ipol.2011.bcm_nlm.

[6] A. Halidou, Y. Mohamadou, A. A. A. Ari, et al., “Review of wavelet denoising algorithms,‘’ *Multimed Tools Appl*, vol. 82, pp. 41539-41569, 2023, doi: 10.1007/s11042-023-15127-0.