UNIVERSITY OF CALIFORNIA
Los Angeles

Augmenting MRI Classified with Synthetic Images Created via Generative Models

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Daniel Kwon

2024

ABSTRACT OF THE THESIS

Augmenting MRI Classified with Synthetic Images
Created via Generative Models

by

Daniel Kwon
Master of Science in Statistics
University of California, Los Angeles, 2024
Professor Yingnian Wu, Chair

WIP - Check back later

The thesis of Daniel Kwon is approved.

Frederic R Paik Schoenberg

Yingnian Wu, Committee Chair

University of California, Los Angeles

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

With the performance of state-of-the-art generative models improving rapidly, synthetic images can be produced with ease using simple text or image prompts. Image generation models such as Dall-E or Stable Diffusion are able to create images with sufficient quality that the margin of difference between real and synthetic images is becoming increasingly thin. As the level of effort in generating synthetic images decreases and the quality of these images increases, the potential to use synthetic images as a means to augment image datasets becomes increasingly viable—especially in fields where gathering images is constrained by costs or other resources.

In the field of medical imaging, where image datasets often require specialized equipment and subject matter experts to capture and label images, gathering enough data to sufficiently train a classification model can be both expensive and time-consuming. For example, with the cost of magnetic resonance imaging (MRI) ranging from $1,600 to $8,400 in the United States, a dataset consisting of a few hundred images can exceed $1 million; image classification models often require thousands of photos.

One way computer vision models have historically attempted to remedy insufficient training datasets has been to employ traditional image augmentation techniques, which involve applying transformations such as flipping or blurring an original image to produce additional images for training. However, such transformations must be applied carefully in order to not lose the fidelity needed to make accurate diagnostic predictions. Transformations that alter the nature of the images can potentially lead to a degradation in model performance.

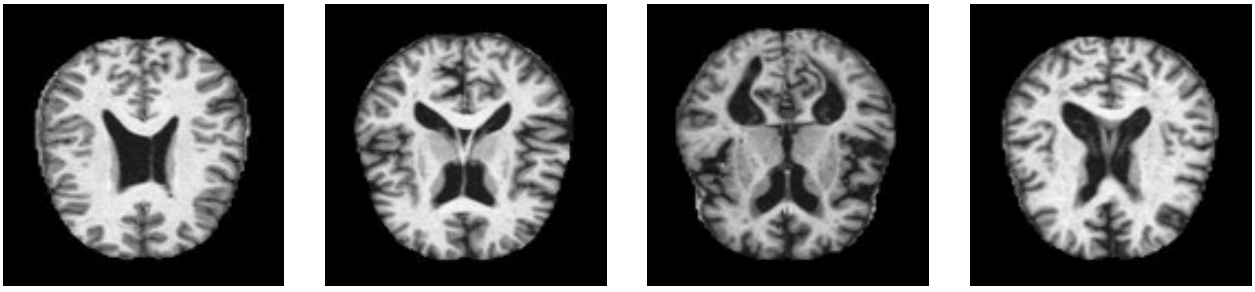
Augmenting image datasets with high quality synthetic images may allow for significant cost-saving while maintaining model performance, and previous research has found that synthetic images may play a complimentary role to image augmentation. The goal of this paper is to train image classification models on a MRI dataset to identify the presence of varying levels of dementia and investigate the effect of different image augmentation policies on model performance as well as compare their performance against an augmentation policy that generates synthetic images.

CHAPTER 2

Dataset

2.1 Alzheimer MRI Preprocessed Dataset

For the purposes of this paper, we use publicly available MRI images of patients with varying levels of dementia, labeled as *not demented*, *very mildly demented*, *mildly demented*, and *moderately demented*. These images were downloaded via the datasets module provided and maintained by Hugging Face. All images are in black and white and have been pre-processed to 128x128 resolution.



(a) Not Demented (b) Very Mildly Demented (c) Mildly Demented (d) Moderately Demented

Figure 2.1: Examples of real MRI images

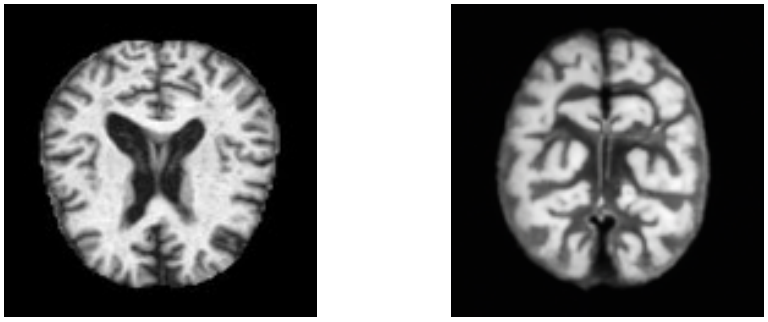
Not Demented	Very Mildly Demented	Mildly Demented	Moderately Demented
2566	1781	724	49

Table 2.1: Count of each class in training data

2.2 Synthetic Images

This paper utilizes OpenAI's Dall-E-2 to produce synthetic images. To generate synthetic images, an original image is supplied as a source and the generative models are prompted to creating a similar image.

In order to generate images using Dall-E-2 we utilize OpenAI's image variation API endpoint, which returns a variation of a given image.



(a) Real Image (b) Synthetic Image

Figure 2.2: Results of Dall-E-2's image variation generation

Dall-E-2 reliably generates images that are similar to the real MRIs that are provided. Given that dementia is often physically tied to brain atrophy in certain areas, the synthetic images produced by Dall-E-2 generally reproduce the ridges, folds, and cavities of the source images as well.

CHAPTER 3

Exploration using Grad-CAM

In order to better understand what parts of an image classification models deem to be more "important" in regards to predicting the presence of dementia, we employ Gradient-weighted Class Activation Mapping, or Grad-CAM, as a way to represent what a convolutional neural network "sees" when classifying both real and synthetic images. Grad-CAM is a technique that maps the gradients of a final convolutional layer in regards to a specific class prediction to product a heat map. The result is a visually intuitive way of illustrating which portions of an image contribute most to a given classification [source here].

3.1 Grad-CAM details

1. First, calculate the gradients of the model's output in the final convolutional layer, with respect to the feature maps. Assuming y_c is the score for class c (before softmax) and δA^k is the feature map activation of the kth layer, compute $\frac{\delta y_c}{\delta A^k}$.

2. Next, calculate the global average pooling of the feature map. Global average pooling is a pooling operation designed to generate one feature map for each corresponding category of the classification task in the last [convolutional] layer and then take the average of each feature map [source here].

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta y_c}{\delta A_{ij}^k}$$

Z here represents spatial dimensions of the feature map—in this case, its height and

width. By dividing by Z, we obtain the average of gradients over all spatial locations. $\frac{\delta y_c}{\delta A_{ij}^k}$ is the gradient for class c with respect to the activation at A_{ij}^k, or spatial location (i,j) on activation layer k.

3. Lastly, we take the resulting importance weights in a_k^c and combine them with A^k to calculate a weighted combination of all forward activation maps. We then apply the ReLU activation function to the resulting combination in order to take the positive values only. This ensures that we are only looking parts of the image that are positively correlated with a given class.

$$L_{Grad-CAM}^c = ReLU(\sum_k \alpha_k^c A^k)$$

Notice that $L_{Grad-CAM}^c$ will have the dimensions of the final convolutional layer and therefore is likely to be smaller than the original input image. In that case, we simply upsample the result to the dimensions of our original image in order to overlay the two.

3.2 Grad-CAM results

When Grad-CAM is

CHAPTER 4

Background on Training Deep Learning Models

4.0.1 Overview of Neural Networks

While deep learning architectures can span many layers and can incorporate many different mechanisms, at its core all deep learning models are neural networks. Training neural networks comprise of a few key steps.

First, training data is input into a neural network and passed through as-is in what is known as a "forward pass." In a fully connected neural network, every individual neuron in a layer of the neural network is connected to each neuron in the preceding layer

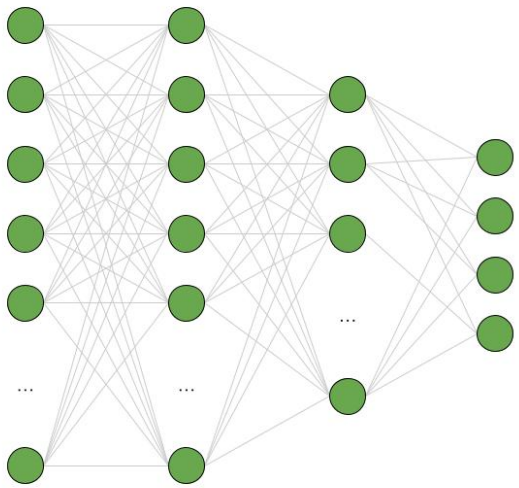


Figure 4.1: An illustration of a typical fully-connected neural network

Each of these connections has a weight that stores how strong of a connection each

preceding node has to the current node. A bias term is also present, which represents whether a neuron is general activated or not. Often, an initialized neural network will comprise of randomized weights and biases whereas a pre-trained model will have the weights and biases already defined from previous training. When training a neural network, the weights and biases are what are being adjusted in order to minimize loss. The formulation of a single neuron with n nodes in the previous layer is below:

$$\sigma(w_0a_0 + w_0a_0 + w_0a_0 + ... + +w_{n-1}a_{n-1} + b)$$

After the forward pass, backpropagation occurs in which the gradient (the vector of partial derivatives) of the loss function with respect to each weight and bias is calculated via chain rule. Each gradient value represents the magnitude and direction of the change in our loss function given a change to that particular weight or bias. Because backpropagation uses the chain rule to propagate the loss backward from the output layer to the input layer, the process of finding the gradient of a loss function remains the same no matter how many layers are in the network or how many neurons are in each layer.

Once the gradient is calculated, the gradient descent algorithm is used to update the weights and biases in order to minimize loss. However, applying the gradient descent on an entire training dataset in a single batch is computationally expensive. Instead, gradient descent is often applied to subsets of the training data, called mini-batches. Each training iteration, or epoch, will then take a mini-batch to apply the forward and backward passes on to to updates its weights and biases. The result is an accurate approximation of the gradient of the loss function while significantly decreasing computational expenses [source].

4.0.2 Loss Function

The loss function used throughout this paper when measuring model performance is cross entropy loss. Cross entropy loss is defined as:

$$L = -\sum_{c=1}^{C} y_c log(p_c)$$

where C = the number of classes, y_c is the true label for class, and p_c is the predicted probability for class c. Cross entropy loss therefore compares the predicted probabilities with the actual labels and penalizes more when the predicted probabilities for a correct class is low [source].

Cross entropy loss is better suited for measuring the performance of classification models when multiple classes are involved than traditional measures of error, such as the sum of residuals, as it is more sensitive to predictions that are "more" incorrect. In an image classification problem, a model may mislabel a given image, but the cross entropy loss will be lower if the predicted probability for the correct class is higher, even if the model did not ultimately end up labeling correctly. Contrast that with the sum of residuals, which only views predictions as completely correct or completely incorrect and fails distinguish between the degrees of how right or wrong a prediction can be [source].

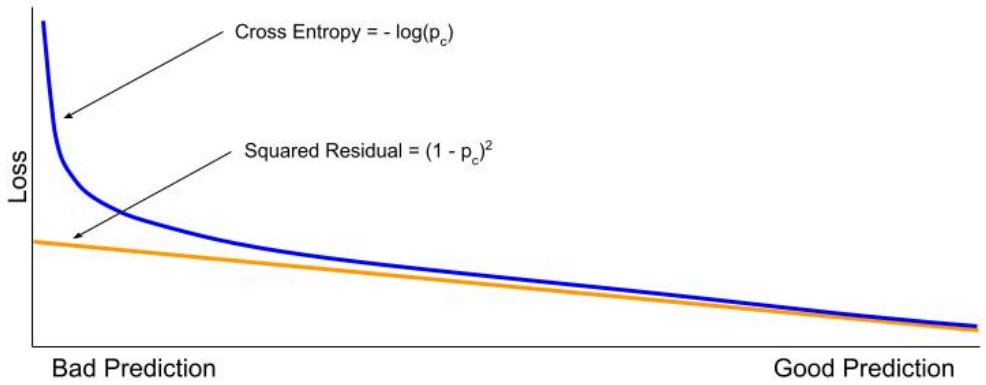


Figure 4.2: Illustration of Cross Entropy Loss vs Squared Residual

Take for example, when the true label is 1 and the predicted class weight y_c is 0.0001—

making our model prediction very bad. The squared residual would be $(1-0.0001)^2 = 0.9998$ while our cross entropy loss would be more punitive at $-log(0.0001) = 4$. Cross entropy loss also has a much larger gradient for very bad predictions, allowing our model to more quickly learn to avoid bad predictions.

4.0.3 Stochastic Optimization

All models mentioned in this paper are trained using the Adaptive Mom

4.1 Overview of Model Architectures

We test two different architectures for our image classification models—a basic Convolutional Neural Network (CNN) and a Vision Transformer (ViT)—in order to gain an understanding as to how both convolution-based and attention-based architectures react to the introduction to synthetic data in training. While the details of either architectures are outside the scope of this paper, a brief over of both are provided below.

4.1.1 Convolutional Neural Networks

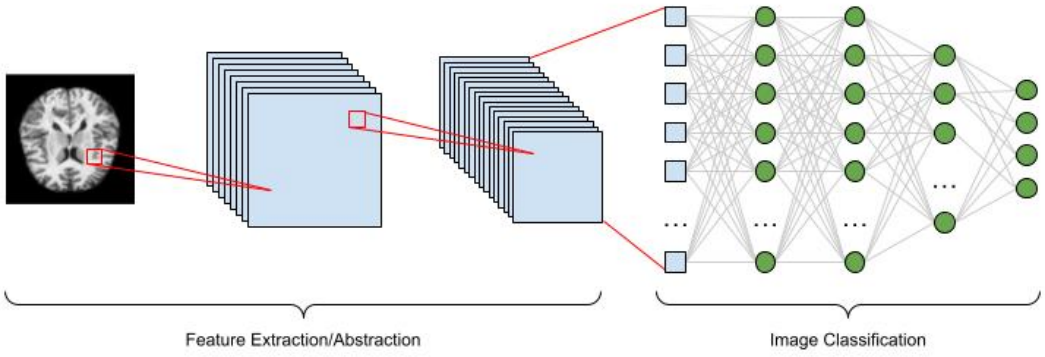


Figure 4.3: Typical CNN Architecture

A CNN a deep learning architecture that is comprised of convolutional layers—which abstract an image to a feature map, pooling layers—which reduce the dimensions of data by combining the outputs of adjacent layers via downsampling, and fully connected layers—which are neural networks that take the final activations from the convolutional and pooling layers to generate class weights.

For the purposes of this paper, we utilize a CNN with 2 convolutional layers, each paired with a ReLU activation function and a max pooling layer. After all feature extraction layers are complete, the final activation layer is fed into a fully connected neural network with 2 hidden layers and an output layer that predicts probabilities weights for each of our four

classes.

4.1.2 Vision Transformers

WIP

4.2 Methodology

4.2.1 Traditional Image Augmentation Policies

4.2.2 Synthetic Image Generation Policy

Given a desired proportion of original training images to utilize (X) and the desired proportion of synthetic images to augment the training data (Y), we combine both synthetic and original images to train our image classification model.

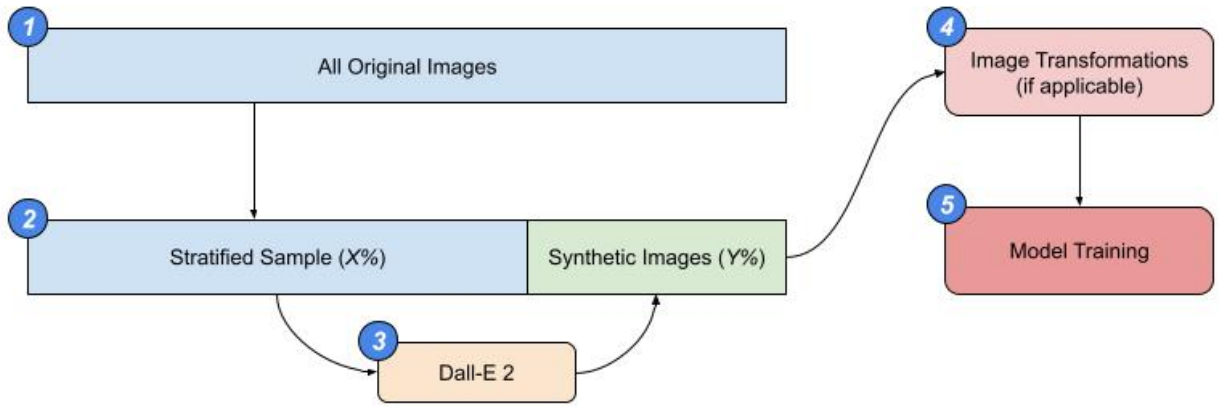


Figure 4.4: Enter Caption

The figure above illustrates one iteration of our image generation policy in order to generate our mixed training set (i.e., containing both original and synthetic images)

1. First, we take a random stratified sample of our original images
2. The stratified sample is sent to Dall-E 2 via OpenAI's API in order to generate an image variation
3. Image transformations are applied to our entire mixed training set (if applicable)
4. Mixed training set, post transformations/augmentation, is used to train the image classification model

5. Steps 1-4 are repeated N times in order to account for the stochastic nature of Dall-E's responses as well as to bootstrap a distribution of performance–measured by cross entropy loss

4.3 Results

4.3.1 CNN Results

When applied to a basic convolutional neural network, augmentation via synthetic images provides the most benefit when the dataset of original images is limited. When training our CNN on a 30% stratified sample, the median validation loss from 100 simulations was 0.219863, compared to a loss of 0.176023 when training on the full dataset—resulting in an increase in our loss by approximately 25%.

However, model performance improves when augmenting the 30% stratified training sample with synthetic images so that the resulting training set contains the same number of images as the full training data set. The resulting model trained on the mix of 30% real and 70% synthetic images has a cross-entropy loss of 0.174202—a decrease of over 20% compared to the model trained on only the 30% of real training images and matching the performance of model trained on the entirety of the training images.

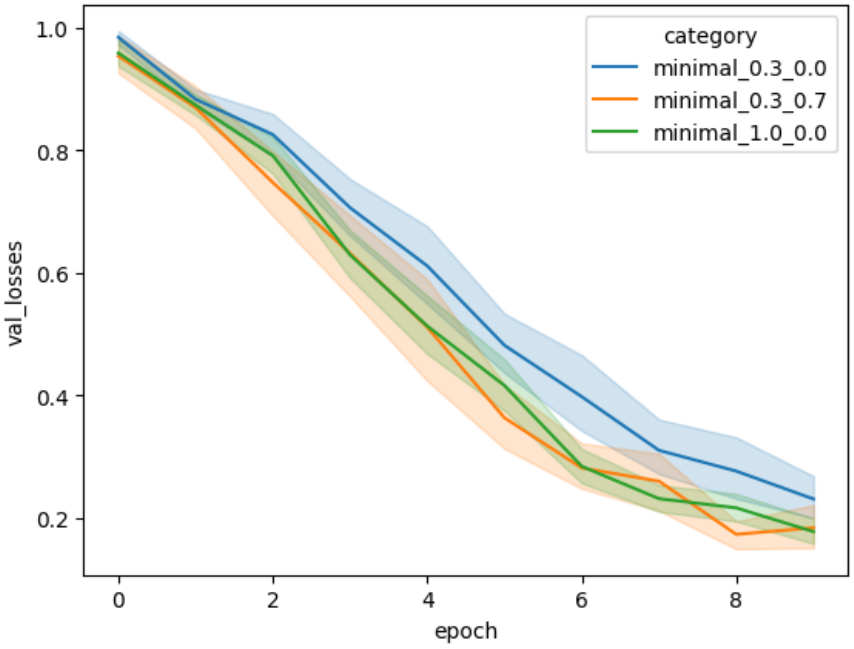


Figure 4.5: Cross Entropy Loss - Model trained on 30% Real, 70% synthetic

Moreover, CNNs trained on synthetic data outperformed CNNs trained on 30% of available training images and paired with traditional image augmentation policies. This is likely due to the fact that the image augmentation policies may alter the nature of the image enough to degrade performance of our classification model. This is not to say that traditional image augmentation methods do not have a role in medical imaging; only that traditional image segmentation policies, when misapplied, can degrade model performance significantly and therefore likely require subject matter experts to evaluate whether image augmentation is necessary. In fact, other studies have found that image augmentation generally improves the performance of image classification models used for brain imaging.

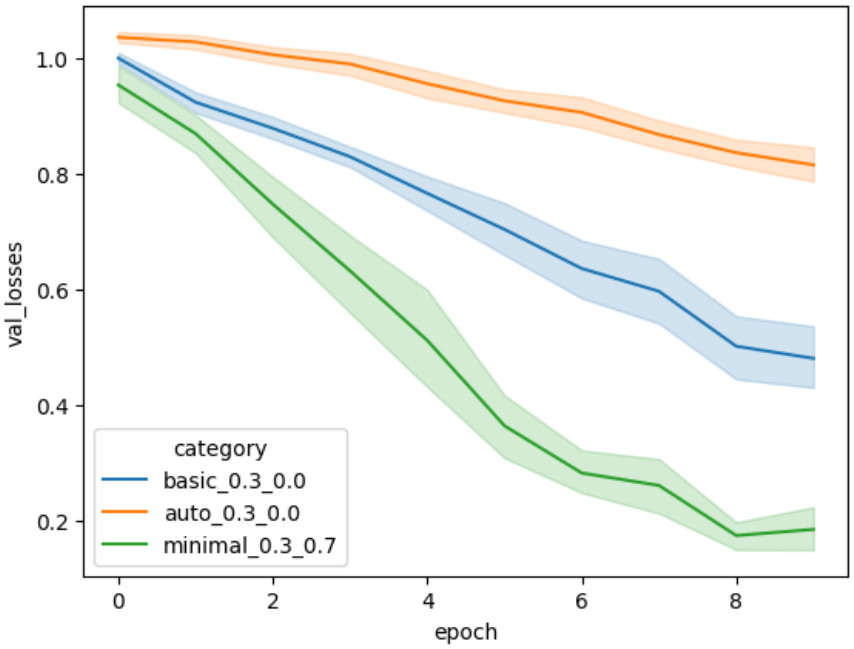


Figure 4.6: Comparison of resulting cross entropy loss across models using different image augmentation policies, all trained on 30% of available images

However, not all training datasets benefit from synthetic image augmentation. CNNs trained on 40-80% of all available training data resulted in virtually no improvement, with the 80% model resulting in a degradation in loss. This alludes to the possibility that syn-

thetic data augmentation may be more beneficial as original training data becomes more constrained.

CHAPTER 5

Conclusion

CHAPTER 6

Additional Considerations