Measuring COVID-19 Progression in Lung CT Images Using Deep Generative Models

Judah Zammit
7839359
zammitj3@myumanitoba.ca
Group 13

Abstract—**Coronavirus disease 2019 (COVID-19) affects those it infects with greatly varying levels of severity. Lung computed-tomography (CT) images from infected patients have successfully been shown to be useful for the prediction of severe outcomes. However, lung CT images from *healthy* patients have not been explored. This study aimed at developing a model that can predict future lung CT images, allowing for more effective prognosis, from a current lung CT image, whether that be from a healthy or unhealthy patient. We do this by proposing a novel generative model based on the ladder variational autoencoder [1], called the U-shared variational autoencoder (U-SVAE). This model utilizes deep convolutional neural networks highly suited for lung CT images and a decoder with skip connection for increased levels of detail. Using the U-SVAE we create a model for predicting future lung CT images, called COVID-aging (CAGE). We show the U-SVAE is effective at generative modeling of lung CT images but that there is an issue with the CAGE model that we refer to as the tourist crisis.**

I. Introduction

The Coronavirus Disease 2019 (COVID-19) affects victims in drastically different ways. The severity of the disease ranges from barely noticeable, to life-threatening. Therefore, it is crucial that we develop accurate methods for identifying individuals that are highly susceptible to severe COVID-19 morbidity and mortality outcomes.

Currently, our primary means of screening for susceptibility is by identifying demographics, such as age and gender, that exhibit higher then normal morbidity and mortality rates. However, the variance of severe outcomes within demographics is still significant. Therefore, the use of clinical features, such as number of underlying diseases, can help to identify susceptible individuals.

Due to the fact that COVID-19 is a respiratory disease, we hypothesis that lung computed-tomography (CT) images can provide valuable insight for high accuracy susceptibility screening.

We seek to accomplish this by first predicting a patient's lung CT image at all stages of the disease with only the patient's healthy lung CT image. Though modern deep-learning techniques provide many tools for accomplishing this, the current COVID-19 CT image datasets do not. This is because, the vast majority of datasets provide a patients CT image at only one stage of the disease. This renders the majority of deep-learning techniques incapable of solving this problem.

Deep generative models excel at inferring realistic missing data from limited observed data. Thus they have the potential to be able to cope with data where all but one of a patient's lung CT images is unobserved. As we are interested in doing inference on a patient's lung CT images and not in the generation of synthesized lung CT images, we use a variational inference based deep generative model, as opposed to an adversarial based deep generative model. The former type of model excels at the task of interest and additionally provides meaningful and feature rich latent variable that can be interpreted as deep imaging phenotypes.

This task is quite similar to that of deep generative face aging. That is, taking the picture of a person's face and predicting what they would look like if they were older or younger. For our task, instead of faces we use lung CT images and instead of age we use the number of days the patient has had COVID-19. We refer to this as the COVID-age and the task as COVID-aging.

In summary, our contributions are as follows:

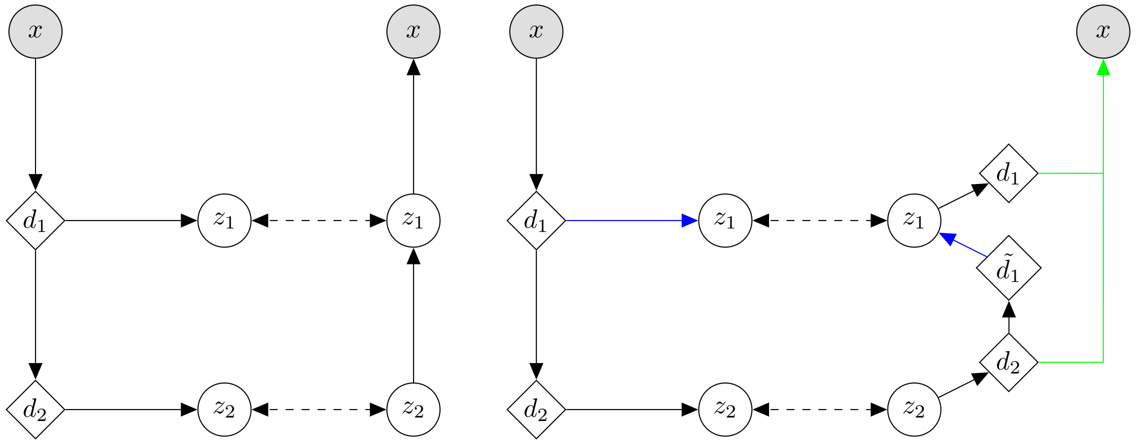
- We develop a novel deep generative model, called the U-Shared Variational Autoencoder (*U-SVAE*), that can synthesis highly realistic lung CT images
- Adapting the *U-SVAE* model, we develop a novel model for COVID-aging called *CAGE*
- We identify an issue with *CAGE* which we call *the tourist crisis*

II. Related Work

The primary means of assessing a healthy patient's risk of severe COVID-19 outcomes is to identify risk factors that are statistically correlated with these outcomes. The Centers for Disease Control and Prevention[1] (CDC) acknowledges many such risk factors. These include age, certain medical conditions (such as cancer, heart conditions and diabetes), certain health conditions (such as smoking and severe obesity) and demographic characteristics (such as racial and ethnic minority groups, people experiencing homelessness and people living in rural communities).

Zheng et al. [2] provide a thorough meta-analysis of thirteen studies investigating COVID-19 risk factors as well as identifying clinical features that may indicate a more severe outcome. While the risk factors found are much the same as the ones acknowledged by the CDC, the clinical features found are worthy of note. These include many laboratory indicators

[1] https://www.cdc.gov/coronavirus/2019-ncov/need-extra-precautions/index.html



(a) Ladder Variational Autoencoder: Dashed lines indicate a combination of parameters

(b) U-SVAE: Shared weights are depicted with blue arrows; U-Net decoder is shown in green

Fig. 1: A comparison between the LVAE and our U-SVAE

such as white blood cells, aspartate aminotranferase, creatinine and others. Notably, they did not investigate any features related to CT imaging.

Though effective at quickly identifying high-risk populations, there is still significant variation of the susceptibility of individuals within these populations. Because COVID-19 is a respiratory disease, we hypothesis that CT imaging features could provide an additional means of assessing risk of severe outcomes, both in healthy and unhealthy patients.

Li et al. [3] extracted many features from the CT images of unhealthy patients and found significant correlation between COVID-19 severity and several of these features. Despite the fact that this approach can only be used on unhealthy patients, this shows that CT images do indeed have promising utility for healthy patients as well.

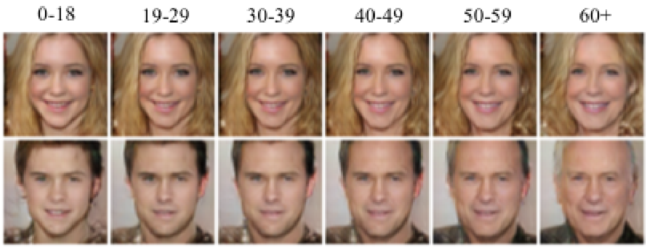


Fig. 2: An example of face aging from Antipov et. al. [4]

This paper seeks to use CT images from both healthy and unhealthy patients to predict what their lungs would look like in later stages of the disease. This task is similar to face aging. Face aging models use an image of person's face to output what they would look like if they were to be a certain age. This is depicted in figure 2. In our context the age is analogous to the number of days the patient has had COVID-19, with age zero indicating that the patient is healthy, and the image is analogous to the patient's CT image.

Many face aging models use one or many conditional generative adversarial networks (GAN) [5], [6], [7], [8]. Broadly speaking, these models take a random noise vector and the age and produce a face that is realistic. As GAN's cannot naturally infer the value for this noise vector from an image, modification have to be made to allow for this. These models excel at producing highly realistic images.

Other face aging modles use a hybrid approach between variational autoencoders (VAE) and advesearial learning [9], [10], [11]. This allows for effortless inference, while still maintaining realistic images. However, these models tend to fall behind the GAN based models in regards to realism.

These face aging models use some form of a generative model. The Ladder Variational Autoencoder (LVAE) [1] is a state of art generative model that models the data with a ladder-shaped hierarchy of latent variables and employs several novel techniques to allow for its training. The model proposed in their work is effective on toy data sets but is not intended for use on real world data such as CT images.

The Generative Adversarial Network (GAN) [12] and all of its successors, are the model of choice when highly realistic images are desired. However, they are extremely difficult to train and have difficulty inferring latent variables from images. This makes them unsuitable, at least on their own, for COVID-aging.

To develop a suitable generative model for CT images, several sophisticated image processing models are needed. The MobileNetV2 [13] is a widely successful image classification deep convolutional network and is widely used as a backbone network for many other applications. It is very efficient, while still achieving excellent performance.

The U-Net [14] is a semantic segmentation network, that is used frequently for medical image applications. It allows for highly detailed semantic segmentation predictions by using an encoder-decoder style architecture with skip connections from the encoder to the decoder.

III. U-Shared Variational Autoencoder (U-SVAE)

In this section we will introduce the theory, implementation and optimization of the *U-SVAE*. The code for this model can be found at https://github.com/JudahZammit/u-svae.

A. Problem Space

Suppose we have a data set, $D = \{x^{(i)}\}_{i=0}^{N}$, of N images. Assuming that these images are i.i.d. samples from some ground-truth distribution, $p(x)$, we wish to approximate that ground truth distribution. This allows us to sample from our approximation, synthesizing new images.

B. Ladder Variational Autoencoder

The Ladder Variational Autoencoder (LVAE) [1] is a recently proposed model that has been shown to be highly effective at modeling such distributions. Here we will briefly summarize their work and discuss some potential issues.

The LVAE assumes that the data is generated in a hierarchical sampling process. Specifically, it assumes that, to generate an image, we first take a sample from a unit-gaussian distributed-latent variable, z_2. We then apply some function to that sample, outputting the parameters to a diagonal-gaussian distributed-latent variable, z_1. This will repeat for how many levels we desire, with the last output distribution being a distribution (their work used a bernoulli distribution) for each pixel in the image. This allows them to assume independence between the pixels when conditioned on the latent variables. This model is denoted by p_θ.

The LVAE uses variational inference to learn both the model p_θ and an approximate posterior to p_θ, q_ϕ. In previous work, q_ϕ will infer the value for z_1 from x and z_2 from z_1. The LVAE differs from this in that their q_ϕ completes a deterministic down pass and then each z is inferred from the intermediate layers of this down pass. The dependencies between latent variables are recovered by combining the inferred distributions' parameters for the latent variables with the generative model's predicted distributions' parameters. This is depicted in figure 1a.

C. U-Net Style Decoder and Shared Parameters

In the research area of semantic segmentation, it is well known that simply applying a series of deconvolutional layers, with dimension preserving layers in between, results in a great loss of detail in the final output. This is exactly what the LVAE's generative network, also known as its decoder, is doing and there is certainly as much detail, if not more, in an image as there is in a segmentation mask (the output of semantic segmentation networks).

Therefore, we attempt to improve on the LVAE by replacing its decoder with the decoder of the widely succesful semantic segmentation network, the U-Net [14]. We do this by first applying an series of recurrent convolution layers to the samples of each latent variable to obtain a deterministic expansion for each latent variables. These are quite similar to the deterministic variables in the inference network, q_ϕ, and so are denoted similarly. We then use the each of these deterministic expansions as input to the U-Net's decoder, outputting the final image.

Now, to find z_1 given z_2, we apply a deconvolutional layer to d_2 to get d_1 and then apply many convolutional layers to d_1 to get z_1. This updated model is depicted in figure 1b.

We note that, in both p_θ and q_ϕ, we have a mapping between d_n and z_n. We hypothesis that this mapping serves the exact same purpose in both and that having both share weights would increase performance. With this final change, we arrive at the U-Shared Variational Autoencoder.

D. Model Specifications

Up to this point, we have been speaking about the mapping between variables in quite vague terms. Here we will detail the exact functional mappings used in this work. As we are processing images, we exclusively use convolutional neural networks for our model.

We will now describe the building blocks of our model. Each of these building blocks comes in two forms, *Low* and *High*. The *Low* version is used when the number of input filters is less then or equal to 64, and the *High* version is used when the number of input filters is greater then 64. We denote each layer of the block as *Name (change in output)*. For example, 2x2 Conv2D (1,1,2) denotes a 2x2 2d convolution that doubles the filters but keeps the image dimensions unchanged. The building blocks are as follows:

- *Transpose)* This block is responsible for doubling the dimensions of a variable. For example, taking a 2x2 variable and outputting a 4x4 variable. We refer to Batch Normaliaziton [15] followed by the ReLU [16] activation as BN-ReLU.
  Low) 4x4 Conv2DTranpose (2,2,1) → BN-ReLU (1,1,1)
  High) 1x1 Conv2D (1,1,1/4) → BN-ReLU (1,1,1) → 4x4 Conv2DTranpose (2,2,1) → BN-ReLU (1,1,1) → 1x1 Conv2D (1,1,4) → BN-ReLU (1,1,1)
- *ResBlock)* This is a residual convolutional block that keeps the dimensions and filters the same. The final output of the block is added to the input. This is referred to as AddInput.
  Low) BN-ReLU (1,1,1) → 3x3 Conv2D (1,1,1) → BN-ReLU (1,1,1) → AddInput (1,1,1)
  High) BN-ReLU (1,1,1) → 1x1 Conv2D (1,1,1/4) → BN-ReLU (1,1,1) → 3x3 Conv2D (1,1,1) → BN-ReLU (1,1,1) → 1x1 Conv2D (1,1,4) → AddInput
- *Smooth)* This is a block that chains together 3 ResBlocks.
  Low) 3x3 Conv2D (1,1,1) → BN-ReLU (1,1,1) → 3x3 Conv2D (1,1,1)→ LowResBlock (1,1,1) → LowResBlock (1,1,1) → LowResBlock (1,1,1) → BN-ReLU (1,1,1)
  High) 1x1 Conv2D (1,1,1/4) → BN-ReLU (1,1,1) → 3x3 Conv2D (1,1,1) → BN-ReLU (1,1,1) → 1x1 Conv2D (1,1,4) → HighResBlock (1,1,1) → HighResBlock (1,1,1) → HighResBlock (1,1,1) → BN-ReLU (1,1,1)
- *LightSmooth)* This block functions the same as Smooth, however only uses one ResBlock.
- *SkelaSmooth)* This block functions the same as Smooth however uses no ResBlocks.
- *Up)* This block increases the filters of the input by a multiplicity, f.
  Low) 3x3 Conv2D (1,1,f) → BN-ReLU (1,1,1)
  High) 1x1 Conv2D(1,1,f) → BN-ReLU (1,1,1)
- *Down)* This block decreases the filter of the input by a multiplicity of f.

TABLE I: The dimensionality of the five latent variables. Level 0 denotes the input image x

Level	z	d
0	(64,64,1)	NA
1	(32,32,1)	(32,32,32)
2	(16,16,1)	(16,16,64)
3	(8,8,1)	(8,8,128)
4	(4,4,1)	(4,4,256)
5	(2,2,1)	(2,2,512)

Low) 3x3 Conv2D (1,1,1/f) → BN-ReLU (1,1,1)

High) 1x1 Conv2D (1,1,1/f) → BN-ReLU (1,1,1) → 3x3 Conv2D (1,1,1) → BN-ReLU (1,1,1)

We may now discuss the actual mappings. First, our model has five layers of latent variables, opposed to the two depicted in figure 1b. The dimensionality of these latent variables and their deterministic expansion is shown in table I.

We use the intermediate and output layers of *MobileNetV2* [13] with the image, x, as input to obtain $d_1, d_2, ...d_5$. Concretely, $MobileNet(x) = (d_1, d_2, ..., d_5)$.

The mapping from d_i to z_i is $FilterCrush(d_i) = (\mu_i(l), \sigma_i(l))$, where $l = LightSmooth(Down(d_i))$ and $\mu_i(l) = \sigma_i(l) = 3x3Conv2D(LightSmooth(Down(l)))$.

The mapping from z_i to d_i is $FilterExpand(z_i) = d_i$. $FilterExpand$ is composed of *Up* (1,1,f/4) → *LightSmooth* (1,1,f/4) → *Up* (1,1,f/2) → *LightSmooth* (1,1,f/2) → *Smooth* (1,1,f), where f is the number of filters at level i.

The mapping from d_i to $d_{\widetilde{i-1}}$ is $Infer(d_i) = LightSmooth(Transpose(d_i)) = d_{\widetilde{i-1}}$

The mapping from $d_1, d_2, ..., d_5$ to x is the decoder from the U-Net [14]. This is denoted by $Decoder(d_1, d_2, ...d_5) = x$

IV. COVID-Aging (CAGE) Model

In this section we will describe our proposed CAGE model in detail. The code for this model can be found at https://github.com/JudahZammit/cage.

A. Problem Space

Suppose we have a data set, $D = \{x_0^{(i)}\}_{i=0}^{N_0} \cup \{x_1^{(i)}\}_{i=0}^{N_1} \cup ... \cup \{x_M^{(i)}\}_{i=0}^{N_M}$, where $x_j^{(i)}$ denotes this i^{th} CT image from the group of patients that have had COVID-19 for j days. For the reminder of this section, we will assume that M is three, to allow for brevity in our notation. We will also refer to all CT images from patients on the j^{th} day as the j^{th} COVID-age group.

We wish to infer $(x_0, x_1, x_2, x_3)^{(i)}$ from $x_j^{(i)}$. That is, we wish to infer the CT images of a specific patient for all future and past days given only the CT image for a single day.

B. Graphical Model

Consider a naive approach to this problem. Suppose we assume x_{j+1} can be determined from x_j. That is, given the CT image on a certain day, we can always infer the next day. Furthermore, assume that x_0 can be inferred from x_3. That is, given the CT image from the latest day of the disease progression, we can infer the healthy CT image. This is reasonable as, on the latest day, the CT image is close to recovered, and so, close to the healthy CT image. This gives us the graphical model depicted in figure 3.

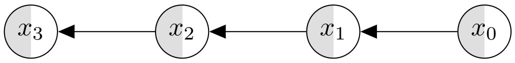


Fig. 3: Naive Graphical Model: The half-filled circles indicate that these variables are partially observed; specifically, exactly one will be observed for any given patient

We can approximate this model by assuming the mapping between variables is parameterized by a variable θ. That is, $p_\theta(x_{j+1}|x_j)$. We can then learn the values for θ that maximizes $p_\theta(x_0, x_1, x_2, x_3)$. This approach has a fatal flaw. It assumes that all the variables, x_0, x_1, x_2, x_3, are observed for every patient. In fact, only one of these variables for each patient is observed with the rest unobserved.

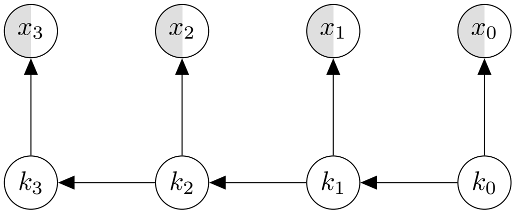


Fig. 4: Proposed Graphical Model: Unfilled circles denote latent variables

To allow for inference in such a constrained setting, we move away from inferring x_{j+1} from x_j and instead model each COVID-age group with a latent variable and do inference on these instead. That is, we assume that each variable x_j can be determined by a latent variable k_j, and that k_{j+1} can be determined by k_j. This is depicted in figure 4.

Once again, we assume that the mappings between variables are parameterized by θ. Concretely, $p_\theta(k_{j+1}|k_j)$ and $p_\theta(x_j|k_j)$.

Such a model gives the distinct advantage that, given an appropriate inference model, we can infer values for all latent variables, even those whose associated CT image is unobserved. This allows us to learn the mapping between these latent variables even when only one of the variables, x_0, x_1, x_2, x_3, is observed.

Qualitatively, this model should infer values for the unobserved variables that are, if not correct, at least realistic, allowing for realistic mappings between the latent variables to be learned.

C. Adaptation of the U-SVAE Model for the CAGE Model

We took pains in the last section to develop an effective graphical model for modeling a variable such as x_j. We will use the U-SVAE model here to model x_j with a latent variable k_j. Note that, here, k_j denotes the entire generative model for the U-SVAE model, not just one variable.

Recall that each variable k_j has a deterministic expansion d_j. We assume that d_{j+1} can be determined from d_j and that d_0 can be determined from d_3.

In addition, the decoders for each of the U-SVAE models share weights. This is depicted in figure 5.

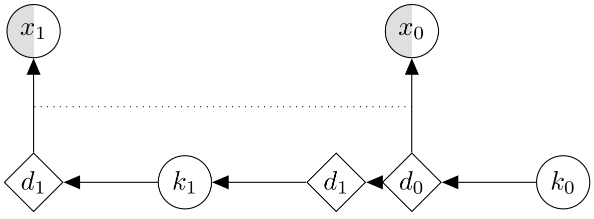


Fig. 5: Updated Graphical Model: Diamonds denote deterministic mappings; dotted lines denote shared weights. We only show COVID-age groups 0 and 1 for brevity

D. Optimization of p_θ)

We wish to find the value for θ that maximizes the log likelihood of the data under $p_\theta(x,k)$. Concretely, we wish to solve

$$\begin{aligned} &max_\theta \textstyle\sum_D log p_\theta(x_j^{(i)}) \\ &\quad = \textstyle\sum_D log \int_{k_0} \dots \int_{k_3} p_\theta(x_j^{(i)}, k_0, ..., k_3). \end{aligned} \tag{1}$$

The existence of the latent variables, and, by extension, the need to integrate over them, makes this task completely intractable. Even if we were able to calculate this, we would still be left with the task of inferring latent variables from $x_j^{(i)}$. That is, calculating $p_\theta(k_0,...,k_3|x_j^{(i)})$. This, also, is completely intractable.

We instead optimize a variational lower bound on the log likelihood of $p_\theta(x,k)$. Concretely, we optimize,

$$\begin{aligned} &\textstyle\sum_D log p_\theta(x_j^{(i)}) \\ &\geq \textstyle\sum_D E_{q_\phi(k_0,...,k_3|x_j^{(i)})} \left[log \frac{p_\theta(x_j^{(i)},k_0,...,k_3)}{q_\phi(k_0,...,k_3|x_j^{(i)})} \right] \end{aligned} \tag{2}$$

Though q_ϕ can be any function of the latent variables, this lower bound is exactly equal to the true log likelihood when q_ϕ is equal to $p'_\theta s$ posterior, $p_\theta(k_0,...,k_3|x_j^{(i)})$. Therefore, q_ϕ has the interpretation of being an approximation to the posterior. When we implement q_ϕ, we will keep this fact in mind.

We can further increase tractability by approximating the calculation of the expectation over q_ϕ. We do this by taking a Monte-Carlo sample from q_ϕ and evaluating the expectation with just this sample. This approximation can be made more precise by taking multiple samples and averaging the expectation, but, for this paper, we used only one. With this, we arrive at our final objective,

$$\begin{aligned} &\textstyle\sum_D E_{q_\phi(k_0,...,k_3|x_j^{(i)})} \left[log \frac{p_\theta(x_j^{(i)},k_0,...,k_3)}{q_\phi(k_0,...,k_3|x_j^{(i)})} \right], \\ &\approx \textstyle\sum_D log \frac{p_\theta(x_j^{(i)},k_0^{(i)},...,k_3^{(i)})}{q_\phi(k_0^{(i)},...,k_3^{(i)}|x_j^{(i)})}, \\ &\equiv J, \\ &where\, k_0^{(i)},...,k_3^{(i)} \sim q_\phi(k_0,...,k_3|x_j^{(i)}). \end{aligned} \tag{3}$$

E. Inference Network

Here we discuss the details of the inference network, q_ϕ. Given a CT image x_j, we infer d_j in the same manner as we did for the U-SVAE model. From d_j we infer $d_{j+1},...,d_3,d_0,...d_{j-1}$. Then from $d_0,d_1,...d_3$, we infer $k_0,k_1,...,k_3$ using a shared FilterCrush function, as in the U-SVAE. Note that each mapping from x_j to d_j is shared across each j. This is depicted in figure 7.

F. Time Invariant Information

As it is, the CAGE model should be able to model the information in the image that changes over time. For example, it should be able to model the severity of COVID-19 over time. However, this model will have a difficultly modeling the information present in the image that does *not* change over time, as it must learn to maintain that information from one latent variable to the next. For example, it will struggle to model the patients gender or age. This is, in fact, what we observed in our initial experiments.

We remedy this by introducing a separate latent variable, z, that is used to model *all* CT images regardless of the COVID-age group. We concatenate this latent variable's deterministic expansion with $k'_j s$ expansion and use that as input to the decoder. This is depicted in figure 6

Of course, the generative and inference network for this new latent variable use the U-SVAE model.

V. Experiments

In this section, we define the experimental setting, data and evaluated models.

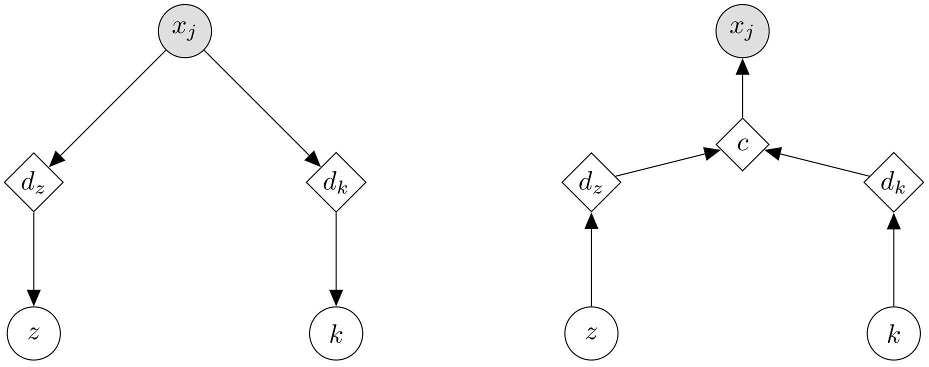
A. Data Set

For the evaluation of our method, we used the China Consortium of Chest CT Image Investigation (CC-CCII) data set, described by Zhang et al. [17], with some significant modifications. This data set contains 444,034 CT images from 2,778 patients. Eighty-five percent of the patients were from the Chinese cities of Yichang, Hefei or Guangzhou and the remainder were from an international cohort.

The patients either had common pneumonia, novel-coronavirus pneumonia, or were part of the control group. In our data set, we excluded the patients with common pneumonia.

Of the novel-coronovirus patients, only 409 had progression information available. This came in the format of number of days since hospitable admission. Of course, the progression information from the control patients were implicitly available, as we know that they have had COVID-19 for zero days. We excluded all patients without progression information.

We will now discuss the data preprocessing and cleaning procedure we employed. The original data set included pixel-level labels of the lung field for a small subset of the data. We used this to train a U-Net [14] semantic segmentation model that effectively segments the lung field present in the CT image. Using this model, as well as the opening and closing



(a) Inference Network

(b) Generative Network: c denotes a concatenation

Fig. 6: CAGE with Time Invariant Latent Variable: We use k as a short hand for the full series $k_0, k_1, ...k_3$

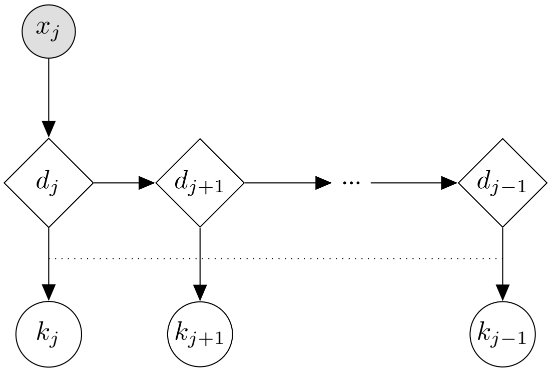


Fig. 7: CAGE Inference Network: Dashed line denotes shared weights

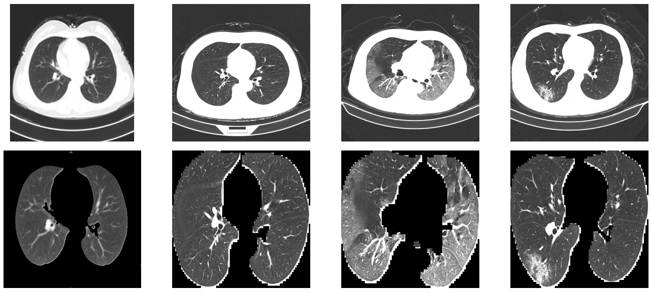


Fig. 8: Before (Top) and After (Bottom) Data Pre-Processing

morphological transformations for noise reduction, we cropped the CT images so that they would only include lung field.

Then, for efficiency reasons, we take the middle most slice of each CT scan and remove all others. This ensures that we have a data set with a similar amount of diversity to the original data set, while being significantly smaller. After this, we manually remove any CT images that do not have the lungs in full view or have a significant amount of non-lung field present in the CT image. The result is shown in figure 8.

After this process, we were left with one CT image from each of 977 healthy patients and 301 from patients with COVID-19. We partitioned the data into the following four COVID-age groups:

- Healthy patients
- Patients who have had COVID for less then 5 days
- Patients who have had COVID for less then 10 days
- Patients who have had COVID for less then 15 days

Fourteen days was the longest time present in the data. We refer to these partitions as the t0, t1, t2 and t3 group, respectively.

Before being used in our models, all images are downsized to a resolution of 64x64 and the pixels values are scaled to be between zero and one.

The data set can be found at https://github.com/JudahZammit/cage-dataset.

B. Models Evaluated

We evaluated the U-SVAE as well as the CAGE model, but trained it on three separate subsets of the data. The first was trained on the t0 group. This will be referred to as CAGE-T0. The second was trained on the t1 group. This will be referred to as CAGE-T1. The third was trained on all the patients. This will be referred to as CAGE-ALL.

C. Training Procedure

We trained each model for 400 epochs. Each epochs ran for 100 iterations with a learning rate of 0.0003. A batch size of 16 CT images, evenly distributed across the COVID-age groups, was used for both the CAGE models and the U-SVAE model.

D. Software and Hardware

The CAGE model was implemented and trained with TensorFlow [18]. We used Segmention-Models [19] for the lung field semantic segmentation U-Net model. We used OpenCv [20] for the opening and closing morphological operations. All experiments were done on a RTX 2070 graphics card.

VI. Results

A. U-SVAE Generative Performance

The U-SVAE model achieved an average negative log-likelihood of -7.98, a reconstruction loss of -8.03 and a kullback-leibler divergence of 0.05. The ground-truth, reconstructed and generated CT images are shown in figure 9.

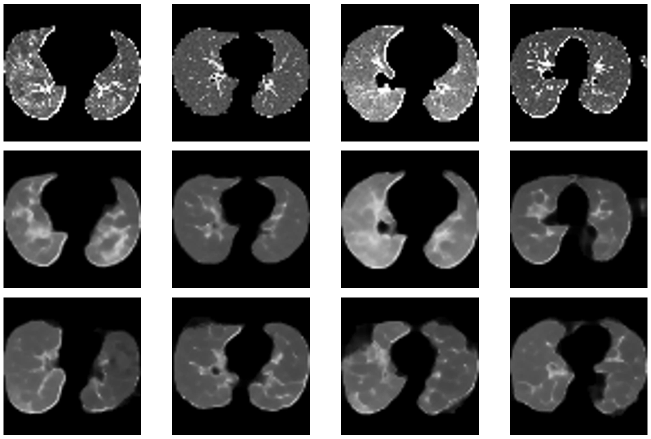


Fig. 9: U-SVAE Visual Comparison: Comparison between the ground-truth (top), the U-SVAE's reconstructions (middle) and the U-SVAE's generated CT images (bottom)

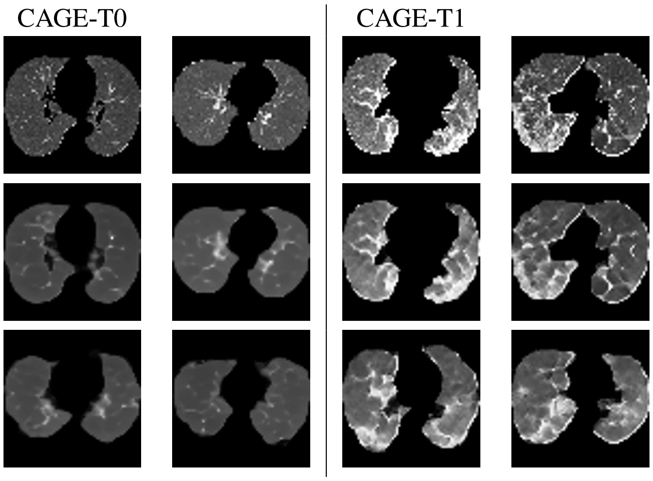


Fig. 10: CAGE-T0 and CAGE-T1 Visual Comparison: Comparison between the ground-truth (top), CAGE's reconstructions (middle) and CAGE's generated CT images (bottom)

B. CAGE Generative Performance

Here we will discuss the generative performance of the CAGE-T0, CAGE-T1 and CAGE-ALL models.

The CAGE-T0 model achieved a negative log-likelihood of -8.35, a reconstruction loss of -8.45 and a kullback-leibler divergence of 0.10. The CAGE-T1 model achieved a negative log-likelihood of -8.14, a reconstruction loss of -8.23 and a kullback-leibler divergence of 0.9.

Both are shown in figure 10. Note that, when trained on only one COVID-age group, performance is quite satisfactory.

The CAGE-ALL model achieved a negative log-likelihood of -8.20, a reconstruction loss of -8.25 and a kullback-leibler divergence of 0.06. The performance of the model is shown in figure 11. Note that the models ability to model the progression of COVID-19 is very poor. We investigate the reason for this in the following section.

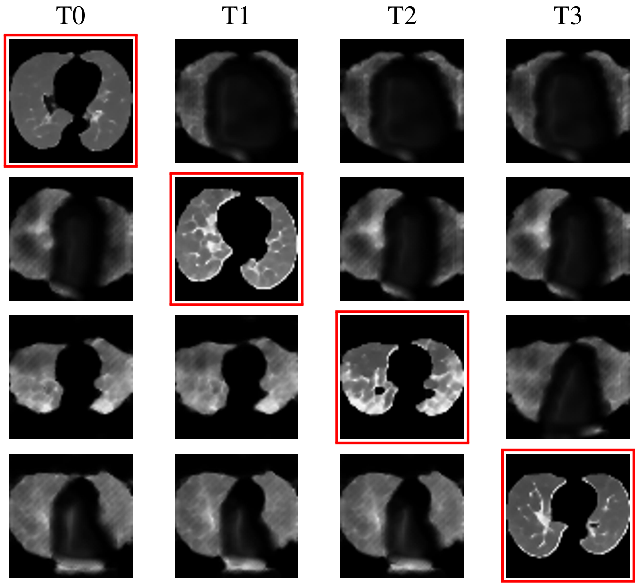


Fig. 11: CAGE-ALL COVID-Aging Performance: Reconstructions from ground-truth CT images are bordered in red

C. The Tourist Crisis

Here we will present a hypothesis explaining the failure of the CAGE model.

We assumed that the t0 COVID-age group is determined by a unit-gaussian distributed latent variables k_0. Our model should infer values for k_0 from the t0 group that follow a unit-gaussian distribution. This enables us to take samples from a unit-gaussian distributed k_0 to generate CT images realistic for patients within the t0 group. This is displayed by the CAGE-T0 model.

We also assumed that the t1 group is determined by a latent variable, k_1. This latent variable follows a diagonal-gaussian distribution when conditioned on k_0. However, this means that CT images from the t1 group are still determined by k_0. Therefore, our model should still infer values for k_0 from the t1 group that follow a unit-gaussian distribution. This enables us to take samples from k_0 and k_1 obtaining CT image that is realistic for patients within the t1 group. This is displayed by the CAGE-T1 model.

Now, if both of these conditions are true for the CAGE-ALL model, then we should be able to take a sample from k_0 and k_1 and obtain two realistic CT images, x_0 and x_1, corresponding to the to and t1 age group. However, instead we observe very unrealistic CT images.

We hypothesis that this is because k_0 is determined by the t1 group just as much as it is by the t0 group. However, k_0 should be used to model only the t0 group, where the t1 group should use, but not determine, k_0.

This is analogous to a country with high tourism rates giving tourist voting rights equal to that of its own citizens. The country would quickly become a poor place to live for its own citizens. Because of this similarity, we call this issue *the tourist crisis*.

In our situation, the CT images determining each variable, k_j, consists of only 1/4 patients from the tj COVID-age group, with the rest being from the other COVID-age group.

VII. Future Work

Currently, our CAGE model is learning latent variables for both the time invariant information, such as age and gender, and time variant information, such as severity of COVID-19. It could be beneficial to explicitly include these clinical features in the graphical model. This would alleviate the need for our model to learn features that clinicians already know. For example, this could be done by including age as a variable that determines the CT image, x, and can be inferred from x. For time variant information, we could include a segmentation mask for the lesions present in the CT image. This would be determined by the latent variable, k.

With the tourist crisis fixed, this model would have the potential to predict the future severity of both healthy and unhealthy patients. This could be done by including some form of severity (such as whether the patient requires a ventilator or the chances the patient will die) as output, determined by the latent variable k.

Finally, our data set, though diverse, is quite small. This is because we can only use CT images where the progression information is known. We could remove this constraint by including the progression information, call it t, as a partially observed latent variable. This would allow us to use all CT images regardless of whether the progression information is known.

VIII. Conclusion

In conclusion, we developed a novel generative model and showed that it has effective performance on lung CT images. We created a data set for measuring COVID-age progression in Lung CT images. In addition, we adapted the U-SVAE model for COVID-age progression and showed that it fails due to an issue which we named *the tourist crisis*. Finally, we proposed some potential solutions to this issue.

References

[1] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, “Ladder variational autoencoders,” in *Advances in neural information processing systems*, 2016, pp. 3738–3746.

[2] X. Li, S. Xu, M. Yu, K. Wang, Y. Tao, Y. Zhou, J. Shi, M. Zhou, B. Wu, Z. Yang, C. Zhang, J. Yue, Z. Zhang, H. Renz, X. Liu, J. Xie, M. Xie, and J. Zhao, “Risk factors for severity and mortality in adult covid-19 inpatients in wuhan,” *Journal of Allergy and Clinical Immunology*, vol. 146, no. 1, pp. 110 – 118, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0091674920304954

[3] K. Li, J. Wu, F. Wu, D. Guo, C. Linli, F. Zheng, and L. Chuanming, “The clinical and chest ct features associated with severe and critical covid-19 pneumonia,” *PubMed*, vol. 55, pp. 327 – 331, 2020.

[4] G. Antipov, M. Baccouche, and J. Dugelay, “Face aging with conditional generative adversarial networks,” in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 2089–2093.

[5] S. Zhou, W. Zhao, J. Feng, H. Lai, Y. Pan, J. Yin, and S. Yan, “Personalized and occupational-aware age progression by generative adversarial networks,” 2017.

[6] J. Zeng, X. Ma, and K. Zhou, “Photo-realistic face age progression/regression using a single generative adversarial network,” *Neurocomputing*, vol. 366, pp. 295 – 304, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231219310926

[7] R. C. Xie and G. S. J. Hsu, “A hybrid network for facial age progression and regression learning,” in *2020 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, 2020, pp. 1–6.

[8] H. Yang, D. Huang, Y. Wang, and A. K. Jain, “Learning continuous face age progression: A pyramid of gans,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2019.

[9] Z. Zhang, Y. Song, and H. Qi, “Age progression/regression by conditional adversarial autoencoder,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[10] P. K. Chandaliya and N. Nain, “Conditional perceptual adversarial variational autoencoder for age progression and regression on child face,” in *2019 International Conference on Biometrics (ICB)*, 2019, pp. 1–8.

[11] J. Zeng, X. Ma, and K. Zhou, “Caae++: Improved caae for age progression/regression,” *IEEE Access*, vol. PP, pp. 1–1, 11 2018.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Commun. ACM*, vol. 63, no. 11, p. 139–144, Oct. 2020. [Online]. Available: https://doi-org.uml.idm.oclc.org/10.1145/3422622

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[14] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[15] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 448–456. [Online]. Available: http://proceedings.mlr.press/v37/ioffe15.html

[16] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 807–814.

[17] K. Zhang, X. Liu, J. Shen, Z. Li, Y. Sang, X. Wu, Y. Zha, W. Liang, C. Wang, K. Wang, L. Ye, M. Gao, Z. Zhou, L. Li, J. Wang, Z. Yang, H. Cai, J. Xu, L. Yang, W. Cai, W. Xu, S. Wu, W. Zhang, S. Jiang, L. Zheng, X. Zhang, L. Wang, L. Lu, J. Li, H. Yin, W. Wang, O. Li, C. Zhang, L. Liang, T. Wu, R. Deng, K. Wei, Y. Zhou, T. Chen, J. Y.-N. Lau, M. Fok, J. He, T. Lin, W. Li, and G. Wang, “Clinically applicable ai system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography,” *Cell*, vol. 181, no. 6, pp. 1423 – 1433.e11, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0092867420305511

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[19] P. Yakubovskiy, “Segmentation models,” https://github.com/qubvel/segmentation_models, 2019.

[20] G. Bradski, “The OpenCV Library,” *Dr. Dobb's Journal of Software Tools*, 2000.