Generative Adversarial Urban Growth Prediction of Doha

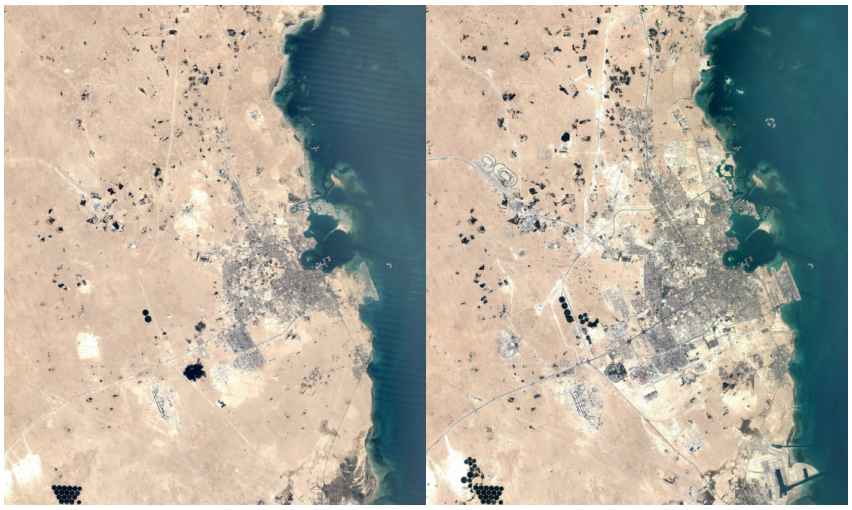
Ziad Khattab

1 Introduction

Doha, Qatar is an emerging city experiencing rapid growth in recent years that is expected to continue to increase rapidly in anticipation of the 2022 FIFA World Cup, as well as planned growth designated by the Qatar 2035 Vision.

In order to predict the urban coverage of Doha, manual urban analysis was performed, along with a machine-learning based computer analysis to generate predictions. This was done using satellite images as the input data, such as the following,

Figure 1: satellite images of Doha in 2006 (left) vs. 2016 (right)



The objective is to create an artificial neural network that can take a sequence of satellite images and generate future simulated satellite maps that meet the following requirements:

(a) Display a viable image of a simulated Doha that looks reasonable and believable.

(b) Preserve the coastline and water elements virtually unchanged.

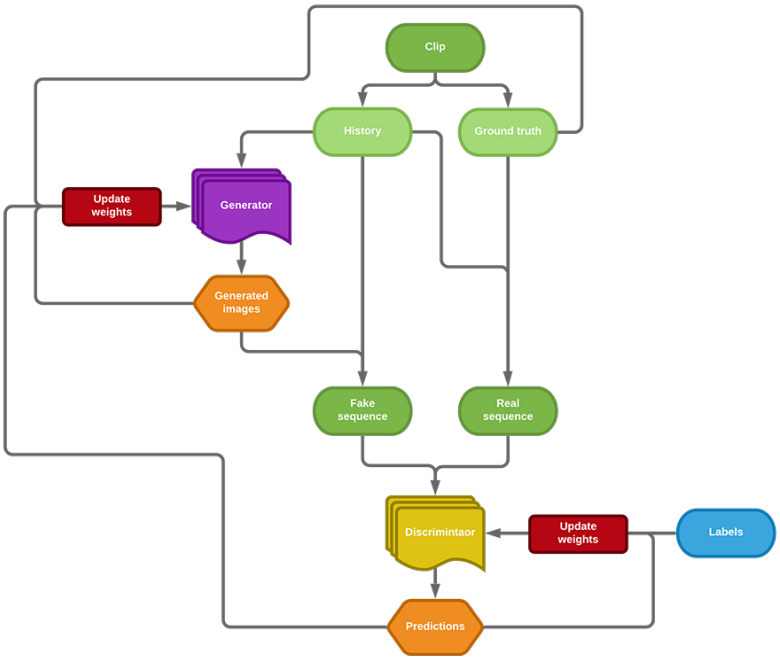
(c) Match expected urban coverage of Doha and display this urban coverage clearly.

2 Network model

Prediction of urban growth from a sequence of satellite images requires not only the identification of growth patterns, but the ability to generate images of future maps with decently realistic quality and good enough definition to visually identify urban and non-urban areas. For this, a generative adversarial network (GAN) is used. [1]

In a GAN, a generator network receives the history frames and attempts to provide a realistic continuation to the clip, and a discriminator network attempts to determine whether the clips it receives are real or fake, assigning a probability to the image between 0 and 1 of how likely it is to be real. The two networks compete against one another, with the generator attempting to fool the discriminator into thinking that the generated output images are areal, while the discriminator attempts to pick apart real and fake images more accurately.

Figure 2: graph of the generative adversarial model

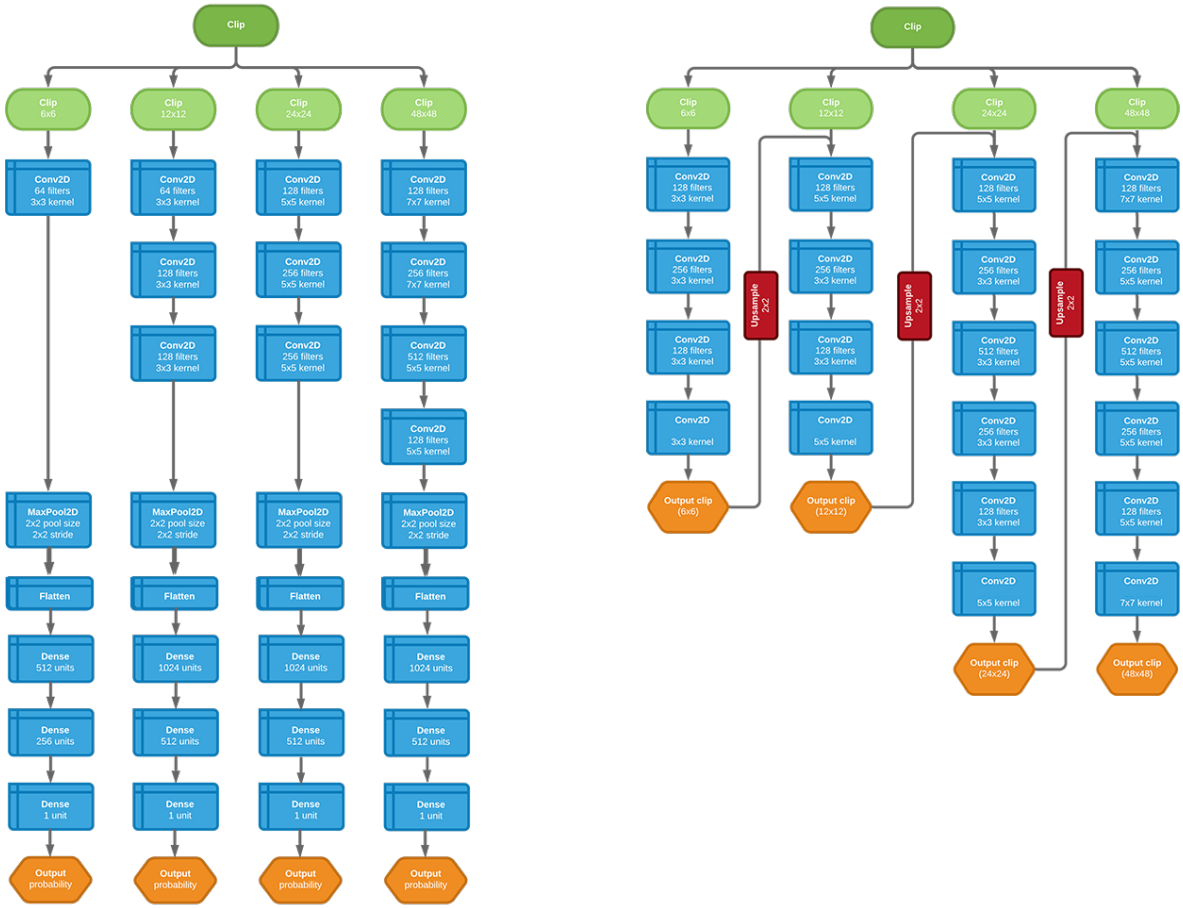


In this model, the generator and discriminator are both multi-scale models [4].

The discriminator is a convolutional neural network image classifier that runs the clips through convolutional layers activated with ReLU, and then to fully connected layers to obtain a scalar prediction between 0 and 1. It operates on a multi-scale model, meaning that a 64 by 64 pixel square clip is downscaled to 8 by 8, 16 by 16, 32 by 32, and the original clip, and a prediction generated for each scale.

The generator is a fully convolutional image generator model that also operates with the same four-scale model. However, a significant difference from the discriminator is that the generator concatenates the upsampled output of each scale to the next one to strengthen the time dependency.

Figure 3: graph of the discriminator and generator models



3 Data preprocessing

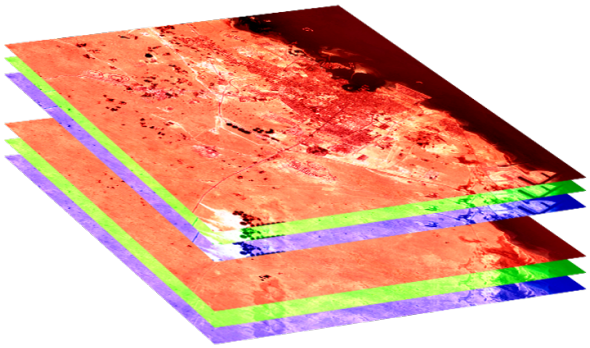
Before being fed to the model, the satellite images need to be preprocessed for compatibility, efficiency, and enhancement.

3.1 Clip stacking

The first element of preprocessing is in stacking a series of time-ordered into a single "clip" unit that can be passed to the model as one object. In this model, clips were treated as either sequences of individual frames sequenced into a list, or as a single three-dimensional matrix of shape,

$$(\text{height}, \text{width}, \text{number of frames} \times \text{number of channels})$$

Essentially, a stacked clip of two RGB images would be visualized as such,

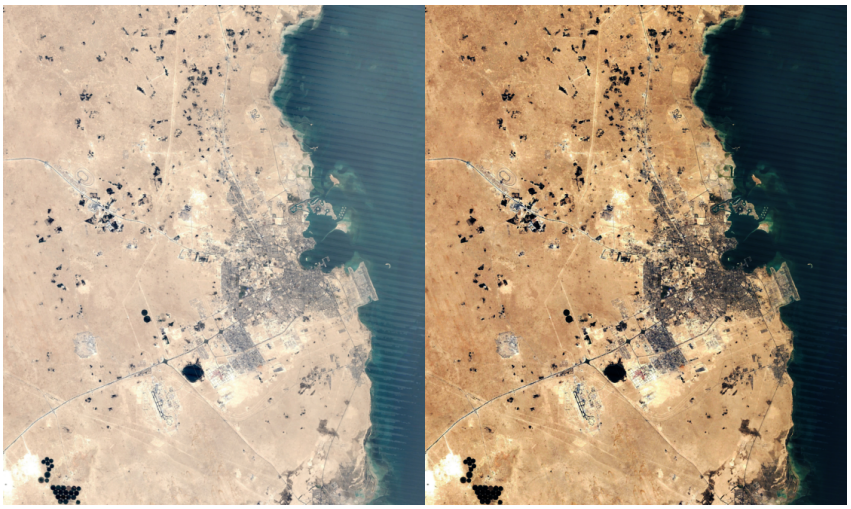


3.2 Clip cropping

Due to memory constraints, as well as ease of upsampling and downsampling images for the sake of the multi-scale model, the images are cropped into 64×64 squares. Since the generator is purely convolutional, it can actually generate images of any size using weights trained on smaller squares, which is why this model is able to generate full-sized maps.

3.3 Filter enhancement

In order to help the model recognize the urbanized areas more clearly, contrast and brightness filters were applied in image preprocessing to make urban areas more distinct compared to the surrounding desert. A satellite map with the filters applied would appear as such,



4 Loss functions

To mathematically define the loss functions used in training, we must first define the generator and discriminator models as functions,

$$\begin{aligned} \text{gen}: \ \text{images} &\to \text{images} \\ \text{disc}: \ \text{images} &\to [0,1] \end{aligned}$$

4.1 ℓ_p loss

We define the ℓ_p loss as,

$$\ell_p(x,y) = |x-y|^p, \ p \in \{1,2\}$$

This loss represents either the absolute difference (if $p=1$), or the absolute squared difference (if $p=2$) between the generated images x and the ground truth images y. This is the simplest metric for the accuracy of the generated images.

4.2 Adversarial loss

First, define the binary crossentropy loss as,

$$\text{bce}(y,y') = -\sum_i y_i \log(y'_i) + (1-y_i)\log(1-y'_i)$$

where y represents the predicted labels given as output of the discriminator, and y' represents the true labels assigned to the data. For each generate image that the discriminator receives, it generates a probability that this image matches the ground truth image. The binary crossentropy loss is a measure of how close this probability is to the truth. It operates on a logarithmic basis, so probabilities that are the same as the true label have a very small loss, while probabilities that are very far from the true label have an enormous loss.

Next, we define the discriminator loss,

$$\text{loss}_{\text{disc}}(x,y) = \text{bce}(\text{disc}(x,y),1) + \text{bce}(\text{disc}(x,\text{gen}(x)),0)$$

4.3 Discriminator loss

The loss used for the discriminator is defined as,

$$\text{adv}(x,y) = \text{bce}(\text{disc}(x,\text{gen}(x)),0)$$

4.4 Gradient difference loss (GDL)

Define the image GDL [4] as,

$$\begin{aligned} \text{gdl}(x,y) = \sum_{i,j} &\big|\,|y_{i,j} - y_{i-1,j}| - |\text{gen}(x)_{i,j} - \text{gen}(x)_{i-1,j}|\,\big|^c \\ &+ \big|\,|y_{i,j-1} - y_{i,j}| - |\text{gen}(x)_{i,j-1} - \text{gen}(x)_{i,j}|\,\big|^c \end{aligned}$$

This is used to penalize images that are significantly blurry and fuzzy, to improve definition of the final predicted image. The value of c is a constant to be determined through arbitrary choice or fine tuning. For the purposes of the `dohamaps` model, the value of $c=1$ was used.

4.5 Generator loss (combined)

Finally, the combined loss, which is used as the loss for the generator,

$$\text{loss}_{\text{gen}}(x,y) = \alpha\,\text{adv}(x,y) + \beta\,\ell_p(x,y) + \gamma\,\text{gdl}(x,y)$$

Once again, the values of α, β, and γ are constants, which in the `dohamaps` model were set to $\alpha = 0.05$, $\beta = 1$, and $\gamma = 1$.

5 Metrics

In addition to the loss functions, the model contains metrics. These metrics do not directly inform the training loop, but are reported regularly to provide measures of the model's performance.

5.1 Peak signal to noise ratio (PSNR)

The PSNR is defined as,

$$\text{psnr}(x, y) = 10 \cdot \log_{10} \left(\frac{N \cdot \max}{\sum(x_i - y_i)} \right)$$

where N is the number of channels, and max is the maximum value of the image signal. In a default RGB image, this value is 255. The PSNR is measured in decibels, where a higher value indicates an image that is harder to distinguish from the original by the naked eye.

5.2 Sharpness difference

The sharpness difference is defined as,

$$\text{sharpdiff}(x, y) = 10 \cdot \log_{10} \left(\frac{N \cdot \max^2}{\sum_i \sum_j |(\Delta_i x + \Delta_j x) - (\Delta_i y + \Delta_j y)|} \right)$$

which measures the loss of sharpness between the true frame x and the predicted image y.

5.3 Structural similarity index measure (SSIM)

Define the SSIM [2] as,

$$\text{ssim}(x, y) = \frac{(2\mu_x \mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

where μ_x is the average of image x, and μ_y is the average of image y, σ_x and σ_y are the variance of x and y, σ_{xy} is the covariance of x and y, and c_1 and c_2 are constants.

6 Hyperparameters

In addition to the losses and metrics defined above, there are several significant hyperparameters that can heavily influence the training and outputs of the model. The generator and the discriminator both use the Adam optimizer [3], which has takes as its main parameter the learning rate. Learning rates that are too high will cause the loss to oscillate and lead to inaccurate results, while learning rates that are too low will not converge in a reasonable amount of time.

The next pair of hyperparameters are the history and prediction length. The history length represents the length of the clip taken as an input to the generator, while the prediction length represents the length of the clip output as a prediction. Predictions beyond the length of the prediction clip are computed recursively from initial prediction. Image sequences that change over time, but not necessarily in a strongly time-linked pattern, benefit from a shorter history and prediction length. For the purposes of urban growth, where time-linked patterns are important, longer history and prediction lengths are beneficial.

The final hyperparameter is a combination of two items:

(a) The size of the clips are preprocessed from the input images.

(b) The scale of the images that the clips are sourced from.

The significance of this is in the ratio between the two. Reducing overall image size and increasing the size of the clips allows capturing more large-scale features at the cost of detail and small-scale definition, while the inverse would lead to very high definition images that miss large time-dependent features.

7 Testing and tuning

Tuning hyperparameters and model architecture is a matter of trial and error. As such, numerous trial runs were conducted with various values of the hyperparameters. This section will discuss the most significant trials.

7.1 Exhibit A

Hyperparameter	**Value**
Generator learning rate	0.00004
Discriminator learning rate	0.00002
History length	4 frames
Prediction length	1 frames
Clip size	32×32
Image size	2800×3320

Figure 4: 2020 (left) vs. generated 2035 (right)

In comparison to the goals of the model stated in the introduction, this output meets two of the three objectives decently well.

(a) It displays a viable image of a simulated Doha that looks reasonable and believable. It lacks some definition in certain areas and has artifacts from the enhancement filters applied in the training, however looking at it clearly resembles a real-life map of Doha.

(b) It preserves the coastline extremely well, with essentially no changes to it, including the airport and the Pearl areas.

(c) It does **not** match the expected urban coverage of Doha. In fact, the two maps look virtually identical on the large scale, with some small changes visible in certain areas, but nowhere near what was expected.

7.2 Exhibit B

Hyperparameter	**Value**
Generator learning rate	0.000005
Discriminator learning rate	0.00002
History length	8 frames
Prediction length	8 frames
Clip size	48×48
Image size	560×664

Figure 5: 2020 (left) vs. generated 2035 (right)

This model improves in some areas compared to the stated goals, although it still needs some work.

(a) It displays a viable image of a simulated Doha that looks somewhat reasonable and believable. It lacks a lot of definition, even compared to Exhibit A's output, however, it still resembles Doha.

(b) It preserves the coastline decently well, with mostly no changes, although some areas near the southern end of the coastline are blurry or altered.

(c) It does **not** match the expected urban coverage of Doha. While there is more noticeable growth, this is the amount of growth that would roughly be expected of a period of 3-5 years rather than 15 years.

7.3 Exhibit C

Hyperparameter	**Value**
Generator learning rate	0.00000085
Discriminator learning rate	0.000005
History length	11 frames
Prediction length	19 frames
Clip size	64×64
Image size	560×664

Figure 6: 2020 (left) vs. generated 2035 (right)

In comparison to the goals of the model stated in the introduction, this output meets all of the three objectives decently well.

(a) It displays a viable image of a simulated Doha that looks quite reasonable and believable. It lacks the extreme fine-detail definition of Exhibit A, however it preserves most features such as roads and specific clusters of buildings that were already present in the city.

(b) It preserves the coastline well, with virtually no changes.

(c) It **decently** matches the expected urban coverage of Doha. The city appears to have enlarged by roughly a factor of 1.5x, especially in the northern side, which matches urban growth projections.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial networks. NIPS, 2014.

[2] Wang, Zhou, Bovik, Alan C., Sheikh, Hamid R., and Simoncelli, Eero P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Im. Proc.*, 13(4):600–612, 2004.

[3] Diederik P. Kingma, and Jimmy Ba, Adam: A Method for Stochastic Optimization, 2014.

[4] Michael Mathieu, Camille Couprie, and Yann LeCun, Deep multi-scale video prediction beyond mean square error, 2015.