Generative Adversarial Urban Growth Prediction

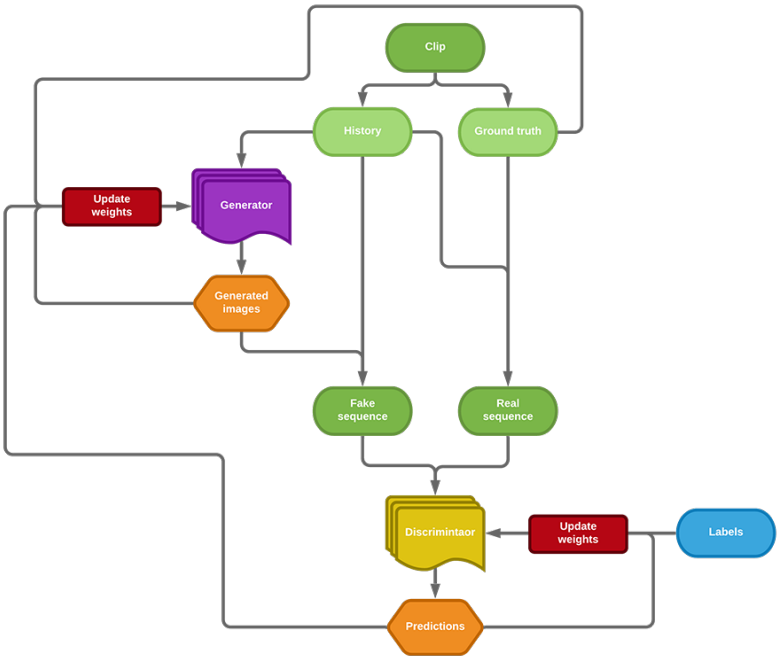
Ziad Khattab

1 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.

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.

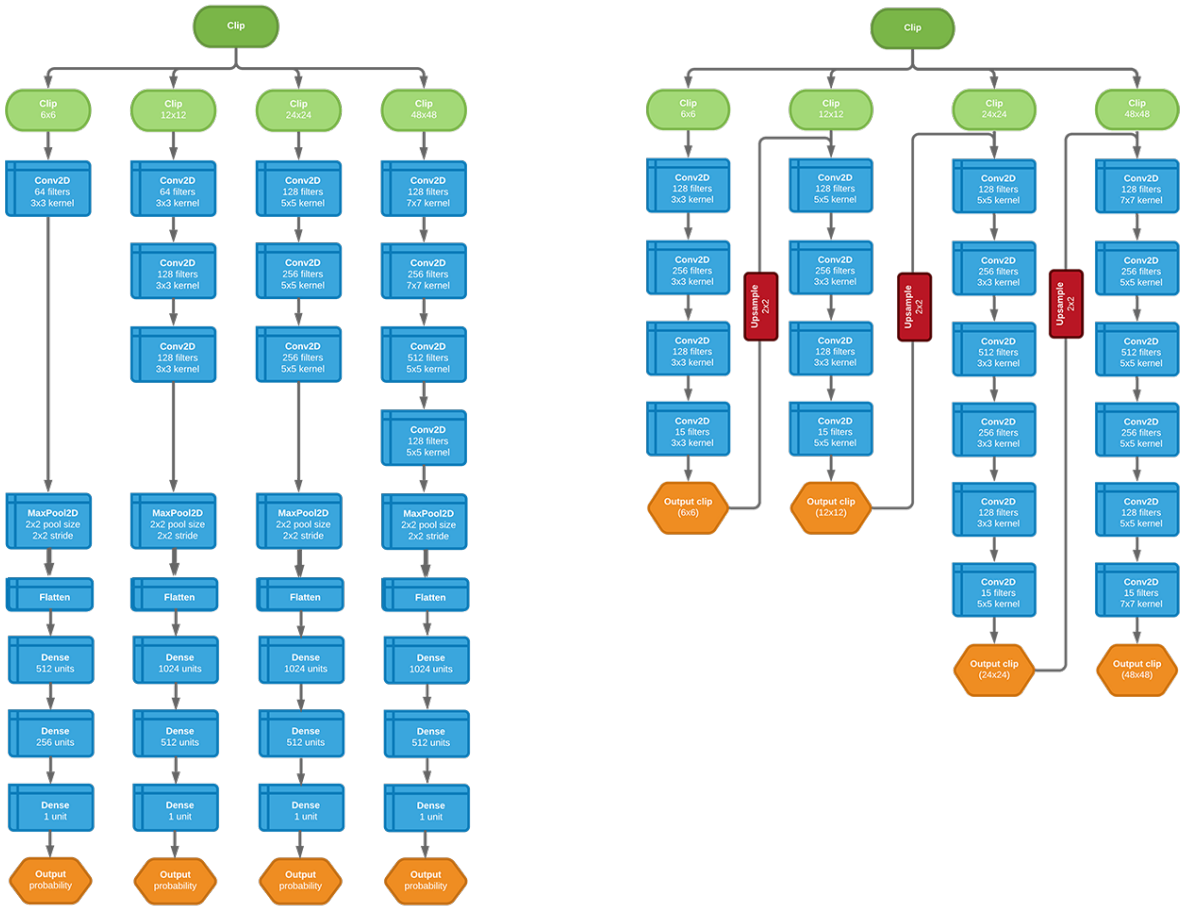
Figure 1: graph of the generative adversarial model



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 48 by 48 pixel square clip is downscaled to 6 by 6, 12 by 12, 24 by 24, 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 2: graph of the discriminator and generator models



2 Loss functions

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

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

2.1 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.

2.2 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,

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

2.3 Discriminator loss

The loss used for the discriminator is defined as,

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

2.4 Gradient difference loss (GDL)

Define the image GDL 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.

2.5 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 $\gamma = 1$.

3 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.

3.1 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.

3.2 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.

3.3 Structural similarity index measure (SSIM)

Define the SSIM (Wang et. al) 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.

4 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 (Kingma et. al), 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 the size of the clips that the input images are cropped to due to memory constraints. Clips that are smaller in size will allow the model to capture more detailed features and output images with higher definition, but will miss out on larger image features that are obtained from larger clips.

The following table displays the values of all these hyperparameters in the `dohamaps` model.

Hyperparameter	**Value**
Generator learning rate	0.0000085
Discriminator learning rate	0.000005
History length	8 frames
Prediction length	16 frames
Clip size	64×64