GENERATIVE STRESNET FOR CRIME PREDICTION

Tran Phong & Hoong Chuin Lau *
School of Computing and Information Systems
Singapore Management University
80 Stamford Rd, Singapore

ABSTRACT

In this work, we combine STResnet (Zhang et al., 2017) with VAE Kingma & Welling (2013) to generate crime distribution. The outputs can be used for downstream tasks such as patrol deployment planning Chase et al. (2021).

1 INTRODUCTION

Crime prediction is an essential task in public safety, yet remains challenging due to its intricate factors and erratic behaviors. In this paper, we present an approach to add stochasticity into spatio-temporal prediction, and apply it to generate scenarios needed for the downstream planning task as reported in Chase et al. (2021). It is done by combining a variational autoencoder with STResnet.

2 RELATED WORKS

Huang et al Huang et al. (2018) employ a hierarchical GRU with attention for binary pixel indication in images, where pixels represent map regions. In Ye et al. (2021), an inception-residual CNN is used to predict crime probabilities in discretized grids. Wang et al. (2017) utilizes STResnet from Zhang et al. (2017) for crime prediction. However, most deep learning methods, including Wang et al. (2017) lack stochastic outputs, thus limiting their use in deployment planning tasks as in Chase et al. (2021), which require multiple scenario inputs for generating robust plans. This paper presents an approach to integrate stochasticity into predictions.

3 GENERATIVE ST-RESNET ARCHITECTURE

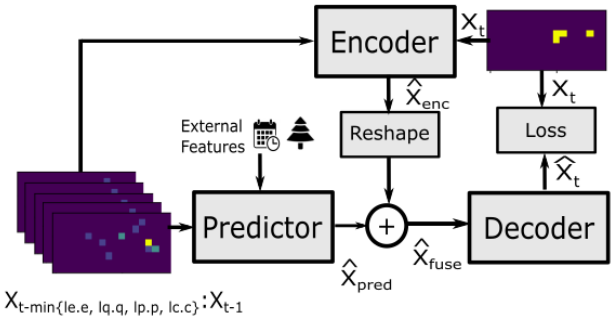


Figure 1: Architecture Diagram of Generative ST-ResNet

Figure 1 introduces the architecture of Generative ST-ResNet which contains the main 3 components: *Predictor*, *Encoder*, and *Decoder*.

The inputs are a time sequence of images, each image being a crime intensity map. To create this map image, we discretize a region (where we want to predict the timings and locations of crimes)

*Corresponding Author hclau@smu.edu.sg.

into an $M \times N$ grid map which is controlled by parameter *resolution* as the length of each square grid. We aggregate incidents during an *interval* and count the number as the intensity of the grid. Given a sequence of created images, we slice them to create four fragments of sequences: *instant, closeness, period*, and *trend* based on the sampling rates denoted by e, c, p and q.

Let l_e, l_c, l_p, l_q denote the length of the instant, closeness, period, and trend fragment sequences. The following equations are used to slice the data in each window into these four sequences:

$$S_e = [X_{t-l_e.e}, X_{t-(l_e-1).c}, ..., X_{t-e}]$$
$$S_c = [X_{t-l_c.c}, X_{t-(l_c-1).c}, ..., X_{t-c}]$$
$$S_p = [X_{t-l_p.p}, X_{t-(l_p-1).p}, ..., X_{t-p}]$$
$$S_q = [X_{t-l_q.c}, X_{t-(l_q-1).q}, ..., X_{t-q}]$$

Thus the window length is $min\{l_e.e, l_c.c, l_p.p, l_q.q\}$ plus 1 to cover the target image X_t.

The instant component presents the *noise* of data, which together with *target* form the input for the *Encoder* block. The output of this block is a random latent variable to capture the noise of data and the uncertainty of the prediction.

The closeness, period, and trend components model the temporal properties, as in Zhang et al. (2017) are fed into *Predictor* block, together with cyclical data from incidents' occasion and day types (weekend, weekday, or holiday).

The inputs to *Decoder* are from *Predictor* and *Encoder*. The block contains a series of transposed convolutional networks to match the size of the output.

More details are found in the Appendix.

4 Experiments and Results

We train the neural network with 1 year of crime incident data of Singapore in 2020 and predict crimes of 6 months in 2021. Each record entry includes multiple fields such as crime types, duration, postal code but we only consider start time, latitude and longitude for prediction. The data spread geographically from 1.2101° to 1.4707° in latitude and from 103.6056° to 104.0436° in longitude.

The network is trained using MSE Loss. The parameters are: $resolution = 0.015, interval = 60$ minutes, $l_e = 24, e = 1, l_c = 12, c = 1$ (hourly closeness and instant), $l_p = 3, p = 24$ (daily period), $l_t = 1, t = 168$ (weekly trend)

We use F1 score and compare STResnetVAE (our method) with STResnet Zhang et al. (2017) and GAN Goodfellow et al. (2014). Once we have the intensity output, we convert it into latitude, longitude and timestamp. If there is more than 1 incident at the same pixel, we randomly select the time in the timestamp duration. We then match the predicted incidents with the ground-truth incident using linear assignment under various distances and durations as shown in the table below.

Table 1: F1 score

DISTANCE	DURATION	GAN	STResnet	STResnetVAE (Ours)
1000 m	10 mins	0.15	0.167	0.150
2000 m	30 mins	0.40	0.598	0.596
2000 m	60 mins	0.43	0.749	0.755

From the table, we see that STResnetVAE is good at predicting incidents when the distance and duration are large, while this does not hold for small instances. That said, STResnetVAE yields on-par performance with STResnet while being able to predict the distribution of outputs, thus giving the downstream planning task the ability to optimize based on diverse predicted outcomes.

URM STATEMENT

The first author Tran Phong meets the URM criteria of ICLR 2023 Tiny Paper Track, as he is the first-time submitter.

REFERENCES

Jonathan Chase, Tran Phong, Kang Long, Tony Le, and Hoong Chuin Lau. Grand-vision: An intelligent system for optimized deployment scheduling of law enforcement agents. *Proceedings of the International Conference on Automated Planning and Scheduling*, 31:459–467, 05 2021. doi: 10.1609/icaps.v31i1.15992. URL https://ojs.aaai.org/index.php/ICAPS/article/view/15992.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL https://arxiv.org/abs/1406.2661.

Chao Huang, Junbo Zhang, Yu Zheng, and Nitesh V. Chawla. Deepcrime. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 10 2018. doi: 10.1145/3269206.3271793.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. URL https://arxiv.org/abs/1312.6114.

Bao Wang, Penghang Yin, Andrea L. Bertozzi, P. Jeffrey Brantingham, Stanley J. Osher, and Jack Xin. Deep learning for real-time crime forecasting and its ternarization. *arXiv:1711.08833 [cs, math, stat]*, 11 2017. URL https://arxiv.org/abs/1711.08833.

Xinyue Ye, Lian Duan, and Qiong Peng. Spatiotemporal prediction of theft risk with deep inception-residual networks. *Smart Cities*, 4:204–216, 03 2021. doi: 10.3390/smartcities4010013. URL https://www.mdpi.com/2624-6511/4/1/13.

Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. *arXiv:1610.00081 [cs]*, 01 2017. URL https://arxiv.org/abs/1610.00081.

A APPENDIX

A.1 STRUCTURES OF PREDICTOR BLOCK

The three temporal sequences: closeness, period and trend share the same network architecture which is a series of connected residual units. For each of the sequences, we concatenate the intensity maps along the last axis (channels) to make it a 3D Tensor. The three tensors together with external features are propagated through ST-ResNet architecture to get output, denoted $\hat{X}_{pred}$ as in Figure 2.

A.2 STRUCTURES OF ENCODER BLOCK

In Training phase, Encoder block receives two types of inputs: sequences of instant inputs and target. We process the sequences as same in Predictor: concatenate along the channels axis to make one tensor. Then we propagate the tensor and target through a separate network (but same architecture). The two outputs are parametric fused:

$$X = W_e \circ X_e^{(L)} + W_{target} \circ X_{target}^{(L)} \tag{1}$$

where $\circ$ is Hadamard product, W_e, W_{target} are trainable parameters. The fused output is then passed through two MLP layers to output mean μ and variance σ to use in KL Loss and sampling $\hat{X}_{enc}$

In the Testing phase, only the sequence of instant inputs is available to flow through the network to sample $\hat{X}_{enc}$

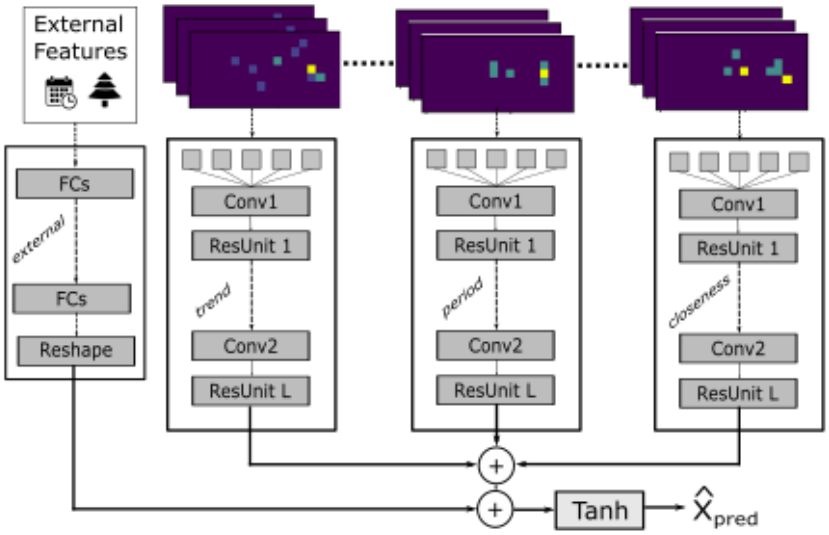
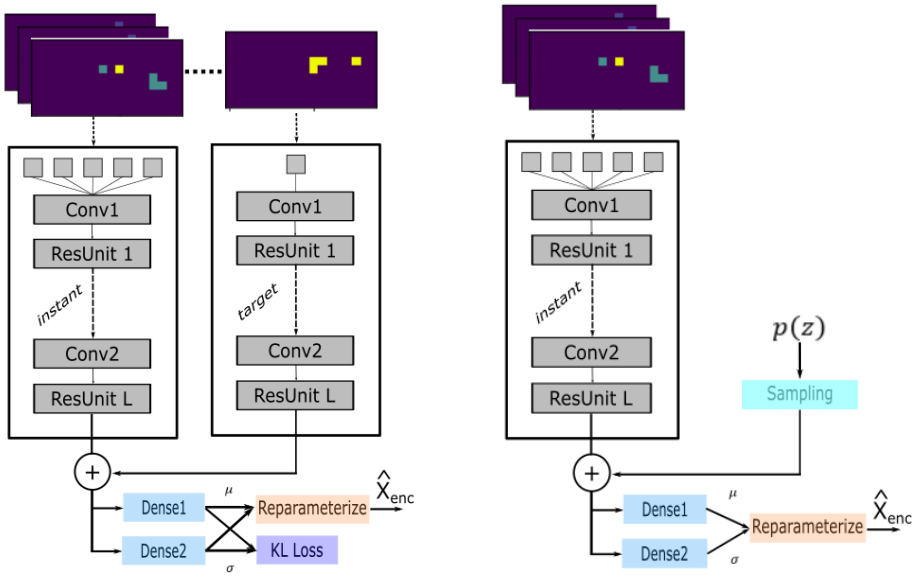


Figure 2: Predictor Block of Generative ST-ResNet



(a) Training Phase (b) Testing Phase

Figure 3: Encoder Block. (a) Training Phase when we incorporate the target to learn the uncertainty; (b) Testing phase when we sample a prior distribution

A.3 Structures of Decoder Block

The input to the Decoder block is $\hat{X}_{fuse}$, which is the addition of $\hat{X}_{pred}$ from the Predictor block and $\hat{X}_{enc}$ from the Encoder block through the Reshape module. Decoder is a series of transposed convolutional neural network layers. We do not use skip connection. The output of Decoder block is our final prediction.

Our Generative ST-ResNet can be trained to predict X_t from the sequences of previous inputs by joinlty minimizing two losses. The first loss is the mean squared error (reconstruction loss) between the predictied output and target output:

$$\mathcal{L}(\theta) = ||X_t - \hat{X}_t||_2^2 \tag{2}$$

where θ is all trainable parameters of all blocks.

The second loss is the KL Divergence loss between the output of Encoder and a prior distribution (we choose normal distribution $\mathcal{N}(0, 1)$:

$$\mathcal{L}(\phi, \theta, x) = D_{KL}(q_\phi(z|x)||p_\theta(z)) \tag{3}$$